INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Virtual Driver - Sistema de Reconhecimento de Sinalização Vertical de Trânsito

RICARDO JORGE DE CASTRO PINHEIRO RAMALHO
Outubro de 2017

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Virtual Driver

Reconhecimento de Sinalização de Vertical de Trânsito

Ricardo Jorge de Castro Pinheiro Ramalho

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas Gráficos e Multimédia

Orientador: Prof. Doutor António Vieira Castro

Júri:

Presidente:

[Nome do Presidente, Categoria, Escola]

Vogais:

[Nome do Vogal1, Categoria, Escola]

[Nome do Vogal2, Categoria, Escola] (até 4 vogais)

Porto, outubro de 2017

Dedico à minha namorada,

à minha família e

aos meus amigos.

Resumo

No quotidiano das sociedades atuais, a sinalistralidade automóvel permanece como uma das principais causas de morte a nível mundial, vitimando todos os anos milhões de pessoas e provocando danos fisicos irreversíveis, que resultam em incapacidades totais ou parciais.

Com o crescimento sustentado das tecnologias inteligentes de transporte, denominadas *"ITS-Intelligent Transportation Systems"*, verifica-se que as mesmas são um excelente meio de atuaçãona prevenção de acidentes rodoviários, particularmente aquelas que auxiliam directamente o condutor dentro do veículo "*ADAS – Advanced Driver Assistance System".*

A introdução dos sistemas ADAS no mercado tarda em profilerar, devido a restrições como o custo e disponibilidade, já que normalmente a sua disponibilização é influenciada pelo foco comercial dos fabricantes de automóveis e está inerente à utilização de carros de gama alta ou à aquisição de extras dispendiosos. Considera-se também, que os requisitos tecnológicos associados a estas soluções são um entrave à sua implementação, uma vez que a idade média dos automóveis é elevada. O défice de alcance deste tipo de soluções pela população surge como motivação desta dissertação, para o desenvolvimento de um sistema capaz de ser acessível por todos.

Esta dissertação tem como objetivo principal a demonstração de uma investigação na área dos sistemas inteligentes de transporte, sendo base para o desenvolvimento de um protótipo de um sistema evolutivo de assistência ao condutor, denominado Virtual Driver.

A sinalização de trânsito é o principal meio de comunicação entre o condutor e estrada, definindo quais são as regras e perigos inerentes na sua utilização. No entanto, a sua não visualização por fatores como distração, falta de foco na condução ou falta de visibilidade, podem comprometer a segurança do próprio condutor e de outros presentes na via, resultando em acidentes rodoviários.

Virtual Driver é uma solução de baixo custo que é executável através de dispositivos presentes no quotidiano, como o *smartphone* ou outros micro-computadores. Desta forma, permite, numa primeira fase, auxiliar o condutor através do reconhecimento de sinalização vertical de trânsito em tempo real, alertando-o através de informações áudio-visuais. O protótipo desenvolvido atua com base na sinalização vertical de trânsito Portuguesa, pretendendo-se, mais tarde, abranger outras realidades.

Numa fase final, será apresentada a avaliação do desempenho da solução, relativamente aos processos de deteção e reconhecimento de sinalização, evidenciando a utilização da metodologia escolhida.

Palavras-chave: sistemas inteligentes de transporte, sistemas avançados de assistência ao condutor, reconhecimento de sinalização de trânsito vertical.

Abstract

In today's society, road accidents remain one of the leading causes of death worldwide, killing millions of people each year and causing irreversible physical damage, resulting in total or partial disability.

With the sustained growth of Intelligent Transportation Systems (ITS-Intelligent Transportation Systems), it's now known that they are an excellent mean for preventing road accidents, particularly those that directly assist the driver in the vehicle "ADAS - Advanced Driver Assistance System".

The introduction of ADAS systems on the market is slow to take, due to constraints such as cost and availability, as their availability is usually influenced by the commercial focus of car manufacturers, and is inherent in the use of high-end cars, or the purchase of costly extras.

It is also considered that the technological requirements associated with these solutions are an obstacle to their implementation, since the average age of cars is high. The lack of reach of this type of solutions by the population, appears as motivation of this dissertation, for the development of a system capable of being accessible by all.

This dissertation has, as main objective, the demonstration of an investigation in the area of the intelligent transport systems, being the base for the development of a prototype of an evolutive system of assistance to the driver, denominated Virtual Driver.

Traffic signaling is the main mean of communication between the driver and the road, defining the rules and dangers inherent in their use. However, their non-visualization, by factors such as distraction, lack of focus on driving or lack of visibility, can compromise the safety of the driver himself and others present on the road, resulting in road accidents.

Virtual Driver is a low cost solution that is executable through everyday devices such as smartphones or other micro-computers. In this way, it allows, in a first stage, to assist the driver through the recognition of vertical traffic signaling in real time, alerting him through audio-visual information. The developed prototype acts based on the Portuguese vertical traffic signalization, aiming later to cover other realities.

In a final phase of this dessertation will be presented the evaluation of the performance of the solution, regarding the processes of detection and recognition of signaling, highlighting the use of the chosen methodology.

Keywords: intelligent transport systems, advanced driver assistance systems, vertical traffic sign recognition system

Agradecimentos

Deixo aqui uma palavra de apreço e de total gratidão para todos aqueles que me apoiaram direta ou indiretamente no processo de investigação e desenvolvimento da dissertação abaixo descrita, bem como, a todos aqueles que me incentivaram, e acompanharam a dedicação diária na resolução da problemática abordada.

Fica também o meu grande agradecimento ao Professor Doutor António Vieira de Castro, pela orientação e dedicação ao projeto Virtual Driver. Sem particularizar, agradeço a todos os docentes do MEI pelo apoio prestado ao longo do desenvolvimento da dissertação.

Agradeço de forma especial à minha namorada, que me acompanhou e incentivou em todos os momentos desta caminhada.

Aos meus pais minha e à minha irmã, que sempre acreditaram no meu trabalho, e foram pilares no meu percurso académico.

À restante família e amigos, um obrigado pela amizade de sempre.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

ITS	*Inteligent Transportation System*
ADAS	*Advanced Driver Assistance System*
TSR	*Traffic Sign Recognition*
LDW	*Lane Departure Warning*
FCW	*Forward Collision Warning*
VRU	*Vulnerable Road Users*
RGB	*Red, Green and Blue*
HSL	*Hue Saturation Lightness*
HSV	*Hue Saturation Value*
HSI	*Hue Saturation Intensity*
FAST	*Features from Accelerated Segment Test*
MSER	*Maximally Stable Extremal Regions*
SIFT	*Scale-Invariant Feature Transform*
SURF	*Speeded Up Robust* Features
ORB	*Oriented FAST and Rotated BRIEF*
BRISK	*Binary Robust Invariant Scalable Keypoints*

1 Introdução

"You, your vehicle and the technologies!"[1]

Comissão Europeia, 27 de Janeiro de 2015

1.1 Contexto

Nos dias de hoje, os acidentes rodoviários surgem como uma das principais causas de morte a nível mundial, constando no *top* 10 de causas de morte em 2012 de acordo com a Organização Mundial de Saúde (OMS).

No Relatório Global Sobre o Estado da Segurança Rodoviária de 2015 (WHO, 2016), as lesões provocadas por acidentes rodoviários, são apresentadas como a principal causa de mortes entre os jovens na faixa etária entre os 15-29 anos (estatística de 2012), sendo que, a nível global, vitimaram cerca de 1,25 milhões de pessoas em 2013. Cerca de metade das mortes registadas estão relacionadas com peões, ciclistas e motociclistas (49%), o restante respeitante a mortes relacionadas com ocupantes de veículos e outros.

Ainda assim, os valores globais destes indicadores estabilizaram desde 2007, face ao aumento da população e da produção automóvel:

> [...] *A população aumentou 4% entre 2010 e 2013 e houve um aumento de 16% do número de veículos no mesmo período, isso sugere que as intervenções implementadas nos últimos anos, para melhorar a segurança no trânsito em nível mundial, têm salvado vidas* […] (WHO, 2016)

Em geral a realidade Europeia de mortes nas estradas é menos grave comparativamente com países em desenvolvimento e realidades em que a legislação e tecnologia ainda carecem de

[1] https://ec.europa.eu/digital-single-market/en/you-your-vehicle-and-technologies

bastante progresso. No ano de 2015 registaram-se segundo a base de dados europeia de acidentes (CARE) 26 100 mortes nas estradas dos países da União Europeia e em Portugal 593 fatalidades (European Commission, 2016). A estes números somam-se os números de todos aqueles que ficaram parcialmente ou totalmente incapacitados para o resto da vida. Segundo a estatística da Comissão Europeia:

> *[...] Por cada morte nas estradas europeias, estima-se 4 feridos com incapacidade permanente com danos cerebrais ou mazelas na medula espinhal, 8 lesões graves e 50 feridos leves[…]* (European Comission, no date) [2]

1.1.1 Fator Humano

Diferentes estatísticas apontam, como principal causa dos acidentes rodoviários, o erro Humano durante a condução. Estima-se que, 90% dos casos estejam relacionados com o fator Humano, sendo que poderão ser combinados com outro tipo de erros, como as condições do ambiente e do veículo (Figura 1).(Volvo Trucks, 2013) Um exemplo de um erro Humano que poderá trazer consequências bastante graves, é o não respeito pela sinalização de paragem obrigatória (STOP).

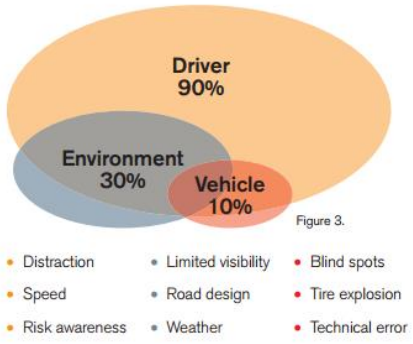


Figura 1 – Fator Humano como principal causa de acidentes de viação[3]

Apesar das medidas implementadas pelos governos e organizações mundiais, o número total de vítimas de acidentes de trânsito é ainda bastante substancial e, existirá sempre necessidade de atuação para que o número possa tender para zero.

A introdução das tecnologias da informação e comunicação no ramo automóvel, surge como elemento chave para o desenvolvimento de soluções e automóveis inteligentes, capazes de

---

[2] Tradução livre do autor. No original "… *For every death on Europe's roads there are an estimated 4 permanently disabling injuries such as damage to the brain or spinal cord, 8 serious injuries and 50 minor injuries*."

[3] Imagem retirada de: http://www.volvotrucks.com/SiteCollectionDocuments/VTC/Corporate/Values/ART%20Report%202013.pdf

alertar o condutor e atuar em situações de perigo iminente, prevenindo acidentes e diminuindo o número de mortes.

Tendo em conta a realidade descrita, esta dissertação surge como base para o estudo, investigação e conceção de um protótipo na área dos Sistemas Inteligentes de Transporte (ITS), nomeadamente aquele que permita auxiliar o condutor (Sistema Avançado de Apoio ao Condutor – ADAS) na vertente de reconhecimento de sinalização de trânsito vertical (*TSR – Traffic Sign Recognition*).

O projeto será realizado no seio do Laboratório Multimédia – LAMU do Departamento de Engenharia Informática do Instituto Superior de Engenharia do Porto, que permita culminar numa solução adequada para o problema identificado, permitindo atuar de forma positiva neste domínio.

1.1.2 Problema

A introdução dos sistemas inteligentes de transporte *(ITS – Intelligent Transportation Systems),* nomeadamente aqueles que auxiliam o condutor [os sistemas avançados de assistência ao condutor *(ADAS – Advanced Driver Assistance System)]*, no setor automóvel, são uma mais-valia para o aumento da segurança e redução de acidentes de trânsito. No entanto, a introdução de sistemas deste tipo no mercado, tarda ainda em proliferar.

Segundo o estudo (McKinsey & Company 2016) existem vários fatores que têm atrasado a implementação destas soluções, como: sistemas ainda em fase de desenvolvimento e afinação; os fatores custo e disponibilidade, uma vez que a maioria dos ADAS estão presentes apenas em carros de topo, só ao alcance de alguns e, principalmente, o desconhecimento dos compradores automóveis relativamente à existência deste tipo de soluções.

O foco comercial dos fabricantes do setor automóvel (European Commission, 2002), bem como os requisitos económicos e tecnológico, têm sido um entrave à implementação em grande escala destes sistemas. A existência de automóveis mais antigos, e que de origem não estão equipados com ADAS, representam uma fatia significativa dos veículos em circulação, para os quais subsiste, à data, necessidade de atuação.

Em Portugal a idade média dos veículos a motor de passageiro, é, segundo estatística de 2015, de 12,4 anos (Instituto Nacional de Estatística, 2015). De que forma poderão estes veículos serem sujeitos a sistemas inteligentes?

A sinalização rodoviária é extremamente importante na comunicação entre a estrada e o condutor, já que, permite estabelecer uma ordem e lei para a utilização da via pública, de forma segura e regrada. Por exemplo, a sinalização de cedência de passagem, limites de velocidade ou passagem de peões. Contudo, os sinais apenas serão úteis se os intervenientes os observarem (Waite and Oruklu, 2012).

A visualização de um sinal de trânsito pode estar comprometida por diversos fatores como, a distração, desconcentração, foco na condução, ou falta de visibilidade num determinado instante, que, consequentemente, estão na origem de um acidente de viação.

A utilização de um sistema de *Traffic Sign Recognition* poderá ser a chave para o problema.

Será possível o desenvolvimento de um sistema de baixo custo, acessível a qualquer automóvel, que em tempo real, reconheça e alerte o condutor sobre sinalização rodoviária vertical, e permita contribuir na prevenção de acidentes rodoviários?

1.2 Objetivo

Esta dissertação tem como principal objetivo o desenvolvimento de uma solução do tipo ITS, para um sistema avançado de apoio ao condutor (ADAS), que permita reconhecer sinalização vertical de trânsito e que comunique informações em tempo real. Apesar de já existirem algumas soluções no mercado, verifica-se que a sua disponibilidade está correlacionada com restrições relativas ao automóvel (ano, marca, gama), à tecnologia e ao custo de aquisição.

Esta solução tem como principal missão, contribuir para o aumento da segurança rodoviária em Portugal, consciencializando os automobilistas para a melhoria dos comportamentos na estrada.

O projeto deverá conceber um sistema em forma de protótipo denominado *Virtual Driver* (fase 1) que deverá ser extensível e evolutivo, segundo orientação futura do Laboratório Multimédia (LAMU) do Instituto Superior de Engenharia do Porto (ISEP). O LAMU assume aqui uma função pedagógica, e direciona o seu foco para a contribuição social na prevenção e diminuição da sinistralidade rodoviária.

É pretendido que, ao longo deste documento, sejam evidenciados desenvolvimentos dos seguintes pontos:

- Compreensão de conceitos e técnicas de sistemas de condução inteligente;
- Análise das soluções existentes no Mercado;
- Entendimento e investigação de algoritmos de reconhecimento de imagem;
- Identificação de bibliotecas, *API* e linguagens com potencial para a solução a desenvolver;
- Avaliação da integração de Sistemas *Text to Speech* para fornecimento de informações auditivas ao condutor;
- Desenvolvimento de componente de comunicação por áudio com o condutor;
- Desenvolvimento de um protótipo denominado Virtual Driver em fase 1 (Reconhecimento de Sinalização Vertical);

- Validação e Testes Funcionais do protótipo;
- Avaliação de solução para resolução do problema.

A implementação da solução assumirá alguns pressupostos para primeira fase.

O catálogo de sinais a reconhecer será restrito a uma seleção do universo de sinalização incluída no código de estrada Português. O reconhecimento dos mesmos será desenvolvido apenas para ambiente diurno.

1.3 Resultados Esperados

Pretende-se que o protótipo *Virtual Driver* consiga ser avaliado numa situação simulada ou real, demonstrando capacidades no reconhecimento de sinalização vertical de trânsito, através de captura de vídeo, e alertando o condutor com informações relevantes em tempo real.

O potencial deste sistema será idealmente medido pela capacidade de prevenir eventuais falhas no processo de condução Humana.

1.4 Análise de Valor

Os ADAS posicionam-se atualmente como uma área em crescimento e, consequentemente, um foco de interesse económico no setor automóvel. Previsões apontam para um possível crescimento do mercado dos sistemas *ADAS* acima dos 50% de taxa anual *CAGR (Compound Annual Growth Rate*) até 2018 (Smithers Group, 2014).

Este mercado, avaliado em cerca de 18,5 mil milhões de dólares, poderá atingir os 165 mil milhões de dólares até 2020 (Smithers Group, 2014). Focando no subconjunto dos Sistemas de Reconhecimento de Sinalização de Trânsito (*TSR*), o mercado poderá crescer cerca de 29,83% *GAGR* entre 2016 e 2020 (Research and Markets, 2016).

Os sistemas *TSR* estão historicamente relacionados com automóveis de gama alta.

Para além da possibilidade de inclusão destes sistemas no automóvel ser exclusiva a modelos de topo, os mesmos não vêm na versão base do fabricante, sendo necessário, na maioria das marcas, pagar por um *pack* extra, que incluí, não apenas os TSR, mas também outras funcionalidades que, no final, inflacionam o preço total para aquisição dos mesmos.

Por exemplo, nas gamas superiores da *Audi*, o sistema *TSR* poderá ser adquirido em Portugal por 355€ que implica a aquisição de um pacote mínimo de navegação de 1765€[4].

Pretende-se que o *Virtual Driver* concretize a conceção de um *software* dotado das funcionalidades que atualmente alguns dos fabricantes já disponibilizam nos seus sistemas *TSR,* tais como:

- Reconhecimento em tempo real de sinalização vertical de trânsito através da captura de vídeo;
- Disponibilização de informações audiovisuais relevantes para o condutor para uma correta tomada da decisão (Limites de velocidade, Proibições, Direções e Outras informações).

Numa perspetiva de análise de valor, e identificando de forma clara o benefício para o cliente, é definido como proposta de valor, a disponibilização de uma aplicação que permite atuar como um sistema do tipo *TSR.*

Pretende utilizar, para a sua operação, recursos de baixo custo, possibilitando a utilização em qualquer automóvel, independentemente da marca, gama, ano de fabrico e tecnologia.

Desta forma, o utilizador final conseguirá obter o *software* por um custo baixo (uso de dispositivo móvel como *smartphone/tablet/*microcomputador), que facilmente utilizará no seu automóvel.

1.4.1 Vantagem Competitiva

De acordo com a estratégia defendida por (Porter, 1980), a vantagem competitiva de um produto sustenta-se em dois princípios: o baixo custo e/ou a diferenciação.

Neste projeto, o princípio de baixo custo assumirá papel preponderante, pois pretende-se que a solução desenvolvida permita atingir também um subconjunto de condutores que, por restrições económicas e tecnológicas, não conseguiriam dotar os seus veículos de um sistema inteligente de apoio à condução, garantindo assim a sua distribuição.

O desenvolvimento da solução deverá estar comprometido com a garantia da qualidade apresentando altas taxas de reconhecimento, só assim poderá competir com outras soluções existentes.

[4] Informação presente no configurador Audi, http://configurator.audi.pt/controller?next=equipment-page&mandant=accx-pt

1.4.2 Criação de Valor

A criação de valor, com a introdução da solução Virtual Driver no mercado, chega pela capacidade de o sistema avançado de apoio à condução para reconhecimento de sinalização vertical, ser executado através de recursos comuns de baixo custo para os utilizadores, como dispositivos móveis do tipo *smartphone*, *tablet*, microcomputadores ou outros.

Desta forma, deverá ser possível alcançar uma maior taxa de utilização, por redução das barreiras tecnológicas do veículo, custo de aquisição dos sistemas ADAS, e por contribuição para a prevenção rodoviária Portuguesa.

Na perspetiva do que representa o valor no negócio, adotou-se a divisão sugerida no módulo de Análise de Valor do mestrado de Engenharia Informática do ISEP.

O esquema abaixo (Figura 2) pretende destacar os elementos chave que compõem tanto o valor de venda para o cliente, como o valor de venda para o negócio.

Ressalva-se que a nível do *Selling Business*, representam-se as mais-valias que vão de encontro à missão do projeto.

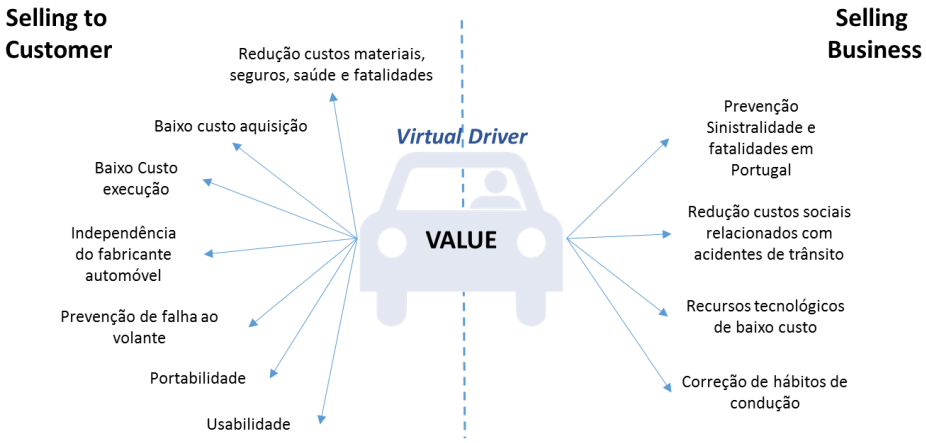


Figura 2 – O que o valor representa no negócio

1.5 Abordagem Preconizada

O projeto *Virtual Driver* incide sobre o desenvolvimento de um protótipo de um Sistema Avançado de Assistência ao Condutor (ADAS).

Será concebida uma aplicação sobre plataforma móvel em forma de protótipo, que deverá contemplar cada uma das etapas de um sistema do tipo *TSR*:

1. Captura de vídeo em tempo real;
2. Deteção e Reconhecimento de sinalização;

3. Categorização de sinais de trânsito;
4. Disponibilização de informação audiovisual ao condutor;
5. Alimentação de base de conhecimento.

O sistema a desenvolver deverá ter como base para categorização o catálogo de sinais de trânsito do código de estrada Português[5]. Esse catálogo deverá ser alimentado ao longo do tempo, e a informação perseverada e armazenada num repositório de dados.

A aplicação a desenvolver deverá ser possível de ser executada num dispositivo móvel como um *smartphone*, *tablet*, microcomputador ou outros.

Será uma vantagem, que tenha a capacidade de ser desenvolvido em ambiente multiplataforma

[5] Informação do catálogo Português de sinalização presente em:
http://www.infraestruturasdeportugal.pt/rede/rodoviaria/seguranca-rodoviaria/normas-de-sinalizacao

2 Contexto e Estado da Arte

2.1 Introdução

Este subcapítulo pretende descrever de forma sucinta e objetiva o contexto e o propósito da dissertação apresentada.

O mesmo divide-se em quatro partes: Definição detalhada do problema, Conceitos de Negócio, Processos e Atores e Restrições.

2.2 Descrição do Problema

Para uma melhor compreensão da problemática atualmente existente, origem da motivação para realização deste projeto, será detalhado o problema identificado.

Por forma a que o leitor consiga compreender todos os aspetos do problema, serão usadas as melhores práticas de Engenharia com base no livro *Engineering Reasoning*, dando resposta às questões e disciplinas definidas no "*The Thinker's Guide to Engineering Reasoning*" (Paul, Niewoehner and Elder, 2006).

2.2.1 Propósito de Engenharia

Nesta seção é descrito o propósito de engenharia para a resolução do problema identificado.

Qual é o propósito desta solução?

Com base na avaliação do estado da arte, em relação aos sistemas *ADAS* e às tecnologias de reconhecimento de sinalização, pretende-se conceber, através de um raciocínio de engenharia, um protótipo de um sistema avançado de assistência ao condutor, capaz de apoiar a condução, e que alerte atempadamente para uma correta tomada de decisão.

A solução "Virtual Driver" deverá ser implementada num cenário simulado ou real, permitindo a captação de imagem, reconhecendo sinais de trânsito do código da estrada Português e transmitindo informações audiovisuais ao condutor.

Pretende-se que o sistema, uma vez implementado, contribua ativamente para a diminuição da sinistralidade automóvel e, consequentemente, possibilite a diminuição das fatalidades e lesões causadas por acidentes de automóvel.

Quais são as oportunidades de mercado?

Assente na força e no crescimento dos programas rodoviários Europeus nesta área, bem como nas projeções bastante otimistas realizadas (McKinsey & Company 2016), existe hoje margem para captação de interesses para investigação e melhoria no desenvolvimento de sistemas avançados de assistência ao condutor.

A capacidade de colocar estes sistemas em execução tomando partido de dispositivos que hoje em dia são utilizados de forma massiva pela população, como o *smartphone*, permite criar mais-valia e ser um foco de oportunidade de mercado.

Quem define as oportunidades de mercado?

As oportunidades emergentes no mercado dos ADAS estão fortemente ligadas aos programas e legislação que possam ser aplicados pelo governo de cada país, bem como ações de sensibilização para adoção destes sistemas pela população.

Quem é o cliente?

Este sistema terá como potenciais clientes todos os automobilistas que não disponham de um ADAS com TSR na sua viatura. Este cenário poderá estar associado a diferentes fatores, como:

- Automóvel de gama baixa;
- Ano de Fabrico longínquo;
- Sistema *TSR* não adquirido na compra da viatura;
- Fabricante não possui oferta de sistemas *TSR.*

Serão potenciais clientes se, e só se, possuírem os recursos tecnológicos mínimos para execução desse sistema, entenda-se, equipamentos móveis do tipo *smartphone, tablet,* microcomputador e outros.

2.2.2 Question at hand

O sistema satisfará os requisitos do cliente?

A utilização do sistema proposto, auxiliará o cliente final de forma ativa, dando indicações para uma correta tomada de decisão durante a condução automóvel e prevenindo eventuais falhas que possam provocar um acidente de viação. Idealmente, conseguirá reduzir fatalidades, lesões e custos associados.

Como poderá ser definido valor pelo cliente?

Com base nas principais preocupações associadas à aceitação de sistemas inteligentes de condução por parte dos utilizadores finais (descrédito na eficácia dos mesmos, medo de ataques ao sistema ou custos extras associados) (McKinsey & Company, 2016a), a solução proposta permitirá, por um baixo custo, a implementação em qualquer automóvel de um ADAS, pela utilização de um dispositivo pessoal, adaptável e com possibilidade de configuração e customização da aplicação.

Um design existente pode ser adaptado?

Modelos existentes de sistemas TSR poderão ser adaptados na conceção da solução para o problema.

A especificidade do catálogo de sinais e legislação rodoviária portuguesa, permitirá estender modelos já adotados para reconhecimento de imagem, bem como *guidelines* para o desenvolvimento de ADAS.

Que importância poderá ter o time-to-market?

> "First movers may have a chance to shape the industry" (McKinsey & Company 2016)

O mercado dos sistemas ADAS ainda se encontra em plena definição e crescimento. Quanto mais cedo forem lançadas para o mercado estas soluções, maior será a capacidade de serem referências para outros *stake-holders*, e mais facilmente conseguirão patentear os seus componentes.

2.2.3 Pontos de vista

Ponto de vista pessoal das Partes Interessadas

Tendo em conta que o desenvolvimento deste projeto é realizado no seio do LAMU, e considerando-o como parte interessada, o LAMU pretende uma solução evolutiva que permita ser acrescida de outros módulos de um sistema ADAS (sistemas de reconhecimento de sinalização horizontal, reconhecimento de fadiga, entre outros).

Ponto de vista pessoal das Entidades Reguladoras

A criação de valor na área dos sistemas inteligentes de transportes ITS vai de encontro aos programas aplicados pela Comissão Europeia para a regulação e prevenção sinistralidade rodoviária nos países membro CARS 2020 (Union and Commission, 2014).

Ponto de vista pessoal socioeconómico

A capacidade de intervenção no problema que são as fatalidades e lesões provocadas pela sinistralidade automóvel deverá ser um fator preponderante para a prevenção e redução do número de mortes nas estradas, e para a diminuição de todos efeitos colaterais, nomeadamente os associados a custos Hospitalares e Administrativos das sociedades.

2.2.4 Pressupostos

Condições de Operação

Uma solução inteligente de auxilio à condução, deverá estar preparada e prever as diferentes condições de operação, regras e leis de trânsito do País/Estado em que opera.

Os catálogos de sinalização vertical variam de país para país, e para um determinado sinal poderão ser definidas cores e formatos diferentes. A título de exemplo, na Figura 3, estão representados dois sinais com o mesmo significado: obrigação de contornar/passar obstáculo. O de cor azul, da legislação Portuguesa (República Portuguesa, 1998), e o a vermelho, da legislação Brasileira (DNER, 1999). Apesar da semântica, os mesmos possuem estrutura e cor diferente.

Figura 3 - Sinal de Passagem Obrigatória Português (D3a) e brasileiro (R-24b)

É um pressuposto deste problema, que a realidade de reconhecimento seja primeiramente o catálogo de sinais Portugueses, evoluindo depois para reconhecimento gradual dos demais.

Pretende-se definir um subconjunto de sinais que serão usados para efeito de validação do sistema, avançando posteriormente para a cobertura total.

Avaliação de riscos técnicos

O sucesso do reconhecimento de sinalização através de captura de imagem, está diretamente ligado a diferentes fatores, relacionados entre si, tais como:

- Qualidade da imagem obtida. Para um determinado sinal poderão ser capturadas inúmeras imagens com características diferentes entre si, como a cor, o brilho, contraste, entre outros, que condicionam o valor da imagem obtida (Figura 4);
- Sobreposição de elementos externos à sinalização, como árvores e outros obstáculos;
- Desgaste do material e alterações da forma, provocados pela exposição ao ambiente ou mesmo por atos de vandalismo;
- Perpendicularidade da captura, que pode afetar o reconhecimento da sinalização.

Figura 4 – Diferentes qualidades captura de imagem, presente em (Gu *et al.*, 2010)

Nível de aceitação dos pressupostos

A aceitação dos pressupostos definidos anteriormente será válida até ser atingido um nível de maturidade significante do sistema no reconhecimento da sinalização.

2.3 Conceitos de Negócio

Nesta secção são descritos alguns conceitos de negócio relacionados com a problemática desta dissertação.

2.3.1 ITS - Intelligent Transportation Systems

ITS ou Sistemas Inteligentes de Transporte, são descritos como os sistemas que fazem uso das tecnologias de informação e comunicação, bem como, como processos de melhoria e inovação na infraestrutura de transportes da sociedade. (Sussman, 2005)

Os ITS permitem atuar sobre problemas de performance, de segurança e de comodidade da população, pela redução do congestionamento de trânsito, prevenção de acidentes rodoviários ou pela disponibilização de informação relevante de navegação. (Ezell, 2010)

2.3.2 ADAS - Advanced Driver Assistance System

Sistemas ADAS são um subconjunto dos sistemas ITS e têm como principal objetivo auxiliar o condutor através de vários mecanismos tecnológicos (Kala and Warwick, 2015). A sua atuação em cenário real divide-se em duas dimensões, a passiva e a ativa.

Dimensão Passiva e Ativa

Funcionalidades passivas são todas aquelas atividades de um sistema ADAS que não atuam diretamente e mecanicamente na condução do automóvel, sendo apenas um auxilio e alerta para possíveis situações perigosas e permitem ao condutor atuar de forma atempada ou não (*worst case cenario)*. Enquadram-se aqui, como alguns dos tipos de funcionalidades passivas, os seguintes:

- *Lane Departure Warning (LDW)* - O sistema alerta o condutor para o desvio não intencionado das linhas de marcação da estrada;
- *Forward Collision Warning (FCW)* - O sistema alerta o condutor da iminente colisão com outro veículo;
- *Traffic Sign Recognition (TSR)* – O sistema alerta o condutor sobre a presença de sinalização de trânsito, possibilitando que o mesmo atue sobre o veículo de acordo com o tipo do sinal (velocidade, proibição, informações textuais entre outros), prevenindo o erro Humano e possíveis acidentes.

Contrastando com os sistemas passivos, existem os sistemas ativos, que têm influência direta sobre a automóvel, sendo por isso mais críticos. Entre eles, destacam-se:

- *Automatic Emergency Braking (AEB)* - Após identificação de uma colisão iminente, realiza a travagem do veículo sem intervenção do condutor;
- *Adaptive Cruise Control (ACC)* - Permite a adaptação controlada da velocidade de circulação do veículo;
- *Lane Keeping Assist (LKA)* - Permite manter o automóvel dentro das guias de marcação da estrada, retomando a sua rota em caso de desvio;
- *Lane Centering* (*LC)* -Permite manter o veículo centrado com as guias de marcação da estrada;
- *Traffic Jam Assist (TJA)* - Este sistema analisa os automóveis em redor do veículo, e permite em cenários de trânsito, controlar a sua velocidade, posição e distância de segurança.

2.3.3 Problemática TSR

Os sistemas TSR fazem parte das principais componentes de um sistema ADAS.

O problema de identificação de sinalização de trânsito a partir da captura de imagem poderá ser dividido em duas partes, a deteção e o reconhecimento. A deteção pretende identificar se

na imagem recolhida está presente uma zona candidata a um sinal de trânsito. O reconhecimento permite determinar se os candidatos são de facto um sinal de trânsito e quais sinais representam.

2.4 Análise de Valor

Este subcapítulo reflete o processo de análise de valor detalhado em diferentes artefactos, como o modelo de Peter Koen New Concept Development Model descrito em (Koen *et al.*, 2001), definição de valor, valor para o cliente, proposta de valor e descrição do modelo de Canvas.

2.4.1 The Fuzzy Front End (FFE)

[…] The innovation process may be divided into three areas: the fuzzy front end (FFE), the new product development (NPD) process, and commercialization […]

(Peter A.Koen, Greg M.Ajamian *et al.*, 2002)

A primeira fase do processo de inovação é denominada de *Fuzzy Front End*. Neste estágio desenvolvem-se e definem-se as oportunidades e ideias que poderão dar outra magnitude ao processo.

2.4.2 New Concept Development Model – NCD

De forma sustentar a primeira fase do processo de inovação foi adotado o modelo teórico *New Concept Development Model* (Koen *et al.*, 2001).

Caracteriza-se em três componentes (Figura 5). Os cinco elementos chave *Opportunity Identitifcation, Opportunity Analysis, Idea Generation & Enrichment, Idea Selection e Concept Definition,* o motor que faz uso dos cincos elementos chave, sob a liderança e cultura da organização e os fatores de influência.

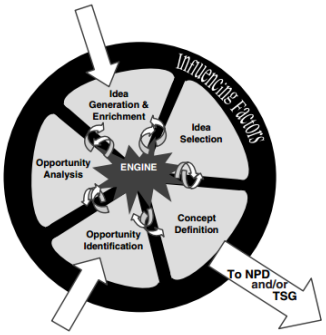


Figura 5 – Estrutura do modelo "*The New Concept Development Model (NCD)*" (Koen *et al.*, 2001)

Pretende-se aqui detalhar cada um destes componentes do modelo no processo de inovação e conceção do *Virtual Driver,* entendendo a natureza do mesmo.

2.4.3 Five NCD Elements

Descrevem-se aqui os cinco elementos chave para nova conceção de um sistema avançado de assistência ao condutor, para reconhecimento de sinalização vertical de baixo custo.

Opportunity Identification

A identificação de oportunidade surgiu através de estudo e investigação, recorrendo a análise de tendência e evolução da tecnologia nos sistemas ADAS ao longo do tempo.

Analisando *roadmaps* da evolução da tecnologia utilizada nos automóveis, verificou-se que existe, nos próximos anos, um exponencial de desenvolvimento, bem como, espaço para lançamento de soluções inteligentes no mercado dos sistemas assistidos e autónomos de condução.
Prevê-se, inclusive, que, no caso das tecnologias sensíveis à resolução de problemas relacionados com *human factors*, existirá maior foco tecnológico a par das tecnologias responsáveis pela comunicação entre o veículo e os *Vulnerable Road Users – VRU*. Na Figura 6 está representado um *roadmap* tecnológico desenvolvido pela plataforma *EPoSS – European Technology Plataform on Smart Systems Integration* onde se evidencia essa prevalência.

A introdução de uma solução tecnológica capaz de intervir no comportamento Humano, no setor automóvel, vai de encontro à tendência do mercado.

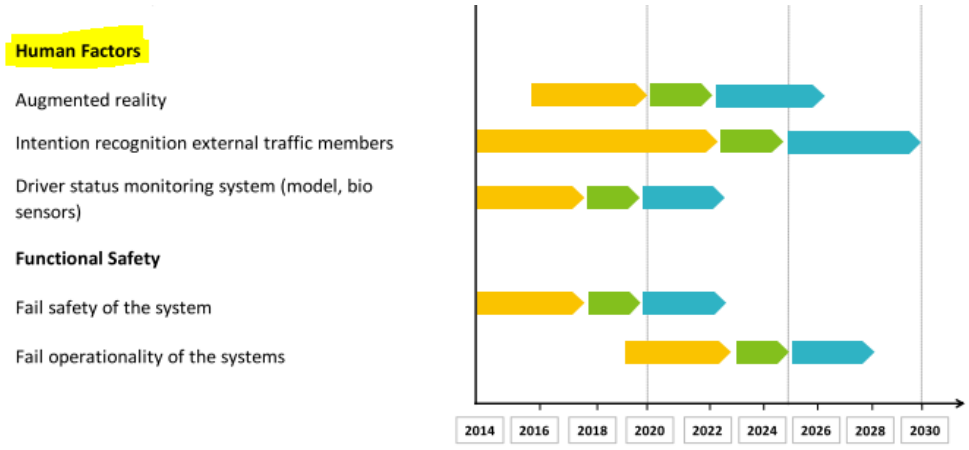


Figura 6 – *Roadmap* de tecnologia da EPoSS para o período 2014-2030 (Dr. Jadranka Dokic, Dr. Beate Müller and Dr. Gereon Meyer, 2015)

Opportunity Analysis

A evolução tecnológica no setor automotivo tem sido um exponencial de investigação e desenvolvimento. No entanto, o lançamento de soluções para o mercado, tem estado ao abrigo dos grandes fabricantes e *OEM's*, cujo principal objetivo é o benefício económico com a venda de sistemas ADAS.

De forma a analisar as tendências do mercado, recorreu-se a análises de mercado realizadas pelas grandes consultoras.

Segundo questionários e previsão de vendas de automóveis para o período 2020 disponibilizada pela *IHS Markit* (IHS Markit, 2016), verificou-se que os sistemas ADAS são os mais desejados pelos consumidores como fator de acréscimo de valor no automóvel.

No entanto, de forma global, os mesmos não pretendem pagar para ter estes sistemas disponíveis (IHS Markit, 2016). Assim, os sistemas de apoio à condução deveriam fazer parte da configuração base do automóvel, e não serem classificados como um extra.

De acordo com análise estatística da *McKinsey* (McKinsey & Company, 2016b), a taxa de adoção dos cidadãos aos sistemas ADAS apresentou-se baixa (dados de 2015). Segundo a estatística disponibilizada, 70% dos compradores de automóveis inquiridos tiveram consciência das funcionalidades dos sistemas ADAS, sendo que 30% os experimentaram e só metade desses adquiriram esses componentes. Desta forma, foi possível concluir, através a amostra selecionada, que o principal fator para o recuo da compra das funcionalidades avançadas de assistência à condução é o preço destes sistemas.

> *"First, falling prices may provide a tailwind: in four of the five countries iawe surveyed, price was the top factor cited in the consideration of ADAS purchases"*
>
> **(McKinsey & Company 2016).**

Como se pode verificar na Figura 7 abaixo, apesar de o rácio de adoção destas tecnologias ser baixo, quem as adquire está altamente satisfeito, verificando-se uma percentagem alta de intenção de voltar a comprar.

