DENIS ISIDORO DE FRANÇA
VICTOR GABRIEL MARACCINI

Sistema de sensoriamento para direção assistida de veículos

São Paulo
2016

DENIS ISIDORO DE FRANÇA
VICTOR GABRIEL MARACCINI

Sistema de sensoriamento para direção assistida de veículos

Trabalho de conclusão de curso apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do título de bacharel em Engenharia

São Paulo
2016

DENIS ISIDORO DE FRANÇA
VICTOR GABRIEL MARACCINI

Sistema de sensoriamento para direção assistida de veículos

Trabalho de conclusão de curso apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do título de bacharel em Engenharia

Área de Concentração: Engenharia elétrica com ênfase em sistemas eletrônicos

Orientador:
Prof. Dr. Leopoldo R. Yoshioka

Co-orientador:
Prof. Dr. Armando A. M. Laganá

São Paulo
2016

AGRADECIMENTOS

Ao Prof. Dr. Leopoldo R. Yoshioka e ao Prof. Dr. Armando A. M. Laganá, por suas orientações durante esse trabalho.

Aos integrantes do Grupo de Eletrônica Automotiva da Poli: Bruno Silva Pereira, Bruno Cesar Fernandes Pereira e Demerson Moscardoni, pelo trabalho realizado na ECU aberta do Polo Sedan 2004 e por toda ajuda prestada.

Aos colegas Pedro Augusto Ebert Gatti e Gabriela Letícia Melo de Souza pelo desenvolvimento da malha de controle utilizada neste projeto e pela ajuda nos testes realizados com o dinamômetro.

A todos os colegas, amigos e familiares que nos acompanharam durante nossa trajetória acadêmica na Escola Politécnica da USP.

RESUMO

O crescimento populacional e a popularização dos veículos particulares têm causado grandes problemas de mobilidade pública e aumento no número de mortes devido a acidentes de trânsito. A maioria desses acidentes são reportados como sendo falha humana. Este trabalho mostrou que é possível propiciar recursos de direção assistida a um veículo, mesmo trabalhando com restrições de baixo custo total. Foi implementado um sistema de controle de cruzeiro adaptativo e de detecção de velocidade limite na via. Além disso, um aplicativo foi desenvolvido para mostrar dados pertinentes em tempo real. Foram realizados testes unitários de cada componente do sistema, e o funcionamento geral foi validado através de testes *hardware-in-the-loop* com o auxílio de um dinamômetro, onde foi possível verificar o funcionamento das ferramentas desenvolvidas em conjunto. O módulo de detecção de velocidade limite na via e o de atuação no carro atingiram os requisitos determinados, enquanto o módulo de estimação de distâncias através de câmeras estereoscópicas ficou aquém do esperado, pois nesse trabalho não foi possível utilizar o radar automotivo inicialmente previsto. Devido à arquitetura escolhida para o sistema, os módulos utilizados são totalmente independentes entre si, tornando fácil a substituição de um sistema por outro de interface semelhante, permitindo melhorar a performance do medidor de distância em trabalhos futuros. Tal iniciativa demonstra que fabricantes poderiam adicionar funcionalidades semelhantes sem elevar consideravelmente o preço de seus veículos, fornecendo ferramentas que auxiliem o motorista.

Palavras-Chave – Controle de cruzeiro adaptativo. Reconhecimento de placas de trânsito. Visão estereoscópica.

ABSTRACT

The population growth and the popularization of non-commuting travel have caused problems in public mobility and an increase in the number of deaths due to traffic accidents. Most of these accidents are reported to be due to human failure. This project demonstrated that it is possible to provide advanced driver assistance systems (ADAS), even when working with low cost restrictions. An adaptive cruise control system and a speed limit recognition module were implemented. In addition, an application was designed to display relevant data in real time on a paired smartphone. Unit tests of each component of the system were performed and the system was validated using hardware-in-the-loop tests. The speed limit detection module and the car actuator reached the required performance, while the distance estimation module through stereoscopic cameras fell short of expectations, since it wasn't possible to use the automotive radar as initially planned. As a result of the chosen system architecture, all modules used are independent of each other, making it easy to replace one system with another with similar interface, allowing to improve the performance of the system in future works. This initiative demonstrates that manufacturers could add similar features without significantly increasing the price of their vehicles by providing tools that help the driver.

Keywords – Adaptive cruise control. Traffic sign recognition. Stereo vision.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Esta monografia trata do projeto de um sistema de sensoriamento para direção assistida de veículos, desenvolvido como projeto de formatura para o curso de engenharia elétrica com ênfase em sistemas eletrônicos, com o auxílio do Grupo de Eletrônica Automotiva da Escola Politécnica da Universidade de São Paulo.

O documento apresenta todas as etapas de desenvolvimento do projeto desde a concepção, implementação e aquisição de resultados. A primeira fase se iniciou com a delimitação das necessidades e objetivos do sistema proposto, passando então para o estabelecimento dos requisitos de engenharia necessários para atingir os objetivos propostos. Um *benchmark* competitivo foi desenvolvido para avaliar o posicionamento do sistema proposto frente às soluções comercialmente disponíveis. Finalmente, o sistema foi dividido em blocos independentes para torná-lo mais dinâmico e fácil de substituir, e uma prova de conceito foi desenvolvida através de simulações computacionais. A segunda etapa trata da implementação propriamente dita do sistema, envolvendo sua prototipagem e testes, e finalmente a aquisição dos resultados de cada bloco do sistema.

O capítulo *Pesquisa de levantamento e soluções atuais* contém uma revisão bibliográfica das tecnologias relevantes e soluções atualmente disponíveis.

O capítulo *Árvore de objetivos* apresenta a árvore de objetivos construída para o sistema, delimitando os conceitos escolhidos e seus pesos relativos para guiar o desenvolvimento do projeto. O capítulo *Especificações dos requisitos* contém os requisitos de marketing e de engenharia do projeto, bem como um *benchmark* competitivo das soluções encontradas.

O capítulo *Geração de Conceitos* trata da tabela de conceitos do projeto, relacionando as necessidades de engenharia e marketing com possíveis implementações.

O capítulo *Decomposição Funcional* desenvolve diagramas de nível 0, 1 e 2 do projeto, com detalhamento cada vez maior da integração entre os componentes presentes no projeto. O capítulo de *Gerenciamento do projeto* contém as tarefas que foram realizadas ao longo do projeto e seu cronograma efetivo, bem como uma análise de riscos relacionados ao projeto.

Em *Prova de conceito*, são desenvolvidas simulações para a validação da concepção inicial do projeto de se adotar uma arquitetura modularizada e desacoplada, apresentando os resultados

desejados no controle de uma planta simulada representando um veículo real.

No capítulo *Implementação*, são vistas as etapas de construção dos protótipos em diferentes estágios do projeto, culminando no protótipo final integrado e com todos os módulos designados na decomposição funcional devidamente implementados. O protótipo desenvolvido foi testado em diferentes condições e níveis de integração, e os resultados dos testes são apresentados no capítulo *Testes unitários e de integração*.

O capítulo de *Resultados* apresenta as capacidades do protótipo construído e sua validação em um ambiente *Hardware-in-the-loop*, mostrando seu funcionamento em um veículo real.

Finalmente, o capítulo *Trabalhos futuros* apresenta uma visão do que pode ser incrementado no sistema atual a fim de melhorar sua usabilidade e performance, e o capítulo *Conclusão* apresenta as considerações finais do projeto.

1.1 Declaração das necessidades

No âmbito de transportes, o crescimento populacional e a popularização dos veículos particulares têm causado grandes problemas de mobilidade pública e aumento no número de mortes devido a acidentes de trânsito. Nesse contexto, vê-se como alternativa inicial a implantação de sistemas de assistência ao motorista de baixo custo, a fim de se reduzir a mortalidade de trânsito e possibilitar a intercomunicação dos veículos em prol de maior fluidez de locomoção.

De acordo com a Organização Mundial de Saúde em um estudo conduzido em 2004, fatalidades no trânsito somam mais de 3 mil mortes diariamente, que equivalem a mais de 1 milhão de óbitos por ano em todo o mundo, das quais 90% são reportadas como sendo falha humana (RUPP; KING, 2010).

Uma primeira abordagem para a redução do número de acidentes é a introdução de elementos autônomos e sistemas inteligentes de assistência ao motorista que podem atuar em condições específicas, mas ainda mantendo o controle final sob supervisão humana.

Soluções autônomas vêm sendo implementadas na indústria automotiva por parte das grandes montadoras, porém pesquisas relacionadas no Brasil são escassas e como consequência há um grande atraso de aderência desta tecnologia no país, o que motiva o desenvolvimento de um sistema para o mercado nacional que atenda a expectativas da indústria local.

Como um primeiro passo no desenvolvimento de um sistema de assistência ao motorista, este trabalho visa implementar um sistema de controle de cruzeiro adaptativo aliado a um sistema de controle de velocidade baseado na velocidade máxima permitida na via.

1.2 Objetivos

A proposta para resolver os problemas citados é a criação de um sistema de sensoriamento capaz de prover os dados necessários para:

- Manter uma distância constante do veículo à frente;
- Manter velocidade compatível com a via;
- Interpretar placas de trânsito;
- Evitar colisões frontais.

A aplicação final do sistema constitui um controle de cruzeiro adaptativo (*Adaptive Cruise Control* - ACC, em inglês), que permite que o veículo controlado siga o veículo à sua frente mantendo uma distância de referência dependente da velocidade, respeitando os limites de velocidade da via. Para esse fim, o projeto consta com um sistema de detecção deste tipo de sinalização de trânsito por meio de capturas de imagem.

2 PESQUISA DE LEVANTAMENTO E SOLUÇÕES ATUAIS

A fim de verificar as diferentes alternativas de implementação para a aquisição de dados sobre veículos adjacentes e sobre o ambiente de trânsito, uma pesquisa sobre tecnologias relevantes foi conduzida de forma a verificar suas qualidades, possíveis restrições e escopo de aplicação. Além disso, foi também feita uma verificação da situação atual do mercado sobre sistemas de controle de cruzeiro adaptativo e tecnologias afins para possibilitar uma compreensão do cenário automotivo atual.

2.1 Tecnologias relevantes

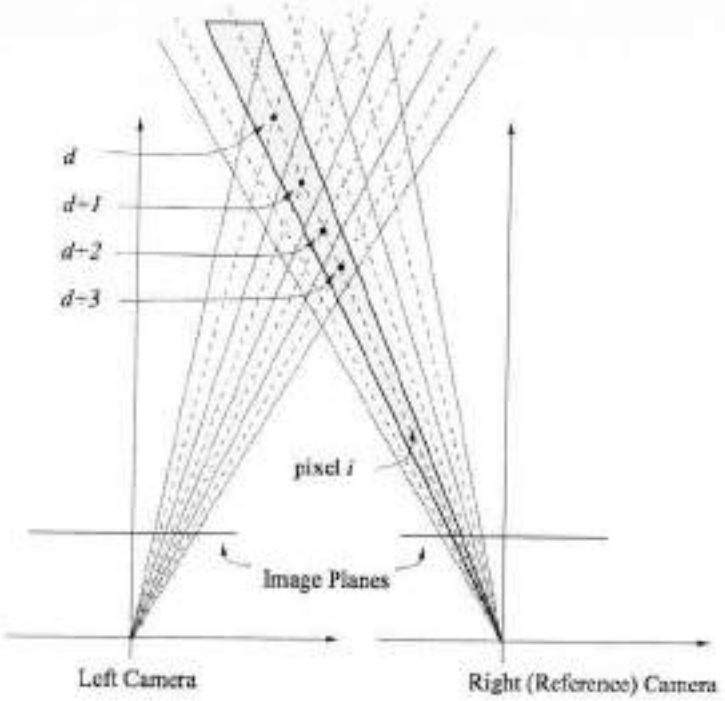
Para a implementação das funcionalidades necessárias ao sistema, diferentes tecnologias foram investigadas em relação à determinação de distâncias a objetos ao redor do veículo e também para a interpretação de placas de trânsito. Foram explorados sistemas de visão computacional, radares automotivos e LIDAR. A viabilidade de cada uma das alternativas será analisada mais a fundo no estudo bibliográfico e análise de requisitos de engenharia.

2.1.1 Estimação de distância por visão estéreo

Sistemas de estimação de distância por visão estéreo, ou estereoscopia, consistem em pares de câmeras arranjadas paralelamente de forma a detectar a diferença entre o posicionamento de objetos (fenômeno conhecido como paralaxe) (SON et al., 2013), e a partir desta diferença estimar a distância do objeto às câmeras.

Algoritmos tradicionais para estimação de distância procuram mapear *pixels* de uma das fontes de imagem a *pixels* da outra, e a partir deste mapeamento medir a disparidade em *pixels* dos pontos. Utilizando dados sobre a distância focal e resolução da câmera, é possível determinar a distância aproximada em metros do alvo, como ilustrado na Figura 1.

Figura 1: Princípio de funcionamento de um par de câmeras em visão estéreo.



Pares de *pixels* são encontrados através de análise das imagens vindas da câmera à esquerda e da câmera à direita, e a diferença entre suas posições é utilizada para determinar a distância do objeto assumindo uma distância focal constante. Fonte: Murray e Little (2000)

O algoritmo implementado por Trinh e McAllester ilustra esta técnica alternativa para estimação de distância. Nele é utilizada segmentação para identificar pontos de interesse em uma imagem e em seguida o mapa de disparidade entre a imagem da fonte esquerda e direita é calculado. Um algoritmo de identificação de características ao mesmo tempo encontra áreas em comum entre as duas imagens e descarta pontos de baixa correlação, permitindo atingir uma análise mais robusta.

Finalmente, os candidatos a pontos de referência para o cálculo das distâncias são selecionados e filtrados ao longo do tempo a fim de manter consistência temporal dos dados e evitar a aparição de objetos indesejados como deteções (*ghosting*). Os resultados obtidos apresentam grande resolução espacial e foram utilizados como referência para a criação de *mockups* para as simulações de sistemas de visão estéreo (TRINH; MCALLESTER, 2010).

2.1.2 Estimação de distância com uma câmera

Em contraste com sistemas de visão estéreo, sistemas monoculares de estimação de distância se baseiam em marcas visuais disponíveis em apenas uma fonte de imagem para determinar a posição relativa de objetos. Um algoritmo proposto por Saxena, Chung e Ng utiliza sistemas

de aprendizado de máquina supervisionado para resolver o problema. Primeiramente, o sistema é treinado para gerar mapas de distância utilizando como base apenas uma imagem e as distâncias reais (*ground truth*) levantadas por outros meios, como LIDAR por exemplo. Após a fase de treinamento, o sistema foi testado com imagens antes desconhecidas e obteve performance comparável a alternativas de visão estéreo (SAXENA; CHUNG; NG, 2005).

Outro algoritmo possível para esta tarefa foi proposto por Wedel et al., onde são criadas hipóteses para o configuração morfológica geral da cena e, sobre ela, são identificados candidatos a obstáculos que são posteriormente verificados através de análise de perspectiva (WEDEL et al., 2006).

2.1.3 LIDAR

Light Detection And Ranging (LIDAR) é uma tecnologia óptica de detecção remota que determina a distância a um objeto medindo a diferença de tempo entre a emissão de um pulso laser e a detecção do sinal refletido (DOYLE; GUNN, 2009).

O sistema é composto essencialmente por um laser, que emite luz ultravioleta, visível ou infravermelha, refletida por retrodifusão; um *scanner* e conjunto ótico, que afetam a velocidade de captura de imagens, a resolução angular e alcance; um fotodetector e eletrônica, que determinam a sensibilidade; e, para certas aplicações, sistemas de navegação e posicionamento, para determinar a orientação e posição absoluta do sensor.

Figura 2: Modelo 3D gerado a partir de dados do LIDAR da Velodyne

Fonte: Adaptado de Himmelsbach et al. (2008)

O LIDAR é capaz de medir distâncias a objetos de diferentes materiais, incluindo obje-

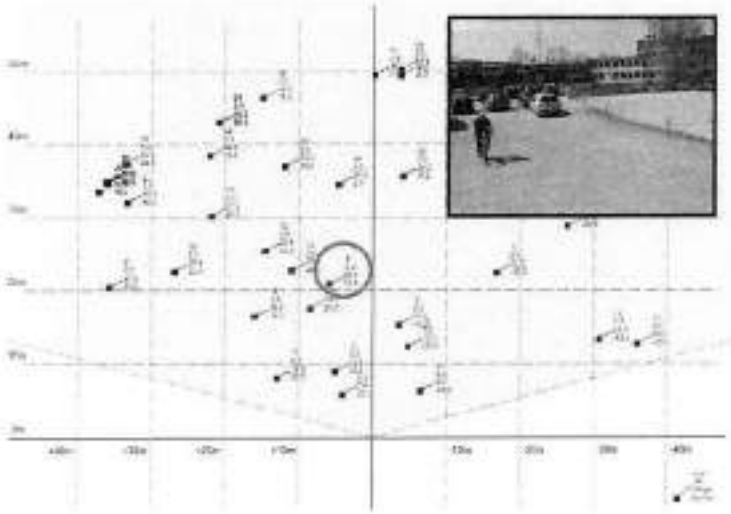
tos não-metálicos, rochas, compostos orgânicos, nuvens e até moléculas (CAMPBELL, 1996), encontrando assim diversas aplicações, como em geomorfologia, geodesia, física e análise atmosférica. Além disso, não é afetado por baixa luminosidade e apresenta alta resolução.

2.1.4 Radar automotivo

Radares para aplicações automotivas consistem em sistemas que utilizam ondas eletromagnéticas de comprimento de onda da ordem de milímetros para detectar objetos através de padrões de reflexão (ROHLING; MEINECKE, 2001).

Um mecanismo de detecção possível, conhecido como *Frequency-Modulated Continuous-Wave* (FMCW), envolve a geração de um sinal que varre uma certa faixa de frequências continuamente e mede a mesma faixa de frequências em busca de uma cópia atrasada do sinal produzido. O hardware interno do dispositivo é capaz de medir a diferença de frequência entre a onda refletida e a onda produzida, e através disso calcular o tempo de reflexão e portanto a distância do objeto à fonte emissora, como ilustra a Figura 4.

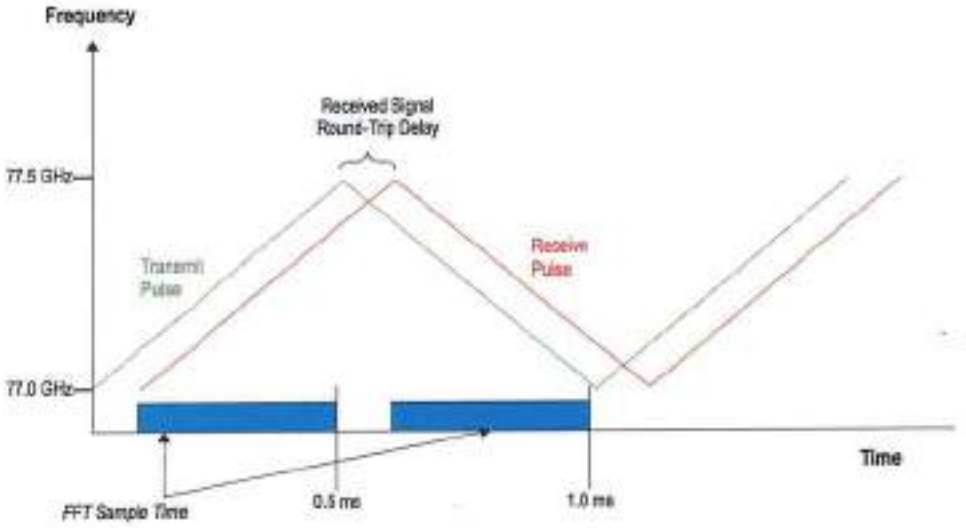
Figura 3: Visualização de dados coletados de um radar automotivo.



Fonte: Continental (2016)

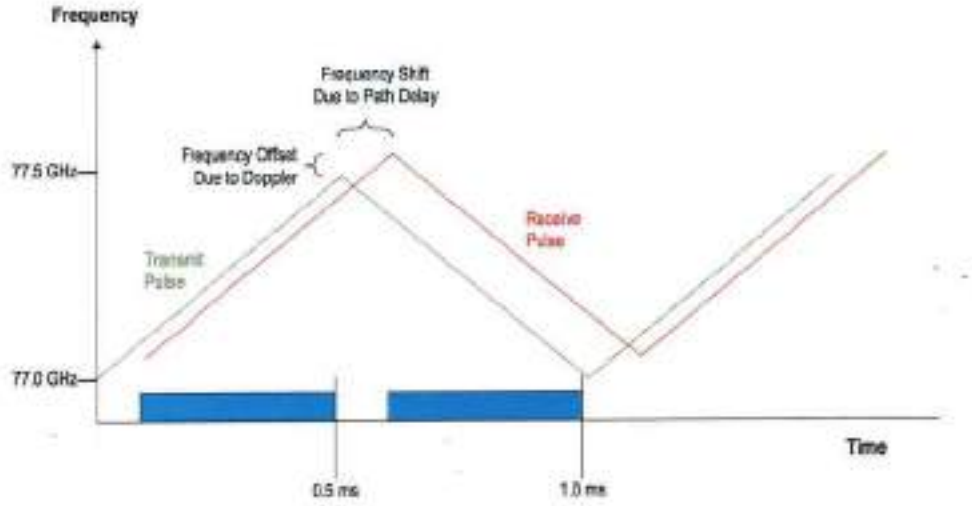
Aliado à detecção de objetos através de reflexões, sistemas de radar de ondas milimétricas também são capazes de determinar a velocidade relativa ao objeto em questão, através do fenômeno de desvio Doppler. Examinando a diferença de frequência entre o pico de frequência do sinal emitido e do sinal recebido é possível determinar através de análise espectral a velocidade de aproximação ou afastamento da fonte de reflexão, como pode ser visto na Figura 5.

Figura 4: Princípio de detecção de objetos utilizado em radares automotivos de ondas milimétricas.



Nota-se a emissão de um sinal com modulação em frequência em formato de onda *sawtooth* e sua respectiva reflexão causada pelo objeto sendo detectado. Fonte: Altera (2013)

Figura 5: Princípio de detecção de velocidade relativa de um radar automotivo.



Utiliza-se a diferença de frequência entre o pico de emissão da fonte e da onda refletida causada pelo efeito Doppler (*Frequency Offset due to Doppler*) para determinar a velocidade de aproximação ou afastamento da fonte de reflexão. Fonte: Altera (2013)

Tradicionalmente na indústria automotiva, radares de frequência de 77 e 79 *GHz* (YASUGI et al., 2013) têm sido amplamente utilizados para detecção de obstáculos e pedestres, disponíveis em soluções comerciais produzidas por diferentes empresas, como a *NXP Semiconductor* (NXP SEMICONDUCTORS, 2014), *Delphi Automotive LLP.* (DELPHI, 2016) e *Continental AG* (CONTINENTAL, 2014).

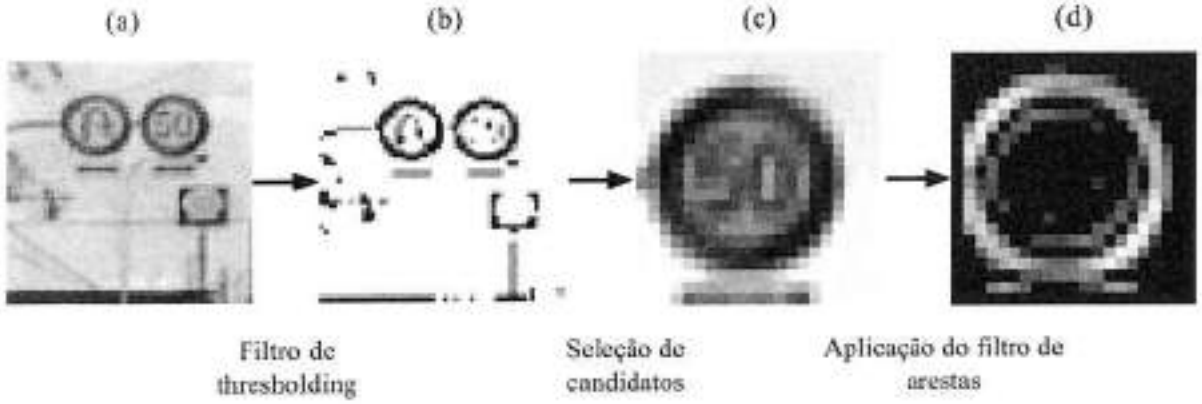
2.1.5 Sistemas de reconhecimento de imagem

Para realizar o reconhecimento de placas de trânsito, sistemas de visão computacional devem ser empregados e utilizados para realizar processamento em tempo real das imagens coletadas. Há várias alternativas de implementação de sistemas de reconhecimento de placas, utilizando algoritmos de pré-processamento para realçar detalhes relevantes de imagens como formas pre-determinadas ou contrastes de cor, ou até mesmo utilizando redes neurais para detecção (SERMANET; LECUN, 2011).

