POLITÉCNICO DE LEIRIA | ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

DETEÇÃO DA FAIXA DE RODAGEM PARA CONDUÇÃO AUTÓNOMA

Mestrado em Engenharia Automóvel

Pedro Filipe Ramos Martins

Leiria, setembro de 2021

POLITÉCNICO DE LEIRIA | ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

DETEÇÃO DA FAIXA DE RODAGEM PARA CONDUÇÃO AUTÓNOMA

Mestrado em Engenharia Automóvel

Pedro Filipe Ramos Martins

Projeto realizado sob a orientação do Professor Doutor Carlos Fernando Couceiro de Sousa Neves, do Professor Doutor Hugo Filipe Costelha de Castro e do Professor Doutor Luís Manuel Conde Bento.

Leiria, setembro de 2021

Originalidade e Direitos de Autor

A presente dissertação de projeto é original, elaborada unicamente para este fim, tendo sido devidamente citados todos os autores cujos estudos e publicações contribuíram para a elaborar.

Reproduções parciais deste documento serão autorizadas na condição de que seja mencionado o Autor e feita referência ao ciclo de estudos no âmbito do qual o mesmo foi realizado, a saber, Curso de Mestrado em Engenharia Automóvel, no ano letivo 2020/2021, da Escola Superior de Tecnologia e Gestão do Politécnico de Leiria, Portugal, e, bem assim, à data das provas públicas que visaram a avaliação destes trabalhos.

Esta página foi intencionalmente deixada em branco

Dedicatória

Dizem que a história se repete, é verdade, há capítulos da vida que terminam da mesma forma que começam.

Uma promessa de beijos,

Dois braços à minha espera,

Frases que marcaram a minha memória e músicas que nunca esquecerei.

O que ambas tem em comum?

Uma pessoa com um coração do tamanho do mundo!

Alguém que me ensinou que por muito que o quadro pareça negro,

A tela é branca e recomeçar é superar.

Se no teatro era fácil interpretar o papel de seu neto, agora sei o porquê.

Foram tantos os momentos marcantes …

Nunca conseguirei ouvir Amália sem me lembrar de si,

E sorrir.

Frases como “Tu não vais ser mecânico, tu vais ser engenheiro mecânico!” e “Estuda Filipe que um dia vais ser tu a ensinar o teu pai!” foram muitas vezes hinos e ficarão para sempre gravadas como recordações.

Não sei se existem palavras que podem descrever o quão importante foi para mim, mas sei o quão me arrependeria se não tentasse.

Obrigado, Madalena.

Esta página foi intencionalmente deixada em branco

Agradecimentos

Aos meus orientadores Professor Doutor Carlos Fernando Couceiro de Sousa Neves, Professor Doutor Hugo Filipe Costelha de Castro e Professor Doutor Luís Manuel Conde Bento, pelo acompanhamento ao longo do meu percurso académico, pelo desafio proposto, pelas sugestões, pelas ajudas nas dúvidas que foram surgindo e, principalmente, pelo encorajamento que foram transmitindo ao longo da realização da dissertação.

Aos meus pais José Martins e Licínia Martins, pelo apoio incondicional tanto agora como em toda a minha vida, pela motivação de seguir os meus estudos, oportunidade que nunca lhes foi dada e que felizmente me puderam proporcionar.

À minha namorada Diana Fernandes, que me acompanhou nesta aventura desde o primeiro dia, partilhando momentos de felicidade e tristeza, sempre com palavras de ânimo e incentivo. Sem dúvida, um dos pilares mais importantes nos desafios diários que foram surgindo.

Aos meus colegas que iniciaram esta etapa comigo e que me acompanharam nas lutas do dia-a-dia, pela partilha de conhecimento e opiniões, sempre com o sentimento de companheirismo e motivação.

À Escola Superior de Tecnologia e Gestão do Politécnico de Leiria e ao grupo de investigação de Robótica Avançada e fábricas Inteligentes (ROBiTECH), pelos meios, condições e recursos disponibilizados para que fosse possível a realização desta dissertação.

Esta página foi intencionalmente deixada em branco

Resumo

Neste documento é apresentado um projeto de investigação e desenvolvimento de um algoritmo de deteção de linhas com recurso a visão computacional, através de técnicas clássicas (i.e., sem recurso a técnicas de aprendizagem automática), aplicado à deteção de linhas delimitadoras de vias de trânsito no contexto da competição de condução autónoma no Festival Nacional de Robótica, aplicado num veículo de condução autónoma (iTruck) à escala 1:10 em ambiente altamente estruturado.

Dada a necessidade de aumentar o campo de visão, fruto da anterior participação no Festival, fizeram-se melhorias na câmara frontal do iTruck que levou ao estudo de diversos métodos de calibração para câmaras com elevado campo de visão. Os desenvolvimentos efetuados neste domínio resultaram na publicação de um artigo científico e respetivo software associado: *P.F.Martins, H.Costelha, L.C.Bento and C.Neves, "Monocular Camera Calibration for Autonomous Driving — a comparative study," IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), 2020*.

No âmbito da deteção de linhas delimitadoras de vias de trânsito no contexto da competição de condução autónoma no Festival Nacional de Robótica, foram pesquisadas diversas soluções disponíveis na comunidade científica com o objetivo de servir de base para um novo desenvolvimento. Durante este processo testaram-se várias técnicas e algoritmos, de forma isolada, para extrair uma avaliação de desempenho no caso em estudo.

O algoritmo final é resultado do desenvolvimento de 3 algoritmos criados especificamente para o cenário da competição de modo a aumentar a redundância e confiança da deteção. Cada um dos algoritmos foi desenvolvido com base nas técnicas avaliadas e otimizado consoante os seus resultados. Entre estas encontram-se técnicas baseadas na transformada de Hough, detetor de bordos Canny e caixas deslizantes. A sua agregação segue uma determinada ordem, por forma a aumentar a sua eficácia, e devolvem um conjunto de equações polinomiais de segunda ordem em cada uma das linha detetadas. A decisão final é tomada em duas etapas: tendo em conta o grau de confiança de cada algoritmo e pela validação dos resultados consoante as hipóteses dos algoritmos paralelos.

Os testes de validação realizados *offline* com base em dados reais, demonstram o correto funcionamento do sistema final. A combinação dos algoritmos colmata as falhas e pontos

fracos de cada um, permitindo desta forma uma correta deteção em todas as zonas da pista. No futuro deverá ser melhorado o algoritmo de decisão e controlo e, consequentemente, efetuados testes dinâmicos no veículo em tempo real para validar o comportamento global.

Palavras-chave: Condução Autónoma, Métodos Clássicos de Visão Computacional, Calibração da Imagem, Transformação de Perspetiva, Pré-processamento de Imagem, Deteção de Linhas.

Abstract

The present document describes a research and development project about an algorithm for lane detection, using classical approaches (that is, without recurring to machine learning techniques), for a 1:10 scale autonomous driving vehicle with the aim to compete in the Portuguese National Robotics Festival.

To increase the field of view on the vehicle's front camera, a new lens was used. This led to a study about calibration methods for cameras with a wide field of view, resulting in an paper untitled *Monocular Camera Calibration for Autonomous Driving - a comparative case study*. The source code, also including a perspective transformation, was made publicly available.

With the road lane detection on scope, on Portuguese National Robotics Festival context, many options were taken into consideration to be the base of new development. During this process, were tested many standalone techniques and algorithms, to extract a performance level.

The final algorithm is a result of the development of 3 algorithms, created specifically for the competition scenario, aiming for increased redundancy and certainty. Each one of the three algorithms was developed based on the evaluated techniques and optimized according to their results. These techniques were based on Hough Transform, Canny Edge Detector and Sliding boxes. Their aggregation follows a particular order to increase their effectiveness and return a set of second order polynomial equations to each line detected. The final decision is taken in two steps, considering the degree of confidence of each algorithm, and validating the results by comparing the algorithms' outputs.

Validation tests were performed offline based on real data, showcasing the correctness of the final system. The combined system overcomes the flaws and weaknesses of each one, thus allowing a correct detection in all areas of the track. In the future, the decision and control algorithm will need to be improved and, consequently, dynamic tests will be carried out to validate the global system behaviour.

Keywords: Autonomous Driving, Computer Vision Classical Methods, Image Calibration, Perspective Transformation, Image Pre-processing, Lane Detection.

Índice

Lista de Figuras

Lista de siglas e acrónimos

ABS	*Anti lock Braking System*
ADAS	*Advanced Driver-Assistance Systems*
BEV	*Bird's Eye View*
CRPL	Clube de Robótica do Politécnico de Leiria
ESTG	Escola Superior de Tecnologia e Gestão
FNR	Festival Nacional de Robótica
FOV	*Field of View*
ICARSC	*International Conference on Autonomous Robot Systems and Competitions*
INE	Instituto Nacional de Estatística
MPC	*Model Predictive Control*
ML	*Machine Learning*
NHTSA	*National Highway Traffic Safety Administration*
OPENCV	*Open-Source Computer Vision*
RANSAC	*RANdom SAmple Consensus*
ROS	*Robotic Operating System*
SAE	*Society of Automotive Engineers*
V-SLAM	*Visual Simultaneous Localization and Mapping*

Esta página foi intencionalmente deixada em branco

1. Introdução

Pela primeira vez, desde 1908 com o Model T de Henry Ford, a indústria automóvel está a sofrer uma grande alteração, uma mudança de paradigma, com a condução autónoma e a eletrificação dos sistemas de propulsão [1]. Por um lado, numa vertente ambiental, o abandono dos combustíveis fósseis está a dar lugar às energias limpas e renováveis, tópico que passa muito pela propulsão elétrica com desenvolvimento das baterias ou sistemas de geração de energia elétrica a bordo, como a "*fuelcell*". Por outro lado, e foco desta dissertação, a remoção do componente menos fiável do veículo "o condutor" com o surgimento da condução autónoma. Hoje em dia, uma das áreas com maior foco a nível de investigação na indústria automóvel, com fim de reduzir a sinistralidade nas estradas. Além disto, pretende melhorar a qualidade de vida, reduzindo o stress do trânsito, diminuindo a poluição, aumentando o tempo de lazer, a eficiência dos transportes e permitindo a deslocação de pessoas de mobilidade reduzida, entre outros fatores.

Esta dissertação surge no seguimento das duas transformações da indústria automóvel referidas e no âmbito da competição de condução autónoma integrada no Festival Nacional de Robótica, na qual o clube de robótica do Politécnico de Leiria participa há vários anos. A competição consiste no desenvolvimento/adaptação de veículos à escala para executar diversas funções de condução autónoma, tais como: condução na faixa de rodagem, manobras de estacionamento em paralelo e perpendicular, deteção de sinais verticais, evasão de obstáculos e até condução através de faixas de rodagem provisórias, e.g. faixa de rodagem provisória resultado de zonas em obras (*vide* Figura 1).

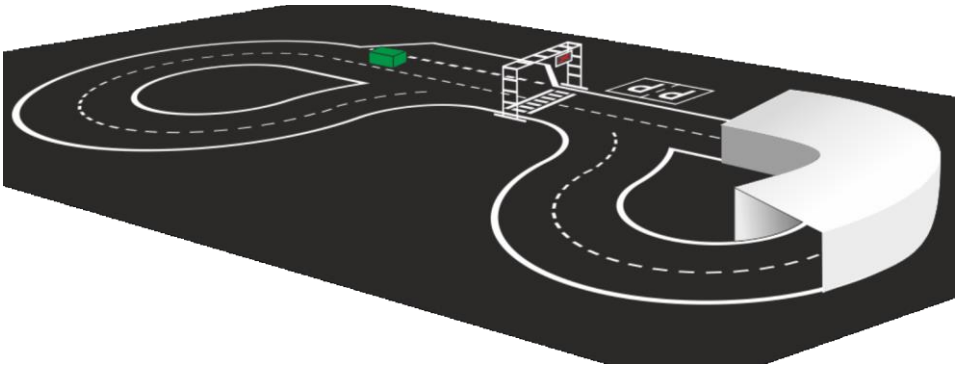


Figura 1 - Pista de condução autónoma do Festival Nacional de Robótica [2].

1.1. Objetivos

Após a última participação do Clube de Robótica do Politécnico de Leiria (CRPL) no Festival Nacional de Robótica (FNR) na edição de 2019 em Gondomar, a equipa fez um levantamento de problemas do iTruck e da participação em geral. Nessa lista, um dos tópicos estava relacionado com a melhoria do sistema de alto nível, ou seja, com a melhoria dos algoritmos de perceção e interpretação (visão computacional), assim como dos algoritmos de controlo e navegação do iTruck. Selecionou-se como prioridade o desenvolvimento de um novo sistema de deteção de faixa de rodagem, um algoritmo que, ao contrário do já desenvolvido, permitisse identificar a faixa de rodagem de uma forma mais robusta e com um alcance frontal superior. Este alcance deveria permitir controlar o iTruck, dinamicamente, de forma mais suave, podendo adaptar a velocidade do mesmo conforme os cenários que se aproximassem e, assim, prever em vez de reagir. Além disto, e no seguimento da lista mencionada anteriormente, pretendia-se também desenvolver um sistema de deteção e interpretação dos sinais verticais presentes ao longo da pista.

1.2. Métodos

Atualmente, os sistemas de deteção baseados em visão computacional podem ser desenvolvidos essencialmente por duas abordagens: com base em técnicas clássica ou com base em aprendizagem automática (*machine learning*) e inteligência artificial (IA). Os sistemas baseado em IA centram-se na acumulação de informação e o uso de algoritmos para tentar imitar o modo como os humanos aprendem, melhorando a precisão gradualmente [3]. A maior desvantagem é a menor previsibilidade e identificação da origem de falhas, dada a "caixa negra" por detrás de todos estes sistemas.

Por este motivo, optou-se por seguir uma abordagem clássica, uma abordagem que, baseada por exemplo em análises morfológicas de imagem, estimação de pontos, filtros e segmentações, permitisse explorar e aprofundar alguns dos métodos e técnicas mais utilizados pela comunidade científica, visando uma melhoria ou combinação de algoritmos para obter melhores resultados.

O funcionamento do iTruck está baseado no sistema de ROS (*Robotic Operating System*), um sistema de desenvolvimento de software flexível, que agrupa diversas ferramentas, bibliotecas e convenções, com o objetivo de simplificar a criação de sistemas robóticos complexos [4]. Uma das bibliotecas muito utilizadas pela comunidade ROS, e uma das mais

utilizada para visão computacional, designa-se por OpenCV (*Open Source Computer Vision*), trata-se de uma biblioteca de código em fonte aberta, ou seja, de uso livre e totalmente documentada na qual se baseará o desenvolvimento da componente de visão computacional, em particular a usando a versão 4.1.2, [5].

1.3. Estrutura

A dissertação divide-se em 4 capítulos: introdução, estado da arte, desenvolvimento e conclusão. A Introdução introduz o contexto do projeto, quais são os objetivos a atingir, estrutura do relatório e os métodos utilizados ao longo do desenvolvimento. O Estado da arte aborda a condução autónoma, tema central de todo o projeto, demonstrando quais os principais desafios, problemas, progressos e vantagens. É ainda resumida a situação atual do iTruck de condução autónoma do clube de robótica, explicando a sua estrutura e pontos de melhoria. O Desenvolvimento aborda as diversas etapas do processo, desde a calibração e retificação da imagem captada pela câmara principal do veículo, passando pelas etapas de pré-processamento de imagem recomendados para este tipo de aplicações, pelos algoritmos desenvolvidos e testados, finalizando com os resultados finais. Esta etapa reporta também técnicas, que face aos resultados esperados, tiveram menos sucesso. Por fim, as Conclusões resumem o trabalho desenvolvido, atribuindo observações críticas aos processos, métodos e resultados, finalizando com algumas sugestões de melhoria futura.

Esta página foi intencionalmente deixada em branco

2. Estado da arte

O Estado da arte irá contextualizar o projeto no que diz respeito aos desenvolvimentos na área da condução autónoma, abordando tópicos como os níveis de automação existentes e algumas das principais mudanças que esta pode trazer para a comunidade. Irá também ser apresentado de uma forma resumida o estado do iTruck do clube do Politécnico de Leiria, alvo central de todo este desenvolvimento. Por fim serão expostas algumas das abordagens mais utilizadas na área da deteção de linhas na faixa de rodagem.

2.1. Condução autónoma

Como já referido, a condução autónoma é, atualmente, um dos tópicos de investigação com maior destaque a decorrer nos dias de hoje, que poderá ter um impacto muito relevante no nosso dia-a-dia. O principal objetivo é corrigir um problema que, em 130 anos de história do automóvel, nunca se solucionou: remover o componente mais defeituoso e com maior taxa de falha do veículo, o condutor [6]. Em 2020, mais de 18 800 pessoas morreram em acidentes rodoviários [7] e, segundo o NHTSA [8], 94% dos acidentes são causados por erro humano.

Hoje em dia, já é possível encontrar um conjunto de sistemas eletrónicos de assistência ao condutor em funções de estacionamento e condução, designadas de ADAS. Estes sistemas ADAS consistem efetivamente na introdução de algum grau de autonomia aos veículos substituindo o condutor em circunstâncias bem definidas. Segundo a SAE, Sociedade de Engenheiros Automóveis, existem 6 níveis de condução autónoma, [9].

- O Nível 0 é totalmente manual, não possui qualquer funcionalidade de condução autónoma. Sistemas como o ABS (sistema de travagem anti bloqueio) não são considerados sistema de ajuda à condução, pois necessitam do condutor para atuar.
- O Nível 1 é, assim, o nível mais baixo de automação, onde se encontram sistemas como *cruise control*, que mantêm a velocidade do veículo constante sem necessidade de pressionar o acelerador. Todos os outros comandos (ex. direção e travagem) são executados pelo condutor.
- O Nível 2 introduz o sistema ADAS, sistema avançado de assistência ao condutor. São funções como, *cruise control* adaptativo, assistência à faixa de rodagem, estacionamento automático, sistemas anticolisão, entre outros, que dão início a um

nível de automação parcial. O condutor já não necessita de atuar nenhum dos 3 elementos principais (acelerador, travão e direção), no entanto, é obrigado a manter o foco na estrada e posição para assumir o controlo a qualquer momento.

- O Nível 3 é notório do ponto de vista tecnológico e subtil do ponto de vista do condutor. Aqui o veículo irá conseguir assumir o controlo na maioria dos ambientes, no entanto, o condutor ainda terá que manter o foco e a prontidão para intervir, se necessário.
- O Nível 4 já é um estado muito avançado de automação. Aqui o condutor apenas terá que intervir em determinadas situações mais pontuais, com a sua atenção à estrada e ao ambiente em redor a ser opcional.
- O Nível 5 representa a estado de condução autónoma pura. O veículo não necessita da intervenção do condutor em nenhuma situação, sendo que o veículo pode até nem possuir comandos para o controlo do mesmo por parte do humano.

Além do problema da sinistralidade, esta revolução poderá mudar o modo como as pessoas de deslocam no seu dia-a-dia. Segundo o INE, em 2017, na área metropolitana do Porto e de Lisboa, a população móvel deslocou-se cerca de 2,66 vezes por dia com uma duração média de 23,05 minutos [10]. Estes números fazem com que Portugal siga a tendência mundial e se possa afirmar que um veículo, em média, se encontre mais de 95% do seu tempo estacionado. A condução autónoma irá certamente alterar este paradigma, fazendo com que a posse de um veículo deixe de ser fundamental, e o aluguer de veículos autónomos através de plataformas de serviços eletrónicos na área do transporte privado (exemplo: UBER e BOLT), numa política de *carsharing*, (compartilhamento de veículo entre utilizadores), seja a opção preferencial da maioria das pessoas. Consequentemente, o trânsito, a disposição das cidades, o ambiente e o estilo de vida das pessoas, serão algumas das coisas que irão sofrer alterações significativas com esta mudança [11]. A questão passará a ser a aceitação desta mudança por parte da população, no entanto, tendo em conta rápida a adoção da tecnologia nos dias de hoje (*vide* Figura 2), prevê-se que a condução autónoma não seja uma exceção a esta tendência.

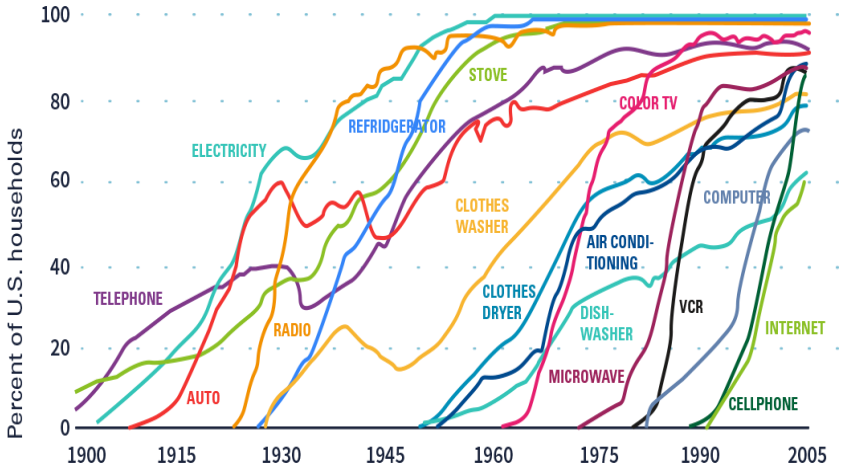


Figura 2 - Adoção da tecnologia [12].

Assim sendo, a sua concretização apenas dependerá do avanço da tecnologia que, segundo a Figura 3, e seguindo a mesma linha de raciocínio, atingirá o nível 5 nas próximas décadas.

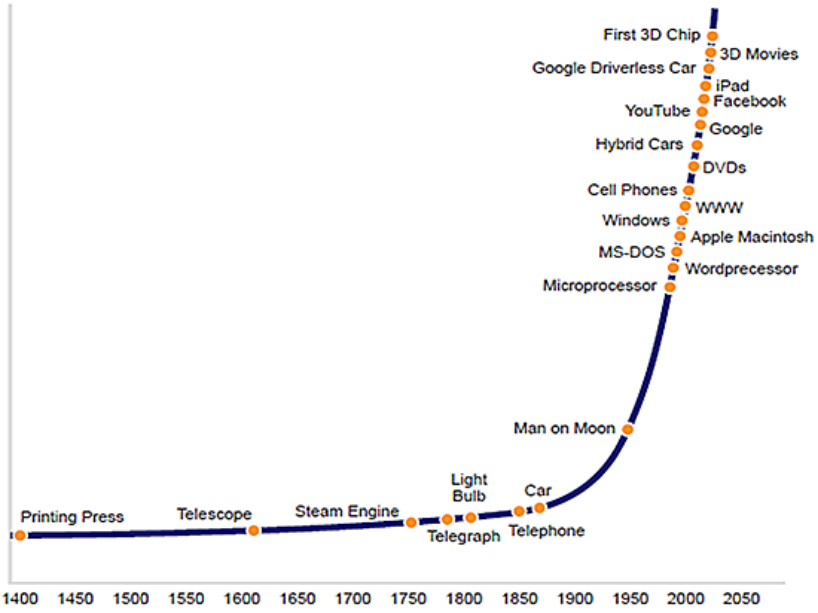


Figura 3 - Avanço da tecnologia [13].

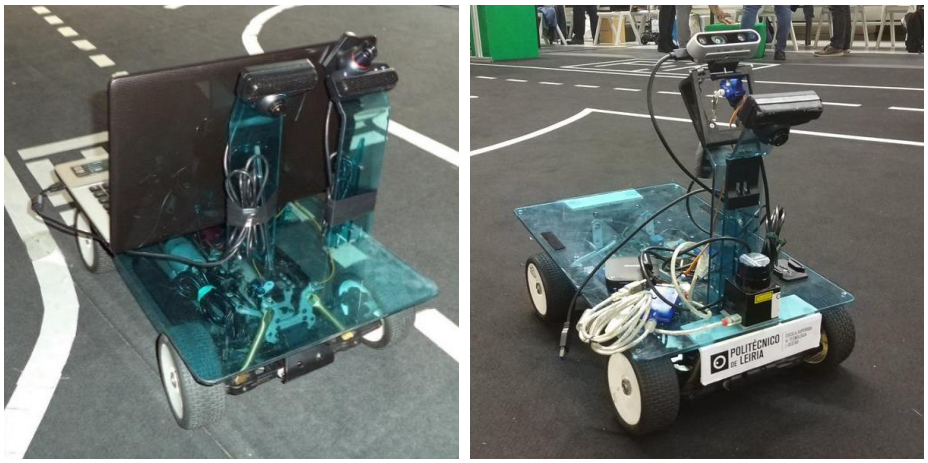
2.2. Veículo do clube de robótica do Politécnico de Leiria

Com o intuito de envolver a comunidade académica nesta pesquisa global, a Sociedade Portuguesa de Robótica[1] decidiu introduzir uma modalidade de condução autónoma no Festival Nacional de Robótica, um evento internacional de competições e ciência na área da robótica, que decorre em Portugal desde 2001. Como tal, o clube de robótica aceitou o

[1] http://www.sprobotica.pt/

desafio e participa nesta competição há vários anos, tendo já atingido o título de campeão por duas vezes, em 2014 e 2015.