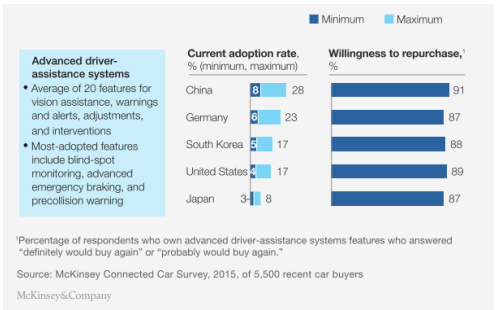


Figura 7 – Taxas de adoção dos sistemas ADAS, analisado através de questionário a 5500 recentes compradores de automóveis (China, Alemanha, Coreia do Sul, Estados Unidos e Japão). Relatório *"Connected Car Survey, 2015"* (McKinsey & Company, 2016b)

A dependência da mobilidade

Analisando o consumo tecnológico da população, verifica-se, hoje, uma utilização massiva à escala mundial dos dispositivos móveis, nomeadamente o *smartphone.*

Estes dispositivos fazem atualmente parte do quotidiano das pessoas. Segundo a *Google*, 87% da população tem o seu *smartphone,* ao seu lado, 24 horas por dia, tendo o mesmo se tornado num objeto pessoal imprescindível (Google, 2017a).

Conclusões

Em suma, após análise de tendências do mercado, verifica-se que, apesar da manifestação de interesse pelo uso dos sistemas inteligentes de condução, os automobilistas, de forma geral, não estão dispostos à aquisição de ADAS, principalmente, pelo custo que lhes está atualmente associado.
Conjugando este fator, com a utilização incessante dos *smartphones*, é criada a oportunidade para o desenvolvimento de uma solução que possa ser competitiva e apelativa ao consumo, colmatando as necessidades previamente identificadas.

Idea Generation & Enrichment

Com base na identificação de oportunidade, foram discutidas ideias com o intuito de responder ao *gap* identificado, atualmente existe no mercado dos sistemas ADAS.

A troca de ideias desenrolou-se através de sessões de Brainstorming entre o LAMU e o Mestrando.

Durante este processo de geração de ideias, assumiram-se alguns tópicos para uma solução inteligente de suporte à condução:

- Captura e processamento de imagem;
- Reconhecimento de sinalização de trânsito vertical;
- Comunicação de informações audiovisuais em tempo real ao condutor;
- Solução executável sobre recursos de baixo custo.

Na abordagem a cada um desses tópicos, foram geradas ideias em diferentes interações (Figura 8).

Primeira Iteração

Solução capaz de utilizar recursos atualmente existentes nos automóveis. Esta ideia por mais embrionária que seria, foi facilmente refutada pela dependência extrema dos equipamentos dos fabricantes e todas as suas restrições.

Segunda Iteração

Solução capaz de ser executada com recursos extra ao automóvel, através da implementação de uma câmara interior ou exterior, capaz de captar imagem frontal e processá-la através de outro dispositivo que permita reconhecer sinalização vertical.

Terceira Iteração

Aplicação nativa ou multiplataforma para dispositivos móveis, que permita a captura, processamento e reconhecimento de sinalização de trânsito vertical, comunicando informação audiovisual pelo display do dispositivo.

Quarta Iteração

Aplicação multiplataforma, concebida através de tecnologias e bibliotecas *open source*, capaz de ser executada em dispositivos móveis, permitindo a captura, processamento e reconhecimento de sinalização de trânsito vertical através de um catálogo de sinais evolutivo.

O processo de reconhecimento deverá utilizar algoritmos de reconhecimento de formas e cores. Considerando a melhoria contínua da aplicação, a mesma poderá recolher dados estatístico de utilização relevantes ao utilizador.

Idea Selection

De acordo com a análise de oportunidades realizada, e verificando-se as tendências atuais do mercado dos ADAS, há necessidade de escolha das ideias que melhor se encaixam na atividade alvo, e que permitirão acrescentar "*business value*" (Koen *et al.*, 2001)

Assumindo como principal fator de competitividade, o desenvolvimento de um sistema avançado de assistência ao condutor baseado em recursos de baixo custo, foi realizada uma seleção de fatores com base no painel da Figura 8:

Implementação de sistema ADAS com a componente de reconhecimento de sinalização de trânsito vertical com as seguintes características:

- Executável em dispositivos móveis comuns, permitindo:
    - Captura > Processamento > Reconhecimento de Sinalização;
    - Algoritmo de reconhecimento de formas e cores;
    - Suporte para catálogo de sinais evolutivo;
    - Recolha de informação estatística;
    - Comunicação e apresentação de informação audiovisual;
    - *Text to Speech.*
- Multi-Plataforma

- *Android, IOS, Windows Phone*

- Utilização de tecnologias e bibliotecas *open-source*

Ideias recusadas

Algumas ideias foram recusadas neste processo, por não serem um benefício e não estarem enquadradas na mais-valia do produto a desenvolver.

A utilização de uma câmara exterior iria acrescer o custo e a praticabilidade do sistema no momento da sua implementação, pois seria necessário, por exemplo, a passagem de cablagem ou a aquisição de uma câmara sem fios e resistente ao meio.

Outra desvantagem seria, ter como requisito a utilização de uma câmara dedicada, em comparação com uma incorporada, e assim acrescer o custo final da solução para o utilizador final. O mesmo se aplica à utilização de recurso dedicados para cada função do sistema.

A utilização da localização atual do automóvel poderá indicar, com base no mapeamento geográfico da sinalização, informação sobre sinalização. No entanto, este tipo de mecanismo está sujeito a diversas falhas: o caso da perda de sinal de GPS ou rede móvel em zonas mais remotas, bem como o desfasamento do mapeamento geográfico com a sinalização temporária.

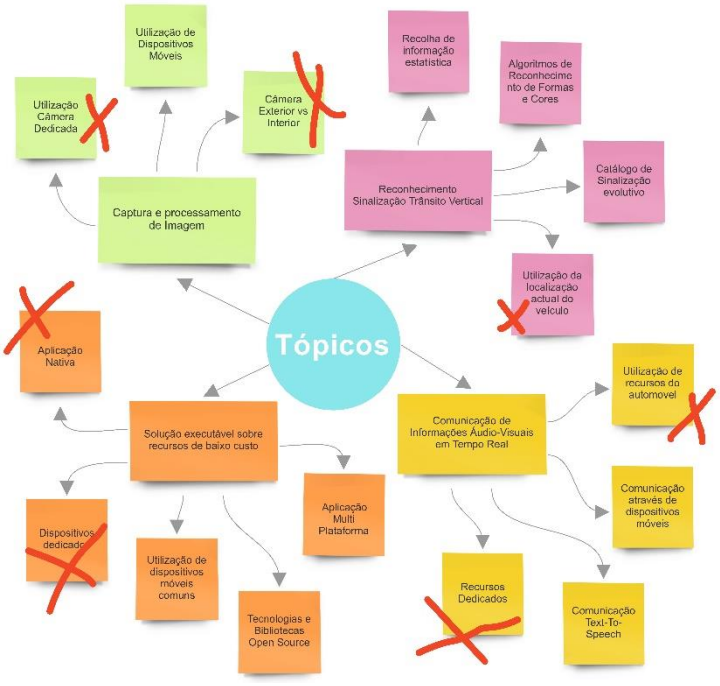


Figura 8 – Painel de geração e seleção de ideias através *Brainstorming*

Concept Definition

O elemento *Concept Definition* define-se como a etapa final do modelo *NCD*. Permite a evolução do processo de inovação para a fase de desenvolvimento de um novo produto através dos

modelos *New Product Development - NPD* ou Technology Stage Gate – TSG (Peter A.Koen, Greg M.Ajamian *et al.*, 2002).

Devido à natureza do projeto e ao grau de certeza técnica do mesmo, optou-se por adotar a seguinte estratégia através das técnicas propostas em *Fuzzy Front End:Effective Methods, Tools,and Techniques* (Peter A.Koen, Greg M.Ajamian *et al.*, 2002) :

- Definição de *Goals and Objectivs* para o desenvolvimento do produto – *Goal Deliberation Approaches.*

Goals and Objectivs

Com base na missão estabelecida para o desenvolvimento deste projeto, foi realizada uma análise de metas e objetivos a seguir de acordo com a missão. Na Tabela 1 está presente uma abordagem de fluxo de informação entre missão, metas e objetivos (Peterson, Jaret and Schenck, 2005)

Goals and Objetivs FlowChart		
Missão do Projecto Desenvolvimento de uma **solução tecnológica na área dos Sistemas Avançados de Assistência à Condução**, que possa ser executada sob recursos comuns de **baixo custo** alcançável a qualquer cidadão e que permita **contribuir para o aumento da segurança rodoviária** em Portugal.		
Goal Solução executável sobre recursos de Baixo Custo	***Goal*** Conceção de sistema ADAS	***Goal*** Atuação na prevenção de sinistralidade rodoviária em Portugal
Objectivs 1) Estudo do estado da arte de soluções *ITS e ADAS* em dispositivos móveis. 2) Investigação ambientes de desenvolvimento para dispositivos móveis *open source*. 3) Estudo de ambientes multi-plataforma 4) Estudo e comparativo de recursos tecnológicos de acordo com o custo e taxas de utilização 1) Estudo de formas de *deploy* e distribuição de baixo custo	***Objectivs*** 1) Conceção arquitetural de sistema *ADAS* para reconhecimento de sinalização vertical *TSR* 2) Implementação de algoritmos de captura e reconhecimento de imagem 3) Implementação de mecanismos de comunicação de informação audiovisual em tempo real **4)** Desenvolvimento de aplicação móvel	***Objectivs*** 1) Contribuição para a diminuição de erros Humanos na condução automóvel 2) Validação e testes com utilizadores Piloto em ambientes reais e virtuais 3) Avaliação de resultados com base em estatísticas e inquéritos

Tabela 1 – Goals and Objetivs FlowChart

2.4.4 Análise de Valor

Nesta secção pretende-se descrever alguns conceitos de valor presentes na literatura, enquadrando-os na temática do problema.

2.4.4.1 Value

"Value is defined as the minimum dollars, which must be expended in purchasing or manufacturing a product to create the appropriate use and esteem factors"

(Lawrence D. Miles, 2015)

O conceito de valor apresenta-se na literatura caracterizado por diversas definições e interpretações. A criação de valor é chave para qualquer negócio e poderá ser definido como a aceitação do cliente para um determinado produto ou serviço, e medido pelo grau de relação estabelecida entre o *supplier* e o mesmo. (Susana Nicola, Eduarda Pinto Ferreira, 2012)

Na perspetiva do custo, poderá ser criado valor com a capacidade de redução de custos na produção de um determinado produto, sem comprometer a qualidade.

"As an example, the cost of producing aluminum has decreased significantly after the necessary electrolytic process was invented in 1886"

(Lindgreen and Wynstra, 2005)

2.4.4.2 Value for the customer

Define-se valor para o cliente, elementos e propriedades que são vividas e percebidas pelo cliente, e que possam explicar a relação do mesmo com um produto ou serviço. A satisfação com um determinado produto é a perceção de valor pelo cliente (Woodall, 2003), bem como, a procura das características de um determinado produto ou serviço para benefício das suas próprias vidas [Frondizi (1971)].

2.4.4.3 Perceived Value

A criação de valor de um produto ou serviço poderá ter diferentes perceções pelos diferentes clientes, com base nas suas necessidades, e características do mesmo. A maximização dos benefícios *versus* a minimização dos sacrifícios poderá estabelecer a preferência do cliente (Lindgreen and Wynstra, 2005).

O valor percebido do ponto de vista do produtor poderá ser diferente do ponto de vista do cliente, pois a sensibilidade e as perceções são diferentes.

2.4.4.4 Value for the Customer (VC) – Benefits and Sacrifices

Seguindo a conceptualização *"Net"* para *Value for the* (Woodall, 2003), decompõem-se aqui (Tabela 2) os benefícios e sacrifícios associados ao valor criado para o cliente com o desenvolvimento da solução *Virtual Driver.*

BENEFITS		***SACRIFICES***
Attributes	***Outcomes***	
Aplicação executável em recursos de baixo custo, *smartphones, tablets ou* microcomputadores	Prevenção de acidentes rodoviários e redução do número de mortes nas estradas (Preocupação social)	Preço de aquisição do dispositivo móvel
Portabilidade do sistema	Redução de despesas materiais com acidentes	Sistema não integrado no automóvel
Independência do ano, marca, modelo e gama do automóvel	Redução de despesas com saúde	Conhecimento de utilização do dispositivo móvel
Independência de restrições tecnológicas do veículo	Consciencialização para uma condução segura	Aprendizagem e formação
Prevenção de erro Humano durante a condução	Segurança	Requisitos mínimos do dispositivo para execução da aplicação
Comunicação de informação audiovisual em tempo real		
Auxilio no cumprimento das regras de sinalização de trânsito		
Solução evolutiva, disponibilização novas componentes ADAS *on-demand*		
Melhoria contínua, através de avaliação estatística e atualização		
Aquisição da aplicação em repositórios comuns, por um baixo custo		
Reconhecimento de sinalização vertical		

Tabela 2 – Benefícios e Sacrifícios associados ao conceito VC (Woodall, 2003), da solução Virtual Driver

2.4.4.5 Longitudinal Perspective on VC

De acordo com a análise das subformas de valor para o cliente, verifica-se que elas têm aspetos de relação temporal e cumulativa. De acordo com a perspetiva longitudinal abordada por

(Woodall, 2003), pretende-se aqui enquadrar os vários benefícios e sacrifícios analisados no ponto anterior.

O valor para o cliente é dividido em quatro "*value temporal positions": "Pre-Purchase, At the point of trade, Post-Purchase e After/use experience".* Na Figura 9, pretende-se representar cada um dos benefícios e sacrifícios nas posições temporais.

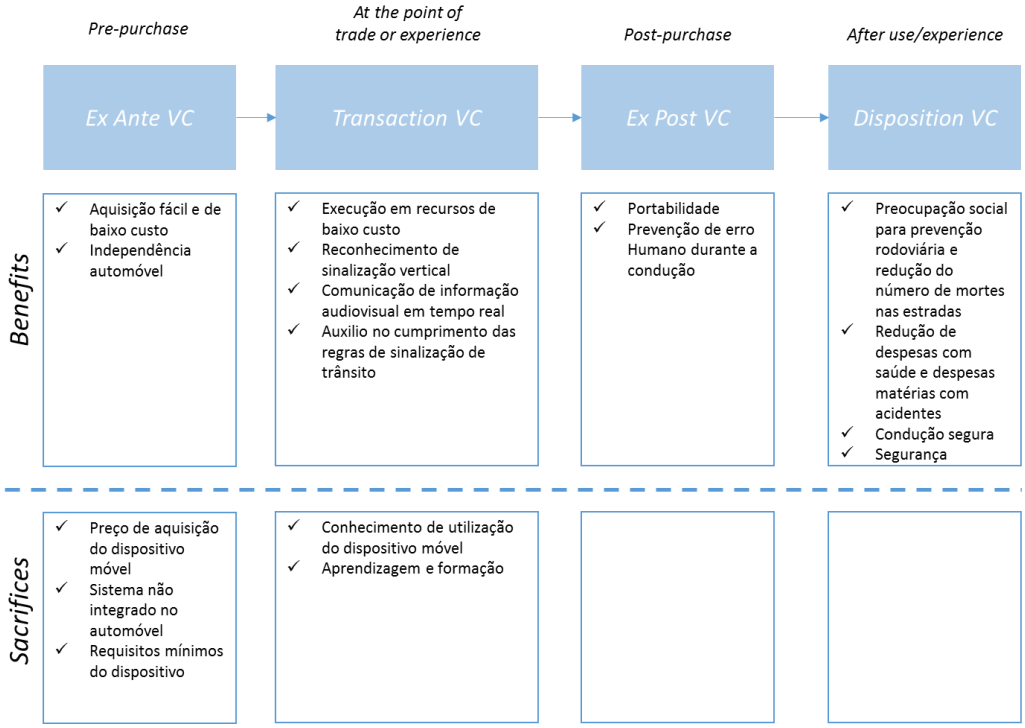


Figura 9 – Perspetiva Longitudinal de *VC*, enquadrada com os benefícios e sacrifícios

Pre-purchase/Ex Ante VC

É nesta posição que se verifica o valor desejado *"Desired Value"* e o valor esperado *"Expected Value"* do cliente para a aquisição do sistema.

De acordo com a análise de mercado realizada anteriormente, verifica-se que o preço de aquisição/execução de um sistema ADAS aparece como um dos principais entraves do processo de compra destes sistemas.

De acordo com o valor esperado, enquadram-se aqui os benefícios relacionados com custo, bem como a independência do automóvel e da tecnologia nele residente se optarem pela adoção do *Virtual Driver*.

Por outro lado, em relação ao valor desejado, verifica-se que os interessados neste tipo de soluções optariam pela sua aquisição se o preço fosse outro, que levaria a disponibilização de uma solução integrada, que contrasta com a solução Virtual Driver, bem como, os requisitos tecnológicos que da sua utilização advêm.

Denota-se aqui como sacrifício, o custo que poderá existir com a aquisição de dispositivo capaz de executar a solução.

At the point of trade or experience/Transaction VC

Dividindo-se em três tipos de valor, *Transaction Value, Acquisition Value* e *Exchange Value*, pretende espelhar o valor para o cliente no ponto de vista da aquisição do produto e da sua experiência.

A vantagem competitiva da solução Virtual Driver reside na capacidade de a mesma se executar através de recursos de baixo custo, estando assim indiretamente ligada a um preço de aquisição baixo. Relacionando esse benefício com os benefícios inerentes as capacidades e utilidades do produto, como reconhecimento de sinalização, redução de erros, comunicação de informação e auxilio no cumprimento das regras rodoviárias, permitem compreender a troca de valor existente.

Os sacrifícios presentes nesta fase estão relacionados com a necessidade de conhecimento de utilização do dispositivo em causa e o Virtual Driver.

Post-purchase/Ex Post VC

Esta posição temporal destaca-se por três tipos de valor, *Received Value, Performance Value* e *Delivered Value*.

Focando-nos no valor entregue para o cliente verificam-se os benefícios relacionados com o valor que foi acrescido, face aquilo que foi gasto pelo mesmo. O fornecimento de uma solução portável, capaz de ser utilizada em diferentes automóveis e em diferentes dispositivos, que possa permitir reduzir o erro Humano durante a condução versus os custos associados à sua execução, determinam o *Delivered Value.*

After use/experience / Disposition VC

Representada pelo tipo de valor Redemption Value ou "value after use/experience" (Woodall, 2003), pretende aqui definir o valor que poderá ser resgatado para futuras acções.

Enquadraram-se com esta posição temporal, os *outcomes* atingidos com a solução, pois é expectável que essas metas sejam atendidas na generalidade após experiência de valor pelo cliente.

2.4.4.6 Value Proposition

"[…] the VALUE PROPOSITION element is an overall view of a firm's bundle of products and services that together represent value for a specific CUSTOMER SEGMENT. It describes the way a firm differentiates itself from its competitors and is the reason why customers buy from a certain firm and not from another […]"

(Osterwalder and Pigneur, 2003)
Com a proposta de valor pretende-se descrever de forma sucinta aquilo que a empresa ou organização pretende oferecer em termos de produtos ou serviços como beneficio para o cliente. A proposta deverá responder a perguntas basilares que elucidam o possível cliente sobre o que se está a oferecer.

Enquadram-se aqui as perguntas e respostas com a proposta de valor do *Virtual Driver*

- Qual é o produto/serviço? - Aplicação móvel do tipo *ADAS* que permite reconhecer sinalização vertical rodoviária e comunicar informações audiovisuais ao condutor, prevenindo acidentes de trânsito.

- Quem são os clientes alvo? - Todos os Condutores de automóveis que possuam dispositivo móvel capaz de executar a solução.

- Que valor é fornecido?

    - Aplicação executável em dispositivos de baixo custo, como *smartphones*, *tablets* ou outros microcomputadores.

    - Permite a independência do ano, marca, modelo, gama e tecnologia do automóvel.

    - Prevenção de acidentes rodoviários

        - Redução de fatalidades e lesões;

        - Redução fatores de adormecimento;

        - Possível redução de custos materiais, saúde e outros.

- Porque o seu produto é único?

    - Porque conseguirá atingir população que, atualmente, por restrições ano, marcar, modelo ou tecnologia não têm acesso a soluções *ADAS*. Recorde-se a idade média do automóvel em Portugal de doze anos. (Instituto Nacional de Estatística, 2016)

2.4.4.7 The Business Model Canvas

Num processo de criação de um produto/serviço inovador é importante conceber um modelo de negócio que sustente racionalmente, como uma organização que cria, entrega e captura valor (Joyce and Paquin, 2016).

Existem vários artefactos que permite espelhar o modelo de negócio, para esta dissertação é detalhado o *Business Model Canvas* proposto por (Osterwalder *et al.*, 2010).

Devido à natureza do projeto *Virtual Driver*, alguns elementos deste artefacto não tem a mesma relevância de um projeto com outra dimensão comercial.

Elenca-se aqui cada um dos "*9 Building Blocks*" que estão representados do modelo de negócio do *Virtual Driver*, que estão concretizados na Figura 10.

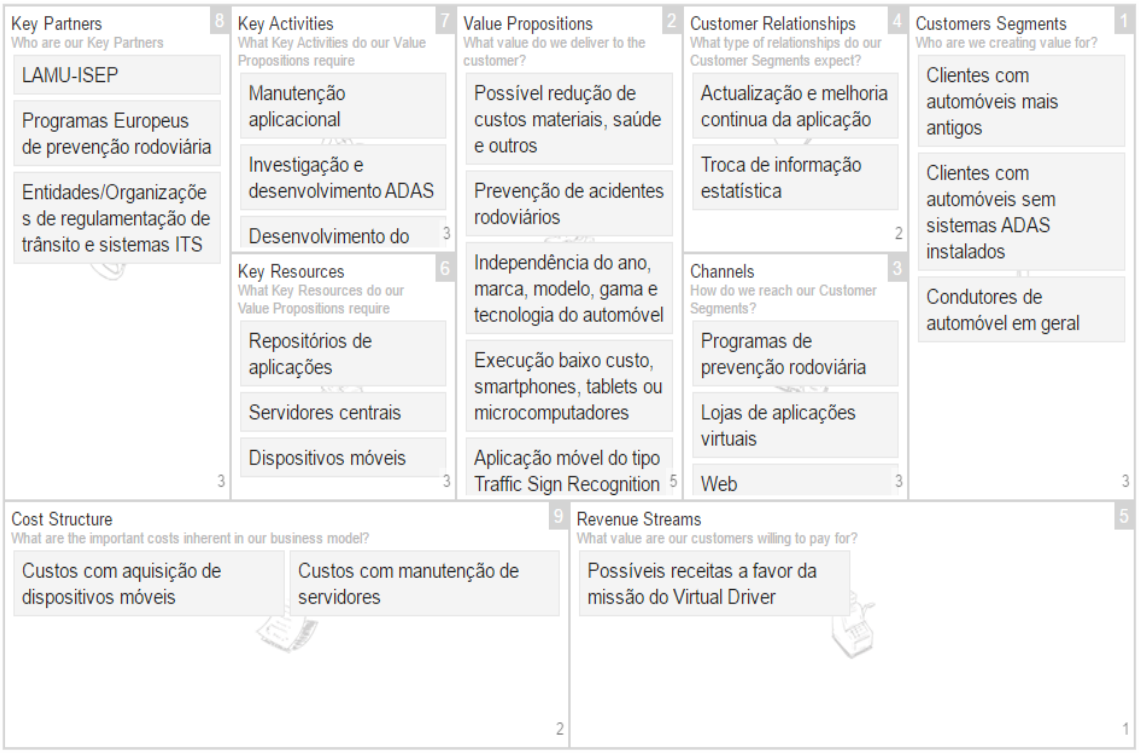


Figura 10 – *The Business Model Canvas* do produto Virtual Driver. Gerado a partir da ferramenta Business Model Fiddle (*Business Model Fiddle*, 2017)

Segmentos de Clientes ("CS-Customer Segments")

Até ao momento identificou-se como segmento de clientes, todos os condutores de automóvel dotados de dispositivos capazes de executarem a aplicação *Virtual Driver*.

Quais as características do segmento?

Caracteriza-se o segmento alvo, como:

- População ativa, habilitada para condução de automóvel;
- Sem sistemas *ADAS* instalados no seu veículo;
- Cliente que não está disposto a adquirir um sistema ADAS por um preço alto do fabricante.

Quais são os potenciais clientes mais importantes?

Identificam-se como potenciais clientes mais importantes, aqueles que por restrições de ano, marca, modelo e tecnologia não conseguem ter acesso a sistemas de condução inteligentes.

Proposta de Valor ("VP-Value Proposition")

Define-se como proposta de valor o fornecimento de um produto sobre forma de aplicação móvel que permite reconhecer sinalização vertical (*Traffic Sign Recognition)* rodoviária e comunicar informações audiovisuais ao condutor, prevenindo acidentes de trânsito.

Que valor é entregue para o cliente?

O valor entregue ao cliente consiste em:

- Aplicação executável em dispositivos de baixo custo, como *smartphones*, *tablets* ou outros microcomputadores;
- Permite a independência do ano, marca, modelo, gama e tecnologia do automóvel;
- Prevenção de acidentes rodoviários.
    - Possível redução de custos materiais, saúde e outros

Quais os problemas dos clientes que se está a tentar resolver?

Os problemas principais que se pretendem resolver são, o preço alto dos sistemas *ADAS* cobrado pela generalidade dos fabricantes automóveis, e as restrições físicas e tecnológicas do automóvel.

Que necessidades do cliente estão a ser satisfeitas?

Com o desenvolvimento do *Virtual Driver*, poderão ser satisfeitas as necessidades relacionadas com a prevenção rodoviária, através de uma solução baixo custo.

Que pacotes de produtos se está a oferecer para cada segmento?

De momento a oferta é transversal aos vários segmentos que possam existir.

Canais ("CH-Channels")

Identificam-se como possíveis canais de distribuição: *Web,* Lojas de aplicações *(Ex. Google Play, Apple Store, Microsoft Store) e* Programas de prevenção rodoviária.

Relacionamento com o cliente ("CR-Customer Relationships")

Prevê-se relacionamento com os utilizadores do *Virtual Driver*, com o possível envio de informação estatística de utilização da aplicação, permitindo despoletar possíveis atualizações. Não estão previstos custos associados a este tipo de transações.

Não foram definidas estratégias de vendas complementares entre outras.

Fontes de Receita ("R$-Revenue Streams")

O foco do desenvolvimento do *Virtual Driver* vai para a missão que reside na capacidade de a solução atuar socialmente na prevenção rodoviária. Possíveis fontes de rendimento que possam surgir poderão ir no sentido da investigação e *upgrade* do sistema.

Do ponto de vista do utilizador, os custos estarão associados fundamentalmente aos recursos necessários para a execução da aplicação

Recursos-Chave ("KR-Key Resources")

Identificam-se recursos chave a utilização de servidores para a recolha de informação estatística e fornecimento de outras informações. Em *client-side*, identificam-se os dispositivos móveis usufruto da aplicação. A nível dos canais de distribuição os recursos chave serão os repositórios de disponibilização da aplicação.

Atividades-Chave ("KA-Key Activities")

Definem-se atividades-chave, o desenvolvimento do produto *Virtual Driver*, investigação ITS, manutenção aplicacional, publicação da aplicação nos canais de distribuição.

Parcerias-Chave ("KP-Key Partnerships")

Verifica-se que poderão ser possíveis parcerias, o LAMU-ISEP, programas Europeus de prevenção rodoviária, entidades e organizações de regulamentação de trânsito e sistemas inteligentes de transporte e empresas de distribuição de soluções.

Estrutura de Custos ("CS-Cost Structure")

Identificam-se aqui os custos relacionados com a manutenção de servidores, custos com a aquisição de dispositivos móveis para desenvolvimento e possíveis integrações de *other-party applications*

2.4.4.8 Value Network

"People naturally network as they work so why not model itself as network"

Verna Allee

A criação e análise de valor está para além do valor associado ao desenvolvimento de um novo produto ou do processo que está subadjacente. A capacidade de entender o tipo de atividade que está relacionada com o produto ou serviço a disponibilizar, o seu ecossistema, bem como definir quais são os intervenientes (pessoas, entidades) e de que forma interagem causando impacto na criação de valor permite definir perceber a respetiva rede de valor (Value Networks, 2011).

A permanente comunicação entre os elementos da rede de valor, permitirá que o fornecedor consiga ou não converter ativos tangíveis ou intangíveis em formas de valor para o cliente.

De acordo com o método de análise de rede valor de (Allee, 2008), distinguem-se dois tipos de fluxo de valor, aquilo que é tangível como produtos ou serviços que se podem diretamente refletir em valor económico e aquilo que é intangível que poderá manter uma relação com o cliente final alimentada através do *feedback* que o mesmo possa dar, convertendo-o ou não em formas de valor.

> *"[…] Intangibles, then, must be understood as intangibles, which includes understanding how they are converted into other negotiable forms of value – and just as importantly, when and why they are not converted […]"*
>
> **(Allee, 2008)**

A proposta de valor do *Virtual Driver* vai para além do benefício direto com a sua disponibilização ao utilizador final, e desta forma podemos representar o fluxo de valor entre o cliente final a solução a fornecer com base em três tipos de troca de valor: *Goods, Services and Revenue, Knowledge* e *Intangible Value* (Verna Allee, 2000) (Figura 11).

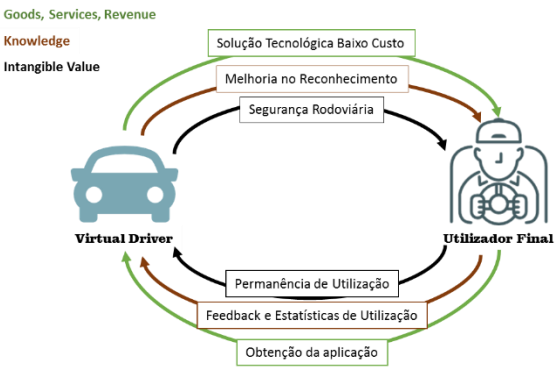


Figura 11 – Mapeamento de fluxo de valor segundo Verna Allee, com base nos três princípios "*The Three Currencies of Value*" (Verna Allee, 2000)

Value Network Mapping, Verna Allee

O modelo de Verna Allee poderá ser utilizado nesta dissertação de forma a espelhar de que forma os *inputs* serão respondidos com contribuições inovadoras e positivas para o mercado dos sistemas *ADAS*. Nesta fase e de um modo hipotético, alguns componentes deste modelo são definidos com base em idealização futura.

Em primeiro lugar deverá ser realizado o mapeamento da rede, entre *Roles, Transactions* e *Deliverables* da atividade de negócio a analisar.

Roles

Roles poderão definidos como as pessoas reais, entidades, organizações, projetos que poderão despoletar interações dentro da rede que permitam atuar na criação de valor. Ilustram-se na seguinte Figura 12, os possíveis *roles* no contexto desta dissertação.

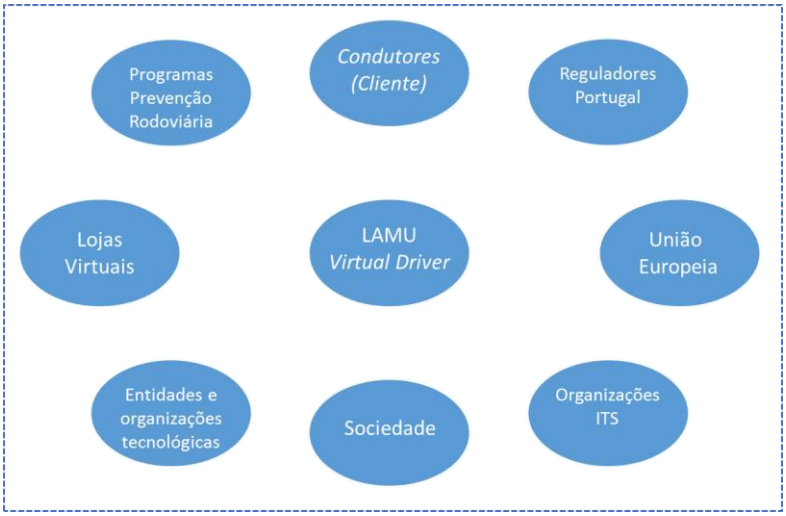


Figura 12 – Identificação dos roles do cenário *Virtual Driver*

Transactions and Deliverables

Transactions procuram determinar o fluxo de informação entre *Roles*. Em termos de representação, assinala-se a movimentação entre dois *Roles* com uma linha direcionada com seta. As linhas poderão ser sólidas e assim representam trocas de informação tangíveis, nomeadamente relacionadas com o produto ou serviço a fornecer ou poderão ser tracejadas e assim representam meios intangíveis, como benefícios indiretos que podem ou não ser convertidos em formas de valor.(Verna Allee, 2008)

Deliverables define aquilo que é passado de forma concreta de *Role* para *Role.*

> *"[…] A deliverable can be physical (e.g. a document or a table) or it can be non-physical (e.g. a message or request that is only delivered verbally). It can also be a specific type of knowledge, expertise, advice, or information about something, or a favor or benefit that is bestowed upon the recipient.[…]"*
>
> **(Verna Allee, 2008)**

O próximo passo é interligar todos *Roles* através *Transactions* e *Deliverables* identificados. Na Figura 13 está representado o mapeamento de *Value Network* do projeto *Virtual Driver.*

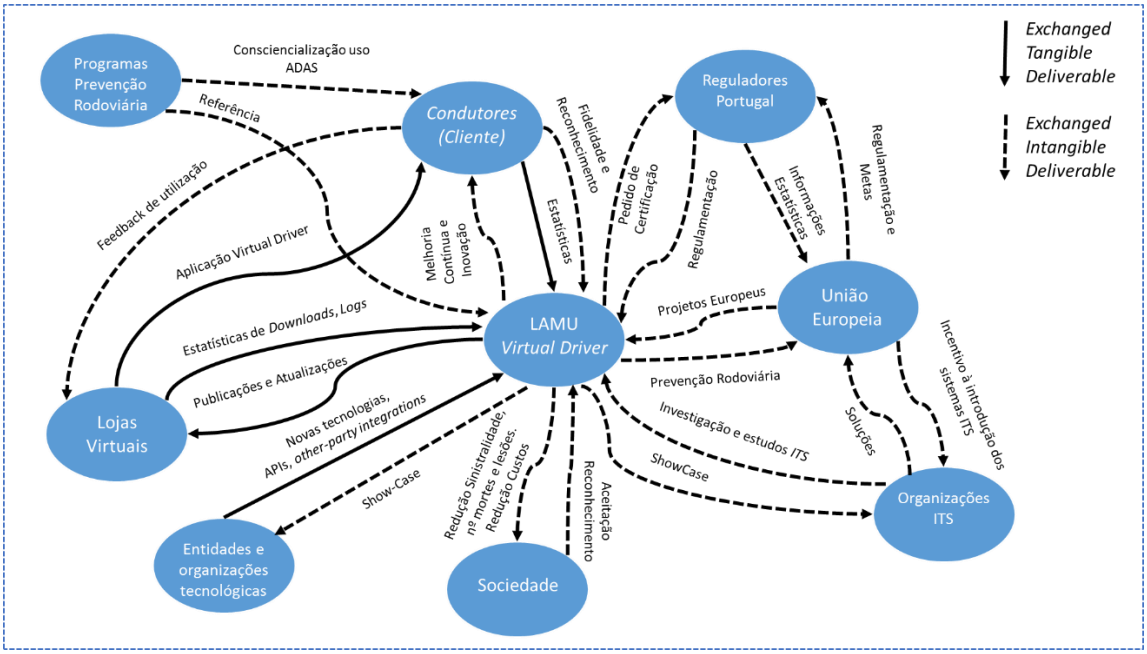


Figura 13 – *Virtual Driver Value Network Mapping*

Após definição dos *Roles* e que tipo de fluxo de entregas ocorre entre cada um deles, é possível de ter uma perceção global de que entidades interagem diretamente e indiretamente com o projeto Virtual Driver.

Posteriormente, deverá ser realizada uma análise à rede de valor, que permitirá perceber qual a magnitude de valor que será convertido a partir de cada uma destas interações. Esta análise baseia-se em três pontos:

- *Exchange Analysis* – Permite perceber até que ponto as interações são coerentes, como também entender se a rede de valor é saudável para a respetiva conversão de valor. (Verna Allee, 2008);

- *Impact Analysis* – Permite analisar qual o impacto dos *inputs* recebidos pelos *Roles* a nível de valor percebido para cada um deles (Verna Allee, 2008);

- *Value Creation Analysis* – Procura de que forma cada um dos *Roles* consegue adicionar valor à rede de que forma poderão ser usados os *outputs* de cada um deles.

2.4.5 Método de Análise Hierárquica

Na conceção de uma solução para um problema, existirá espaço para que sejam tomadas algumas decisões com base em critérios e alternativas com graus de importância diferentes ou pesos variáveis. Nestes casos será necessário a aplicação de um método multicritério para a decisão.

Aplica-se aqui um caso prático do método *Analytic Hierarchy Process* (AHP) (Saaty and Vargas, 2012). Nas seguintes secções são detalhadas as diferentes fases do método, que foram lecionadas no módulo de análise de valor do Mestrado de Engenharia Informática do Instituto Superior de Engenharia do Porto.

Fase 1 – Construção Árvore de Decisão

Em primeiro lugar definimos a hierarquia de decisão com base no problema identificado.

1. **Problema:** Escolha que *Smartphone* comprar para execução do Virtual Driver.
2. **Definição de Critérios:** Preço, Câmara, Resolução, Processador e GPU
3. **Alternativas:** Smartphones: OnePlus 3, Xiaomi Mi 5s, LeEco Le Max 2

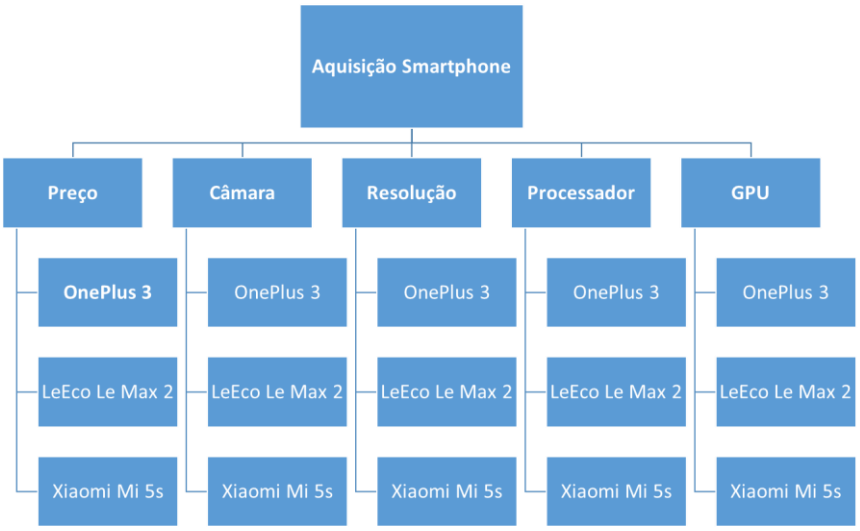


Figura 14 – Árvore de decisão para *Analytic Hierarchy Process*

Para uma melhor compreensão do problema enunciam-se as características técnicas de cada smartphone na Tabela 3.

	OnePlus 3	LeEco Le Max 2	Xiaomi Mi 5s
Preço	400 €	350 €	310 €
Câmara	16 MP	21MP	12MP
Resolução	401ppi	515ppi	428ppi
Processador	2,2 GHz	2,2 GHz	2,4 GHz
GPU	624 MHz	624 MHz	624 MHz

Tabela 3 – Características técnicas de cada critério definido. [6]

[6] Informação retirada do site http://www.gsmarena.com/

Nesta fase atribuem-se prioridades entre os elementos de cada nível da hierarquia (critérios e alternativas). As prioridades são estruturadas numa matriz de comparação.