2.1.5.1 Sistemas tradicionais

Sistemas tradicionais de detecção de imagens se baseiam em buscas espaciais por cor, intensidade e arestas, processando as imagens adquiridas com filtros de binarização (*thresholding*) no espaço de intensidade e então utilizando filtros de detectores de arestas para encontrar formas pré-definidas nas imagens (MIURA; KANDA; SHIRAI, 2000). Por fim, o texto é extraído do interior do contorno obtido pela última etapa. Uma visualização das etapas aplicadas pode ser vista na Figura 6.

Figura 6: Visualização das etapas de detecção de imagem utilizando um sistema tradicional.



A partir da imagem original (a), foi extraída a versão binária (b) a partir da qual foram selecionados candidatos de tamanho e contraste apropriados (c), que finalmente passaram pelo filtro detector de arestas (d). De posse do contorno da placa de trânsito, o texto é extraído através de reconhecimento do texto interior ao contorno. Fonte: Adaptado de Miura, Kanda e Shirai (2000)

2.1.5.2 Sistemas baseados em redes neurais

Sistemas de aprendizado de máquina estão sendo cada vez mais utilizados em tarefas de detecção de padrões em imagens por dispensarem uma modelagem determinística do espaço de busca e utilizarem classificação (com ou sem supervisão) para determinar se uma amostra pertence ou não a um conjunto de dados.

No contexto de detecção de placas de trânsito, Sermanet e LeCun desenvolveram um sistema de reconhecimento de placas utilizando redes neurais convolucionais, utilizando filtros de *downsampling* e conversão de espaço de cores antes de utilizar os dados como entrada da rede neural. Os resultados obtidos foram extremamente acurados e competitivos em relação a outros sistemas baseados em métodos tradicionais de detecção, com taxas de acerto em torno de 98.97%, ainda mais quando comparados à taxa de detecção humana, que foi de 98.81% (SERMANET; LECUN, 2011).

2.2 Soluções atuais

Sistemas de controle de cruzeiro adaptativo já são implementados em alguns veículos, principalmente aqueles destinados ao mercado de luxo e com alto valor agregado. As principais montadoras européias e americanas, como Ford (FORD, 2016), BMW (BMW, 2016), Mercedes Benz (MERCEDES-BENZ, 2016), entre outras, possuem pelo menos um veículo com a funcionalidade em sua linha de produtos.

Normalmente, os sistemas implementados em veículos comerciais operam com base em radares automotivos e são ativados com ação ativa do usuário. Tradicionalmente, os sistemas incluídos em veículos de passeio não apresentam funções avançadas de localização e classificação de objetos, e se limitam apenas ao reconhecimento básico de obstáculos para a manutenção de distâncias pré-definidas.

Uma alternativa aos sistemas embarcados nos próprios veículos e que em tese poderia ser aplicada a qualquer carro é o sistema oferecido pela empresa MobilEye (STEIN; MANO; SHASHUA, 2003). Este sistema é totalmente baseado em visão computacional e pode realizar o controle de cruzeiro adaptativo apenas com uma única câmera de vídeo, utilizando métodos similares aos explorados como alternativas tecnológicas para sistemas de visão computacional monocular. O produto da empresa tem alcance de até 100m e precisão de 30% em suas medidas (STEIN; MANO; SHASHUA, 2003), já que infere a distância de veículos adjacentes através de análise de perspectiva e posição deste em relação à pista.

O produto final oferecido pela empresa, no entanto, não controla de fato o veículo por não

possuir integração alguma com o sistema eletrônico ou mecânico do mesmo. O sistema se limita a alertar o usuário de situações de perigo caso detecte algum objeto de aproximando a uma velocidade elevada e ao mesmo tempo verifique que o motorista não está respondendo à situação.

Vale notar que no mercado brasileiro as aplicações deste sistema ainda são escassas e por isso uma alternativa de baixo custo implementada pela indústria nacional teria grande impacto local para a redução de acidentes de trânsito.

Ainda, sabe-se que a eficácia de sistemas de controle de cruzeiro para melhorar a mobilidade urbana depende da porcentagem de veículos equipados com a tecnologia trafegando em uma mesma via, o que pode ser considerado mais um motivo para a utilização desta tecnologia em todos os tipos de veículos, não restringindo somente aos modelos de luxo.

3 ÁRVORE DE OBJETIVOS

Foram escolhidos os seguintes conceitos para o projeto:

1. Funcionais:
   (a) Interpretar placas de trânsito
      i. Detectar o contorno da placa
      ii. Detectar o texto interno
   (b) Detectar obstáculos
2. Generalidade:
   (a) Funcionar independentemente do veículo
   (b) Alimentado pela bateria do veículo
3. Precisão da detecção de obstáculos:
   (a) Utilizando-se um sensor especializado:
      i. Detectar obstáculos de 2 m de largura a 50 mm de distância
      ii. Possuir erro de distância menor que 2 m
   (b) Utilizando-se um sensor de baixo custo:
      i. Detectar obstáculos situados entre 1 m e 20 m

Uma estimativa inicial de valores razoáveis para distância mínima de detecção foi feita tomando-se um tempo de reação estimado de um sistema hipotético integrando o produto em desenvolvimento a um veículo real como sendo de 1.5 s (no pior caso), e assumindo uma velocidade de cruzeiro de 80 km/h, o que se traduz a uma distância mínima de detecção de aproximadamente 33 m. Assumindo uma margem de segurança de 50%, o valor preliminar para a restrição de distância mínima medida foi adotado como 50m.

Para estimar as restrições de precisão, foi adotada a necessidade de reconhecer um objeto de tamanhos compatíveis com um veículo (estimado em 2 m de largura) à distância de 50 m, adotada como distância mínima de medida.

Analogamente, o erro de distância deve ser compatível com o comprimento de um veículo, também estimado em 2 m.

Também foram considerados objetivos alternativos para um implementação de baixo custo, que não utiliza sensores dedicados para o meio automotivo. Nesse caso, devido às restrições de hardware, admite-se restrições mais brandas.

3.1 Atribuição de pesos relativos

Para definir a importância de cada conceito, foram utilizadas tabelas relacionando a importância relativa de cada conceito e em seguida normalizando os valores encontrados para obter os pesos utilizados na árvore de objetivos (Figura 7).

Definiu-se como prioritário o aspecto funcional do sistema e, em segundo lugar, a precisão esperada. A generalidade do sistema e sua aplicação em diferentes ambientes foram consideradas, em um primeiro momento, menos importantes, pois na fase inicial planejou-se a implementação de um protótipo. A atribuição de pesos pode ser vista na Tabela 1.

Tabela 1: Pesos dos objetivos fundamentais do projeto

Objetivo	Importância relativa	Peso
Funcional	3	0.50
Preciso	2	0.33
Genérico	1	0.17

Fonte: Autores

Dos aspectos funcionais, foi eleita como mais importante a capacidade de detectar obstáculos a uma distância mínima do veículo, e como fatores secundários a detecção e interpretação de placas de trânsito, como ilustra a Tabela 2.

Tabela 2: Pesos dos objetivos funcionais do projeto

Objetivo	Importância relativa	Peso
Interpretar placas de trânsito	1	0.25
Detectar obstáculos	3	0.75

Fonte: Autores

Dentro dos aspectos funcionais, a interpretação de placa de trânsito merece subdivisões entre a tarefa de detecção dos contornos das placas para determinar placas de trânsito válidas e a tarefa de interpretação de texto encontrado. Adotou-se que a importância maior é da etapa de interpretação do texto, como ilustra a Tabela 3.

Tabela 3: Pesos dos objetivos funcionais de interpretação de placas de trânsito

Objetivo	Importância relativa	Peso
Detectar o contorno da placa	1	0.33
Detectar o texto interno	2	0.67

Fonte: Autores

Em termos de precisão, julgou-se mais importante ter melhor acurácia e resolução espacial do que precisão em termos de distância medida, conforme indicado na Tabela 4.

Tabela 4: Pesos dos objetivos de precisão do projeto

Objetivo	Importância relativa	Peso
Detectar obstáculos de 2m de largura a 50m de distância	2	0.67
Possuir erro de distância menor que 2m	1	0.33

Fonte: Autores

Quanto à generalidade do sistema, ambos os conceitos foram considerados de igual importância para o produto, como visto na Tabela 5.

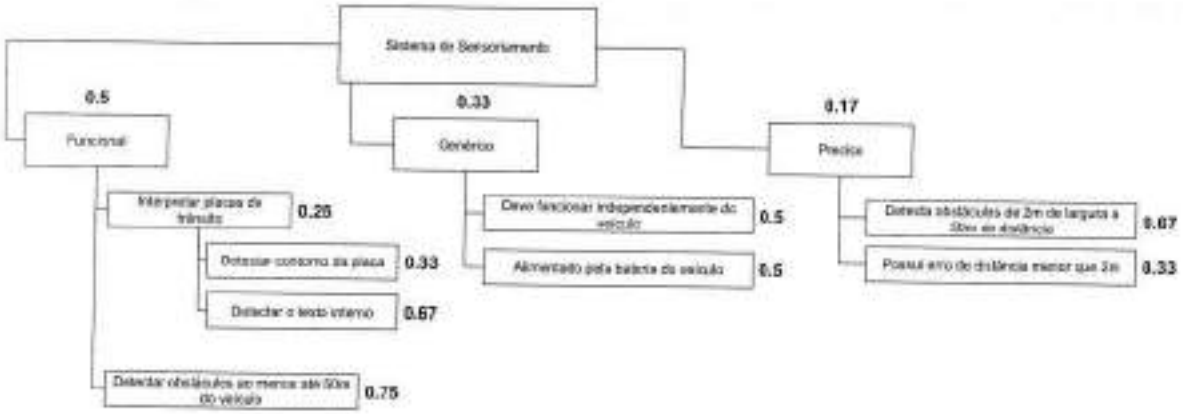
Tabela 5: Pesos dos objetivos de generalidade do sistema

Objetivo	Importância relativa	Peso
Funcionar independentemente do veículo	1	0.5
Alimentado pela bateria do veículo	1	0.5

Fonte: Autores

Finalmente, a Figura 7 apresenta a árvore de objetivos do projeto com pesos relativos atribuídos aos nós.

Figura 7: Árvore de objetivos do projeto com pesos relativos.



Fonte: Autores

4 ESPECIFICAÇÕES DOS REQUISITOS

Os requisitos de marketing do projeto determinam que o dispositivo deve:

1. Detectar placas de trânsito e interpretar a velocidade limite da via
2. Funcionar enquanto embarcado no veículo
3. Detectar obstáculos à frente do veículo
4. Conseguir distinguir múltiplos obstáculos
5. Conseguir estimar a velocidade de obstáculos

4.1 Benchmark competitivo

Muitos estudos envolvendo *cruise control* têm sido feitos através do uso de radares. Encontram-se exemplos cujo sensor utilizado opera em frequências entre 76 *GHz* e 77 *GHz*, capaz de detectar veículos até 200 *m* à frente e velocidades relativas de até 250 *km/h* (RAJAN KUMAN, 2011) (KUMAR; PATHAK, 2012).

Soluções comerciais que apresentam *Advanced Driver Assistance Systems* (ADAS) com *cruise control* geralmente empregam radares associados a outros sensores, como câmeras e sensores ultrasônicos, usados para redundância de dados e implementação de outras funcionalidades, como no Audi A8 (KUMAR; PATHAK, 2012) e no Tesla Model S.

Uma comparação entre implementações de *cruise control* com radar e com LIDAR foi feita por Widmann et al., com a conclusão de que não há diferenças significativas de resultados entre as duas tecnologias para este fim.

4.2 Requisitos de Engenharia

Os requisitos de engenharia foram levantados em função dos requisitos de marketing adotados para o produto, e referências para validação de cada um destes requisitos foram escolhidas de acordo com normas nacionais e internacionais, quando aplicável. O resultado deste levantamento encontra-se na Tabela 6.

Tabela 6: Requisitos de Engenharia

Requisitos		Validação
Marketing	Engenharia	
1	Utilizar câmera embarcada no veículo para detectar placas de trânsito	Observar norma ABNT NBR ISO 3864-1 sobre cores e sinais de segurança para identificação de placas de trânsito
1	Utilizar algoritmos de visão computacional para interpretar placas de trânsito em tempo real	Observar norma ABNT NBR ISO 3864-1 e o Manual de Sinalização Vertical de Regulamentação proposto pelo Conselho Nacional de Trânsito(Conselho Nacional de Trânsito, 2007) para a construção do sistema de interpretação
2	Utilizar sistemas que ocupam pouco espaço físico e são compatíveis com as condições de operação de um veículo em termos de temperatura, umidade, vibração etc.	Utilizar como referência a norma IEC 6068-2-27 Ea sobre choque mecânico mínimo suportado, e a norma IEC 60068-2-60 para resistência a água e poeira
3	Utilizar sensor(es) de distância para detecção de obstáculos	Observar padrão ISO 11898-1:2015 referente a padrões físicos de comunicação inter-veicular operando sob o protocolo CAN.
4	Utilizar dados de múltiplos sensores para detecção de múltiplos obstáculos ou utilizar sensor capaz de identificar múltiplos obstáculos de uma só vez	Observar se o resultado da fusão de múltiplos sensores e/ou utilização de um sensor de múltiplos obstáculos está dentro da faixa de precisão, resolução e alcance adequados ao projeto.
5	Utilizar sensor com capacidade de verificar a velocidade relativa de objetos à frente do veículo ou implementar um algoritmo capaz de fazer a mesma função.	Observar se a aferição de velocidade está dentro da faixa de precisão e escala adequadas ao projeto, e se o seu uso não causa problemas ao funcionamento do sistema como diminuição da velocidade de processamento.

Fonte: Autores

5 GERAÇÃO DE CONCEITOS

Para melhor analisar as alternativas de implementação deste projeto, foi criada uma tabela de conceitos (Tabela 7) que relaciona as necessidades do projeto e possíveis soluções encontradas.

Tabela 7: Tabela de conceitos do sistema

Medição de distância	**Unidade de processamento**	**Interpretação de placas de trânsito**
Lidar	Computador embarcado	Redes neurais
Radar	FPGA	Processamento tradicional
Visão estéreo		

Fonte: Autores

Dadas as restrições iniciais do projeto, uma análise completa das alternativas de implementação dos sensores pôde ser efetuada, enquanto em relação à unidade de processamento, foi possível apenas uma análise preliminar por não haver restrições bem definidas em termos de necessidades computacionais do sistema.

5.1 Medição de distância

Para avaliar a viabilidade de implementação das alternativas propostas para o sistema de medição de distância, uma análise de Pugh foi conduzida.

As métricas utilizadas para avaliar os sistemas de medição de distância foram: Distância máxima medida (Alcance), Precisão da medida, Resolução (Número de amostras por unidade angular) e Custo.

Os pesos relativos de cada medida foram convencionados seguindo a Tabela 8, atribuídos de forma coerente ao escopo do projeto. Vale notar que o parâmetro considerado de maior importância para a análise foi o alance do sistema, enquanto que o de menor importância foi a precisão da medida.

A análise de Pugh segue com a atribuição de conceitos a cada uma das alternativas de implementação com respeito às características adotadas na análise. Para tal determinação, va-

Tabela 8: Pesos relativos para análise de Pugh dos sensores de distância

	Precisão	Alcance	Resolução	Custo	Média	Peso
Precisão	1.00	0.25	1.00	0.50	0.595	0.13
Alcance	4.00	1.00	3.00	1.50	2.060	0.45
Resolução	1.00	0.33	1.00	0.50	0.639	0.14
Custo	2.00	0.67	2.00	1.00	1.278	0.28

Fonte: Autores

lores de referência de cada um dos critérios foram levantados para criar uma tabela auxiliar (Tabela 9) que permitiu calcular os conceitos da tabela de Pugh adotando-se o melhor sistema da tabela como referência (valor 1 na escala) e determinando os valores dos outros sistemas com uma escala linear.

Tabela 9: Tabela de valores de referência para análise de Pugh

Sistema	Erro (m)	Alcance (m)	Resolução @ 100m (m)	Custo (USD)
Lidar	0.03	100	0.1	8.000
Radar	0.25	200	1	3.300
Visão estéreo	1.68	45	5	500

Fonte: Autores

Em termos relativos, o conceito de precisão do sistema Lidar foi considerado o mais alto, estando em torno de $\pm 3\ cm$ (VELODYNE, 2016), seguido pelo Radar com aproximadamente $\pm 0.25\ m$ (CONTINENTAL, 2014) e finalmente da visão estéreo, com precisão estimada de apenas $\pm 1.68\ m$[1].

No quesito alcance, o Radar mostrou-se mais capaz do que todos os outros sistemas, com alcances da ordem de $200\ m$ (CONTINENTAL, 2014), seguido pelo Lidar com aproximadamente $100\ m$ (VELODYNE, 2016) e finalmente pelo par de câmeras, com apenas $45\ m$ (BERTOZZI; BROGGI, 1998).

Determinou-se também que o sistema de melhor resolução é o Lidar, com capacidade de medição na faixa de $0.5°$ em curto alcance e $1°$ em longo alcance, seguido pelo Radar, com capacidade de $1°$ fixo, e finalmente pela câmera, com resolução de medição dependente da resolução e do algoritmo de visão estéreo utilizado, mas sempre inferior às anteriores.

Finalmente, uma tabela de custos foi construída adotando-se os preços dos sensores Li-

[1] A estimativa da precisão do par de câmeras de visão estéreo foi feita com o auxílio de (KHOSHELHAM; ELBERINK, 2012), utilizando uma regressão linear dos dados de precisão obtidos por Khoshelham e Elberink numa faixa de 1 a 7 m para extrapolar a precisão na faixa de operação de $100\ m$.

dar e Radar mais competitivos para o escopo deste projeto, a saber, o Lidar *Velodyne VLP-16* (VELODYNE, 2016), o Radar *Continental* da linha *ARS 30x* (CONTINENTAL, 2014) e foi estimado um custo para duas câmeras de alta qualidade como sendo de USD 500,00. A lista completa de conceitos pode ser vista na Tabela 10.

Tabela 10: Análise de Pugh para sensores de distância

Sistema	**Precisão**	**Alcance**	**Resolução**	**Custo**	**Nota**
Lidar	1	0.5	1	0.06	0.51
Radar	0.12	1	0.1	0.15	0.52
Visão estéreo	0.02	0.23	0.02	1	0.39

Fonte: Autores

Após a análise de Pugh podemos concluir que os sensores mais adequados para implementação no projeto são o Lidar e o Radar, o que intuitivamente se justifica por estes possuírem alcance muito superior à implementação utilizando visão estéreo, e pelo fato de que o alcance é a métrica mais importante do sistema.

5.2 Unidade de processamento

Para a escolha da unidade de processamento, pôde ser feita apenas uma consideração inicial limitada já que à época não estavam definidos os sensores a ser utilizados no sistema final. Tendo isso em mente, uma análise de forças e fraquezas foi realizada para salientar as diferenças entre as implementações estudadas.

Tabela 11: Análise de forças e fraquezas das unidades de processamento propostas

Sistema	**Forças**	**Fraquezas**
Computador embarcado	Facilidade de implementação $+++$ Flexibilidade $+++$	Preço $--$ Consumo de energia $--$
FPGA	Velocidade de processamento $++$ Baixo consumo energético $++$	Recursos limitados $--$ Pouca flexibilidade $---$ Dificuldade de implementação $--$

Fonte: Autores

6 DECOMPOSIÇÃO FUNCIONAL

O sistema foi detalhado e decomposto em diagramas de nível 0 e nível 1, abordando seu funcionamento *macro* com suas entradas e saídas, e seus blocos internos, tendo uma visão primária da interligação de sensores, interfaces de protocolo e unidades de processamento.

6.1 Nível 0

A Figura 8 apresenta o diagrama de nível 0 do projeto inicial, detalhando as entradas e saídas do sistema e esboçando seu funcionamento principal. Como entradas, o sistema conta com informações vindas do veículo como velocidade e nível dos pedais, informações sobre o ambiente externo que serão captadas pelos sensores internos do sistema e uma fonte de energia para alimentação dos componentes internos.

Figura 8: Diagrama de nível 0 do projeto

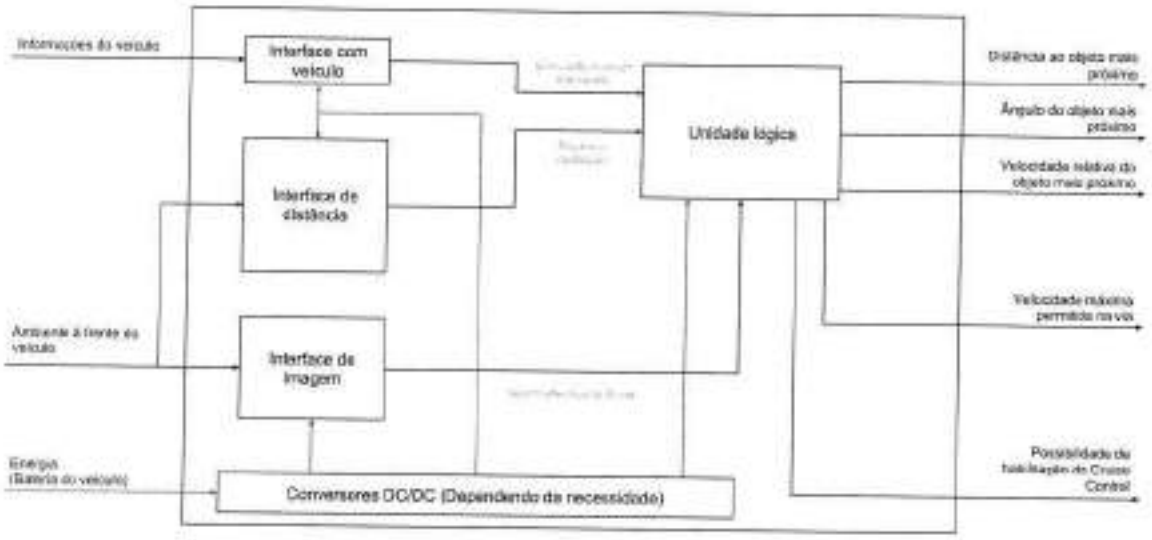
Fonte: Autores

O sistema tem como saídas os dados sobre o obstáculo mais próximo do veículo, aspecto mais relevante para o controle de cruzeiro adaptativo, a velocidade máxima permitida na via, e finalmente um sinal que determina se o sistema possui a capacidade de ser ativado nas circunstâncias atuais, necessário pelo fato de que o sistema só poderá atuar em determinadas faixas de velocidade.

6.2 Nível 1

A Figura 9 apresenta o diagrama de nível 1 do projeto inicial. Os componentes principais de sensoriamento do projeto fazem parte da malha de aquisição de dados, composta por uma interface visual, uma de distância e uma do veículo.

Figura 9: Diagrama de nível 1 do projeto inicial



Fonte: Autores

A interface de imagem é responsável pela visualização e interpretação de sinais de trânsito, tendo como saída a velocidade da via.

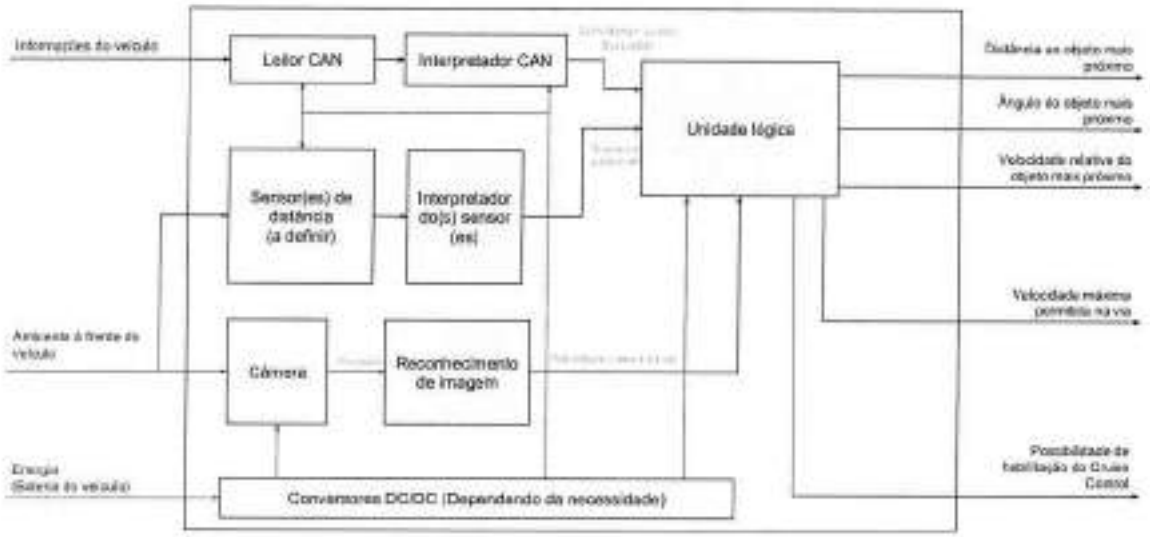
A interface de distância é incumbida da detecção de obstáculos, podendo ser baseada em tecnologia *LIDAR*, radar ou de visão computacional.

Finalmente, a unidade lógica é capaz de processar os dados vindos das interfaces sensoriais, tendo como saídas àquelas do sistema como um todo.