O veículo usado na competição tem por base um carro de rádio modelismo à escala de 1:10 adaptado ao longo dos anos para esta mesma função.



a) **b)**

Figura 4 - Veículo de condução autónoma do Clube de Robótica: a) Versão 2015 - b) Versão 2019.

Após a participação em 2019, onde foi alcançado o 3° Lugar, identificaram-se alguns pontos de melhoria: rigidez estrutural, atenuação de vibrações, aderência, campo de visão, perceção e o controlo do veículo. Os primeiros 3 tópicos evoluíram-se sob um projeto de final de curso de licenciatura de Engenharia Automóvel, onde foi desenvolvido uma estrutura em acrílico para suportar o hardware do veículo, como mostra a Figura 5.

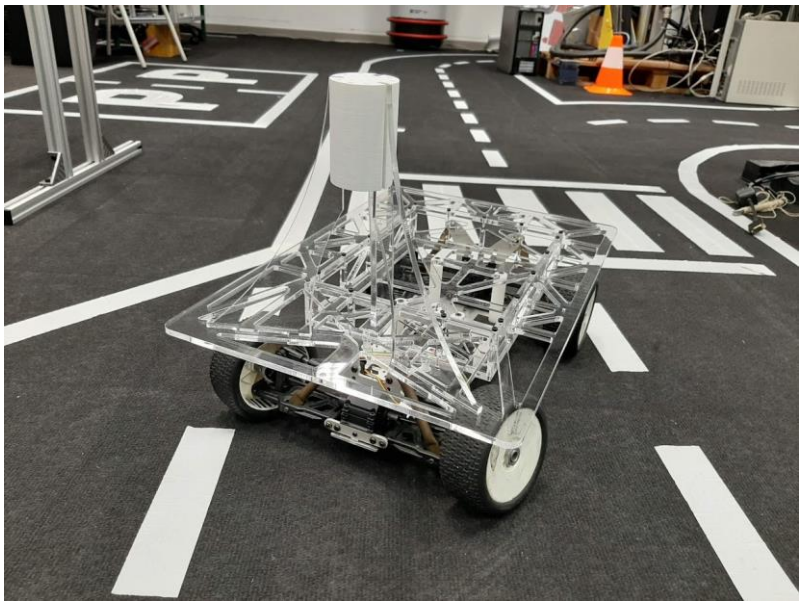


Figura 5 - Melhoria estrutural do veículo em 2020.

Com a base do veículo remodelada, decidiu-se melhorar também o aspeto visual e, como tal, foi nesse ano impresso em 3D todo um corpo exterior para dar a aparência de um camião. Este corpo, além de melhorar a aparência do veículo, tinha também o objetivo de proteger alguns componentes como as câmaras, suportes e até a unidade de processamento a bordo, i.e. um computador portátil. Para reduzir danos em caso de impacto, foi também adicionado um para-choques recorrendo a um sistemas de molas para efetuar o amortecimento de impactos. Na Figura 6 é possível visualizar a versão atual do veículo de condução autónoma do clube de robótica com todas estas alterações integradas.

Figura 6 - Veículo de condução autónoma do Clube de Robótica, assinalado com um círculo azul encontra-se a câmara frontal.

O nível de arquitetura global, um veículo para fins de condução autónoma necessita de desenvolver 3 áreas distintas: hardware, software e periféricos. O hardware responsabiliza-se por recolher a informação de sensores, garantir o seu processamento e comunicar com os atuadores. O software consiste na implementação dos algoritmos que, com base na informação dos sensores, identificam as várias características dos elementos externos para tomar decisões e exercer controlo sobre atuadores de forma a manobrar o veículo de acordo com estas decisões. Por fim, nos periféricos incluem-se todos os componentes sensoriais responsáveis por recolher informação do ambiente em redor e do próprio veículo. Sendo este um projeto para deteção das linhas da faixa de rodagem, o sensor mais relevante é a câmara frontal, assinalada a azul na Figura 6. Esta é responsável pela captação frontal da informação visual do veículo e resulta do processo de adaptação da câmara de uma PlayStation 3.

2.3. Deteção de faixa de rodagem

Existem já inúmeras abordagens e soluções testadas ao longo dos últimos anos, no entanto, este projeto foca-se na pesquisa nos métodos clássicos.

Um algoritmo de deteção de faixas de rodagem é formado por várias etapas que, de uma forma geral, seguem a ordem apresentada na Figura 7. Começam pelo pré-processamento de imagem, onde se executam algumas tarefas de tratamento, de seguida passam para a extração de características, onde o principal objetivo é identificar as linhas e faixa de rodagem. Posto isto, é realizada uma aproximação de um modelo às linhas detetadas, finalizando com a validação dos resultados tendo em conta o histórico no domínio do tempo.

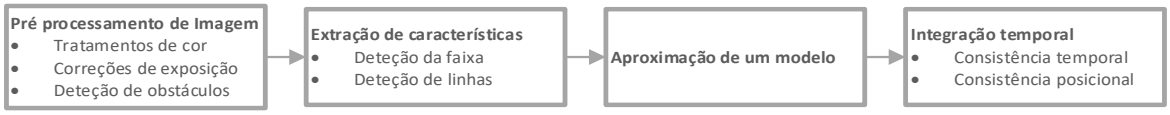
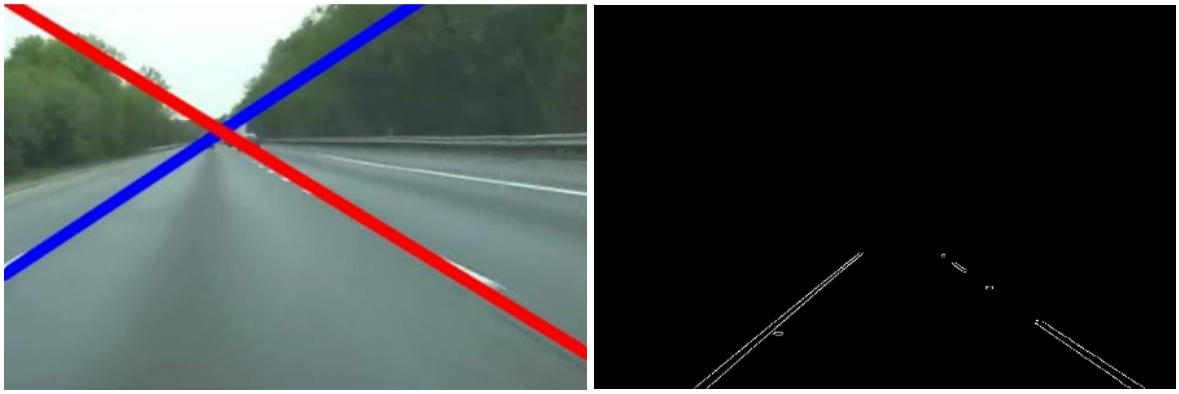


Figura 7 – Etapas para a elaboração de um algoritmo de deteção de faixas de rodagem [14].

Segundo, Mandlik e Deshmukh, [15], dois dos algoritmos mais utilizados que servem de base à deteção de faixas de rodagem, são a transformada de *Hough* e a deteção de bordos *Canny*, como mostram a Figura 8 a) e b), respetivamente. Estas técnicas são de baixa complexidade, mas com inúmeras aplicações, sendo utilizadas em diversos problemas na área de visão computacional.

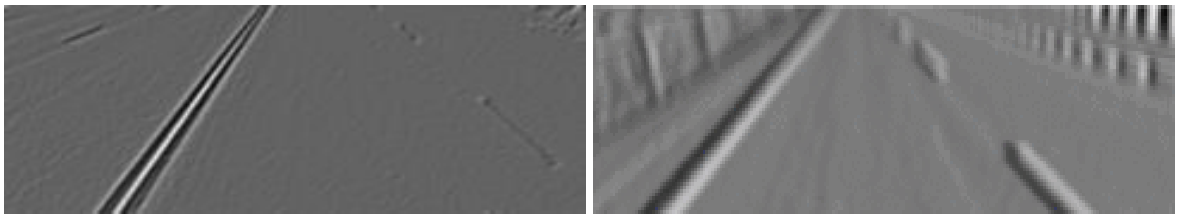


a) b)

Figura 8 - Algoritmos mais utilizados: a) Transformada de Hough; b) Detetor Canny.

Existe também um conjunto de abordagens que, através de filtros avançados, conseguem extrair diversas características das linhas. Filtros gaussianos direcionáveis e *HaarLike Features* duas das técnicas consideradas mais relevantes por Mandlik e Deshmukh,

representados visualmente na Figura 9 a) e b). Servem principalmente de técnicas auxiliares para os algoritmos principais.



a) b)

Figura 9 - Filtros: a) Filtros direcionais [16]; b) HaarLike Filter [17].

Técnicas de deteção através de alinhamento espaço-tempo e RANSAC são alguns exemplos de algoritmos mais avançados usados na deteção que combinam diversos métodos e abordagens por forma a conseguir retornar as linhas da faixa de rodagem, Figura 10.

a) b)

Figura 10 - Técnicas avançadas: a) Alinhamento espaço-tempo [18] b) RANSAC [19].

Uma das soluções finais bastante utilizadas pela comunidade científica é o método das caixas deslizantes, um algoritmo robusto que se baseia na colocação de caixas de procura, tendencialmente ao longo das linhas, procurando sempre seguir a trajetória correta. Este algoritmo é bastante flexível e permite fazer ainda diversas melhorias face ao contexto em questão, podendo ser otimizado para faixas com raios de curvatura mais apertados e diferentes tipos de linhas [20].

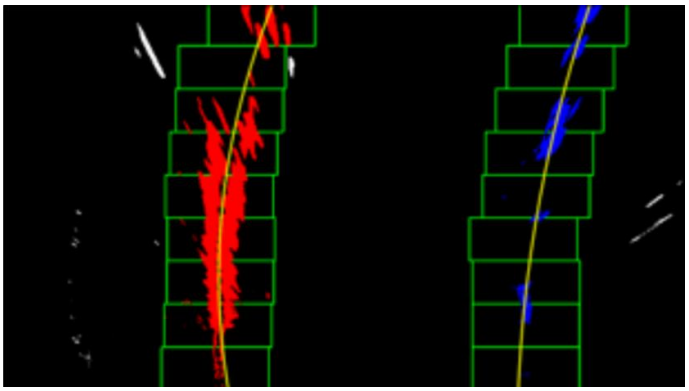


Figura 11 - Caixas deslizantes [20].

Ainda assim, os resultados são bastante condicionados na utilização desta técnica, **i.e.** qualquer imperfeição na estrada como sombras, linhas desvanecidas, linhas ocultas por outros veículos ou até objetos, podem levar a deteções erradas. Algoritmos com inteligência artificial, tendem a extrair melhores resultados dada a sua aprendizagem e adaptação constante. Prova disto é o seu uso disseminado em soluções académicas [21], a Figura 12 apresenta a correta deteção das linhas nas múltiplas faixas de rodagem mesmo tendo veículos a ocultá-las parcialmente.

Figura 12 - Deteção com inteligência artificial [21].

Por fim, todos necessitam de um algoritmo de controlo e, na condução autónoma, o sistema de controlo tem que prever comportamentos dinâmicos e responder e adaptar consoante diversos parâmetros de entrada. Uma abordagem possível e muito utilizada nesta aplicação é o *Model Predictive Control* (MPC) [22].

3. Desenvolvimento

Este tópico irá abordar alguns dos métodos clássicos mais utilizados para processamento de imagem no âmbito da condução autónoma. O objetivo é detetar da faixa de rodagem testando diversas técnicas e combinando os melhores algoritmos estudados. Numa primeira fase irá ser abordada a retificação e transformação de perspetiva de uma imagem, passando posteriormente para pré-tratamentos e algoritmos isolados, finalizando com a combinação dos melhores resultados.

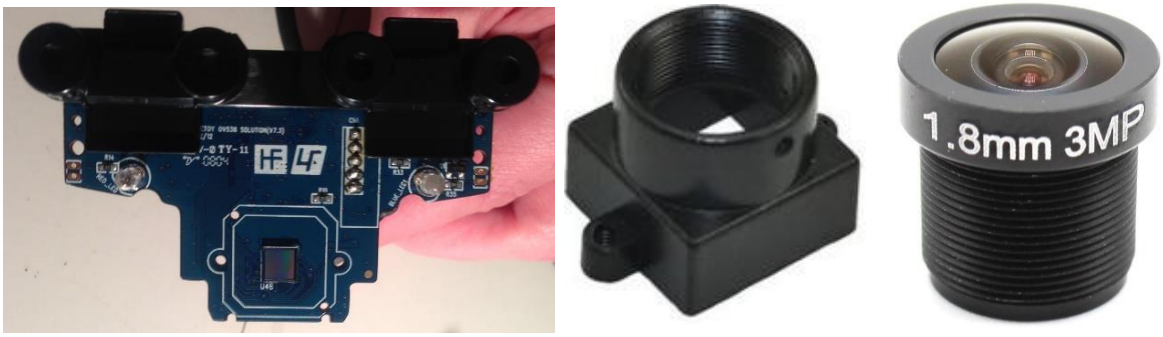
3.1. Calibração/Remoção da distorção da imagem

Esta secção descreve uma melhoria na câmara frontal do veículo. A participação anterior na competição de condução autónoma no Festival Nacional de Robótica revelou a necessidade de aumentar o campo de visão, e neste sentido fizeram-se melhorias na câmara frontal do iTruck que levaram ao estudo de diversos métodos de calibração para câmaras com elevado campo de visão. Os desenvolvimentos efetuados neste domínio resultaram na publicação de um artigo científico e respetivo software associado: *P.F.Martins, H.Costelha, L.C.Bento and C.Neves, "Monocular Camera Calibration for Autonomous Driving — a comparative study," IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), 2020.*

Uma vez que a competição se realiza numa pista, ainda que à escala, bastante pequena, as curvas são consideravelmente apertadas. Em alguns pontos, junto da passadeira por exemplo, o campo de visão de 75º do veículo não permitia recolher informação da faixa de rodagem divergente, Figura 13. Numa situação real, numa mudança de direção, os veículos sofrem do mesmo problema e como tal, era essencial ampliar o campo de visão.

Figura 13 - Campo de visão (Versão 2019).

Existem 3 soluções para ampliar o campo de visão da câmara do veículo: aumentar o número de câmaras com diferentes orientações e posições no veículo, alterar a câmara por completo, e escolher uma que já possua originalmente um FOV (*Field Of View* – Campo de Visão) superior, ou adaptar uma lente com um FOV superior ao da câmara atual, Figura 14. Por questões de orçamento optou-se por adquirir um conjunto de lentes de baixo custo e que fossem compatíveis com o suporte do sensor de captação da *PS3EYE* (Câmara da PlayStation 3) para executar alguns testes de desempenho preliminares.



a) b) c)

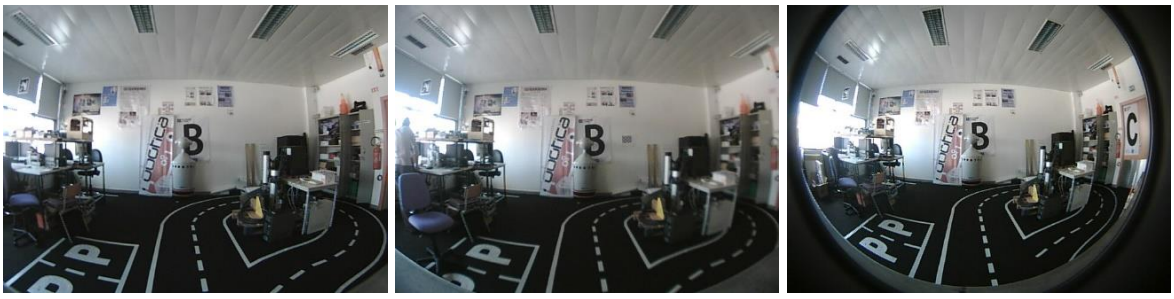
Figura 14 - Componentes da câmara: a) sensor de captação b) suporte da lente c) lente da câmara.

Esta opção, ao aumentar o campo de visão, compromete a qualidade da imagem, pois, a quantidade de informação será maior e a densidade de pixéis no sensor não se irá alterar, fazendo com que a imagem reduza a sua nitidez. Ainda assim, procuraram-se várias lentes que fossem compatíveis e tivessem um FOV superior ao da câmara original. Adquiriram-se três lentes com diferentes tipologias, duas *wideangle* (grandes angulares) e uma *fisheye* (olho-de-peixe), ambas com campos de visão superiores ao da câmara/lente original.

Durante a realização de testes preliminares, começaram a surgir duvidas quanto à veracidade do campo de visão anunciado pelo fabricante e da compatibilidade entre o suporte da lente, o sensor e a própria lente. Por esse motivo, foram executados alguns testes para medir o FOV e testar a compatibilidade dos componentes, como mostra a Figura 15.

Figura 15 - Modificações das lentes.

Esta medição consistiu na fixação do sensor a uma estrutura de alumínio estática e, sem alterar a sua posição, recolher várias fotografias com as diferentes lentes, Figura 16.



a) b) c)

Figura 16 - Campo de visão das três lentes: a) 108º (anunciado 120º); b) 110º (anunciado 120º) e c) 176º (anunciado 180º).

Com base em referências geométricas do ambiente em redor, foi possível concluir que as lentes não estavam a produzir os resultados enunciados na folha de especificações, na realidade as lentes possuíam os seguintes FOV:

a) 108º (anunciado 120º)
b) 110º (anunciado 120º)
c) 176º (anunciado 180º)

Apesar destas medições poderem conter algum erro associado, claramente os campos de visão não eram correspondentes com a folha de especificações. Além disso, devido à fraca

qualidade de produção e defeitos do próprio produto, encontraram-se incompatibilidades entre componentes. Estas falhas provocaram desalinhamentos das lentes relativamente ao sensor, muito evidentes na Figura 16 c), com o aparecimento não uniforme do corpo da lente ao longo dos bordos. Dada a semelhança entre os resultados a) e b), os testes focaram-se na comparação destas duas lentes, nomeadamente a lente b) com a lente c).

Apesar dos 110° de FOV da lente b) serem superiores aos 75° da câmara original, estes ainda são insuficientes para uma deteção numa zona de cruzamento. Por este motivo, optou-se pela utilização da lente c), que apresenta um campo de visão muito próximo dos 180°, não obstante apresentar uma distorção bastante acentuada e o desalinhamento ainda ser relevante. As "sombras" observadas nas extremidades da imagem são o corpo da própria lente que, devido à sua incompatibilidade com o sensor em questão, permite a visualização interna do corpo da lente. A imagem não poderia ser processada no seu estado original e, como tal, é necessário remover a distorção e estas zonas internas do corpo.

Todos os métodos, modelos e processos de calibração foram descritos com maior detalhe no artigo mencionado no Anexo A, além dos vários testes com as outras 2 lentes. Como tal não se irão aprofundar estes procedimentos nesta secção. Muito resumidamente, a calibração de uma câmara passa pela caracterização dos parâmetros intrínsecos por forma a remover a distorção. Tendo por base o princípio da câmara estenopeica (Figura 17), a calibração pode usar um conjunto de modelos de calibração consoante a tipologia da lente: normais, grandes angulares, teleobjetivas, macro, olho de peixe, entre outras.

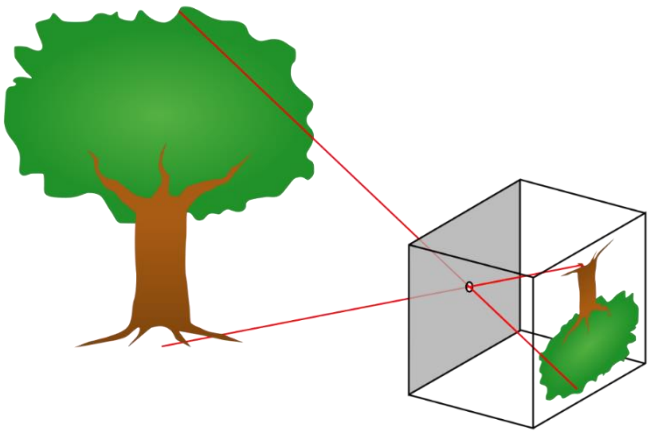


Figura 17 - Princípio da câmara estenopeica (Pinhole Camera Model) [23].

A calibração da imagem, principalmente nos casos onde a distorção é elevada devido ao maior campo de visão, deve ser o primeiro passo a tomar diretamente da imagem "virgem" proveniente da câmara sem nenhum tratamento prévio.

Após todos os modelos testados e ensaios efetuados, descritos no artigo publicado e exposto no Anexo A, a calibração usou o método *rational*, o resultado e da calibração encontra-se na Figura 18.

a) **b)**

Figura 18 - Retificação da imagem virgem.

Como se pode observar, o principal objetivo deste passo é remover a distorção gerada pela lente, e retornar uma imagem o mais próxima da realidade possível. No entanto, devido à baixa qualidade e à incompatibilidade com o sensor da câmara, ainda se mantêm alguns artefactos, como a zona negra apresentada no topo da imagem retificada. Esta zona advém da falta de informação dessa área e como tal é preenchida com pixéis pretos. É possível remover a zona mencionada, no entanto, isso também implicaria a perda de informação na imagem resultando numa área de trabalho menor e consequentemente numa resolução menor. Em todo o caso, concluiu-se que, para o fim proposto, os resultados obtidos são bastante satisfatórios.

3.2. Transformação de Perspetiva

O segundo passo mais comum no processamento de imagem, é a transformação para uma "vista de cima", técnica bastante útil para minimizar o efeito de perspetiva. Este efeito provoca mudanças geométricas significativas nos elementos a detetar ao longo da imagem, como a espessura das linhas. Como se pode comprovar na Figura 19, quanto mais afastados do veículo estes elementos tiverem, mais evidente se torna esta deformação.

Figura 19 - Efeito de perspetiva [24].

O processo de transformação de perspetiva pode também ser designado como homografia, e consiste na definição de uma matriz transformação, por forma a mapear os pontos de uma imagem para um referencial diferente. Neste contexto em particular, é simular o posicionamento virtual da câmara sobre o objeto do qual se pretende uma "vista de cima" e, para tal, esta transposição necessita de resolver duas questões: Onde estou? Para onde quero ir?

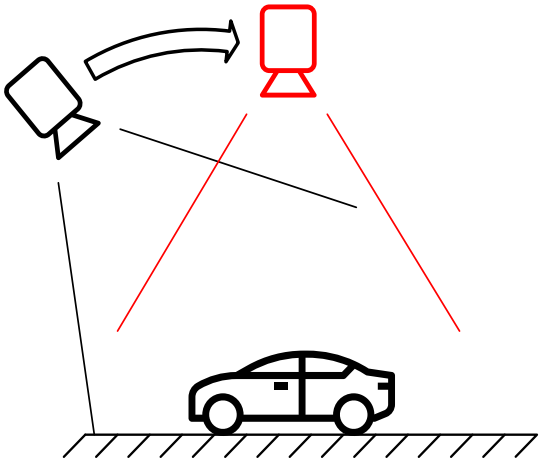


Figura 20 - Posicionamento da câmara virtual para transformação de perspetiva.

O processo de transformação, à semelhança da calibração, usa um xadrez para servir de referência, através dos vértices assinalados com circunferências (*vide* Figura 21).