Os valores possíveis para atribuição de importância, são estipulados pela Escala Fundamental definida por Saaty

Intensity of importance on an absolute scale	Definition	Explanation
1	Equal importance	Two activities contribute equally to the objective
3	Moderate importance of one over another	Experience and judgment strongly favor one activity over another
5	Essential or strong importance	Experience and judgement strongly favor one activity over another
7	Very strong importance	An activity is strongly favored and its dominance demonstrated in practice
9	Extreme importance	The evidence favoring one activity over another is of the highest possible order of affirmation
2, 4, 6, 8	Intermediate values between the two adjacent judgments	When compromise is needed
Reciprocals	If activity i has one of the above numbers assigned to it when compared with activity j, then j has the reciprocal value when compared with i	
Rationals	Ratios arising from the scale	If consistency were to be forced by obtaining n numerical values to span the matrix

Figura 15 – Escala Fundamental definida por Thomas Saaty (Thomas L. Saaty, 1990)

Definindo a matriz de comparação para os critérios definidos:

	Preço	Câmara	Resolução	Processador	GPU
Preço	**1**	1/4	1/2	1/3	1/3
Câmara	4	**1**	3	2	2
Resolução	2	1/3	**1**	1/3	1/5
Processador	3	1/2	3	**1**	1/4
GPU	3	1/2	5	4	**1**

$$= \begin{vmatrix} 1 & \frac{1}{4} & \frac{1}{2} & \frac{1}{3} & \frac{1}{3} \\ 4 & 1 & 3 & 2 & 2 \\ 2 & \frac{1}{3} & 1 & \frac{1}{3} & \frac{1}{5} \\ 3 & \frac{1}{2} & 3 & 1 & \frac{1}{4} \\ 3 & \frac{1}{2} & 5 & 4 & 1 \end{vmatrix}$$

Tabela 4 - Matriz de comparação dos critérios de segundo nível

Fase 2 – Prioridade Relativa

Pretende-se determinar a prioridade relativa de cada critério. Dessa forma é necessário realizar a normalização da matriz de comparações. Cada valor da matriz é dividido pelo total da sua coluna.

	Preço	Câmara	Resolução	Processador	GPU
Preço	1	1/4	1/2	1/3	1/3
Câmara	4	1	3	2	2
Resolução	2	1/3	1	1/3	1/5
Processador	3	1/2	3	1	1/4
GPU	3	1/2	5	4	1
Soma	13,00	2,58	12,50	7,67	3,78

Tabela 5 - Matriz de comparação dos critérios de segundo nível, com somatório por coluna

	Preço	Câmara	Resolução	Processador	GPU
Preço	0,08	0,10	0,04	0,04	0,09
Câmara	0,31	0,39	0,24	0,26	0,53
Resolução	0,15	0,13	0,08	0,04	0,05
Processador	0,23	0,19	0,24	0,13	0,07
GPU	0,23	0,19	0,40	0,52	0,26

Tabela 6 - Matriz normalizada dos critérios de Segundo nível

Fase 3 – Prioridade relativa de cada critério

Pretende-se nesta fase identificar a ordem de importância de cada critério. É realizada a média aritmética dos valores de cada linha da matriz normalizada e dessa forma se obtém o vetor de prioridades ou vetor próprio.

	Preço	Câmara	Resolução	Processador	GPU	Prioridade Relativa
Preço	0,08	0,10	0,04	0,04	0,09	0,07
Câmara	0,31	0,39	0,24	0,26	0,53	0,34
Resolução	0,15	0,13	0,08	0,04	0,05	0,09
Processador	0,23	0,19	0,24	0,13	0,07	0,17
GPU	0,23	0,19	0,40	0,52	0,26	0,32

Tabela 7 - Matriz normalizada cálculo da prioridade relativa

Pode-se concluir que a partir dos resultados obtidos o critério Câmara tem maior prioridade com peso **0.34**, seguido do GPU, Processador, Resolução e Preço.

Representação Matricial:

$$\begin{vmatrix} 1 & \frac{1}{4} & \frac{1}{2} & \frac{1}{3} & \frac{1}{3} \\ 4 & 1 & 3 & 2 & 2 \\ 2 & \frac{1}{3} & 1 & \frac{1}{3} & \frac{1}{5} \\ 3 & \frac{1}{2} & 3 & 1 & \frac{1}{4} \\ 3 & \frac{1}{2} & 5 & 4 & 1 \end{vmatrix} \rightarrow normalizar \rightarrow \begin{vmatrix} 0.08 & 0.10 & 0.04 & 0.04 & 0.09 \\ 0.31 & 0.39 & 0.24 & 0.26 & 0.53 \\ 0.15 & 0.13 & 0.08 & 0.04 & 0.05 \\ 0.23 & 0.19 & 0.24 & 0.13 & 0.07 \\ 0.23 & 0.19 & 0.40 & 0.52 & 0.26 \end{vmatrix} \rightarrow vetor\ prior. \rightarrow \begin{vmatrix} 0.07 \\ 0.34 \\ 0.09 \\ 0.17 \\ 0.32 \end{vmatrix}$$

Fase 4 – Avaliar a consistência das prioridades relativas

De seguida deverá ser calculada a Razão de Consistência (RC) para avaliar a consistência dos julgamentos em relação a grandes amostras de juízos completamente aleatórios.

1. RC > 0.1 , então os julgamentos não são confiáveis, porque o seu valor está próximo do conforto de aleatoriedade. Assim os resultados obtidos não apresentam valores consistentes.

2. Em primeiro lugar, é necessário obter o maior valor próprio λ_{max} da matriz original A, que poderá ser obtido através da equação $Ax = \lambda_{max}x$

    a. Substituindo cada um dos elementos A (matriz original) e x (vetor próprio) na equação e obter λ_{max}

$$\begin{vmatrix} 1 & \frac{1}{4} & \frac{1}{2} & \frac{1}{3} & \frac{1}{3} \\ 4 & 1 & 3 & 2 & 2 \\ 2 & \frac{1}{3} & 1 & \frac{1}{3} & \frac{1}{5} \\ 3 & \frac{1}{2} & 3 & 1 & \frac{1}{4} \\ 3 & \frac{1}{2} & 5 & 4 & 1 \end{vmatrix} * \begin{vmatrix} 0.07 \\ 0.34 \\ 0.09 \\ 0.17 \\ 0.32 \end{vmatrix} \cong \lambda_{max} * \begin{vmatrix} 0.07 \\ 0.34 \\ 0.09 \\ 0.17 \\ 0.32 \end{vmatrix} \leftrightarrow$$

$$\leftrightarrow \begin{vmatrix} 0.36 \\ 2.04 \\ 0.46 \\ 0.84 \\ 1.83 \end{vmatrix} = \lambda_{max} * \begin{vmatrix} 0.07 \\ 0.34 \\ 0.09 \\ 0.17 \\ 0.32 \end{vmatrix}$$

$\leftrightarrow \lambda_{max} \cong$ average{0.3597/0.0688, 2.1517/0.3858, 0.3844/0.0757, 0.9404/0.1726, 1.7654/0.2972}$\cong 5{,}4553$

3. Depois de calculado o λ_{max} (valor próprio), pretende-se calcular agora o Índice de Consistência (IC).

   a. A fórmula é dada por $IC = \frac{\lambda_{max} - n}{\mathrm{n}-1}$, entende-se n por o número de critérios

$$IC \cong \frac{5.34 - 5}{5 - 1} \cong 0{,}08$$

4. Calculando a Razão de Consistência através de:

   a. A fórmula é dada por:

$$RC = \frac{IC}{\mathrm{IR}}$$

[…] "Sendo que **IR** é um índice aleatório calculado para matrizes quadradas na ordem n pelo Laboratório Nacional de Oak Ridge, EUA." […]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59

Tabela 8 – Valores IR para matrizes quadradas de ordem n

Fonte: Módulo de Análise de Valor, MEI, ISEP 2016

Desta forma utilizando a ordem 5, RC será calculada por RC = 0.08 / 1.12 ≅ **0,07**

Conclui-se que com RC ≅ **0,07 < 0.1,** os valores das prioridades relativas estão consistentes.

Fase 5 – Construção da matriz de comparação paritária para cada critério, considerando cada uma das alternativas

Da mesma forma que foi feito anteriormente para cada um dos critérios. Agora o processo é realizado para cada um dos critérios face às alternativas.

Preço	OnePlus 3	LeEco Le Max 2	Xiaomi Mi 5s	Prioridade Relativa
OnePlus3	0,11	0,10	0,12	**0,11**
LeEco Le Max 2	0,33	0,30	0,29	**0,31**
Xiaomi Mi 5s	0,56	0,60	0,59	**0,58**

Tabela 9 – Matriz de comparação e prioridade relativa para o critério Preço

Câmara	OnePlus 3	LeEco Le Max 2	Xiaomi Mi 5s	Prioridade Relativa
OnePlus3	0,19	0,18	0,27	**0,21**
LeEco Le Max 2	0,75	0,72	0,64	**0,70**
Xiaomi Mi 5s	0,06	0,10	0,09	**0,09**

Tabela 10 – Matriz de comparação e prioridade relativa para o critério Câmara

Resolução	OnePlus 3	LeEco Le Max 2	Xiaomi Mi 5s	Prioridade Relativa
OnePlus3	0,13	0,14	0,09	**0,12**
LeEco Le Max 2	0,63	0,69	0,73	**0,68**
Xiaomi Mi 5s	0,25	0,17	0,18	**0,20**

Tabela 11 – Matriz de comparação e prioridade relativa para o critério Resolução

Processador	OnePlus 3	LeEco Le Max 2	Xiaomi Mi 5s	Prioridade Relativa
OnePlus3	0,17	0,17	0,17	**0,17**
LeEco Le Max 2	0,17	0,17	0,17	**0,17**
Xiaomi Mi 5s	0,67	0,67	0,67	**0,67**

Tabela 12 – Matriz de comparação e prioridade relativa para o critério GPU

GPU	OnePlus 3	LeEco Le Max 2	Xiaomi Mi 5s	Prioridade Relativa
OnePlus3	0,33	0,33	0,33	**0,33**
LeEco Le Max 2	0,33	0,33	0,33	**0,33**
Xiaomi Mi 5s	0,33	0,33	0,33	**0,33**

No critério GPU, verifica-se que o componente para as diferentes alternativas é igual, gerando a mesma prioridade relativa

2.4.5.1 Fase 6 – Obter a prioridade composta para as alternativas

Nesta última etapa, são obtidas as prioridades compostas das alternativas. Para isso, é necessário multiplicar os valores das prioridades relativas obtidas para cada critério/alternativa com as prioridades relativas obtidas anteriormente para cada critério.

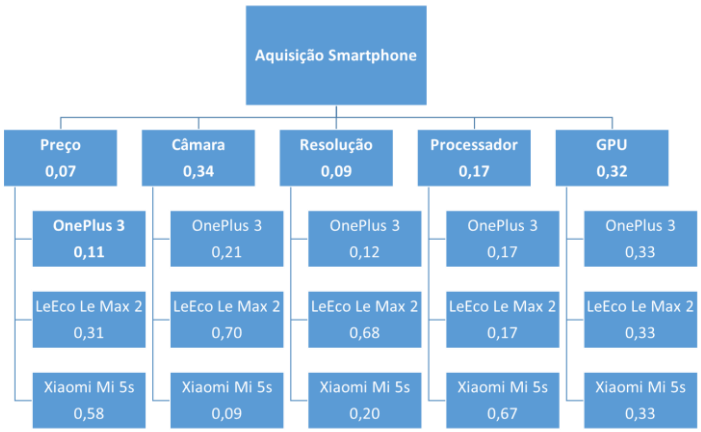


Figura 16 – Distribuição Hierárquica das prioridades relativas

Calculando a multiplicação da matriz das prioridades relativas de cada alternativa pelo vetor de prioridades relativas dos critérios, ou seja Matriz Prioridade * Pesos dos Critérios:

$$\begin{vmatrix} 0.11 & 0.21 & 0.12 & 0.17 & 0.33 \\ 0.31 & 0.70 & 0.68 & 0.17 & 0.33 \\ 0.58 & 0.09 & 0.20 & 0.67 & 0.33 \end{vmatrix} * \begin{vmatrix} 0.07 \\ 0.34 \\ 0.09 \\ 0.17 \\ 0.32 \end{vmatrix} \cong \begin{vmatrix} 0.22 \\ 0.46 \\ 0.31 \end{vmatrix}$$

2.4.5.2 Fase 7 – Obter a prioridade composta para as alternativas

$$\begin{vmatrix} 0.11 & 0.21 & 0.12 & 0.17 & 0.33 \\ 0.31 & 0.70 & 0.68 & 0.17 & 0.33 \\ 0.58 & 0.09 & 0.20 & 0.67 & 0.33 \end{vmatrix} * \begin{vmatrix} 0.07 \\ 0.34 \\ 0.09 \\ 0.17 \\ 0.32 \end{vmatrix} \cong \begin{vmatrix} 0.22 \\ \mathbf{0.46} \\ 0.31 \end{vmatrix}$$

O *smartphone* "LeEco Le Max 2" surge como o mais indicado para execução da aplicação *Virtual Driver*, com base nos critérios e importâncias definidas.

2.5 Estado da Arte

Este subcapítulo pretende demonstrar o estudo e análise de algumas soluções ITS concebidas e disponíveis no mercado, nomeadamente os sistemas ADAS.

É detalhado o estudo da norma de sinalização de trânsito Portuguesa[7].

[7] Norma presente no site: http://www.infraestruturasdeportugal.pt/rede/rodoviaria/seguranca-rodoviaria/normas-de-sinalizacao

Pretende-se também detalhar abordagens e algoritmos atualmente usados no reconhecimento de sinalização vertical de trânsito, bem como realizar uma revisão tecnológica que dará suporte ao desenvolvimento da solução.

2.5.1 Sinalização Rodoviária Vertical em Portugal

De acordo com a problemática e pressupostos assumidos, neste capitulo, será abordada a estrutura *standard* da sinalização de trânsito vertical, em vigor atualmente em Portugal.

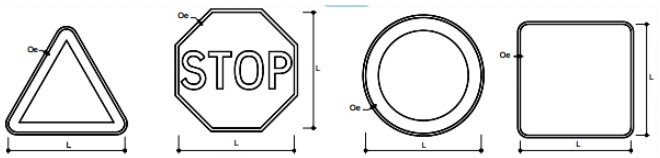
Em Portugal a sinalização de trânsito vertical é definida pelo Regulamento de Sinalização do Trânsito (RST), Decreto Regulamentar nº 22-A/98 de 1 de outubro de 1998.

Existem vários tipos de sinais: sinais de perigo, regulamentação, indicação, mensagem variável e turístico cultural.

A coerência e regulamentação do formato dos sinais é definida pela Norma de Sinalização Vertical (NSV) disponibilizada pelas Estradas de Portugal ou, agora, Infraestruturas de Portugal. A norma define regras de formatos e tamanhos permitidos para todo o sistema de sinalização vertical, exceto sinais de mensagem variável, bem como a definição dos caracteres alfanuméricos e outros símbolos presentes nos sinais.

A nível de sinalização de código de estrada, existem dimensões pré-definidas (Figura 17) segundo:

"*[…] quadro dimensões nominais: reduzida (60 cm), normal (70 cm ou 90 cm)* e *grande (115 cm)* ["…"] (Instituto da Mobilidade e dos Transportes (IMT, no date)



DIMENSÕES (cm)	REDUZIDO	NORMAL		GRANDE
L	60.0	70.0	90.0	115.0
Oe	1.0	2.0	2.0	5.0

Figura 17 – Dimensões para os formatos comuns de sinais de código (Instituto da Mobilidade e dos Transportes (IMT, no date)

Segundo a norma NSV, os sinais ditos "de código" foram desenhados de acordo com a dimensão de 70 cm e um quadrado de 10 cm de lado, para a simbologia de informação.

Repara-se que existem sinais de código com particularidades diferentes para efeitos de identificação, nomeadamente os sinais de regulamentação de prescrição específica de zona. A dimensão total do sinal vai para além dos 70 cm, isto é o sinal original é englobado numa caixa de zona com altura de 1,4 metros e 1,15 de largura (Figura 18).

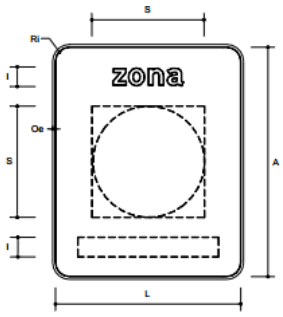


Figura 18 – Sinal de regulamentação de prescrição especifica de zona (Instituto da Mobilidade e dos Transportes (IMT, no date)

Este é um cenário que poderá influenciar o método de identificação do sinal.

No desenvolvimento desta dissertação, para efeitos de protótipo, procura-se evidenciar a identificação em primeiro lugar dos sinais de código de formato normal, sendo evoluído para os restantes sinais.

Caracteres Alfanuméricos

Em relação aos caracteres presentes nos sinais, estão determinadas regras quanto à sua apresentação. Para sinais com fundo de cor azul, verde, laranja, castanha, cinzenta ou vermelha, devem ser usados os abecedários e numerários do tipo negativo. Caso o fundo seja de cor branca ou amarela são utilizados os caracteres positivos (Tabela 13). Os tipos de letra são fornecidos pelas Infraestruturas de Portugal.

Negativo	Positivo
A12 a	A12 a

Tabela 13 – Abecedários e numerários negativos e positivos (Instituto da Mobilidade e dos Transportes (IMT, no date)

Analisando as características da diferente gama de sinais disponíveis, verificamos a predominância da cor vermelha e forma triangular nos sinais de Perigo (Figura 19).

Figura 19 – Subconjunto de Sinais de perigo em Portugal[8]

Nos sinais de regulamentação existem diferentes categorias, predominando as formas quadradas, circulares, octogonais e também triangulares. Existem ainda diversas subcategorias que derivam deste tipo: sinais de cedência de passagem, proibição, prescrição específica, informação, pré-sinalização, direção, confirmação, identificação de localidades, entre outros complementares.

No *web site* das infraestruturas de Portugal (Portugal, no date) é possível obter a imagem de cada sinal, em formato *AutoCad* ou em formato GIF.

2.5.2 Sistemas ADAS e TSR no mercado

Nesta secção, serão destacados alguns sistemas *ADAS* existentes, tanto no mercado automóvel, como em lojas de aplicações. Será dado foco à componente de *Traffic Sign Recognition*

2.5.2.1 Roadly

É uma aplicação inteligente patenteada pela RoadAR, inc [9], disponível nas plataformas *Google Play e Apple Store*, que permite auxiliar os condutores, alertando para a presença de sinalização e regras trânsito, deteção de passadeiras para peões e permite apresentar um sistema de navegação baseado em GPS.

Roadly alimenta uma base de conhecimento que corre sobre o dispositivo móvel, colecionado informação sobre a estrada e sinalização recolhida através da utilização de mecanismos *Computer Vision* e *Deep Neural Networks*. Destaca-se também a introdução realidade aumentada no processo de navegação.

[8] Imagem presente em: http://www.infraestruturasdeportugal.pt/rede/rodoviaria/seguranca-rodoviaria/normas-de-sinalizacao

[9] Informação presente em http://www.road.ly/

Figura 20 – Interface com o utilizador e principais funcionalidades do Roadly.[10]

Não foi possível obter informação detalhada sobre o processo de reconhecimento de sinalização de trânsito utilizado pelo Roadly. No entanto foram realizados testes à aplicação, mas não foi possível verificar o funcionamento do processo de reconhecimento, por existirem talvez restrições geográficas. Destaca-se a boa sensação na interação com o sistema.

2.5.2.2 Mobileye

Mobileye é uma empresa Israelita de tecnologia, fundada em 1999. Tem como principal missão a utilização da visão Humana para o desenvolvimento de soluções tecnológicas, que poderão tornar as estradas mais seguras, reduzir o congestionamento de trânsito, e salvar vidas.

A empresa é hoje líder no fornecimento de *software* para sistemas avançados de apoio à condução de automóveis autónomos, com mais de 25 parcerias com fabricantes automóvel. Através de uma parceria com a BMW e Intel, planeiam produzir veículos autónomos para 2021. Possuem também parceria com cerca de 27 parceiros de equipamentos OEM (Mobileye, 2017a).

Seguindo a sua filosofia, a *Mobileye* baseia-se no princípio da visão Humana, utilizando nas suas soluções ADAS uma câmara *single-lensed* ou *mono-camera,* que permite capturar imagem e reconhecer formas, veículos, peões, linhas de estrada e texto em sinais de trânsito.

> *"The Mobileye mono-camera was inspired by human vision, which only uses both eyes to obtain depth perception for very short distances"*
>
> ***Amnon Shashua, co-founder, chairman and CTO of Mobileye***

[10] Imagem retirada do site do oficial: http://www.road.ly/

No entanto, a empresa reconhece que a adição de uma segunda câmara represente um benefício para reconhecimento em curtas distâncias.

Funcionalidades ADAS

A Mobileye procura dar resposta a dois tipos de funcionalidades de um sistema avançado de assistência ao condutor: a componente passiva e ativa. O mesmo suporta as funcionalidades *AED, LDW, FCW, LKA, LC, TJA, TSR e Intelligent High-beam control (IHC)* utilizando a captura realizada por uma câmara única montada no para-brisas e processada pelo *chip EyeQ®*.

EyeQ®

A solução Mobileye baseia-se num sistema integrado do tipo *system-on-chip (SoC)*, fornecendo um conjunto de funcionalidades *ADAS* baseadas numa câmara única. Nas últimas versões o *chip* garante largura de banda e capacidade para processamento e *stream* de imagem para múltiplas câmaras. (Mobileye, 2017b)

Atualmente está a ser desenvolvido a 5ª geração deste *chip*, que fornecerá funções para a autonomia total do veículo, em 2020.

O *chip EyeQ®* possui um conjunto de *cores* completamente programáveis denominados de *"accelerators",* desenhados especialmente para dar respostas às funcionalidades *ADAS* e condução autónoma. Cada *accelerator* poderá ser otimizado para atender a um conjunto específico de algoritmos e metodologias, como *signal-processing, machine-learning* ou *deep neural networks.* Este tipo de flexibilidade permite maximizar o poder de computação para tarefas específicas, reduzindo o tempo de processamento. (Mobileye, no date). Os núcleos programáveis são:

- *Vector Microcode Processors (VMP)* – É um processador do tipo *VLIW SIMD*, que permite um acesso menos custoso e flexível à memória e possui suporte de *hardware* para operações relacionadas com aplicações *computer-vision* em cenário multiprocessador;

- *Multithreaded Processing Cluster (MPC)* – É um processador mais versátil que qualquer *GPU* e mais eficiente que um *CPU;*

- *Programmable Macro Array (PMA)* – Aproximação da densidade de computação das funções dos aceleradores de *hardware* sem comprometer a sua programação.

O desenvolvimento do *EyeQ®5* 2020 permitirá elevar os níveis de performance a valores bastante significativos. O novo *chip* possibilitará o processamento da informação disponibilizada por mais de 16 câmaras *muti-mega-pixel* e outros sensores. Terá capacidade para processar 15^{18} operações por segundo, com um consumo estimado entre os 5 e 6 *Watts*.

Destaca-se o diagrama de blocos (Figura 21) deste *chip*, a largura de banda dedicada de 40 *Gbps* para dispositivos de *Input/Output*, os 18 *Gbps* para entradas *interfaces* PCIe, as portas *Gigabit Ethernet*, bem como, a inclusão de 18 processadores *computer-vision*.

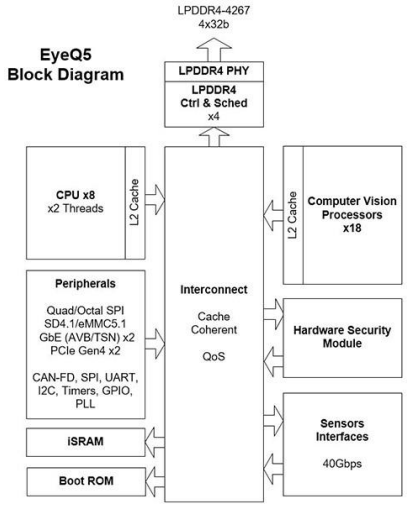


Figura 21 – *Block Diagram Mobileye EyeQ®5*

2.5.2.3 Mercedes

O fabricante de automóveis Mercedes fornece aos seus clientes o sistema *Traffic Sign Assist*[11]. Através de uma câmara interior na frente do automóvel, permite capturar e realizar o reconhecimento de sinalização com formato circular. O reconhecimento realizado é cruzado com a informação persistida no seu sistema de navegação "COMAND Online Display", permitindo que em casos em que o reconhecimento de sinais de velocidade não seja possível, o sistema consiga alertar para zonas de velocidade controlada através apenas de navegação. O sistema comunica informações audiovisuais através do respetivo quadrante e painel de navegação.

2.5.2.4 Ford

O *TSR* disponibilizado pelo fabricante *Ford* captura sinalização de trânsito através de uma câmara montada no espelho retrovisor, e apresenta a identificação através do computador de bordo no quadrante (Figura 22).

O sistema está limitado à disponibilização simultânea de dois sinais e, considerando informação pública disponível, aparenta reconhecer apenas sinalização de velocidade.

[11] Informação presente em: http://techcenter.mercedes-benz.com/en/traffic_sign_assist/detail.html

Figura 22 - *TSR* Ford[12]

2.5.2.5 Mazda

Em 2017, a Mazda lançará nos seus automóveis a componente TSR através do seu módulo inteligente de segurança o i-ACTIVESENSE.

O sistema alerta o utilizador através de notificações visuais no painel central e através de áudio. Pela informação obtida deste sistema (Mazda, no date), o mesmo será dotado de identificar outros sinais para além dos de limite de velocidade (Figura 23).

Figura 23 – *TSR Mazda*

2.5.3 Identificação de Sinalização de Trânsito

A revisão bibliográfica aqui detalhada pretende distinguir as diferentes fases de um sistema *Traffic Sign Recognition*, assinalar as dificuldades de identificação e evidenciar algumas abordagens utilizadas por outros autores.

O processo de identificação de sinais de trânsito é composto por duas fases fundamentais, a deteção do objeto e o reconhecimento do mesmo como um sinal de trânsito (Fleyeh and Dougherty, 2005).

Na fase de deteção, a imagem capturada é submetida a pré-processamento inicial, com a aplicação de filtros, a conversão de espaço de cores, entre outros. Pretende-se que da fase de deteção resulte a segmentação de uma imagem em potenciais regiões, contendo sinais de trânsito (*ROIs – Regions of Interest*) (Fleyeh and Dougherty, 2005).

Cada região candidata identificada é submetida a um processo de reconhecimento, através de aplicação de um padrão que permita determinar se realmente se trata de um sinal. De seguida,

[12] Imagens obtidas a partir de https://www.youtube.com/watch?v=kJfa2HsTtlg

é feita uma classificação para uma determinada categoria: sinais triangulares, redondos, etc. (Fleyeh and Dougherty, 2005).

Após atribuição de uma categoria, é realizada uma análise denominada *Pictrogram Analysis* (Fleyeh and Dougherty, 2005) que, de acordo com as formas e texto presentes, determina qual é realmente o sinal de trânsito.

Por fim é realizada a transmissão de informações sobre o sinal reconhecido. Os autores defendem que a combinação das características de cor e de forma poderá trazer melhores resultados na identificação, ainda que não seja um requisito para um sistema *Traffic Sign Recognition* (Fleyeh and Dougherty, 2005).

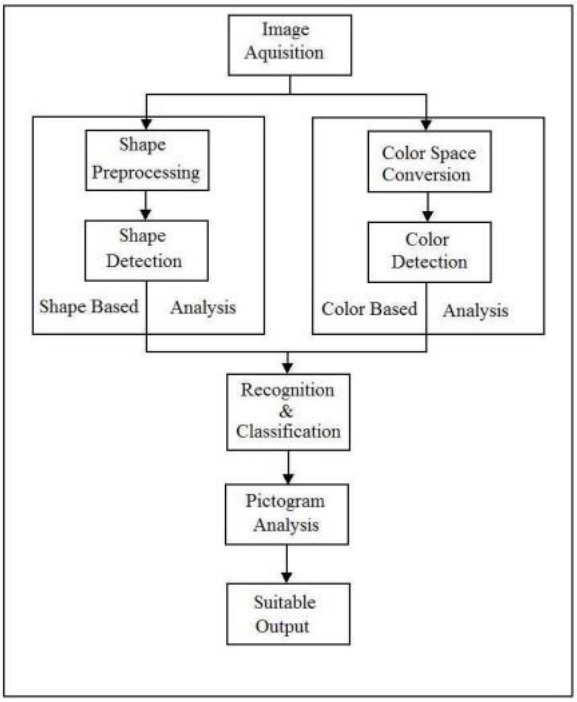


Figura 24 – Protótipo de Sistema de Reconhecimento definido por (Fleyeh and Dougherty, 2005).

2.5.3.1 Deteção

A fase de deteção permite identificar potenciais regiões da imagem contendo sinalização de trânsito. São realizados diferentes processos para extração dessas regiões, explicados nas secções seguintes.

2.5.3.2 Color Space

A utilização de um espaço de cores no processo de deteção e reconhecimento deverá ter em conta as restrições do mesmo face às variações de cores das imagens captadas. Os espaços de

cores poderão ser agrupados em quatro famílias principais, segundo E. Cernadas, M. Fernández-Delgado, E. González-Rufino e P. Carrión (Cernadas *et al.*, 2016). São elas:

- Espaços baseados em teorias tricromáticas, como o RGB e XYZ;
- Espaços baseados nos valores da cor e da luminosidade, ou seja, *luminance-chrominance*, como Lab, Luv e YUV;
- Espaços baseados na cor percebida, aproximação à cor visualizada do ponto vista do ser Humano, como HSI, HSV e HSL;
- Espaços baseados na independência e não correlação dos eixos das cores, como $I_1\ I_2\ I_3$.

A maioria dos dispositivos capturam imagens em *RGB* (Cernadas *et al.*, 2016). Devido à sensibilidade desse modelo a variações de luminosidade, poderá ser necessário converter o espaço de cores para um modelo aproximado à perceção Humana, já que a sinalização também está sujeita a variações de cores, devido à sua exposição ambiente e outros fatores.

A eficácia do sistema de identificação será fortemente influenciada pelo espaço de cores escolhido. O espaço de cores mais frequentemente usado nos algoritmos *TSR* é o *Hue Saturation Itensity (HSI)* (Maldonado-Bascon *et al.*, 2007)

2.5.3.3 Deteção Baseada em Cor

A abordagem *Color-Base* faz uso das características das cores para detetar potenciais regiões na imagem, que corresponderão a um sinal de trânsito, reduzindo assim os falsos-positivos no processo de identificação (Baro *et al.*, 2009). Em termos gerais, atribuem-se limites aos valores de *hue, saturation, value (HSV)* de uma imagem, por forma a encontrar zonas (denominadas regiões) com forte possibilidade de serem sinais de trânsito.

2.5.3.4 Deteção Baseada em Escala de Cinzentos

O método *gray-scale based* (baseado em escala de cinzentos) foca-se na forma e geometria do objeto. Através da conversão da imagem para escala de cinzentos, é potenciada a possível deteção de limites e formas geométricas, podendo ser usadas transformações do tipo *Hough Transformation.*

Hough Tranformation é um algoritmo utilizado em visão computacional, que permite reduzir as imperfeições numa determinada imagem, realizando aproximação a linhas e formas (Duda and Hart, 1972)

2.5.3.5 Segmentação

É um dos processos base na fase de deteção de um sinal trânsito a partir de uma imagem, com base nas suas características. Permite realizar a divisão da imagem em regiões e categorias, que corresponderão a diferentes objetos ou partes. (Glasbey and Horgan, 1995) Neste processo cada pixel é atribuído a uma categoria.

G. A. Glasbey e G.W. Horgan descrevem três tipos de segmentação (Glasbey and Horgan, 1995):

- Thresholding – Os pixéis da imagem são distribuídos em categorias de acordo com uma faixa de valores. Um píxel com um valor até X é colocado numa categoria, os restantes são colocados noutra. As diferentes categorias são demarcadas com limites a branco na imagem, permitindo perceber uma correta segmentação da imagem;

- Edge-based – Baseia-se na aplicação de um filtro "*edge filter"* sobre a imagem para deteção de limites, e os píxels são caracterizados como *edge* ou *non-edge*, sendo que, os pixels não *edge*, são agrupados na mesma categoria;

- Region-based – Este método permite agrupar de forma iterativa os pixéis mais próximos e com valores similares e separar aqueles que não têm semelhanças.

2.5.3.6 Reconhecimento

Existem diferentes métodos para reconhecimento de um objeto numa imagem. De acordo com revisão da literatura de visão computacional, pretende-se aqui destacar duas técnicas usadas para reconhecimento de sinalização vertical tendo por base um modelo pré-estabelecido: a *Template matching* e a *Feature matching.*

2.5.3.7 Template Matching

Em visão computacional, *Template-Matching* é uma técnica de identificação de partes de uma imagem, que contêm um *template* submetido para comparação (Figura 25).

Figura 25 – Exemplo de *Template Matching*[13]

Pretende-se comparar o *pixel* de cada coordenada da imagem original, com todos os pixels do *template*. O match de características é realizado segundo critérios de similaridade. (Adaptive Vision, 2017).

Cross-Correlation e Normalized Cross-Correlation são técnicas comuns para obtenção de similaridade entre os dois objetos a comparar. Destaca-se, no entanto, que a correlação cruzada é sensível a mudanças de intensidade/brilho.

2.5.3.8 Feature Matching

A técnica *Feature Matching* consiste na comparação de características (*features)* entre diferentes imagens, de forma a providenciar o reconhecimento de objetos representados na mesma cena.

Antes de ser descrito o funcionamento do método de comparação de características, é importante apresentar os conceitos chave que estão na origem deste processo: *Features, Features Detection, Features Extraction e Features Matching.*

Features

A caracterização de uma determinada imagem poderá ser realizada de uma forma global através de um descritor global, ou poderá ser caraterizada por cada região que a compõe.

Existem dois tipos de *Features* que podem ser extraídas de uma imagem: as *Global Features* e as *Local Features*.

Global Features

São características gerais presentes na imagem como a cor, textura e forma da imagem. Normalmente são representadas por um único vetor capaz de representar histogramas, texturas, limites ou outras características correspondentes a toda a imagem.

[13] Imagem retirada de http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html

Local Features

"A local feature is an image pattern which differs from its immediate neighborhood"

(Tuytelaars and Mikolajczyk, 2008)

As Local Features pretendem descrever numa determinada imagem, zonas de interesse de forma distintiva, que permitam potenciar a identificação de ocorrências de um determinado objeto, dentro de uma imagem. A representação das características locais de uma imagem é realizada através de um conjunto de vetores que descrevem cada ponto de interesse *(keypoint)* (Figura 26).

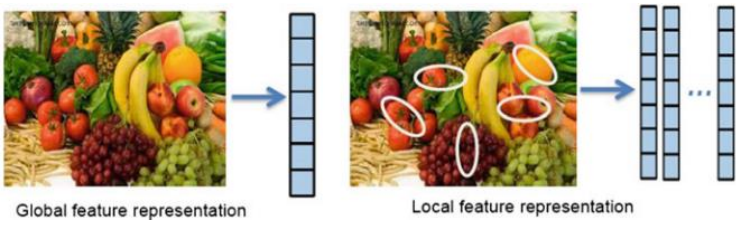


Figura 26 – Global Features and Local Features[14]

Aplicando estes conceitos no tema desta dissertação, verifica-se que as *Local Features* serão o elemento chave para a representação das características de uma determinada forma ou figura presente num sinal de trânsito, já que permitem representar distintivamente os pormenores das mesmas.

A representação dessas características e respetiva deteção será forte, se e só se, conseguirem atingir os seguintes requisitos: (Tuytelaars and Mikolajczyk, 2008)

- Repetibilidade - Na presença de duas imagens do mesmo objeto em diferentes perspetivas de visualização, a maior parte das características detetadas no objeto, deverão ser encontradas em ambas as imagens;

- Distintivas – A estrutura de representação das características do objeto deve apresentar variações, que permitam a sua distinção de outras estruturas;

- Locais – A representação de características deve ser local, por forma a evitar fenómenos de oclusão que interfiram na representação global da imagem;

- Quantidade – O número de características detetadas deve ser suficientemente grande, para garantia de uma taxa de certeza exata para efeitos de comparação. É claro que, dependendo do objeto, esse número poderá ou não atingir valores elevados;

- Eficácia – A sua deteção deverá, de forma eficaz, realizar a localização dos pixels que compõem determinadas características;

[14] Imagem presente no artigo em (M. Hassaballah, 2016)

- Eficiência – A deteção de características deverá prever uma deteção rápida, suportando a sua aplicabilidade em sistemas críticos.

Features Detection

A primeira etapa para a recolha das *Local Features* passa por submeter as imagens em comparação a um algoritmo de deteção de pontos de interesse, que começa pela definição na imagem, de elementos distintos que possam representá-la.

Considere-se, por exemplo, a deteção de pontos de interesse na forma presente no interior de um triângulo de sinalização de trânsito. Neste caso, o algoritmo de deteção rastreia a imagem através de um pequeno segmento e verifica, em cada interação, se os pixels nele contidos, conseguem estabelecer uma variação e uma relação distinta de intensidade.

Claramente, é possível verificar na Figura 27, que os cantos e os limites da forma são pontos muito mais descritíveis do que a região preta. Este tipo de análise é garantido pelos algoritmos de *Features Detection.*

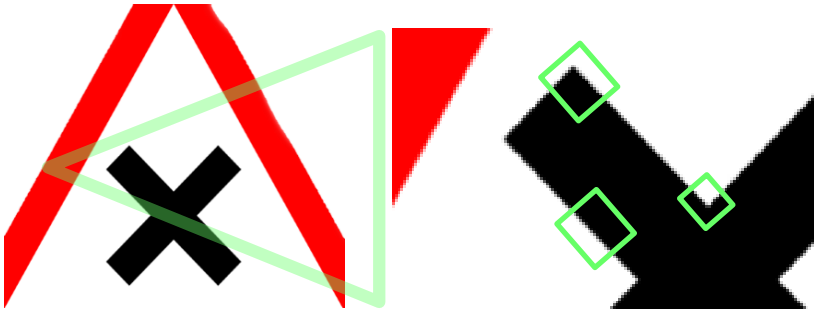


Figura 27 – Deteção de pontos de interesse

Os algoritmos de *Features Detection* podem ser divididos em 3 categorias (M. Hassaballah, 2016): *Single-Scale Detectors, Multi-Scale Detectors* e *Affine Invariant Detectors.*

Os algoritmos *Single-Scale Detectors* apenas garantem representação para as características de um determinado ponto de interesse. São invariantes a transformações realizadas sobre as imagens, como as rotações, translações, mudanças de luz de ruído. São, contudo, sensíveis a escalamentos. Assim sendo, para garantia de representação invariante a escalamentos, devem ser utilizados algoritmos de múltiplas representações (*Muti-Scale Detectors*).

Neste contexto, existe ainda a necessidade de prever que os escalamentos não são sempre uniformes, e podem ocorrer em diferentes orientações. Os algoritmos *Affine Invariant Detectors* pretendem ser invariantes a transformações afins sobre a imagem.

Dentro da panóplia de algoritmos existentes na literatura, distinguem-se aqueles que atuam com base na deteção de três elementos presentes na imagem: limites (*edges*), cantos (*corners*) e regiões de interesse (*blobs)*.

De acordo com a problemática desta dissertação, verifica-se que o processo de reconhecimento atuará sobre as formas integrantes de cada sinal. Devido à simplicidade e clareza da forma, o conjunto de detetores que atuam com base em deteção de cantos e *blobs* (interior do sinal monocromático) aparentam ser uma boa solução.

Com base nessas duas premissas, a respetiva popularidade e compatibilidade com as bibliotecas de processamento de imagem avaliadas nesta dissertação, foram revistos na literatura os algoritmos FAST e MSER.

Na seção "Features Extraction" são descritos também os algoritmos *SIFT- Scale Invariant Feature Transform, SURF- Speeded-Up Robust Features Descriptor e ORB - Oriented FAST and Rotated BRIEF* que agregam algoritmos de deteção de características, sendo que no caso do ORB, o mesmo incorpora a deteção via FAST.

FAST – Features from Accelerated Segment Test

FAST é um algoritmo *single-scale* de deteção baseado em deteção de cantos desenvolvido e apresentado por Edward Rosten e Tom Drummond em 2005 (Rosten and Drummond, 2005) e revisto em 2010 (Rosten, Porter and Drummond, 2010).

Surge da problemática relacionada com os elevados tempos de resposta dos algoritmos existentes para deteção de pontos de interesse, em aplicações de *tracking* e processamento de imagem em tempo real. (Rosten and Drummond, 2006).

Seguindo o princípio de que um canto poderá ser representado por uma cunha "*wedge"* (Rosten and Drummond, 2006), o algoritmo FAST ganha tempo no processo de deteção dos cantos, através da utilização de um segmento da imagem que tem como base o círculo "Bresenham" (Rosten and Drummond, 2006) presente na Figura 28. Neste caso, em vez de 8 pontos simétricos, são definidos 16 pontos (16 pixels) que englobam o ponto de interesse.