6.3 Nível 2

A Figura 10 apresenta o diagrama de nível 2 do projeto inicial. Sistemas existentes no diagrama de nível 1 foram decompostos em unidades mais específicas, demonstrando um melhor desacoplamento entre os blocos funcionais apresentados.

Figura 10: Diagrama de nível 2 do projeto inicial



Fonte: Autores

A saída do sensor de distância assume natureza diferente conforme a tecnologia empregada e o modelo do dispositivo. O Radar ARS308 da Continental, por exemplo, possui como saídas o número de objetos próximos ao sensor e o número de objetos distantes, entre outros (CONTINENTAL, 2014), enquanto o radar SRL1C, da mesma empresa expõe a distância do alvo em coordenadas polares (CONTINENTAL, 2012). A fim de tornar a unidade lógica agnóstica ao sensor utilizado, esses dados devem ser traduzidos e padronizados.

Para comunicação com o veículo, o *bus* CAN foi utilizado, já que este permite monitorar vários dados pertinentes ao sistema, como velocidade atual, rotação do motor, entre outras. No entanto, tais sinais seguem um protocolo físico e lógico normalmente incompatível com soluções comerciais de processamento, e portanto é necessário adicionar uma camada de acoplamento para interpretação dos dados vindos do sistema CAN e conversão para lógica compatível com o sistema de lógica escolhido.

6.4 Análise de acoplamento e coesão

O sistema não é altamente acoplado. Embora a unidade lógica dependa das interfaces sensoriais, estas não são interdependentes. A saída da interface de distância pode ser avaliada independentemente da aquisição de dados dos pedais do veículo, por exemplo.

O sistema é altamento coeso. As interfaces sensoriais podem ter implementações variadas, através do uso de tecnologias diferentes, de forma a continuar compatíveis com a unidade lógica utilizada. Além disso, cada módulo pode ser desenvolvido e testado independentemente. As interfaces sensoriais são desacopladas, como já dito. Em um nível inferior, o reconhecimento de imagem pode ser desenvolvido empregando-se imagens já gravadas, ao invés daquelas capturadas em tempo real, por exemplo. De maneira similar, o módulo de lógica pode ser testado utilizando-se entradas simuladas.

7 GERENCIAMENTO DO PROJETO

7.1 Definição de tarefas

O projeto foi dividido em quatro tarefas de primeira ordem, a saber:

- Escolha do(s) sensor(es) a serem utilizados
- Interação preliminar com sensor(es) escolhido(s)
- Desenvolvimento do software de interface e processamento dos dados
- Testes funcionais e comportamentais

As sub-tarefas relacionadas às principais podem ser vistas na Figura 11.

Figura 11: EAP do projeto

Fonte: Autores

7.2 Cronograma efetivo

Partindo de esboços de tarefas originalmente desenvolvidos durante as fases iniciais do projeto, encontra-se na Figura 12 o cronograma das atividades efetivamente executadas ao longo do projeto, em formato de Diagrama de Gantt.

As tarefas seguiram os cronogramas tentativos com grande fidelidade; o único contraste consistiu na postergação da tarefa relativa à aquisição de dados do veículo, frente à necessidade de trabalho na aquisição e tratamento dos dados dos sensores de distância e de imagem.

Figura 12: Diagrama de Gantt revisitado do projeto

	Nome da Tarefa	Início	Fim	Duração
1	Estudo bibliográfico	01/03/2016	13/05/2016	54d
2	Pesquisa de sensores	15/03/2016	10/05/2016	41d
3	Especificação do projeto	02/05/2016	15/06/2016	33d
4	Aquisição de sensores	11/05/2016	09/08/2016	65d
5	Prova de conceito (Milestone)	30/06/2016	30/06/2016	0d
6	Familiarização com os sensores	10/08/2016	08/09/2016	22d
7	Interpretação dos dados do sensor de distância	09/09/2016	22/09/2016	10d
8	Sistema de aquisição de dados do veículo	20/09/2016	14/11/2016	40d
9	Software de reconhecimento de imagem	18/07/2016	20/10/2016	69d
10	Software da unidade lógica	15/08/2016	31/10/2016	56d
11	Validação e testes	19/09/2016	25/11/2016	50d
12	Relatório final	19/09/2016	25/11/2016	50d

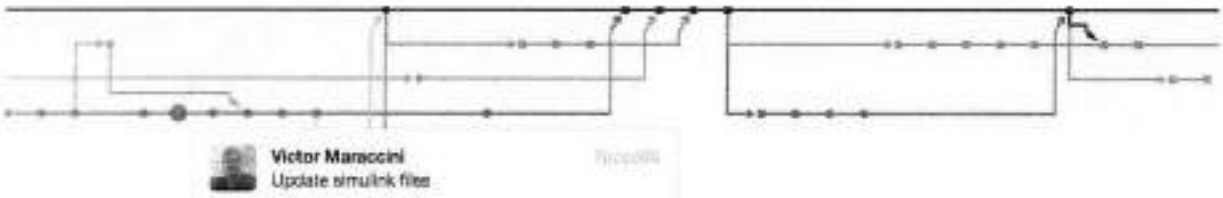
Fonte: Autores

7.3 Versionamento e revisão de código

Para o desenvolvimento dos softwares envolvidos no sistema, foi utilizado o *git* (SOFTWARE FREEDOM CONSERVANCY, 2016), um sistema de controle de versão distribuído e de gerenciamento de código fonte, com ênfase em velocidade. Esse tipo de sistema é muito presente em empresas e instituições de tecnologia e desenvolvimento de software, e também muito comum no desenvolvimento de software livre.

Essa ferramenta tornou possível o trabalho em paralelo, mesmo quando mais de um integrante do grupo atuou na edição de código de um mesmo módulo. O modelo de *branching* adotado consistiu no conjunto formado por uma branch *master*, sempre estável, *development*, com as *features* mais recentes e funcionais, e *branches* temporárias, com *bugfixes* e novos recursos. (DRIESSEN, 2010)

Figura 13: Ilustração do *branching* adotado



Fonte: Autores

Cada círculo na Figura 13 representa uma nova contribuição feitas ao projeto. Pode-se observar que as *branches* convergem para a *master*, simbolizada pela linha preta, quando estas ficam estáveis.

O repositório git foi hospedado na plataforma GitHub, que permitiu a revisão do código pelo outro integrante do grupo, antes de ser propagado para *branches* de níveis superiores, por meio de *pull requests*.

7.4 Riscos

Por se tratar de um sistema que pode oferecer risco à vida dos usuários e a terceiros, o gerenciamento dos riscos do *cruise control* deve ser parte fundamental do projeto. Como a Tabela 12 mostra, sob condições de mau funcionamento, o sistema pode implicar desde multas até acidentes de trânsito, dependendo da severidade.

Tabela 12: Matriz de riscos

	Impacto		
Probabilidade	**Médio**	**Alto**	**Muito alto**
Alta	Determinação incorreta da velocidade máxima da via		
Média	Erro nos atuadores	Estimativa incorreta da velocidade do carro à frente	
Baixa	Erro de regime		Falha na detecção do carro à frente

Fonte: Autores

A fim de minimizar os riscos associados, uma rotina intensiva de testes deve ser feita. A integridade de cada módulo deve ser garantida separadamente e o sistema como um todo avaliado por diversas simulações.

8 PROVA DE CONCEITO

8.1 Introdução

Durante o desenvolvimento do sistema, simulações computacionais foram conduzidas para verificar a viabilidade e eficácia do uso de diferentes classes de sensores para a realização do controle de cruzeiro adaptativo de um veículo de passeio. As simulações focaram-se em casos onde haveria potencial risco de colisão, como em frenagens do veículo de referência, ou quando fosse necessária intervenção direta do sistema de controle para a manutenção das condições desejadas de operação, como durante acelerações do veículo de referência.

A simulação da operação do sistema nessas condições permitiu verificar a viabilidade da aplicação das técnicas de sensoriamento e controle empregadas no projeto e ao mesmo tempo testar o sistema em condições típicas de operação. Tais verificações possibilitam o avanço seguro do desenvolvimento do produto, posto que corroboram com a teoria e técnicas empregadas até este ponto.

8.2 Ambiente de simulação

Para maior agilidade durante o processo de testes, os softwares Matlab e Simulink foram utilizados para as simulações. Com o uso destes softwares, modelos podem ser facilmente criados e interligados usando interfaces de blocos com baixa complexidade, porém grande versatilidade.

8.2.1 Arquitetura

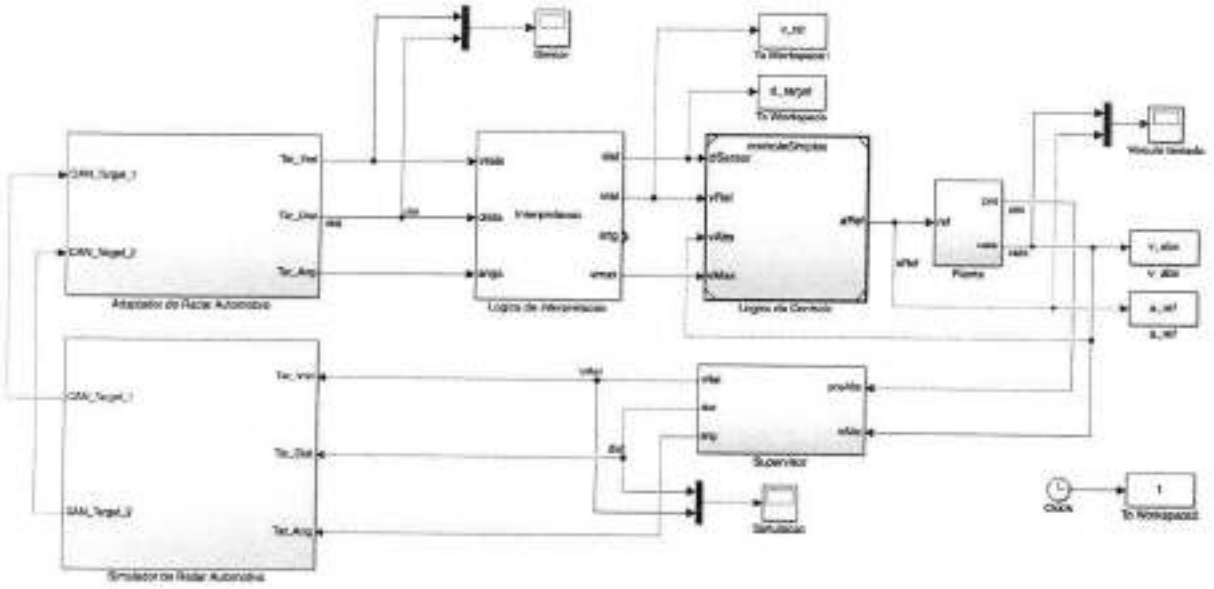
O sistema em si foi arquitetado de forma a ser altamente modular e reutilizável, de forma que diferentes sensores podem ser utilizados através da mera substituição de blocos de adaptação intermediários e novos algoritmos de controle podem ser implementados com igual facilidade.

Uma ilustração da malha de simulação pode ser vista na Figura 14, neste caso configurada para o consumo de dados vindos de um radar automotivo, particularmente o modelo ARS 30x da Continental (SON et al., 2013).

Para este tipo de simulação, o protocolo de comunicação do sensor utilizado como referência através do sistema CAN foi estudado e reproduzido dentro do ambiente de simulação de forma a gerar pacotes de informação equivalentes aos que seriam produzidor pelo sensor real.

A estrutura de dados utilizada para transferência de informação do sensor ARS 30x, disponibilizada pela empresa Continental (CONTINENTAL, 2015), foi implementada no sistema de simulação através de blocos do software Simulink e também em linguagem C para ser posteriormente utilizado com o sensor real em condições embarcadas.

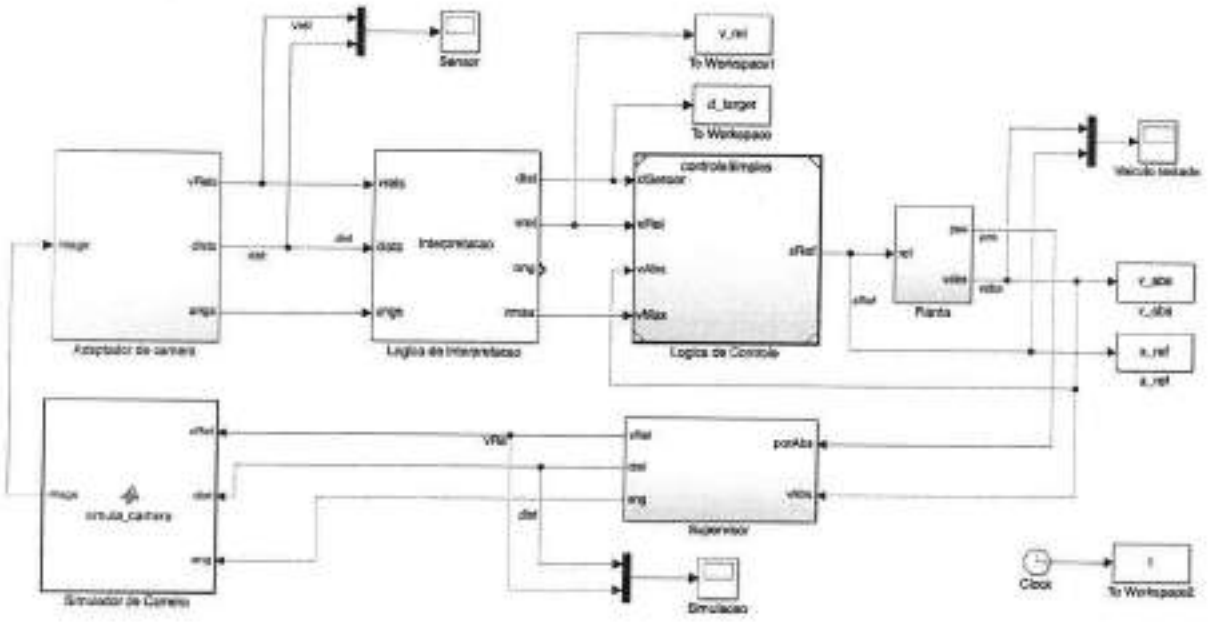
Figura 14: Visualização da malha de simulação configurada para receber dados de um radar automotivo



Fonte: Autores

A mesma malha pode ser facilmente configurada para receber dados de outros tipos de sensores, como por exemplo de um sistema de visão computacional com estimação de profundidade, como ilustrado na Figura 15.

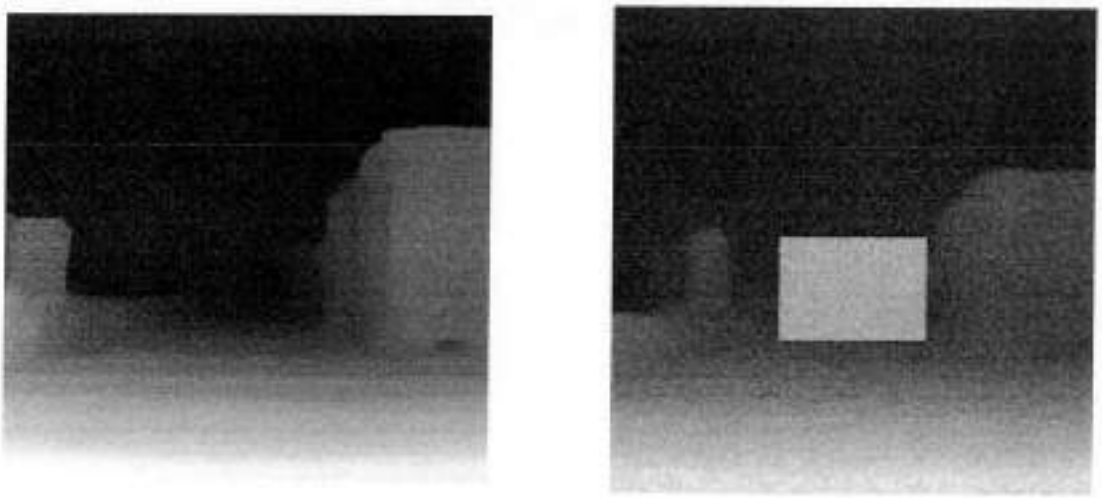
Figura 15: Visualização da malha de simulação configurada para receber dados de sistemas de visão computacional



Fonte: Autores

Para a simulação de um sistema de visão computacional, uma imagem modelo é gerada para cada *frame* de interação do sistema. Para cada *frame* se coloca um fundo vindo de um vídeo real de um sistema de estimação de distância através de visão estéreo (TRINH; MCALLESTER, 2010), acrescido de ruído branco, reproduzindo condições adversas de detecção. Além disso, é adicionada a representação de um veículo utilizando um simples retângulo com a informação de distância contida em seu tamanho e coloração, de forma semelhante a como o sistema detecta outros objetos, como ilustrado pela Figura 16. Ao fim da simulação, os *frames* são agregados para a formação de um vídeo onde é possível verificar a evolução da distância e velocidade relativa dos veículos

Figura 16: Visualização de um *frame* de simulação do sistema de visão computacional



À esquerda, a imagem original extraída de Trinh e McAllester (2010), e à direita a imagem acrescida de ruído para simular efeitos adversos de detecção e com o veículo simulado. Nota-se a presença de um retângulo representando o veículo de referência, cujo tamanho e coloração refletem a distância do mesmo em relação ao veículo controlado no ambiente de simulação em tempo real. Fonte: Adaptado de Trinh e McAllester (2010)

O ambiente de simulação se mostra também extensível: outros tipos de sensores e modelos de controle podem ser adicionados em paralelo para testes de novas e mais complexas funcionalidades, ainda mantendo sua modularidade.

8.2.2 Simulações realizadas

Alguns casos de simulação foram selecionados por terem sido considerados mais pertinentes na análise, descritos pela Tabela 13.

Tabela 13: Relação de situações simuladas para validação inicial

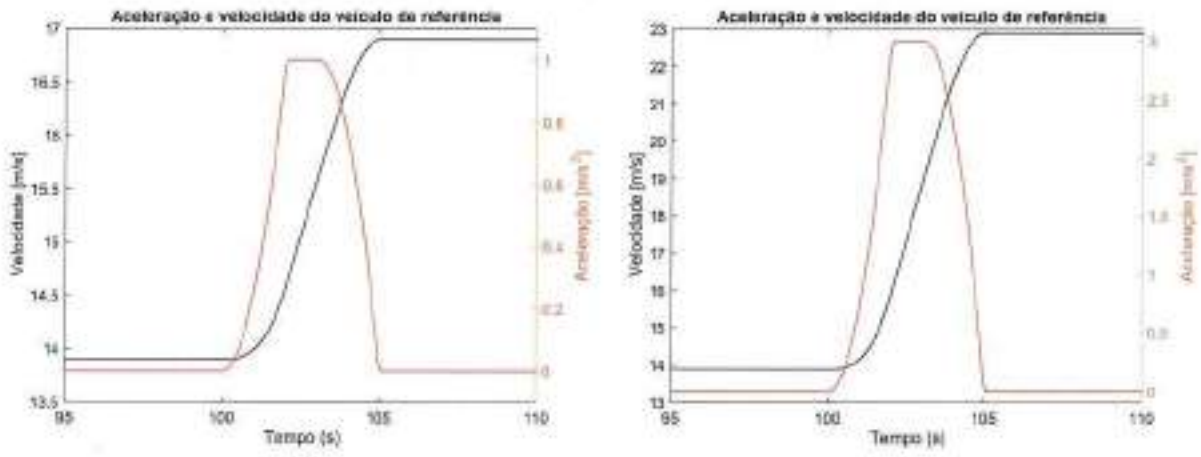
Caso	Descrição
1	Aceleração suave ($1m/s^2$) do veículo de referência
2	Aceleração abrupta ($3m/s^2$) do veículo de referência
3	Frenagem suave ($-1m/s^2$) do veículo de referência
4	Frenagem abrupta ($-3m/s^2$) do veículo de referência
5	Situação real de frenagem suave
6	Situação real de aceleração suave

Fonte: Autores

8.2.3 Estímulos

Para as curvas geradas sinteticamente (casos 1 a 4), foi utilizada uma função contínua com subida e descida aproximadas por uma função quadrática para simular a aceleração do veículo de referência, já que seu valor não pode sofrer descontinuidades.

Figura 17: Curvas de velocidade e aceleração do veículo de referência para aceleração.



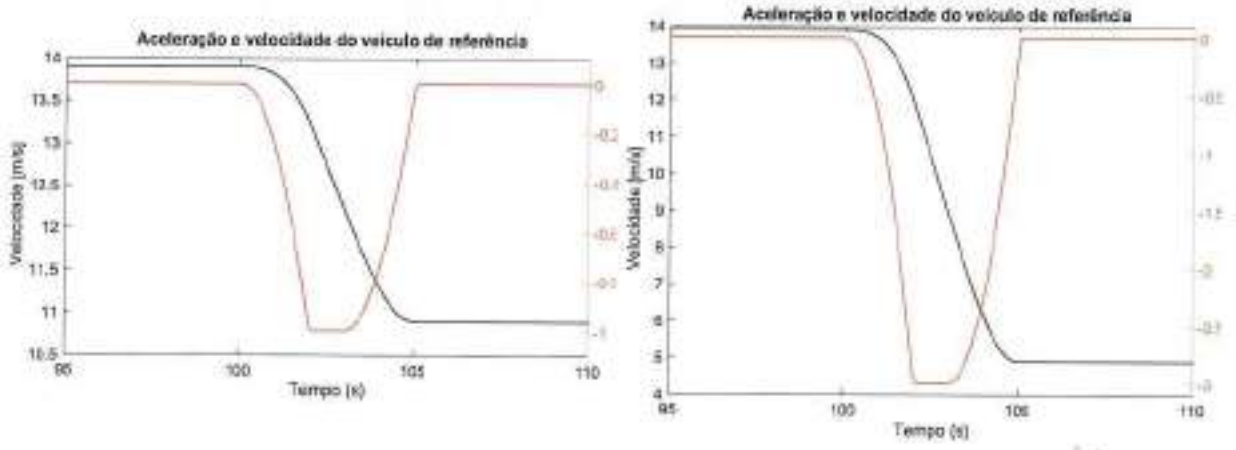
Fonte: Autores

As curvas reais de aceleração e frenagem foram capturadas pelos autores utilizando um *scanner* OBD-II conectado ao veículo de testes, como ilustra a Figura 18. O sensor é capaz de ler mensagens vindas do *bus* e interpretá-las para identificar os sinais de interesse. Para as simulações, a velocidade em km/h foi utilizada para inserir o veículo no ambiente de simulação.

Figura 18: Exemplo de *scanner* OBD para monitoramento da interface CAN

Fonte: Elm Electronics

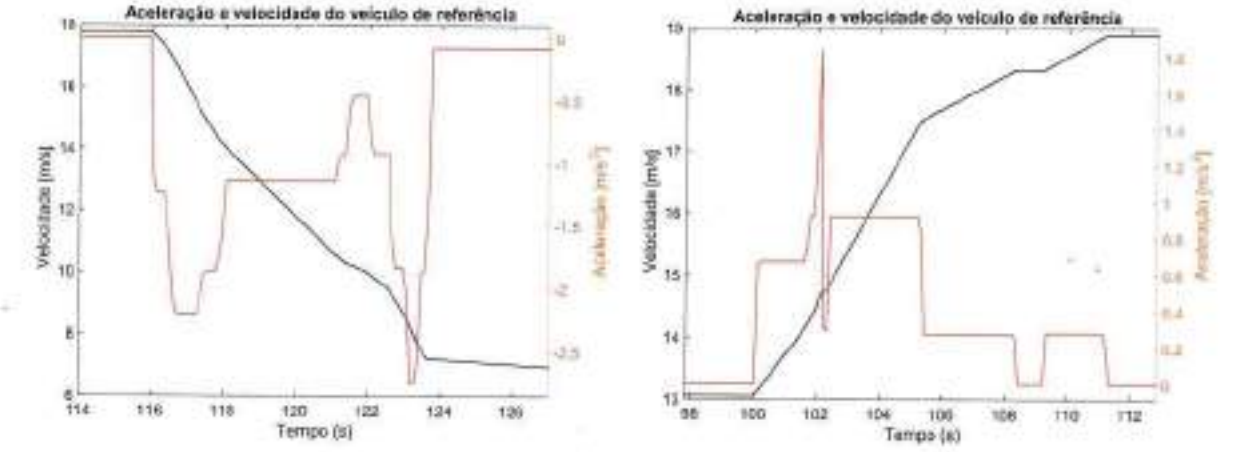
Figura 19: Curvas de velocidade e aceleração do veículo de referência para frenagem.



Fonte: Autores

Após inseridas no ambiente de simulação, as curvas de velocidade foram derivadas a fim de se obter a aceleração do veículo real, dado este que não está disponível via CAN. Os estímulos gerados e utilizados para as simulações 5 e 6 podem ser vistos na Figura 20.

Figura 20: Curvas de velocidade e aceleração do veículo de referência um uma situação real de frenagem e aceleração suaves.