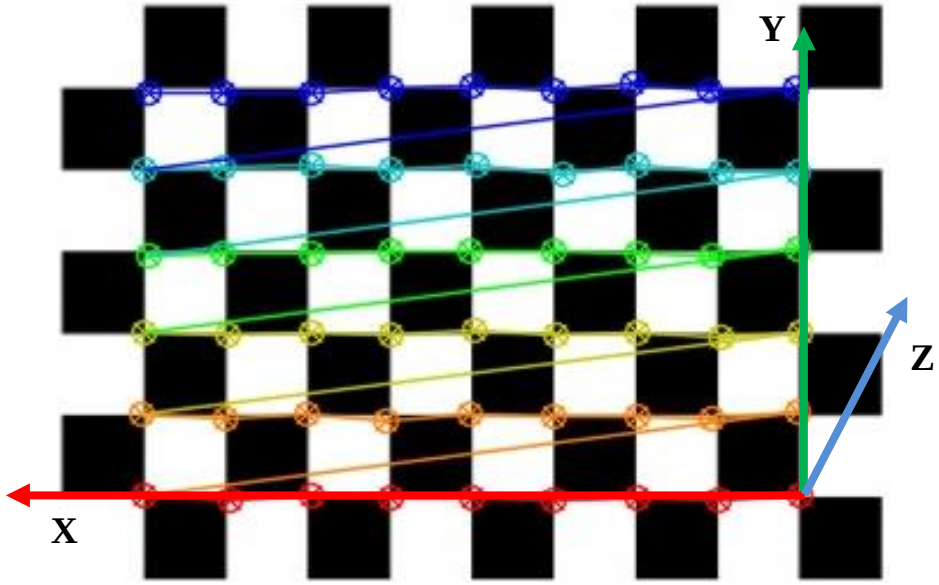


Figura 21 - Xadrez de calibração [25].

Com esta lista de pontos é feita uma estimação da posição relativa ao objeto, adquirindo o vetor de translação e rotação, com auxílio da função *solvePnP* do OpenCV.

Tal como se encontra descrito no artigo em Anexo A, a transposição dos pontos 3D do mundo para o plano 2D do sensor da câmara pode ser definido pela Equação (1).

$$s\begin{bmatrix} u \\ v \\ 0 \end{bmatrix} = \overbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}^{K} \overbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}}^{{}^{c}M_0} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} \quad (1)$$

Assim, tendo em consideração que os parâmetros intrínsecos da câmara se mantêm iguais, a transposição entre os referenciais da câmara e do mundo podem ser representados pela Equação (2).

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = {}^{c}M_0 \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} \quad (2)$$

Consequentemente, a matriz de transformação de pontos do referencial de uma câmara (C1) para o referencial de outra câmara (C2) expressa-se segundo a Equação (3).

$$
\begin{aligned}
{}^{c_2}M_{C_1} &= {}^{c_2}M_0 \cdot {}^{0}M_{C_1} = {}^{c_2}M_0 \cdot \left({}^{C_1}M_0\right)^{-1} = \\
&= \begin{bmatrix} {}^{c_2}R_0 & {}^{c_2}t_0 \\ 0_{3\times1} & 1 \end{bmatrix} \begin{bmatrix} {}^{c_1}R_0{}^{T} & -{}^{c_1}R_0{}^{T} \cdot {}^{c_1}t_0 \\ 0_{3\times1} & 1 \end{bmatrix}
\end{aligned} \tag{3}
$$

Neste caso, os parâmetros colocados no vetor de translação e rotação da câmara C2 são os responsáveis por ditar o posicionamento virtual da câmara no referencial do xadrez. Ou seja, para obter a "vista de cima" os vetores deverão apresentar a seguinte forma:

$$\vec{r} = \begin{bmatrix} 0 \\ 0 \\ A \end{bmatrix} \; e \; \vec{t} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

O vetor de rotação, para obter a vista perpendicular, só poderá alterar o parâmetro "A" que é responsável pelo ângulo em torno do eixo perpendicular ao xadrez. Já o vetor de translação define a posição relativa da câmara e é totalmente configurável, segundo o referencial da Figura 21.

Visto que a função *solvePnP* devolve apenas vetores, é necessário recorrer à fórmula de Rodrigues, função também disponível no *OpenCV*, para transformar o vetor de rotação em matriz. A equação (4) traduz essa mesma conversão sabendo que θ resulta da normal do vetor e I é a matriz identidade.

$$R = \cos(\theta)\, I + (1 + \cos(\theta))\vec{r}\vec{r}^{T} + \sin(\theta) \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \tag{4}$$

Numa penúltima etapa, a matriz de homografia é dada pela Equação (5), onde n é o vetor normal ao plano do xadrez e d a distância ao mesmo, tendo por base a Figura 22.

$${}^{c_2}H_{C_1} = {}^{c_2}R_{C_1} - \frac{{}^{c_2}t_{C_1} \cdot \vec{n}^{T}}{d} \tag{5}$$

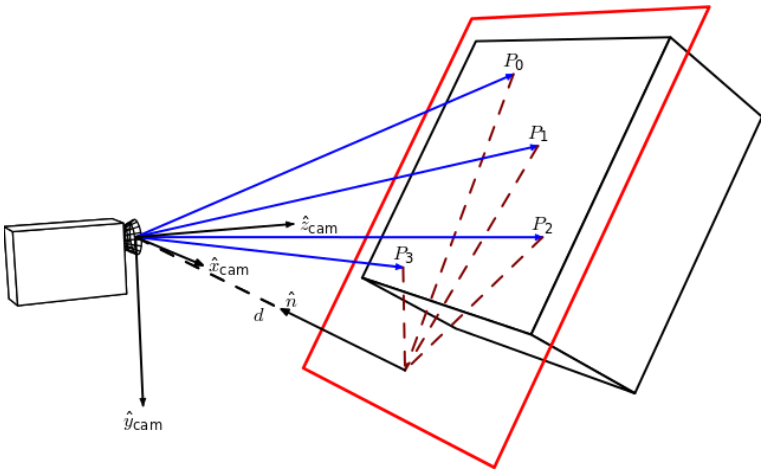


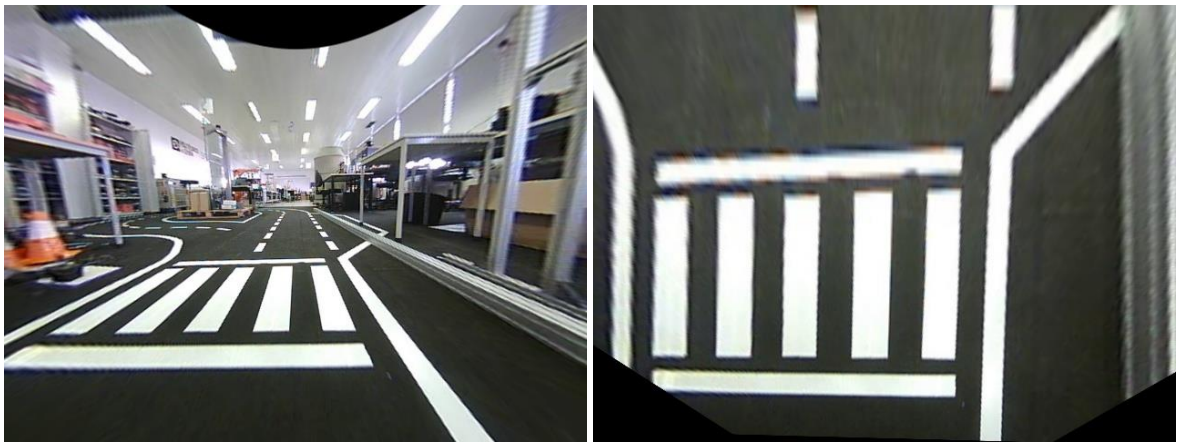
Figura 22 - Estimação de posição através da homografia [26].

Por fim, com auxílio da função *warpPerspective*, é possível aplicar a matriz de homografia calculada nas iterações anteriores. A Figura 23 exemplifica uma mudança da perspetiva com todos os parâmetros de rotação a "0" e uma translação de -150 píxeis em X, -400 em Y e -375 em Z. Nota-se que a pista está ligeiramente desalinhada, pois, o referencial está no xadrez e este não se encontra paralelo à faixa de rodagem.

a) b)

Figura 23 - Mudança de perspetiva com xadrez: a) Vista de perspetiva b) Vista Ortogonal.

Este procedimento apenas necessita de ser realizado uma vez, devendo repetir-se caso a posição da câmara seja alterada relativamente ao veículo. Todos os dados são armazenados para depois poderem ser usados nas imagens recebidas diretamente da câmara em tempo real. A Figura 24 exemplifica o resultado final da transformação para uma vista de topo já afinada, tendo tido o cuidado de alinhar o veículo e o xadrez com a faixa de rodagem.



a) b)

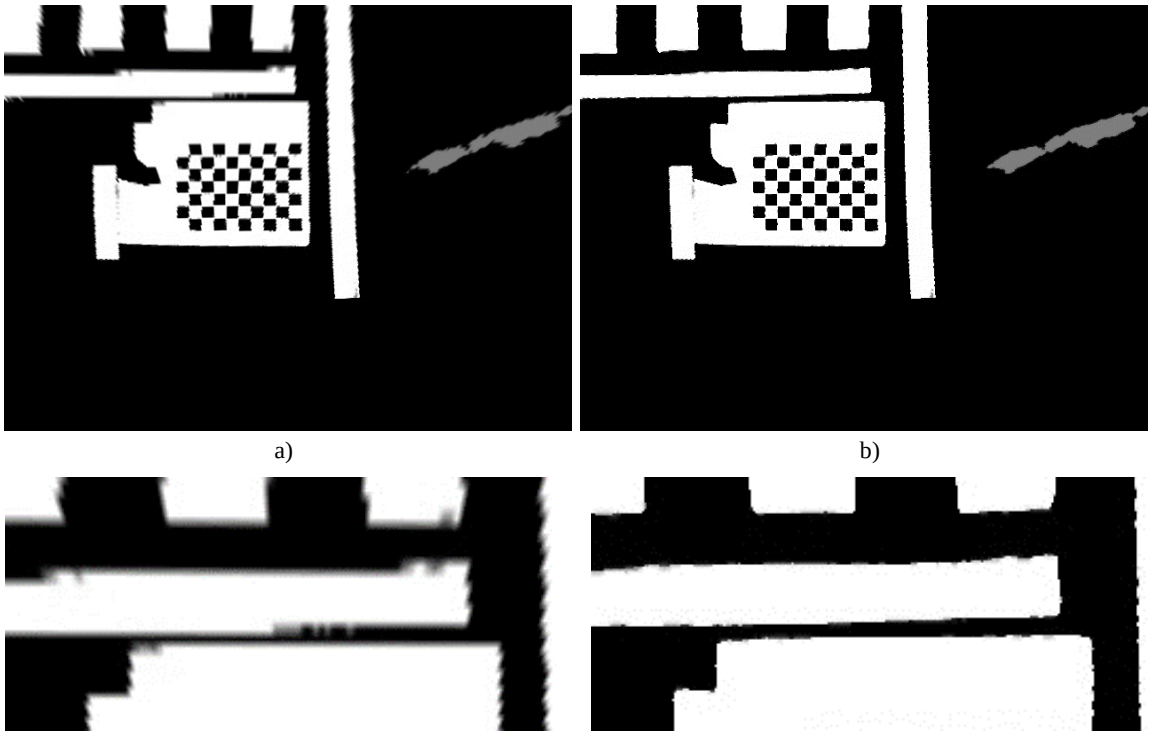
Figura 24 - Correção de perspetiva para BEV: a) Vista em perspetiva b) Vista em BEV.

Por forma a facilitar a utilização deste algoritmo pela comunidade, foi disponibilizado um *Jupyter Notebook* na página do *GitHub* do grupo de robótica do Politécnico de Leiria, *ipleiria-robotics,* com o nome de *MonoCamCalib4AD*[2].

Existem alguns problemas que resultam desta transformação. Em particular, sendo a resolução da câmara um parâmetro fixo, os elementos mais afastados do veículo tendem a perder nitidez. Durante a transformação de perspetiva, devido ao efeito de "ampliação" em certas zonas da imagem, é necessário preencher os espaços vazios. Esses pixéis assumem valores baseados na sua vizinhança, tendo em conta uma interpolação linear. Na Figura 24 b), é possível observar este efeito comparando a linha horizontal da passadeira inferior com a linha horizontal superior, onde existe uma clara perda de nitidez nos objetos mais distantes da câmara.

Qualquer correção de cor, desfoque, binarização ou outro tipo de operação, irá perder eficácia devido ao processo de retificação e de transformação de perspetiva, aumentando os defeitos anteriormente mencionados. Isto é, a imagem final poderá representar detalhes não provenientes da imagem original, mas sim das operações efetuadas. Dada a necessidade, durante os processos de calibração e transformação de perspetiva, de "preencher" alguns pixéis com valores da vizinhança, esses valores podem afastar-se da realidade.

[2] https://github.com/ipleiria-robotics/MonoCamCalib4AD



a) b)

c) d)

Figura 25 - Efeito dos tratamentos antes do processo de calibração e transformação de perspetiva: a) Calibração - Binarização - Transformação de perspetiva, b) Calibração - Transformação de perspetiva – Binarização, c) Ampliação da imagem a), d) Ampliação da imagem b).

A Figura 25 comprova esta alegação, demonstrando a diferença entre duas imagens exatamente iguais, mas com fluxos de processamento diferentes, nomeadamente:

a) Calibração ➔ Binarização ➔ Transformação de perspetiva
b) Calibração ➔ Transformação de perspetiva ➔ Binarização

Nos testes que levaram obtenção das imagens ilustradas na Figura 25, todos os parâmetros de todas as transformações são exatamente iguais, tendo apenas se invertido a ordem com a qual são executadas e, desta forma, evidenciando que a ordem faz diferença, e que se deve evitar qualquer tipo de tratamento *a priori*.

Conclui-se, assim, que qualquer pré-processamento de imagem relativo a cor, filtros ou outros, não deve ser realizado antes da retificação e/ou transformação de perspetiva de imagem, por forma a manter o máximo de qualidade, detalhes e nitidez.

3.3. Pré-processamento de Imagem

Independentemente da abordagem a adotar (métodos clássicos ou métodos baseados em *Machine Learning* (ML) existem alguns pré-processamentos de imagem que poderão

facilitar e melhorar o desempenho dos algoritmos de deteção. Estas etapas são uteis para remover zonas, objetos, detalhes e características indesejáveis ou, por outro lado, realçar as desejáveis, para uma melhor deteção.

3.3.1. Codificações de Cor

Os algoritmos de visão computacional, por forma a simplificar e acelerar o processamento de dados, tendem a tratar imagens monocromáticas, contudo, na condução autónoma, a cor pode trazer segmentações bastantes interessantes. Uma das maneiras de destacar as linhas na faixa de rodagem é aplicando filtros de cor na imagem original. Este filtro tem como o objetivo evidenciar as cores desejadas, neste caso o branco ou amarelo para linhas normais ou temporárias, respetivamente.

Neste sentido, interessa abordar algumas das codificações de cor mais utilizadas (RGB, HSL e HSV). A codificação RGB baseia-se na definição de intensidade para cada uma das cores primárias aditivas (vermelho, verde e o azul). A codificação HSL e a codificação HSV foram criadas tendo como objetivo aproximar da perceção do ser humano, pelo que separam as cores por matiz (graduação da cor), saturação, intensidade e brilho, tudo isto em canais separados, permitindo assim trabalhar cada uma das componentes individualmente.

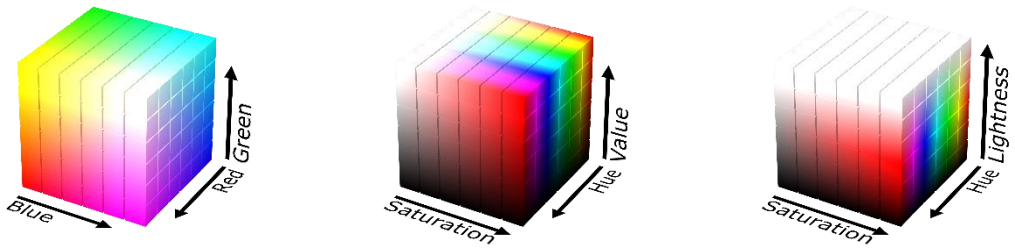


Figura 26 - Codificações de cor [7].

Esta técnica é particularmente útil para faixas de rodagem provisórias que, pela cor e contraste, são bastante evidentes em codificações HSL [8]. Estas codificações podem ser usadas em conjunto com máscaras para separar determinados elementos. Na Figura 27 foi utilizada esta mesma técnica para isolar a linha amarela.

Figura 27 - Codificação HSL para linhas amarelas [9].

No entanto, no caso em estudo, dado o alto contraste apresentado e a ausência de sombras ou outros elementos parasitas, as linhas, sendo todas brancas, conseguem-se separar com alguma facilidade independentemente da codificação utilizada.

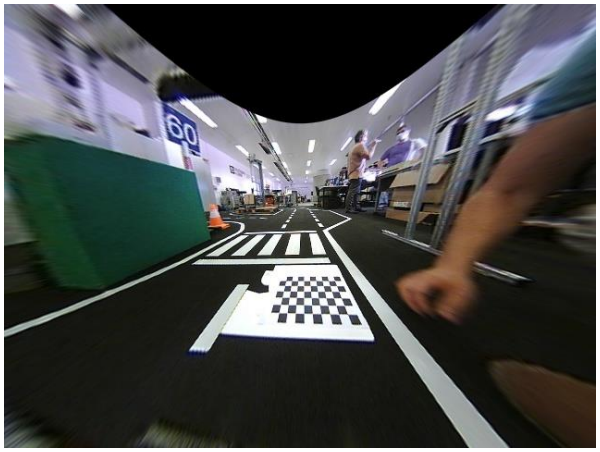


Figura 28 - Imagem combinada na codificação RBG.

R) G) B)

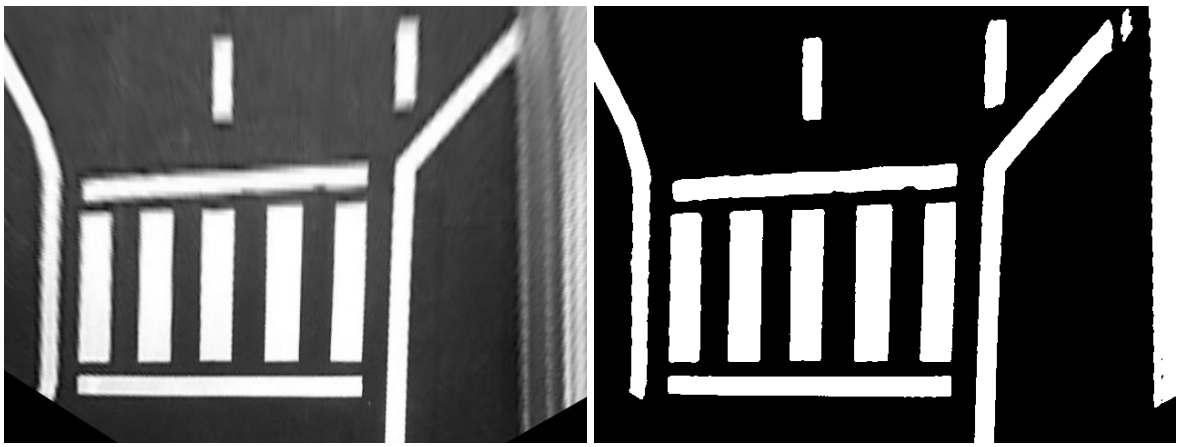
Figura 29 - Imagem separada nos diversos canais da codificação RGB.

Através das Figura 28 e Figura 29, é possível observar as diferenças, principalmente nos 3 objetos assinalados, o obstáculo verde, o sinal de transito azul e o cone laranja. Comparando estes 3 objetos, é possível perceber que os mesmos são sempre mais intensos na imagem em que estão assinalados, a imagem que representa o canal da sua cor dominante. Para a deteção

das linhas, a separação dos canais não resultou em grandes diferenças, visto que o branco das mesmas refletiu sensivelmente a mesma intensidade em todos os canais.

3.3.2. Binarização e limiares adaptativos

Apesar do processamento de cores simplificar o tratamento da imagem, os canais que compõem as diversas codificações podem ainda conter muita informação desnecessária. A binarização consiste na transformação dos pixéis de uma imagem em escala de cinzentos para apenas dois valores: "0" ou "1", resultando numa imagem a preto e branco. Para executar esta técnica é necessário definir o limiar de separação, vulgarmente conhecido por *threshold*, valor que serve de referência para tornar o píxel branco ou preto. Numa imagem de 8 *bits* em escala de cinzentos, cada píxel varia entre 0 (preto) e 255 (branco), sendo que, consoante o valor do limiar, estes passarão a assumir um dos dois resultados possíveis, "0" ou "1", passando a imagem para preto e branco (binarizada), Figura 30.



a) b)

Figura 30 - Binarização de uma imagem: a) Escala de cinzentos; b) Binarizada.

Em situações de luz irregular, como sombras e reflexos na estrada, esta técnica deixa de ser tão eficaz como mostrado na Figura 31. Neste caso, algoritmos adaptativos pela média, podem fazer a diferença. Estes não se baseiam em valores fixos, mas sim na análise da imagem e na perceção da zona vizinha. Os valores médios da zona vizinha são o que definem o valor de binarização (*threshold*) ou, no caso do algoritmo de Otsu, análise de um histograma de toda a imagem.

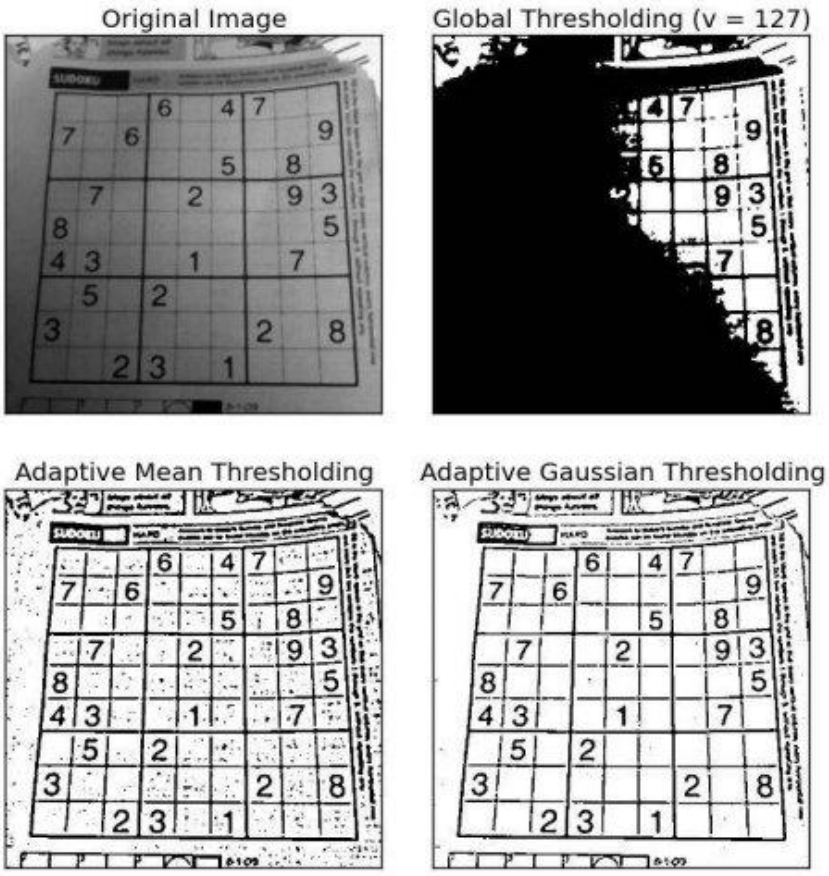


Figura 31 - Binarização adaptativa (Binarização de Otsu's) [27]**.**

Plataformas como o *OpenCV* já incluem este tipo de algoritmos para facilitar o seu uso, sendo que o utilizador apenas precisa de parametrizar algumas variáveis, como o limiar ou o tamanho da matriz.

3.3.3. Filtros

Uma das características inerente ao mundo real é o ruído e as imagens não são exceção. Defeitos provenientes do ruído dos sensores devem ser atenuados para não interferirem com os algoritmos, no entanto, este processo deve ser meticuloso e na quantidade correta, pois demasiada suavização pode levar à perda de detalhes importantes. Segundo Xu e Li [11], a média, mediana (Figura 32) e filtros Gaussianos são os filtros mais eficientes e tipicamente utilizados para suavizar imagens.

Figura 32 - Filtro por mediana [28]**.**

Estes filtros de mediana têm como base uma matriz, por exemplo, de 3x3, e definem o valor do píxel central pela mediana dos adjacentes dentro dessa mesma matriz. Os filtros Gaussianos partem do mesmo princípio, mas o peso do pixéis mais afastados tende a sempre a ser menor consoante a função gaussiana definida que, num plano de duas dimensões como as imagens, se aproxima de uma distribuição normal em todas as direções, como mostra a Figura 33.