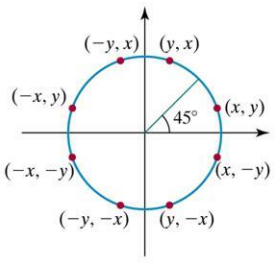


Figura 28 – Bresenham Circle[15]

Detalhando o funcionamento geral do algoritmo:

1. Identifica o círculo de 16 pixels em torno de um pixel ρ (Figura 29);

[15] Imagem retirada de:
http://www.eenadupratibha.net/pratibha/Engineering/cse_sem_out_pri_un2.html

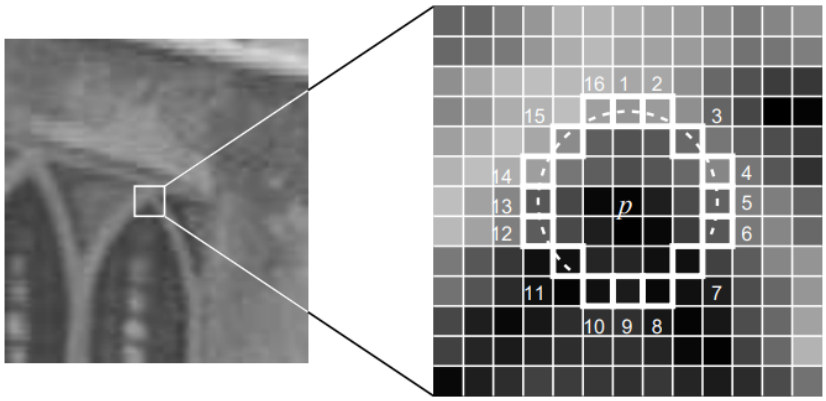


Figura 29 – Círculo de pesquisa de 16 pixels defendido por (Rosten and Drummond, 2006)[16].Os quadrados com borda a branco (16) são aqueles que são analisados no processo de deteção do canto. O pixel ρ representa o centro do canto candidato. A linha a tracejado, passa pelos $n = 12$ pixels que são validados se são mais brilhantes que o pixel ρ tanto ou mais que um limite definido.

2. Após identificar um pixel na imagem ρ como um ponto interesse, caracteriza a sua intensidade através de I_ρ;

3. Define um limite t para a diferença de brilho entre o pixel ρ e os n pixels em comparação;

4. Seleciona o candidato ρ , que será um canto, caso exista um conjunto de n pixels contínuos entre os 16 que sejam mais brilhantes em $I_\rho + t$ ou menos brilhantes em $I_\rho - t$;

5. Para a exclusão de falsos positivos, realiza um teste denominado "*high-speed test"*(Rosten and Drummond, 2006), que valida apenas os 4 pixels, com o número 1,9,5 e 13.

   Para os primeiros (1 e 9) verifica se, cada um deles é, em conjunto, os mais ou menos brilhantes. Caso se comprove, realiza o mesmo teste para os pixels 5 e 13.

   O pixel ρ representará um canto se pelo menos 3 destes 4 pixels forem mais brilhantes em $I_\rho + t$, ou os menos brilhantes em $I_\rho - t$. Em caso de positivo, o candidato passa para a validação com os restantes pixels.

O comportamento inicial deste detetor apresenta algumas vulnerabilidades (Rosten and Drummond, 2006) como:

[16] Imagem retirada de : https://www.edwardrosten.com/work/fast.html

1. A não exclusão de candidatos com o valor $n < 12$;

2. A escolha dos pixels não é otimizada porque está relacionada com a ordem e distribuição de apresentação dos possíveis cantos;

3. Os resultados do teste "*hight-speed test*" são voláteis;

4. São detetados múltiplos pontos de interesse adjacentes a outros.

Os autores defendem como melhoria do processo a inclusão de dois artefactos, *machine learning* para a resolução das vulnerabilidades (1,2,3) e *non-maximal suppression* para a resolução do ponto 4.

***Machine Learning* orientada à deteção de cantos**

O algoritmo FAST incorpora a técnica de aprendizagem baseada em máquina, mecanismo que treina uma série de modelos de forma a incorporar uma pequena base de conhecimento, para decisão de qual o melhor candidato. Esta abordagem, de acordo com (Rosten and Drummond, 2006), segue os seguintes passos:

1. Seleciona um conjunto de imagens para treino, de preferência imagens relacionadas com o domínio a que o algoritmo será submetido;

2. Aplica o algoritmo FAST a cada imagem selecionada, de forma a detetar pontos de interesse;

3. Para cada imagem, obtêm o vetor com os 16 pixels de cada pixel ρ, gerando no final (Quinlan, 1986)um vetor P com todos os 16 pixels pertencentes aos pontos ρ de todas as imagens;

4. Para cada pixel x em cada um dos blocos de 16 pixels, existirão 3 estados (*escuro, similar, brilhante)* descritos pela equação abaixo ilustrada (Figura 30);

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \leq I_p - t & \text{(darker)} \\ s, & I_p - t < I_{p\to x} < I_p + t & \text{(similar)} \\ b, & I_p + t \leq I_{p\to x} & \text{(brighter)} \end{cases}$$

Figura 30 – Estados possíveis para cada pixel do circulo de pesquisa.[17];

5. Com base nesses 3 estados, divide o vetor P em 3 subconjuntos P_d, P_s, P_b

[17] Imagem retirada de: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_fast/py_fast.html

6. Define uma variável K_p que representa se ρ é um canto ou não.

7. Aplica o algoritmo de decisão do tipo *decision tree classifier* denominado ID3 algorithm (Quinlan, 1986), que percorre cada subconjunto de P, utilizando a variável K_p, e seleciona qual o pixel x que apresenta mais informações sobre se o candidato é um canto, através da entropia K_p.

8. O processo repete-se recursivamente a todos os subconjuntos até a entropia ser 0.

9. A árvore de decisão, obtida após o treino é utilizada na deteção das imagens seguintes, acelerando o processo.

Non-maximal Suppression

É uma técnica capaz de pós-processamento que pretende interligar os resultados de deteção que pertençam ao mesmo objeto (Hosang, Benenson and Schiele, 2017).

O método FAST utiliza esta técnica para tratar o problema de deteção de cantos adjacentes. Para cada canto detetado é computada um função de pontuação V .Essa pontuação é comparada entre cantos adjacentes, sendo descartado o canto com menor pontuação

MSER - Maximally Stable Extremal Regions

É um algoritmo de deteção de regiões de interesse numa imagem (*blobs*) proposto por Jiri Matas em 2002 (Matas *et al.*, 2002).

Tem como objetivo principal a deteção de regiões covariantes de uma determinada imagem denominadas *Maximally Stable Extremal Regions*.

O seu funcionamento baseia-se no princípio de determinação de quais as regiões que se encontram próximas de um determinado intervalo de limites definido. Todos os pixels abaixo de um determinado limite são definidos como preto e acima do limite são brancos.

De acordo com o autor (Matas *et al.*, 2002), considerando todas as possibilidades para intensidade de uma imagem em escala de cinzentos (*gray-scale)* como I, será definido um intervalo de limites a aplicar sobre a mesma.

Exemplificando com base numa sequência de limites efetuados sobre uma imagem de acordo com cada intensidade dada por I_t em que t representa o limite definido, verifica-se que a primeira imagem é completamente branca e, nos *frames* seguintes, as zonas pretas (*local intensity minim*a) vão aumentando até convergirem numa imagem totalmente preta (Figura 31). As regiões que conseguem manter as formas inalteradas, através da aplicação dos limites em cada valor de intensidade, para um determinado limite (*maximal regions*), são selecionadas como regiões de interesse que depois passarão ao processo de extração.

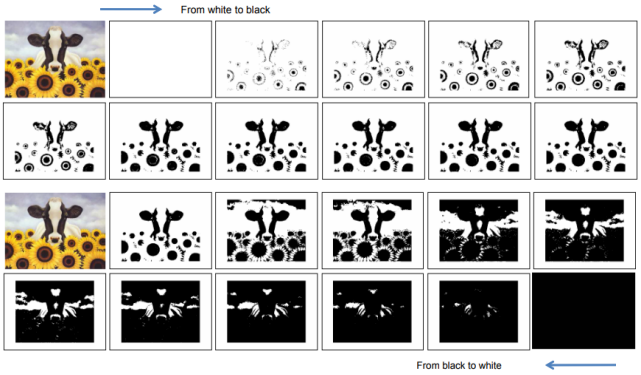


Figura 31 – Aplicação limites para cada intensidade de escala de cinzentos de forma a convergir resultados[18]

Os conceitos chave do algoritmo e a base matemática de apoio, são os descritos na Figura 32:

Image I is a mapping $I : \mathcal{D} \subset \mathbb{Z}^2 \rightarrow \mathcal{S}$. Extremal regions are well defined on images if:

1. $\mathcal{S}$ is totally ordered, i.e. reflexive, antisymmetric and transitive binary relation $\leq$ exists. In this paper only $S = \{0, 1, \ldots, 255\}$ is considered, but extremal regions can be defined on e.g. real-valued images ($\mathcal{S} = R$).
2. An adjacency (neighbourhood) relation $A \subset \mathcal{D} \times \mathcal{D}$ is defined. In this paper 4-neighbourhoods are used, i.e. $p, q \in \mathcal{D}$ are adjacent (pAq) iff $\sum_{i=1}^{d} |p_i - q_i| \leq 1$.

Region $\mathcal{Q}$ is a contiguous subset of $\mathcal{D}$, i.e. for each $p, q \in \mathcal{Q}$ there is a sequence $p, a_1, a_2, \ldots, a_n, q$ and $pAa_1, a_iAa_{i+1}, a_nAq$.

(Outer) Region Boundary $\partial\mathcal{Q} = \{q \in \mathcal{D} \setminus \mathcal{Q} : \exists p \in \mathcal{Q} : qAp\}$, i.e. the boundary $\partial\mathcal{Q}$ of $\mathcal{Q}$ is the set of pixels being adjacent to at least one pixel of $\mathcal{Q}$ but not belonging to $\mathcal{Q}$.

Extremal Region $\mathcal{Q} \subset D$ is a region such that for all $p \in \mathcal{Q}, q \in \partial\mathcal{Q} : I(p) > I(q)$ (maximum intensity region) or $I(p) < I(q)$ (minimum intensity region).

Maximally Stable Extremal Region (MSER). Let $\mathcal{Q}_1, \ldots, \mathcal{Q}_{i-1}, \mathcal{Q}_i, \ldots$ be a sequence of nested extremal regions, i.e. $\mathcal{Q}_i \subset \mathcal{Q}_{i+1}$. Extremal region $\mathcal{Q}_{i^*}$ is maximally stable iff $q(i) = |\mathcal{Q}_{i+\Delta} \setminus \mathcal{Q}_{i-\Delta}|/|\mathcal{Q}_i|$ has a local minimum at i^* (|.| denotes cardinality). $\Delta \in \mathcal{S}$ is a parameter of the method.

Table 1: **Definitions** used in Section 2

Figura 32 – Conceitos chave dos algoritmo MSER presente no *paper* ((Matas *et al.*, 2002))

Todos os pixels pertencentes a uma região MSER serão mais brilhantes que o seu limite ou mais escuros, no caso da inversão de intensidade, daí o nome *extremal*.

Features Extraction

No processo de deteção, são definidos pontos de interesse de uma determinada imagem, e posteriormente extração dessas características, concretizadas através de *Features Descriptors.*

Features Descriptors representam vetores de características, obtidos através de algoritmos de extração, que permitem, através dos pontos de interesse detetados (*keypoints)*, descrever as

[18] Exemplo retirado de: http://www.micc.unifi.it/delbimbo/wp-content/uploads/2011/03/slide_corso/A34%20MSER.pdf

suas características e codificá-las. Os mesmos são insensíveis a transformações, podendo sempre representar um ponto de interesse.

Existem diversos algoritmos de deteção e de extração/descrição presentes na literatura. Considera-se a sua divisão, em 5 famílias (Scott Krig, 2014):

- *Local binary descriptors* – Representam as características de uma imagem através de vetores de valores binários. São processados e comparados pares de pixels resultando em valores 0 e 1. São eficientes na utilização de processador, no seu armazenamento (estruturas mais pequenas), e robustas e eficientes no processo de *matching* através da utilização da distância *Hamming* e de comparação com base em operações XOR. Um descritor binário poderá ser composto por três etapas (Gil, 2013):
    - Padrão de amostragem (*sampling pattern)* - Permite representar os pontos em redor do ponto de interesse;
    - Compensação de orientação – Mecanismo responsável por determinar a orientação de um determinado ponto de interesse, aplicando uma rotação, de forma a prever efeitos relacionados com a variação de rotação;
    - Pares de amostragem – Define quais são os pares de comparação, para a construção do descritor final.
- *Spectra descriptors;*
- *Basis space descriptors;*
- *Polygon shape descriptors;*
- *3D, 4D and volumetric descriptors;*

No âmbito desta dissertação, com base no critério de popularidade e compatibilidade com as bibliotecas de processamento de imagem, são revistos os seguintes algoritmos agrupados por família:

- *Spectra descriptors;*
    - *Scale Invariant Feature Transform (SIFT, 1999);*
    - *Speeded-Up Robust Features Descriptor (SURF, 2006).*
- *Local binary descriptors;*
    - *Binary Robust Independent Elementary Features (BRIEF 2010);*

- *Binary Robust Invariant Scalable Keypoints (BRISK, 2011);*
- *Oriented FAST and Rotated BRIEF (ORB, 2011);*
- *Fast Retina Keypoints (FREAK,2012).*

- *Polygon shape descriptors;*
  - *Maximally Stable Extremal Regions (MSER, 2002).*

SIFT - Scale Invariant Feature Transform

Este algoritmo, proposto por *David G.Lowe* em 1999 (Lowe, 1999), permitiu resolver uma série de problemas relacionados com a extração de características de objetos numa imagem. O algoritmo é invariante a transformações como translações, variação de tamanho, rotações e outras transformações, bem como, efeitos de iluminação.

Por exemplo, a extração de um canto de um determinado objeto deverá ser invariante à escala em que o mesmo está representado numa imagem. Na Figura 33, verifica-se o cenário em que o algoritmo varia na deteção de acordo com o tamanho do objeto.

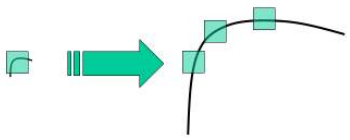


Figura 33 – Exemplo de identificação de cantos *Harris Corner Detector*[19]

O algoritmo SIFT atua com base quatro etapas (Lowe, 2004):

- *Detection of scale-space extrema* – É a fase em que se pretende determinar possíveis localizações e escalas da imagem. Faz uso do cálculo matemático *Difference of Gaussians* para deteção de regiões de interesse. Posteriormente, cada pixel é comparado em diferentes escalas e orientações, permitindo identificar corretamente *keypoints*;

- *Keypoint Localization* - Permite identificar de forma mais refinada a localização e escala que representa cada região candidata;

- *Orientation Assignment* – Para cada *keypoint* são atribuídas diferentes orientações baseadas nas propriedades iniciais da imagem, que permite a independência a fatores de rotação;

[19] Imagem retirada de http://docs.opencv.org/3.1.0/da/df5/tutorial_py_sift_intro.html

- *Keypoint Descriptor* – Através dos gradientes da imagem para cada *keypoint,* são representadas as regiões em torno do mesmo, através de vetores de 128 elementos.

SURF - Speeded-Up Robust Features Descriptor

É um algoritmo de deteção e descrição de *local features* desenvolvido e apresentado em 2006 por Herbert Bay (Bay *et al.*, 2007).

Foi desenvolvido como alternativa ao SIFT, já que revelou conseguir melhores tempos de computação e mais robustez (M. Hassaballah, 2016). Para deteção de pontos de interesse na imagem, o algoritmo utiliza a aplicação de filtros do tipo 2D box e deteção baseada no determinante da matriz Hessiana. Os filtros 2D box (Figura 34), permitem definir através de um quadrado de w*h pixels a respetiva diferença de gradiente

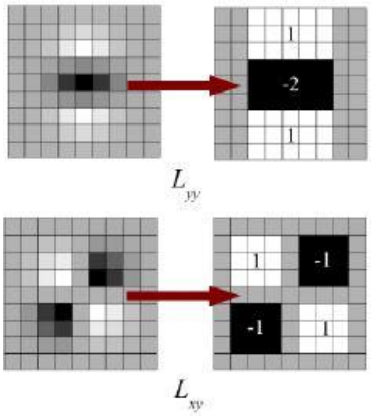


Figura 34 – Fitros 2D box[20]

A representação das *local features* é realizada através de um vetor de 64 dimensões, baseado na divisão do ponto de interesse em blocos de 4*4.

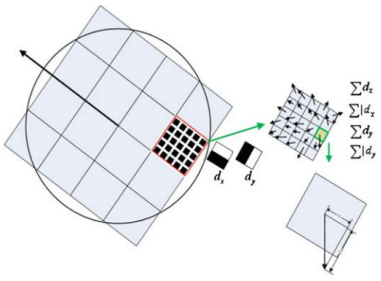


Figura 35 – Representação da divisão dos pontos de interesse em blocos 4*4 (M. Hassaballah, 2016)

[20] Imagem retirada de http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html

Este método, comparativamente com o SIFT, apresenta um *output* de informação mais reduzido, vetor 64 dimensões vs. 128 dimensões respetivamente, sendo mais limitado a transformações (M. Hassaballah, 2016).

BRIEF - Binary Robust Independent Elementary Features

É um algoritmo de descrição binária de características de uma imagem, proposto por Michael Calonder, Vincent Lepetit, Christoph Strecha e Pascal Fua em 2010. A motivação evidenciada está relacionada com a capacidade do algoritmo conseguir utilizar descritores de características mais pequenos, acelerando o processo de *matching* e reduzindo o consumo de memória (Calonder *et al.*, 2010).

O método proposto consiste na realização de uma comparação de intensidades entre pares de pontos, que permite alimentar e treinar árvores de decisão e modelos probabilísticos. Exemplo disso é o classificador do tipo *Naive Bayesian* (Calonder *et al.*, 2010), usado para o reconhecimento de segmentos de imagem através de diferentes pontos de vista.

Sobre um segmento da imagem definido pela dimensão SxS pixels, é aplicado uma desfocagem (redução de ruído) e são percorridos os pixels desse segmento realizando uma comparação da intensidade entre pares de pontos. O valor de cada bit do par é definido como 1 para o maior valor de intensidade, e 0 para o menor valor de intensidade:

$$\tau(\mathbf{p};\mathbf{x},\mathbf{y}) := \begin{cases} 1 & \text{if } \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & \text{otherwise} \end{cases},$$

Equação 1 – Comparação da intensidade entre pares de pontos, em que $p(x) e\, p(y)$ representa a intensidade de cada pixel respetivamente (Calonder *et al.*, 2010).

O resultado da comparação poderá ser definido através de um vetor de bites com tamanho de 128, 256 ou 512 posições. A escolha dos pontos em comparação é realizada através de uma seleção aleatória de uma distribuição Gaussiana isotrópica em torno do ponto de interesse (Calonder *et al.*, 2010).

A comparação de descritores *BRIEF* no processo de *matching* é realizada com base na distância *Hamming*, dando margem para um processo de reconhecimento altamente eficiente.

Este algoritmo responde a mudanças de luminosidade, desfocagem e transformações de perspetiva, sendo no entanto sensível a transformações de rotação (Angrish and Kaur, 2013).

BRISK - Binary Robust Invariant Scalable Keypoints

É um algoritmo de deteção e extração de características, apresentado por Stefan Leutenegger, Margarita Chli e Roland Y.Siegwart em 2011, que surge da necessidade de resposta a dois objetivos competitivos entre si: a descrição com alta qualidade, e os baixos requisitos de processamento (Leutenegger, Chli and Siegwart, 2011).

O processo de deteção de pontos de interesse tem como base os algoritmos FAST (descrito anteriormente) ou AGHAST. Por forma a atingir a invariância relativa ao escalamento, ao processo de deteção, é acrescentado um mecanismo de deteção da escala do ponto de interesse denominado: *Scale-Space Keypoint Detection* (Leutenegger, Chli and Siegwart, 2011).

Para a extração de características, o algoritmo faz uso de um padrão circular simétrico de 60 pares de pontos em torno do ponto de interesse (Figura 36), dispostos em 4 anéis concêntricos que permitem estabelecer um padrão de amostragem dos pontos vizinhos ao ponto de interesse detetado (Leutenegger, Chli and Siegwart, 2011). Aplica uma transformação do tipo desfocagem *Gaussian* para cada ponto de amostragem, para que consiga salvaguardar os efeitos do aliasing.

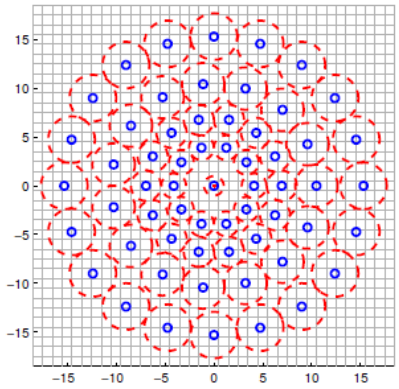


Figura 36 – *Sampling Pattern Brisk*[21]

O resultado deste algoritmo é composto por um conjunto de valores binários "*binary string" de* 512 bits, que representa a concatenação da comparação de testes de brilho.

ORB - Oriented FAST and Rotated BRIEF

ORB é um algoritmo de deteção e extração de características, desenvolvido por Ethan Rublee, Vincent Rabaud, Kurt Konolige and Gary.R.Bradski em 2011. Surge como uma boa alternativa aos algoritmos SIFTe SURF em termos de custos de computação e tempos de resposta (Rublee *et al.*, 2011).

Este algoritmo agrega a utilização do algoritmo FAST para a deteção de pontos de interesse, e do algoritmo BRIEF para a descrição, atingindo uma série de melhorias. No processo de deteção é aplicada a medida do algoritmo de deteção *Harris Corner* para a escolha dos melhores N pontos , com base em pontuação calculada (Rublee *et al.*, 2011). É também endereçado neste processo, o problema da produção de pontos de interesse em múltiplas escalas, que não é realizado pelo algoritmo FAST. ORB permite representar uma pirâmide de escalas da imagem, iterando a utilização do FAST em cada nível da pirâmide.

[21] Imagem retirada de: https://gilscvblog.files.wordpress.com/2013/08/brisk2.png

De forma a colmatar a falha de invariância a rotações do método BRIEF, os autores de ORB, desenvolveram um método denominado *Intensity Centroid*, que permite computar a orientação de um determinado canto determinado no processo de deteção. Com base na orientação definida para cada ponto de interesse, são extraídos os descritores através do algoritmo BRIEF.

O descritor ORB é definido através de um vetor de 256 bit, e resulta da aplicação de um algortimo *Greedy*, que permite determinar a menor correlação entre blocos de pixels (Yu, Yu and Gong, 2015).

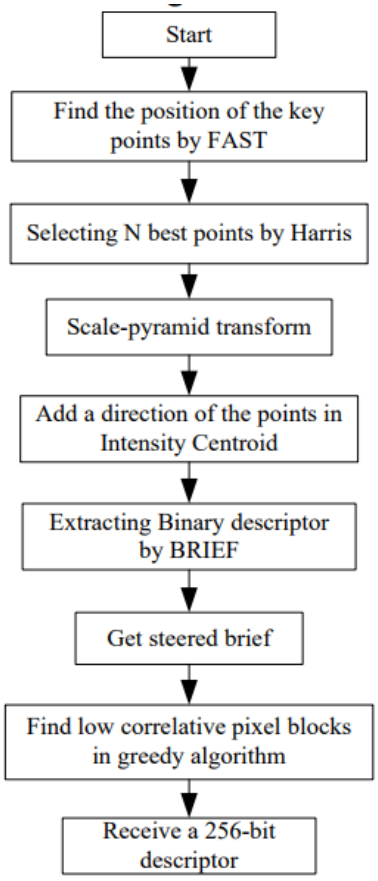


Figura 37 – Fluxograma do algoritmo ORB apresentado por (Yu, Yu and Gong, 2015)

Apesar da introdução de uma pirâmide de escalas neste algoritmo, os autores defendem que a invariância de escalas não foi totalmente endereçada (Rublee *et al.*, 2011).

FREAK - Fast Retina Keypoints

Surge da motivação e analogia com a visão Humana, nomeadamente a retina do olho, para a descrição de características de uma imagem.

Este algoritmo foi proposto por Alexandre Alahi, Raphael Ortiz e Pierre Vandergheynst em 2012, tendo como base o desenvolvimento de sistemas de computação gráfica em dispositivos *embebbed* e *smartphones,* em que os recursos são limitados. Tem como objetivo a definição de

um processamento mais rápido, mais compacto e mais robusto a escalamentos, rotações e ruído (Alahi, Ortiz and Vandergheynst, 2012).

Este algoritmo funciona através da aplicação de um padrão de amostragem semelhante à retina Humana (Figura 38), já que pretende dispor de forma exponencial a densidade dos pontos em torno de um ponto de interesse, definindo também uma ordem para futura comparação.

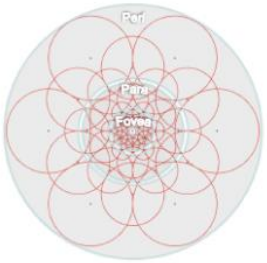


Figura 38 – Padrão de amostragem Freak[22]

O algoritmo FREAK permite compensar o fenómeno de rotação, realizando a deteção do valor da orientação do ponto de interesse, e aplicando a respetiva rotação aos pares de amostragem.

2.5.3.9 Dificuldades no reconhecimento de sinais de trânsito

É fundamental prever, no desenvolvimento da solução, as dificuldades no processamento e reconhecimento de sinais de trânsito numa imagem capturada.

Existem diferentes fatores que poderão condicionar o comportamento do sistema. A. de la Escalera e J.Ma Armingol, M. Mata (Karthiga.PL, 2016), definem o seguinte conjunto de dificuldades (Figura 39):

1. Condições de luminosidade, que estão sempre em variação ao longo do dia, quer seja pelas diferentes etapas do dia, nuvens, ou outras condições climatéricas;

2. A integridade de todo o sinal na imagem, já que, num cenário real, existem sempre outros objetos que estão perto da sinalização que podem obstruir a total visualização e criar sombra;

3. A impossibilidade de pré-carregamento de todas as aparências possíveis de um sinal, já que existem infindáveis possibilidades.

4. O tamanho do sinal a reconhecer, que varia conforme a distância a que câmara de captura estiver do sinal;

[22] Imagem retirada de: https://gilscvblog.com/2013/12/09/a-tutorial-on-binary-descriptors-part-5-the-freak-descriptor/

5. A orientação da captura, que influencia a escala dos eixos, alterando a disposição do sinal na imagem.

Os autores assinalam também como fator importante, as condições físicas dos sinais, relacionadas com a idade e integridade dos mesmos (sinais rodados, forma alterada, etc.).

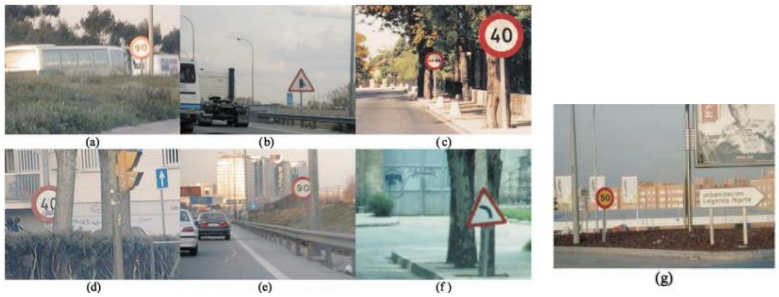


Figura 39 – Dificuldades presentes na deteção de sinais (Karthiga.PL, 2016)[23]

2.5.3.10 Abordagens Traffic Sign Recognition

Nesta secção serão descritas, de forma breve, algumas abordagens relevantes para sistemas *Traffic Sign Recognition* presentes na literatura.

Abordagem Support Vector Machines

Saturnino Maldonado-Bascón, Sergio Lafuente-Arroyo, Pedro Gil-Jiménez, Hilario Gómez-Moreno, Member, e Francisco López-Ferreras descrevem um sistema de deteção e reconhecimento de trânsito vertical baseado em *Support Vector Machines (SVM)* (Maldonado-Bascon *et al.*, 2007). O sistema contempla as seguintes fases: Segmentação; Classificação de formas; e Reconhecimento, representadas no algoritmo sistematizado na Figura 40.

[23] Traffic sign detection problems. (a) Reflections; (b) not controlled lighting; (c) shadows; (d, e) partial occlusions; (f) sign rotation; (g) shape deformation.

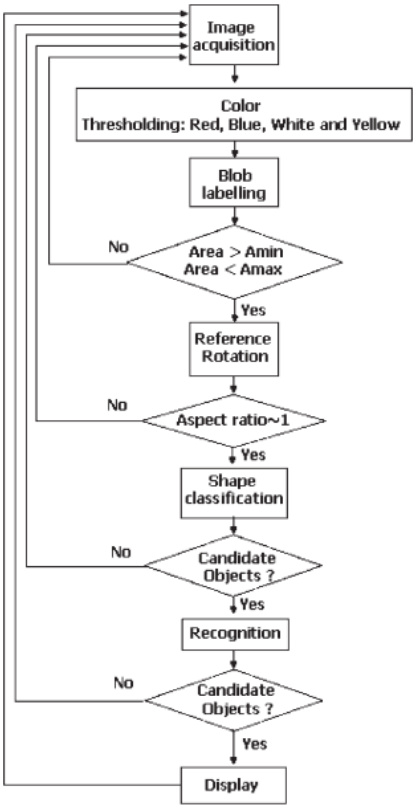


Figura 40 - Algoritmo definido em (Maldonado-Bascon *et al.*, 2007)

Na Segmentação, são geradas as respetivas regiões de interesse (*blobs*) através de aplicação de limites sobre o espaço de cores HSI para sinais cromáticos (sinais de cor), sendo que, para a sinalização branca é realizada uma decomposição acromática.

Para obtenção de limites a aplicar, foram construídos histogramas de *hue* e *saturation* para as cores vermelho, azul e amarelo. Para uma melhor definição dos limites a aplicar no sistema, foram capturadas imagens sob diferentes condições climatéricas, condições de luz e diferentes câmaras. Para obtenção de regiões com possíveis sinais de cor branca, foi utilizada a seguinte função (Figura 41):

$$f(R,G,B) = \frac{(|R-G| + |G-B| + |B-R|)}{3D}$$

Figura 41 – Decomposição de cor sugerida em (Maldonado-Bascon *et al.*, 2007)

"[…] where R, G, and B represent the brightness of the respective color, and D is the degree of extraction of an achromatic color, and in our case, we get the best segmentation results by setting D to 20. An f(R, G, B) of less than 1 represents achromatic colors, and an f(R, G, B) of greater than 1 represents chromatic colors […]" (Maldonado-Bascon *et al.*, 2007)

Após obtenção de possíveis regiões, é realizada uma seleção com base em critérios de tamanho, *aspect ratio*. Os *blobs* que sejam menores que a área mínima, e maiores que a área máxima desejada, são descartados. Após geração dos *blobs* de interesse, (*BoI*), os mesmos são submetidos a uma rotação, uma vez que se pretende que o cenário ideal seja a perpendicularidade da captura face à posição original do sinal.

Depois da fase da segmentação, os *BoI* são classificados através de *linear Support Vector Machine (SVM)*. É realizado um mapeamento entre a cor e o tipo de formas possíveis de identificar pela *SVM,* de acordo com a Figura 42.

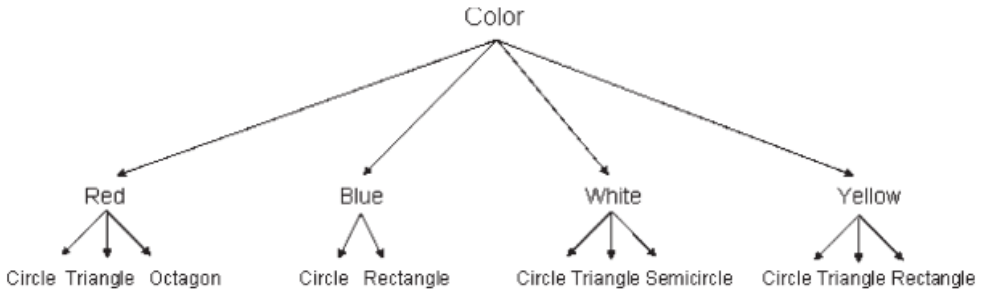


Figura 42 – Determinação de possíveis formas através das cores dos *BoI*

Na utilização das *SVMs,* importa realçar o uso de vetores do tipo *DtBs*, que representam a distância entre os limites dos *BoI* para os limites da *bounding box*, identificada anteriormente. Após classificação dos *BoI* o processo de reconhecimento baseia-se na utilização também de *SVMs*.

Abordagem Feature Matching

FeiXiang Ren, Jinsheng Huang, Ruyi Jiang e Reinhard Klette propõem uma abordagem para identificação de sinalização trânsito, utilizando o método *Feature Matchin*g (Ren *et al.*, 2009). O sistema proposto percorre três passos: *Image Preprocessing, Feature Extraction, Feature Matching and Recognition.*

Na fase de pré-processamento é realizada uma conversão do espaço de cores *RGB* para *HSV*, por forma a combater a sensibilidade do *RGB* à iluminação e fatores relacionados com o ambiente, incidentes na captura da imagem. São definidos limites para os valores de *Hue*, para sinalização com limites a vermelho entre 230 e 250, azul entre -10 e 10, verde entre 110 e 130, e amarelo entre 170 e 190. De seguida na região identificada é realizada uma extração de formas que possam existir, como círculo, triangulo ou quadrado de acordo com a cor extraída.

> "The extraction of a small patch will save computation time and improve the detection rate when working on the whole image"
>
> **(Ren *et al.*, 2009)**

É utilizada uma transformação *Hough transform* sobre o *edge map*, detetando-se na região a existência desses elementos. No caso dos círculos é aplicada uma transformação *Hough Circle*

Transform. O processo atribui restrições quanto ao tamanho, de forma a potenciar a seleção. Para identificação de triângulos é realizada uma transformação *Hough line transform* seguida de um algoritmo para identificação de polígonos. São aplicadas, nesta fase, restrições para consideração de potenciais sinais, como o tamanho do polígono detetado e os graus do triângulo, que deverão estar perto de 60 graus. Para os quadrados é realizado o mesmo processo, mas consideram-se ângulos de 90 graus.

Após a extração dos potenciais sinais, é realizada uma normalização. A forma extraída é redimensionada para o tamanho do *template*, neste caso (128*128). O aspeto e a geometria do objeto são normalizados através de transformação de perspetiva, e transformação afim de forma a colocar a forma na posição ideal, ou seja, *frontal orthogonal projection*.

O próximo passo do sistema é o *Feature extraction and matching*. Através da utilização dos algoritmos Scale-invariant Feature Transform (SIFT) e Speeded Up Robust Features (SURF) é realizada a comparação das características entre as *RoI* e as características dos *templates* armazenados em base de dados, e são definidos limites de características para aceitação da forma como um sinal de trânsito (Figura 43).

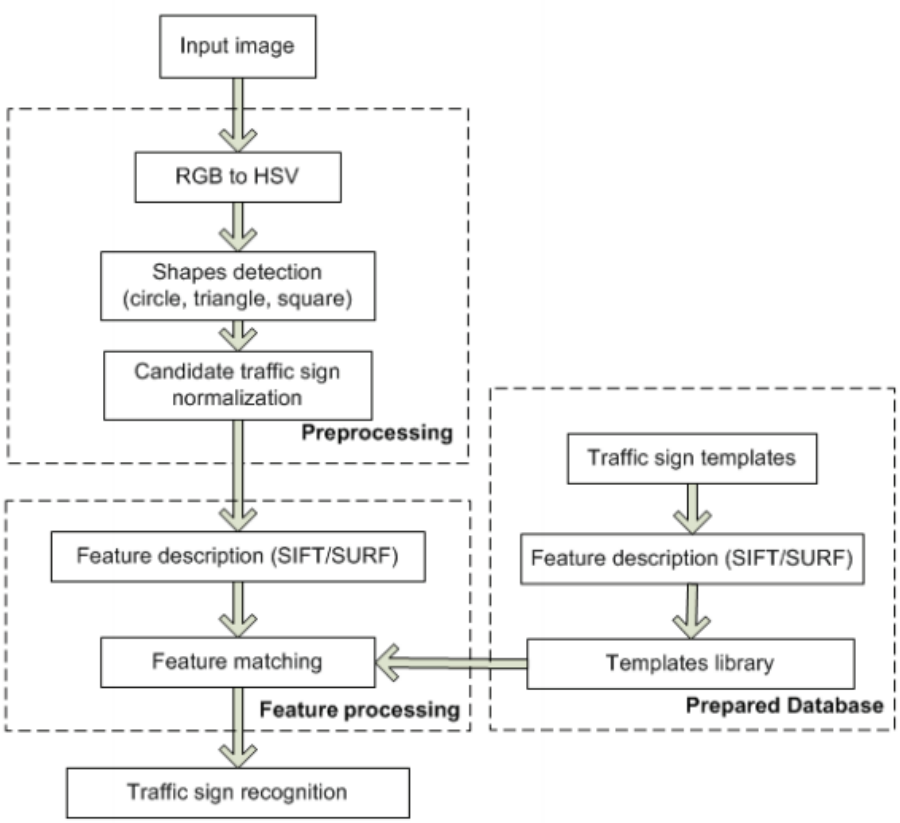


Figura 43 - *FlowChart* do sistema proposto (Ren *et al.*, 2009)

2.5.4 Bibliotecas de Visão Computacional

"[…]" Vision is a classic example of a problem that humans handle well, but with which machines struggle. As you go through your day, you use your eyes to take in a huge amount of visual information and your brain then processes it all without any conscious thought. Computer vision is the science of creating a similar capability in computers and, if possible, to improve upon it [...]"

(Kurt Demaagd, Anthony Oliver, Nathan Oostendorp, 2012)

Existem várias bibliotecas para processamento de imagem e realização de operações de visão computacional. Pretende-se aqui descrever algumas dessas bibliotecas, tendo como pressuposto os requisitos da solução.

2.5.4.1 SimpleCV

SimpleCV é uma *framework Simple Computer Vision,* desenvolvida em *Python,* e executável nas plataformas Windows, Mac OS e Ubuntu Linux.

Apresenta como uma das principais mais-valias a facilidade de utilização (Código 1), disponibilizando bibliotecas e algoritmos para resolução de problemas de visão computacional. Por exemplo, a segmentação baseada em cor poderá ser dada por instruções bastante simples e atómicas (Código 1).

Desconstruindo o exemplo presente no livro *"Pratical Computer Vision with SimpleCV"* (Kurt Demaagd, Anthony Oliver, Nathan Oostendorp, 2012), na Figura 44, onde está presente uma pistola de cola amarela, verifica-se que o pretendido é a realização da segmentação da cor amarela que representa o objeto, através da função *colorDistance()*. Esta função passa a cor amarela pelo espaço *RGB*, e devolve a imagem em escala de cinzentos: as cores aproximadas com o amarelo são devolvidas em tons pretos, as restantes cores (neste caso o fundo com tonalidade roxa) representadas com branco.

Seguidamente, a função *binarize()* transforma a imagem para que apresente apenas os tons preto e branco (imagem binária). Através da passagem de um limite com valor 50 (o valor poderá ir de 0 a 255, em que 255 é a cor branca), é indicado que cada pixel da imagem em escala de cinzentos com um valor abaixo dos 50, será convertido em cor branca e os restantes em cor preta.

Para colocar a pistola com a cor preta, realiza-se a inversão através da função *invert()*. A segmentação apenas do objeto, realiza-se através da subtração simples entre a imagem original e a imagem binária.

```
from SimpleCV import Image, Color

yellowTool = Image("yellowtool.png")

yellowDist = yellowTool.colorDistance((223, 191, 29))

yellowDistBin = yellowDist.binarize(50).invert()

onlyYellow = yellowTool - yellowDistBin

onlyYellow.show()
```

Código 1 – Exemplo de segmentação pela cor amarela (Kurt Demaagd, Anthony Oliver, Nathan Oostendorp, 2012)

Figura 44 – Imagem segmentada (Kurt Demaagd, Anthony Oliver, Nathan Oostendorp, 2012)

2.5.4.2 OpenCV

*O*penCV é uma biblioteca *open source* para o desenvolvimento de aplicações de visão computacional concebida inicialmente pela *Intel®* em 2000. A sua implementação rege-se pela licença *BSD license*.