Fonte: Autores

8.3 Modelo matemático

Para a simulação do veículo controlado, foram coletados dados experimentais de um veículo real e em seguida foi criado um sistema de primeira ordem refletindo a resposta obtida. O experimento buscou encontrar a resposta da velocidade do veículo sem carga à posição do pedal

de aceleração do veículo, donde se obteve o sistema de primeira ordem descrito pela Equação 8.1:

$$G(s) = e^{-T_{delay} \cdot s} \cdot \frac{K_n}{T_s \cdot s + 1} \tag{8.1}$$

Onde os parâmetros obtidos do sistema estão descritos na Tabela 14

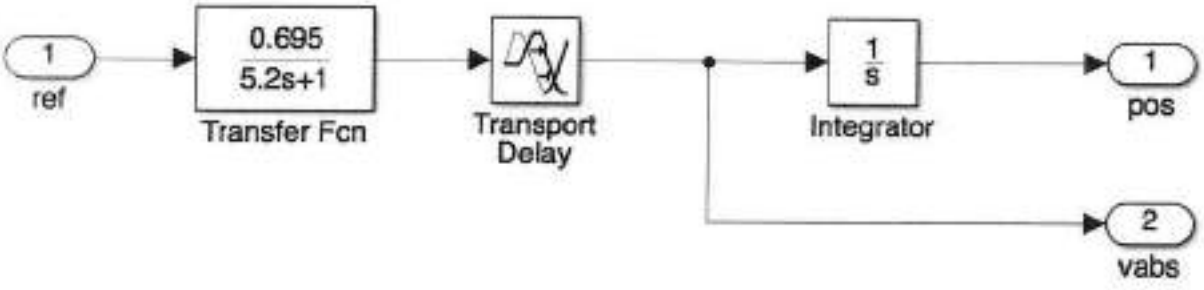
Tabela 14: Parâmetros identificados do sistema de primeira ordem do veículo de testes

Parâmetro	Valor
T_{delay}	0.5
K_n	0.6944
T_s	5.2

Fonte: Autores

O sistema foi realizado utilizando um atraso de transporte e uma função de transferência de primeira ordem, como ilustrado pela Figura 21.

Figura 21: Realização da planta para simulação do veículo de testes

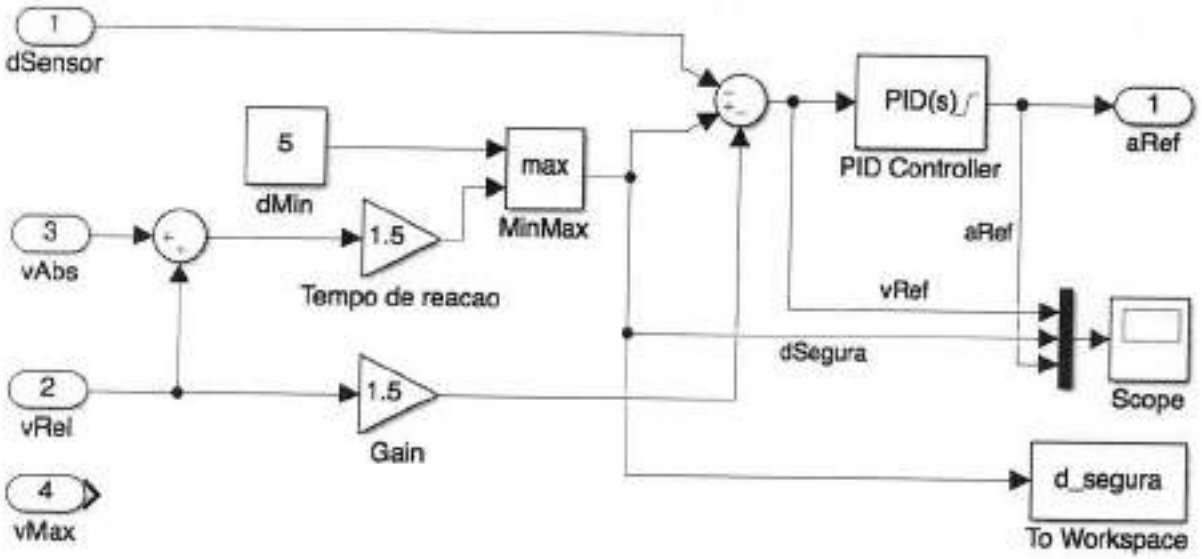


Fonte: Autores

O sistema de controle foi implementado de forma simplificada nesta etapa do projeto, apenas para a validação do funcionamento do sistema como um todo. Foi utilizada uma malha de controle de posição com realimentação negativa de velocidade conforme estudado na disciplina de Laboratório de Controle da Escola Politécnica [2], como ilustra a Figura 22.

[2] Agradecimentos a Bruno Silva Pereira, Bruno Cesar Fernandes Pereira, Demerson Moscardoni e Prof. Dr. Armando Antonio Maria Laganá, do Grupo de Eletrônica Automotiva do Departamento de Engenharia de Sistemas Eletrônicos da Poli-USP, por suas contribuições durante os testes experimentais para o levantamento do perfil do sistema.

Figura 22: Malha de controle simplificada implementada no projeto



Fonte: Autores

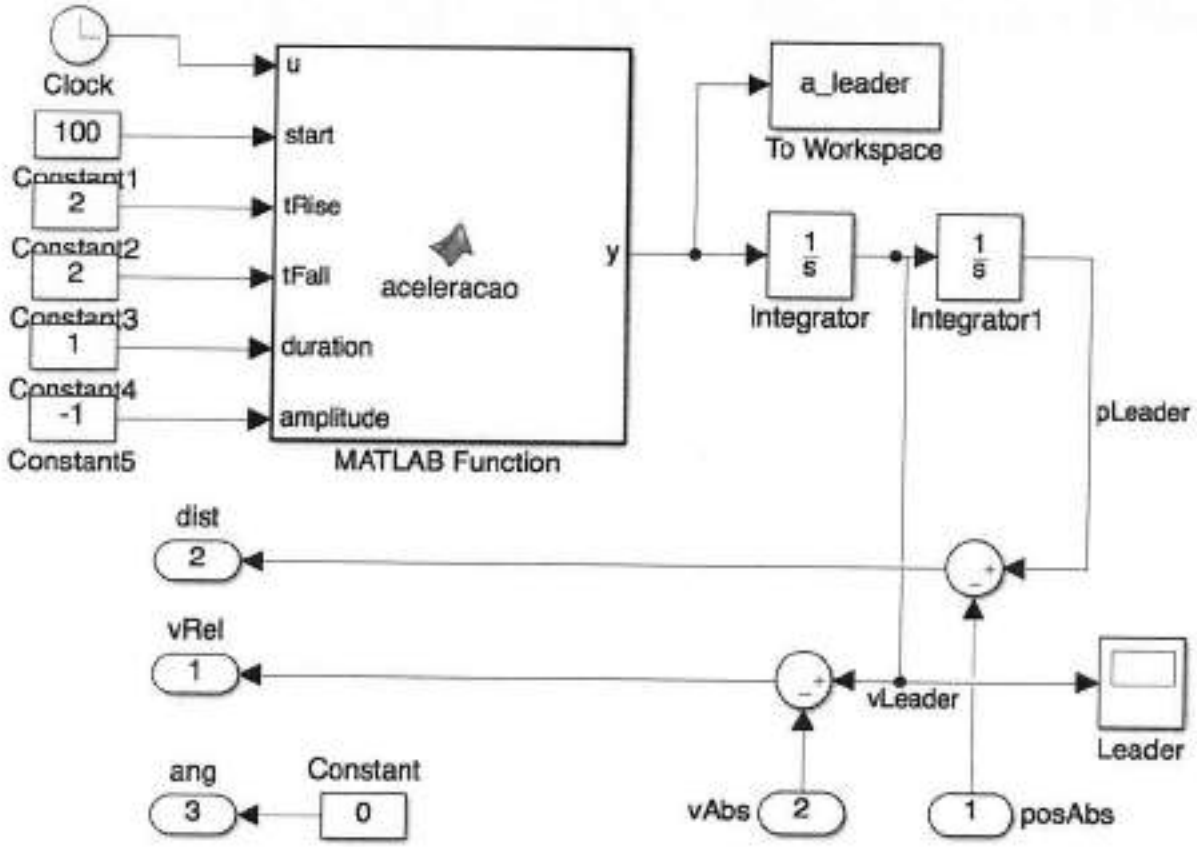
Admitiu-se também, de forma simplificada, que a distância segura para trafegar a uma determinada velocidade segue a relação apresentada pela Equação 8.2.

$$d_{segura} = t_{reação} \cdot v_{abs} \tag{8.2}$$

O tempo de reação admitido foi de $1.5s$, seguindo estimativas dos próprios autores.

Finalmente, o sistema supervisor de simulação é responsável pela manutenção do estado atual e processar a posição do veículo de referência no instante seguinte da simulação, seguindo um comportamento pré-determinado através de parâmetros fornecidos a um bloco que gera o sinal de aceleração do veículo, como ilustra a Figura 23.

Figura 23: Implementação do sistema de supervisão de simulação, configurado para uma frenagem suave



Fonte: Autores

8.4 Resultados

O sistema foi testado recebendo os estímulos desenvolvidos nas duas configurações desenvolvidas e ilustradas pelas figuras 14 e 15, que simulam a interação com um radar automotivo e com um sistema de visão computacional, respectivamente.

Para visualizar os resultados das simulações, foi desenvolvido um *script* em Matlab para imprimir o resultado consolidado das simulações juntamente com uma representação gráfica dos veículos envolvidos no processo, como ilustrado pela Figura 24.

Figura 24: Ilustração do visualizador de simulações desenvolvido para observar os resultados da prova de conceito

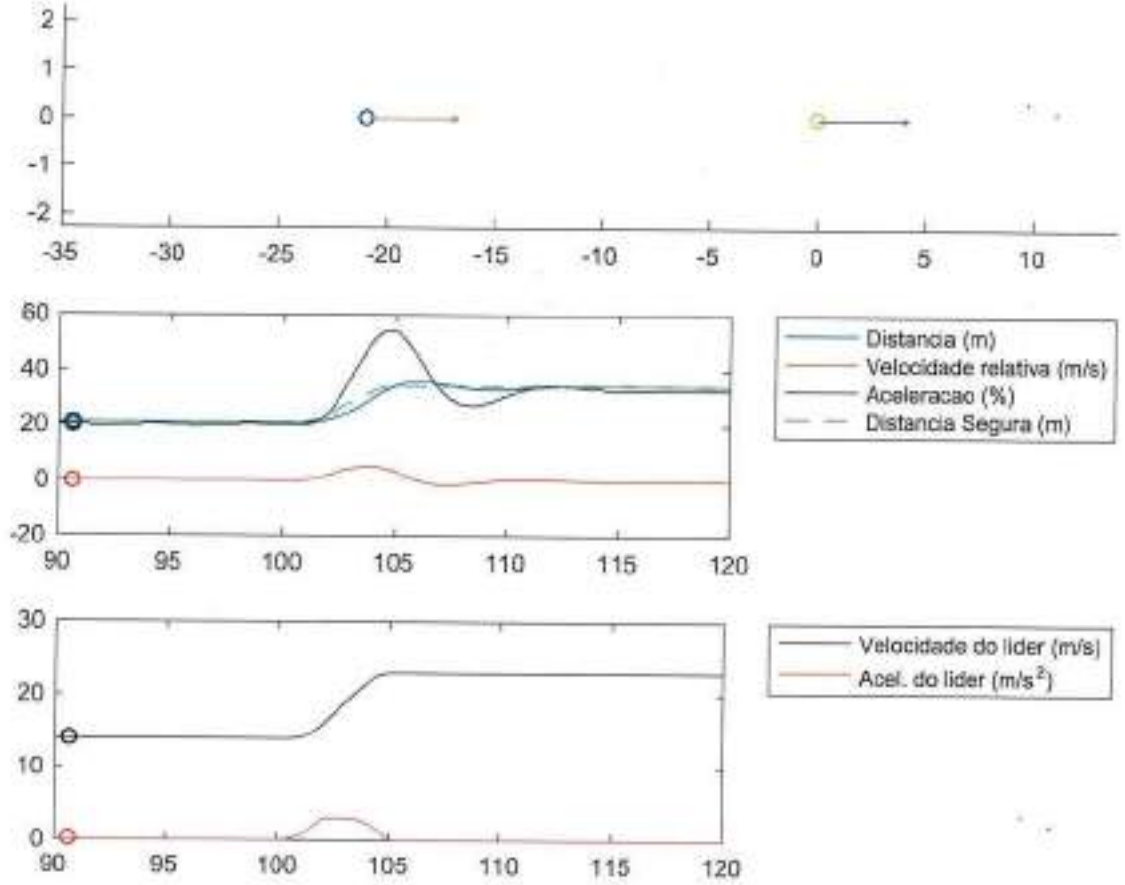
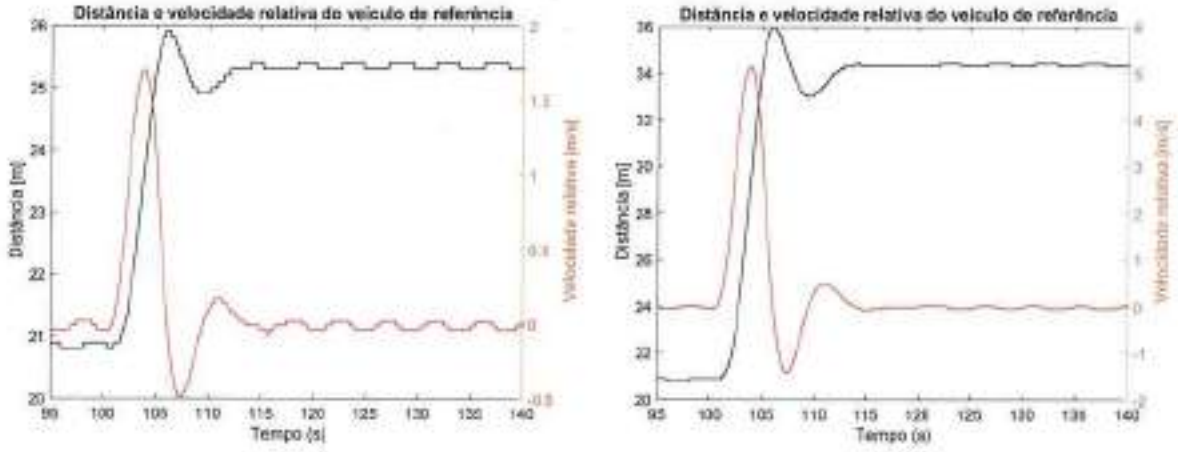


Ilustração do visualizador de simulações desenvolvido para observar os resultados da prova de conceito. Um vídeo da visualização de uma simulação pode ser visto no link: https://goo.gl/KRoIb3. Fonte: Autores

8.4.1 Simulação de radar automotivo

As simulações sintéticas revelam grande capacidade de adaptação por parte do sistema em resposta a diferentes manobras do veículo de referência. As variáveis de saída monitoradas para a simulação foram a distância e a velocidade relativa do veículo de referência em relação ao veículo controlado. A Figura 25 representa o resultado da simulação para as situações de aceleração sintéticas (casos 1 e 2).

Figura 25: Resultados da simulação de aceleração suave e abrupta utilizando a malha de simulação para radar

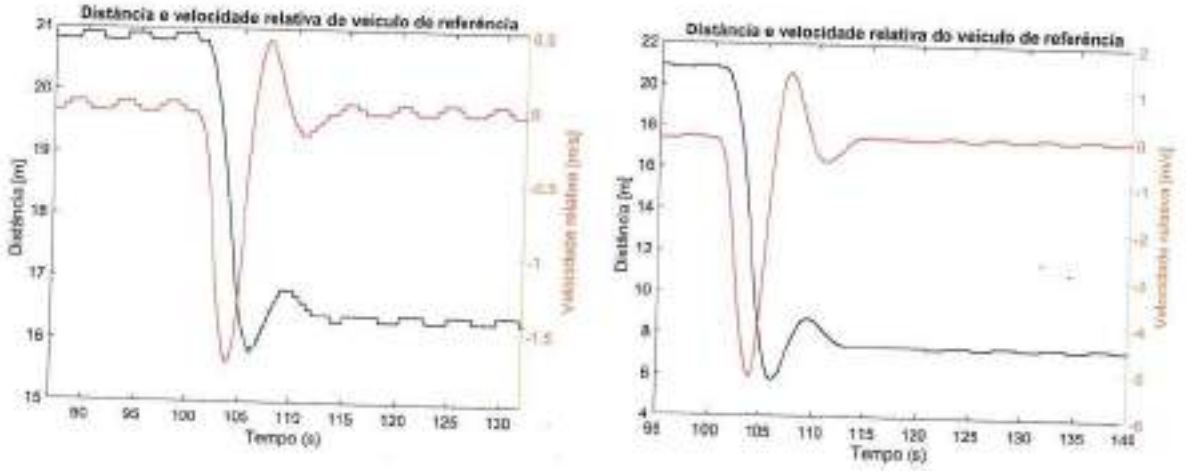


Fonte: Autores

Nota-se que a distância relativa entre os veículos permanece sempre positiva, isto é, não há colisões entre o veículo controlado e o veículo de referência. Durante o período transitório onde a velocidade do veículo controlado está sendo ajustada há uma discrepância entre a distância instantânea e a distância segura, mas uma vez atingido o regime a distância permanece constante e igual à de segura.

Foram observadas oscilações na distância e velocidade relativas remanescentes após a atuação do sistema, mas acredita-se que este efeito é devido a pequenos problemas de ajuste de constantes do sistema de controle, e podem ser facilmente resolvidos. O mesmo pode ser observado para as situações de frenagem simuladas (casos 3 e 4), como ilustra a Figura 26.

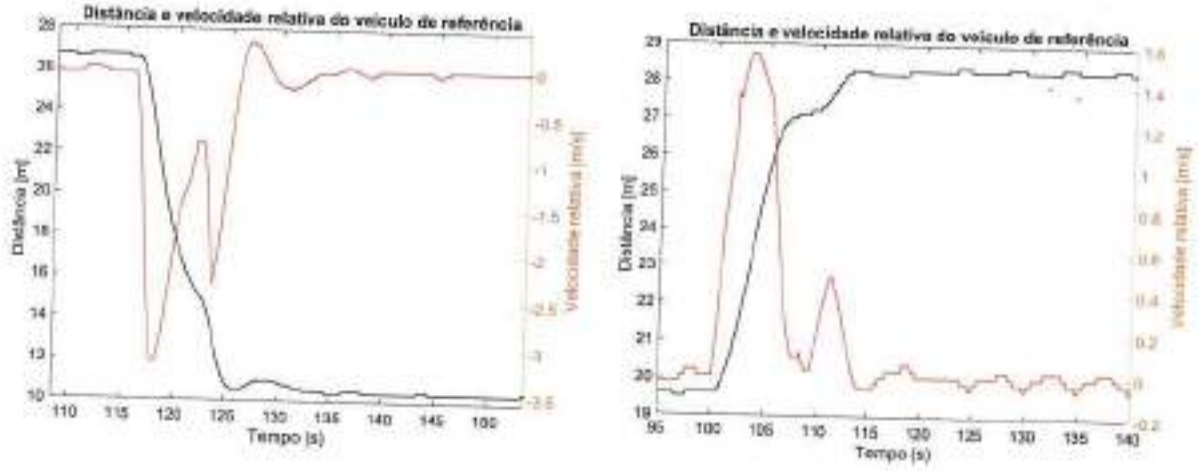
Figura 26: Resultados da simulação de frenagem suave e abrupta utilizando a malha de simulação para radar



Fonte: Autores

Finalmente, as simulações utilizando dados reais revelam comportamento similar ao já observado, sem colisões e com a manutenção da distância segura e regime, porém com pequenas oscilações, como pode ser visto na Figura 27.

Figura 27: Resultados da simulação de frenagem e aceleração suaves com dados reais utilizando a malha de simulação para radar



Fonte: Autores

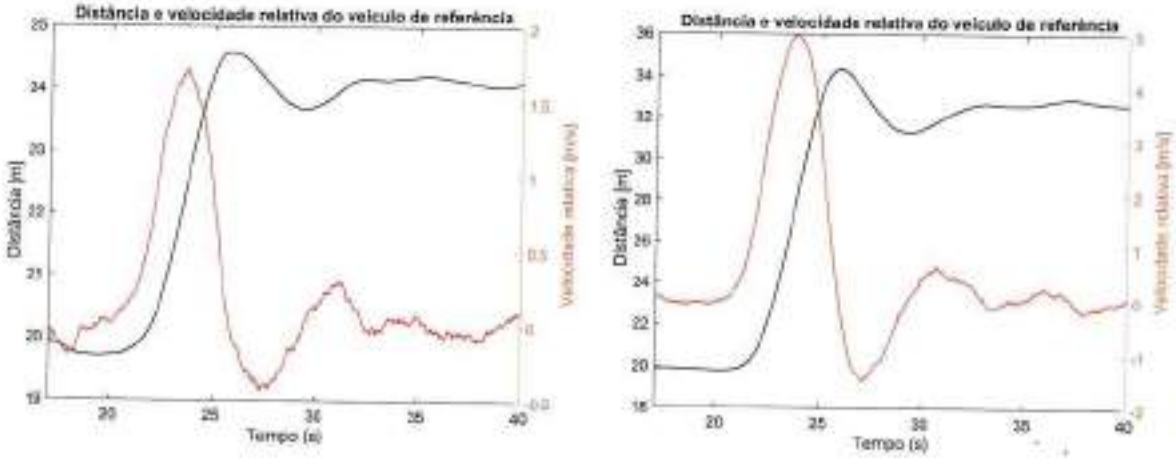
8.4.2 Simulação de sistema de visão computacional

Os resultados obtidos através de simulações utilizando a malha configurada para visão computacional foram similares aos obtidos para o radar automotivo, mostrando tempos de resposta semelhantes e sendo igualmente eficazes na prevenção de colisões e manutenção de distâncias

de segurança em todas as situações.

A Figura 28 traz os resultados das simulações de aceleração sintéticas (casos 1 e 2), a Figura 29, os resultados para frenagens sintéticas (casos 3 e 4) e a Figura 30, os resultados das simulações com dados reais (casos 5 e 6).

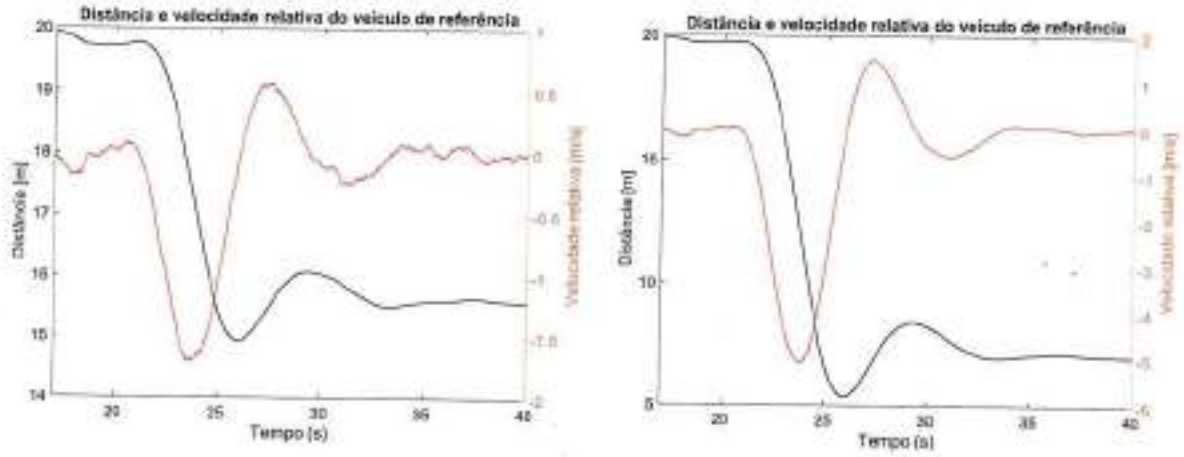
Figura 28: Resultados da simulação de aceleração suave e abrupta utilizando a malha de simulação para visão computacional



Da esquerda para a direita, aceleração suave (caso 1) e abrupta (caso 2). Fonte: Autores

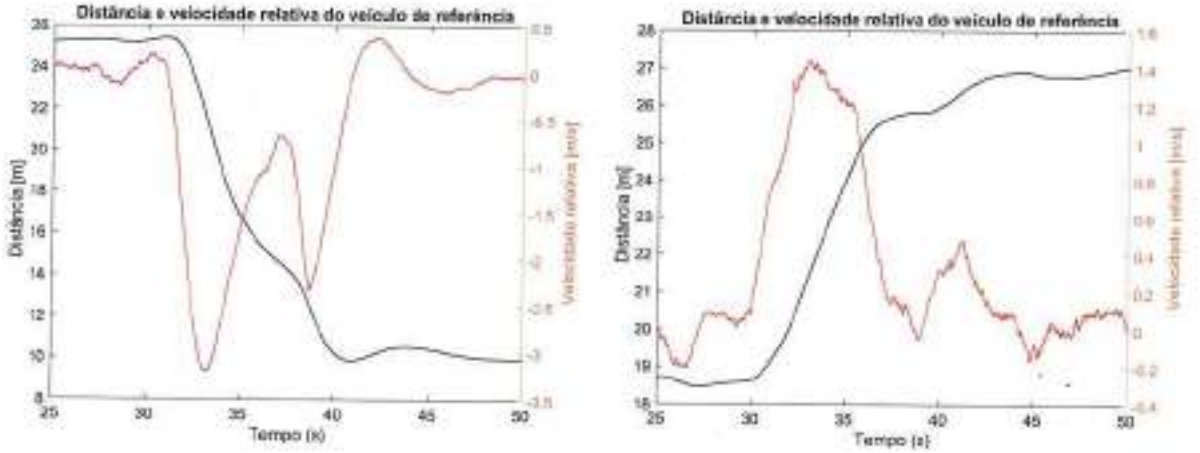
Notam-se também nas simulações de visão computacional a presença de oscilações remanescentes após a atuação do sistema, o que reforça a hipótese de que estas são causadas não pela malha de sensoriamento mas sim pela malha de controle, que ainda precisa de refinamento.