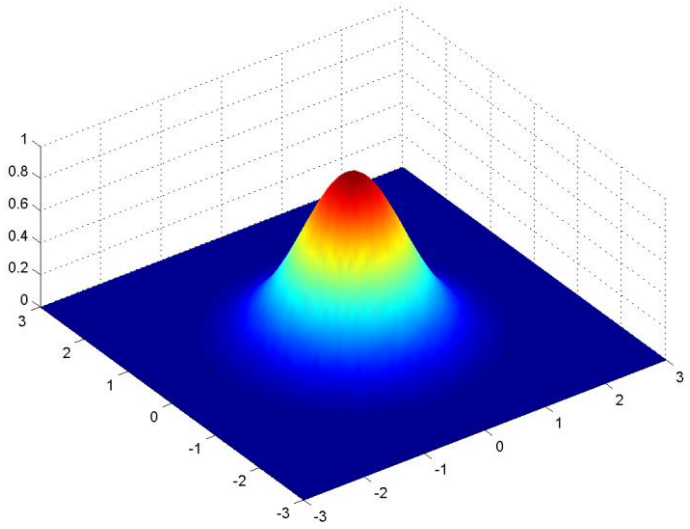


Figura 33 - Matriz gaussiana 7x7 [13].

À semelhança do sucedido no processo de calibração e transformação de perspetiva, a ordem com que se executam as operações são determinantes na eficiência dos mesmos e a aplicação de filtros não é exceção. Quando o objetivo é manter a nitidez, a binarização deve ser o último passo, por forma a manter os contornos bem delineados. Na Figura 34, pode-se comparar os resultados do uso deste pré-processamento na ordem incorreta. Além de não remover o ruído de forma eficaz, ainda resulta numa imagem mais desfocada, que pode causar dificuldades na deteção dos contornos.

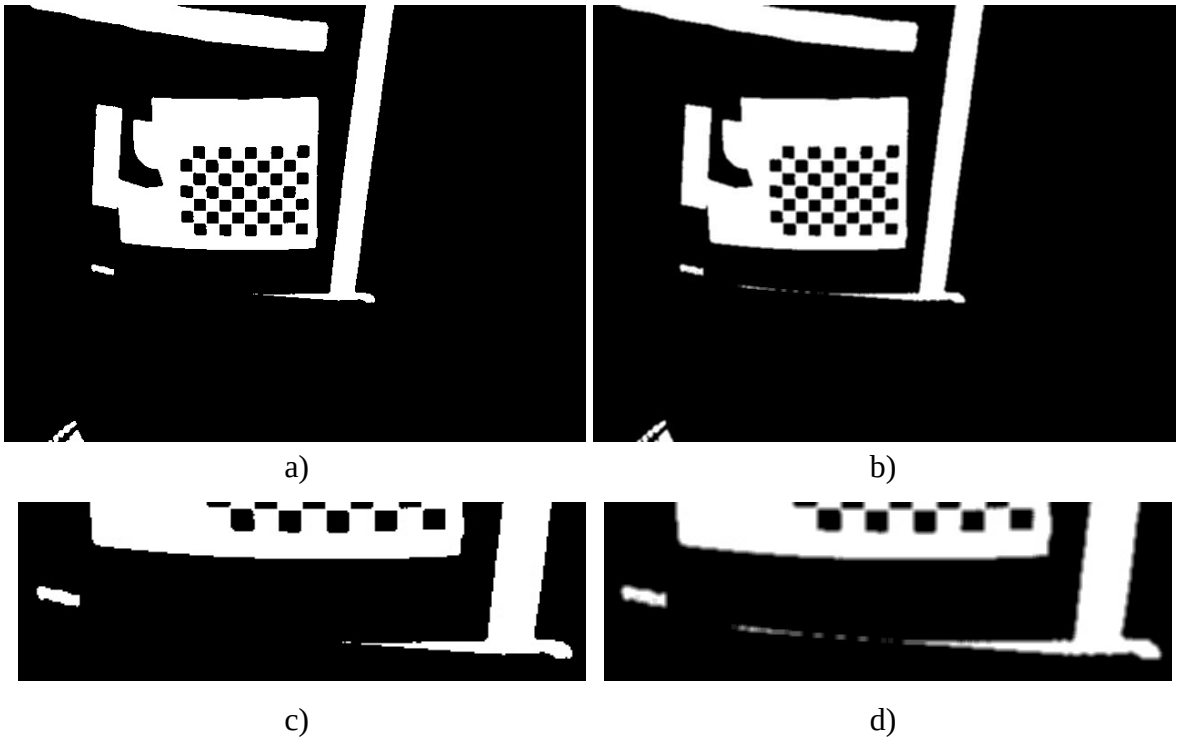


Figura 34 - Ordem de procedimentos quanto aos filtros: a) Filtro Binarização; b) Binarização Filtro; c) Ampliação da imagem a); d) Ampliação da imagem d).

Numa outra perspetiva, os filtros podem ainda ser usados para fazer uma seleção bem mais refinada, deixando apenas passar elementos que o utilizador quer usar, nomeadamente as linhas da faixa de rodagem, em função de características que não apenas a intensidade dos píxeis. Isto é, um algoritmo perfeito devia apenas detetar as linhas que formam a faixa de rodagem, no entanto, secções como a passadeira ou outros objetos circundantes à pista são captados e apenas acrescentam informação que contribui negativamente para a deteção.

Por forma a reduzir este impacto, testaram-se diversos filtros convolucionais. Estes filtros percorrem todos os pixéis da imagem realizando uma convolução entre um *kernel* (matriz personalizada) e uma imagem. A Figura 35 demonstra de forma gráfica o resultado de um filtro simples utilizando apenas um vetor.

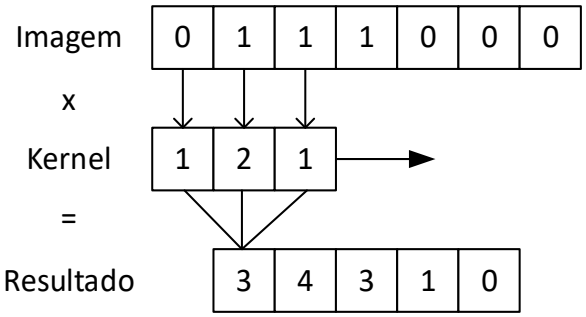
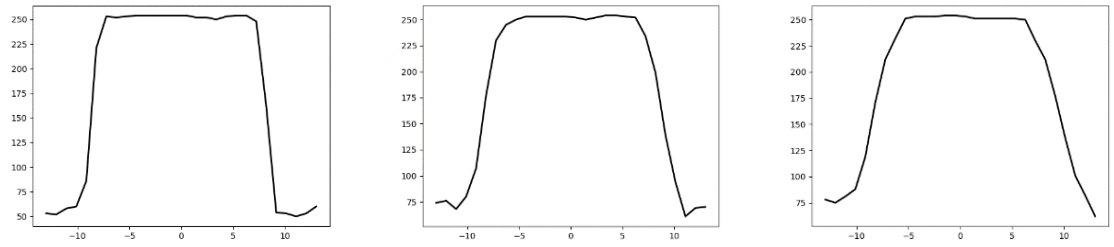


Figura 35 - Filtro convolucional aplicado a um sinal 1D.

No caso em questão, os filtros testados pretendiam excluir tudo o que não fossem linhas pertencentes à faixa de rodagem, linhas estas que têm uma característica bem definida, a largura. Para se poder caracterizar corretamente a matriz, recolheram-se algumas amostras em diversas zonas da pista, cujos resultados estão ilustrados na Figura 36, onde o eixo das ordenadas representa a intensidade do píxel e as abcissas a largura da amostra. A diferença entre as várias amostras deriva da distância a que estas foram recolhidas, em relação ao veículo, quanto mais longe menos nítido, e como tal, menos acentuado vai ficando o filtro.



1) 2) 3)

Figura 36 - Amostras de filtragem.

No final optou-se pela utilização do filtro representado na Figura 37. Para remover as linhas mais grossas é necessário aplicar uma penalização e, por esse motivo, linhas com uma largura superior a 20 píxeis são sancionadas. Essa penalização é dada pelos valores negativos presentes na matriz abaixo da linha a tracejado.

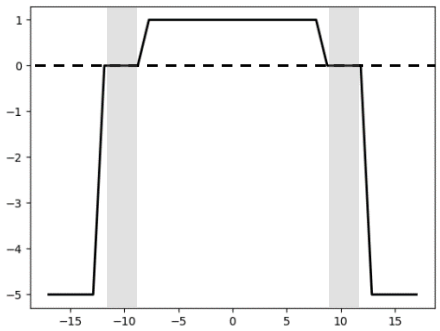


Figura 37 - Filtro convolucional (ideal).

Para não aplicar uma matriz demasiado agressiva, e aceitar as possíveis imperfeições da captação, o filtro contém uma zona cinzenta, zona esta que ignora os dados de entrada. Esta zona tem uma banda cinzenta de 4 píxeis de cada lado e desta forma permite aceitar linhas entre 16 e 24 píxeis, Figura 37

A Figura 38 apresenta um exemplo de resultado final. Para melhorar o filtro, principalmente nas curvas, podem ainda ser sobrepostas diversas matrizes com ângulos diferentes combinando os seus resultados.

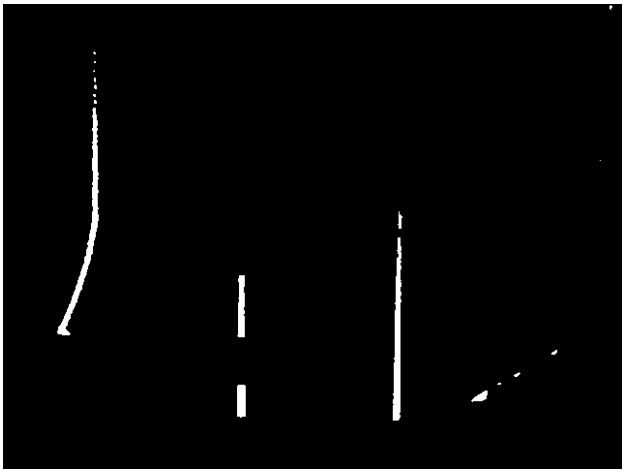


Figura 38 - Resultado do filtro convolucional.

3.3.4. Região de Interesse

Habitualmente as câmaras frontais que captam as linhas na faixa de rodagem das estradas, e são responsáveis pela identificação e processamento das mesmas, são posicionadas com uma inclinação que favorece: a monitorização do espaço próximo do veículo para deteção de ameaças, assim como a deteção das linhas que delimitam as faixas de rodagem. Inevitavelmente, as câmaras capturaram informação desnecessária na deteção das linhas que delimitam as faixas de rodagem, como zonas acima do horizonte e zonas exteriores à faixa de rodagem. Estes dados secundários devem ser removidos do processamento computacional, aumentando assim a eficácia e eficiência do algoritmo. Como mostra a Figura 39**,** a seleção de uma região de interesse, tipicamente denominada de ROI (*Region Of Interest*), é uma das técnicas mais utilizadas para este fim, removendo geralmente informação acima da linha de horizonte (a vermelho na figura) ou, em casos mais agressivos, selecionando apenas um pequeno trapézio na zona da faixa de rodagem (a verde na figura)**.**

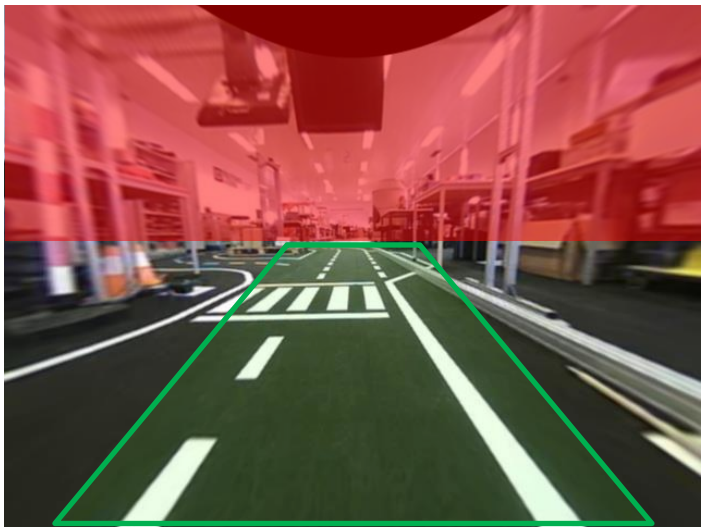


Figura 39 - Região de interesse.

Esta técnica pode ainda ser aplicada para auxiliar a deteção de determinados objetos ou padrões na imagem, isolando as áreas onde se espera encontrar esses mesmo objetos, como, por exemplo, as setas indicadoras de direção desenhadas nas faixas de rodagem ou a própria passadeira. No entanto, esta região de interesse pode sofrer problemas resultantes da dinâmica do veículo. Ou seja, visto que a seleção da região tem em conta o referencial da câmara, durante acelerações, travagens e mudanças de direção, veículos com sistemas de suspensão tendem a ter problemas na utilização desta técnica, visto que os referenciais da câmara e do mundo deixam de estar alinhados.

3.4. Técnicas e Algoritmos Isolados

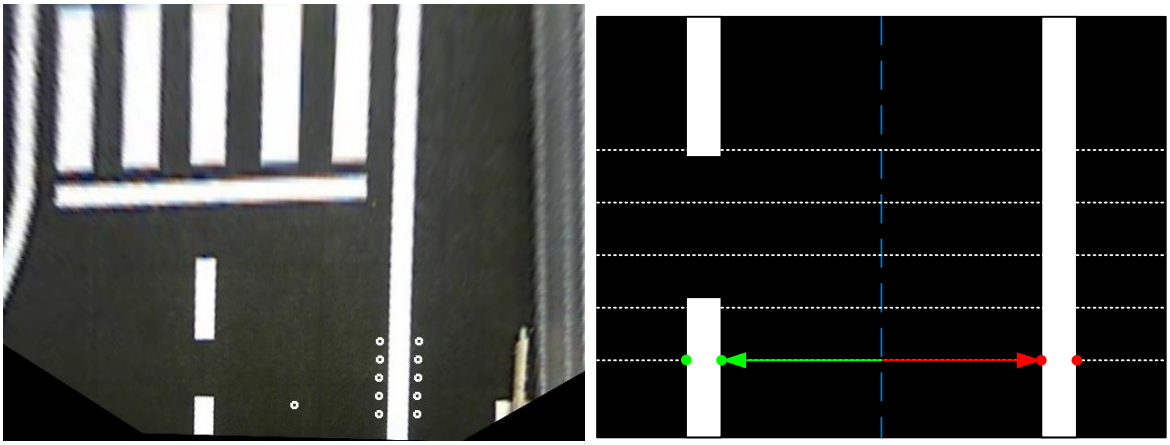
Neste tópico irão ser abordados vários algoritmos de forma isolada, algoritmos estes que terão como objetivo explorar diversas técnicas para deteção de linhas e, indiretamente, da faixa de rodagem. Além disso, são também exploradas algumas metodologias que não têm como objetivo a deteção propriamente dita, mas sim o auxílio aos restantes algoritmos.

3.4.1. Varrimento Horizontal

Partindo da imagem otimizada para a deteção das linhas (*vide* Figura 29 b)) que compõem a faixa de rodagem, abordaram-se diversas alternativas para alcançar esse mesmo objetivo.

O algoritmo de *varrimento horizontal* é bastante robusto, no entanto, está sobreajustado ao cenário em questão, o que o torna difícil de usar caso haja alguma mudança no cenário. Este consiste em vários varrimentos horizontais, partindo do centro da imagem até atingir ambas

as linhas da faixa de rodagem. Este processamento é executado sobre uma imagem binarizada e repete-se em diferentes cotas da imagem por forma a evitar erros provocados pela ausência da linha, principalmente nas zonas tracejadas, como mostra a Figura 40.



a) b)

Figura 40 - Algoritmo de varrimento horizontal: a) real; b) diagrama representativo.

Ou seja, partindo de uma posição central, representada a azul na Figura 40 b), o algoritmo começa a percorrer cada uma das linhas, em direções opostas, até encontrar duas transições de estado consecutivas, preto-branco e branco-preto. Com estes dois pontos, assinalados a verde e vermelho na Figura 40 b), é possível calcular a largura da linha e validar a deteção. A caracterização das linhas tem em conta a média dos pontos em cada uma das linhas horizontais.

As zonas de passadeira e mudanças abruptas de direção na linha criam alguns problemas de interpretação e, por este motivo, foi necessário criar diversas validações para garantir a correta deteção das linhas pretendidas. A situação presente na Figura 41, por exempo, gerou muitos problemas. O facto da faixa de rodagem se dividir em duas, o tracejado e a linha para a zona de parque, obrigou a desenvolver validações adicionais, nomeadamente, deteções de declive e integração do histórico de deteção.

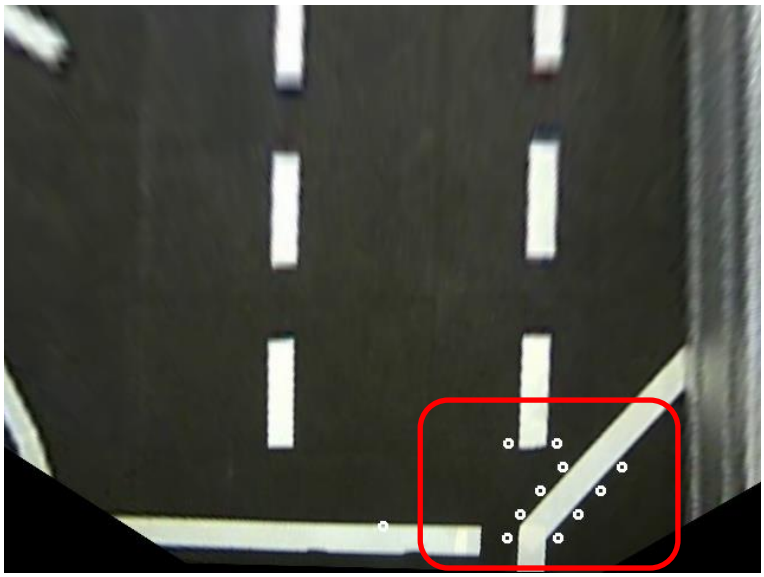
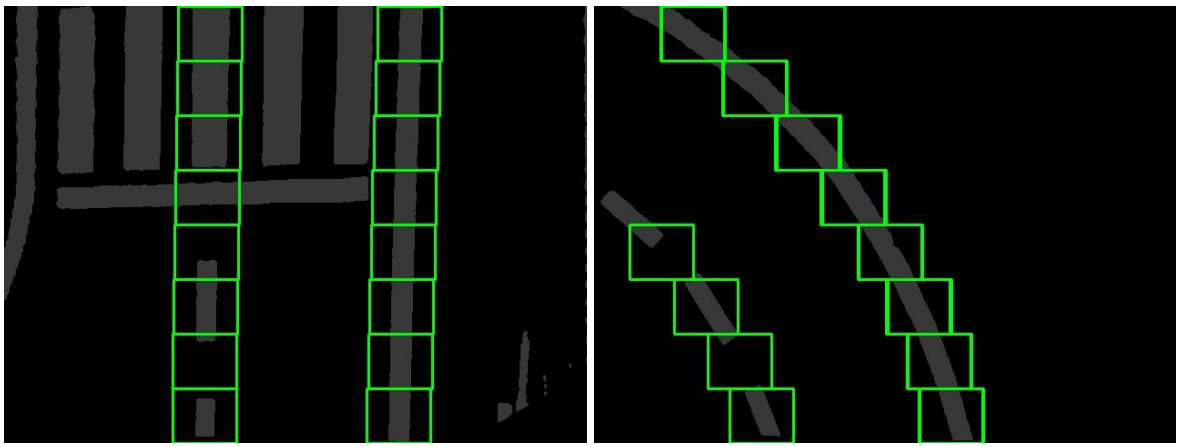


Figura 41 - Problema de mudanças abruptas no algoritmo de varrimento.

Estas alterações acabaram por deixar o método demasiado sobreajustado, levando à necessidade de encontrar outras soluções para o complementar.

3.4.2. Caixas Deslizantes

O método das *caixas deslizantes* é um processo iterativo que se baseia no posicionamento de diversas caixas (ROI) ao longo da imagem, procurando seguir a faixa de rodagem.



a) b)

Figura 42 - Algoritmo caixas deslizantes.

Numa primeira fase, o algoritmo procura a posição inicial das linhas, tendo por base a soma vertical de todos os pixéis brancos existente no primeiro terço da imagem. Desta soma resulta um gráfico, semelhante a um histograma, com a localização mais provável das duas linhas da faixa (a Figura 43 representa esse mesmo histograma, gerado a partir da Figura 42 a)).

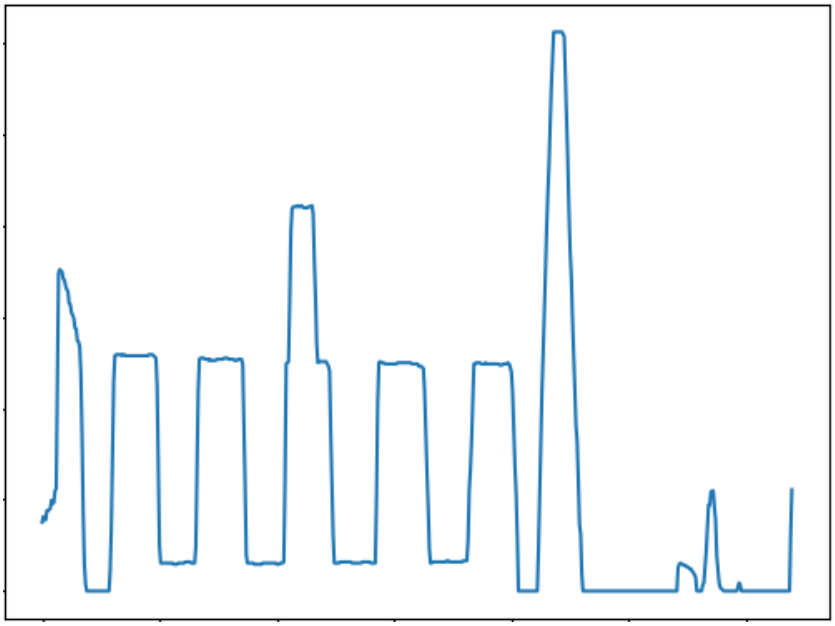


Figura 43 - Histograma (somatório vertical).

Sendo este um processo iterativo, o posicionamento das "novas" caixas tem em conta a posição média dos pixéis dentro da caixa anterior, no entanto, rapidamente se constatou que, em situações de curva, este não era capaz de acompanhar a linha pretendida, como mostra a Figura 44 a). Para corrigir este problema, a posição das caixas não se pode basear apenas na média anterior, mas ter também em conta a tendência com base nas últimas posições através do declive por exemplo, como mostra a Figura 44 b).

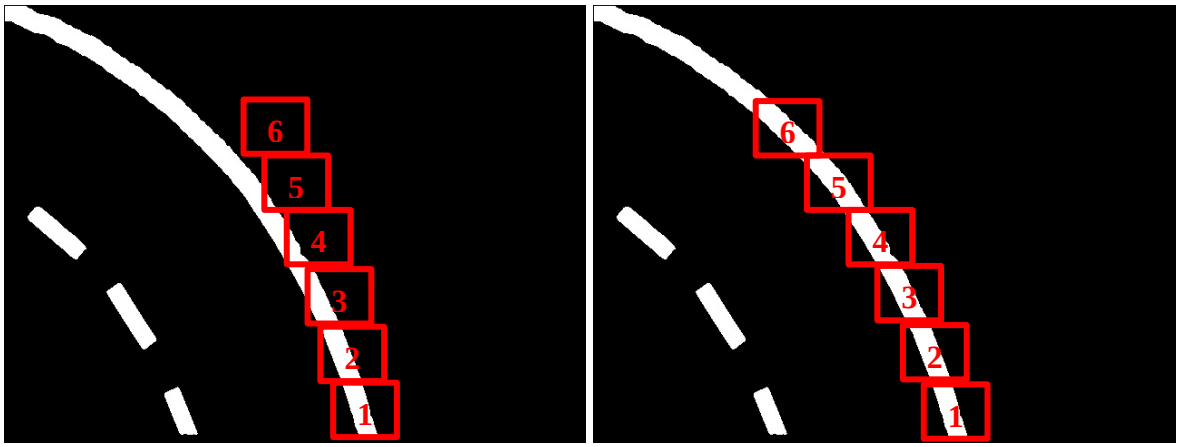


Figura 44 - Posicionamento das caixas.

Todos os cálculos e posicionamentos das caixas são efetuados individualmente para cada uma das linhas, esquerda e direita. O resultado final deste método pode ser visualizado na Figura 45.