A biblioteca foi desenvolvida nas linguagens C e C++, e executa em diferentes plataformas, como *Windows, Linux, Mac OS, Android* e *IOS*.

Assume-se como uma biblioteca robusta e eficiente para computação de aplicações em tempo real, (Bradski *et al.*, 2008), tirando partido do processamento *multi-core.* Suporta aceleração através de *hardware* com utilização de *OpenCL*

Atualmente a biblioteca *OpenCV* é amplamente utilizada pela comunidade, estimando-se 47 000 utilizadores e cerca de 9 milhões de downloads realizados [24].

A estrutura da biblioteca é composta por cinco elementos (Figura 45) principais, CV, MLL, HighGUI e CXCORE (Bradski *et al.*, 2008). CV contem todos os comportamentos relacionados com o processamento de imagem e algoritmos de visão computacional. MLL disponibiliza

[24] Informação retirada através do site http://opencv.org/

funcionalidades de *machine learning* como classificadores e *clustering tools*. HighGUI inclui todas as operações de interface e input/output de imagem para carregamento e armazenamento de conteúdos. CXCORE define a estrutura de dados e outra funções.

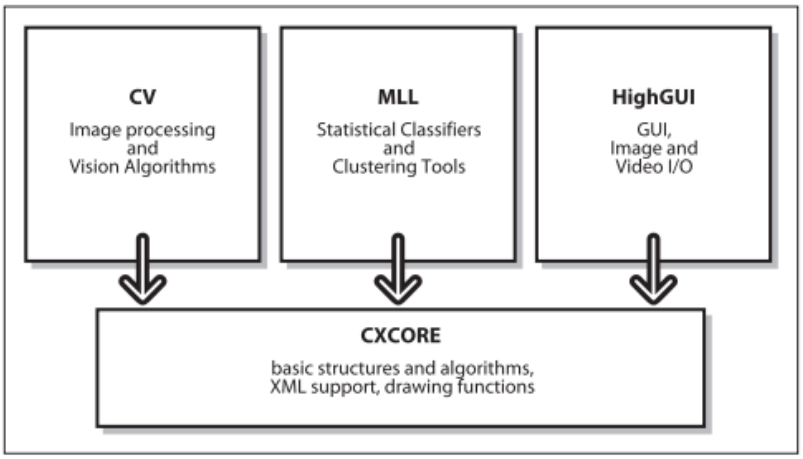


Figura 45 - Estrutura OpenCV (Bradski *et al.*, 2008)

2.5.4.3 EmguCV

Emgu CV é um *wrapper* da *framework .Net* para a biblioteca de processamento de imagem da *OpenCV*. O mesmo permite a invocação de funções *OpenCV* a partir de linguagens familiares como o *C#, Visual Basic, Visual C++, IronPython* entre outras.

Esta API apresenta-se como multiplataforma, sendo totalmente escrita em C#, que permite ser compilada em Mono e executada em sistemas operativos como *Windows, Linus, Mac OS X, iOS, Android e Windows Phone*. O *EmguCV* permite a integração com os IDEs Visual Studio, Xamarin Studio e Unity.

A licença de utilização deste *wrapper* divide-se em duas componentes: a componente *open source* e na componente comercial. Esta divisão é feita consoante a utilização dada aos projetos que envolvam esta API. Se forem tomados públicos à comunidade open source a sua utilização é livre, caso contrario trata-se de uma licença comercial. A última versão (3.1.0-r16.12) desta API foi lançada em 17 de dezembro de 2016. Verifica-se, no entanto, que no caso da versão destinada ao desenvolvimento para dispositivos móveis, é necessário adquirir uma licença comercial, que apresenta um preço mínimo por programador de cerca de 399 dólares[25].

Na Figura 46 está representada a arquitetura da API Emgu CV. Destaca-se aqui que, na sua estrutura, há uma separação em duas camadas para o processamento de imagem: na primeira

[25] Informação retirada de: http://www.emgu.com/wiki/index.php/Commercial_License_Purchase#Commercial_License_3.0_Professional_Version. O valor da licença individual poderá ser diminuído com a aquisição de uma licença múltipla (*Work Group por Enterprise*)

camada estão representados todos os mapeamentos de estruturas e comportamentos relacionados com a biblioteca *OpenCV*,; na segunda camada é são incluídas as classes que fazem a ponte para a framework .NET. Cada componente principal da arquitetura está contida numa dll que poderá ser incluída em diferentes projetos em ambiente de desenvolvimento. São estas:

- .Net Utilities - Contêm uma coleção de utilidades .NET comuns;
- Image Processing – Disponibiliza um conjunto de algoritmos de processamento de Imagem da biblioteca *OpenCV*;
- Presentation – Controlos gráficos do Emgu;
- GPU Processing – Funcionalidades de processamento sob GPU Nvidia Cuda;
- OpenCL – Capacidade utilização de computação paralela sobre CPU e GPU;
- Machine Learning – Algoritmos de aprendizagem e classificação.

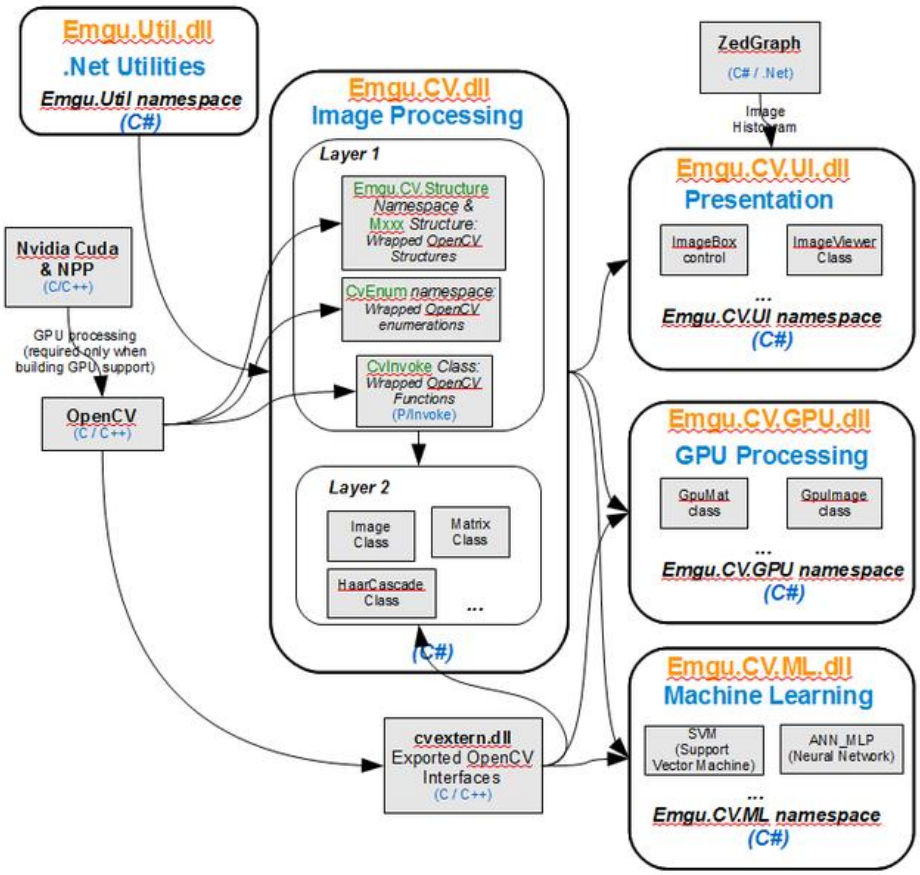


Figura 46 – Arquitectura EmguCV [26]

[26] Imagem retirada do site http://www.emgu.com/wiki/index.php/File:EmguCVArchitecture.png

2.5.4.4 AForge.NET

É uma *framework open source* para desenvolvimento de soluções de visão computacional, processamento de imagem e inteligência artificial, desenvolvida originalmente por *Andrew Kirillow* (Microsoft, 2017). Foi totalmente escrita em C# e disponibiliza o seguinte conjunto de componentes principais (AForge.NET, no date):

- AForge.Imaging – Biblioteca de rotinas para processamento de imagem e aplicação de filtros. Destacam-se alguns comportamentos como *Edge Detectors* e *Hough Transformations;*

- AForge.Vision – Biblioteca de visão computacional. Disponibiliza algoritmos para deteção e processamento de movimentos;

- AForge.Math – Biblioteca que permite agregar diferentes algoritmos matemáticos utilizados internamente pela *framework,* e que podem ser utilizados por outros fins;

- *AForge.Video* – Biblioteca para processamento de vídeo. Estão implementadas aqui funções para obtenção e escrita de ficheiros de vídeo, como AVI;

- *AForge.*Robotics – Biblioteca que disponibiliza algumas funcionalidades para utilização de robots (ex. Lego);

- AForge.Neuro – Biblioteca para programação utilizando redes neuronais;

- AForge.Genetic – Biblioteca para programação de algoritmos genéticos;

- AForge.Fuzzy – Conjunto de funcionalidades para programação de problemas de lógica difusa;

- AForge.MachineLearning – São disponibilizadas classes para manipulação algoritmos de aprendizagem.

Esta *framework* atua com base no licenciamento *LGPL v3* e a sua primeira versão foi lançada em 2006, tendo sido terminado o *road map* em 2013 com a versão 2.2.5.

Novos desenvolvimentos foram introduzidos através do seu encapsulamento na *framework Accord.NET* desenvolvida por César Souza em 2005 (Microsoft, 2017), que permitiu estender as suas funcionalidades.

2.5.5 Desenvolvimento em Dispositivos Móveis

O desenvolvimento da solução recorrendo à utilização dos dispositivos móveis, carece do reconhecimento geral dos sistemas operativos atualmente existentes no mercado.

Pretende-se nesta secção descrever o funcionamento da arquitetura dos três sistemas operativos mais utilizados na atualidade[27], o *Android, iOS e Windows Phone* respetivamente.

2.5.5.1 Android

É um sistema operativo (SO) basedo no *kernel Linux* divulgado inicialmente pela parceria *Open Handset Alliance em 2005,* e o seu código fonte, suportado pela Google. É um SO de código aberto que poderá conter software de código fechado, como desenvolvimento de aplicações proprietárias.

Atualmente o SO *Android* está desenvolvido na versão 8.0 *Oreo* [28].

A arquitetura do *Android* subdivide-se em diferentes camadas (Figura 47), possibilitando a sua alteração e utilização pelas partes interessadas. As operações sobre *hardware* são realizadas através do *Kernel Linux*, não acessível às aplicações, diretamente. A plataforma disponibiliza uma camada de abstração de *hardware HAL,* que permite regular o acesso ao *hardware* pelas camadas acima (Google, 2017b).

O SO Android garante a sua execução em dispositivos móveis através do *Android runtime (ART).* (Google, 2017a). Este componente permite executar múltiplas máquinas virtuais, em que as instruções são processadas independentemente do *hardware* físico do dispositivo, através de ficheiros DEX. Inicialmente as versões do *SO* inferiores à 5.0, executavam segundo a máquina virtual Dalvik, baseada na máquina virtual Java, otimizada para *Android.*

Destacam-se as bibliotecas nativas escritas nas linguagens C e C++, considerando as funcionalidades gráficas e persistência de dados. A biblioteca *android.database.sqlite* [29] disponibiliza um conjunto de classes para manipulação de uma base de dados local. A exploração destas bibliotecas poderá ser realizada pela camada acima *Java API Framework.*

A Java API Framework disponibiliza um conjunto de comportamentos que permite a construção de aplicações *Android* de forma modular e de alto nível. Por defeito, o *SO*, já disponibiliza um

---

[27] Estatística recolhida de http://www.idc.com/promo/smartphone-market-share/os

[28] Página oficial Android https://www.android.com/intl/pt_pt/

[29] Detalhe de biblioteca retirado de https://developer.android.com/reference/android/database/sqlite/package-summary.html

conjunto de aplicações de sistema base como SMS, Câmara entre outros, sendo possível, a aplicações de terceiros, o acesso direto a estas funcionalidades.

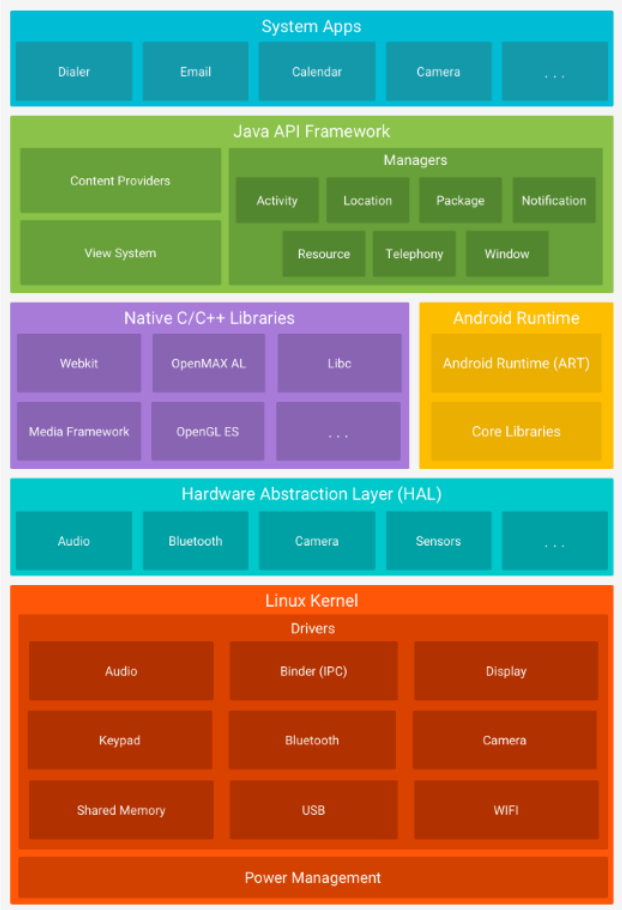


Figura 47 – Arquitectura Android[30]

2.5.5.2 iOS

É o sistema operativo móvel desenvolvido e patenteado pela *Apple Inc.* para os seus dispositivos *iPhone, iPod Touch* e *iPad.*

Rege-se por uma arquitetura em camadas (Apple, 2017), em que o nível de abstração aumenta desde da camada inferior até à de topo. A camada *Core OS*, incide em si, funcionalidades nucleares como a gestão de memória, gestão de ficheiros e operações com *hardware*.

A camada *Core Services* fornece tecnologia para suporte a funcionalidades como localização, *iCloud*, gestão de base dados (*SQLite*), entre outros.

[30] Imagem retirada de https://developer.android.com/guide/platform/index.html?hl=pt

Media contêm tecnologias para implementação de multimédia nas aplicações. As principais funcionalidades são a reprodução de áudio, vídeo, biblioteca *OpenGL* entre outras.

A camada de topo a *Cocoa Touch* disponibiliza um conjunto de bibliotecas para o desenvolvimento de aplicações, como *Document Picke, AirDrop, TextKit,* entre outras.

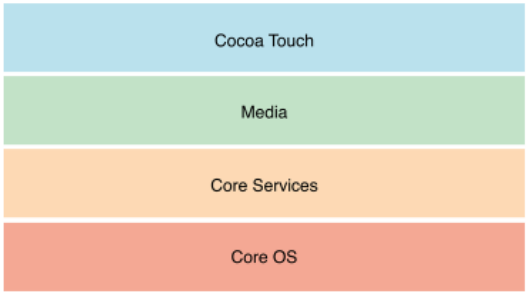


Figura 48 – Arquitectura iOS[31]

Atualmente a versão do SO é a *iOS 11.*

2.5.5.3 Windows

Windows Phone ou recentemente *Windows 10 Mobile*, é o sistema operativo proprietário da *Microsoft* para dispositivos móveis.

Com o lançamento do Windows 10 em 2015 a *Microsoft* pretendeu unificar as diferentes plataformas através das "*Universal Apps"* (Figura 49).

Através da experiência *Universal Windows Plataform (UWP) app,* o utilizador poderá executar aplicações *Windows* em qualquer dispositivo sem necessidade de especificidades do dispositivo.

Universal Windows Platform – UWP

O *Windows 10* disponibiliza a UWP como plataforma comum para execução de aplicações em qualquer dispositivo. Esta plataforma permite que as aplicações possam fazer uso de *APIs* para além daquelas que o *Windows Runtime* permitia (Microsoft, no date).

O desenvolvimento destas aplicações poderá ser realizado através de linguagens mais familiares para os programadores, como por exemplo *Javacript e Html.*

[31] Imagem retirada de: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html#//apple_ref/doc/uid/TP40007898-CH1-SW1

Figura 49 – *Windows 10 Universal Apps* [32]

2.5.6 Desenvolvimento em Multiplataforma

Com a evolução tecnológica e a proliferação dos dispositivos móveis, nomeadamente os *smartphones*, foram surgindo novas plataformas, e as que existiam foram sujeitas a uma constante mutação.

A diversidade de recursos e plataformas disponíveis para o utilizador final, aumenta o desafio de quem pretende desenvolver uma solução para dispositivos móveis. O desenvolvimento de uma aplicação em linguagem nativa poderá acarretar um maior custo de fabrico, uma manutenção mais complexa e um *time-to-market* mais longo. Uma das desvantagens desta abordagem é a possível replicação de código.

Existem diferentes formas de solucionar este problema. Poderá ser utilizada uma abordagem *Cross-Compilation,* que significa que a programação da solução poderá ser realizada numa determinada linguagem, diferente da linguagem da plataforma de destino, onde a solução é compilada por um compilador multiplataforma que transforma as instruções para a linguagem nativa.

Poderá ser utilizada a virtualização, já que, através de uma máquina virtual é possível a abstração da linguagem nativa, no entanto, a execução deste tipo de técnicas é mais lenta.

Outra abordagem poderá ser o desenvolvimento de uma solução *Web*, em tecnologias transversais e executadas de forma ampla pelos *browsers,* como HTML, *Javascript* e CSS.

De acordo com as *frameworks* de desenvolvimento de visão computacional analisadas anteriormente, a utilização da linguagem C# é um ponto em comum (exceto no caso do *SimpleCV*). Dessa forma, e após análise do mercado tecnológico, identifica-se o Xamarin como uma potencial solução para o desenvolvimento multiplataforma, que faz uso da linguagem C# e é totalmente compatível com a biblioteca EmguCV.

[32] Imagem retirada de https://docs.microsoft.com/pt-pt/windows/uwp/get-started/universal-application-platform-guide

2.5.6.1 Xamarin SDK

É um *Software Development Kit (SDK) open source* para o desenvolvimento de aplicações móveis multiplataforma, criada pela *Xamarin* (adquirida pela *Microsoft* [33]) em 2011. A plataforma disponibilizada é atualmente utilizada por cerca de 15000 clientes, envolvendo 1,4 milhões de programadores em todo o mundo[34].

O seu principal foco vai para a capacidade de desenvolvimento de alto nível de soluções para plataformas como *iOS, Android* e *Windows Phone*, maximizando a reutilização de código, através da utilização de Xamarin Forms, *"Applications can be written to share up to 90% of their code"*[35].

Esta plataforma tem como base o projeto *Mono*, para garantia da execução da aplicação em diferentes plataformas. No caso do *iOS* o código é compilado à cabeça (AOT) para linguagem nativa *ARM assembly code* e no Android código é realizada a compilação *Just in Time – JIT* e a execução virtualizada através de *MonoVM + JIT'ing.* (Xamarin, 2017b). O *Xamarin Mobile Profil*e permite englobar a chamada aos compiladores multiplataforma para *iOS (MonoTouch) e Android (MonoAndroid)* respetivamente.

Através da utilização da linguagem C#, permite a interoperabilidade com as linguagens nativas como *Objective-C, Swift e Java* para desenvolvimento de especificidades

O *Xamarin* permite o desenvolvimento nativo das interfaces de cada plataforma através de *APIs* como *MonoTouch.UIKit (iOS), Android.Views (Android)* ou XAML (Windows).

Figura 50 – Arquitectura Xamarin [36]

[33] Anúncio de acordo para aquisição https://blogs.microsoft.com/blog/2016/02/24/microsoft-to-acquire-xamarin-and-empower-more-developers-to-build-apps-on-any-device/

[34] Informação obtida em https://www.xamarin.com/about

[35]Citação retirada de https://developer.xamarin.com/guides/cross-platform/getting_started/introduction_to_mobile_development/

[36] Retirado de https://developer.xamarin.com/guides/cross-platform/application_fundamentals/

A arquitetura deste *SDK* permite realizar a separação dos outros componentes da aplicação, através de uma camada inferior e independente denominada *Core Library (*Figura 50*).* A taxa de reutilização varia de acordo com a quantidade de recursos que são partilhados nesta camada. (Xamarin, 2017b).

Destacam-se alguns serviços transversais como o acesso a base de dados como *SQLite-NET,* e *Xamarin Plugins* para acesso a componentes do dispositivo como a câmara, localização e outros.

Destacam-se dois ambientes de desenvolvimento, o *Xamarin Studio IDEI* e a integração com o IDE da *Microsoft,* o Visual Studio, através de *plugin.*

2.5.6.2 Xamarin Forms

Representa um conjunto de ferramentas (*toolkit)* disponível no Xamarin SDK para o desenvolvimento da interface com utilizador multiplataforma, abstraindo os seus utilizadores das plataformas nativas Android, iOS e Windows (Xamarin, 2017a). Xamarin Forms é a base para a reutilização de código e aceleração do processo de desenvolvimento, permitindo desenvolver uma única interface através da utilização das linguagens C# e XAML, que depois são geradas em linguagem nativa e apresentadas em cada respetiva plataforma.

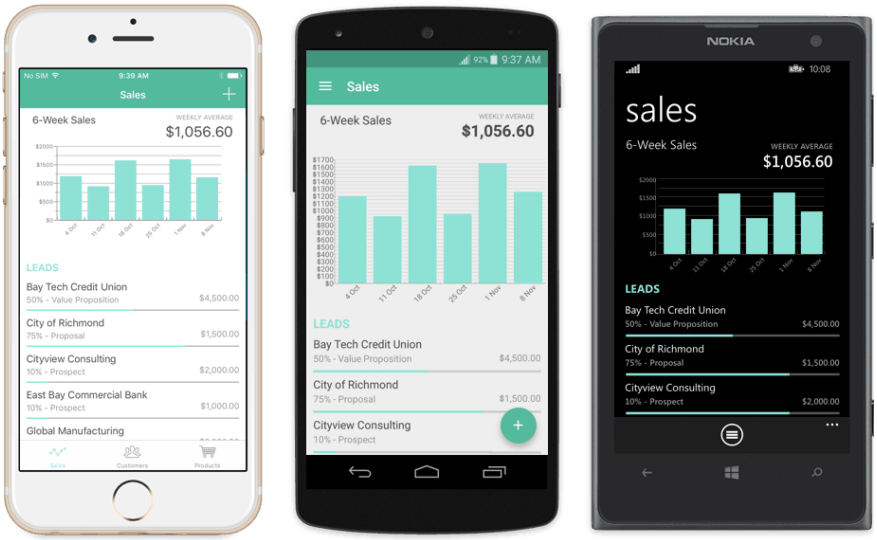


Figura 51 – Exemplo da mesma user interface executada nas 3 plataformas (iOS, Android e Windows) [37].

[37] Imagem retirada de: https://www.xamarin.com/forms

2.6 Conclusão

De acordo com a descrição do problema que representa a motivação desta dissertação e o estudo da análise de valor realizado, podem retirar-se várias conclusões relacionadas com o valor que é gerado com disponibilização da solução Virtual Driver.

Verifica-se que no *roadmap* tecnológico existe espaço para o crescimento destas tecnologias pelo menos até 2030, particularmente para tecnologias que procuram resolver problemas relacionados com fatores Humanos.

Conclui-se uma alta taxa de aceitação de sistemas ADAS segundo uma amostra estatística, no entanto o principal fator para a não proliferação, está relacionada com o preço dos mesmos. Outro fator, é o principio de venda ser a motivação principal dos fabricantes automóveis e OEM's.

Através da extensa investigação do estado da arte, relacionado com os sistemas avançados de apoio ao condutor existentes no mercado, e em particular sistemas de reconhecimento de sinalização de trânsito, pode concluir-se que grande parte destes sistemas estão relacionados com grandes empresas automóveis.

Dos sistemas analisados, verifica-se que apesar da aparente robustez, existe limitação no catalogo de sinalização a reconhecer, sendo apenas comum o reconhecimento de sinalização de velocidade. Destaca-se dentro destas soluções, o poderoso sistema ADAS da empresa *MobileEye*, no entanto o seu alcance só é possível através de aquisição dos automóveis compatíveis (Tesla e BMW). Relativamente a soluções disponíveis em plataformas publicas e comuns, conclui-se que à exceção do Roadly, não existem aplicações que aparentam solucionar o problema desta dissertação.

O estudo do estado da arte, permitiu também investigar um conjunto de artefactos de visão computacional passíveis de utilização no processo de deteção da sinalização como a segmentação baseada em cor e em forma. Já no processo de reconhecimento da sinalização foram estudados dois mecanismos, a comparação de imagens com *templates* (*template matching)* ou pela computação e comparação de característica de uma imagem com as características de um *template* (*feature matching).* Foram elencados também duas metodologias presentes na literatura, uma que atua com base em comparação de características e outra que utiliza máquinas de vetores.

Foram exploradas também as bibliotecas de visão computacional SimpleCV,OpenCV, EmguCV e AForge.Net, e foi possível concluir a total compatibilidade da biblioteca EmguCV para execução multiplataforma.

Para o desenvolvimento da solução foram exploradas as principais plataformas móveis, que poderão ser agregadas através da utilização de o ambiente de desenvolvimento multiplaforma Xamarin, que apresenta uma total compatibilidade com a biblioteca acima EmguCV .

3 Soluções/Abordagens existentes

Após realização do estudo do estado da arte na área dos sistemas avançados de assistência ao condutor, nomeadamente os sistemas de reconhecimento de sinalização vertical, é aqui detalhado que soluções/abordagens que foram usadas no âmbito deste projeto.

3.1 Soluções/Abordagens existentes a adotar

Descrevem-se aqui as soluções identificadas para resolução do problema desta dissertação, tendo como base as suas premissas e a análise do estado da arte realizada.

Pretende-se descrever qual a metodologia de reconhecimento de sinalização de trânsito adotada, os algoritmos, a biblioteca de processamento de imagem e o ambiente de desenvolvimento quer foi utilizado.

A avaliação e comparação das soluções e abordagens existentes são descritas no subcapítulo "3.2-Avaliação de Soluções/Abordagens Existentes".

3.1.1 Metodologia de Reconhecimento de Sinalização de Trânsito

De acordo com a investigação e estudo das abordagens existentes para sistemas de reconhecimento de sinalização vertical, verifica-se que a abordagem *Featuring Matching*, apresenta bastante potencial para responder às necessidades do problema a resolver.

Os resultados e experiências evidenciadas no método *Feature Matching,* descrito na secção 2.5.3.10, são bastante positivos. Os autores defendem uma taxa de sucesso na ordem dos 95%, uma vez que o pré-processamento de cada *frame* demora 80 ms, e a deteção e reconhecimento de sinais de trânsito 100 ms, que perfaz um total aproximado de 0.18 segundos para todo o processo.

A aplicação do *matching* de características a regiões com probabilidade alta permite aumentar significativamente a performance e o tempo total de resposta. Como se pode verificar na Tabela

14, existe uma diferença de 2,806 segundos entre o reconhecimento de características entres as regiões prováveis, e toda a imagem.

TABLE I
EXPERIMENTAL RESULTS

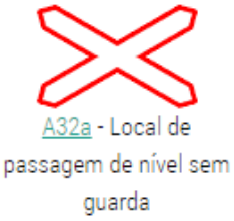
	SIFT on entire image	SIFT on extracted regions
Image Size	1024 × 768	128 × 128
Extraction time	2.827 s	0.079 s
Matching time	0.060 s	0.002 s
Total features	2,812	55
Matched features	18.49	25.29

Tabela 14 – Resultados experimentais do método *Feature Matching* proposto em (Ren *et al.*, 2009)

Aliado aos indicadores de performance, o grau de complexidade do método *Feature Matching* não aparenta comprometer a sua execução em dispositivos móveis, devido à sua experiência ter sido realizada numa máquina com características bastantes limitadas "(Intel Duo Core, 2.4 GHz, 2GB memory, Linux"(Ren *et al.*, 2009). No entanto, só com a realização de testes se conseguirá provar sua eficácia e eficiência no cenário desta dissertação.

Durante o estudo desta abordagem foram detetados possíveis pontos de melhoria que poderão alavancar a adoção deste método no desenvolvimento da solução *Virtual Driver*.

- No cenário da sinalização Portuguesa existem sinais com formas incomuns, a que a extração de formas sugerida não responderia, como por exemplo, o sinal de perigo "Local de passagem de nível" (Figura 52). Este tipo de exceções deverá ser previsto no desenvolvimento da solução.



A32a - Local de passagem de nível sem guarda

Figura 52 – Sinal de perigo A32a (Portugal, no date)

- No processo de deteção, extração e comparação de características, poderá existir um *overhead* no momento da comparação dos candidatos com todos os *templates*. Nesse momento, poderiam ser mapeadas as características das regiões identificadas como a cor e a forma, com a lista de *templates* possíveis, já que:
    - Seria uma mais-valia neste processo serem pré-calculadas as características para que no momento do *match* essa informação já estivesse indexada;
    - Com a implementação desta abordagem, seria necessário verificar qual a significância do fenómeno de oclusão da sinalização;

3.1.2 Algoritmos Feature Matching

No âmbito desta dissertação, pretende-se verificar a aplicabilidade de alguns algoritmos abordados em "2.5.3.8-Feature Matching" nos processos de deteção, extração e comparação de características, evidenciando aqueles que melhores resultados trarão a esta metodologia.

Pretende-se a escolha da solução que permita a execução sobre diversos algoritmos, evidenciando os melhores resultados. Devem ser tidos em conta os seguintes conceitos:

- Tempo de resposta de processamento de imagem em tempo real;
- Qualidade e eficácia de reconhecimento (minimização de falsos positivos).

A escolha da utilização desses algoritmos é realizada com base nos resultados da avaliação obtida, detalhados no subcapítulo seguinte, de avaliação.

3.1.3 Biblioteca de processamento de imagem

Através do estudo das bibliotecas para desenvolvimento de soluções de visão computacional, verifica-se que a biblioteca EmguCV apresenta-se como a mais indicada para o desenvolvimento da componente de deteção e reconhecimento de sinalização de trânsito vertical. Destacam-se algumas das vantagens que vão de encontro às necessidades do projeto:

- Desenvolvimento em linguagem de alto nível C# - Permite acelerar o processo de desenvolvimento da solução, pela familiaridade dos intervenientes com a linguagem;
- Compilação e execução multiplataforma compatível com o projeto *Mono* – Cumpre com o requisito multiplataforma gerado na definição de ideias de negócio;
- Disponibilização e compatibilidade com um conjunto de algoritmos de *feature detection, feature extraction* e *feature matching*;
- Compatibilidade com ambiente de desenvolvimento multiplataforma através do *Xamarin SDK;*
- Vasta comunidade, e atualização regular da biblioteca.

3.1.4 Ambiente de desenvolvimento

De acordo com a revisão bibliográfica evolutiva das soluções tecnológicas existentes, verificou-se que existem inúmeros ambientes de desenvolvimento multiplataforma que permite o desenvolvimento em diferentes linguagens de programação.

No entanto elegeu-se o *Xamarin SDK* como o ambiente de desenvolvimento a adotar devido a diversos fatores preponderantes:

- Linguagem C#;
- Capacidade de desenvolvimento na *interface* nativa de cada dispositivo;
- Compatibilidade com a *framework EmguCV* e bibliotecas nativas do .NET, executáveis em *iOS, Android e Windows Phone*;
- Licença *open source;*
- Compilação e execução através do projeto *Mono.*

3.2 Avaliação de Soluções/Abordagens Existentes

A avaliação das soluções existentes foi realizada através das evidências presentes na literatura. Para as metodologias de reconhecimento de sinalização de trânsito, foi fundamentalmente avaliada, a grandeza relacionada com a eficácia do método.

As soluções tecnológicas são avaliadas de acordo com a comparação entre diferentes variáveis, como o licenciamento, documentação, facilidade de utilização, performance e outras.

3.2.1 Avaliação Metodologias de Reconhecimento de Sinalização de Trânsito

Comparando as duas abordagens de reconhecimento de sinalização de trânsito vertical, a Abordagem SVM e Abordagem *Features Matching* (2.5.3.10 Abordagens Traffic Sign Recognition), verifica-se uma maior taxa de sucesso na abordagem de *Features Matching (*93.24% vs 95% respetivamente.

Destaca-se a eficácia de 95%, demonstrada pela abordagem *Features Matching,* bem como, o menor grau de complexidade do método, capaz de ser executada por um processador comum de um dispositivo móvel, razões pelas quais foi selecionado para implementação.

3.2.2 Avaliação Algoritmos Feature Matching

Com base na revisão da literatura relacionada com algoritmos de deteção e descrição de características de uma imagem, abordados em "2.5.3.8-Feature Matching", nesta secção serão evidenciados alguns resultados comparativos, nela incluídos.

Analisando os principais processos de deteção de características, verifica-se, no estudo apresentado em *"Evaluation of Local Detectors and Descriptors for Fast Feature Matching"* (Miksik and Mikolajczyk, 2012), que submete um conjunto de 100 imagens com origem nos repositórios *Graffiti* e *Pascal VOC*, que o algoritmo FAST apresenta-se como o mais rápido, e o que deteta mais pontos de interesse, seguido pelos algoritmos ORB e BRISK (Tabela 15).

Detector	Run time [ms.]	Speed-up [-]	# keypoints
SURF	176	1.9	2 911
DoG	338	1.0	1 552
FAST	**2**	169.0	**5 158**
STAR	17	19.9	849
MSER	60	5.6	483
BRISK	**10**	33.8	1 874
ORB	**7**	48.3	594

Tabela 15 – Tempos de resposta médios para deteção de *keypoints* (Miksik and Mikolajczyk, 2012)

Relativamente aos tempos de descrição, verifica-se que BRIEF, ORB e BRISK apresentam o melhor tempo médio de execução (Tabela 16).

Descriptor	Run time [ms.]	Speed-up [-]
SURF	117.1	3.83
SIFT	448.6	1.00
BRIEF	**3.8**	118.05
BRISK	**10.6**	42.32
ORB	**4.2**	106.80
LIOP	1 801.1	0.25
MROGH	2 976.8	0.15
MRRID	5 625.1	0.08

Tabela 16 – Tempos médios de execução para os algoritmos descritores.

De acordo com um estudo comparativo realizado sobre alguns algoritmos de deteção e extração de *features*, *"A Performance Evaluation of Detectors and Descriptors for UAV Visual Tracking"* (Cowan *et al.*, 2016), submeteram-se todas as conjunções possíveis de detetores e descritores, ao processo de deteção e reconhecimento, para um conjunto de 48 imagens que sofreram transformações. Foi obtido um número médio de pontos de interesse detetados e computados (Tabela 17) e, adicionalmente, tempos por imagem. O resultado combinado é descrito na Tabela 18, dele pode concluir-se:

- Os algoritmos SIFT e SURF são bastante mais lentos que os algoritmos BRISK e ORB;
- O algoritmo SURF é cerca de 30% mais rápido do que o SIFT;
- O algoritmo ORB extrai menos *keypoints* que os outros algoritmos.

	Detector										
Descriptor	Agast	AKAZE	BRISK	Fast	GFTT	KAZE	MSER	ORB	SIFT	SURF	Star
AKAZE		3087				2975					
BRISK	15 168	3087	8051	15 026	947	2796	932	417	5224	4729	915
KAZE		3087				2975					
ORB	13 818	3075	7803	13 717	874	2612	994	496		5002	915
BRIEF	14 034	3087	7873	13 928	885	2649	1011	496	4879	5070	915
DAISY	15 997	3087	8051	15 823	1000	2975	1220	496	5477	5363	915
FREAK	14 375	3087	6987	14 258	903	2635	804	245	4965	3755	908
LATCH	14 107	3087	7894	14 000	889	2662	1019	496	4903	5092	915
SIFT	15 997	3087	8051	15 823	1000	2975	1220	496	5477	5363	915
SURF	15 997	3087	8051	15 823	1000	2975	1220	496	5477	5363	915

Tabela 17 – Média de pontos de interesse detetados e computados por imagem (Cowan *et al.*, 2016)

Algorithm	Total time/ms	Total keypoints	Time per keypoint/µs
AKAZE	9137.58	148 192	80.0792
BRISK	7733.16	386 463	21.7529
KAZE	34 649.7	142 800	338.398
ORB	1000.2	23 830	42.085
SIFT	26 738.6	262 872	140.111
SURF	24 343.0	257 441	98.299

Tabela 18 – Resultado combinado para todas as imagens, relacionando tempos de resposta (Cowan *et al.*, 2016)

3.2.2.1 Conclusão

Após revisão de alguns resultados comparativos, relacionados com os algoritmos de deteção e descrição, verifica-se que os algoritmos de representação binária têm melhores tempos de resposta. Destacam-se, no processo de deteção, os algoritmos FAST, ORB e BRISK, e no processo de descrição os algoritmos ORB, BRIEF e BRISK, como sendo os de execução mais rápida.

Em relação ao algoritmo FREAK, apesar dos autores defenderem um tempo de processamento mais baixo do que SIFT, SURF e até BRISK (Tabela 19), ao analisar os estudos comparativos, não foi possível chegar-se a essa conclusão. Pretende-se, como trabalho futuro, verificar a aplicabilidade deste método.

Time per keypoint	SIFT	SURF	BRISK	FREAK
Description in [ms]	2.5	1.4	0.031	0.018
Matching time in [ns]	1014	566	36	25

Tabela 19 – Tempos de descrição e matching apresentados pelos autores de FREAK (Alahi, Ortiz and Vandergheynst, 2012)

Os algoritmos SURF e SIFT, apesar de se apresentarem como bastante fiáveis, desempenham tempos mais altos do que aquilo que se pretende para uma solução em *real time.* Para além disso, os dois algoritmos são patenteados, o que acresce o custo final da solução.

Com base na premissa de que a solução Virtual Driver tem de ser capaz de atuar em tempo real, a taxa de resposta passa a ser fator preponderante para a escolha deste conjunto de algoritmos. Pretende-se que a solução permite a inclusão de diferentes algoritmos, permitindo uma comparação efetiva de resultados.

3.2.3 Avaliação Frameworks Computer Vision

A *framework SimpleCV* demonstrou, como uma das principais vantagens, a simplicidade de desenvolvimento. No entanto, foi descartada, pelo facto de não revelar capacidade para execução em plataforma móvel.

De acordo com a comparação das *frameworks OpenCV, EmguCV e AForge* realizada em *Emgu CV Essentials* (Shi, 2013), a EmguCV apresenta-se como a melhor *framework* para processamento de imagem (Tabela 20), tendo em conta a comparação de diferentes fatores como o licenciamento, a documentação e material de apoio, a facilidade de utilização e a performance Tabela 21).

Code	Library	Grayscale (ms)	Binarization (ms)	Rank
C	OpenCV	9.3432	6.9332	1
C#	AForge.NET	32.6548	19.8743	5
C#	Emgu CV	11.2369	9.6853	3
C#	OpenCV (P/Invoke)	10.2355	8.0332	2
C#	Custom method	33.6742	17.2009	4

Tabela 20 – Resultado da comparação da performance das *frameworks*. Foram executadas duas operações sobre imagem, o processamento para escala de cinzas e a conversão para pretos e brancos "*binarization"*(Shi, 2013).

Library	OpenCV	Emgu CV	AForge.NET
License	BSD	GPLv3 or Commercial	LGPLv3
Documentation	Great	Good	Poor
Ease of use	Poor	Good	Great
Performance	Awesome	Great	Very poor

Tabela 21 – Resultado da comparação para cada fator (Shi, 2013).