Figura 29: Resultados da simulação de frenagem suave e abrupta utilizando a malha de simulação para visão computacional



Da esquerda para a direita, frenagem suave (caso 3) e abrupta (caso 4). Fonte: Autores

Figura 30: Resultados da simulação de frenagem e aceleração suaves com dados reais utilizando a malha de simulação para visão computacional.



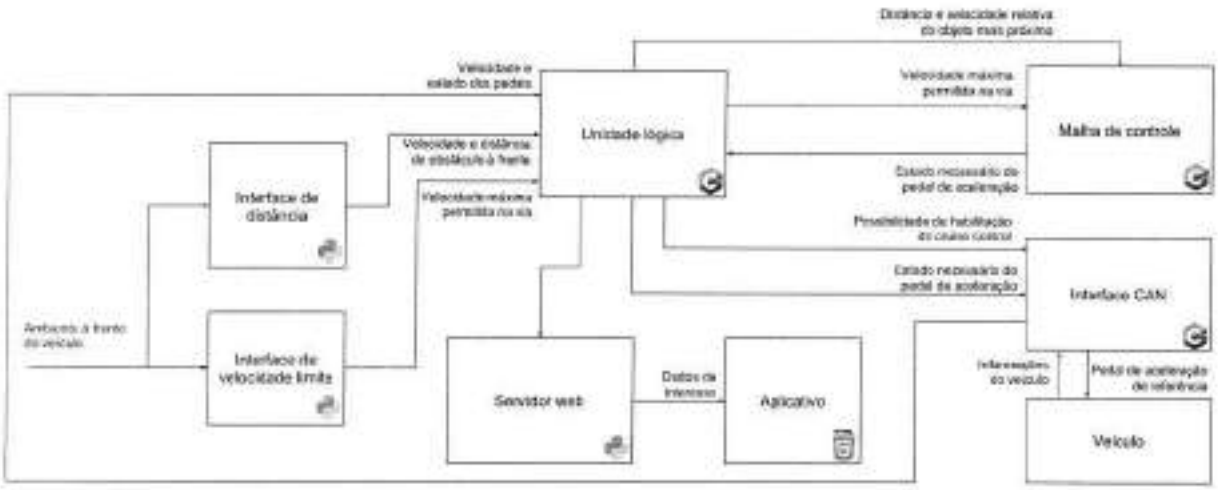
À esquerda, uma situação de frenagem real, e à direita, uma situação de aceleração. Fonte: Autores

9 IMPLEMENTAÇÃO

9.1 Introdução

Segundo a decomposição funcional estabelecida na fase conceitual do projeto, o projeto foi implementado conforme a Figura 31:

Figura 31: Diagrama da implementação de nível 1 do projeto.



Fonte: Autores

De acordo com o previsto, cada módulo lógico foi escrito de maneira independente e agnóstico aos sensores utilizados ou a algoritmos externos.

A seguir serão explicadas em detalhes as decisões sobre o hardware e software utilizados.

9.2 Processador principal

Para a computação das funções principais do sistema, foi utilizado um Raspberry Pi Modelo 3, um computador embarcado baseado na arquitetura ARM que conta com uma unidade de processamento Broadcom BCM2387, que contém um processador ARM Cortex-A53 64 *bits* de quatro núcleos com uma frequência de 1.2 GHz e um co-processador gráfico embarcado

VideoCore IV de dois núcleos. O computador embarcado conta com 1 *GB* de RAM LPDDR2 e também possuí conectividade 802.11 b/g/n e Bluetooth 4.1 (RS, 2016). Uma ilustração do hardware em questão pode ser vista na Figura 32.

Figura 32: Raspberry Pi Model 3

Fonte: RS (2016)

Esta placa foi escolhida por contar com todos os recursos necessários ao desenvolvimento do projeto do ponto de vista de conectividade e apresentar performance compatível com as tarefas inicialmente planejadas utilizando um radar automotivo. Adicionalmente, a placa apresenta dimensões reduzidas ($85x56x17\ mm^3$), baixo consumo energético ($5\ V$, $2.5\ A$ máximo, portanto $12.5\ W$) e conta com um banco GPIO de 40 pinos, permitindo futuras expansões se necessário.

O computador embarcado realiza todos os cálculos para a implementação dos sistemas de detecção de velocidade limite e de detecção de distâncias, bem como cálculos relacionados à malha de controle e ao servidor web que fornece dados ao aplicativo desenvolvido.

Durante o desenvolvimento do projeto, no entanto, foi percebido que o processador embarcado no Raspberry Pi Model 3 não seria suficiente para computar todos os blocos implementados em uma taxa de amostragem suficiente (superior a $10\ Hz$), o que motivou o uso de um computador com processador Intel Core i5 para realizar os cálculos no protótipo construído. Acredita-se que com devida otimização dos algoritmos de computação de disparidade e uma tradução do código desenvolvido em linguagem Python para C seja possível atingir a performance desejada com o Raspberry Pi 3.

9.3 Detecção de placas de trânsito

O algoritmo de detecção de placas de trânsito foi desenvolvido com o objetivo de detectar apenas placas de velocidade no padrão brasileiro determinado pela norma ABNT NBR ISO 3864-1, que determina o conteúdo, apresentação e cores das placas de trânsito brasileiras. Dentre as categorias de implementação investigadas, foi adotado o método tradicional de detecção por *feature extraction* frente à implementação com redes neurais por questões de praticidade e custo computacional.

9.3.1 Descrição do algoritmo utilizado

O método utilizado para detectar a velocidade limite da via consiste em uma série de operações aplicadas a um *stream* de vídeo capturado por uma câmera acoplada ao veículo com o intuito de extrair a informação textual contida nas placas visíveis na via, de maneira similar ao que foi encontrado na literatura (COSTA, 2013) (ESCALERA et al., 2003). Em uma visão geral, é possível dividir o processo nas seguintes etapas principais:

1. Detecção de contornos de placas de velocidade
2. Determinação do conteúdo interno à placa
3. Detecção dos caracteres internos à placa
4. Reconhecimento dos caracteres detectados e interpretação da velocidade final

9.3.2 Hipóteses simplificadoras

Algumas hipóteses foram adotadas com o objetivo de simplificar o problema de busca de placas de trânsito em imagens. Como determinado pelos requisitos do sistema na fase de planejamento, as placas a serem detectadas devem seguir o padrão brasileiro de sinalização como determinado pelo Conselho Nacional de Trânsito (CONSELHO NACIONAL DE TRÂNSITO, 2005), e portanto devem atender às seguintes características relevantes:

1. Devem possuir forma circular, seguindo a especificação do sinal R-19
2. Devem possuir orla e tarja na cor vermelha no padrão Munsell 7.5R 4/14
3. Devem possuir fundo na cor branca e símbolos e legendas na cor preta
4. Devem possuir fundo na cor branca e símbolos e legendas na cor preta

5. Devem ser utilizadas fontes dos seguintes tipos: *Helvética Medium, Arial, Standard Alphabets for Highway Signs and Pavement Markings* ou similares. (CONSELHO NACIONAL DE TRÂNSITO, 2005)

Além disso, também foi adotado que as placas de trânsito a serem detectadas devem conter entre dois e três dígitos numéricos, podendo representar velocidades entre 10 e 999 km/h, mas limitados via software a um intervalo de 10 a 120 km/h para evitar detecções indevidas. (Conselho Nacional de Trânsito, 2007)

Tais hipóteses simplificadoras permitiram a criação de um algoritmo otimizado à detecção de placas no padrão brasileiro e portanto mais eficiente não só em termos computacionais, mas também em termos de detecção de falsos positivos do que um algoritmo genérico de busca textual em placas de qualquer tipo. Em termos concretos, as seguintes otimizações foram aplicadas:

- A detecção de contornos (passo 1) foi simplificada para a detecção da cor vermelha da orla das placas, no padrão Munsell 7.5R 4/14 (CONSELHO NACIONAL DE TRÂNSITO, 2005)
- A detecção do conteúdo interno à placa (passo 2) foi simplificado para a detecção do círculo determinado pelo contorno reconhecido no passo 1
- O reconhecimento dos caracteres (passo 4) foi feito com base na fonte Arial

9.3.3 Realização do algoritmo

A detecção e interpretação de placas de trânsito foi realizada com o auxílio da biblioteca de software OpenCV (ITSEEZ, 2016e), uma iniciativa de código aberto com o intuito de prover uma infraestrutura comum para aplicações de visão computacional. A biblioteca contém várias funções que aceleram o desenvolvimento de algoritmos de processamento de imagens e vídeos, indo desde operações básicas como carregar imagens do disco até operações complexas como detecção de *features*, operações de perspectiva, entre outras.

Foi escolhida a linguagem Python (PYTHON, 2016) para a implementação do algoritmo, já que esta é uma linguagem de alto nível que proporciona grande flexibilidade e facilidade para o desenvolvimento de protótipos, mas que tem desempenho comparável ao de linguagens compiladas, como C e C++.

As etapas do algoritmo, conforme descritas em 9.3.1, foram implementadas utilizando funções disponíveis na biblioteca OpenCV, conforme será descrito nas seções subsequentes.

9.3.3.1 Detecção de contornos de placas de velocidade

A imagem adquirida da câmera embarcada é segmentada (ITSEEZ, 2016b) no espaço de cores Hue, Saturation, Value (HSV) (SMITH, 1978) em torno do padrão Munsell 7.5R 4/14 [3], como determinado pelo Conselho Nacional de Trânsito (Conselho Nacional de Trânsito, 2007), admitindo uma pequena margem de erro empírica decorrente de imprecisões na pintura e na calibração de cor da câmera utilizada.

9.3.3.2 Determinação do conteúdo interno à placa

Após a seleção da região do contorno da placa em vermelho, utiliza-se a transformada de Hough (ITSEEZ, 2016a) para a detecção de círculos presentes na figura segmentada. Após a transformada, espera-se obter o centro e raio dos círculos de contorno detectados pelo passo anterior, associados às placas de trânsito visíveis na imagem. De posse do círculo que determina o contorno da placa, seleciona-se a área interna da figura para a etapa seguinte.

Uma etapa adicional de filtragem dos resultados da transformada se mostrou necessária, visto que com frequência eram detectados múltiplos círculos ao redor de uma mesma posição da imagem, convergindo para o contorno real da placa. Sendo assim, os círculos encontrados na imagem são analisados com base em seu centro e raio, e caso sejam similares, são agregados em um só círculo médio.

9.3.3.3 Detecção dos caracteres internos à placa

Com base na área detectada pela etapa anterior, aplica-se uma nova segmentação binária no espaço HSV selecionando as regiões em preto da imagem, correspondente aos caracteres presentes na placa. Uma função de detecção de *blobs* (ITSEEZ, 2016d) é aplicada no resultado da segmentação em busca de caracteres de tamanho aceitável proporcionalmente ao tamanho da placa, dentre os quais os três maiores são selecionados.

9.3.3.4 Reconhecimento dos caracteres detectados e interpretação da velocidade final

Com os três maiores *blobs* selecionados através da etapa anterior, comparam-se os potenciais dígitos com referências para cada um dos dígitos de 0 a 9 na fonte Arial, calculando-se a correlação normalizada entre as imagens (ITSSEZ, 2016). A imagem de referência com a maior correlação com o *blob* detectado é tomada como o dígito detectado, e os dígitos detectados são

[3] No sistema HSV, a cor Munsell 7.5R 4/14 corresponde a (358, 84, 74).

concatenados para formar a velocidade detectada da placa.

9.4 Medição de distância

O método principal para a medição de distância de carros adjacentes a ser utilizada no projeto é baseado em estereoscopia, como descrito na seção 2.1.1. Foram desenvolvidos um aparelho de detecção estereoscópico e um algoritmo de medição de distância que implementa a técnica de mapeamento de profundidade, conforme descrito na seção 2.1.1.

9.4.1 Construção física

A construção de uma câmera estereoscópica foi feita com a utilização de duas câmeras Logitech C270 dispostas adjacentemente com seus centros ópticos alinhados a uma distância constante.

Figura 33: Produto final que conta com o computador embarcado e a câmera estereoscópica

Fonte: Autores

9.4.2 Descrição do algoritmo utilizado

A detecção de distância através de estereoscopia consiste na criação de uma mapa de profundidade a partir da disparidade entre as duas imagens adquiridas. A geração do mapa de profundidade também foi realizada com o auxílio da biblioteca de software OpenCV (ITSEEZ,

2016e). Implementou-se o algoritmo descrito em 2.1.1 na linguagem Python (PYTHON, 2016), utilizando funções nativas para a criação de mapas de disparidade a partir de um conjunto de imagens de uma câmera estereoscópica (ITSEEZ, 2016c). Uma ilustração destas etapas pode ser vista com detalhes na Figura 49b.

Após a construção do mapa de distâncias, pequenas áreas inconsistentes e detecções espúrias são eliminadas através de um filtro de detecção de regiões que utiliza como métrica a área detectada. Os *pixels* de distância restantes são agrupados de acordo com seu valor de distância calculado, segmentando a imagem em diferentes campos de distância. Isto permite encontrar objetos sólidos e diferenciá-los do plano de fundo sem utilizar segmentação por arestas ou cores.

Uma vez detectadas as potenciais regiões de interesse, uma função custo é computada de forma a estabelecer a detecção mais confiável. Tal função custo leva em consideração a distância da região ao eixo central da imagem (O) e a média da distância calculada ao longo de todos os seus *pixels* (D). Escolhe-se como melhor detecção a região cujo produto $O.D$ é mínimo.

9.5 Malha de controle

Um controlador estável para a planta levantada do veículo foi construído por um grupo de Projeto de Formatura da Escola Politécnica da Universidade de São Paulo também associado ao Grupo de Eletrônica Automotiva da Poli-USP em parceria com este projeto.

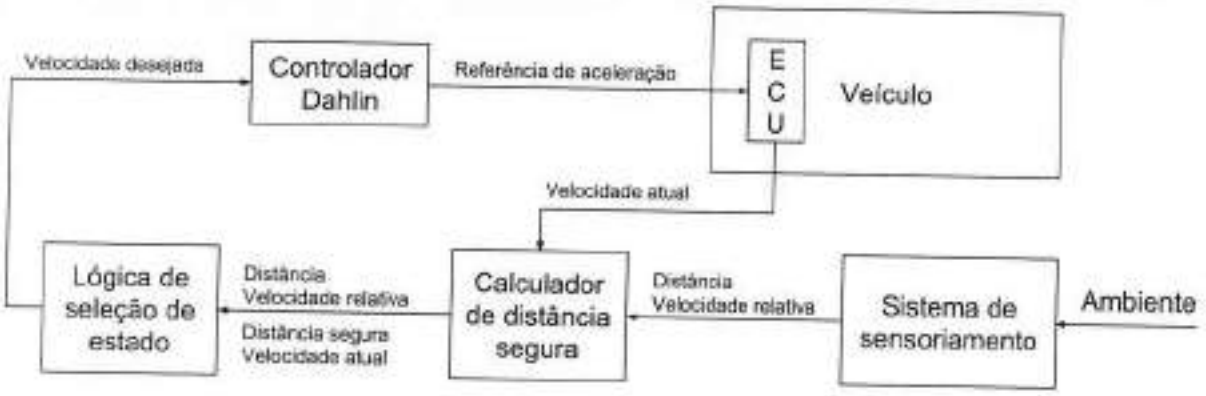
O grupo formado pelos alunos do curso de Engenharia Elétrica com ênfase em Controle e Automação projetou um controlador de tempo discreto para aplicação no sistema de controle de cruzeiro utilizando o algoritmo de controle de Dahlin (SEBORG et al., 2016), especializado em controles de plantas de primeira ordem com atraso de transporte. (SOUZA; GATTI, 2016)

A aplicação do algoritmo foi adequada visto que o levantamento da planta do veículo indicou que este pode ser aproximado por um sistema de primeira ordem com um atraso de transporte de acordo com a Equação8.1. O controlador desenvolvido tem função de transferência descrita pela Equação 9.1, onde $G_{MF}(z)$ é a função de transferência da malha fechada do sistema e $G(z)$ é a função de transferência de malha aberta.

$$C(z) = \frac{1}{G(z)} \frac{G_{MF}(z)}{1 + G_{MF}(z)} \tag{9.1}$$

Após a determinação da função de transferência do controlador, a malha de controle foi fechada através de realimentação pela interface CAN e dados obtidos do sistema de sensoriamento, conforme ilustrado na Figura 34.

Figura 34: Diagrama de blocos da malha de controle fechada



Fonte: Autores

O cálculo da distância segura de trânsito foi feito com base na velocidade instantânea do veículo controlado, tendo em vista que maiores velocidades necessitam de uma distância maior para correções de velocidade de cruzeiro, a fim de tornar o sistema seguro e confortável para o usuário.

9.5.1 Implementação

O sistema de controle em malha aberta, após discretizado através da transformada Z, foi implementado em linguagem C com o uso de *buffers* de atraso na entrada e saída do bloco do controlador. A função de transferência do controlador foi separada em dois vetores de coeficientes: numerador e denominador, a partir dos quais o resultado do próximo coeficiente de saída é calculado.

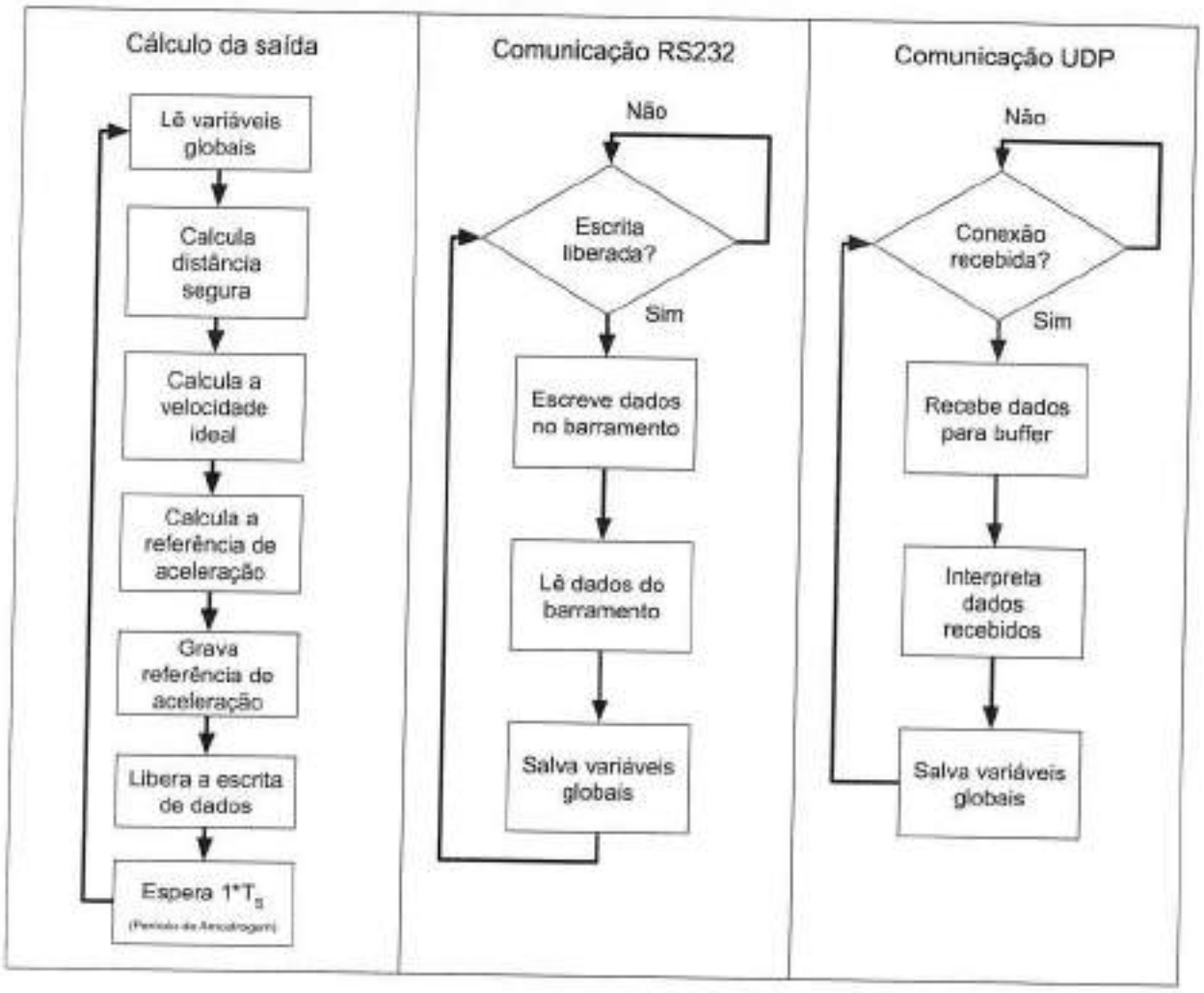
Para completar a malha fechada, lógicas de seleção do modo de controle (entre controle de cruzeiro com velocidade constante e controle de cruzeiro adaptativo) bem como referências de distância segura com base na velocidade foram adicionadas.

Foi adotada precisão dupla (*double*) para as análises iniciais de comportamento do sistema, podendo esta escolha ser revista em caso de problemas de complexidade computacional ou incompatibilidade com o hardware final escolhido.

O sistema consome dados vindos da rede CAN do veículo, através de pacotes especificamente construídos para este fim como descrito na seção 9.8. A malha de controle foi calculada para operar com período de amostragem de 0.1 s, ou seja, a uma frequência de 10 Hz. Um diagrama do fluxo do software implementado para o controlador, comunicação serial e

comunicação UDP pode ser visto na Figura 35.

Figura 35: Diagrama do software da malha de controle



Fonte: Autores

Cada parte do sistema de controle, isto é, o bloco de cálculo dos dados de saída, a comunicação serial e a comunicação UDP foram implementadas em *threads* separadas utilizando a biblioteca *pthread*, na linguagem C. Mecanismos de sincronização como semáforos e *mutexes* foram empregados para garantir a consistência de dados e a execução do código em intervalos precisos de tempo, correspondendo ao período de amostragem adotado (100 *ms*).

9.6 Comunicação com o veículo

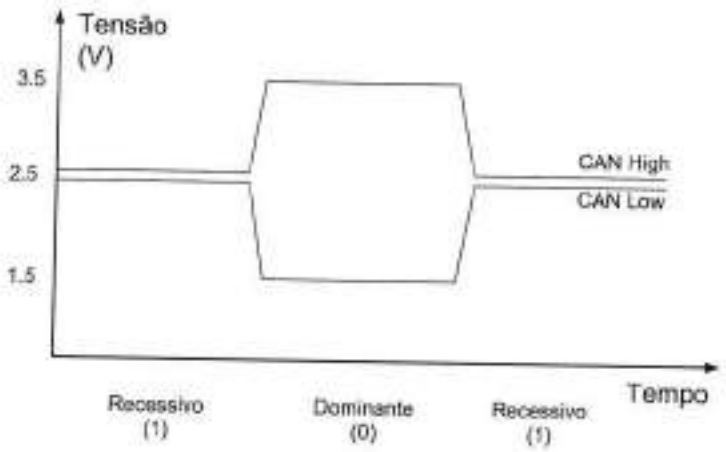
O mecanismo de atuação e leitura de informações do veículo empregado neste projeto utiliza o barramento CAN, presente em todos os veículos modernos e regulado pelas normas ISO

11898-2 e ISO 11898-1. Durante o desenvolvimento do projeto, um sistema de interface com o barramento CAN e de adaptação de dados foi desenvolvido com o intuito de possibilitar a atuação no sistema controlando a aceleração do veículo, bem como a leitura de dados de interesse como a velocidade atual do mesmo.

9.6.1 Especificações físicas

O barramento CAN utilizado na indústria automotiva é projetado para permitir a comunicação entre centenas nós, com extensão de centenas até milhares de metros de cabos. O barramento ainda prevê velocidades de comunicação de até 1 $Mbps$ no padrão tradicional de comunicação, podendo ser estendido para 5 $Mbps$ no modo de alta velocidade. O bit 1 no barramento é considerado recessivo, e o bit 0, ativo. Tais bits são expressos através da diferença de potencial entre as linhas CAN High e CAN Low, que são mantidas em estado intermediário $VCC/2$ quando não endereçadas. O bit recessivo é representado por uma diferença de tensão entre as linhas High e Low de $-500\ mV$ até $50\ mV$, e o bit dominante, por uma diferença de tensão de $1.5\ V$ até $3.0\ V$ (RICHARDS; Microchip Technology, 2002), segundo a especificação ISO11898-2 e conforme ilustra a Figura 36.