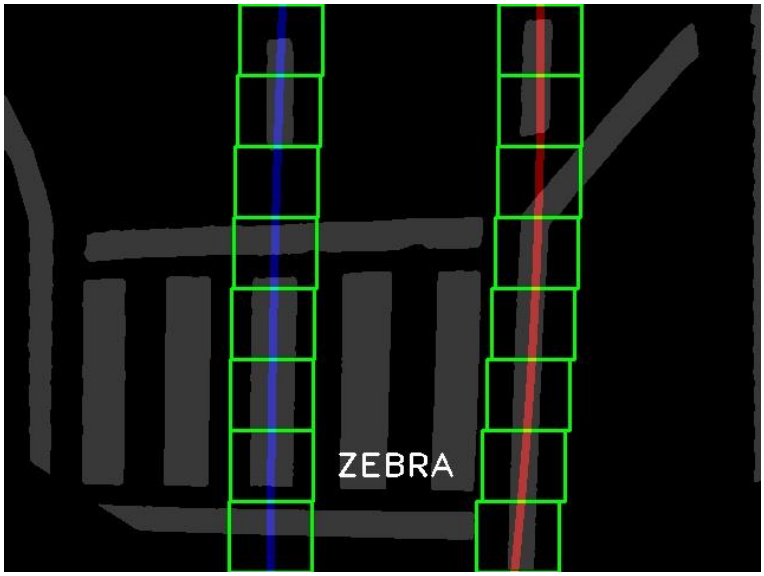


Figura 45 - Resultado do algoritmo de caixas deslizantes.

Este algoritmo devolve, como solução final, os coeficientes da aproximação polinomial de segunda ordem gerada através do posicionamento da caixas.

Em algumas situações, principalmente em alterações abruptas na faixa de rodagem ou bifurcações, o posicionamento das caixas pode não ser fiel, gerando aproximações polinomiais erráticas. Por este motivo, foi necessário criar validações que tivessem em conta tanto o histórico assim como a relação geométrica entre linhas, assuntos que serão abordados nos tópicos 3.4.3 e 3.4.4.

3.4.3. Histórico e Odometria

O histórico das deteções anteriores é uma das ferramentas mais poderosas na validação das aproximações atuais. O conhecimento da localização das linhas nas iterações anteriores permite prever a sua posição futura tendo em conta os dados da odometria, isto é, sabendo o deslocamento do veículo. No entanto, por motivos extrínsecos ao desenvolvimento deste projeto, não foi possível adquirir dados com a odometria do veículo na fase de testes do algoritmo de deteção de faixa de rodagem.

Numa fase inicial optou-se por uma abordagem de comparação com as iterações anteriores, ou seja, o cálculo atual é comparado com a iteração anterior e esta não podia diferir da

primeira mais do que um determinado valor. Esta diferença era aplicada diretamente nos parâmetros da curva resultante, ou seja, na comparação entre coeficientes da curva de segunda ordem e analisando a sua variação. A técnica obteve resultados bastante positivos, sendo possível eliminar falsas deteções que, por motivos diversos, eram aceites como linhas válidas da faixa de rodagem. Desta forma a captação das linhas ficou muito mais suave e linear ao longo da pista toda.

Não obstante, testou-se a odométrica visual, um algoritmo designado de *viso2*[3] que consiste na análise e processamento de imagens consecutivas, de forma a resultar num vetor de deslocamento. O algoritmo baseia-se na captação de pontos chave, geralmente objetos com contornos bastante acentuados e os seus vértices, para servirem de base no cálculo do deslocamento num referencial estático.

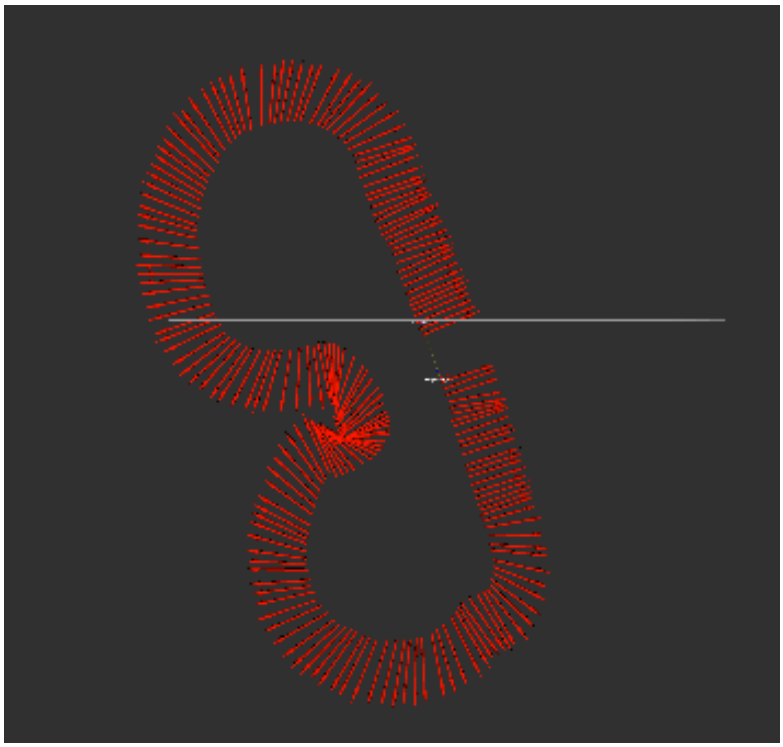


Figura 46 - Resultado da odometria visual.

Devido à complexidade do algoritmo de odometria de visual, verificou-se que o processamento era demasiado pesado, o que impossibilitava adequado o seu uso em tempo real. A obtenção da imagem anterior necessitou de um tempo de reprodução 100 vezes inferior ao real, sendo que qualquer valor acima desse introduzia um erro muito elevado,

[3] ROS Viso2, wiki.ros.org/viso2

tornando o resultado inutilizável. Como se pode observar na Figura 46, o resultado possui algumas irregularidades, mesmo quando executado bastante abaixo do tempo real.

De modo a utilizar a odometria visual, a solução passou por remover este processamento do computador e realizá-lo externamente. Tendo disponível uma câmara Intel RealSense T265 no laboratório (Figura 47), testou-se a sua utilização.

Figura 47 - Câmara Intel RealSense T265 [29]**.**

Esta câmara possui duas lentes olho de peixe, um giroscópio e acelerómetro, ambos de 3 eixos, que, em conjunto com um algoritmo de V-SLAM (*Visual - Simultaneous Localization and Mapping*), executado diretamente na unidade de processamento interna, devolve com baixa latência e alta frequência o vetor de deslocamento. A Figura 48 mostra o resultado da utilização da T265, onde se pode observar a estimativa da pose do veículo ao longo de uma volta completa com o algoritmo embutido da câmara. Não existindo *groundthruth*, todas as análises são obtidas de forma puramente qualitativa, **i.e.** observação visual.

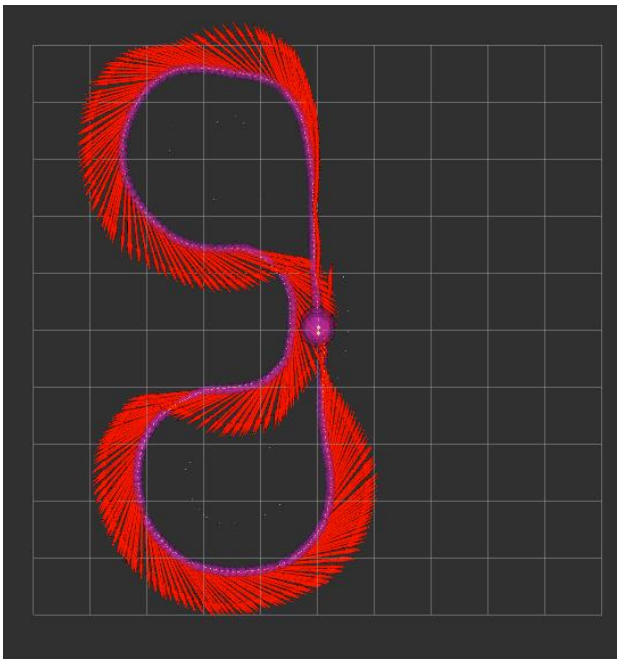


Figura 48 - Resultado da odometria visual de T265.

O resultado obtido pelo algoritmo presente nesta câmara é sem dúvida muito superior ao obtido com o algoritmo *viso2* e é capaz de correr em tempo real. Adicionalmente, por forma a testar os limites desta câmara, na utilização num veículo de condução autónoma à escala real, aplicou-se a mesma no veículo de *Formula Student* da equipa do Politécnico de Leiria, cujos resultados podem ser observados na Figura 49.

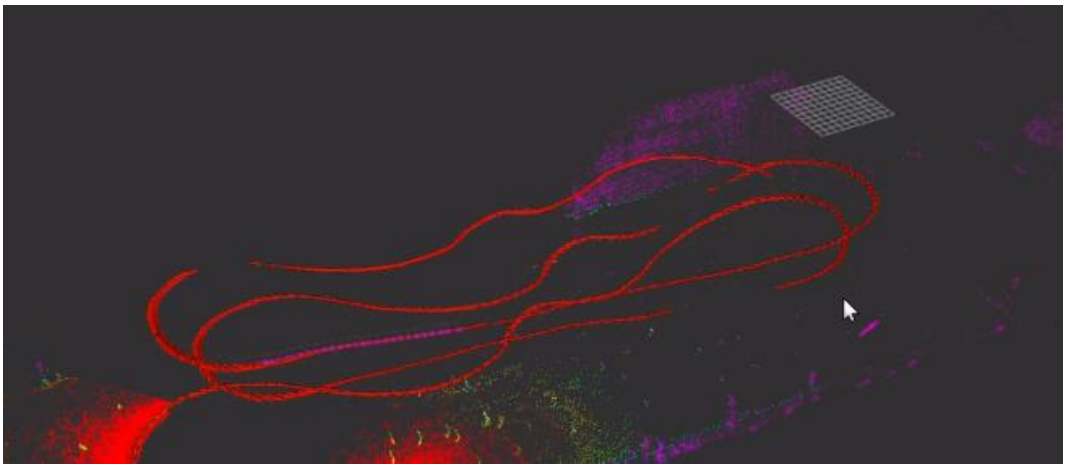


Figura 49 - Resultado da odometria visual proveniente da Intel T265.

É possível observar um grande desvio ao longo de uma volta. As especificações da câmara indicam o erro de 0.1% em cada metro, no entanto, o erro observado da figura supera os valores esperados. Uma possível explicação para este fenómeno, pode ser a discrepância entre imagens consecutivas demasiado grande, o que não permite captar pontos comuns suficientes para que o algoritmo de V-SLAM. Ainda assim, esta conclusão é demasiado preliminar e necessitaria de um estudo aprofundado, seguindo, por exemplo, alguns dos métodos usados no artigo de Peter Hausamenn, [30].

Figura 50 - Pista utilizada no teste do Formula Student.

3.4.4. Paralelismo

A relação geométrica entre as linhas é um método também bastante eficaz para perceber o grau de confiança das deteções. Quando se usam os algoritmos totalmente independentes e com abordagens distintas, a probabilidade é baixa de coincidirem no mesmo tipo de falhas que resultem em linhas similares erradamente identificadas. Por esse mesmo motivo, elaborou-se um algoritmo para o cálculo do paralelismo entre linhas. A Figura 51 demonstra a definição de paralelismo e a base de todo o algoritmo desenvolvido.

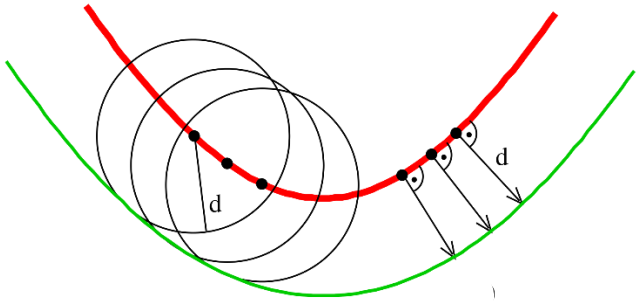


Figura 51 - Curvas paralelas [31].

Para o cálculo do paralelismo é necessário recorrer à parametrização das equações polinomiais que, partindo da equação (6), deve apresentar a forma encontrada na equação (7).

$$y(x) = Ax^2 + Bx + C \tag{6}$$

$$\begin{cases} x(t) = t \\ y(t) = At^2 + Bt + C \end{cases} \tag{7}$$

Posto isto, a representação dos pontos paralelos, $x_d(t)$ e $y_d(t)$ segue-se as equações (8) e (9), onde d representa a distância da curva pretendida e $x'(t)$, $y'(t)$ as derivadas das equações paramétricas representadas na equação (7).

$$x_d(t) = x(t) + \frac{d \cdot y'(t)}{\sqrt{x'(t)^2 + y'(t)^2}} \tag{8}$$

$$y_d(t) = y(t) - \frac{dx'(t)}{\sqrt{x'(t)^2 + y'(t)^2}} \quad (9)$$

No entanto, o algoritmo é uma representação discreta, isto é, baseia-se num determinado número de pontos e gera uma curva paralela com base nessa amostra. No entanto, no contexto da validação, dado o ruído, resolução e vibração da câmara, entre outros fatores, o algoritmo tem de ter alguma tolerância, ou seja, tem de gerar um intervalo de verificação, como mostra Figura 52.

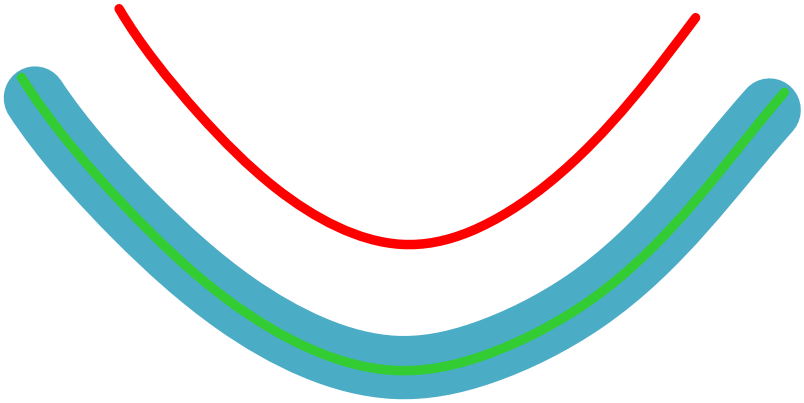


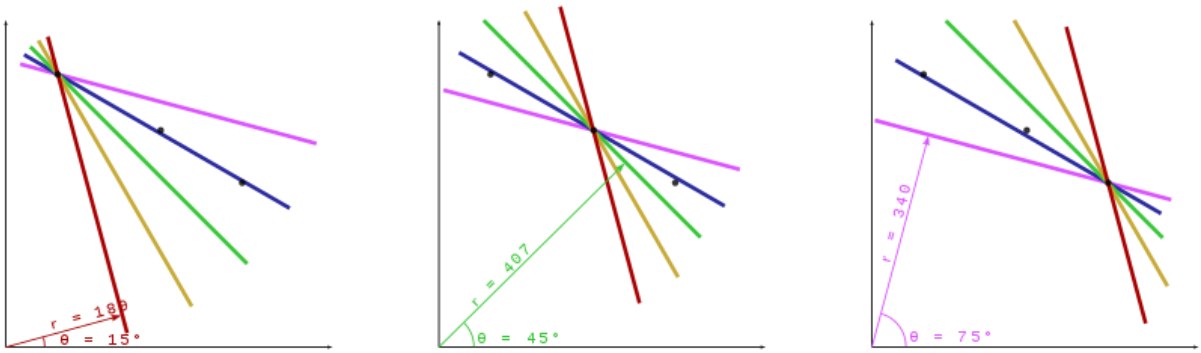
Figura 52 - Intervalo de paralelismo gerado.

Assim sendo, o algoritmo de verificação de paralelismo retorna um valor de confiança que se baseia no número de pixéis brancos dentro do intervalo a azul. Na prática, é realizada a contagem dos pixéis válidos dentro de um intervalo na zona azul. Esse valor pode ser facilmente ajustável consoante o contexto em que se pretende utilizar, assim como definir trechos específicos de validação, quando não se pretende verificar a linha na sua totalidade.

3.4.5. Transformada de *Hough*

A transformada de Hough é um algoritmo popular utilizado para encontrar qualquer forma que possa ser caracterizada por uma função matemática que, para melhor perceção, se usará aqui como exemplo a linha reta [32].

A técnica passa pela caracterização das linhas que passam por todos os pontos da imagem em coordenadas polares. Ao acumular todos os parâmetros θ e r (Figura 53) numa matriz, irão a surgir pontos de convergência.



θ	r
15	189.0
30	282.0
45	355.7
60	407.3
75	429.4

θ	r
15	318.5
30	376.8
45	407.3
60	409.8
75	385.3

θ	r
15	419.0
30	443.6
45	438.4
60	402.9
75	340.1

Figura 53 - Matriz de parâmetros O e P [33].

Tendo por base o exemplo da Figura 53, o objetivo é aproximar uma reta dos 3 pontos representados a preto, sendo as linhas representas, as diversas iterações do algoritmo em cada ponto. Assim sendo a matriz irá começar a apresentar pontos concentrados nos valores $\theta = 60$ e $r \approx 405$, visto que são estes os valores que caracterizam a linha azul, linha esta que passa pelos 3 pontos. A Figura 54 é uma representação gráfica da matriz acima mencionada, no entanto, é um exemplo relativo a duas retas, daí as duas concentrações de pontos.

Figura 54 - Convergência dos parâmetros de Hough [34].

Neste caso em particular, a ideia é utilizar a transformada de Hough em secções da imagem de forma isolada. A Figura 55 mostra a extração de segmentos de linhas numa das secções.

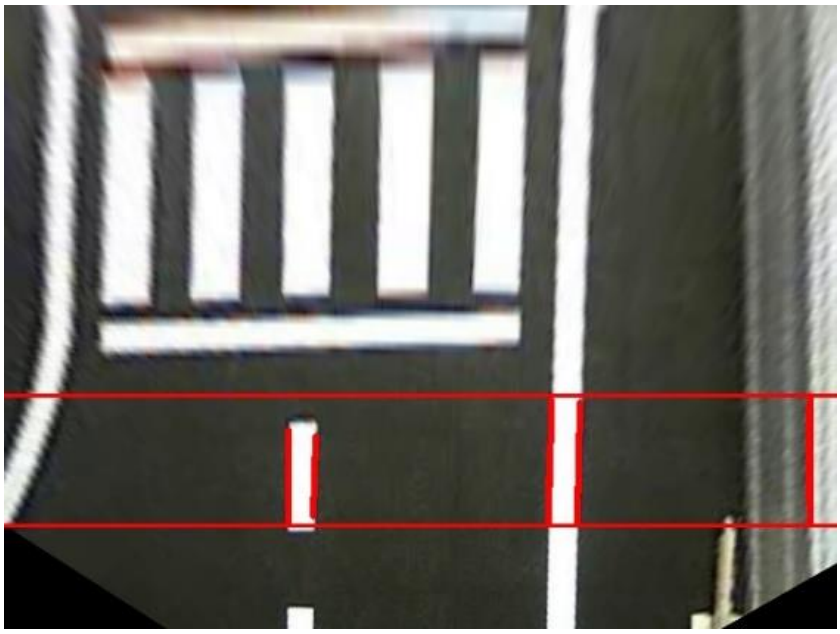


Figura 55 - Hough por secções.

Daqui resultam os diversos segmentos que permitem perceber a tendência da faixa de rodagem dentro dessa mesma área. Com o agrupamento das secções e o declive dos vários segmentos, é possível formar uma linha central que representa a orientação dessa secção no centro da faixa de rodagem, como mostra a Figura 56.

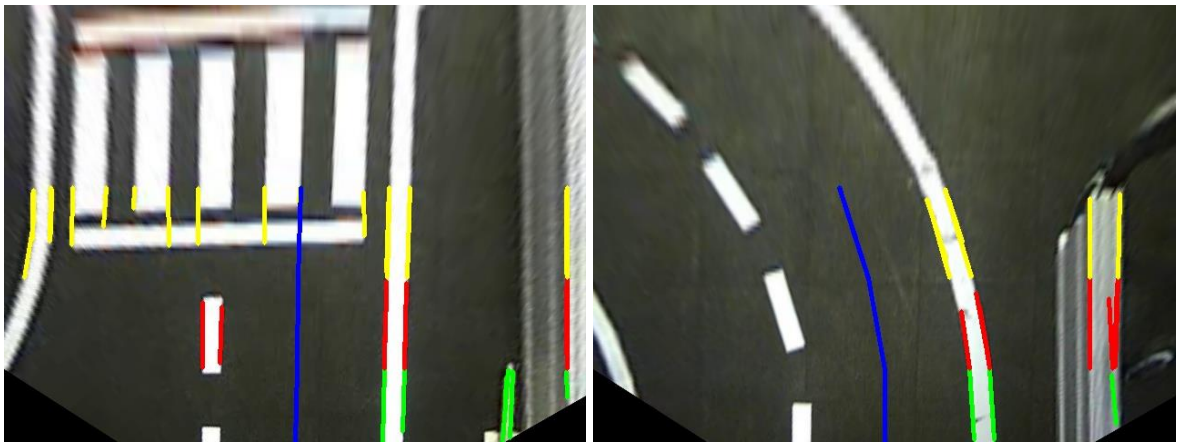


Figura 56 - Transformada de Hough por secções ao longo da imagem.

Esta técnica mostrou resultados promissores. No entanto, tem uma grande vulnerabilidade: se o veículo sair do centro da faixa de rodagem este não a consegue detetar, pois, baseia-se apenas na tendência da faixa e não na localização das linhas. Isto é, baseando-se a linha azul

apenas nos declives das várias segmentações, e tendo esta sempre inicio no centro da imagem, se o veículo estiver fora da faixa de rodagem, o resultado do algoritmo mantear-se-á igual desde que a média dos declives seja a mesma. Contudo, em conjunto com outro algoritmo adicional para identificar o centro da faixa a curta distância, este pode trazer resultados muito positivos, secção 3.4.1.

3.4.6. Deteção de linhas tracejadas

Os tracejados das faixas de rodagem são zonas particularmente difíceis de detetar, pois as suas constantes interrupções tornam o seu processamento mais imprevisível. Por forma a contornar este problema, decidiu-se desenvolver um algoritmo dedicado à deteção das linhas a tracejado.

Tendo em conta algumas das técnicas usadas até então, a primeira etapa tem como objetivo a segmentação da imagem, ou seja, tentar isolar apenas os tracejados. Para tal utilizou-se um operador de deteção de bordos designado de *Canny*, um algoritmo que integra técnicas já mencionadas como: filtros gaussianos, intensidade de gradientes e limiares de seleção com histerese. Estas etapas incluem vários parâmetros de entrada que podem ser ajustados consoante o objetivo, para devolver apenas os contornos relevantes de uma imagem, como mostra a Figura 57 [35].

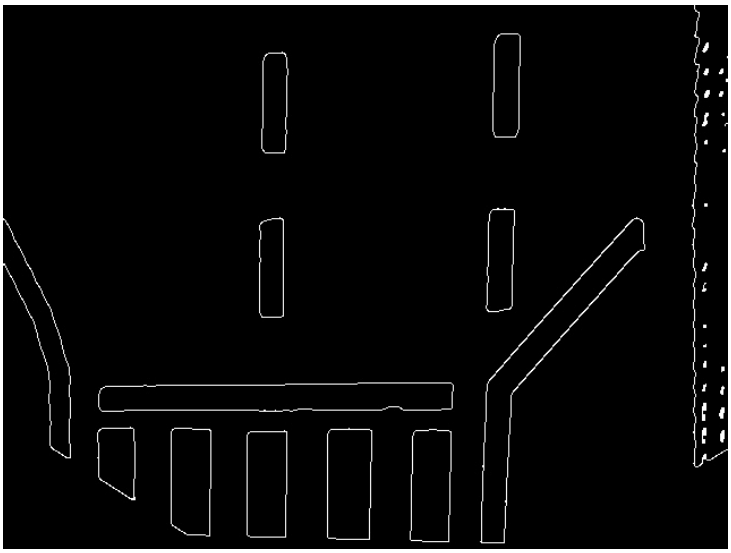


Figura 57 – Exemplo do operador de deteção de bordos Canny, aplicado ao caso de estudo.

Recorrendo à função "*findcontours*"[4] do OpenCV, extrai-se uma lista dos contornos presentes da imagem anterior. Esta função é uma ferramenta bastante útil para análise de formas geométricas, deteção e reconhecimento de objetos [36], percorre todos os píxeis da imagem, agrupando aqueles que possuem continuidade.