Para além dos resultados acima mencionados, a Emgu CV é capaz de atuar sobre OpenCL, permitindo implementar tempos de resposta muito baixos, através da utilização de processamento paralelo por CPU e GPU.

Relativamente ao licenciamento, o EmguCV é adquirível através de uma licença comercial vitalícia, para o desenvolvimento com dispositivos móveis.

Verifica-se como uma das principais vantagens do EmguCV, a compatibilidade com os principais algoritmos de deteção e computação de características, sendo o laboratório *OpenCV*, propulsionador de alguns deles (ORB[38]).

3.3 Conclusão

Com base nas abordagens investigadas foi realizado um estudo comparativo de forma a determinar a adaptabilidade das mesmas no desenvolvimento da solução Virtual Driver.

No que diz respeito às metodologias de reconhecimento de sinalização de trânsito estudadas, conclui-se que a metodologia baseada em comparação de características apresenta uma maior

[38] Informação retirada de: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_orb/py_orb.html

taxa de sucesso e por isso foi escolhida como base para o desenvolvimento do processo TSR da solução Virtual Driver.

Através da comparação dos tempos de deteção e computação dos algoritmos detetores e extratores investigados, conclui-se que o algoritmo *ORB – Oriented FAST and Rotated BRIEF* apresenta o melhor rácio entre o tempo e a percentagem de características detetadas e descritas e desta forma foi escolhido como o algoritmo principal a adotar. A solução desenvolvida prevê a avaliação da utilização dos outros algoritmos.

Como biblioteca de visão computacional, foi escolhida a EmguCV por apresentar uma compatibilidade total com o desenvolvimento para multiplataforma e por apresentar os melhores indicadores segundo a documentação, material de apoio, a facilidade de utilização e a performance.

4 Design

Neste capítulo será incluída a descrição da solução de Design construída para o Virtual Driver, nas vertentes de análise de requisitos funcionais e não funcionais, interface com o utilizador, arquitetura, metodologia de reconhecimento de sinalização de trânsito e estrutura de base de dados a implementar. Serão também apresentadas alternativas ao *Design* da solução e as respetivas comparações.

4.1 Análise de Requisitos

Nesta secção são apresentados o(s) ator(es) e partes interessadas do sistema, bem como, os requisitos funcionais e não funcionais aos quais o Virtual Driver deverá dar resposta.

4.1.1 Atores do sistema

No âmbito deste projeto, foram identificados os seguintes atores e partes interessadas:

- **Condutor de veículo**: qualquer individuo que tenha habilitação para a condução automóvel e que poderá instalar a aplicação no seu dispositivo móvel. A sua interação com o sistema permitirá configurar o sistema *ADAS* e ativar/desativar o reconhecimento;

- **Gestor de conteúdos**: para futura evolução da solução, pretende-se que exista possibilidade de configurar centralmente a distribuição de conteúdos para o Virtual Driver através de um Portal. Contempla-se ações de *upload* de novas sinalizações para computação de características, e possível disponibilização para descarregamento.

De forma a garantir a evolução da solução, estão previstos estes dois atores, que são intervenientes no sistema como um todo, englobando a aplicação móvel e um Portal central.

4.1.2 Requisitos Funcionais

Após levantamento e análise de requisitos inerentes à disponibilização da solução Virtual Driver, definiu-se o seguinte conjunto de requisitos funcionais:

- A aplicação móvel deverá permitir a definição de um conjunto de configurações base à ativação do processo de reconhecimento. As configurações deverão incidir em:
    - Alterar Resolução de Captura;
    - Ativar/Desativar alertas sonorous;
    - Temporizar de sinalização;
    - Alteração do método de reconhecimento.
- A aplicação deverá permitir ativar o reconhecimento em tempo real do cenário, e alertar o utilizador através de Áudio, Imagem e Texto.

4.1.3 Casos de Uso

Nesta secção estão elencados os principais diagramas de caso de uso, que permitem observar a interação entre os atores e o sistema Virtual Driver.

O sistema Virtual Driver é composto por dois subsistemas, a aplicação móvel e o portal de gestão de conteúdos, dessa forma, são descritos na Figura 53 e Figura 54, os respetivos diagramas de caso de uso de nível zero.

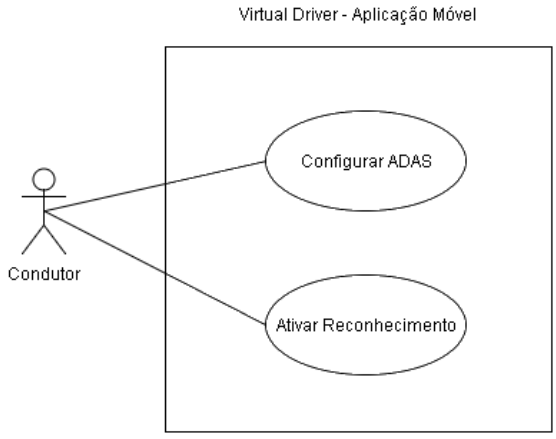


Figura 53 – Diagrama Casos de Uso Nível 0 para a aplicação móvel

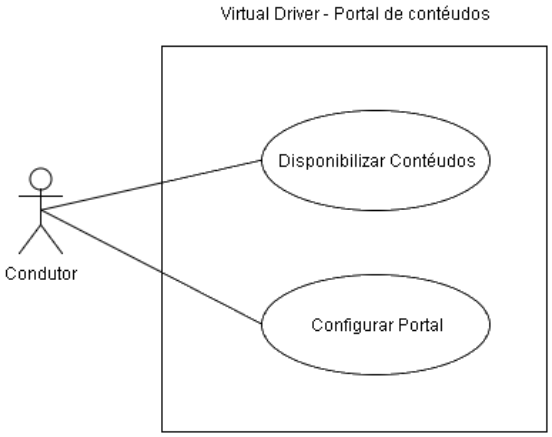


Figura 54 – Diagrama de Casos de Uso Nível 1 para o portal de conteúdos

Por forma a esclarecer a ilustração dos casos de uso do sistema, será apresentado em formato breve, o fluxo dos mesmos. Apesar de não existir ainda informação suficiente para ser realizada uma análise de pormenor ao subsistema Portal de Conteúdos, são detalhados em linhas gerais os respetivos casos de uso.

Virtual Driver - Aplicação móvel

Caso de uso: UC1

Nome: Ativar Reconhecimento.

Atores: Condutor.

Descrição (fluxo principal): O utilizador acede à aplicação móvel e requisita o início do reconhecimento avançado de apoio ao condutor, com base nas configurações que estão carregadas inicialmente. O sistema, de seguida, apresenta ao utilizador a informação em tempo real, *frames* do cenário capturado, sinalização/elementos reconhecidos e alertas em forma de Áudio, Imagem e Texto.

Caso de uso: UC2

Nome: Configurar ADAS.

Atores: Condutor.

Descrição (fluxo principal): O utilizador, através de uma lista de opções disponibilizada pelo sistema, poderá alterar a configuração base para o motor de reconhecimento. O sistema apresentará o sucesso dessa alteração.

Virtual Driver – Portal Conteúdos

Caso de uso: UC3

Nome: Disponibilizar Conteúdos.

Atores: Gestor de Conteúdos.

Descrição (fluxo principal): O utilizador acede ao portal de conteúdos e carrega uma série de novos elemento, tais como, novas sinalizações, imagens e meta dados. O sistema deverá processar esses conteúdos e indicar o sucesso dessa operação.

Caso de uso: UC4

Nome: Configurar Portal

Atores: Gestor de Conteúdos

Descrição (fluxo principal): O utilizador, através do acesso ao Portal de Conteúdos, configura uma série de elementos que têm influencia direta no funcionamento da aplicação móvel. O sistema apresentará a opção de tornar público um conjunto de sinais de trânsito pré-processados.

4.1.4 Requisitos Não Funcionais

Os Requisitos Não Funcionais do sistema foram classificados em cinco tipos chave, que influenciam diretamente a arquitetura da solução. Os mesmos, são definidos através de Requisitos de Interface, Usabilidade, Desempenho, Portabilidade e Manutenção.

Requisitos de Interface

Deverão existir três áreas distintas de apresentação da informação ao utilizador: área para a sinalização reconhecida, área para informação textual associada à sinalização e uma área predominante com a pré-visualização do cenário em captura.

De forma a tirar o máximo partido da resolução da câmara do equipamento vs disposição da estrada, a aplicação deverá estar preparada para ser visualizada em formato horizontal.

Requisitos de Usabilidade

O sistema deverá ser responsivo e adaptar-se à resolução disponível em cada dispositivo. Pretende-se que a ativação do reconhecimento seja realizada de forma rápida e com menos interações possíveis com o sistema.

O utilizador deverá ser autónomo na configuração de alguns parâmetros que afetam diretamente o motor de reconhecimento.

Requisitos de Desempenhos

Tendo em conta que o sistema de reconhecimento atua em tempo real, e o seu resultado é crítico, deverá ser assegurado um tempo de resposta muito baixo para as configurações *standard*, considerando-se que o cenário ideal, é que consiga acompanhar o mais próximo

possível a cadência de captura da câmara do dispositivo. Ou seja, caso exista uma taxa de captura de 20 FPS, o sistema deverá conseguir processar uma boa parte desses *frames*.

Devido à limitada e reduzida capacidade de execução em alguns equipamentos, o sistema deverá ser adaptável o suficiente para conseguir executar, com poucos recursos e com baixo custo de desempenho (processador vs bateria).

Requisitos de Portabilidade

A aplicação móvel Virtual Driver deverá ser executável nas plataformas *Android, Universal Windows Platform, e iOS*.

Requisitos de Manutenção

A evolução e manutenção da aplicação deverão ser garantida através de atualizações periódicas com novas sinalizações compatíveis de reconhecimento e melhorias de processo.

O desenvolvimento do sistema deverá prever a inclusão de novos módulos ADAS ao longo do tempo.

4.2 Solução proposta

Este subcapítulo descreve o modo de funcionamento da solução proposta, e a forma como foi tecnologicamente pensada.

4.2.1 Modo de Funcionamento

O modo de funcionamento da solução proposta permitirá que a mesma seja instalada num dispositivo móvel como um *smartphone* , *tablet* ou outro micro computador, a ser utilizado durante a condução automóvel (Figura 55).

Através do usufruto da câmara do dispositivo, o sistema é capaz de detetar em tempo real a presença de sinais de trânsito ao longo do percurso. Após identificação do sinal, a aplicação representa graficamente, no ecrã, o sinal ou sinais detetados, alertando de imediato o condutor através de Imagem, Áudio e Texto. A informação áudio é transmitida através das colunas do dispositivo.

São capturadas imagens a uma determinada cadência (*frames por segundo).* Cada *frame* é submetida a um processo de processamento de imagem de forma a identificar possíveis sinais de trânsito.

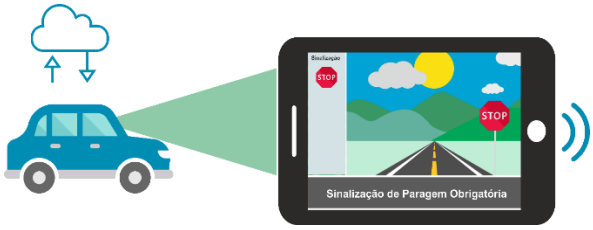


Figura 55 – *Virtual Driver* em modo de funcionamento

Com vista a uma melhoria contínua, a aplicação poderá recolher dados estatísticos e características de captura de forma a conseguir aumentar a sua taxa de reconhecimento, e aumentar o catálogo reconhecível.

4.3 Alternativas à solução proposta

Neste subcapítulo serão detalhadas as alternativas ao tipo de sistema descrito anteriormente, nomeadamente alternativas para o ambiente de execução da solução.

Apesar de o sistema ser desenhado para execução em dispositivos comuns como o *smartphone e tablet,* existem outras possibilidades de implementação.

Devido à utilização de tecnologias multiplataforma no processo de desenvolvimento, a execução do sistema é possível numa vasta lista de dispositivos, que suportem sistemas operativos *Android, IOS e Universal Windows Platform.*

4.3.1 Alternativa 1 – Raspberry PI

Na construção da solução, poderiam ter sido utilizados microcomputadores, como por exemplo o Raspberry Pi (Figura 56) para a execução da aplicação, já que os mesmos conseguem executar sistemas operativos Linux e por consequente Android.

Estes dispositivos não contêm, por defeito, ecrã, câmara, microfone, nem colunas incorporadas. Uma abordagem possível seria a de utilização de um ecrã externo ligado diretamente ao dispositivo, no entanto, esta alternativa poderia levar a que fosse descartada a informação visual.

Figura 56 – Raspberry Pi 3[39]

O áudio poderia ser emitido pelos próprios altifalantes do automóvel, com a ligação entre o microcomputador e os altifalantes do automóvel. A integração do Raspberry Pi deveria ser realizada por uma placa de som compatível com a voltagem do automóvel, como por exemplo, a X400 Expansion Board (Figura 57).

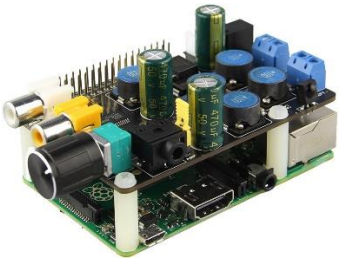


Figura 57 – X400 Expansion Board para integração com automóvel no Raspberry Pi 3[40]

Seria ainda necessário instalar uma câmara externa no microcomputador para o processo de captura, e um microfone para receção de comandos de voz.

4.3.2 Alternativa 2 – Dispositivos Pre-Builded

Atualmente existem produtos desenvolvidos que incorporam microcomputadores e outros componentes, que permitem ter uma solução completa a nível de *hardware*.

Um desses exemplos é iCarus, um dispositivo que incorpora um Raspberry Pi (A, B, A+, B+ ou 2) e que permite executar o sistema operativo Linux (Figura 58). Inclui também uma ligação com um ecrã *touch* e uma placa de ICR, capaz de gerir o processamento de áudio e realizar as ligações com os altifalantes e com a alimentação do automóvel.

Este dispositivo na versão mais completa, tem a vantagem de apresentar uma camara de alta resolução com aceleração de *hardware*.

[39] Imagem retirada de https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[40] Imagem retirada de http://www.suptronics.com/Xseries/x400.html

Figura 58 – Dispositivo iCarus[41]

A instalação do Virtual Driver neste dispositivo obrigaria a uma extensão do sistema ou até a mesmo à instalação do S.O Android.

Já o Auto Pi, é um *dongle* (pequeno dispositivo externo inserível em portas de comunicação) construído a partir do Raspberry Pi Zero, capaz de ser ligado diretamente ao interface OBD-II de um automóvel, permitindo alimentar-se automaticamente e interagir com algumas métricas de diagnóstico do automóvel. O Auto Pi conjuga as seguintes características:

- Processador do Raspberry Pi Zero capaz de correr S.O. Linux;
- Modem 3G/4G;
- Módulo GPS;
- OBD Chipset – Capaz de realizar comunicação com veículos com OBD II;
- Acelerómetro;
- Altifalante incorporado;
- Conectividade através de USB, HDMI,GPIO;
- Wifi e Bluetooth.

Considerando a implementação do Virtual Driver, o dispositivo Auto Pi permitiria a inclusão de um ecrã, camara e microfone , uma vez que o seu sistema permite a extensão a outros módulos de *hardware* (Figura 59).

[41] Imagem retirada de: http://i-carus.com/

Figura 59 – Dispositivo Auto Pi

O Auto Pi vem compilado com a versão Raspbian, que é uma versão minimalista do Debian Linux para Raspberry PI, com a instalação do software base do Auto PI.

Considerando a flexibilidade apresentada pelo produto (Auto Pi, 2017), seria necessário desenvolver um componente, para o Virtual Driver, capaz de executar sobre esta plataforma, ou então instalar o sistema operativo Android diretamente no dispositivo.

4.3.3 Alternativa 3 – Interligação com sistemas Auto

Apesar de todas as restrições tecnológicas que os veículos mais datados apresentam, uma das soluções para execução da solução proposta, poderia ser a utilização dos recursos de cada veículo.

Recentemente foi desenvolvida pela Google uma aplicação, denominada Android Auto, capaz de ser executada diretamente no ecrã dos automóveis que a suportam, ou apenas no *smartphone* (será sempre necessária a utilização de um *smartphone* Android)(Figura 60) . Este ambiente permite executar algumas das aplicações presentes no S.O. do *smartphone* como Maps, Comunicações, Leitores de Musica entre outros. Para além do comodismo da integração com o display do automóvel, as linhas de orientação Android Auto pretendem garantir que a interação entre o utilizador e o sistema seja simples, segura e não retire a concentração do condutor da condução, usando para isso comandos de voz, regras de contrastes de cor e outros artefactos. Qualquer aplicação que se pretenda incluir sobre a plataforma Android Auto, deverá responder a um conjunto de *guidelines* de qualidade (Google, 2017c).

A utilização do Virtual Driver nesta plataforma permitiria conferir-lhe um maior conforto de utilização, ainda que fosse necessário estender algumas das funcionalidades da solução para cumprir com os requisitos do Android Auto.

Figura 60 – Apresentação do Android Auto [42]

À semelhança do que existe para o sistema operativo Android, no caso do IOS, existe o sistema CarPlay (Figura 61).

Este sistema permite a utilização do iPhone de uma forma integrada com o ecrã e controlos do automóvel, através do uso de controlos de voz (Siri) e dos botões do automóvel, para garantia de uma interação cómoda e segura para o condutor.

A aplicabilidade deste sistema está restrita a um conjunto de marcas e modelos de automóveis compatíveis.

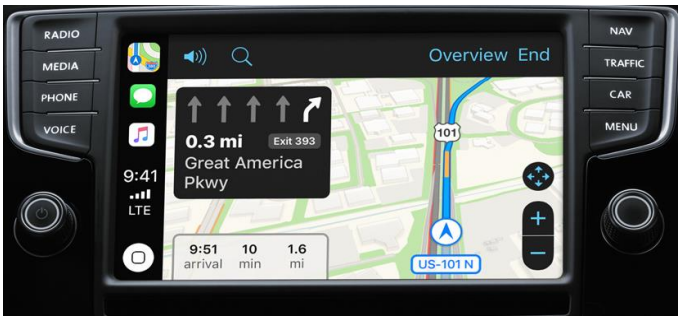


Figura 61 – Apresentação do CarPlay

A possibilidade da inclusão do Virtual Driver nesta plataforma estaria comprometida ao de algumas regras e linhas de orientação no desenvolvimento da solução.

4.3.4 Comparação de Alternativas

A utilização de dispositivos que atualmente já estão presentes no quotidiano dos cidadãos é uma vantagem para a implementação do *design* inicial descrito.

[42] Imagem retirada de : https://www.android.com/auto/

Na maior parte dos casos, em que o utilizador já possui um *smartphone/tablet*, não haverá custos adicionais na aquisição de novos equipamentos. No entanto, as alternativas descritas acima, podem dar resposta.

As alternativas propostas foram sujeitas a uma análise comparativa segundo alguns critérios (Tabela 22 – Análise comparativa das alternativas) relacionados com a acessibilidade em obter o dispositivo, custos com a aquisição do mesmo, dependência de outros dispositivos, requisitos tecnológicos, dificuldade de instalação e uma analogia do custo com o desenvolvimento da solução Virtual Driver proposta.

Concluiu-se que as opções relacionadas com sistemas integrados no automóvel, como Android Auto e Car Play, têm bastante potencial para garantir o conforto do utilizador e segurança da interação, mas o facto de necessitarem igualmente de um *smartphone*, e de só serem compatíveis com uma série de veículos, ruma contra o propósito desta tese: conseguir atingir o maior número cidadãos.

Das três soluções com base no microcomputador Raspberry Pi, a mais promissora é o Auto Pi, já que, a capacidade de evolução do produto, e a alargada possibilidade de integração de novos componentes, fazem da integração de um sistema de reconhecimento de trânsito uma mais-valia de foco pela detentora do produto. A sua instalação *plug&play* apresenta-se como uma vantagem competitiva, no entanto acaba por restringir a utilização apenas a automóveis compatíveis com OBD II.

Entre as alternativas Raspberry Pi e iCarus, verifica-se que o iCarus poderá ser a solução mais rápida e fácil de instalar para o utilizador final, no entanto o seu custo é em muito superior ao do Raspberry Pi. Ambas não dependem de requisitos específicos do automóvel para serem executados. Desta forma, acredita-se que o Raspberry Pi, com a adição de outros componentes, será a alternativa que poderá atingir o maior número de utilizadores por um custo mais baixo.

Critério	Raspberry Pi	iCarus	Auto Pi	Android Auto	Car Play
Alcançável pelo utilizador comum?	Sim	Sim	Sim	Sim	Sim
Custo Hardware	Baixo	Médio	Possivelmente baixo (Desconhecido valor exato)	Nenhum	Nenhum
Dependência de outros dispositivos?	Não	Não	Não	Sim, necessário *smartphone* Android	Sim, necessário *iPhone*
Requisitos Tecnológicos	Qualquer automóvel	Qualquer automóvel	Carros com conectividade OBD II	Apenas automóveis compatíveis	Apenas automóveis compatíveis

Dificuldade de instalação	Difícil	Média	Muito fácil	Fácil	Fácil
Custo Desenvolvimento Virtual Driver comparativamente com solução proposta	Semelhante	Semelhante	Maior	Maior	Maior

Tabela 22 – Análise comparativa das alternativas

4.4 Interface com o utilizador

Descrever-se-á aqui, a forma de interação do utilizador com o sistema, bem como, o fluxo de interações entre cada um dos ecrãs.

4.4.1 Interação Pessoa-Máquina

A interação entre o utilizador e o Virtual Driver deverá ser minimalista, eficiente e eficaz. A concentração do utilizador no ato de condução não deverá ser prejudicada com a interação com o sistema. Caso isso aconteça, o propósito e a missão desta solução deixariam de fazer sentido (pretende-se que o sistema tenha um papel ativo na prevenção da sinistralidade rodoviária, não que a impulsione).

Assumindo os princípios básicos de um sistema interativo e baseado no *design* centrado no utilizador, pretende-se a implementação da interface com o utilizador através de 4 ecrãs (Figura 62). A disposição dos mesmos deverá ser realizada num plano horizontal (*landscape)* de forma a retirar máximo partido da captura horizontal do plano da estrada.

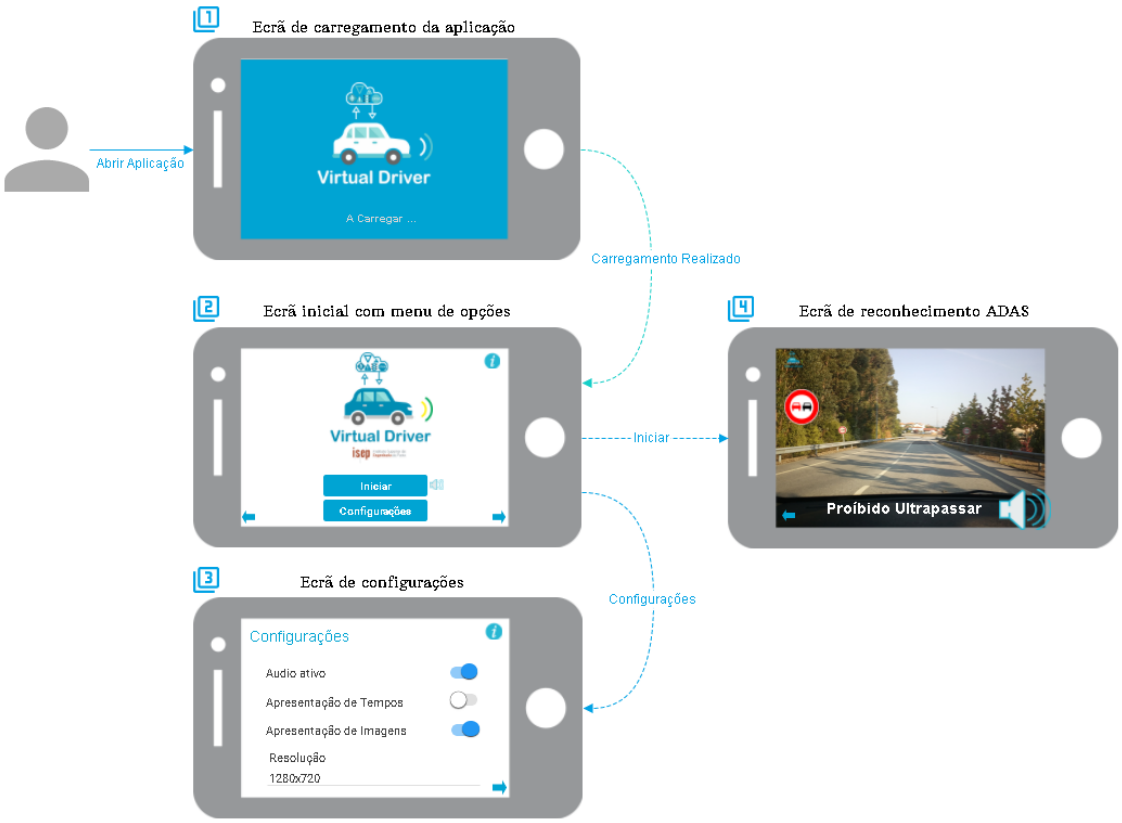


Figura 62 - Ecrãs da interface com o utilizador

Descrevendo cada ecrã da aplicação:

1. **Ecrã de carregamento da aplicação:** Pretende-se que neste ecrã se realize o primeiro contacto do utilizador com a aplicação. Esta deverá, de forma simples, indicar ao utilizador que está a ser realizado um carregamento inicial da solução. Este é o momento em que o sistema descarrega centralmente configurações e sinalizações a reconhecer;

2. **Ecrã de menu de opções:** Representa o ecrã pivot da aplicação, onde é possibilitada a navegação para outros ecrãs. Pretende-se que a interação com este ecrã seja realizada através do toque no visor *Touch* do equipamento, ou através da interação por comandos de voz. As opções disponíveis permitirão avançar para o ecrã de reconhecimento "Iniciar" e para a opção "Configurações";

3. **Ecrã de configurações:** Permite definir um conjunto de configurações base para o motor de reconhecimento. As configurações são transversais à aplicação e permanentes. A qualquer momento, o utilizador poderá voltar para o ecrã inicial de opções;

4. **Ecrã de reconhecimento ADAS:** Este ecrã apresentará em tempo real o resultado do reconhecimento ADAS. É apresentada a pré-visualização da captura realizada, e a

informação sobre a sinalização reconhecida em formato de Áudio, Imagem e Texto. Não está prevista a existência de possíveis ações do utilizador neste ecrã, ou seja, apenas servirá o propósito de leitura da sinalização de trânsito. A qualquer momento, o utilizador poderá voltar para o ecrã inicial de opções.

A descrição do fluxo de interação entre o sistema e o utilizador é ilustrado no diagrama de fluxo de dados presente na Figura 63.

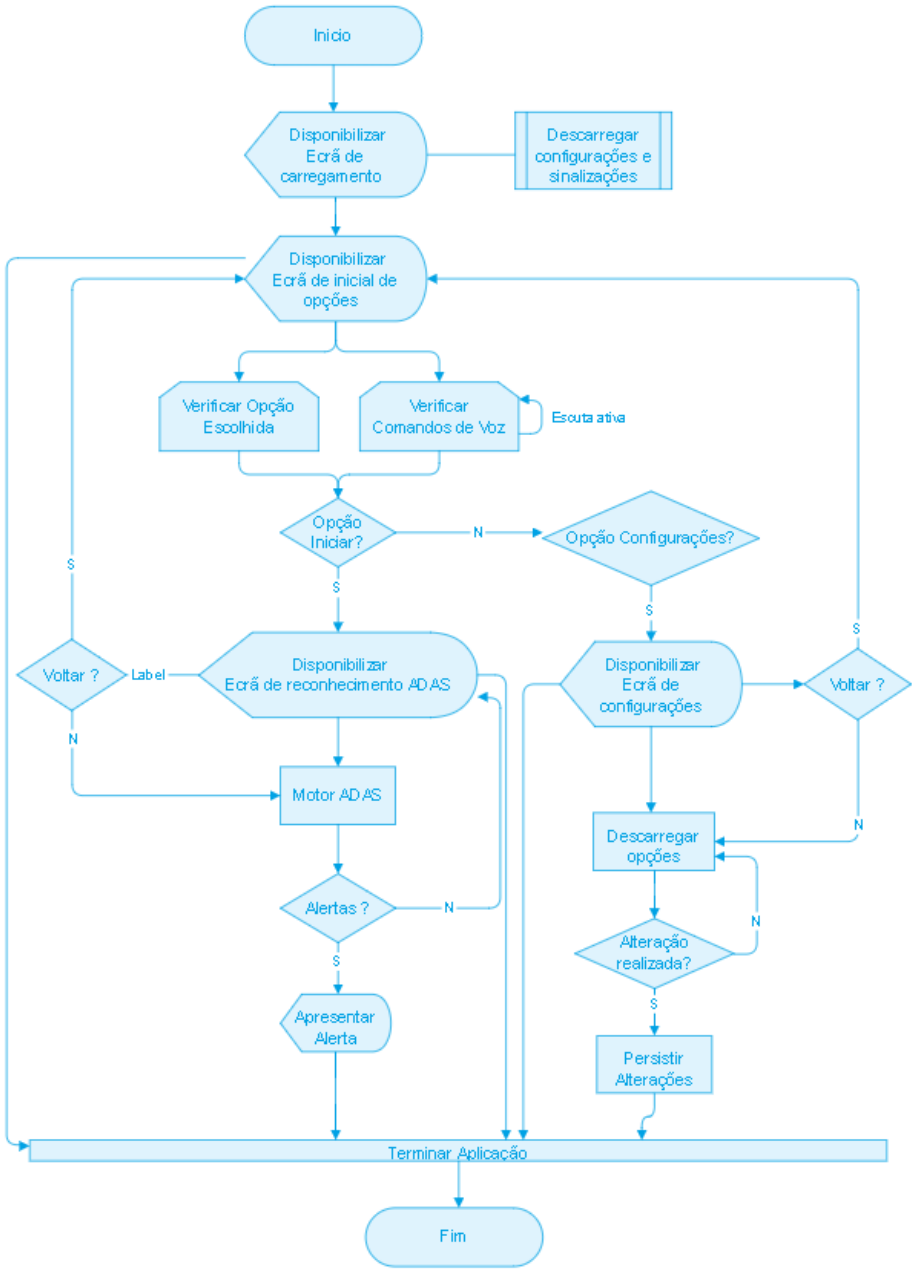


Figura 63 - Diagrama de Fluxo de Dados, Interação utilizador-sistema

4.5 Arquitetura

De forma a conceber a arquitetura lógica do sistema, foi adotada uma organização em camadas.

A solução a desenvolver será dividida em quatro camadas, de forma a garantir extensibilidade e flexibilidade no desenvolvimento. Cada camada é independente e poderá ser entendida como um subsistema, que poderá ser invocada por outros. As camadas são *Interface, ADAS Business Logic, ADAS Modules, Data Access* e *DataBase* (Figura 64). As mesmas interligam-se através de troca de informação, que será concretizada na disponibilização de resultados ao utilizador.

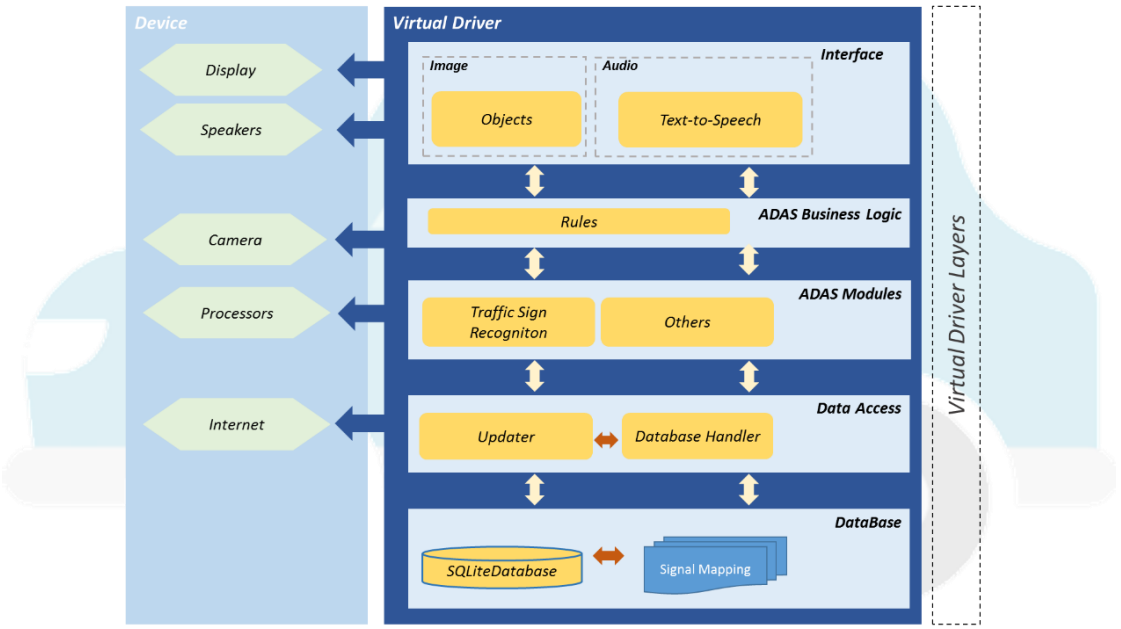


Figura 64 – Representação da arquitetura em camadas

Cada camada poderá aceder a diferentes recursos do dispositivo em que está a executar. A camada *DataBase* permitirá desenvolver uma estrutura de dados em que deverá ser persistido o mapeamento de sinalização vertical entre cores, formas e simbologias, bem como as características que serão usadas nos algoritmos da camada *ADAS Modules.* O modelo adotado poderá ser um modelo relacional, através da utilização do gestor de base de dados *SQLiteDatabase.*

A camada *Data Access* consiste na disponibilização de um conjunto de operações de mais alto nível para obtenção e persistência de dados. A lógica de acesso a dados é camuflada pelo *Database Handler*. Nesta camada existe o componente *Updater* que permitirá, de forma dinâmica, alimentar o sistema com atualizações de sinais suportados e outras características. Permitirá também o envio de informação estatística para servidor central.

ADAS Modules é a camada de negócio onde se desenrola todo processo *core* do sistema avançado de assistência à condução. Pretende-se uma disponibilização modular das várias componentes *ADAS* a disponibilizar. Numa primeira fase, será disponibilizado o módulo de

Traffic Sign Recognition, dotado de um conjunto de algoritmos para identificação de sinalização. Esta camada requer informações relativas à sinalização à camada abaixo e, após reconhecimento, transmite informação gráfica à cada *ADAS Bussiness Logic*, podendo posteriormente persistir estatísticas relevantes.

As regras de negócio sobre o que deve ser apresentado ou não para o utilizador final estão presentes na camada *ADAS Business Logic.* A mesma, dispõe de inteligência para tentar perceber, por exemplo, se a sequência de sinais pode gerar alguma informação extra como, zonas de velocidade controlada, entre outros.

Por fim, a camada *Interface* recebe um conjunto de sinais identificados e outras informações, e realiza a sua interpretação, disponibilizando quase em simultâneo, informação para o utilizador. Para a transmissão áudio integra-se aqui a utilização de uma componente *Text-to-Speech* e *ORC* para possível reconhecimento de texto a partir da imagem obtida.

4.6 Metodologia de Reconhecimento de Sinalização de Trânsito Vertical

Pretende-se aqui detalhar a metodologia de deteção e reconhecimento de sinalização de trânsito vertical do Virtual Driver. A aplicação dos mecanismos falados neste subcapítulo será evidenciada e ilustrada posteriormente, no capítulo de implementação.

Tendo como base a abordagem de FeiXiang Ren, Jinsheng Huang, Ruyi Jiang e Reinhard Klette detalhada na secção 2.5.3.4 “Deteção ”,e evoluindo-a, a metodologia proposta no âmbito desta dissertação pretende realizar uma comparação entre as características de um *template* de um sinal de trânsito (pré-carregado em base de dados) com as características de um potencial sinal computado a partir do processo de deteção.

O processo de deteção de sinalização de trânsito presente na Figura 65, inicia com o *input* de uma imagem RGB capturada pelo sistema. O *frame* capturado passará por 3 processos nucleares, em primeiro lugar um pré-processamento para maximizar as probabilidades de identificação, de seguida, é realizada uma segmentação baseada em cor e uma segmentação por forma

A fonte de informação presente numa imagem poderá muitas das vezes não ser conclusiva. Pretende-se potenciar zonas da imagem que poderão passar despercebidas. Em primeiro lugar é realizada uma equalização do histograma da imagem (equalização de intensidade), que permite o melhoramento do contraste da imagem, favorecendo assim imagens com pouca luz. Para isso, será necessário converter o espaço cores RGB para um que faça a separação do canal da intensidade (Exemplo: YCrCb)

De forma a diminuir detalhe da imagem e assim melhorar a performance do processo, é aplicado um filtro de desfocagem (*Gaussian Blur)*, que permite o agrupamento de alguns pixels, processo que facilitará o processo de segmentação de cor.

Como é sabido, o espectro de cores RGB é bastante sensível a fatores de iluminação, e qualquer mudança dos seus valores, resultam em significativas mudanças de cor. Perante isso, existe necessidade de realizar uma conversão do espaço de cores para *HSV*.

A imagem gerada em HSV é segmentada em regiões através de aplicação de limites em *Hue* e *Saturation e Value*. Os valores a definir serão avaliados de forma empírica através de recolha de imagens de sinalização Portuguesa.

Através da segmentação de cor (para as possíveis cores da sinalização a reconhecer, isto é, é realizada uma segmentação por Vermelho, Azul, Amarelo e outras) são geradas imagens binárias (preto e branco) de onde é extraída uma lista de cores existentes na imagem que passará como contexto para a próxima etapa. A verificação da presença de uma determinada cor é realizada através da contagem de bits. É também definido um limite empírico para aceitação de presença de cor.

Para maximizar os resultados da segmentação de cor são combinadas respetivamente as operações de dilatação (maximização das zonas brancas - Figura 65) e erosão (minimização das zonas brancas - Figura 66) das imagens binárias. Este tipo de artefacto permitirá reduzir o ruído e ultrapassar possíveis falhas e "buracos" na morfologia dos sinais presentes na imagem. Pretende-se em primeiro lugar realizar a abertura de formas (*erode+dilate)* que permitirá reduzir algum ruído na imagem e seguida realizar o fecho (dilate+erode).

Figura 65 – Operação de dilatação da imagem binária [43]

Figura 66 – Operação de erosão da imagem binária [44]

A imagem binária resultante é submetida ao algoritmo de deteção de *blobs* o *MSER-Maximally stable extremal regions*. Este algoritmo é suportado pelo OpenCV, e tem como resultado uma

[43] Imagem retirada da página da comunidade OpenCV - http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html

[44] Imagem retirada da página da comunidade OpenCV - http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html

lista de regiões de interesse (retângulos) e uma lista de vetores com os respetivos pontos que constituem cada retângulo. Cada *blob* passa um processo de validação de área mínima.

A seguinte etapa, denominada "Segmentação por Forma", consiste na extração de novas regiões de interesse com base em formas e contornos detetados. Poderá ser reduzido algum *overhead* neste passo, pelo pré-carregamento da lista de formas possíveis de acordo com o mapeamento de sinais por cor. Por exemplo, caso seja identificada uma região azul no passo anterior, só poderão ser detetados círculos, e assim o foco passará para a deteção de círculos.

No processo de segmentação por forma, as regiões são submetidas a um algoritmo de deteção de limites do tipo *edge-effect* (Figura 67) que transforma a imagem binária num conjunto de contornos.

Figura 67 – Exemplo de aplicação de algoritmo de deteção de limites

Alguns contornos poderão estar incompletos em fenómenos de oclusão, seja por falha de segmentação de cor (existência de outras cores/reflexos), ou por fisicamente existir um objeto sobreposto. Desta forma, consoante as formas geométricas possíveis de deteção, são aplicadas transformações do tipo *Hough Transformations (*Figura 68*)* sobre a estrutura de limites. O tipo de transformação poderá ser por aproximação por linha ou por círculo.