Figura 36: Especificação elétrica dos bits de comunicação CAN



Fonte: Autores

9.6.2 Especificações de mensagens

Uma mensagem CAN é composta de vários campos de funções específicas, cumprindo papéis de *handshake*, *acknowledgement*, identificação e transmissão de dados. Cada mensagem do protocolo é iniciada por um bit dominante *Start of Frame* (SOF), seguida pelo seu campo

de arbitragem, que contém o identificador da mensagem sendo enviada. Este campo é utilizado para definir a prioridade da mensagem no barramento, como será discutido adiante. O campo de controle contém o tamanho da mensagem de dados sendo enviada, já que o campo de dados pode conter de 0 a 8 *bytes* de dados. Há um campo de verificação (CRC) de 16 *bits* para permitir a detecção de erros de transmissão, e um campo de *acknowledgement* (ACK) de 2 *bits*, seguido por um bit dominante de *End of Frame* (EOF) (BOSCH, 1991).

Tabela 15: Ilustração dos campos de uma mensagem CAN

Mensagem						
SOF	Campo de arbitragem (11 bits)	Campo de controle (6 bits)	Dados (0 - 8 bytes)	CRC (16 bits)	ACK	EOF

Fonte: Autores

9.6.3 Arbitragem

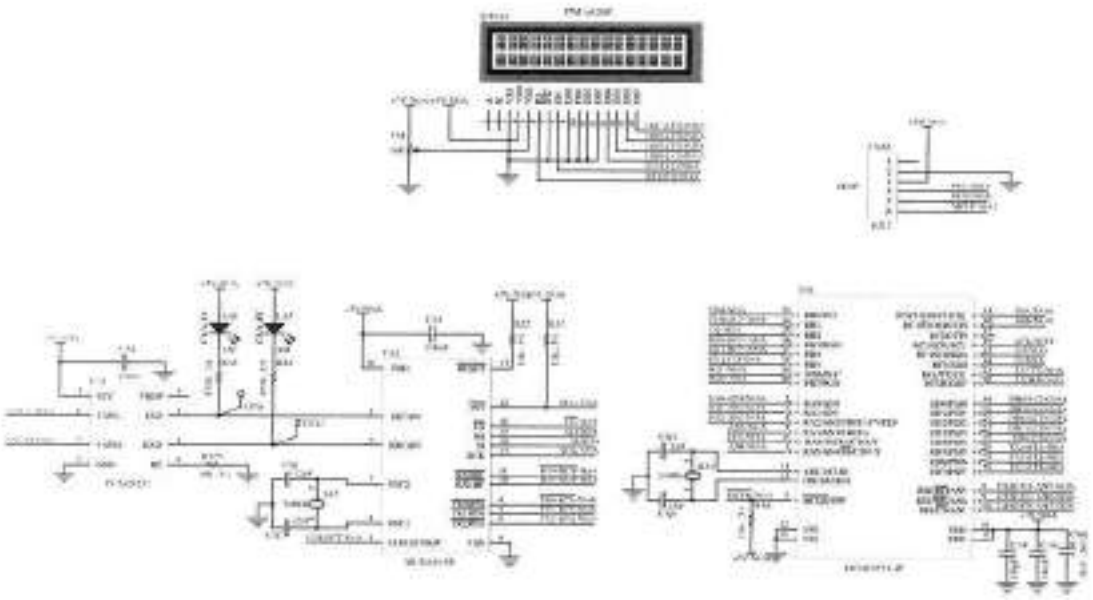
O protocolo também conta com um mecanismo de resolução de conflito entre mensagens que impede a colisão entre nós durante a comunicação. O processo conhecido como arbitragem define uma ordem de precedência entre os nós de comunicação CAN baseado em seu identificador. Identificadores de menor ID têm maior precedência no processo de arbitragem, e este acontece bit-a-bit durante a chamada fase de arbitragem do barramento. Nessa fase cada nó presente tenta escrever seu ID no barramento um bit por vez, lendo em seguida o resultado compartilhado entre os nós. Já que existem no barramento bits dominantes ('0') e recessivos ('1'), é possível que um nó que esteja tentando escrever um bit recessivo leia no barramento compartilhado um bit dominante. Se este for o caso, o nó perdeu a arbitração nesse momento e deve entrar em modo somente-leitura até que uma nova fase de arbitragem se inicie. É um requisito do sistema que cada nó opere com IDs de mensagens distintos, uma vez que esta é a única forma de garantir que não haverá colisões entre envios de mensagens no barramento.

9.6.4 Hardware

A interface de hardware com o barramento CAN do veículo foi feita utilizando uma placa didática (Figura 38) disponibilizada pelo Grupo de Eletrônica Automotiva da Escola Politécnica

da Universidade de São Paulo, que contém 3 nós CAN equipados com um transceptor CAN MCP2515 (MICROCHIP, 2012) e um microcontrolador programável PIC16F877A (MICROCHIP, 2013), conforme ilustra o diagrama esquemático na Figura 37. A comunicação entre o transceptor CAN e o microcontrolador é feita através do protocolo SPI, e o microcontrolador, por sua vez, recebe e fornece informações para o mundo externo através da linha serial RS232 também incorporada. O equipamento foi conectado no barramento CAN do veículo conforme ilustra a Figura 39

Figura 37: Diagrama esquemático de um nó da placa de comunicação CAN



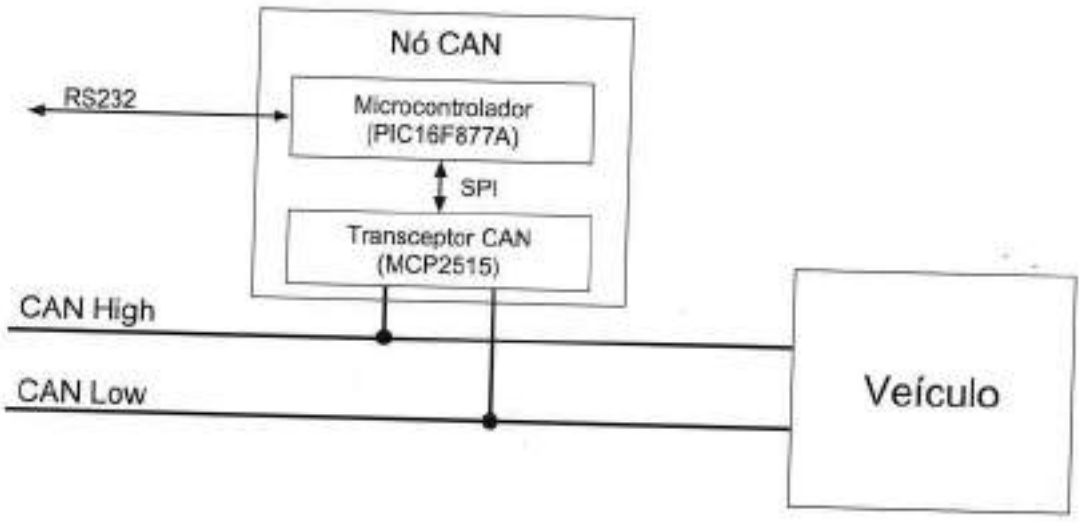
Fonte: Autores

Figura 38: Placa de comunicação CAN

Fonte: Autores

Toda comunicação adicional entre o nó CAN e o resto do sistema foi feita através de comunicação serial utilizando o protocolo RS232, que permite o envio dos valores de referência calculados pela malha de controle ao microcontrolador PIC, bem como a leitura do valor de velocidade atual do veículo, utilizado para fechar a malha de controle.

Figura 39: Diagrama de conexão entre o módulo de interface CAN, o veículo e o sistema externo



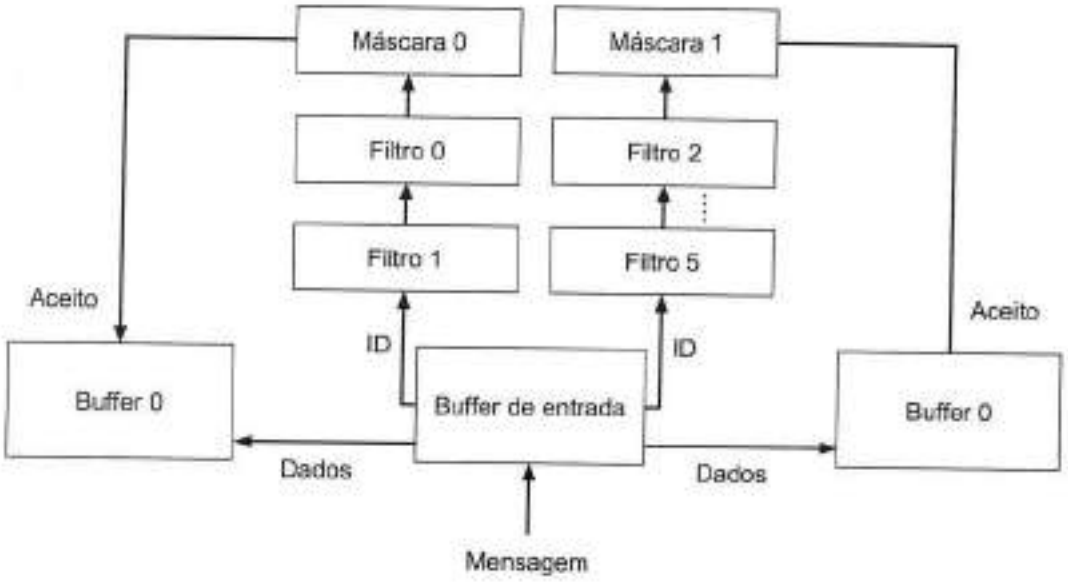
Fonte: Autores

9.6.5 Controlador MCP2515

O controlador MCP2515 é um dispositivo independente para interface com a rede CAN presente no veículo de testes. Ele apresenta capacidade de ler e escrever mensagens de 0 a 8 *bytes* de dados na rede CAN a uma velocidade de 1 $Mpbs$, e possui filtros e máscaras implementados em hardware que permitem segmentar as mensagens aceitas durante a leitura do barramento.

Uma vez que o número de mensagens que são trocadas entre os diferentes dispositivos presentes num veículo moderno são muito elevados, a rede CAN apresenta atividade constante e muitas mensagens com identificadores e conteúdos distintos. Dessa forma, é essencial ao projeto ter a capacidade de selecionar apenas os IDs desejados dentre as mensagens trafegando no barramento, e isto é feito diretamente pelo controlador MCP2515 através de suas máscaras e filtros de hardware, como ilustra a Figura 40

Figura 40: Diagrama do sistema de filtragem de mensagens CAN presente no controlador MCP2515. Fonte: Microchip, traduzido e adaptado pelos autores.



Fonte: Autores

Nesta aplicação, o controlador foi configurado de forma a aceitar apenas mensagens com ID $0x590$, que corresponde à mensagem da ECU modifica que fornece, entre outros dados, a velocidade atual do veículo.

Quando uma mensagem que atende aos requisitos de filtro e máscara chega ao controlador, este encaminha seu conteúdo e ID a um dos dois *buffers* de entrada, dependendo de qual grupo de filtros registrou a ocorrência. Um bit de sinalização de novas mensagens, $CANINTF.RXnIF$,

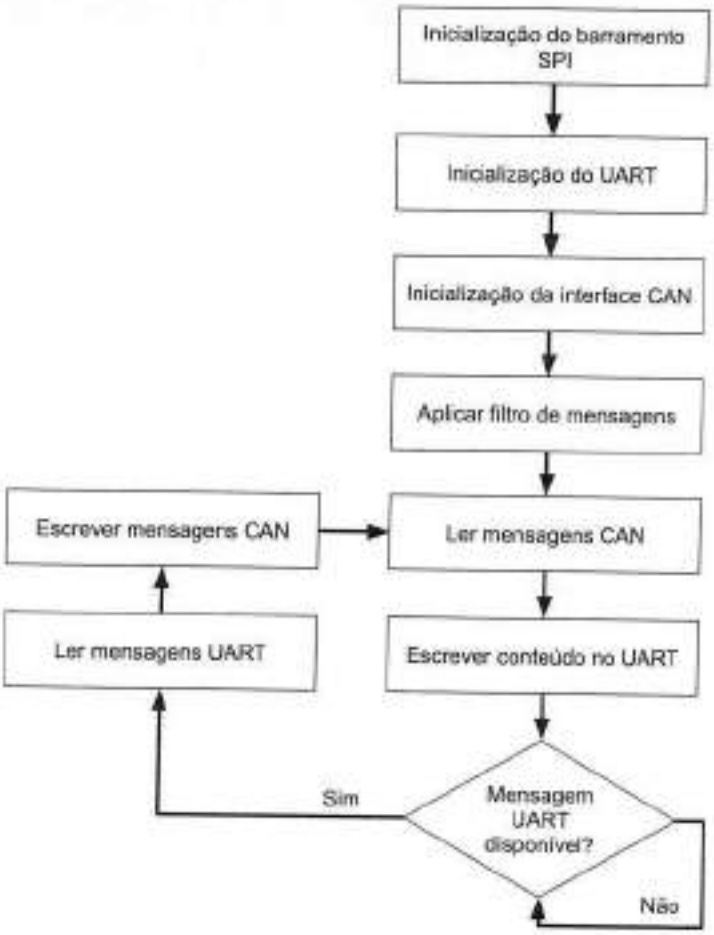
fica em valor lógico '1' quando uma nova mensagem está disponível para leitura no *buffer* n. Durante a interface com o controlador, este bit do registrador *CANINTF* é lido com frequência no loop principal para determinar se há uma mensagem de interesse disponível.

Para a transmissão de informação através do protocolo CAN, o controlador MCP2515 possui três registradores de saída de 14 bytes cada um, 5 dos quais são utilizados para o identificador das mensagens, 1 é reservado para bits de controle e os 8 restantes podem ser utilizados para o envio de dados arbitrários. Uma vez carregados os registradores com os dados desejados, o processo de envio é iniciado através da escrita de valor lógico '0' no bit *TXnRTS* do registrador de controle de envio, onde n corresponde ao canal de envio que foi populado, dentre os 3 presentes.

9.6.6 Microcontrolador PIC16F877A

A interação entre o microcontrolador presente na placa e o transceptor CAN foi mediada pelo microprocessador PIC16F877A programável disponível na placa. Um fluxo de software foi desenvolvido de forma a implementar não só a comunicação entre o veículo e o nó CAN, mas também para permitir a interface do sistema externo com o carro. O software implementado se encarrega de toda a configuração dos periféricos conectados ao microcontrolador, bem como de suas funções de interface com o controlador MCP2515 através do barramento SPI. Em sua função principal, o software inicializa a comunicação SPI, a interface serial universal (UART), a interface CAN e configura o controlador CAN para receber apenas as mensagens de interesse através de filtragem por hardware. Uma vez que a inicialização tenha sido concluída, o software passa para um loop infinito onde receberá mensagens CAN do veículo, escreverá o conteúdo das mensagens para o mundo externo através da interface UART, lerá a referência de aceleração calculada pela malha de controle e escreverá este valor na malha CAN do veículo para interface com a ECU modificada, conforme ilustra a Figura 41. O sistema implementado no PIC16F877A funciona como *slave* com respeito ao sistema externo, e portanto tem uma oportunidade de sincronização através de *busy waiting* durante o loop principal do programa. O dispositivo aguarda a chegada de uma mensagem válida no *buffer* vinda do sistema externo pela UART, e só então prossegue com a leitura da mensagem e respectiva escrita na rede CAN.

Figura 41: Diagrama de fluxo do software de comunicação embarcado no microcontrolador PIC16F877A

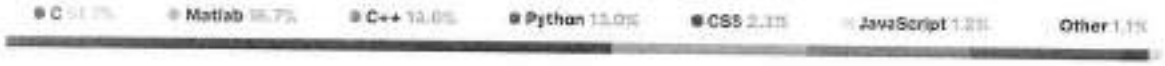


Fonte: Autores

9.7 Comunicação entre módulos

Conforme indica a Figura 42, foram utilizadas múltiplas linguagens de programação no desenvolvimento do sistema, empregando-se uma abordagem de software chamada de microsserviços, que é uma especialização e implementação de arquiteturas orientadas a serviço usada para construir sistemas de software flexíveis e de *deploy* independente. (FOWLER, 2014)

Figura 42: Distribuição das linguagens de programação utilizadas no projeto, conforme mostrado no repositório no GitHub



Fonte: Autores

O módulo de detecção de distância age como servidor UDP e envia dois dados: a distância do obstáculo à frente, em *cm*, e a velocidade absoluta deste, em cm/s. Para a minimização do *payload* da mensagem, utilizou-se 2 *bytes* para cada variável, permitindo a representação de distâncias de até 655 *m* e de velocidades superiores a 200 km/h.

O módulo de detecção de velocidade limite na via também age como servidor UDP e envia apenas um dado: a velocidade detectada, em km/h. Utilizou-se 1 *byte* para tanto, permitindo a representação de velocidades de até 255 km/h.

A unidade lógica possui clientes UDP para consumir os dados produzidos pelos dois blocos acima. Além disso, trafega dados à interface CAN por meio do protocolo RS232 e de um cabo serial USB-DB9. A mensagem possui 1 *byte* que indica a referência do pedal de aceleração, em uma escala que vai de 0 a 255 e 1 *byte* de segurança cujo primeiro bit indica se é seguro ativar o *cruise control*. Finalmente, a unidade lógica persiste no disco um arquivo JSON com os dados referentes ao veículo e ao obstáculo à frente.

Um servidor HTTP, programado na linguagem Python, expõe o conteúdo desse arquivo JSON, um formato leve e difundido para intercâmbio de dados computacionais (NURSEITOV et al., 2009), no *endpoint* da forma *http://192.168.0.13:8080/data.json*, onde o endereço IP será diferente conforme a rede. Ademais, o *endpoint* da forma *http://192.168.0.13:8080* disponibiliza uma página HTML que será descrita na Seção 9.9, passível de ser consultada por qualquer aparelho com um navegador de Internet moderno.

A Tabela 16 sumariza os protocolos e as mensagens trocadas entre as unidades que constituem o sistema.

Tabela 16: Especificação das mensagens utilizadas para comunicação entre módulos

<table>
<tr><th rowspan="2">Servidor</th><th rowspan="2">Cliente</th><th rowspan="2">Protocolo</th><th colspan="4">Dados</th></tr>
<tr><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th></tr>
<tr><td>Dist</td><td>UL</td><td>UDP</td><td colspan="2">Distância
(cm)</td><td colspan="2">Velocidade
(cm/s)</td></tr>
<tr><td>Veloc</td><td>UL</td><td>UDP</td><td>Velocidade limite
(km/h)</td><td></td><td></td><td></td></tr>
<tr><td>UL</td><td>CAN</td><td>RS232</td><td>Referência do pedal
(0-100%)</td><td>Byte de segurança</td><td></td><td></td></tr>
<tr><td>UL</td><td>Servidor Web</td><td>Persistência em disco</td><td colspan="4" rowspan="2">Dados de interesse do veículo e do obstáculo à frente, em formato JSON
(Payload variável)</td></tr>
<tr><td>Servidor Web</td><td>App</td><td>HTTP</td></tr>
</table>

Fonte: Autores

A independência entre as unidades, cuja comunicação segue um protocolo bem definido garante facilidade na troca de componentes. Para se trocar o algoritmo de detecção de distâncias de modo a usar um sensor de outra tecnologia, por exemplo, não é necessário reescrever, recompilar ou fazer o *flash* de todo o código-fonte do sistema. Para tanto, basta fazer as alterações no módulo de distância e iniciar uma nova instância deste módulo, desde que os protocolos de comunicação continuem sendo respeitados.

9.8 Veículo de testes

Neste trabalho, utilizou-se um Volkswagen Polo Sedan 2004 equipado com uma ECU aberta, desenvolvida pelo Grupo de Eletrônica Automotiva da Poli-USP, que permite a atuação no automóvel.

Figura 43: Polo Sedan 2004

Fonte: Autores

A mensagem de leitura customizada possui um payload de 8 bytes e indica, dentre outros, a velocidade medida do carro, qual marcha está engatada e informações sobre erros na comunicação. Os erros podem ser de software, como falha na execução de rotina, ou de hardware, como desconexão de sensores. Esses dados alimentam em parte a lógica de controle.

Para atuação no veículo, é enviada uma mensagem com um payload de 1 byte com o valor

de referência do pedal, compreendido entre 0 e 255, onde 0 representa um pedal não pressionado e 255 um pedal totalmente pressionado. Por segurança, todas as mensagens enviadas à ECU devem ser precedidas por um token de segurança de 3 chars formando a palavra "ECU".

Tabela 17: Frames de mensagem de escrita e leitura

ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x500	**Ref. Torque** 0 - 100% Fator 0,01		**Velocidade Medida** Fator 0,01 Km/h		**Marcha Engatada** 0 (neutro) - 6	**Pedal_Embreagem** Fator 0,01 Preso, 0 Solto **Pedal_Freio** 1 Preso, 0 Solto **Modo_Cruzeiro** 1 ligado, 0 desligado **Modo_Pedal_Sim** 1 ligado, 0 desligado	**Ref. Cruzeiro** Fator 1,0 Km/h	**Erro de Hardware** 0 - Sem erro > 0 - Erro detectado
0x200	"E" Token de segurança	"C" Token de segurança	"U" Token de segurança	**Pedal Simulado** 0 - 100% 0 - 255				
0x201	"E" Token de segurança	"C" Token de segurança	"U" Token de segurança	**Modo_Pedal_Sim** 1 Liga, 0 Desliga	**Modo_Operação** 1 Esporte, 0 Normal	**Modo_Ref_Marcha** 0 - 11 0 = 8000RPM RL Lenta (RPM) = 600 + (Modo-1)*50		

Fonte: Autores

9.9 Aplicativo

De forma a melhor mostrar os resultados para o usuário, foi desenvolvido um aplicativo que exibe informações como a distância e a velocidade do carro a frente, a velocidade máxima da via e a velocidade instantânea do veículo.

O aplicativo foi escrito usando HTML5, uma linguagem de marcação utilizada na construção de páginas *web*, e pode ser utilizado por qualquer dispositivo eletrônico com conexão à internet e equipado com um navegador de Internet moderno, como *notebooks*, *smartphones* e *tablets*.

A página HTML faz requisições periódicas ao *endpoint* que expõe o arquivo *JSON*, descritos na Seção 9.7, via Javascript, a principal linguagem para programação *client-side* em navegadores *web*. Caso haja um erro de conexão ou de interpretação de dados, uma mensagem é mostrada ao usuário descrevendo o problema.

Figura 44: *Screenshots* do aplicativo

Screenshots do aplicativo funcionando em um Samsung Galaxy S4, que possui como sistema operacional o Android. O app mostra, quando disponível, a velocidade atual do veículo, o limite detectado na via, a distância do veículo à frente e a velocidade deste. Fonte: Autores

10 TESTES UNITÁRIOS E DE INTEGRAÇÃO

10.1 Introdução

Conforme descrito na Seção 7.1, diversos testes, tanto de software quanto de hardware, são necessários para garantir o bom funcionamento do sistema e a segurança dos usuários.

Propiciados pela modularização do sistema, os testes unitários garantiram a integridade de cada bloco de maneira isolada, simulando as dependências de cada unidade. Em seguida, foram feitos testes de integração conectando dois ou mais módulos. Finalmente, o teste de aceitação garantiu a integridade do sistema como um todo.

10.2 Detecção de placas de trânsito

O sistema implementado apresentou um bom índice de detecção com fotos estáticas de placas de trânsito brasileiras, bem como quando aplicado a um *stream* de vídeo vindo da câmera embarcada.

Figura 45: Resultados de cada etapa do algoritmo de detecção de limite da via

No sentido horário: Imagem original, imagem segmentada no espaço de cores, círculos detectados na imagem, texto interno à placa detectado. Fonte: Autores

Como teste de validação do sistema, foi criada uma matriz de confusão para determinar a taxa de acerto e erro do sistema frente a diferentes situações de uso. 13 imagens de diferentes tipos de placas de velocidade foram utilizadas, e 5 imagens que representavam paisagens ou outras placas de trânsito que não de velocidade representavam amostras negativas reais. Os resultados deste teste podem ser vistos na Tabela 18.

Tabela 18: Matriz de confusão do sistema de detecção de placas.

	Detectado: **Positivo**	Detectado: **Negativo**
Real: **Positivo**	13	2
Real: **Negativo**	3	3

Fonte: Autores

Dentre as placas de velocidade corretamente detectadas como verdadeiras, o sistema teve 10 acertos dentre 13 imagens, correspondendo a uma precisão de detecção de aproximadamente

77%. Possíveis técnicas para melhorar os resultados até então obtidos serão discutidas na seção a seguir.

10.3 Medição de distância

O sistema de medição de distância por estereoscopia tem como base o cálculo de um mapa de disparidade entre duas imagens vindas de câmeras posicionadas a uma distância fixa uma da outra, porém para o bom funcionamento do algoritmo de cálculo de disparidade, é necessário conhecer os parâmetros intrínsecos e extrínsecos das câmeras sendo utilizadas.