Utilizaram-se duas técnicas de análise para identificar os tracejados, tendo por base o comprimento/perímetro e o momento dos contornos. Através dos comprimentos/perímetro é possível remover todos aqueles que, por serem demasiado longos ou curtos, não se enquadram na forma dos tracejados. No entanto, existem falsos positivos que, apesar de passarem no filtro de comprimento/perímetro, não se enquadram no perfil de tracejados das faixas de rodagem (*vide* Figura 58).

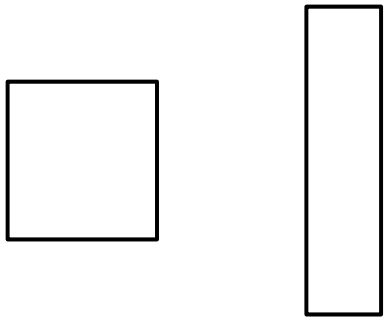


Figura 58 - Objetos com o mesmo perímetros e formas bastante diferentes.

O momento de um contorno caracteriza as propriedades do contorno em termos do seu centro geométrico e momento de inércia. Conhecendo *a priori* as características aproximadas destes objetos, é possível identificar quais é que se encaixam e quais devem ser rejeitados. A Figura 59 mostra o resultado de uma seleção utilizando o comprimento dos contornos e os respetivos momentos de inércia, partindo da imagem presente na Figura 57.

[4] https://docs.opencv.org/3.4.15/df/d0d/tutorial_find_contours.html

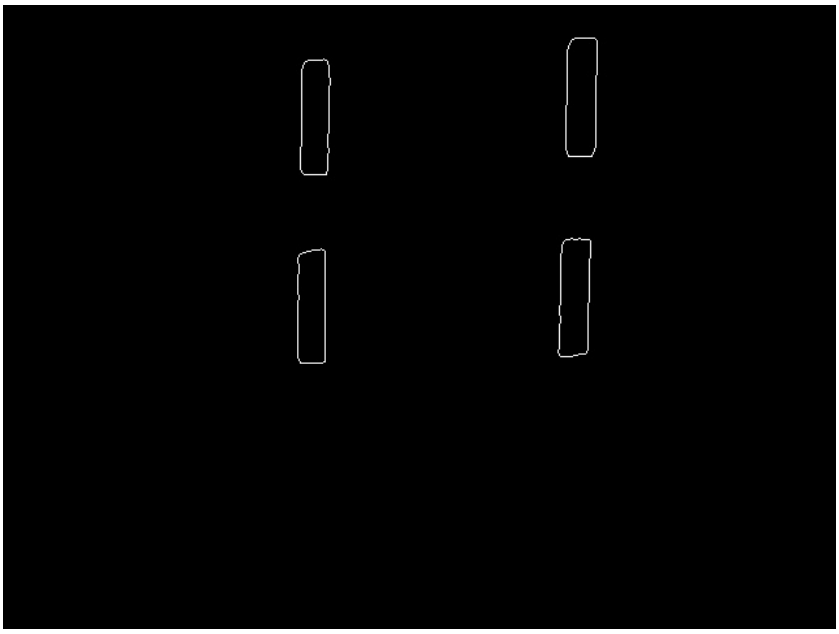


Figura 59 - Resultado da seleção por momentos dos contornos.

Todavia, este processo resulta apenas na identificação de diversos objetos de forma individual, não lhes atribui nenhuma ligação. Por forma a contornar este problema é necessário agrupar os diversos contornos identificados em conjuntos lógicos, ou seja, separá-los por linha (esquerda, direita) e organizá-los de baixo para cima. Para identificar que um determinado contorno se deve associar a uma linha da faixa de rodagem, *i.e.* a que contorno se deve juntar um contorno anterior, e assim sucessivamente. Tendo em consideração que existem múltiplas linhas nas faixas de rodagem, é necessário desenvolver um algoritmo de junção tendo em conta todos estes fatores.

Numa primeira abordagem, isolaram-se os pontos relevantes de cada contorno, ou seja, o ponto médio do par de vértices mais afastados, como se de uma linha se tratasse (*vide* Figura 60). Posto isto, é necessário fazer a ligação de todos os pontos, procurando o mais próximo do contorno seguinte e assim sucessivamente.

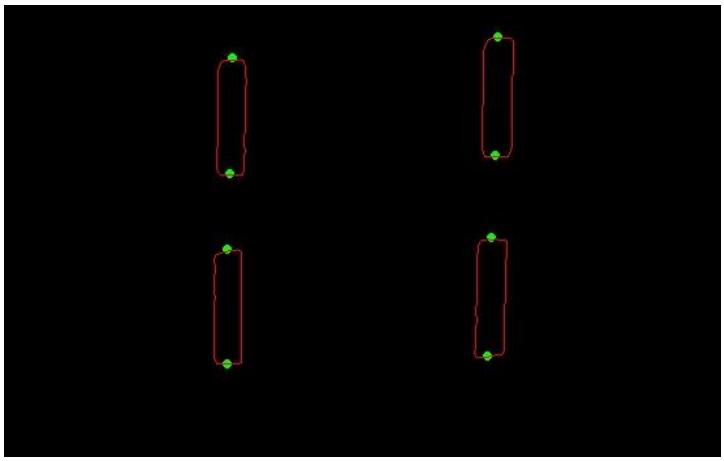


Figura 60 - Vértices dos tracejados.

Uma das alternativas que se testou foi a utilização de um algoritmo designado de KD-TREE, uma proposta que organiza todos os pontos numa árvore binária e que, por esse motivo, consegue encontrar qualquer ponto de uma forma mais eficiente. Tendo por base a Figura 61, se o utilizador quiser descobrir o ID (número a vermelho) da caixa com o número 25, ao procurar numa lista de forma consecutiva este teria que passar por 6 verificações caso esta estivesse organizada em ordem crescente. Usando uma KD-Tree, o mesmo número é encontrado em apenas 3 iterações pois, em cada passo, o algoritmo verifica o conteúdo e seleciona o ramo mais próximo do "alvo".

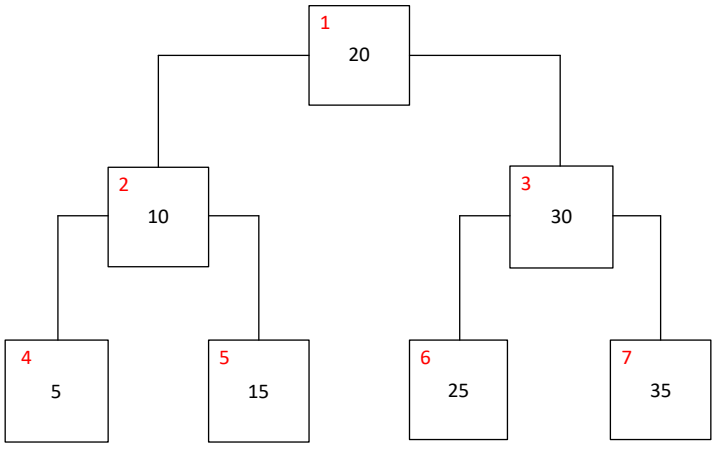


Figura 61 - Exemplo de uma KD-Tree.

A eficiência desta técnica vai naturalmente depender do número de pontos utilizados, sendo mais relevante quanto maior o número de pontos. Neste caso em particular, esta técnica foi apenas testada numa matriz exemplo criada para o efeito pois, dado o reduzido número de pontos presentes na situação real, não compensava a sua utilização, pelo custo adicional associado à criação da lista de pontos.

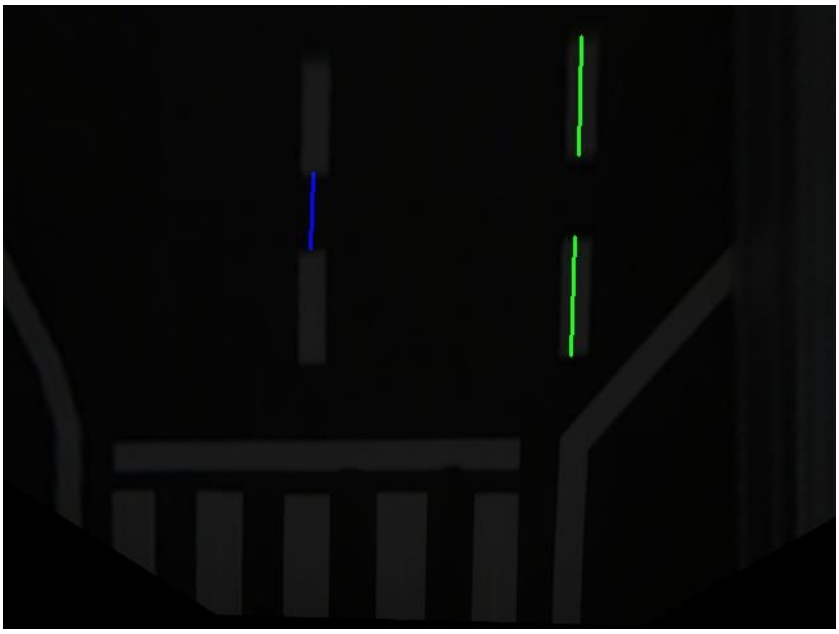


Figura 62 - Ligação de todos os vértices.

Posto isto, e para suavizar as linhas e remover os diversos segmentos de reta, utiliza-se uma aproximação polinomial ao conjunto de pontos já separados. Esta técnica vai ser abordada com maior detalhe na secção 3.5 sendo esta base do algoritmo B da solução final. A título de exemplo, a Figura 63 mostra o resultado da cálculo da linha central da faixa de rodagem, tendo por base a imagem ilustrada na Figura 62.

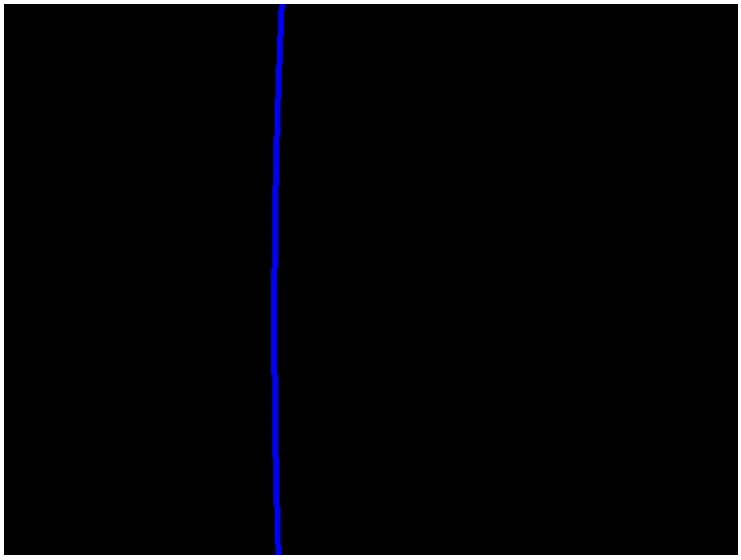
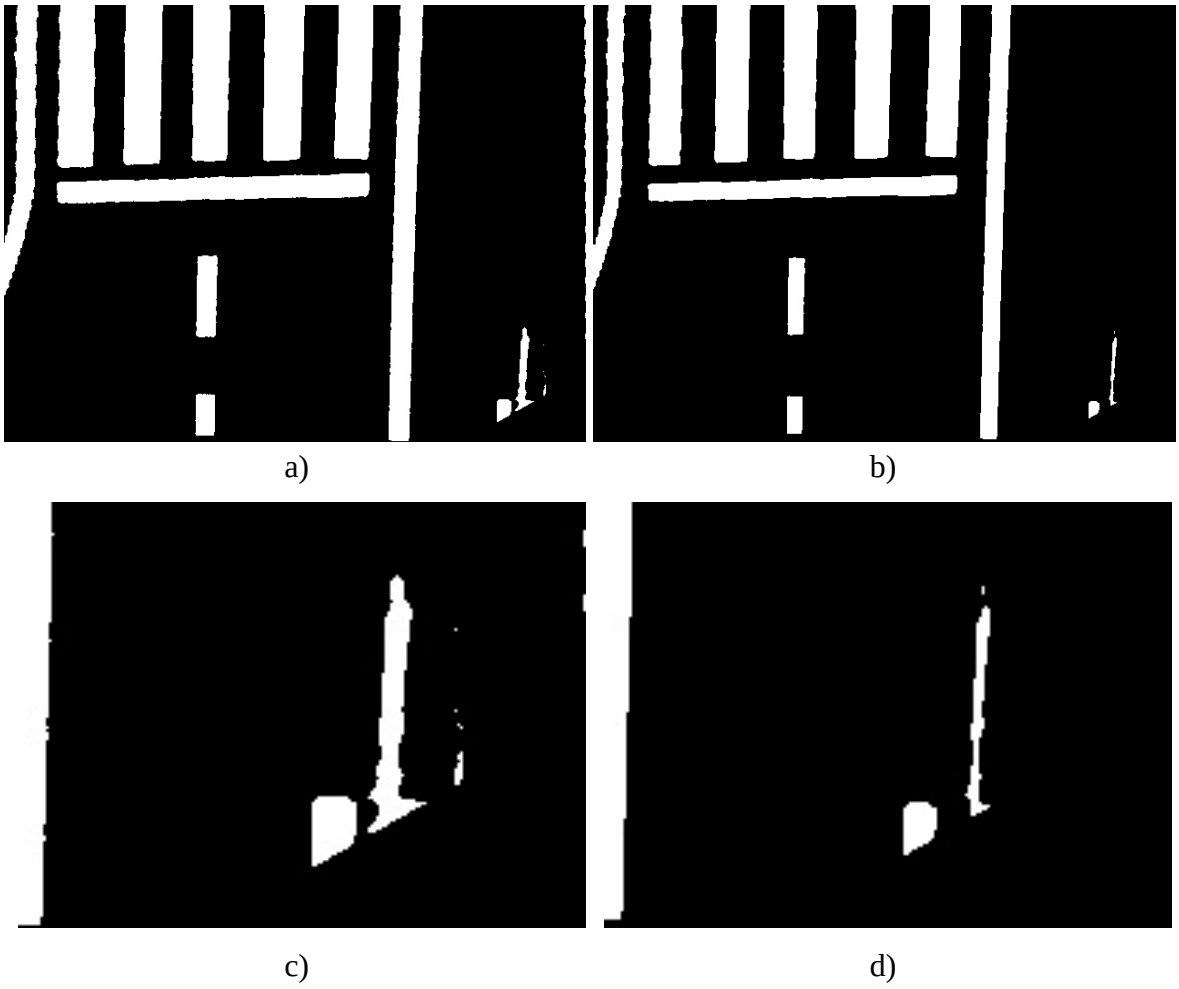


Figura 63 - Aproximação polinomial tendo por base a ligação dos vértices.

3.4.7. Erosão, Dilatação e Esqueletização

Como já foi referido, uma das formas de melhorar o processamento de dados é simplificar o problema e, em termos de imagem, uma das possibilidades para simplificar a deteção das linhas numa etapa posterior, é remover uma das suas características, a espessura.

Para tal, a primeira técnica que se avaliou foia utilização da operação morfológica de erosão, ou seja, a redução dos contornos de forma iterativa que, por sua vez, vai reduzindo a espessura das linhas, como mostra a Figura 64.



a) b)

c) d)

Figura 64 - Erosão com 3 iterações: a) Antes; b) Depois; c) Ampliação da imagem a); d) Ampliação da imagem d).

Este processo vai removendo a espessura dos objetos presentes na imagem de forma uniforme, ou seja, em ambos os eixos X e Y. Sendo o objetivo desta técnica reduzir a espessura das linhas, idealmente a um 1 píxel de espessura, esse resultado iria também afetar o comprimento das linhas, especialmente dos tracejados, como mostra a Figura 65.

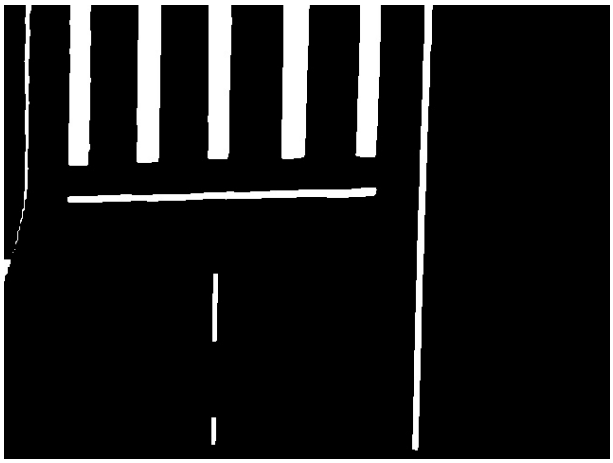


Figura 65 - Erosão com 8 iterações.

Além disso, seria muito difícil aplicar o número de iterações correto para que a redução fosse a ideal em todas as linhas. Na Figura 65, é possível perceber que um determinado número de iterações ainda não foi suficiente para reduzir a espessura das linhas a um 1 píxel em determinados objetos, enquanto noutros esse mesmo número de iterações já faz com que a linha seja interrompida ou removida.

No entanto, durante a sua utilização, esta técnica demonstrou ser bastante útil para remoção de ruído e até de pequenos objetos. Da Figura 64 a) para a Figura 65 é possível perceber que um dos objetos captados, não relevantes para a deteção da linha da faixa de rodagem, foi removido com sucesso. Neste sentido, este processo pode ser utilizado a priori para tratar imagens, sendo que o seu efeito negativo terá que ser mitigado e, para tal, irá usar-se o processo dual da erosão, a dilatação. Esta função expande todos os contornos dos objetos de forma uniforme, no entanto, ao contrário do que se possa pensar, o resultado final não será igual à imagem inicial.

a) b)

Figura 66 - Dilatação 7 iterações: a) antes; b) depois.

As Figura 64, Figura 65 e Figura 66 demonstram o efeito da aplicação de uma erosão seguida de uma dilatação com a mesma matriz e número de iterações. Como se pode observar na Figura 66 b), os artefactos no canto inferior direito foram removidos na erosão e, ao aplicar a dilatação, como esses pixéis já não existem, estes não podem ser expandidos, logo não voltam a surgir.

Ainda assim, o objetivo inicial continua sem se conseguir atingir, pelo que a opção recaiu pelo algoritmo designado de "*Thinning*" ou esqueletização. Esta é uma operação morfológica presente no O*penCV*, aplicada em imagens, que tem como como objetivo a remoção seletiva de pixéis. Ao contrário da erosão e da dilatação, esta operação reduz os objetos tendo em conta a sua forma, tentando sempre preservar o seu "esqueleto", [37].

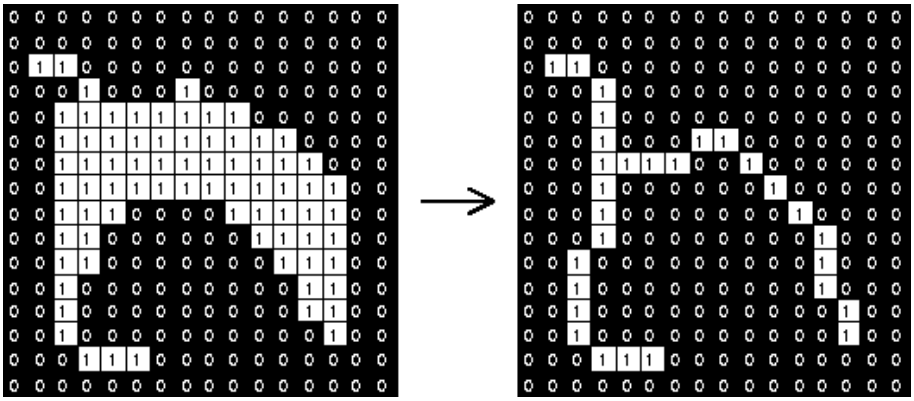


Figura 67 - Esqueletização [37]**.**

No caso particular da deteção das linhas, o resultado inicial foi bastante promissor, a "esqueletização" dos objetos a detetar foi bastante precisa, como mostra a Figura 68. Contudo, a sua velocidade de processamento é muito dependente da área dos elementos a detetar, algo extremamente crítico quando se procura um algoritmo para deteção em tempo real e, neste caso, o seu desempenho não foi suficiente para o hardware utilizado. As zonas de passadeira precisam de aproximadamente 2 vezes mais tempo de processamento do que as zonas rápidas como as retas. Em determinadas regiões com objetos parasitas de dimensão relevante, o tempo de cálculo alcançou máximos de até 20 vezes o tempo de processamento, tornando impossível a sua utilização em tempo útil.

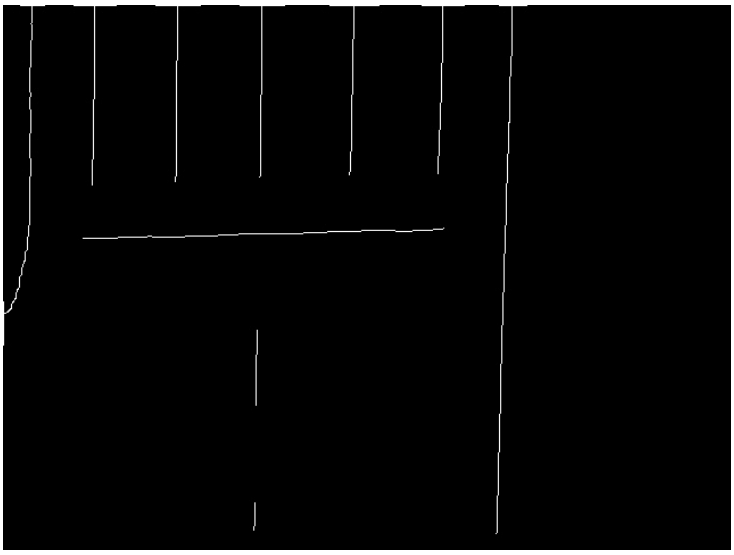


Figura 68 - Esquetelização da faixa de rodagem.

Ainda assim, com a devida otimização e eliminando ou filtrando todos aqueles objetos que à partida não servirão para a deteção, este algoritmo poderá trazer bons resultados.

3.4.8. Estimação de Horizonte

A estimação de horizonte é uma técnica que pode também ser usada para diversos fins neste campo da condução autónoma. Esta consiste na utilização da transformada de *Hough* (Secção 3.4.5) para calcular o ponto de fuga da imagem, ou seja, tendo como base a imagem em perspetiva (não a imagem de BEV), com as linhas limites da faixa de rodagem a tender para um ponto único, o qual se denomina de "ponto de fuga", que coincide com o horizonte da imagem. Esta técnica pode auxiliar o processo de seleção de região adaptativa mencionada na Secção 3.3.4, indicando onde se encontra o horizonte e, assim, isolar a região de interesse do algoritmo principal [14].

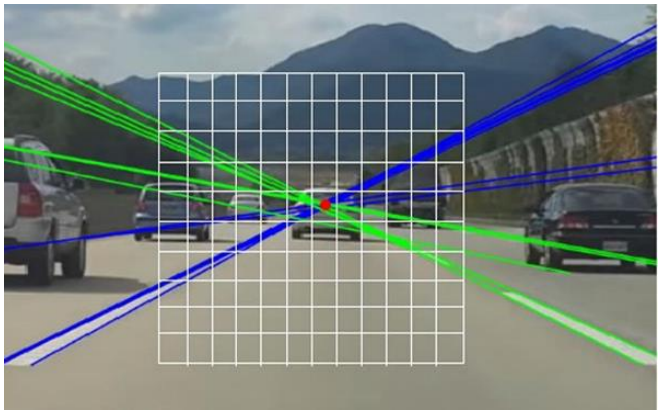


Figura 69 - Cálculo do ponto de fuga [15].

Além de auxiliar a região adaptativa, este processo pode ainda ser utilizado para um cálculo muito primitivo da curvatura da estrada. Isto é, conhecendo a tendência das linhas é possível distinguir o fluxo das mesmas e, desta forma, saber o raio da curvatura aproximado numa determinada região das faixas de rodagem, como mostra a Figura 70.

Figura 70 - Previsão de curvatura [16].

No entanto, dado que os raios na pista de testes são muito reduzidos, esta técnica não surtiu resultados positivos e por esse motivo não foi utilizada em nenhum dos algoritmos finais.

3.4.9. RANSAC

RANSAC (*RANdom SAmple Consensus*) é um método robusto para estimar os parâmetros de um modelo matemático na presença de ruído que, neste caso em particular, irá ser usado para estimar as diversas linhas da faixa de rodagem. Por forma a simplificar a explicação irá assumir-se o modelo matemático de uma reta. Este método começa por gerar uma aproximação com base numa amostra de um determinado número de pontos da imagem, aleatoriamente escolhidos. De seguida, com base nessa aproximação, o algoritmo irá calcular o erro com base num conjunto de pontos que serão designados de *inliers* e *outliers*, consoante estes se encontrem dentro ou fora respetivamente de um determinado limiar, respetivamente, como mostra a Figura 71.