Figura 68 – Maximização de resultados com a utilização de transformação Hough (Captura>Deteção de Limites>*Hough* Transformação>Disjunção lógica)

Este método deverá ser utilizado com precaução, para que não se excluam elementos verdadeiros da imagem. A partir do resultado deste processo, é executado um processo de deteção de contornos e respetivas hierarquias, que são submetidos a verificações de limites de área, perímetro, comprimento, largura e ângulos internos. Estes limites são definidos de forma empírica, baseados na estrutura de cada forma geométrica (Ex: Triângulo Equilátero possui os ângulos internos com 60°).

As regiões de interesse (*ROI – Region Of Interest)* que passarem nas validações geométricas são submetidas a um processo de normalização em tamanho, orientação e rotação, formando assim potenciais sinais de trânsito. Pretende-se submeter cada *ROI* a uma função afim, resultante da combinação de algumas transformações lineares (Figura 69) para que o resultado consiga ser, o mais semelhante possível, ao *template* a comparar. Estes tipos de transformações surgem da existência de divergências entre o plano de captura e o plano horizontal da sinalização.

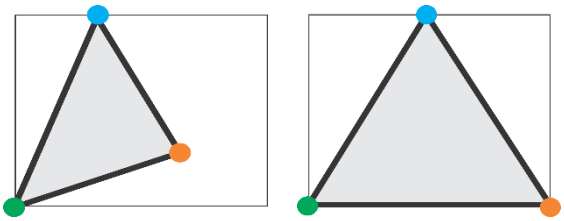


Figura 69 – Exemplo de transformação linear afim de um triângulo

Depois de realizadas as normalizações, é passado para o processo de reconhecimento, uma lista de candidatos e os respetivos mapeamentos, entre as cores e formas associados aos potenciais sinais (Figura 70).

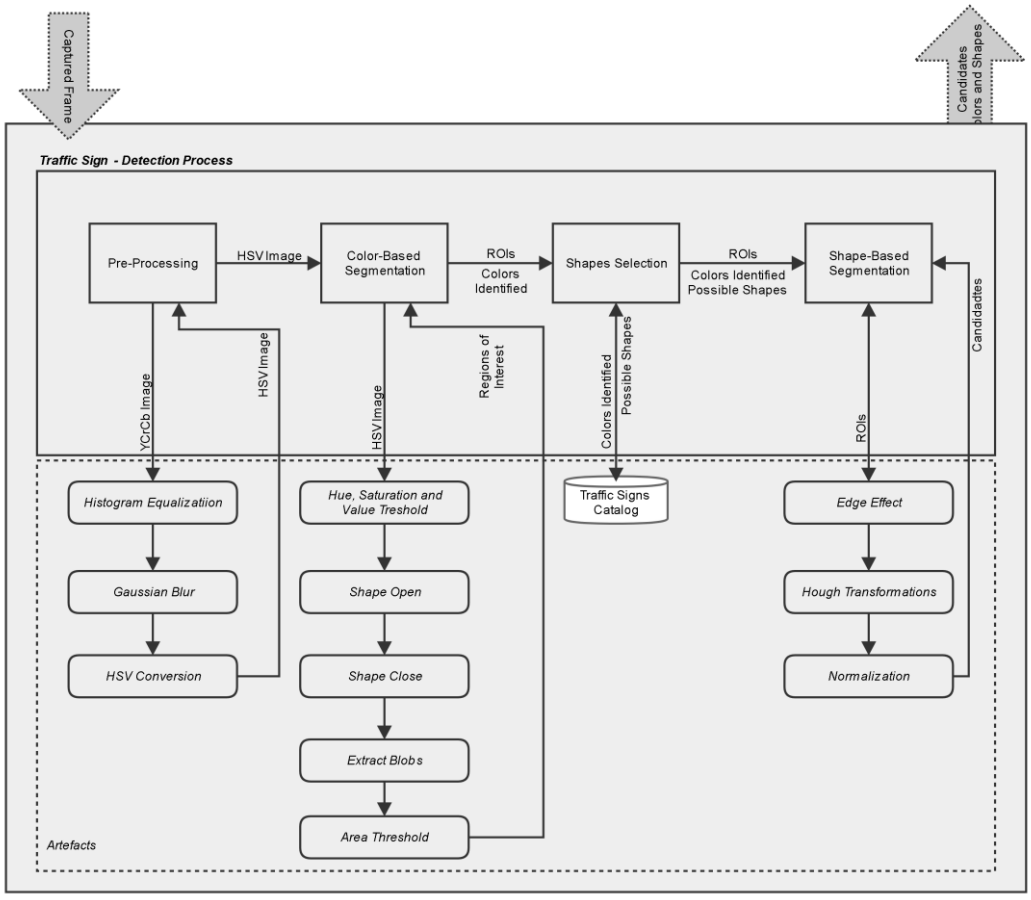


Figura 70 – Processo de Deteção *Virtual Driver*

Depois de detalhado o processo de deteção de sinalização, pretende-se esclarecer o processo de reconhecimento que recebe como input, o output do processo anterior.

De forma a minimizar o *overhead* no processo de reconhecimento, juntamente com os potenciais sinais identificados, são passadas informações sobre as cores e formas que foram detetadas na imagem inicial, por forma a serem somente carregados os modelos e características (pré-extraídas) de sinais possíveis, para esses parâmetros (Figura 71).

As características dos modelos já foram previamente extraídas e processadas antes da disponibilização do modelo para o Virtual Driver, o que permite reduzir o tempo total de reconhecimento, já que apenas será necessário realizar a extração e processamento das características dos candidatos e realizar a respetiva comparação.

Pretende-se dotar o sistema da compatibilidade com vários algoritmos de deteção de pontos de interesse e algoritmos de extração de características (*Features Detection and Features Extraction)*, para que na fase de avaliação da solução seja possível realizar comparação da eficiência e eficácia de cada processo para a melhor escolha. No âmbito desta dissertação, pretende-se explorar os algoritmos ORB para deteção de pontos de interesse e extração de características (*features descriptors)*.

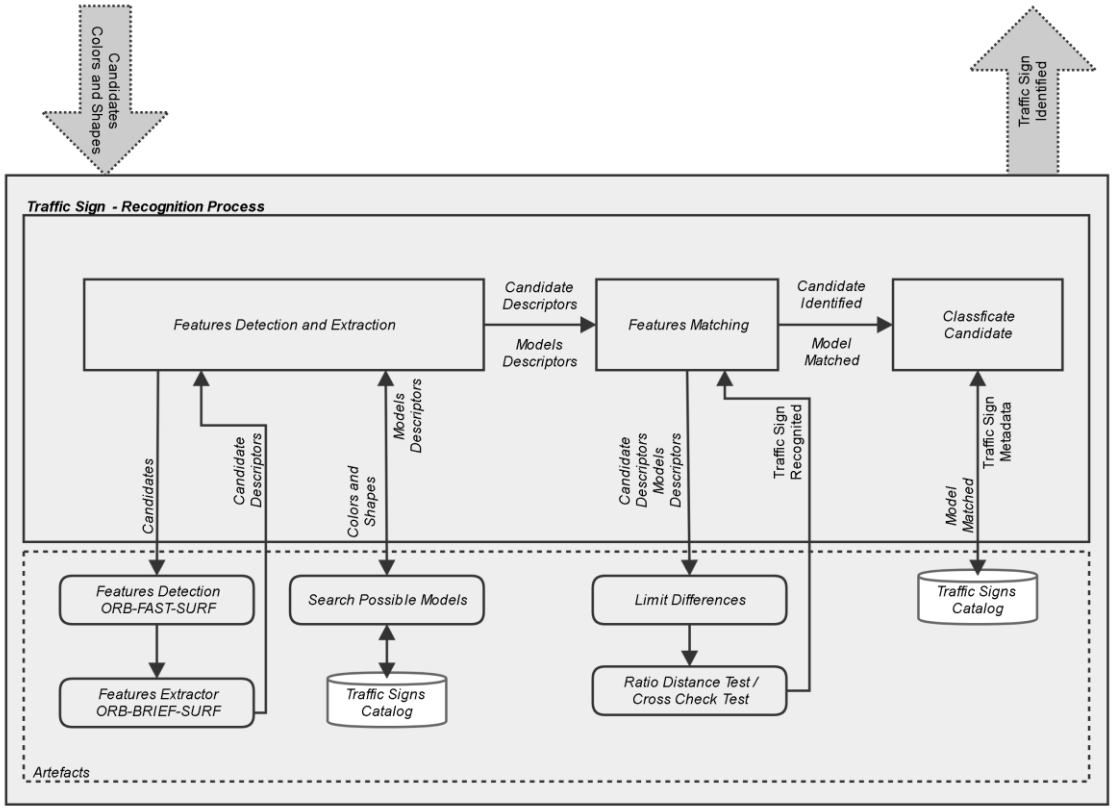


Figura 71 – Processo de Reconhecimento *Virtual Driver*

Após a o processamento dos descritores de características (*features descriptors)* é realizada uma comparação entre os descritores das regiões e os descritores dos possíveis modelos

(*features matching*). As diferenças são limitadas por valores de referência, que serão definidos empiricamente para cada modelo de sinal. Caso o total de semelhanças (*matches)* esteja acima do valor mínimo para aceitação, são realizados os testes de relação de distância (*Ratio Distance Test)*, entre os pontos de interesse do modelo e o candidato de pertencentes a cada *match,* tal como proposto por (Lowe, 2004). Se a distância entre os dois valores for menor de que um limite definido, então a relação de distância entre os pontos de interesse é aceitável. Conforme a configuração do sistema, poderá ser realizado o *Ratio Distance Test*, ou um algoritmo de comparação de *matches* para descritores binários, denominado *Cross Check.* O mesmo permite determinar a melhor relação entre as distâncias binárias calculadas, fornecendo para cada ponto de interesse a melhor semelhança.

Após aceitação do processo *feature matching,* o sistema classifica o sinal e prepara o *output* a passar à camada superior, através da leitura de informações relacionadas com o sinal (persistidas na base de dados).

4.7 Base de Dados

A estrutura de base de dados a utilizar, tem como objetivo a persistência de informação que alimentará a aplicação *Virtual Driver*. Neste subcapítulo, detalhar-se-á a estrutura do módulo de reconhecimento de sinalização. Pretende-se a persistência de informação relativa a:

- Sinal de Trânsito - Identificador do sinal de trânsito, família, tipo, subtipo, identificador do modelo e outras informações em formato áudio, texto ou imagem (Blobs);
- Modelo - Identificador do modelo, dimensões, forma, cor, *Blob* com a imagem do modelo, vetor com os pontos de interesse, *features* descriptor, numero mínimo de semelhanças, número mínimo de pontos de interesse;
- Forma – Identificador da forma, descrição, tipo, número de lados e ângulos internos;
- Cor – Identificação da cor, valores RGB, descrição;
- Detetor - Identificação do método, descrição, parâmetros;
- Extrator – Identificação do método, descrição, parâmetros.

O modelo “Entidade de Relacionamento”, representado na Figura 72, descreve a estrutura normalizada pensada para o módulo de reconhecimento de sinalização de trânsito. Deverá ser contemplada em trabalho futuro, uma estrutura de base de dados, que garanta a persistência de informação estatística.

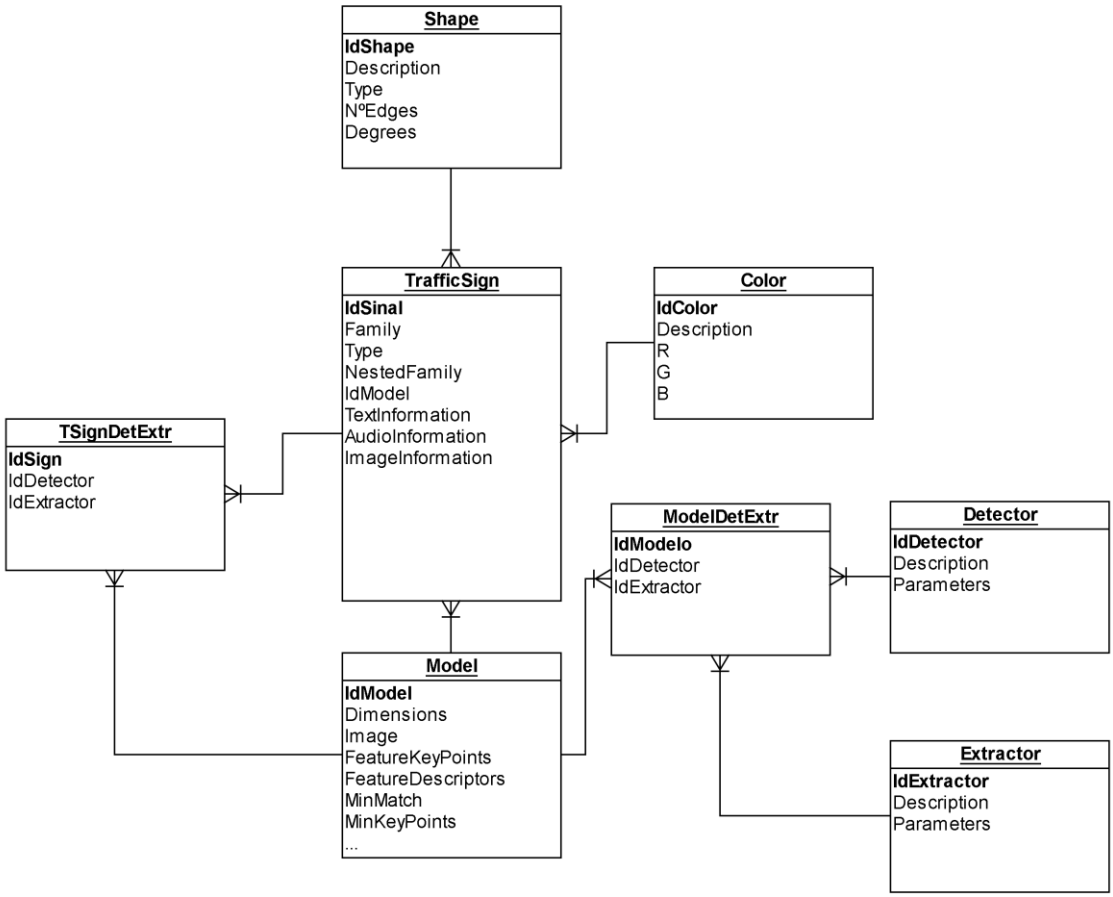


Figura 72 – Modelo entidade de relacionamento e atributos para módulo *TSR*

4.8 Conclusão

Durante este capítulo foi descrita a conceção do sistema Virtual Driver. Deste processo podem retirar-se várias conclusões relacionadas com o desenvolvimento de uma solução evolutiva.

A capacidade de o sistema poder ser instalado em diferentes dipositivos e meios, poderá permitir a que a solução evolua no sentido de ser incorporada com outros softwares.

Pretendeu-se definir uma arquitetura em camadas de forma isolar cada responsabilidade num domínio especifico, isto é, o processo de desenvolvimento da interface com o utilizador poderia ser realizado independentemente do processo de desenvolvimento do motor de reconhecimento de sinalização de trânsito.

Relativamente à metodologia de reconhecimento de sinalização de trânsito, pretendeu-se adotar a metodologia de *features matching* investigada em “2.5.3.10 -Abordagens Traffic Sign Recognition”, adoptando um conjunto de melhorias.

5 Desenvolvimento

Este capítulo apresenta, de forma detalhada, o processo de implementação do protótipo do sistema de reconhecimento de sinalização de trânsito vertical, Virtual Driver.

Orientados pelo design da solução definido no capítulo "4-Design", serão descritos os pressuposto base para desenvolvimento da solução, bem como, o ambiente de desenvolvimento, as tecnologias utilizadas, padrões aplicados, algoritmos e metodologia *Traffic Sign Recognition*.

Serão também referidas as ameaças que poderão condicionar a eficiência da solução.

5.1 Pressupostos para protótipo

Por forma a estabelecer um cenário controlado para validação do protótipo Virtual Driver, foram definidos um conjunto de pressupostos:

- **Sinalização**

  O conjunto de sinais de trânsitos alvo de reconhecimento deste protótipo restringe-se à realidade da sinalização Portuguesa. Numa primeira fase, o protótipo reconhecerá a sinalização de trânsito de formato triangular com delimitador vermelho (Figura 73).

Figura 73 - Tipologia de sinalização suportada

Enquadram-se nestas características a sinalização de perigo e cedência de passagem (imagens disponíveis no Anexo 1)

- **Condições de Captura**

Define-se como pressuposto para captura, imagens obtidas durante o meio-dia (existência da fonte de luz solar paralela ao solo).

A resolução de captura foi estabelecida em 1280x720 pixels, por ser uma resolução capaz de ser capturada pela maioria dos dispositivos atuais e permitir realizar um processamento rápido.

- **Cenário**

Foi selecionado um cenário controlado para uma primeira fase do protótipo, devido às divergências de formatos de sinalização.

5.2 Tecnologias utilizadas

Este subcapítulo tem como objetivo a descrição do ambiente de desenvolvimento e tecnologias utilizadas no protótipo Virtual Driver.

As tecnologias usadas serão descritas considerando dois processos: o de desenvolvimento da *interface* com o utilizador, e o de desenvolvimento do motor de reconhecimento ADAS.

5.2.1 Ambiente de Desenvolvimento

O desenvolvimento do protótipo foi realizado sobre o IDE Visual Studio Community Edition 2017 tendo em conta a sua total compatibilidade com o SDK Xamarin. Já para biblioteca de computação e processamento de imagem, foi selecionada a *EmguCV*.

Construiu-se uma solução multiplataforma sob a versão 4.5 da .NET Framework, estabelecendo-se uma organização em camadas para o código partilhado pelos diferentes códigos nativos (Figura 74).

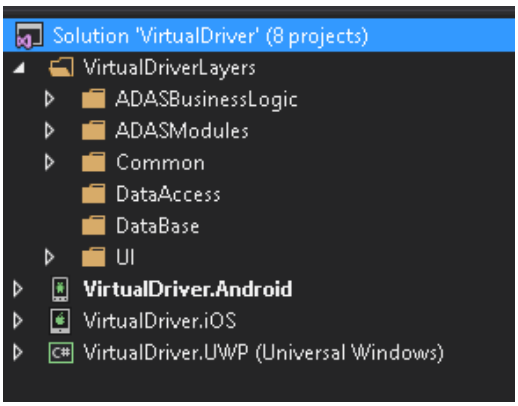


Figura 74 – Estrutura solução Virtual Driver

5.2.2 .NET Framework 4.5

Com a utilização da versão 4.5 da framework .NET da Microsoft é possível realizar a partilha de código comum entre as diferentes tecnologias dos sistemas nativos (Android, Windows, IOs), através da criação de bibliotecas do tipo "Portable Class Library".

A utilização deste tipo de bibliotecas possibilita que o desenvolvimento para multiplataforma tenha um custo e um tempo muito mais reduzido do que código nativo, porque toda a lógica e código comum podem ser partilhados com os projetos de cada plataforma, sem necessidade de repetição. O subconjunto de bibliotecas compatíveis com as PCL é mais reduzido que um processo *standard*, permitindo estabelecer maior robustez na compilação nativa.

Após a criação de um projeto do tipo PCL é possível definir as plataformas para as quais o código será compilado (Figura 75).

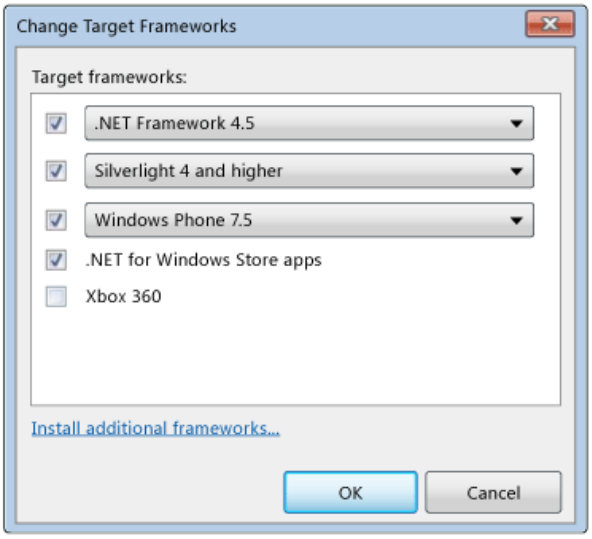


Figura 75 – Escolha de plataformas de compilação em PCL[45]

5.2.3 Xamarin Forms

De acordo com a solução *Visual Studio Cross Platform* criada, pretende-se retirar o máximo partido da reutilização de código para o desenvolvimento de *User Interfaces (UIs)*, facto garantido pela utilização de *Xamarin Forms.*

A tecnologia *Xamarin Forms* permite acelerar o processo de desenvolvimento de *Uis* nativas, através de desenvolvimento em código comum C# e/ou linguagem declarativa XAML. Na Figura

[45] Imagem retirada de: https://docs.microsoft.com/en-us/dotnet/standard/cross-platform/cross-platform-development-with-the-portable-class-library

76 é ilustrada a *UI* correspondente ao ecrã de configurações da aplicação Virtual Driver, produzida a partir do *XAML* presente no Código 2.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:VirtualDriverShell"
             x:Class="VirtualDriverShell.Settings">
   <TableView Intent="Settings" BackgroundColor="WhiteSmoke">
       <TableView.BindingContext>
           <local:SettingsViewModel/>
       </TableView.BindingContext>
       <TableRoot>
           <TableSection Title="Definições">
…
               <SwitchCell Text="Som de Aviso" On="{Binding WarningSound,
Mode=TwoWay}" />
               <EntryCell Label="Percentagem minima de semelhanças"
Text="{Binding MinMatchPercent, Mode=TwoWay}" />
       …
           </TableSection>
       </TableRoot>
   </TableView>
</ContentPage>
```

Código 2 – Código XAML da página de configurações

Os controlos visuais desenvolvidos em tempo de execução são mapeados para os controlos nativos de cada plataforma. Por exemplo, o controlo *Switch Cell* é mapeado em iOS para um *UISwitch* dentro de uma *UITableViewCell*, mas em Android é apenas uma *Switch* [46].

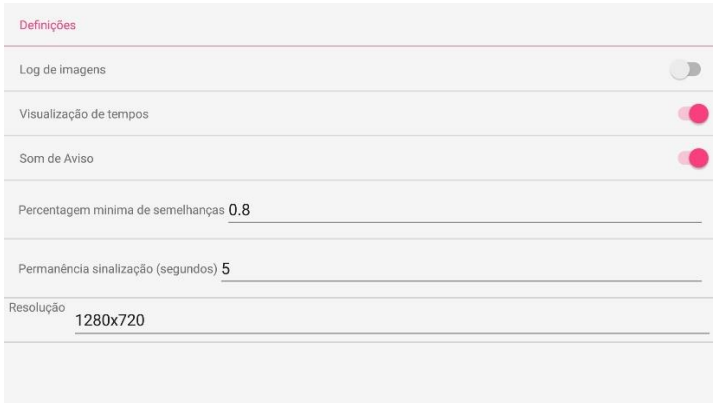


Figura 76 – Ecrã de configurações Virtual Driver

O desenvolvimento em *Xamarin Forms* viabiliza também a utilização do padrão *Model-View-ViewModel (MVVM).*

Este padrão procura separar as responsabilidades da apresentação e *User Interface* dos processos de gestão, e a manipulação de informação de negócio, reduzindo significativamente

[46] Informação poderá ser consultada em: https://developer.xamarin.com/guides/xamarin-forms/application-fundamentals/custom-renderer/renderers/#Views

o impacto da mudança em cada ecrã (*vista*). Esta separação facilita a capacidade de desenvolvimento do aspeto gráfico ("look") de cada ecrã, já que a informação de negócio fica completamente desacoplada da interface gráfica. Estes factos permitem que a camada de informação de negócio *Models* possa ser desenvolvida e testada independentemente.

A chave deste padrão é o uso de uma classe intermédia que faz a ponte entre a *UI* e a informação de negócio, denominada *ViewModel*. Esta atua sobre um processo dinâmico, que permite atualizar bidireccionalmente os dados, tanto através das *Views* como através de informação obtida pelo acesso a dados. O Xamarin possibilita a utilização do artefacto *Binding* para esse efeito.

Este padrão foi adoptado no desenvolvimento da camada de apresentação do Virtual Driver.

Xamarin Forms Controls

Importa aqui introduzir, alguns conceitos chave, sobre a forma como a UI do Xamarin Forms está estruturada.

Existem 4 tipos de controlos principais (Xamarin, 2017c) que podem estruturar uma aplicação em Xamarin Forms:

- *Page* – É o elemento raiz e que ocupa a maior parte do ecrã, ou seja, é a estrutura base para o desenvolvimento de uma vista em *Xamarin Forms*. Apenas poderá conter um filho. Alguns exemplos estão presentes na Figura 77;

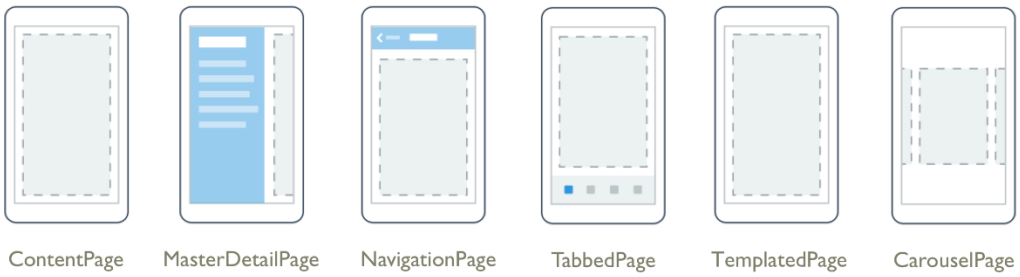


Figura 77 – *Xamarin Forms Pages*[47]

- *Layout* – Representa o controlo capaz de organizar e dimensionar o conteúdo do ecrã em *Xamarin Forms.* Funciona como contentor (*container)* e permite albergar outros controlos, nomeadamente *Views*. Alguns exemplos estão presentes na Figura 78;

[47] Imagem retirada de: https://developer.xamarin.com/guides/xamarin-forms/user-interface/controls/pages/

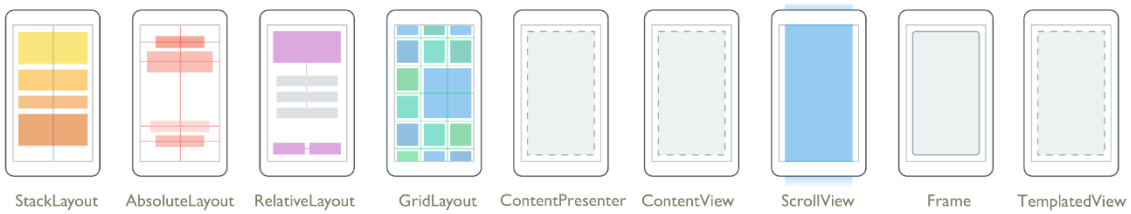


Figura 78 – *Xamarin Forms Layouts*[48]

- *View* – Representa um objeto visual comum, como botão, etiqueta, caixa de texto, seleção de data, caixa de seleção, entre outros (Figura 79);

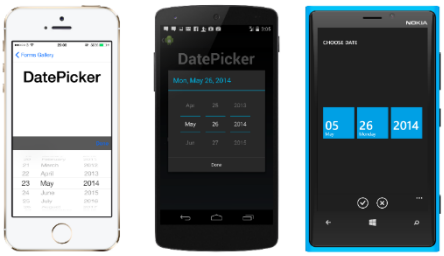


Figura 79 – Exemplo de controlo *DatePicker* para as 3 plataformas (iOS,Android e Windows)

- *Cell* – Representa o elemento capaz de estar presente dentro de uma lista ou tabela. Não é um elemento visual, mas sim um elemento que especifica de que forma será desenhado o controlo (*template*) (Figura 80).

Figura 80 – Exemplo de controlo *Cell* – *Entry Cell* para as 3 plataformas (iOS,Android e Windows)

5.2.4 EmguCv

Após contacto com a comunidade EmguCV, foi fornecida gratuitamente, uma cópia da biblioteca e do código fonte da versão compilável "Emgu CV v3.2 Professional", que garante

[48] Imagem retirada de: https://developer.xamarin.com/guides/xamarin-forms/user-interface/controls/layouts/

suporte para o desenvolvimento sob as plataformas Android, iOS e UWP e compatibilidade com Xamarin Forms.

A utilização desta *framework* é realizada com a simples adição de uma referência DLL ao projeto da camada responsável pelo processamento e reconhecimento de sinalização.

O EmguCv permite acelerar o processo de desenvolvimento, já que eleva o acesso a funções existentes no OpenCV, de forma prática e limpa. Exemplificando a possível conversão de uma imagem no espaço de cores RGB para escala de cinzentos, a mesma poderá ser realizada com as instruções em código C# incluídas no Código 3.

```
Bitmap bitmap = BitmapFactory.DecodeByteArray(vetorBytes, 0,
vetorBytes.Length);
Mat imagemMatricial = new Mat(bitmap);
CvInvoke.CvtColor(imagemMatricial, grayimage, ColorConversion.Bgr2Gray);
```

Código 3 – Conversão de imagem RGB para escala de cinzentos

Utilizando um vetor de *bytes* representativo de imagem, é possível a sua conversão para um Bitmap e de seguida a sua representação através de uma estrutura de dados em Matriz suportada pela nova versão do EmguCV, denominada Mat.

De seguida é consumida a função *CvtColor* proveniente do OpenCV, capaz de realizar conversões de espaço de cor.

5.3 Desenvolvimento Virtual Driver

Assente na arquitetura definida no subcapítulo "4.5 - Arquitetura" , foi realizada uma separação das responsabilidades de cada elemento do sistema numa estrutura em camadas:

- A camada UI é responsável pela definição da interação com o utilizador e eventos relacionados com a interação do dispositivo;
- A camada *ADASBusinessLogic* concebe em si, todas as regras de negócio, que definem que informação será disponibilizada para a camada de interface com o utilizador;
- *ADASModules* alberga todos os módulos inteligentes de apoio à condução. Para esta dissertação, foi desenvolvido o módulo de reconhecimento de sinalização de trânsito vertical;
- A camada *DataAccess* expõe todos os serviços de acesso a dados sobre sinalização, modelos, cores, formas e outras configurações. No âmbito desta dissertação, esta camada ainda não foi desenvolvida, sendo os recursos de informação fornecidos por *Embebbed Resources;*
- A persistência da informação em base de dados é garantida pela camada *DataBase*.

Todas estas camadas são maioritariamente desenvolvidas em projetos do tipo *PCL*, no entanto o código relativo aos módulos ADAS é endereçado a projetos de código nativo (Android, iOS e UWP), porque a biblioteca *EmguCV* possui funções adaptadas a cada plataforma.

Para efeitos de entrega de protótipo, foi desenvolvido o motor de reconhecimento de sinalização de trânsito para a plataforma Android.

5.3.1 Metodologia de desenvolvimento

De acordo com o requisito previsto inicialmente pelo LAMU, que definia que a solução deveria ser evolutiva, com a possibilidade de adoção de novos módulos de reconhecimento ADAS, agilizou-se o processo de desenvolvimento, por forma a garantir um baixo acoplamento e uma grande extensibilidade do sistema, através da utilização do padrão *Inversion Of Control.*

Inversion of Control

> "This inversion of control is sometimes named the Hollywood principle, "Do not call us, we call You".
>
> **(Michael Mattsson, 1999)**

O principio do padrão *Inversion Of Control* defende o controlo do sistema delegado para outros. Isto é, no processo de desenvolvimento de um componente de software, existe muitas vezes a necessidade de referenciar bibliotecas externas ao projeto, referência essa realizada em tempo de compilação. Com este padrão, pretende-se passar esse controlo para o momento de execução. A aplicação apenas necessita de saber o que invocar, sendo a informação sobre que o vai implementar, completamente agnóstica.

Focando num exemplo da solução VirtualDriver, a camada ADASBusinessLogic para invocação das funcionalidades dos módulos ADAS, necessitaria de uma referência a cada respetivo módulo em tempo de compilação (Figura 81).

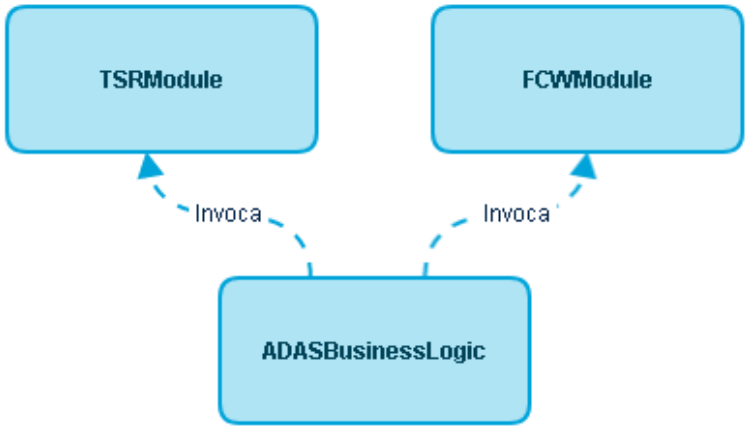


Figura 81 – Exemplo Virtual Driver cenário referência estática

Ainda assim, esta abordagem apresenta um conjunto de problemas[49] e limitações, tais como:

- Atualização ou mudança das dependências dos módulos obriga à alteração no próprio código da aplicação;
- Necessidade de saber quem vai implementar um determinado módulo, em tempo de compilação;
- Dificuldade de realizar testes independentes, já que não é possível substituir as dependências externas por um "*emulador"* em tempo de execução;
- Tempos de desenvolvimento maiores, porque a conclusão do desenvolvimento das funcionalidades com a utilização de componentes externos, ficará dependente destas referências.

A solução para este tipo de problemas poderá ser o usufruto de dois mecanismos: *Service Locator* ou *Dependency Injection*. No desenvolvimento do Virtual Driver optou-se pela utilização de *Dependency Injection,* com a utilização da *framework Unity.*

Dependency Injection

A solução para problemas descritos anteriormente na secção 0 *Inversion Of Control,* é fornecida pelo mecanismo *Dependency Injection*. O mesmo consiste em desacoplar a aplicação da dependência direta a outros projetos, introduzindo um elemento no meio, denominado "Assembler" (Martin Fowler, 2004), que em tempo de execução, permite determinar quem vai implementar um determinado comportamento (Figura 82).

[49] Informação presente na página Microsoft: https://msdn.microsoft.com/en-us/library/ff921087.aspx

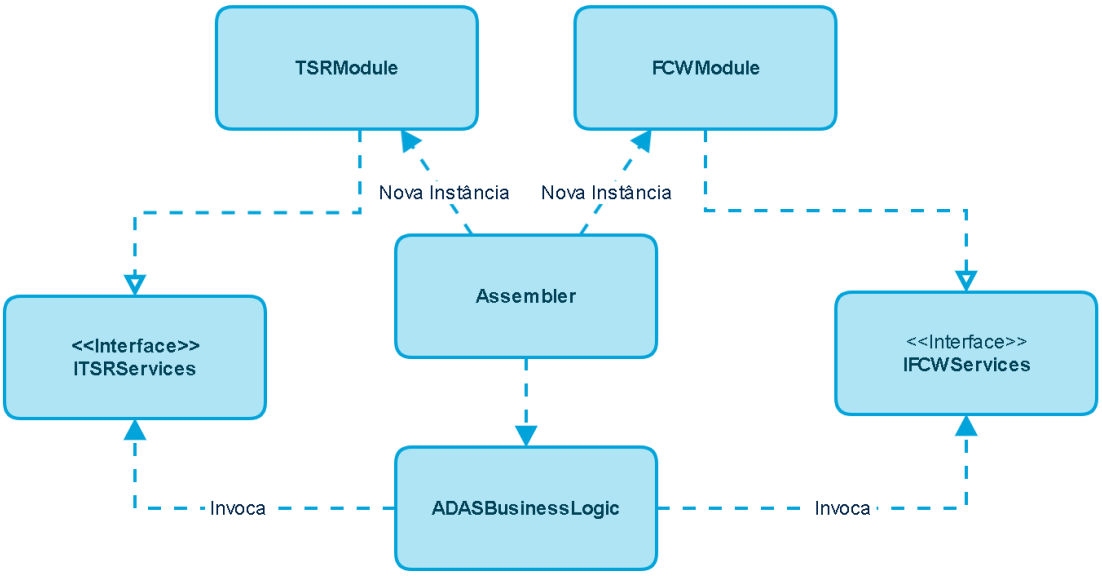


Figura 82 – Implementação *Dependency Injection*

Unity

O papel de Assembler descrito na seção anterior é desempenhado por um *Container* que representa uma instância única (Singleton), que perdura durante o ciclo de vida da aplicação. Este elemento é capaz de resolver todas as dependências automaticamente, em tempo de execução. Existem vários tipos de *Containers* presentes no mercado, o que foi utilizado para o desenvolvimento deste protótipo foi o *Unity Container*[50], suportado pela Microsoft.

Seguindo os passos de implementação da solução com mecanismo de *Dependency Injection*:

- Em primeiro lugar, é necessário realizar a declaração da instância representativa do *Container* no ponto de arranque da aplicação. Neste caso, será necessário declarar a instância no projeto de cada plataforma especifica (Android, iOS e UWP) ou num projeto comum e transversal à solução (Código 4);

```
…
protected override void OnCreate(Bundle bundle)
{
…
    base.OnCreate(bundle);
    global::Xamarin.Forms.Forms.Init(this, bundle);
    VirtualDriverCommonApp.Container = new UnityContainer();
…
```

Código 4 – Instrução para criação de *Unity Container*

[50] Fonte: https://msdn.microsoft.com/en-us/library/ff647202.aspx

- O próximo passo consiste em definir um gestor, capaz de gerir o acesso às instâncias que são resolvidas em tempo de execução pelo *Unity*, e registar essa responsabilidade (Código 5);

```
…
private VirtualDriverADASBusinessLogic()
{
VirtualDriverCommonApp.Container.RegisterType<VirtualDriverTSRManag
er>(new ContainerControlledLifetimeManager());
…
```

Código 5 – Atribuição para gestor *Unity*

- A definição do gestor contempla a receção, nos parâmetros do construtor, as *Interface*s que serão resolvidas pelo o *Unity Container*. O gestor é responsável pela disponibilização da invocação às referências externas. Foi definida a classe *VirtualDriverTSRManager,* capaz de aceder diretamente à instância responsável por executar os métodos de reconhecimento de sinalização de trânsito (Código 6);

```
…
public class VirtualDriverTSRManager
      {
private readonly ITSRServices _tsrEngine;

        public VirtualDriverTSRManager(ITSRServices tsrEngine)
        {
            _tsrEngine = tsrEngine;
        }
public  ADASModulesResponse  ProcessImage(Bitmap  imageData,  string
format, int width, int height)
                    {
return _tsrEngine.ProcessImage(imageData, format, width, height);
       }
…
```

Código 6 – Definição da classe gestor *VirtualDriverTSRManager*

- Seguidamente, a camada *ADASBusinessLogic* consegue, de forma dinâmica, invocar o módulo que alberga o comportamento *TSR,* através da invocação do método disponível no gestor (Código 7);

```
…
VirtualDriverCommonApp.Container.Resolve<VirtualDriverTSRManager>();
…
```

Código 7 – Acesso ao gestor *VirtualDriverTSRManager*

Outra das vantagens da utilização de *Unity* neste projeto é a possibilidade de definir em tempo de execução, e conforme a plataforma que está a ser executada, qual a instância compatível para uma determinada interface. Ou seja, se a aplicação corre em Android, então será instanciado o módulo *TSR* desenvolvido em Android. No ponto de partida de cada plataforma, define-se qual a classe que implementa uma determinada interface (Código 8).