Os parâmetros extrínsecos da câmera definem a transformação geométrica em 3 dimensões que deve ser aplicada a cada ponto do espaço para se obter o ponto equivalente no sistema de coordenadas da câmera. Em outras palavras, esses parâmetros definem a translação e rotação da câmera em relação ao sistema de coordenadas adotado no espaço. Nesta aplicação em particular, devido à existência de múltiplas câmeras no sistema, os parâmetros extrínsecos também definem a relação física, isto é, a translação e rotação entre as câmeras (HEIKKILA; SILVEN, 1997).

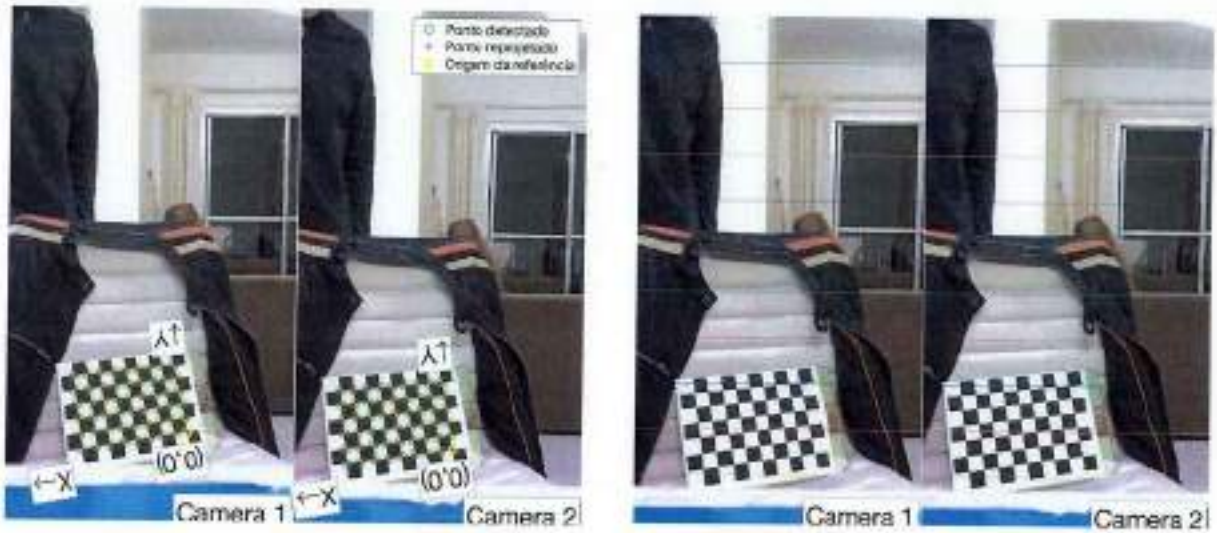
Complementar aos parâmetros extrínsecos, os parâmetros intrínsecos de uma câmera definem a transformação que mapeia pontos tridimensionais no sistema de coordenadas da câmera a pontos no plano-imagem bidimensional desta. Utilizando um modelo simplificado que exclui os efeitos de lentes, tais parâmetros definem a distância focal da câmera, seu fator de escala e seu ponto principal, isto é, o centro da imagem.

10.3.1 Calibração do par estereoscópico

Os parâmetros intrínsecos e extrínsecos das câmeras foram obtidos através de um processo de calibração usual e já padronizado na literatura de visão estereoscópica utilizando uma representação denominada *checkerboard*, composta por um padrão alternado de quadrados pretos e brancos. Ao se capturar uma imagem do padrão conhecido, é possível estimar os parâmetros intrínsecos da câmera.

Utilizando múltiplas imagens do padrão de calibração, tiradas ao mesmo tempo com cada uma das câmeras do par estereoscópico, é possível estimar também a posição relativa entre as câmeras, e com isso seus parâmetros extrínsecos. A Figura 46 ilustra o processo de calibração e uma visualização dos dados obtidos através do software Matlab.

Figura 46: Visualização do processo de calibração de câmeras estereoscópicas

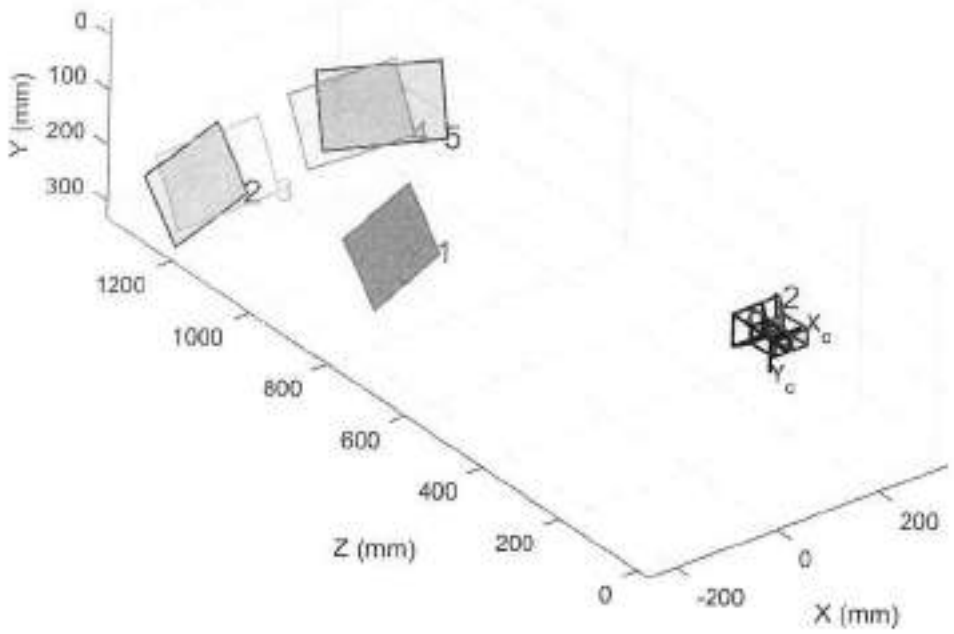


À esquerda, ilustra-se o processo de detecção dos padrões de *checkerboard* utilizados para calibração, e à direita, mostra-se o par de imagens retificado. Note que linhas traçadas entre as imagens retificadas cruzam os mesmos pontos físicos. Fonte: Autores

De posse dos parâmetros intrínsecos e extrínsecos das câmeras, é possível calibrar o sistema estereoscópico para prepará-lo para a criação dos mapas de disparidade. O algoritmo computa a diferença de posição entre dois pontos homólogos da imagem no eixo paralelo ao deslocamento entre as câmeras, porém para que esta computação tenha êxito, é necessário que tais pontos estejam sob uma mesma linha horizontal, como ilustrado pela Figura 46.

Finalmente, buscando estimar com confiança os parâmetros das câmeras, foram realizadas 5 capturas de imagens do padrão de calibração, e o conjunto de imagens foi utilizado para computar os parâmetros intrínsecos e extrínsecos. Uma vez computados, os parâmetros podem ser utilizados para obter a posição dos objetos originais no espaço tridimensional a partir das imagens obtidas, e esta técnica foi aplicada pra visualizar a situação atual da calibração e verificar a consistência dos dados obtidos. Uma visualização dos padrões de calibração reconstituídos no espaço tridimensional pode ser vista na Figura 47.

Figura 47: Visualização do resultado da calibração das câmeras estereoscópicas

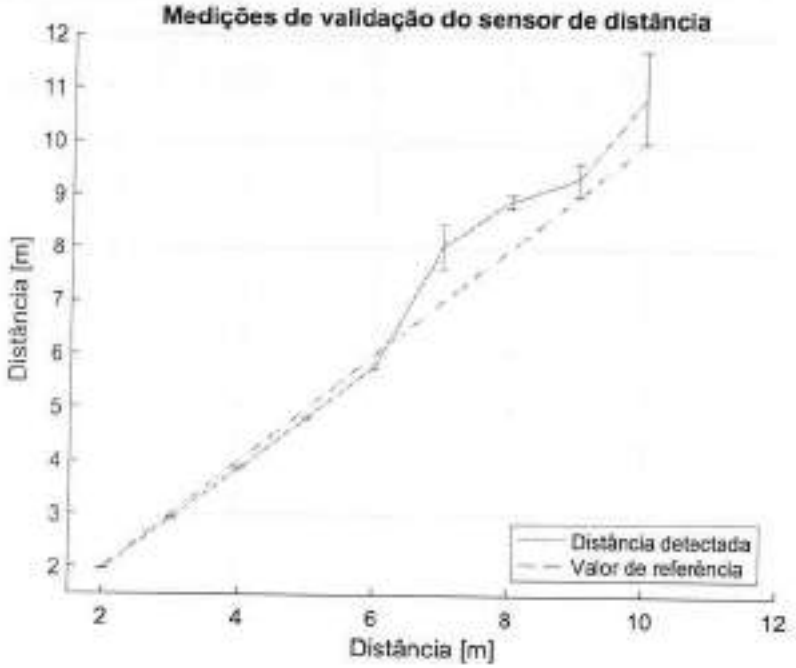


Os semiplanos de 1 a 5 representam as posições dos planos de calibração carregando o padrão de *checkerboard* utilizado no processo, e as câmeras 1 e 2 estão posicionadas de acordo com a estimativa atual dos parâmetros extrínsecos. Fonte: Autores

10.3.1.1 Validação do funcionamento

Para a validação dos dados de distância medidos pela câmera estereoscópica, foram comparadas medições provenientes do sistema a distâncias conhecidas de um anteparo em diferentes situações, representando potenciais casos de uso. Foram levantadas distâncias entre 5 e 10 metros para a validação do sistema, e os resultados obtidos podem ser vistos na Figura 48. O valor final da medição do sistema foi determinada através da média dos pontos detectados do anteparo, e o desvio padrão das amostras foi adotado como medida da incerteza de medição, denotado na Figura 48 pelas barras de erro.

Figura 48: Resultados da validação do par estereoscópico

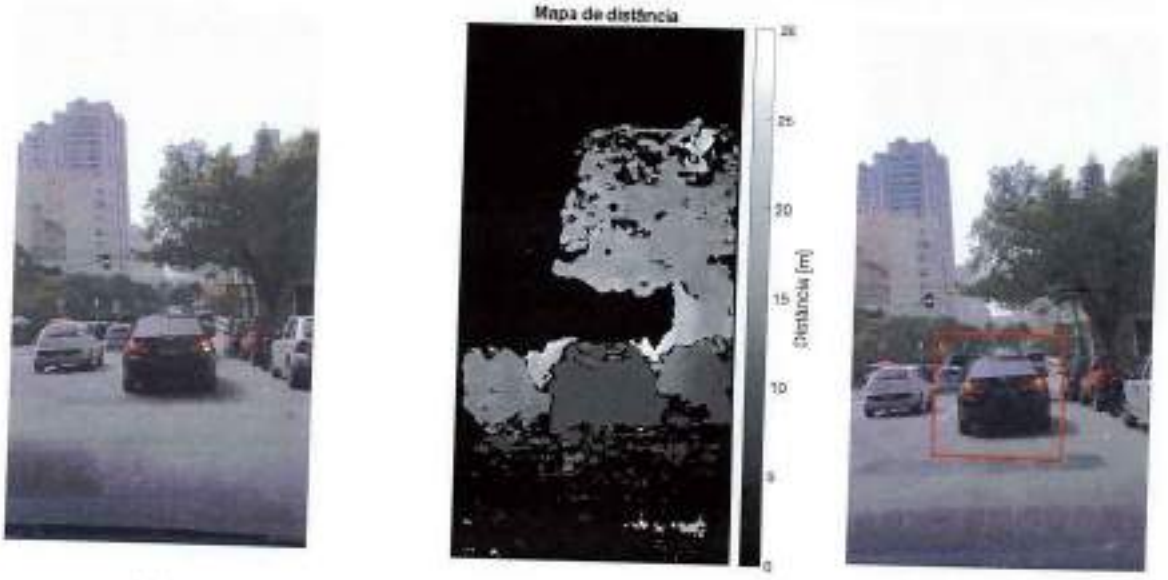


Fonte: Autores

10.3.1.2 Testes veiculares

Buscando testar o sistema de detecção de distância em uma situação real de uso, o par estereoscópico em sua montagem final foi embarcado em um veículo comum e foram gravados diversos vídeos de situações encontradas em rodovias. Tais vídeos foram introduzidos no algoritmo de detecção de distâncias para uma computação *offline* e posterior visualização dos dados obtidos. A Figura 49 ilustra o processo de detecção de uma amostra de dados partindo de um vídeo salvo previamente.

Figura 49: Ilustração do sistema de detecção de distâncias funcionando em um ambiente veicular



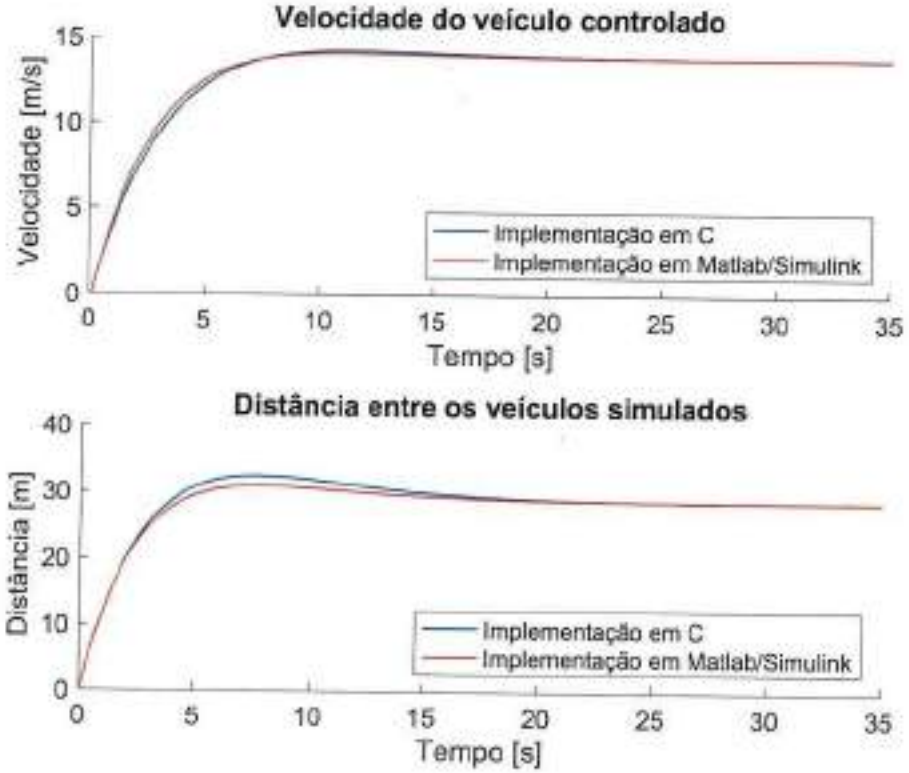
(a) (b) (c)

(a) Imagem de uma das câmeras do par estereoscópico. (b) Mapa de distância gerado a partir dos parâmetros intrínsecos, extrínsecos e do mapa de disparidade das câmeras estereoscópicas. (c) *Frame* demonstrando o sistema de detecção de obstáculos. O retângulo vermelho delimita a região de detecção, e o valor da distância média ao obstáculo encontra-se inscrito ao retângulo, onde se lê 8.1247 m. O vídeo completo desta ilustração pode ser visto em: https://goo.gl/SIgMA2. Fonte: Autores

10.4 Malha de controle

A malha de controle desenvolvida no software Matlab vista na Figura 34 representa o conjunto final de elementos necessários para o controle do veículo. Tal malha foi realizada na linguagem C para que pudesse ser embarcada em um hardware portátil e utilizada em tempo real no sistema. Após a implementação descrita na seção 9.5.1, desejou-se comparar o resultado obtido pela implementação em C com o controle inicialmente projetado. Para isso, foi aplicado um degrau unitário à entrada de ambos os sistemas e os resultados da saídas foram comparados, apresentando boa correlação dentro da precisão numérica adotada. Os resultados da resposta ao degrau obtidos para cada sistema podem ser vistos na Figura 50.

Figura 50: Resultados da comparação entre o sistema de controle implementado em Matlab/Simulink e em código C.



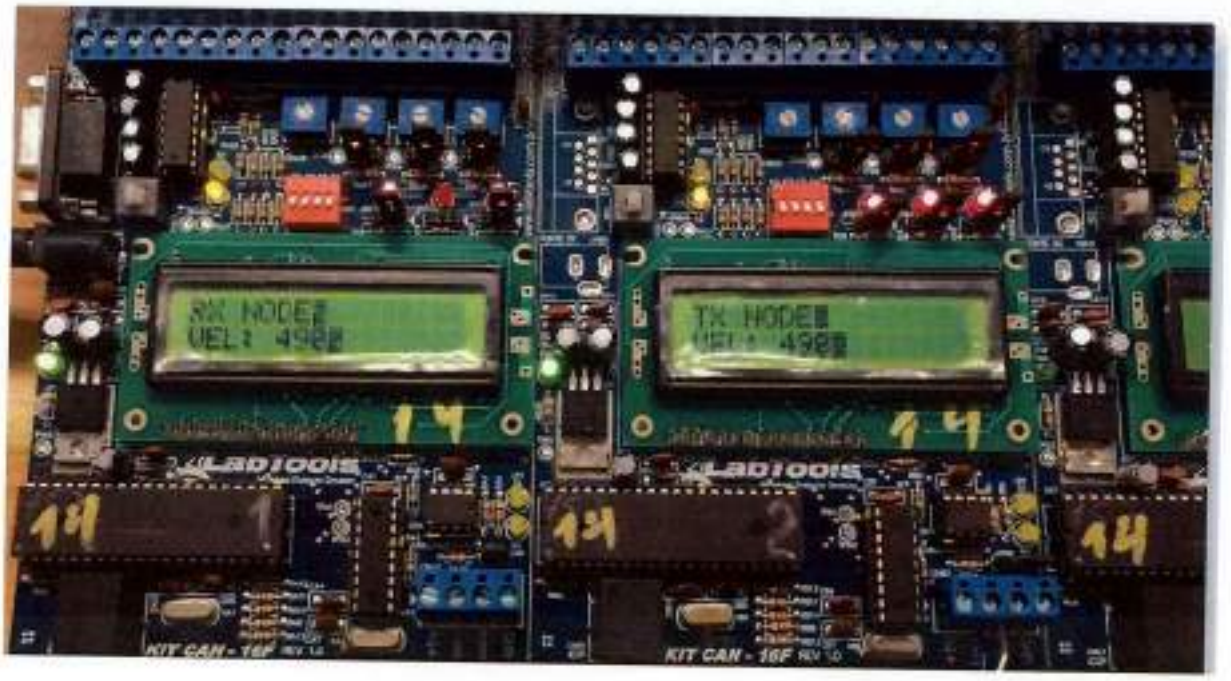
Fonte: Autores

O teste realizado mostra que há apenas pequenas diferenças entre o controlador simulado no ambiente do Simulink e o implementado em linguagem C, e portanto é possível verificar que o controlador desenvolvido no software Matlab foi reconstituído em linguagem C sem perdas de funcionalidade. As divergências encontradas podem ser atribuídas a diferenças na ordem dos cálculos no *loop* de controle e diferenças na precisão adotada para os cálculos nas plataformas de simulação

10.5 Comunicação com o veículo

Testes preliminares do sistema de interface com a rede CAN envolveram a comunicação entre apenas 2 nós da própria placa didática de interface, sendo que um nó era responsável por enviar mensagens de diferentes IDs e o outro, por ler apenas os IDs de interesse, sendo capaz portanto de discriminar entre diferentes mensagens, como ilustra a Figura 51

Figura 51: Teste de comunicação CAN entre dois nós da placa didática de interface



À esquerda encontra-se o nó receptor, que lê o conteúdo e o ID da mensagem vinda do nó transmissor, à esquerda. Fonte: Autores

Uma vez que a base de comunicação havia sido estabelecida e validada através de testes offline apenas entre dois nós da placa, o sistema didático foi conectado à rede CAN do veículo de testes onde buscou-se ler as mensagens de interesse vindas da ECU modificada. As leituras ocorreram com sucesso e a discriminação entre as mensagens a partir de seus identificadores únicos seguiu como esperado, o que permitiu o teste final envolvendo também a escrita de valores arbitrários de referência de aceleração para o veículo. O teste de escrita foi conduzido com o veículo montado em um dinamômetro, onde foi possível verificar que os valores de referência de aceleração enviados para a malha CAN estavam sendo corretamente lidos pela ECU modificada e utilizados no controle de rotação do motor.

10.6 Testes de integração do sistema

Os testes de integração ocorreram em etapas. Com o veículo fixado no laboratório do IEE-USP, foi testado primeiramente o código de envio de mensagens para a ECU, com um sinal de referência de pedal fixo e constante. Isso garantiu que era possível atuar no carro com confiabilidade.

Nesta etapa, a integridade da malha de controle e dos algoritmos de detecção de distâncias

já tinha sido avaliada e pôde-se, portanto, testar a atuação no carro com a malha de controle conectada. Como o veículo encontrava-se fixo, a distância do veículo para a alimentação da malha foi simulada, caracterizando um sistema de hardware-in-the-loop. Os resultados dos testes de integração do sistema podem ser vistos na Seção 11.2

11 RESULTADOS

11.1 Capacidades do sistema

O sistema desenvolvido se demonstrou capaz de controlar a referência de aceleração do veículo de forma a controlar sua velocidade através de interação direta com a ECU modificada pelo barramento CAN. A referência de aceleração é calculada por uma malha fechada de controle que tem como entradas a distância do veículo à frente, a velocidade deste mesmo veículo, a velocidade limite da via e a velocidade atual do veículo controlado. A velocidade limite da via é detectada pelo sistema através de técnicas de processamento de imagem descritas na seção 9.3, e a distância e velocidade relativa do veículo à frente são providas pelo sistema estereoscópico descrito na seção 9.4. A velocidade atual do veículo é lida da ECU modificada utilizando o barramento CAN e o controle é feito escrevendo-se mensagens contendo a referência do pedal de aceleração no mesmo barramento.

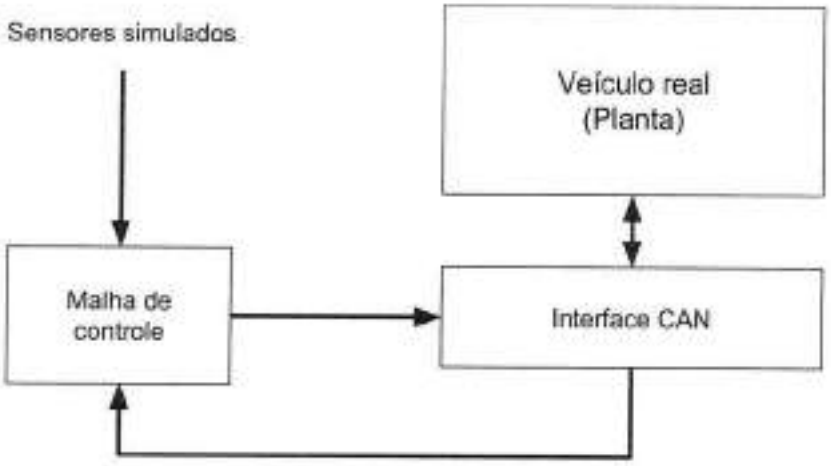
11.2 Validação HIL

O funcionamento do sistema foi validado utilizando-se testes *hardware-in-the-loop* em diferentes situações de interesse para verificar se seu comportamento se dá de acordo com o esperado.

Simulações HIL são caracterizadas pela utilização de parte do hardware real do sistema em presença de elementos digitalmente simulados. Esta técnica permite obter uma simulação mais fidedigna do sistema real do que seria possível apenas com um modelo matemático dos sensores, atuadores e planta envolvidos (BACIC, 2005). Neste projeto, os atuadores e a planta estão integrados no veículo em si, sendo representados pela ECU e o controlador de rotação do motor, e o subsistema composto pelo motor e transmissão, respectivamente.

Para as simulações deste projeto, o veículo real foi colocado em um dinamômetro, aparelho que permite a livre rotação das rodas de tração do veículo em presença de uma carga variável, e as entradas do sistema de detecção de placas e distância foram simuladas, como ilustra a Figura 52. Todos os testes foram conduzidos com o dinamômetro oferecendo 10% de carga, o que representa uma situação típica de asfalto em uma rua plana.

Figura 52: Diagrama da simulação Hardware in the Loop.



Fonte: Autores

Os testes realizados verificam a operação do sistema tanto em modo de controle de cruzeiro adaptativo, isto é, com um veículo de referência sendo seguido e mantido a uma distância constante, como no modo de controle de cruzeiro direto, onde procura-se manter o veículo a uma velocidade constante. A Tabela 19 contém uma descrição das situações simuladas.