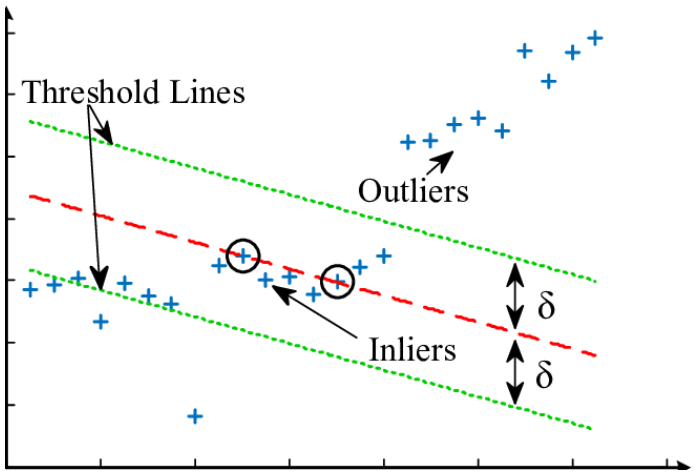


Figura 71 - Exemplo de RANSAC [38].

Este processo é feito de forma iterativa com os diversos pontos da imagem até que se atinja um erro mínimo (número de *inliers* máximo) ou se exceda o número de iterações máximo. Na Figura 72 é possível observar que, apesar do ruído na imagem (pontos incorretamente detetados), o RANSAC é um algoritmo eficaz a enquadrar a linha correta (a verde na figura), ao contrário de uma aproximação típica linear (a vermelho na figura) que tivesse por base a amostra completa.

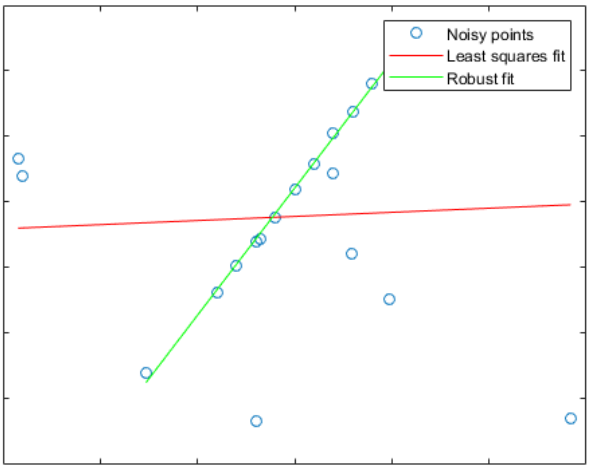


Figura 72 - Aproximação por RANSAC vs Aproximação por mínimos quadrados [38].

Contudo, no contexto deste projeto, este algoritmo foi considerado demasiado pesado computacionalmente quando aplicado numa imagem para a deteção da faixa de rodagem. Por ser iterativo, em cada imagem o algoritmo tem de processar todas as hipóteses, avaliando os modelos matemáticos gerados, e procurar o erro mínimo dentro do conjunto total. Nos testes preliminares realizados, o algoritmo demonstrou ter um impacto reduzido, pois, a aproximação tem em conta os pontos todos e não permite a separação das linhas. O resultado pode ser visualizado na Figura 73.

Figura 73 - Resultado do algoritmo RANSAC.

No entanto, aplicando algumas das técnicas de pré-processamento mencionadas na Secção 3.3, este pode melhorar a sua eficiência isolando cada uma linhas e aplicando o RANSAC individualmente.

3.5. Algoritmo Final, Resultados e Testes

Depois de testar os vários algoritmos e avaliar o seu desempenho em diferentes situações, a solução final passa pela combinação de algumas das técnicas anteriormente mencionadas. O objetivo é unir os pontos fortes dos diferentes algoritmos colmatando as suas falhas ou cenários mais complexos.

Como já foi referido, o primeiro passo é, sem dúvida, a retificação da imagem captada pelo sensor e, para tal, é necessário seguir todas as notas apresentadas na Secção 3.1. De seguida, é necessário realizar a transformação de perspetiva. Esta etapa serve essencialmente para remover o efeito de perspetiva responsável pelas alterações geométricas dos objetos mais distantes da câmara. Por forma a aumentar a eficácia destas técnicas, nenhum tipo de pré-tratamento de imagem deve ser realizado até esta fase.

Assim sendo, é realizado um pré-tratamento cromático começando por uma redução à escala de cinzentos, permitindo assim reduzir a quantidade de informação e facilitar o processamentos. Neste caso em particular, dado o alto contraste da estrada, a transformação para escala de cinzentos advém da junção dos três canais (RGB), atribuindo a mesma relevância para todos. Posteriormente, e de forma a reduzir ruído e minimizar algumas imperfeições da deteção é aplicado um filtro gaussiano, suavizando assim a imagem. Esta é uma operação delicada, sendo necessário encontrar um balanço para remover os defeitos sem perder demasiados detalhes. Neste caso, o *kernel* utilizado possui um tamanho de 3x3. Para finalizar, é efetuada uma binarização da imagem para lhe aumentar o contraste e nitidez nos contornos e simplificar a informação disponível, colocando todos os pontos com um de dois valores, “0” ou “1”.

A Figura 74 mostra um fluxograma do processo.

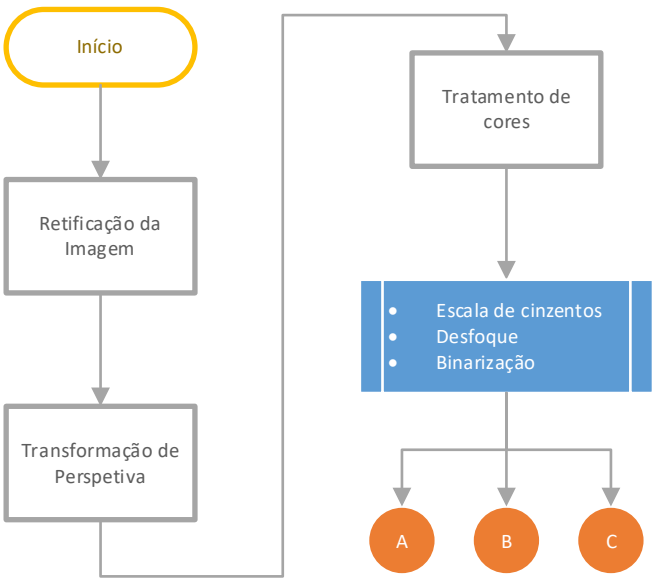


Figura 74 - Fluxograma do pré-processamento.

Posto isto, inicia-se o processamento de 3 algoritmos em paralelo (A, B e C), os três melhores que, de acordo com os resultados, combinados, conseguem detetar a faixa de rodagem em toda a pista.

O primeiro algoritmo (Algoritmo A) baseia-se na Transformada de *Hough* (Secção 3.4.5) e, como já fora referido, consiste no cálculo do declive médio dos contornos presentes em diversos sectores horizontais da imagem. Começa-se por utilizar o detetor de contornos *Canny*, e definir a primeira secção de busca. Em cada secção este aplica a Transformada de *Hough* Probabilística, função já disponível no *OpenCV*, isto porque, comparando com a função normal, a função probabilística é computacionalmente mais rápida, bastante importante quando se tenta desenvolver um algoritmo capaz de correr em tempo real em conjunto com outros dois algoritmos. De seguida, é realizado o cálculo do declive sobre todas as semirretas presentes na secção, que resultaram da transformada de *Hough*. Todos os declives que não estejam dentro dos valores esperados, como linhas horizontais, são descartados e não entram para avaliação. É então representada uma semirreta que representa a média dos declives. Esta deve ser colocada no centro estimado da faixa de rodagem e os segmentos seguintes, das secções consequentes, devem ser agrupados aos segmentos anteriores, formando assim um conjunto de semirretas, como mostra a Figura 75. Todos os pontos que formam as diversas semirretas podem ser agrupados para executar uma aproximação polinomial e suavizar a sua representação, adicionalmente possibilita a definição matemática da curva.

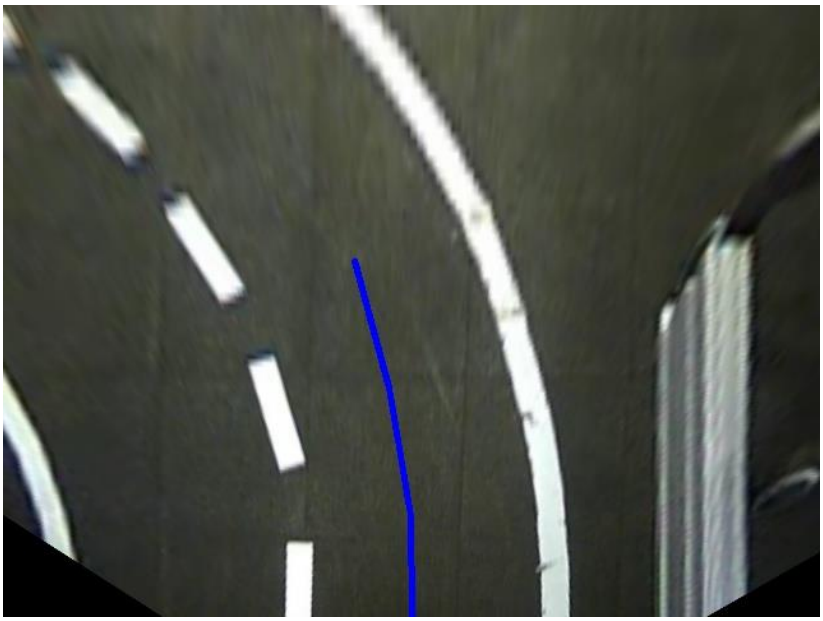


Figura 75 - Resultado do primeiro algoritmo.

O algoritmo A encontra-se representado no fluxograma apresentado na Figura 76, onde A representa o início e AF o final.

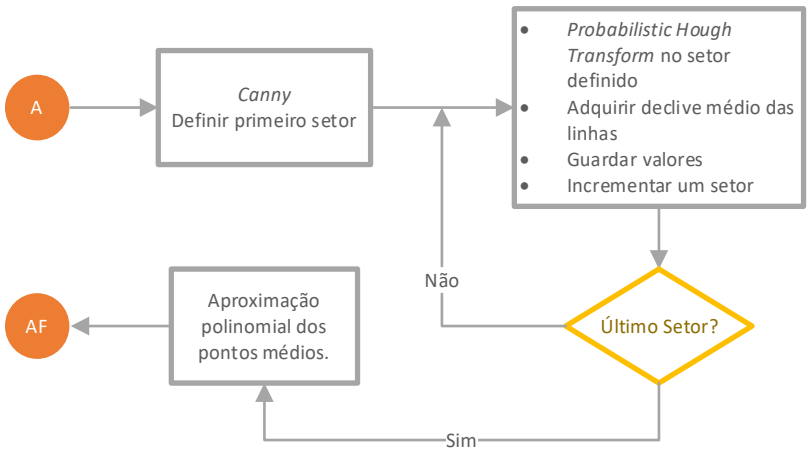


Figura 76 - Fluxograma do algoritmo A

Paralelamente, e como segundo algoritmo (algoritmo B), está o conjunto de operações resumido no fluxograma apresentado na Figura 77, dedicado à deteção de linhas interrompidas, ou seja, os tracejados presentes na faixa de rodagem, onde B representa o início e BF o final. Começa-se novamente por extrair todos os contornos da imagem com auxílio do detetor de bordos *Canny*. Posto isto, passa por um conjunto de validações começando pelo comprimento, sabe-se *a priori* que este deve rondar os 50cm, o perímetro

do traço típico e, de seguida, valida-se a geometria, característica esta que deve apresentar um rácio de 4:1 (altura:largura).

Todos os contornos que passam nas validações, antes de serem armazenados numa matriz, são reduzidos a dois pontos, início e fim, removendo a sua espessura, representando assim a sua linha central.

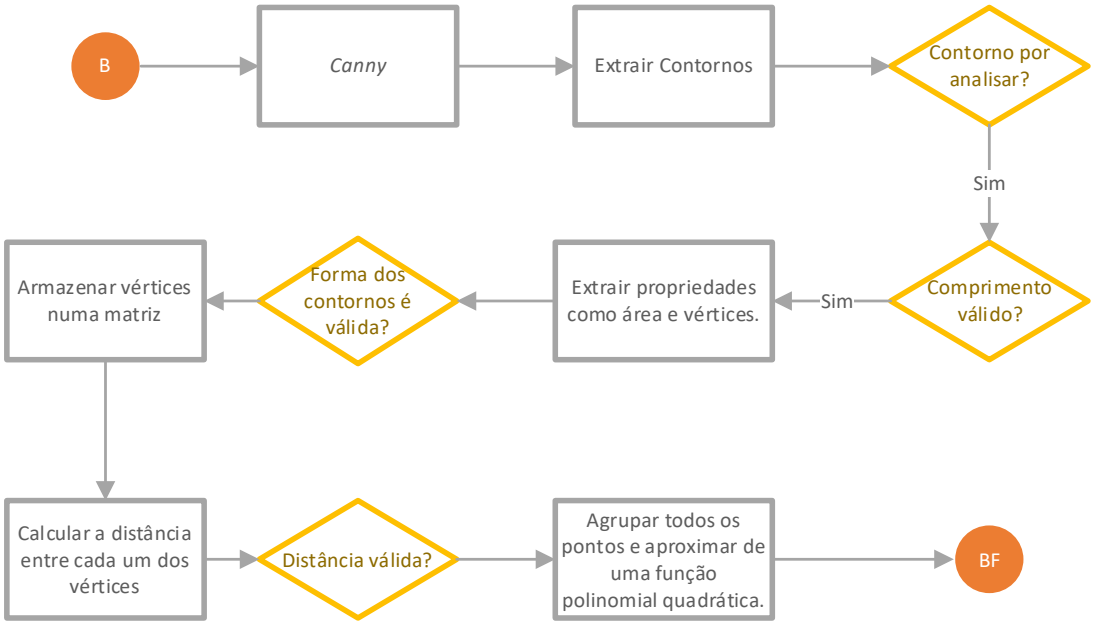


Figura 77 - Fluxograma do algoritmo B.

Nisto, inicia-se um processo de cálculo de distâncias entre todos os pontos existentes nessa matriz segundo a equação (10) para verificar que contornos são colineares.

$$d_{AB} = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2} \tag{10}$$

Segmentos de reta colineares, possuirão um conjunto de pontos (de cada segmento) onde a distância será relativamente inferior aos restantes, uma vez que estes se encontram na mesma linha, e consequentemente a distância entre eles é menor do que a largura da faixa de rodagem. Tendo como referência a Figura 78, onde se podem visualizar dunas linhas tracejadas na vertical, os pontos A-B são claramente mais próximos do que A-C por exemplo. Por forma a aumentar a eficiência, deve-se evitar calcular a distância dos pontos do mesmo contorno, ou seja, seguir a sequência também demonstrada na Figura 78. Além disso, deve-se sempre seguir a lógica dos números triangulares, isto é, para calcular as várias

combinações, o número de cálculos máximos não deve ser superior ao resultado da equação (11), em que n é o número de linhas do problema e T o número de cálculos máximo.

$$T_n = \frac{n(n+1)}{2} \tag{11}$$

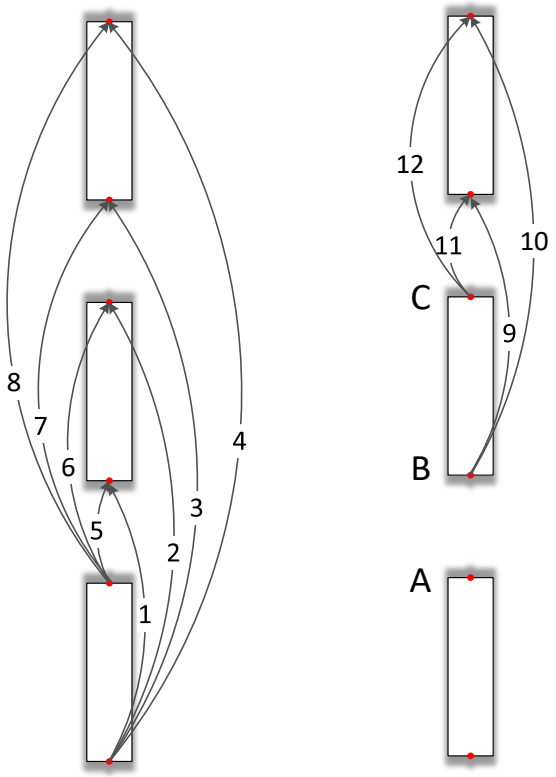


Figura 78 - Cálculo de distâncias.

Por fim, tendo agrupado todos os pontos válidos e identificado quais são "colineares", deve-se aproximar esses mesmos segmentos a uma equação polinomial de segundo grau para expressar o conjunto de uma forma matemática. O resultado pode ser observado na Figura 62 e Figura 63.

O terceiro e último algoritmo desenvolvido (algoritmo C), centra-se na combinação de diversas técnicas mencionadas na Secção 3.4. Numa primeira fase, este foca-se na busca por varrimento horizontal, ou seja, seguindo a lógica explicada na Secção 3.4.1, no entanto, com uma maior flexibilidade. Visto que este só tem como objetivo dar uma estimativa de posicionamento das linhas da faixa de rodagem junto do veículo, os limiares de deteção podem ser menos restritivos. Esta estimativa serve de parâmetro de entrada para a fase

seguinte, a utilização do algoritmo das caixas deslizantes (Secção 3.4.2). Ao usar a estimativa anterior como ponto inicial, não necessita de efetuar o cálculo do "histograma", que geralmente se encontra associado ao uso desta técnica. Daí em diante ele mantém toda a estratégia anteriormente descrita.

Todo este processo poderá ser visualizado no fluxograma apresentado na Figura 79, onde C representa o início do algoritmo e CF o final.

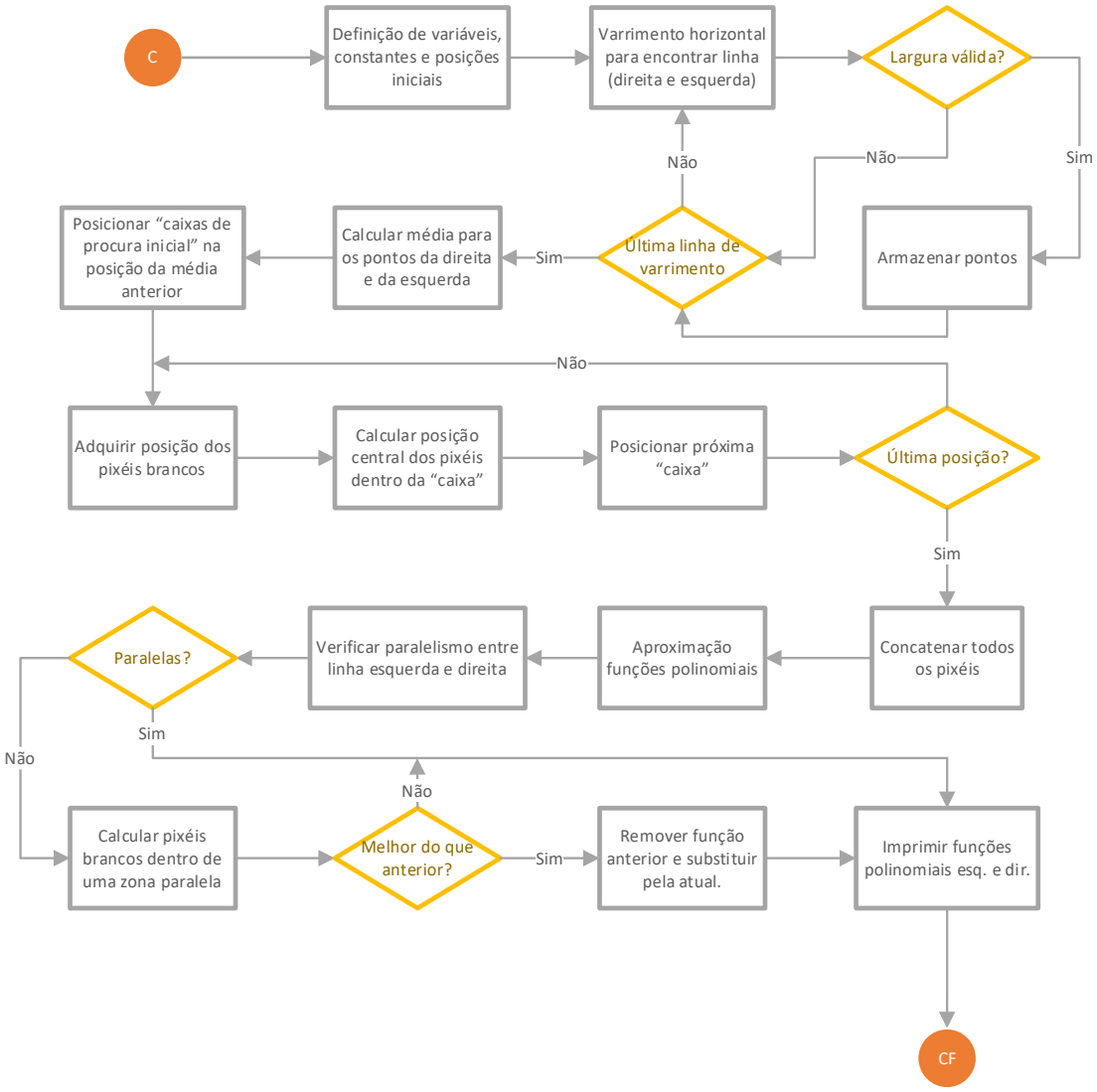


Figura 79 - Fluxograma algoritmo C.

Tendo todos os algoritmos a correr em paralelo, é necessário agora avaliar os seus resultados e tomar uma decisão baseada no grau de confiança de cada um deles e na coerência entre linhas.

Isto é, o grau de confiança é atribuído segundo um conjunto de validações de cada algoritmo, isoladamente, analisando as equações polinomiais geradas, através dos seus coeficientes. Numa primeira fase é verificado a sua curvatura, o objetivo é identificar se os raios são plausíveis e se enquadram nas curvas esperadas. Por exemplo, qualquer curva com raio inferior a 1,5m é automaticamente penalizada, e para isto basta verificar o valor do coeficiente de maior grau. Numa segunda fase, as penalizações são atribuídas consoante as posições das linhas, isto é, estas não devem surgir em locais inesperados. Tendo por base a Figura 80, quanto mais próximo do vértice do triângulo verde, as linhas começarem, maior será o seu grau de confiança.

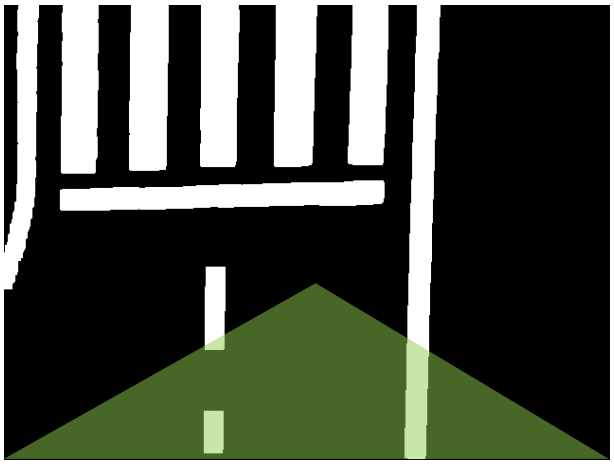


Figura 80 - Posição esperada das linha e respetivo grau de confiança.

Para tal deve-se ter em consideração as ordenadas na origem e as curvaturas, utilizando uma vez mais o coeficiente de maior grau. Por fim, são efetuadas comparações temporais, ou, por outras palavras, é calculado a discrepância das equações relativamente às últimas iterações, etapa que se baseia nas técnicas descritas na Secção 3.4.3.

Por outro lado, a coerência é avaliada comparando as curvas provenientes dos diversos algoritmos. Ou seja, verificando a harmonia entre as diversas hipóteses, resultado dos diversos algoritmos. Para tal, optou-se, uma vez mais, pela utilização do algoritmo desenvolvido na Secção 3.4.4, o algoritmo de verificação de paralelismo, no entanto, para validar a sobreposição das linhas, Figura 81, atribuindo um intervalo à esquerda e à direita da linha em estudo. Quando comparando com a linha verde, este vai calcular a média da distância à linha azul, isto porque, a linha vermelha encontra-se fora do limite aceitável e como tal não irá ser tomada em consideração. Quanto menor a distância média entre as linhas aceitáveis, maior será o grau de coerência.