Tabela 19: Relação dos testes de validação HIL executados

#	Velocidade do líder [km/h]	Distância inicial do líder [m]	Velocidade máxima [km/h]
1	20	200	33
2	$20 \rightarrow 90$	50	33
3	$50 \rightarrow 40$	200	60
4	-	∞	33
5	-	∞	50

Fonte: Autores

O teste número 1 representa uma situação onde o veículo controlado começa a uma distância elevada do veículo líder e num primeiro momento trafega dentro da velocidade limite da via, 33 km/h. Após se aproximar do veículo à frente, o sistema alterna para modo de controle de cruzeiro adaptativo e passa a trafegar à mesma velocidade do veículo líder, a uma velocidade constante. A Figura 53 representa a situação mencionada.

Figura 53: Resultados de distância e velocidade relativa do teste HIL número 1.

(a) O veículo controlado inicialmente acelera até o limite de velocidade imposto pela via, alcança o veículo líder e então passa a trafegar em uma velocidade semelhante à sua.

(b) Nota-se que o veículo controlado parte de uma distância de 200m do veículo líder, alcança-o e passa para o modo de controle de cruzeiro adaptativo, de forma que a velocidade relativa ao veículo líder se anula e é mantida uma distância constante de segurança.

Fonte: Autores

O teste número 2 simula uma situação onde o veículo controlado começa a uma distância de 50 metros do veículo líder, atinge a distância de regime nesta velocidade e então percebe o veículo líder se afastar com velocidade superior à limite da via. Nessa situação, o veículo mantém a velocidade máxima de 33 km/h após o veículo líder se afastar, como demonstra a Figura 54.

Figura 54: Resultados de velocidades e esforço de controle do teste HIL número 2.

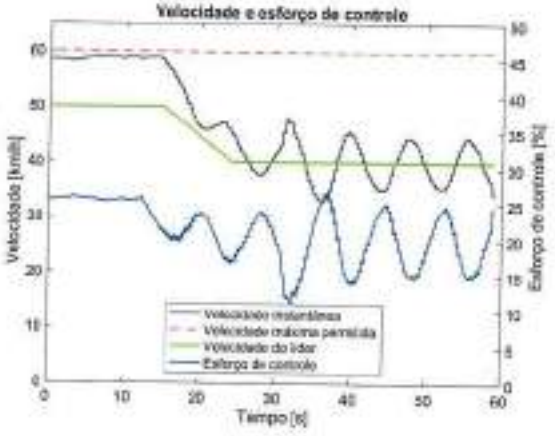
(a) O veículo controlado atinge regime e acompanha o veículo à frente, até que este acelera para uma velocidade superior à da via, situação onde o veículo controlado deve respeitar o limite de velocidade.

(b) Nota-se que o veículo controlado atingiu uma distância constante e velocidade relativa nula no início do teste, e então percebe uma distância cada vez maior do veículo de referência já que este está trafegando a uma velocidade muito superior à da via.

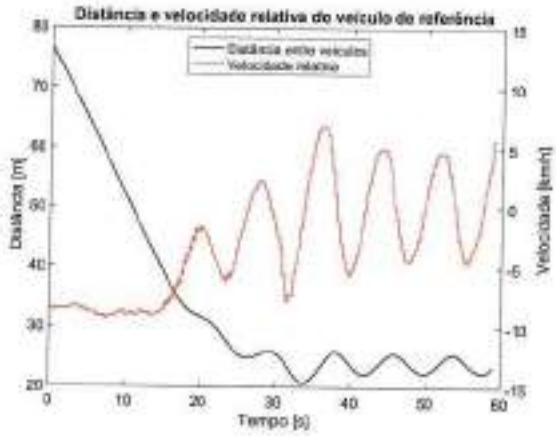
Fonte: Autores

O teste número 3 simula uma situação onde o veículo controlado começa a uma distância elevada do veículo líder, atinge a velocidade máxima da via de 60 km/h buscando se aproximar do veículo líder, e durante a aproximação percebe que este diminuiu sua velocidade de 50 para 40 km/h, e o acompanha na redução de velocidade de forma a evitar uma colisão, como ilustra a Figura 55.

Figura 55: Resultados de velocidades e esforço de controle do teste HIL número 3.



(a) O veículo controlado atinge a velocidade máxima de 60 km/h imposta pela via até perceber uma desaceleração do veículo a frente, situação onde reduz sua própria velocidade para acompanhá-lo.

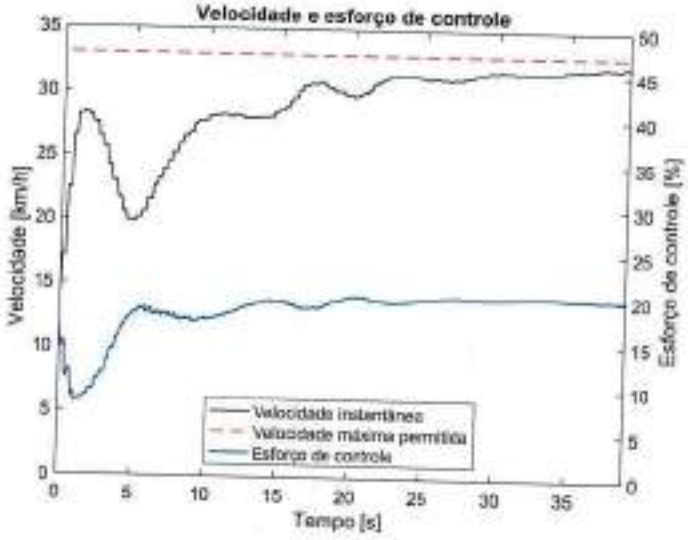


(b) Verifica-se uma aproximação inicial do veículo controlado ao veículo de referência, seguido por uma estabilização da distância entre os dois após a frenagem do veículo líder.

Fonte: Autores

O teste número 4 simula uma situação onde não há veículo de referência e o veículo controlado parte do estado de marcha lenta e atinge a velocidade limite da via, neste caso, de 33 km/h, como ilustra a Figura 56.

Figura 56: Resultados de velocidades e esforço de controle do teste HIL número 4.



O veículo controlado parte do estado de marcha lenta e atinge a velocidade máxima de 33 km/h imposta pela via. Fonte: Autores

O teste número 5 simula uma situação onde também não há veículo de referência e o veículo controlado parte do estado de marcha lenta e atinge a velocidade limite da via, de 50 km/h, como pode ser visto na Figura 57.

Figura 57: Resultados de velocidades e esforço de controle do teste HIL número 5.

O veículo controlado parte do estado de marcha lenta e atinge a velocidade máxima de 50 km/h imposta pela via. Fontes: Autores

12 TRABALHOS FUTUROS

12.1 Detecção de placas de trânsito

Com o objetivo de melhorar o índice de acertos do algoritmo e sua performance geral na detecção de placas de velocidade, algumas alternativas podem ser investigadas, a saber:

1. Utilização de um filtro para remoção de imperfeições nos contornos das placas
2. Ajustes de cor, brilho e saturação da imagem para maximizar o alcance dinâmico
3. Operações de filtragem para remoção de pequenas detecções indesejadas
4. Utilizar outros métodos de detecção de dígitos além da correlação normalizada.

Além disso, são possíveis também melhorias de performance do algoritmo implementado, que consistiriam não só nas otimizações gerais de programação linear, mas também na redução da resolução de processamento das imagens, quando possível, para diminuir o custo computacional das operações da biblioteca OpenCV.

12.2 Medição de distância

Para resultados mais precisos, seria possível a utilização de um radar automotivo ou de um Lidar, conforme já descritos na Seção 2.1. Durante o desenvolvimento deste trabalho adquiriu-se um radar automotivo da Bosch, modelo 5Q0.907.561.F para o Grupo de Eletrônica Automotiva da Escola Politécnica. O dispositivo já está em posse do Grupo, porém ainda é necessária uma customização do *firmware* por parte da Bosch que permita sua utilização, uma vez que ele não possui *datasheet* nem qualquer interface para desenvolvimento por terceiros.

Figura 58: Radar automotivo da Bosch com customização de *firmware* pendente

Fonte: Autores

12.3 Malha de controle

Na seção 11, foi visto que a implementação da malha de controle em C utilizada no projeto foi estável e conseguiu manter o controle do veículo tanto no modo de controle de cruzeiro direto quando de controle de cruzeiro adaptativo. No entanto, nota-se um comportamento oscilatório indesejado no sistema, que pode causar desconforto ao usuário, gastos excessivos de combustível e até provocar instabilidades mecânicas.

As oscilações presentes são inicialmente atribuídas a divergências entre o modelo originalmente levantado para o veículo utilizando o dinamômetro e o modelo real da planta composta pelo subsistema do motor e da transmissão controlado pela ECU modificada. Seria possível obter novas constantes para o modelo de primeira ordem com atraso de transporte originalmente proposta, e com isso calcular os coeficientes de um novo controlador melhor ajustado à planta real. Outra possibilidade para melhorar o sistema seria modificar a ordem ou tipo do sistema que representa o veículo, buscando melhor aproximar o modelo da realidade.

12.4 Comunicação com o veículo

Embora tenha sido de grande valia o *kit* de desenvolvimento CAN utilizado, para a construção de um produto comercialmente viável, componentes menores e mais integrados poderiam ser

utilizados.

Em um primeiro momento, o sistema final dispensaria o uso de um *display* LCD, nesta etapa tendo sido utilizado somente para fins de depuração. Além disso, poderia ser utilizado um controlador CAN MCP2515 discreto, e este poderia ser integrado diretamente às portas GPIO da placa de processamento principal, dispensando o uso de um microcontrolador PIC dedicado.

Isso permitiria também dispensar o uso do conversor USB-Serial USB-DB9, reduzindo consideravelmente o tamanho do sistema e sua capacidade de comunicação (*throughput*).

12.5 Veículo de testes

O veículo Polo Sedan 2004 utilizado possui câmbio manual, o que limita a atuação do sistema de forma a ter uma velocidade compatível apenas com a marcha naquele momento. Uma melhoria considerável seria permitir que o sistema alterasse também a marcha do veículo, permitindo um maior alcance de velocidades num dado instante.

De maneira semelhante, não foi possível atuar na frenagem do carro, já que o sistema de freio do veículo era inteiramente mecânico. Se este possuísse a tecnologia *brake-by-wire*, o sistema de *cruise control* poderia inclusive parar o carro completamente. Esta modificação aumentaria consideravelmente a desaceleração máxima do sistema e conferiria maior segurança em situações de frenagens repentinas.

13 CONCLUSÃO

Este trabalho mostrou que é possível propiciar recursos de direção assistida a um veículo, a baixo custo. Foi implementado um sistema de controle de cruzeiro adaptativo e de detecção de velocidade limite na via. Além disso, um aplicativo foi desenvolvido para mostrar dados pertinentes em tempo real.

Foram realizados testes unitários de cada componente do sistema e testes de integração utilizando-se um dinamômetro.

Através de uma simulação do tipo *hardware-in-the-loop* com os dados dos sensores simulados, observou-se que a atuação no veículo foi realizada com boa confiabilidade, de forma que este nunca ultrapassasse a velocidade limite da via nem colidisse com o veículo à frente em todos dos cenários reproduzidos.

O módulo de estimação de distâncias foi realizado utilizando um par de *webcams* porém não apresentou o alcance e confiabilidade necessários para uso em um veículo trafegando em uma rodovia, por exemplo. Contudo, em um trabalho futuro poderá ser utilizado o radar adquirido pelo Grupo de Eletrônica Automotiva da Poli, que garantirá a precisão e alcance necessários ao projeto.

O módulo de detecção de velocidade limite na via foi feito utilizando-se uma das *webcams* e possui precisão de aproximadamente 77%.

Da forma como foi realizado, o sistema tem aplicabilidade limitada no aspecto de integração com veículos de produção em massa, tendo em vista que o mecanismo de interface utilizado depende da modificação de sua ECU. Outras formas de integração podem ser estudadas, principalmente em se tratando de veículos mais modernos com tecnologias *drive-by-wire* e *pedal-by-wire*.

Por outro lado, o projeto demonstrou que fabricantes poderiam adicionar funcionalidades semelhantes sem elevar consideravelmente o preço de seus veículos, fornecendo ferramentas que auxiliem o motorista, mas que não dispensem a ação deste.

REFERÊNCIAS

ALTERA. *Implementing Digital Processing for Automotive Radar Using SoCs*. Altera, 2013. Disponível em: ⟨https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/wp/wp-01183-automotive-radar-socfpga.pdf⟩. Acesso em: 08 de jun. 2016.

BACIC, M. On hardware-in-the-loop simulation. In: *44. IEEE CONFERENCE ON DECISION AND CONTROL. Proceedings...* [S.l.: s.n.], 2005. p. 3194–3198.

BERTOZZI, M.; BROGGI, A. GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection. *Image Processing, IEEE Transactions on*, v. 7, n. 1, p. 62–81, 1998. Disponível em: ⟨http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=650851⟩. Acesso em: 24 de mai. 2016.

BMW. *Series 7 Driver Assistance page*. 2016. Disponível em: ⟨http://www.bmw.com/com/en/newvehicles/7series/sedan/2015/showroom/driver_assistance.htm⟩. Acesso em: 20 de jun. 2016.

BOSCH. *CAN Specification .v:2.0*. 1991. Disponível em: ⟨http://esd.cs.ucr.edu/webres/can20.pdf⟩. Acesso em: 10 de out. 2016.

CAMPBELL, J. B. *Introduction to remote sensing*. [S.l.]: Guilford Press, 1996.

CONSELHO NACIONAL DE TRÂNSITO. *Manual Brasileiro de Sinalização de Trânsito, Volume I*. 2005. Acesso em: 16 de set. 2016.

Conselho Nacional de Trânsito. *Sinalização Vertical de Regulamentação*. 2007. Disponível em: ⟨http://www.denatran.gov.br/publicacoes/download/manual_vol_i.pdf⟩. Acesso em: 24 de mai. 2016.

CONTINENTAL. *SRL1C Short-Range Lidar Datasheet*. [s.n.], 2012. Disponível em: ⟨https://drive.google.com/file/d/0Bzul4ZPCWjj5UmdtWEhXVTIleDQ/view?usp=sharing⟩. Acesso em: 24 de set. 2016.

CONTINENTAL. *ARS30X Long Range Radar Datasheet*. 2014. Disponível em: ⟨https://drive.google.com/file/d/0Bzul4ZPCWjj5UmdtWEhXVTIleDQ/view?usp=sharing⟩.

CONTINENTAL. *Standardized ARS Interface - Technical Documentation*. 2015. Acesso em: 20 de jun. 2016.

CONTINENTAL. *Radar PLC Manual*. 2016. Disponível em: ⟨https://drive.google.com/file/d/0Bzul4ZPCWjj5UmdtWEhXVTIleDQ/view?usp=sharing⟩. Acesso em: 24 de mai. 2016.

COSTA, C. Miguel Correia da. *Traffic sign detection*. [S.l.]: Faculdade de Engenharia do Porto, 2013.

DELPHI. *Delphi Electronically Scanning Radar*. 2016. Disponível em: ⟨http://delphi.com/manufacturers/auto/safety/active/electronically-scanning-radar⟩. Acesso em: 15 de mai. 2016.

DOYLE, W. P.; GUNN, C. L. *LiDAR for Terrain Mapping on the Alaska Pipeline Corridor*. [s.n.], 2009. Disponível em: ⟨http://www.arlis.org/docs/vol1/AlaskaGas/Paper/Paper_OFC_2009_LiDAR_TerrainMapping.pdf⟩. Acesso em: 23 de mai. 2016.

DRIESSEN, V. *A successful Git branching model*. 2010. Disponível em: ⟨http://nvie.com/posts/a-successful-git-branching-model/⟩. Acesso em: 23 de ago. 2016.

ESCALERA, D. la et al. Traffic sign recognition and analysis for intelligent vehicles. *Image and vision computing*, v. 21, n. 3, p. 247–258, 2003.

FORD. *Ford Fusion Hybric safety features*. 2016. Disponível em: ⟨https://owner.ford.com/how-tos/vehicle-features/safety.html?year=2016&model=Fusion%20Hybrid&sync=myTouch⟩. Acesso em: 20 de set. 2016.

FOWLER, M. *Microservice - A definition of this new architectural term*. 2014. Disponível em: ⟨http://martinfowler.com/articles/microservices.html⟩. Acesso em: 20 de nov. 2016.

HEIKKILA, J.; SILVEN, O. A four-step camera calibration procedure with implicit image correction. In: *IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. Proceedings...* [S.l.: s.n.], 1997. p. 1106–1112.

HIMMELSBACH, M. et al. LIDAR-based 3d object perception. In: *Proceedings of 1st international workshop on cognition for technical systems*. [s.n.], 2008. v. 1. Disponível em: ⟨http://www.cs.princeton.edu/courses/archive/spring11/cos598A/pdfs/Himmelsbach08.pdf⟩.

ITSEEZ. *Documentação da função HoughCircles do OpenCV*. 2016. Disponível em: ⟨http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/hough_circle/hough_circle.html⟩. Acesso em: 12 de nov. 2016.

ITSEEZ. *Documentação da função inRange do OpenCV*. 2016. Disponível em: ⟨http://docs.opencv.org/2.4/modules/core/doc/operations_on_arrays.html#inrange⟩. Acesso em: 3 de out. 2016.

ITSEEZ. *Documentação do algoritmo StereoBM do OpenCV*. 2016. Disponível em: ⟨http://docs.opencv.org/trunk/d9/dba/classcv_1_1StereoBM.html⟩. Acesso em: 23 de nov. 2016.

ITSEEZ. *Documentação do detector de blobs do OpenCV*. 2016. Disponível em: ⟨http://docs.opencv.org/trunk/d0/d7a/classcv_1_1SimpleBlobDetector.html⟩. Acesso em: 10 de nov. 2016.

ITSEEZ. *OpenCV - Open Source Computer Vision*. 2016. Disponível em: ⟨http://opencv.org/about.html⟩. Acesso em: 24 de out. 2016.

ITSSEZ. *Documentação da função matchTemplate do OpenCV*. 2016. Disponível em: ⟨http://docs.opencv.org/2.4/modules/imgproc/doc/object_detection.html#matchtemplate⟩. Acesso em: 27 de out. 2016.

KHOSHELHAM, K.; ELBERINK, S. O. Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications. *Sensors (Basel, Switzerland)*, v. 12, n. 2, p. 1437–1454, fev. 2012. ISSN 1424-8220. Disponível em: ⟨http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3304120/⟩. Acesso em: 24 de mai. 2016.

KUMAR, R.; PATHAK, R. Adaptive cruise control–towards a safer driving experience. *International Journal of Science and Engineering Res*, v. 3, n. 8, 2012. Disponível em: ⟨http://www.ijser.org/researchpaper%5CAdaptive-Cruise-Control-Towards-a-Safer-Driving-Experience.pdf⟩. Acesso em: 24 de mai. 2016.

MERCEDES-BENZ. *Distronic Plus technology page*. 2016. Disponível em: ⟨https://www.mbusa.com/mercedes/owners/videos/detail/videoId-6f31735fd7cee310VgnVCM1000007c184335RCRD⟩. Acesso em: 20 de ago. 2016.

MICROCHIP. *MCP2515 Datasheet*. 2012. Disponível em: ⟨http://ww1.microchip.com/downloads/en/DeviceDoc/21801G.pdf⟩. Acesso em: 25 de out. 2016.

MICROCHIP. *PIC16F87XA Datasheet*. 2013. Disponível em: ⟨http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf⟩. Acesso em: 25 de out. 2016.

MIURA, J.; KANDA, T.; SHIRAI, Y. An active vision system for real-time traffic sign recognition. In: *INTELLIGENT TRANSPORTATION SYSTEMS. Proceedings* ... IEEE, 2000. p. 52–57. Disponível em: ⟨http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=881017⟩. Acesso em: 22 de mai. 2016.

MURRAY, D.; LITTLE, J. J. Using real-time stereo vision for mobile robot navigation. *Autonomous Robots*, v. 8, n. 2, p. 161–171, 2000.

NURSEITOV, N. et al. Comparison of json and xml data interchange formats: a case study. *CAINE*, v. 1, n. 1, 2009. Disponível em: ⟨https://pdfs.semanticscholar.org/8432/1e662b24363e032d680901627aa1bfd6088f.pdf⟩. Acesso em: 04 de nov. 2016.

NXP SEMICONDUCTORS. *Automotive Radar High-resolution 77 GHz radar*. 2014. Acesso em: 24 de ago. 2016.

PYTHON. *Python Programming Language*. 2016. Disponível em: ⟨https://www.python.org/about/⟩. Acesso em: 18 de nov. 2016.

RAJAN KUMAN, K. Cruise Control Operation from Zero to Preset Speed-Simulation and Implementation. *International Journal of Information and Education Technology*, v. 1, n. 1, 2011. Disponível em: ⟨http://www.ijiet.org/papers/2-W13.pdf⟩. Acesso em: 23 de mai. 2016.

RICHARDS, P.; Microchip Technology. *AN228 - A CAN Physical Layer Discussion*. 2002. Disponível em: ⟨http://ww1.microchip.com/downloads/en/AppNotes/00228a.pdf⟩. Acesso em: 14 de out. 2016.

ROHLING, H.; MEINECKE, M. M. Waveform design principles for automotive radar systems. In: *INTERNATIONAL CONFERENCE ON RADAR. Proceedings*... [S.l.: s.n.], 2001. p. 1–4.

RS. *Raspberry Pi Model 3 Datasheet*. 2016. Disponível em: ⟨http://uk.rs-online.com/webdocs/14ba/0900766b814ba5fd.pdf⟩. Acesso em: 27 de set. 2016.

RUPP, J. D.; KING, A. G. Autonomous Driving-A Practical Roadmap 2010-01-2335. 2010. Disponível em: ⟨http://www.fujitsu.com/downloads/MICRO/fma/marcom/convergence/data/papers/2010-01-2335.pdf⟩. Acesso em: 22 de mai. 2016.

SAXENA, A.; CHUNG, S. H.; NG, A. Y. Learning depth from single monocular images. In: *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS. NIPS2005, 18. Conference...* [s.n.], 2005. p. 1161–1168. Disponível em: ⟨http://machinelearning.wustl.edu/mlpapers/paper_files/NIPS2005_684.pdf⟩. Acesso em: 4 de jul. 2016.

SEBORG, D. E. et al. *Process Dynamics and Control*. 4. ed. Wiley, 2016. ISBN 978-1-119-28595-3. Disponível em: ⟨http://www.wiley.com/WileyCDA/WileyTitle/productCd-111928595X.html⟩. Acesso em: 23 de jun. 2016.

SERMANET, P.; LECUN, Y. Traffic sign recognition with multi-scale convolutional networks. In: *(IJCNN), THE 2011 INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS. Proceedings...* IEEE, 2011. p. 2809–2813. Disponível em: ⟨http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6033589⟩. Acesso em: 22 de mai. 2016.

SMITH, A. R. Color Gamut Transform Pairs. In: *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY: ACM, 1978. (SIGGRAPH '78), p. 12–19. Disponível em: ⟨http://doi.acm.org/10.1145/800248.807361⟩. Acesso em: 24 de out. 2016.

SOFTWARE FREEDOM CONSERVANCY. *Git Version Control*. 2016. Disponível em: ⟨https://git-scm.com/⟩. Acesso em: 10 de nov. 2016.

SON, H.-S. et al. AD-Census Based Real-Time Stereo Matching Hardware Architecture. 2013. Disponível em: ⟨http://onlinepresent.org/proceedings/vol19_2013/49.pdf⟩. Acesso em: 24 de mai. 2016.

SOUZA, G. L. M.; GATTI, P. A. E. *Desenvolvimento e aplicação de um Controle de Cruzeiro Adaptativo. Trabalho de conclusão de curso - Escola Politécnica da Universidade de São Paulo*. 2016.

STEIN, G. P.; MANO, O.; SHASHUA, A. Vision-based ACC with a single camera: bounds on range and range rate accuracy. In: *IEEE INTELLIGENT VEHICLES SYMPOSIUM, 2003. Proceedings...* [S.l.: s.n.], 2003. p. 120–125.

TRINH, H.; MCALLESTER, D. Structure and motion from road-driving stereo sequences. In: *COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION - WORKSHOP. Proceedings...* [S.l.: s.n.], 2010. p. 9–16.

VELODYNE. *VLP-16 Datasheet. Rev-C*. 2016. Disponível em: ⟨http://velodynelidar.com/docs/datasheet/63-9229_Rev-C_VLP16_Datasheet_Web.pdf⟩. Acesso em: 20 de mai. 2016.

WEDEL, A. et al. Realtime depth estimation and obstacle detection from monocular video. In: FRANKE, K. et al. (Ed.). *Pattern Recognition: DAGM Symposium 28. Berlin. Proceedings...* [s.n.], 2006. p. 475–484. ISBN 978-3-540-44414-5. Disponível em: ⟨http://dx.doi.org/10.1007/11861898_48⟩.

WIDMANN, G. R. et al. *Comparison of lidar-based and radar-based adaptive cruise control systems*. SAE International, 2000. Disponível em: ⟨http://bit.ly/2i6w9Uc⟩. Acesso em: 23 de mai. 2016.

YASUGI, M. et al. 79ghz-band radar cross section measurement for pedestrian detection. In: *ASIA-PACIFIC MICROWAVE CONFERENCE (APMC). Proceedings...* [S.l.: s.n.], 2013. p. 576–578.