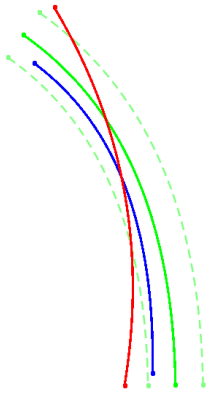


Figura 81 - Cálculo da distância média entre linhas para verificação da coerência.

O seu grau de coerência é atribuído em função da distância média entre linhas ao longo de um intervalo previamente determinado.

Por fim, a decisão é calculada subtraindo os grau de confiança às distâncias médias da coerência. Quanto mais baixo for o valor, mais assertiva tende a ser a hipótese. O fluxograma desta fase final pode ser observado na Figura 82.

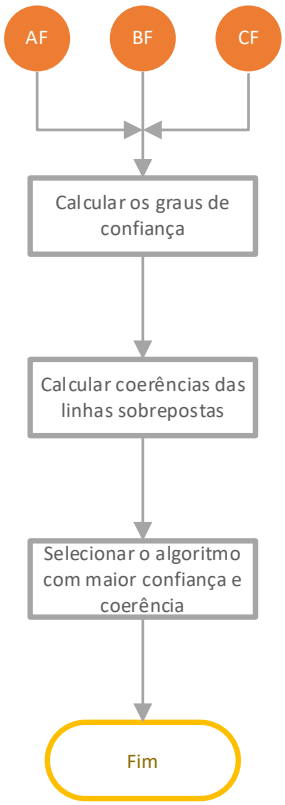
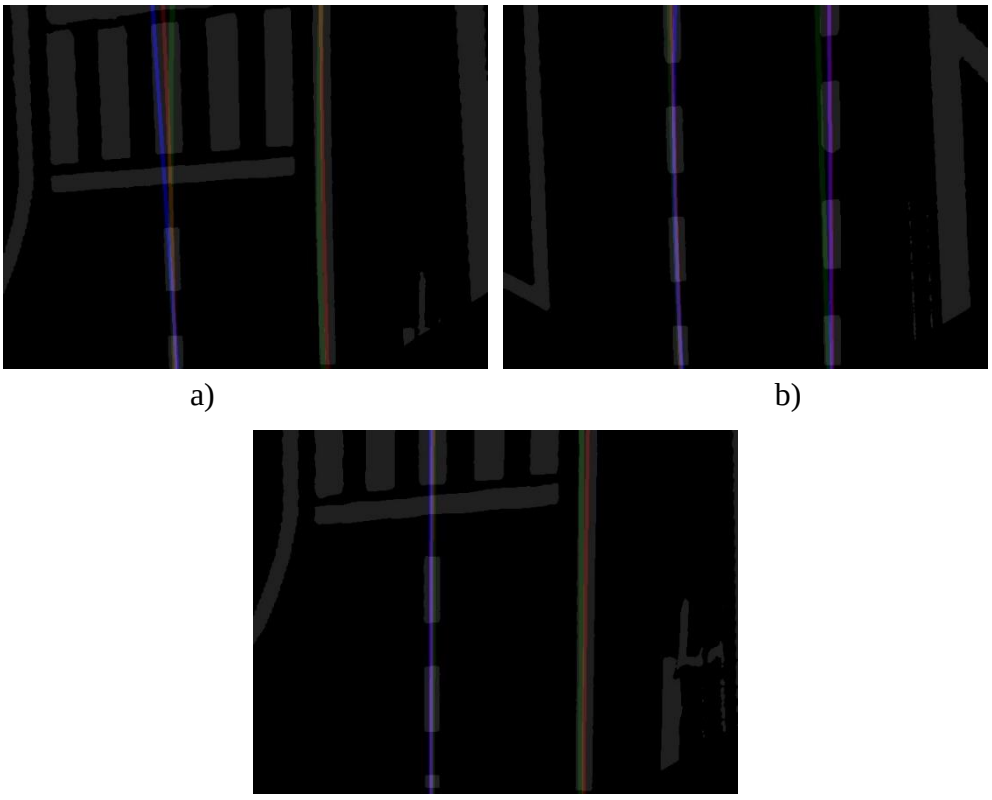


Figura 82 - Fluxograma de decisão de controlo.

A solução final é por sua vez enviada para o algoritmo de controlo e consequentemente para o hardware de baixo nível, responsável por atuar os servomotores e comandar a velocidade do veículo. Infelizmente estas etapas finais, **i.e.** o controlo do veículo não puderam ser testadas em ambiente real, tendo sido validadas apenas em simulação. O veículo de condução autónoma do clube de robótica, sendo um desenvolvimento em equipa, tem estado a sofrer alterações por outro grupo de trabalho ao nível do hardware, não estando operacional à data da realização dos testes e, como tal, não foi possível efetuar no veículo. O único algoritmo testado diretamente no veículo real em prova foi o algoritmo descrito na Secção 3.4.1, embora com ligeiras alterações, dado que foi esse o algoritmo que utilizado na edição de 2019 do Festival de Robótica e que permitiu alcançar o terceiro lugar[5]. Na Figura 83 encontra-se o resultado da combinação dos 3 algoritmos desenvolvidos em algumas das situações encontradas ao longo da pista, onde as linhas a vermelho representam o algoritmo A, as linhas a azul o algoritmo B e as linhas a verde o algoritmo C.



a) b)

c)

Figura 83 - Resultados de secções retas.

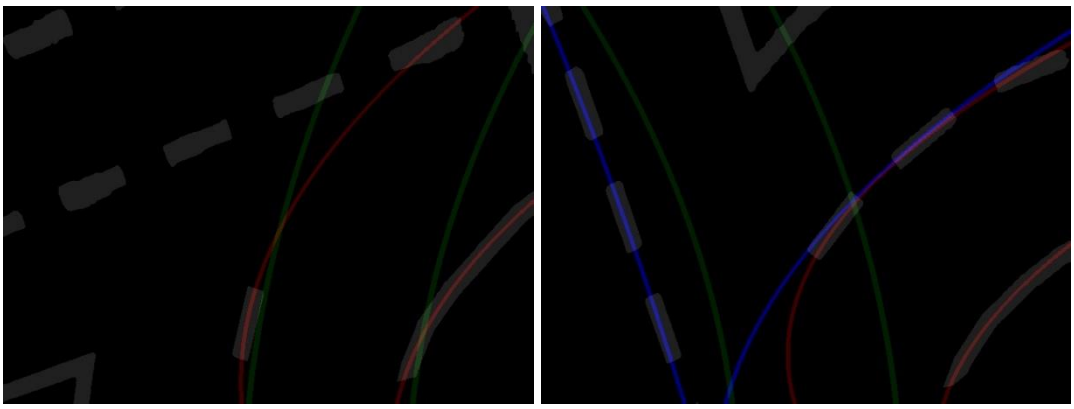
[5] https://www.ua.pt/pt/noticias/0/57946

Como se pode observar, situações de linha reta como as presentes nas Figura 83 a), b) e c) são relativamente fáceis de detetar, tendo apenas de ter atenção às questões das linhas tracejadas e da passadeira. A maioria dos algoritmos encontrados/desenvolvidos na comunidade científica conseguiam ter prestações satisfatórias nestas situações, tendo apenas de ajustar alguns pormenores devido à passadeira. A maioria dos algoritmos encontrados não está desenvolvida para detetar curvaturas mais acentuadas, mas sim desenvolvidos para a deteção em estradas de alta velocidade, como por exemplo, autoestradas.

a) b)

Figura 84 - Resultados de secções curvas.

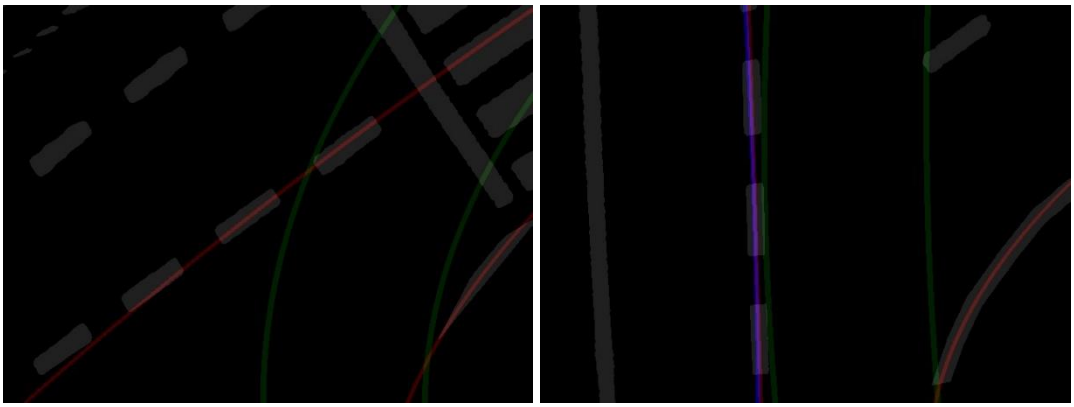
As Figura 84 a) e b) demonstram que o algoritmo final consegue detetar as curvas com bastante precisão, mesmo aquelas com raio menor.



a) b)

Figura 85 - Resultados em secções de curvas ocultas.

O maior desafio foi nas zonas onde a faixa tem linhas "ocultas", zonas em que, apesar de não existir uma linha desenhada, ela está lá virtualmente, como mostram as Figura 85 a) e b). É em situações como estas que o histórico e o paralelismo têm um peso muito grande, permitindo a validação de linhas praticamente inexistentes.



a) b)

Figura 86 - Resultados em secções com múltiplas opções.

Por fim, foi também possível detetar bifurcações da faixa de rodagem, apresentando as diversas alternativas, como mostram as Figura 86 a) e b) e a Figura 85 b). Desta forma, o algoritmo de decisão poderá optar por escolher a direção pretendida e descartar a outra deteção.

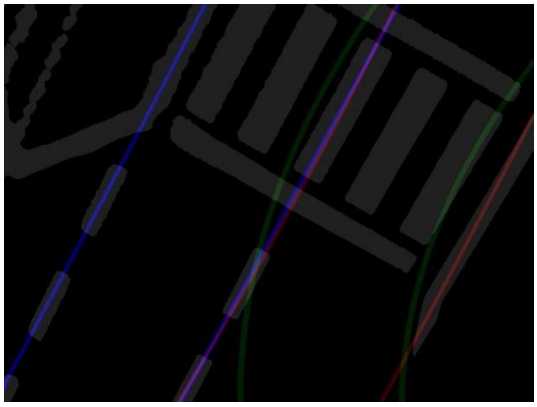


Figura 87 - Resultados da deteção de múltiplas faixas.

Em algumas situações é ainda possível detetar múltiplas faixas de rodagem, como mostra a Figura 87.

Esta página foi intencionalmente deixada em branco

4. Conclusões e Trabalho Futuro

Este projeto teve como objetivo o desenvolvimento de um algoritmo de deteção de linhas da faixa de rodagem. Este surgiu para colmatar alguns dos problemas levantados após a participação de 2019, do Clube de Robótica do Politécnico de Leiria, no Festival Nacional de Robótica.

Para melhorar os algoritmos de deteção, o primeiro passo foi o aumento do campo de visão, procedimento realizado com sucesso apesar do material de baixa custo. Deste processo resultou um artigo publicado e apresentado na conferência 2020 IEEE ICARSC, que inclui os métodos e processos utilizados durante a calibração da câmara frontal do veículo. Por forma a agilizar os algoritmos seguintes, foi também realizada uma transformação de perspetiva para vista aérea (*birds eye view*), a implementação deste procedimento encontra-se disponibilizado publicamente no *GitHub* com o designação de *MonoCamCalib4AD*.

No decorrer do desenvolvimento dos algoritmos de perceção e interpretação da faixa de rodagem, foram testados diversos algoritmos de pré-processamento de imagem e técnicas de identificação de linhas. A análise dos pré-processamentos resultou em soluções que permitiram isolar com sucesso todos os elementos pretendidos, soluções estas que podem também ser utilizadas no futuro para auxiliar métodos com inteligência artificial. Os algoritmos de deteção desenvolvidos, conseguem identificar com sucesso as linhas da faixa de rodagem ao longo de uma volta completa, incluindo zonas de passadeira e partes com linhas ocultas. No entanto, por motivos extrínsecos ao desenvolvimento, não se puderam executar testes *online* com o veículo real, facto este que impediu também o teste de algoritmos de controlo. Ainda assim, os testes efetuados com dados reais, previamente adquiridos, foram suficientes para comprovar o correto funcionamento do sistema de deteção de faixa de rodagem. A ausência de ensaios dinâmicos em tempo real não permite retirar conclusões assertivas, comportamentos inauditos do veículo poderão ter algum efeito inesperado.

Numa perspetiva crítica, dada as adaptações que tiveram de ser realizadas para as corretas deteções, pode-se concluir que este sistema pode estar sobreajustado ao cenário em questão tornando-o eventualmente menos robusto noutras configurações de pista/estrada, algo que, de acordo com a pesquisa efetuada é mais comum para abordagens baseadas em métodos clássicos, [22]. Neste sentido, poderá ser benéfico, no futuro, optar por uma abordagem

baseada em aprendizagem automática, nomeadamente redes neuronais, e perceber a sua dependência e desempenho sobre o cenário atual, comparando com cenários reais.

Bibliografia

[1] S. Bratzel, "Electromobility: Why Tesla teaches carmakers to fear | TIME ONLINE," *Zeit*, 2016.

[2] IPCB, "Robótica 2009 - Festival Nacional de Robótica." [Online]. Available: http://www.est.ipcb.pt/robotica2009/new/index.php?pagina=competicao. [Accessed: 08-Sep-2021].

[3] IBM Cloud Education, "What is Machine Learning?," *Machine Learning*, 15-Jul-2020. [Online]. Available: https://www.ibm.com/cloud/learn/machine-learning. [Accessed: 07-Sep-2021].

[4] "ROS.org | About ROS." [Online]. Available: https://www.ros.org/about-ros/. [Accessed: 07-Sep-2021].

[5] G. Boesch, "The 11 Most Popular Computer Vision Tools in 2021 - viso.ai," 27-Jun-2021. [Online]. Available: https://viso.ai/computer-vision/the-most-popular-computer-vision-tools/. [Accessed: 23-Sep-2021].

[6] C. Urmson, "TED Talk Transscript: How a driverless car sees the road", TEDx, 2015.

[7] European Commission, "2020 road safety statistics: what is behind the figures? | Mobility and Transport," 2020. [Online]. Available: https://ec.europa.eu/transport/modes/road/news/2021-04-20-road-safety-statistics-2020_en. [Accessed: 01-Sep-2021].

[8] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A Survey of Autonomous Driving: Common Practices and Emerging Technologies," *IEEE Access*, vol. 8, pp. 58443–58469, 2020.

[9] Synopsys, "The 6 Levels of Vehicle Autonomy Explained | Synopsys Automotive," *Synpsys.Com*, 2019. [Online]. Available: https://www.synopsys.com/automotive/autonomous-driving-levels.html.

[10] INE, "Predomínio do automóvel nas deslocações dos residentes das Áreas Metropolitanas de Porto e Lisboa - 2017," Porto - Lisboa, Jul. 2018.

[11] N. Larco, “How Will Autonomous Vehicles Transform Our Cities?,” in *TEDxCollegePark*, 2018.

[12] H. Ritchie and M. Roser, “Technology Adoption,” *Our World Data*, Oct. 2017.

[13] Legacy Research, “Blockchain’s Kicking Off the Fourth Great Tech Revolution.” [Online]. Available: https://www.legacyresearch.com/the-daily-cut/blockchains-kicking-off-the-fourth-great-tech-revolution/. [Accessed: 01-Sep-2021].

[14] A. Bar Hillel, R. Lerner, D. Levi, and G. Raz, “Recent progress in road and lane detection: A survey,” *Machine Vision and Applications*, vol. 25, no. 3. Springer Verlag, pp. 727–745, 2014.

[15] P. T. Mandlik and A. B. Deshmukh, “A review on lane detection and tracking techniques”, IJIERT, 2016.

[16] G. Indumathi and V. Sathananthavathi, “Microaneurysms detection for early diagnosis of diabetic retinopathy using shape and steerable gaussian features,” *Telemed. Technol. Big Data, Deep Learn. Robot. Mob. Remote Appl. Glob. Healthc.*, pp. 57–69, Jan. 2019.

[17] H. Jung, J. Min, and J. Kim, “An efficient lane detection algorithm for lane departure detection,” *IEEE Intell. Veh. Symp. Proc.*, pp. 976–981, 2013.

[18] S. Jung, J. Youn, and S. Sull, “Efficient lane detection based on spatiotemporal images,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 289–295, Jan. 2016.

[19] J. Guo, Z. Wei, and D. Miao, “Lane Detection Method Based on Improved RANSAC Algorithm,” *Proc. - 2015 IEEE 12th Int. Symp. Auton. Decentralized Syst. ISADS 2015*, pp. 285–288, Apr. 2015.

[20] M. Virgo, “Lane Detection with Deep Learning,” *Towar. Data Sci.*, May 2017 [Online]. Available: https://towardsdatascience.com/lane-detection-with-deep-learning-part-1-9e096f3320b7 [Acessed: 16-Jun-2021]

[21] W. Van Gansbeke, B. De Brabandere, D. Neven, M. Proesmans and L. Van Gool, "End-to-end Lane Detection through Differentiable Least-Squares Fitting," 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 905-913, doi: 10.1109/ICCVW.2019.00119.

[22] Izquierdo, Asier & Lopez-Guede, Jose & Graña, Manuel. (2019). Road Lane Landmark Extraction: A State-of-the-art Review. 10.1007/978-3-030-29859-3_53.

[23] Jèahne, B. and Haussecker, H., 2000. Computer vision and applications. San Diego: Academic Press, p.679.

[24] WebGL, "WebGL 3D Perspective." [Online]. Available: https://webglfundamentals.org/webgl/lessons/webgl-3d-perspective.html. [Accessed: 01-Sep-2021].

[25] S. Sels, B. Ribbens, S. Vanlanduit, and R. Penne, "Camera Calibration Using Gray Code," Sensors, vol. 19, no. 2, p. 246, Jan. 2019.

[26] O. C. and T. P. and P. Sturm, "The Geometric Error for Homographies," *Comput. Vis. Image Underst.*, vol. 97, no. 1, pp. 86–102, Jan. 2005.

[27] R. C. Gonzalez, "Image Thresholding by OPEN_CV," *Image Tresholding*, 02-Apr-2021. [Online]. Available: https://docs.opencv.org/4.5.2/d7/d4d/tutorial_py_thresholding.html. [Accessed: 03-Jul-2021].

[28] X. Dong, "Median Filter in AWK," 24-Jun-2015. [Online]. Available: http://onetipperday.sterding.com/2015/06/median-filter-in-awk.html. [Accessed: 12-Sep-2021].

[29] "Tracking camera T265 – Intel RealSense Depth and Tracking Cameras." [Online]. Available: https://www.intelrealsense.com/tracking-camera-t265/, 2020.

[30] P. Hausamann, C. B. Sinnott, M. Daumer, and P. R. Macneilage, "Evaluation of the Intel RealSense T265 for tracking natural human head motion," *Sci. Reports* |, vol. 11, p. 12486, 2021.

[31] Ag2gaeh, "Offset Definition Poss." CC-BY 20 license. Nov 2015 [Online]. Available: https://commons.wikimedia.org/wiki/File:Offset-definition-poss.svg [Acessed: 2-Set-2021]

[32] D. Ding, C. Lee, and K. Y. Lee, "An adaptive road ROI determination algorithm for lane detection," in *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, 2013.

[33] R. O. Duda and P. E. Hart, “Use of the Hough Transformation to Detect Lines and Curves in Pictures,” *Comm. ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972.

[34] Daf-de, “Hough Example.” CC-BY 2.0 license. Aug 2006 [Online]. Available: https://commons.wikimedia.org/wiki/File:Hough-example-result-en.png [Acessed: 15-Aug-2021]

[35] J. Canny, “A Computational Approach to Edge Detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 6, 1986.

[36] S. Suzuki and K. A. be, “Topological structural analysis of digitized binary images by border following,” *Comput. Vision, Graph. Image Process.*, vol. 30, no. 1, pp. 32–46, Apr. 1985.

[37] F. R, P. S, W. A, and W. E, “Morphology - Thinning,” *HYPERMEDIA IMAGE PROCESSING*. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/thin.htm. [Accessed: 04-Sep-2021].

[38] P. H. S. Torr and A. Zisserman, “MLESAC: A New Robust Estimator with Application to Estimating Image Geometry,” *Comput. Vis. Image Underst.*, vol. 78, no. 1, pp. 138–156, Apr. 2000.

Anexo A

Monocular Camera Calibration for Autonomous Driving — a comparative study

Pedro Filipe Martins*, Hugo Costelha*†, Luis Conde Bento*‡ and Carlos Neves*§
*ESTG, Politécnico de Leiria, Portugal
Email: {pedro.r.martins, hugo.costelha, luis.conde, carlos.neves@ipleiria.pt}@ipleiria.pt
†INESC TEC, Porto, Portugal
‡ISR-UC, Coimbra, Portugal
§INESCC, Coimbra, Portugal

***Abstract*—Autonomous driving is currently a widely researched topic worldwide. With a large research effort being taken by industrial research units in the automotive sector, it is no longer exclusive to academic research labs. Essential to this ongoing effort towards level-5 vehicle autonomy, are the sensors used for tracking and detection, mainly lasers, radars and cameras. Most of the cameras for automotive application systems use wide-angle or fish-eye lens, which present high distortion levels. Cameras need to be calibrated for correct perception, particularly for capturing geometry features, or for distance-based calculations. This paper describes a case-study concerning monocular camera calibration for a small scale autonomous driving vehicle vision system. It describes the fundamentals on camera calibration and implementation, with results given for different lenses and distortion models. The aim of the paper is not only to provide a detailed and comprehensive review on the application of these calibration methods, but to serve also as a reference document for other researchers and developers starting to use monocular vision in their robotic applications.**

***Index Terms*—Autonomous Driving, Camera Calibration, Robotics.**

I. Introduction

Research on autonomous vehicles, particularly ground vehicles, is currently a very hot topic, with many research institutions and companies working on the subject. Perception is key for autonomous driving to be successful, which means having to process information from various sensors in order to perceive the world.

Regarding the vehicle surroundings, LIDARs, radars and cameras are the most common used sensors for perception in autonomous vehicles. Each sensor type shows different degrees of performance, in terms of their range, resolution and accuracy, for different environment conditions, namely weather and day/night conditions [1]. Therefore, a good perception system is obtained when these sensors are combined [2].

Due to its high range and precision, high-end LIDARs continue to be the main sensors used for autonomous vehicle driving, particularly for level-5 autonomy vehicles. However, these have a higher cost when compared to radars or cameras. Radar can provide long range awareness and is able to provide reliable sensor measurements in harsh environments, but it lacks the fidelity and information richness. Cameras provide for a lower cost and a large capabilities set, with many developments being made on using monocular vision for additional cost reductions [3], [4], particularly when used together with deep learning approaches [5].

This work summarizes the main camera models used for perspective cameras (narrow and wide-angle), and fisheye cameras, describing the various distortion parameters involved. It provides results on the calibration of a wide-angle camera setup and a fisheye lens setup, comparing the use of various distortion models. The main goal of this paper is to provide in a single document an overview on monocular camera calibration, serving as a starting point for researchers beginning to work with monocular vision.

This paper is organized as follows: Section I provides for the introduction, Section II describes existing camera models and calibration approaches, while Section III details the experiments and results obtained. Finally, Section IV provides conclusions and future work.

II. Camera Models and Calibration

This section provides a summary of existing camera models, focusing on monocular cameras, and considering the most common approach as a starting point, as described by Brown in [6].

A. Perspective Camera Models

A camera consists mainly on a planar optical sensor and a lens, together with the electronics needed to obtain the digital image of the scene projected onto the optical sensor. Considering a pinhole camera model, and given the camera pose in world coordinates, a 3D point in space is projected onto a 2D point in the camera image, as show in Fig. 1. This projection model can be represented by Eq. 1-2 [7].

A point $p_w = [x_w, y_w, z_w]^T$ in world coordinates, corresponds to a point $p_c = [x_c, y_c, z_c]^T$ in camera coordinates through Eq. 1.

$$p_c = [R|t]\, p_w \Leftrightarrow$$
$$\Leftrightarrow \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}, \quad (1)$$

This project was partially financed by the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, through national funds, and co-funded by the FEDER, where applicable

Figura 88 - Página frontal do artigo publicado no ICARSC