```
…
// Android
protected override void OnCreate(Bundle bundle)
{
…
VirtualDriverCommonApp.Container.RegisterType<ITSRServices,
AndroidTSRHandler>();
…
// iOS
public override bool FinishedLaunching(UIApplication app,
NSDictionary options)
       {
            global::Xamarin.Forms.Forms.Init();

VirtualDriverCommonApp.Container.RegisterType<ITSRServices,
iOSTSRHandler >();
            LoadApplication(new App());

            return base.FinishedLaunching(app, options);
}
```

Código 8 – Definição de implementação da interface *ITSRServices* por plataforma

Em suma, a utilização do *Unity* no desenvolvimento do Virtual Driver, permite estruturar a solução, ao desenvolvimento dos diferentes módulos de *ADAS* de forma independente e em paralelo, para cada uma das plataformas, agilizando o processo de desenvolvimento.

5.3.2 Desenvolvimento User Interface

Com base no Design apresentado no subcapítulo "4.4 -Interface com o utilizador" foram desenvolvidos quatro ecrãs. Os mesmos são controlados por uma vista mãe denominada *Shell* que contêm um controlo *CarouselPage,* permitindo uma mudança fácil e intuitiva de vista, através de deslizamento (Figura 83).

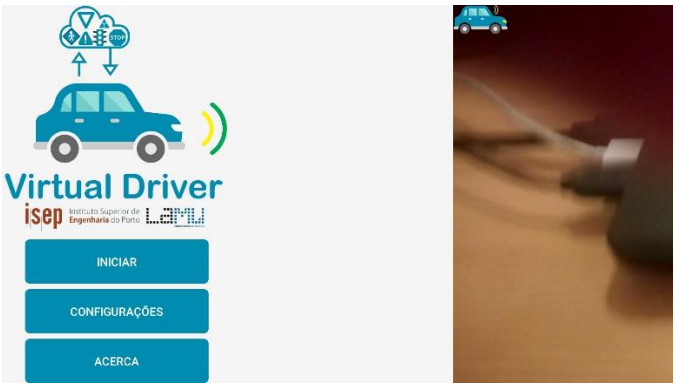


Figura 83 – Comportamento do controlo *CarouselPage*

Ecrã Inicial de carregamento de informação

Esta primeira vista permite indicar ao utilizador que está a ser efetuado um carregamento inicial de informação (Figura 84). O controlo *Xamarin Forms* que apresenta essa indicação é denominado *Activity Indicator.*

Internamente o VirtualDriver carrega um conjunto de configurações base do sistema, que já estejam ou não em cache na aplicação, e nesta primeira versão do protótipo, processa um conjunto de modelos disponíveis em *Embebed Resources*, para utilização pelo motor de reconhecimento.

Figura 84 – Apresentação do ecrã de entrada do Virtual Driver

Ecrã de navegação com menus de opções

É o primeiro ecrã que possibilita a interação do utilizador, concebendo em si a navegação do Virtual Driver. Assume-se como *pivot*, permitindo navegar para o ecrã de configurações e o ecrã de reconhecimento (Figura 85). Pretende-se que a evolução desta *UI* permita incluir outras opções, bem como, deter em si a possibilidade de responder a comandos de voz.

A estrutura desta *interface* é composta por uma série de botões que permitem adicionar a navegação da *Shell* para a página indicada.

Figura 85 – Navegação do Virtual Driver

Ecrã de Configurações

O Ecrã de configurações permite ao utilizador gerir alguns parâmetros vitais do processo de reconhecimento, bem como, o rastreamento da informação. Nesta fase não se definiram configurações específicas do aspeto da *UI.*

Esta UI é composta por uma tabela de opções do tipo *TableView ,* que para cada linha contém um conjunto de controlos como *SwitchCell, EntryCell e* extensões do tipo *ViewCell* (Figura 86). Através do padrão MVVM, a alteração da informação presente no ecrã, reflete-se num modelo global de configurações da aplicação, através da utilização de um *plugin* denominado "`Plugin.Settings.Abstractions`".

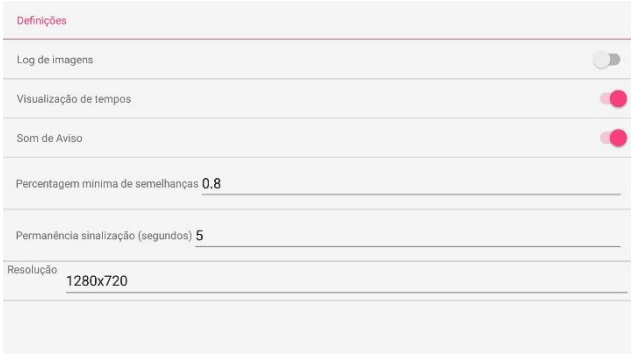


Figura 86 - Ecrã de configurações Virtual Driver

As opções disponíveis de configuração para o utilizador são as seguintes:

- Possibilidade de gravação de *log* das imagens reconhecidas pelo motor de reconhecimento;
- Possibilidade de apresentação dos tempos de processamento;
- Ativação/Desativação de alertas sonoros;
- Possibilidade de alteração da percentagem mínima de aceitação das semelhanças entre possíveis candidatos e os respetivos modelos;
- Definição do tempo de permanência da sinalização no ecrã de reconhecimento;
- Definição de resolução de captura a utilizar no motor de reconhecimento.

Ecrã de reconhecimento *ADAS*

Este ecrã assume em si o resultado principal da aplicação, apresentando uma pré-visualização da captura do meio ambiente, em tempo real, e alertando através de áudio, imagem e texto, sobre a sinalização identificada. Poderá incluir informações de tempos de processamento, entre outras.

Na Figura 87, está presente a disposição atual do ecrã de reconhecimento, que é composto por 4 zonas de informação:

Figura 87 – Zonas do ecrã de reconhecimento

- Zona de previsualização de captura;
- Zona de imagens de sinalização reconhecida;
- Zona de texto relacionado com a sinalização reconhecida;
- Zona de métricas (tempos de reconhecimento, rate de captura, etc);

A informação em formato áudio é transmitida através da utilização do motor Text-To-Speech disponível no dispositivo. As informações relacionadas com a sinalização são dinamicamente faladas pelo motor na linguagem definida no aparelho.

A implementação Text-To-Speech não é possível de ser implementada diretamente pelo Xamarin Forms, esse comportamento é delegado a código nativo (desenvolvido por plataforma) através da utilização de *Dependency Services*.

Devido à limitação do *Xamarin Forms* para controlos que permitam visualizar, em tempo real, *frames* capturados pela câmara do dispositivo, foi necessário recorrer a um mecanismo previsto pelo *Xamarin,* que é o desenvolvimento de um *Custom Renderer.*

Por defeito, os controlos *Xaramin Forms* são desenhados para controlos nativos correspondentes. Com o *Custom Renderer*, procura-se circuitar esse passo e desenvolver nativamente um controlo capaz de apresentar o resultado da câmara do dispositivo. Para isso será necessário o desenvolvimento de um *Custom Renderer* para cada plataforma (Figura 88). Este tipo de flexibilidade, permite que, neste caso, seja desenvolvido um processo de captura e pré-visualização de imagem mais otimizado para cada plataforma, garantido uma melhor gestão dos recursos do dispositivo.

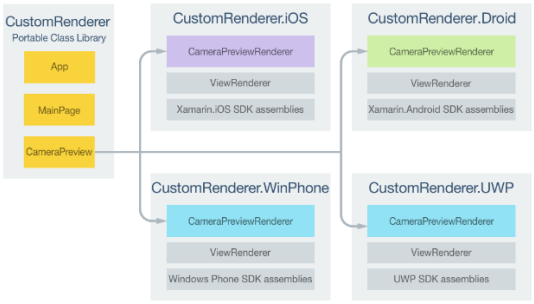


Figura 88 – Mapeamento entre controlo Xamarin Forms e o respectivo *Renderer*[51]

Descrevendo os passos para o desenvolvimento do *Custom Renderer,* para a pré-visualização da captura*:*

1. Na *Portable Class Library* que representa a camada de apresentação do Virtual Driver, foi necessário a criação de um controlo com herança *View,* que compõe a vista de reconhecimento (Código 9);

```
…
public class ADASCamera : View, IDisposable, INotifyPropertyChanged
    {
…
<pages:ADASCamera x:Name="cameraComponent" Camera="Rear" Grid.Column="0"
Grid.ColumnSpan="2" Grid.Row="0" Grid.RowSpan="3"
HorizontalOptions="FillAndExpand" VerticalOptions="FillAndExpand"
ImageCapturedEvent="cameraComponent_ImageCapturedEvent"
ImageCapturedFormat="png"/>
…
```

Código 9 – Declaração do controlo *Custom Renderer* no ecrã de reconhecimento

2. No projeto da plataforma Android, foi criado o respetivo *Custom Renderer*, que permite registar que, quando o controlo *ADASCamera* for desenhado, deverá ser realizado o *override* para o controlo nativo *ADASCameraView2* (Código 10). Neste caso o tipo de *Renderer* é um *ViewRenderer,* já que permite uma maior flexibilidade no desenvolvimento do controlo. O controlo nativo é composto por um controlo do tipo *TextureView*, capaz de apresentar um *stream* de imagem;

```
[assembly: ExportRenderer(typeof(VirtualDriverShell.ADASCamera),
typeof(ADASCameraRenderer))]
namespace VirtualDriver.Droid
{
    public class ADASCameraRenderer :
ViewRenderer<VirtualDriverShell.ADASCamera,
VirtualDriver.Droid.ADASCameraView2>
    {

        private ADASCameraView2 cameraPreview;
```

[51] Imagem retirada de: https://developer.xamarin.com/guides/xamarin-forms/application-fundamentals/custom-renderer/view/

```
        protected                          override                          void
OnElementChanged(ElementChangedEventArgs<VirtualDriverShell.ADASCamera> e)
        {
            base.OnElementChanged(e);

            if (Control == null)
            {
                cameraPreview = new ADASCameraView2(Context, e, this);
                SetNativeControl(cameraPreview);
            }
        …
```

Código 10 – Definição do *Custom Render* na plataforma Android.

3. Na criação deste tipo de controlos, é necessário realizar a respetiva gestão do ciclo de vida. Para isso, foi reescrito o evento OnElementChanged, para que o seja controlado o processo de captura e apresentação de *frames.*

Camera2 API

A utilização da câmara do dispositivo para captura e pré-visualização em tempo real, requer o desenvolvimento de um processo robusto, responsivo e eficiente, uma vez que os recursos são limitados e partilhados pelas restantes aplicações, e pretende-se uma resposta instantânea para processo de reconhecimento.

De acordo com o desenvolvimento do *Custom Renderer* para a plataforma Android, considerando a pré-visualização e captura de imagem, foi utilizada a API Android para acesso a funções da câmara, denominada Camera2 API.

Descrevendo por ordem de processo, os tópicos gerais do diagrama de classes da API, representado na Figura 89:

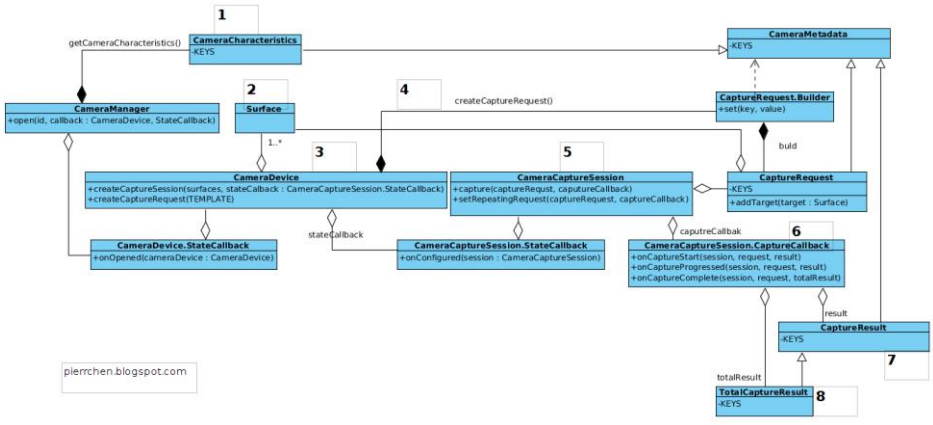


Figura 89 – Diagrama de classes da *Camera2 API*[52]

[52] Imagem retirada de: http://pierrchen.blogspot.pt/2015/01/android-camera2-api-explained.html

- É disponibilizado um elemento responsável pela gestão das câmaras do dispositivo denominado *CameraManager.* Este elemento é percorrido de forma a recolher as características de cada câmara. No caso do Virtual Driver é procurada a câmara traseira do dispositivo;

- Após a recolha das características da câmara (*Camera Characteristics*), são definidos parâmetros de superfícies que vão representar a captura. São configuradas duas *Surfaces*, aquela que apresentará o resultado da captura no ecrã do tipo *SurfaceTexture* e aquela que será responsável por armazenar o *frame* para processamento e reconhecimento do tipo *ImageReader*;

- De acordo com a orientação do dispositivo, são realizadas transformações nas superfícies, para que a orientação da captura seja coincidente com a mesma;

- Através da classe *CameraDevice* é realizada abertura da camara para início de captura. O processo é assíncrono, permitindo obter resultado através da implementação de um *callback*, denominado CameraDevice.*StateCallback*;

- Em caso de sucesso na abertura, é preparada a criação de uma sessão de captura através da classe *CameraCaptureSession.* Esta sessão corre assincronamente, e é responsável pela gestão de pedidos de captura. Recebe como contexto a lista de superfícies para as quais vai dar resposta;

- Após criação da sessão (resposta recebida através de *callback*, denominada *CameraCaptureSession.StateCallBack),* é definido um pedido de captura em ciclo para a previsualização de captura. Esse pedido é construído através da classe *CaptureRequest.Builder* tipificado como sendo de pré-visualização, definindo também, como superfície de destino, o controlo *TextureView*, responsável por apresentar o *stream* de imagem.

  Através do método SetRepeatingRequest disponível pela instância da sessão, é possível iniciar o ciclo de pedidos. A resposta de captura de cada *frame* é recebida através de um *callback* do tipo *CameraCaptureSession.CaptureCallback*;

- Até este ponto, ainda nenhum *frame* foi enviado para o motor de reconhecimento ADAS. Esse momento acontece após a apresentação do primeiro *frame* de pré-visualização. Desta forma, sobre a sessão de captura é realizado um novo pedido único através do método *Capture* da classe *CameraCaptureSession*, reescrevendo o pedido de pré-visualização em ciclo a decorrer. O resultado desse pedido de captura é direcionado para uma superfície do tipo *Allocation,* que permite alocar os resultados dos *framess* capturado em formato original *YUV.* Este tipo de superfície permite fazer

uso da *framework* RenderScript que tem como objetivo acelerar o processo de computação através de aceleração por hardware.

- O pedido de captura é realizado de forma assíncrona e despoletará a respetiva *callback* responsável por avisar a chegada de uma nova imagem à superfície. O processo de pré-visualização retoma instantaneamente, após a captura desse *frame*. Já o frame segue para o processo de reconhecimento. A próxima captura, será requisitada pelo motor de reconhecimento quando tiver disponibilidade para processar um novo frame. Pretende-se aqui realizar um escalamento deste processo para melhoria de tempo final.

RenderScript

De forma a acelerar o processo de captura e reconhecimento foi utilizada a *framework* RenderScript. Ela permite executar tarefas intensivas de processamento através de uma alta taxa de resposta em sistemas Android.

A sua utilização é orientada para tarefas de computação paralela, rentabilizando os recursos do dispositivo, nomeadamente processadores com vários núcleos e processamento através de *GPU* (Google, 2017d).

No desenvolvimento do Virtual Driver, são utilizados *RenderScripts* para processamento de imagens e conversão de formato original de imagem capturada (YUV) para representação em RGB.

Gestão de Permissões

De acordo com a política de permissões associada à instalação de aplicação no dispositivo móvel, foi definido um conjunto de permissões base para a execução do Virtual Driver. O sistema é capaz de identificar a falta de permissões e alertar o utilizador para a respetiva autorização (Figura 90).

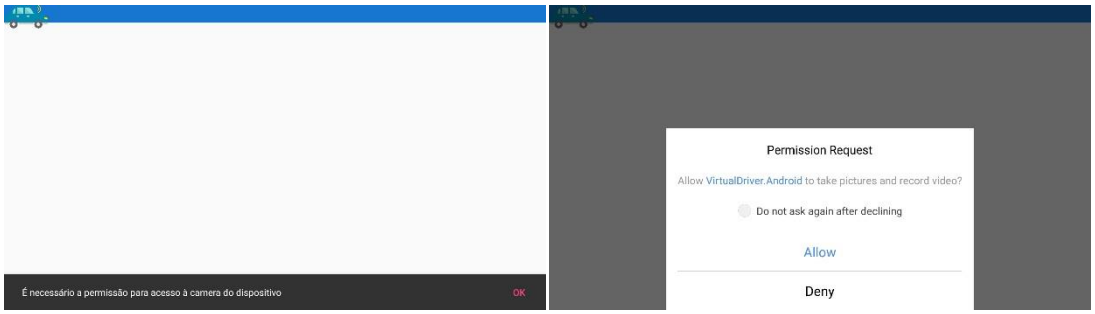


Figura 90 – Exemplo de gestão de permissões em Android

5.3.3 Desenvolvimento Módulo TSR

De acordo com a metodologia proposta no subcapítulo "4.6 - Metodologia de Reconhecimento de Sinalização de Trânsito Vertical", serão descritos aqui os processos de deteção e reconhecimento da sinalização de trânsito desenvolvidos para o protótipo.

5.3.3.1 Deteção

Após a obtenção de um *frame* através da câmara do dispositivo, o mesmo é submetido a um pré-processamento.

Pré-Processamento

Em primeiro lugar é realizada uma equalização do histograma da imagem sobre a componente da intensidade de luz. De forma resumida, o histograma de uma imagem, é uma representação da distribuição de cada *pixel* pelas intensidades presentes resultando numa relação ilustrada na Figura 91.

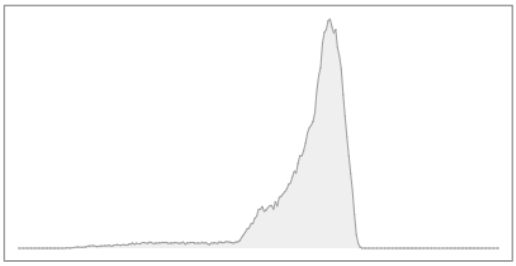


Figura 91 – Histograma de intensidades

Pretende-se elevar as pequenas intensidades, para que objetos, discretos pela falta de luz , possam ser reconhecidos.

Para realizar esta operação através da a biblioteca EmguCV, é necessário em primeiro lugar converter a imagem capturada para um espaço de cores que isole o canal da intensidade. Para isso optou-se pela conversão da imagem em RGB para o espaço YCrCb (espaço de cores normalmente utilizado para a transmissão de vídeo digital). Este espaço concebe em si um canal dedicado à luminância (Y). Dessa forma, após a conversão para este formato, é realizada uma extração do canal número 0 (Y) e realizada a respetiva equalização do histograma. Os canais são unidos novamente, e a imagem convertida para RGB. As instruções para esta transformação, estão presentes no Código 11, e o resultado ilustrado na Figura 92.

```
using (VectorOfUMat channels = new VectorOfUMat())
{
    CvInvoke.CvtColor(image,                        preprocessedImage,
  ColorConversion.Bgr2YCrCb);

    CvInvoke.Split(preprocessedImage, channels);

    CvInvoke.EqualizeHist(channels[0], channels[0]);
```

```
            CvInvoke.Merge(channels, preprocessedImage);

            CvInvoke.CvtColor(preprocessedImage,          preprocessedImage,
            ColorConversion.YCrCb2Bgr);
        }
```

Código 11 - Instruções para equalização do Histrograma da imagem em EmguCV

Figura 92 - Resultado da equalização do Histograma da imagem (original à esquerda)

Outro mecanismo realizado no pré-processamento da imagem do Virtual Driver é o da desfocagem.

Para uma melhoria do processo de segmentação baseado em forma, que tem como matéria de operação um conjunto de contornos detetados na imagem, é realizada uma redução ao detalhe da imagem para diminuição do o ruído nela existente. O filtro de desfocagem permite, para um determinado *pixel*, enquadrá-lo num conjunto de características comuns entre os pixels vizinhos. Para isso, foi utilizado o método *Gaussian Blur* (Desfocagem Gaussiana) disponível no OpenCV, que tal como o nome indica, faz uso da função Gaussiana. Os parâmetros relacionados com o desvio e o *kernel* Gaussiano foram definidos empiricamente através dos testes realizados. O resultado da utilização deste método é ilustrado na Figura 93.

Figura 93 - Resultado da ligeira desfocagem Gaussiana (original à esquerda)

Após este passo, o pré-processamento fica concluído através da conversão do espaço de cores RGB para o espaço HSV, capaz de representar melhor as variações relativas ao brilho da imagem, sem descurar as cores nela presente.

HSV permite definir representação da cor através de três canais, Hue*, Saturation* e *Value*. Detalhando um pouco a ilustração cónica deste espaço de cores (Figura 94), *Hue* representa a roda da tonalidade da cor. Um elemento azul será igual em *Hue* a uma objeto azul sombreado.

Saturation ou saturação, em Português, permite definir o quão pura e carregada será uma cor. Um objeto azul puro tem saturação máxima, já um objeto azul sombreado, terá uma menor saturação.

O Valor (*value)* determina a quantidade da luz presente na imagem, se não existir luz ela poderá ser representa pelo preto, com valor 0.

Na maioria das escalas de valores disponíveis para este espaço, os valores possíveis são para Hue [0-359], *Saturation* [0-100] e *Value* [0-100], no entanto o OpenCV apresenta uma escala diferente para a qual foi necessária a conversão de valores, em que H [0-180], S [0-255] e V [0-255].

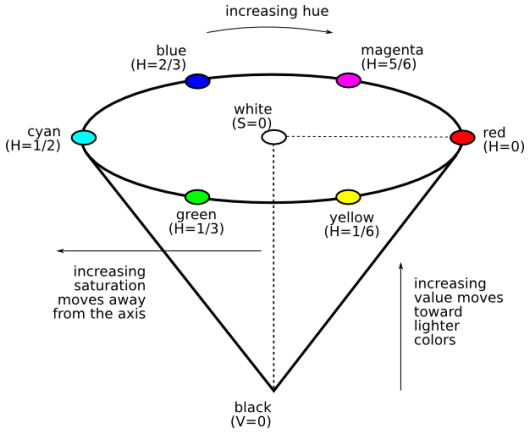


Figura 94 – Representação do espaço HSV através do "*single-hexcone model"*[53]

5.3.3.2 Segmentação baseada em cor

Este é o momento em que é procurada na imagem pré-processada, a presença de determinadas cores (como o Vermelho, Azul, Amarelo), que poderão representar potenciais sinais de trânsito. Para isso, será necessário realizar uma segmentação para cada uma das cores, através da definição de limites de aceitação para cada um dos elementos do espaço HSV. Os limites foram definidos empiricamente.

Descrevendo a segmentação por vermelho, em primeiro lugar, é realizada uma verificação de limites exclusivos de valores para *Hue,* porque o canal é "circular". Pretendem-se excluir todos os valores que estão no intervalo de não aceitação (Figura 95).

[53] Imagem retirada de : http://infohost.nmt.edu/tcc/help/pubs/colortheory/web/hsv.html

Figura 95 – Zona de exclusão *Hue*

Em EmguCV, esta operação poderá ser realizada através das seguintes instruções: *InRange* para sinalizar todos os *pixels* da imagem com o valor 1 (resultado em imagem binária) que se encontrem na zona de exclusão. De seguida, com o resultado criado (máscara), é realizada uma inversão binária através da função *BitwiseNot*, para obter o resultado esperado (todos os *pixels* com valores *Hue* aceitáveis para vermelho) (Código 12).

```
using (ScalarArray lower = new ScalarArray(colorRange.Value[0].Hue))
       using (ScalarArray upper = new ScalarArray(colorRange.Value[1].Hue))
                CvInvoke.InRange(hue, lower, upper, mask);
 …
 CvInvoke.BitwiseNot(mask, mask);
```

Código 12 – Instruções para segmentação do canal *Hue*

O mesmo processo se repete para os seguintes elementos HSV, resultando no final uma máscara representativa da segmentação baseada em cor (Figura 96).

Figura 96 - Resultado do processo de segmentação baseada na cor vermelha.

De forma a potencializar o resultado deste subprocesso a máscara foi sujeita a duas transformações de abertura de forma (erosão e dilatação) e fechamento de forma (dilatação e erosão), respetivamente, através das funções *Dilate* e *Erode* presentes no *OpenCV*. Estas operações permitem reduzir o ruído presente na imagem e procuram minimizar o impacto de possíveis falhas nos limites da sinalização, que possam existir na imagem.

Após a aplicação de limites de cor, é invocado o algoritmo MSER para extração de zonas de interesse denominadas *blob*s e realizada uma validação quanto à área mínima de aceitação de blobs.

O processo de segmentação por cor fornece à etapa seguinte, uma lista de regiões de interesse e as respectivas cores, que passaram o processo de aceitação de limites.

5.3.3.3 Segmentação baseada em forma

Após a determinação das regiões da imagem onde estão presentes cores que poderão representar sinalização de trânsito, existe um passo intermédio do processo de identificação do Virtual Driver, que se baseia no mapeamento da Cor> Forma > Sinal, para definir "à cabeça" uma lista de formas possíveis de serem segmentadas neste processo.

Para cada região de interesse é aplicado um algoritmo de deteção de limites sobre a respetiva máscara. É invocada a função *Canny* presente no *OpenCV*, que permite implementar o algoritmo *Canny Edge Detector* desenvolvido por John Canny. (Canny, 1986). Os limites inferior e superior definidos para aceitação de cada *pixel* como fazendo parte constituinte de um limite, foram definidos empiricamente.

Para colmatar possíveis pequenas falhas, devido à presença de elementos sobre o sinal, é realizada uma transformação *Hough Line* para deteção de linhas retas nos limites definidos anteriormente. Esta transformação procura, de acordo com uma série de parâmetros, como número de interseções entre pontos, tamanho mínimo da reta ou número máximo de falhas entre pontos, definir um conjunto de linhas retas como *output.*

A possibilidade de ser tolerante a falhas entre pontos, permite colmatar possíveis defeitos nos limites detetados, ainda que a utilização deste parâmetro não resolva o fenómeno de oclusões em caso de *gaps* significativos. A Figura 97 apresenta o resultado do processo executado sobre a máscara inicial, sujeita a deteção de limites *Canny*, seguido de uma transformação *Hough Line* e da unificação dos dois resultados. As instruções *EmguCV* para este comportamento estão presentes no Código 13.

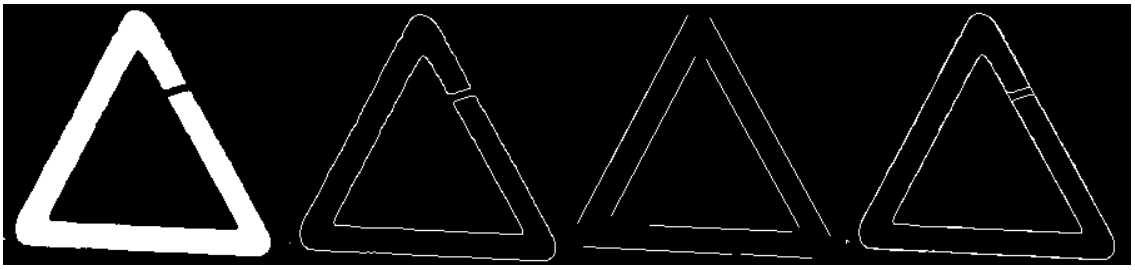


Figura 97 - Aplicação dos algoritmos *Canny Edge Detector e Hough Tranform* sobre a mascara

```
        using (UMat canny = new UMat())
         {
                        CvInvoke.Canny(roi canny, 50, 150);

    LineSegment2D[] lines = CvInvoke.HoughLinesP(canny, 1, Math.PI / 180, 25, 30,
    8);

                        var gray = new Image<Gray, Byte>(canny.Size);

                        foreach (LineSegment2D line in lines)
                            gray.Draw(line, new Gray(255), 1);
                        CvInvoke.BitwiseOr(gray, canny, canny);
          …
```

Código 13 – Implementação EmguCV para deteção de limites, *Hough Tranform* e unificação.

Após a maximização dos contornos existentes na zona candidata, existe necessidade de realizar a validação geométrica de cada elemento, a fim de verificar se correspondem às formas dos sinais de trânsito a reconhecer.

É ainda necessário obter a lista de contornos e a respetiva hierarquia entre eles.

No caso da sinalização triangular com borda vermelha, pretende-se a obtenção do triângulo interno, descartando o triângulo exterior. Desta forma, é utilizada a funcionalidade *OpenCV* denominada *FindContourTree,* que permite mapear numa estrutura de dados as relações entre os objetos presentes na imagem, como definir para um determinado objeto, se tem um pai, ou se é um filho.

Em OpenCV essa hierarquia é representada por um vetor de 4 valores do tipo [Seguinte, Anterior, Primeiro Filho, Pai] (OpenCV, 2015). O valor "Seguinte" diz respeito ao próximo elemento no mesmo nível de hierarquia, o "Anterior" permite definir na mesma hierarquia o contorno anterior, o "Primeiro Filho" representa qual o primeiro filho no interior de um determinado contorno e o "Pai", o agregador dos contornos na mesma hierarquia de verificação.

Considerando a geometria de um triângulo, um cenário possível da utilização deste artefacto poderá ser representado como na Figura 98.

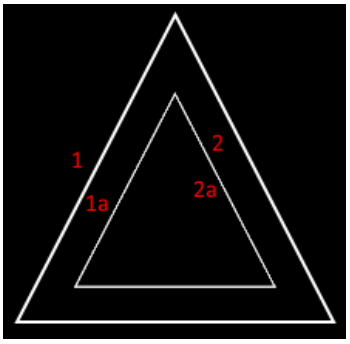


Figura 98 –Estrutura Hierárquica de contornos

Estão presentes na imagem, 4 contornos (1,1a,2, 2a.), sendo que, a respetiva relação entre eles poderá ser definida ser pelo seguinte vetor:

$$\begin{bmatrix} -1 & -1 & 1a & -1 \\ -1 & -1 & 2 & 1 \\ -1 & -1 & 2a & 2 \\ -1 & -1 & -1 & 2a \end{bmatrix}$$

O algoritmo de deteção procura, na estrutura de relações de forma recursiva, o último elemento da hierarquia [-1,-1,-1,2a] e submete-o a um processo de validação de forma. Em caso de insatisfação, o algoritmo verifica no elemento imediatamente acima, se é satisfeita a validação de forma. Este mecanismo permitirá maximizar os resultados, evitando que na busca ao último elemento da hierarquia, seja retornado um falso positivo, ou seja, um não triângulo,

pelo simples facto de existir ruído na imagem. Dessa forma, o possível triângulo, poderá estar nos níveis imediatamente acima.

Aproximação de Contornos

Para cada contorno pesquisado é invocada uma a função *OpenCV ApproxPolyDP* que permite com precisão, realizar uma aproximação da curva do objeto, através de outra curva com menos vértices, e de acordo um parâmetro que define um limite máximo entre as duas curvas. O comportamento desta funcionalidade implementa o algoritmo "Douglas-Peucker". Descrevendo o respetivo funcionamento (através de um exemplo [54] disponibilizado pela comunidade *OpenCV),* ilustrado na Figura 99, compreende-se que o pretendido é que, na imagem à esquerda, seja realizada uma aproximação de curva, para obter o quadrado implícito na forma.

A sua deteção foi realizada com sucesso (imagem do meio) com a definição do parâmetro *epsilon* (distância máxima entre curvas) em cerca de 10% do tamanho da curva original do objeto, no entanto, o resultado, a 1%, já não é animador (imagem da direita).

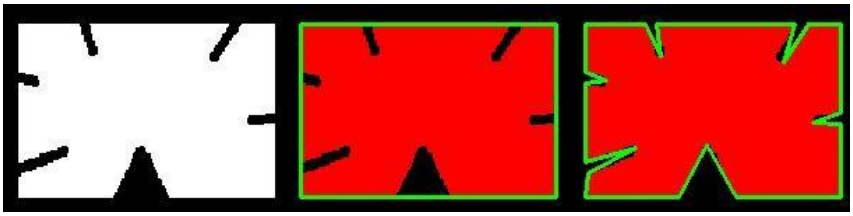


Figura 99 – Aproximação de contorno[55]

No âmbito do protótipo estabeleceu-se um limite de 5%, que poderá ser mutado com base nos resultados obtidos em fase de avaliação.

Validação de Forma

Após obtenção do resultado da curva de aproximação, é agora necessário submetâ-la um processo de validação de forma. São definidas restrições relativamente a:

- Área mínima aceitável para o contorno, valor que é definido com base nos testes empíricos efetuados;
- Total de lados deverá ser igual 3;
- Ângulos internos do contorno onde, no caso do triângulo, é aplicado um limite mínimo e máximo empírico a cada ângulo interno, com base na geometria do triângulo equilátero, que é formado por 3 ângulos de valor 60°.

[54] Exemplo retirado de https://docs.opencv.org/3.1.0/dd/d49/tutorial_py_contour_features.html

[55] Imagem retirada de https://docs.opencv.org/3.1.0/dd/d49/tutorial_py_contour_features.html

Em caso de sucesso, é sinalizada a região do contorno através de um retângulo, que permite formular uma nova região de interesse, a passar para o processo de normalização.

5.3.3.4 Normalização

Devido às transformações realizadas involuntariamente sobre um sinal capturado, quer seja pelo plano ortogonal de captura não estar de acordo com a orientação do sinal, quer pelo facto de ter sido modificado fisicamente, a forma do sinal poderá comprometer o processo de reconhecimento e torná-lo menos eficiente.

Desta forma, para o caso do triângulo, pretende-se que a ROI seja submetida a uma transformação *Afim* (Figura 100) por forma a colocar o possível sinal na posição frontal, e escalado para o tamanho do *template* a comparar (todos os *templates* assumem o mesmo tamanho).

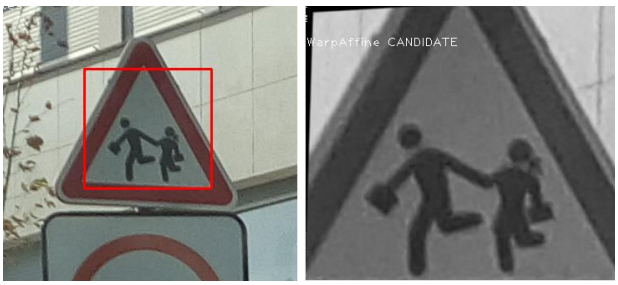


Figura 100 – Transformação Afim de acordo com o tamanho do *template*

As instruções *EmguCV* para realizar estas transformações estão presentes no Código 14. Em primeiro lugar é necessário obter a matriz de transformação através da instrução *GetAffineTransform,* esta função recebe dois argumentos: um vetor de pontos (3 vértices do triângulo) para os quais se pretende realizar a transformação, e um vetor de pontos representativo dos 3 vértices presentes na imagem. Após a geração da matriz de transformação, a mesma é consumada através da instrução *WarpAffine.*

```
…
Mat trnsfMat = CvInvoke.GetAffineTransform(newpts, oldppts);
      CvInvoke.WarpAffine(ROIs[i], ROIs[i], trnsfMat, ModelSize);
…
```

Código 14 – Transformação afim em EmguCV

5.3.3.5 Reconhecimento

Finalizado o processo de Deteção, dele resulta o *input* para a fase de reconhecimento da sinalização de trânsito. É nesta fase em que cada ROI é validada e classificada como sendo representação de um sinal de trânsito.

Em traços gerais, é realizada uma comparação das características presentes em cada ROI com as características presentes em cada *template* (*Feature Matching).*

Os *templates* passíveis de comparação são aqueles que estiverem mapeados pelas cores e formas, que foram detetadas na etapa de deteção.

Características de templates pré-computadas

No momento da comparação de características, por forma a acelerar o tempo de reconhecimento, o vetor de pontos de interesse e os respetivos descritores já são calculados para cada *template.*

Essa pré-computação pretende a extração de apenas os pontos de interesse relativos às formas presentes no interior do sinal. Para cada imagem é realizada conversão para escala de cinzentos (Figura 101 – Pré-processamento e computação de características de *template*). Esta abordagem é realizada também para cada candidato identificado, para que a comparação seja maximizada. O processo de deteção e extração de características é descrito na secção seguinte.

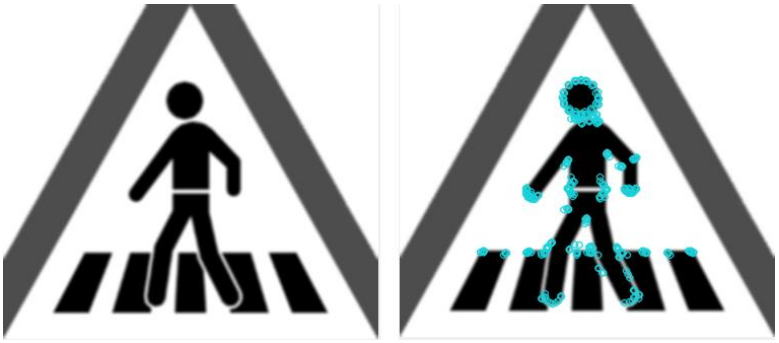


Figura 101 – Pré-processamento e computação de características de *template*

5.3.3.6 Features Matching

Para cada ROI são processados os respetivos pontos de interesse, utilizando para isso o algoritmo de deteção de pontos de interesse (*feature detection)* ORB.

Após a deteção desses *keypoints* para cada *ROI,* é necessário realizar a extração dos detalhes de cada um desses pontos. Essa operação é realizada pelo algoritmo do tipo *feature extractor*, que permite representar sobre *descriptors,* as características de cada *keypoint* detetado. O algoritmo adotado para esse efeito é o ORB (*Oriented FAST and Rotated BRIEF)*.

Tanto os algoritmos de *feature detection,* como os algoritmos de *feature extractor* escolhidos, são suportados e estão disponíveis na biblioteca EmguCV. A sua utilização é bastante simples, através do método *DetectRaw* é possível realizar a deteção de *keypoints* e o método *Compute* permite realizar a respetiva extração dos *descriptors*.

```
…
_orbDetector.DetectRaw(ROIs[i], _roiKeypoint);
```

```
_orbDetector.Compute(ROIs[i], _roiKeypoint, _roiDescriptor);
…
```

Código 15 – Deteção e extração de características em EmguCV

À semelhança do que foi realizado para cada *template*, é, para cada candidato, comparada somente a forma presente no interior do triângulo em formato binário.

Os descritores do candidato são comparados com os descritores de cada *template*, através da utilização do algoritmo de *matching* implementado pela classe *BFMatcher*, denominado *KnnMatch*. O resultado da comparação é um vetor de correspondências.

Para comparação foram definidos dois tamanhos de templates que são escolhidos de acordo com o tamanho mais próximo do tamanho do candidato.

É aplicada uma regra de proporção de quantidade de características em comparação, que exclui de comparação alguns modelos que diferem significativamente do total de características do candidato.

BFMatcher

A classe *BFMatcher ou Brute-Force Matcher* permite percorrer cada característica presente num descritor e compará-lo com todas as características de outro, com base em cálculo de distâncias. De acordo com a compatibilidade com o algoritmo ORB, esta comparação poderá ser realizada utilizando o algoritmo *cross checking*, permitindo gerar resultados realmente consistentes (Figura 102).

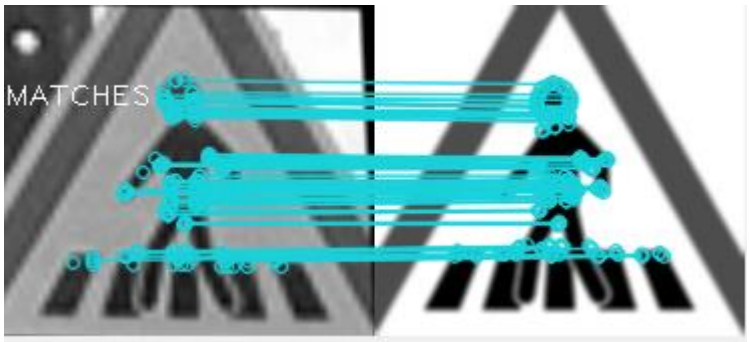


Figura 102 – Matching de características

Filtragem de boas semelhanças

Computar as semelhanças não é suficiente, uma vez que poderão existir nestas semelhanças com falsos positivos. Para filtragem desses resultados são adoptados os seguintes métodos:

- *Cross Checking* – Permite no processo de comparação de características entre o modelo e o candidato, encontrar as melhores semelhanças em (A, B) = (B,A). Retornando para um ponto de interesse do candidato o melhor *match*.

- *Ratio Test* – É um método proposto por com (Lowe, 2004) e permite verificar para dois matches de um determinado ponto de interesse se os dois são o suficientemente semelhantes para serem excluídos.

É realizada também uma, verificação de distância dos pontos de cada match , por exemplo, 1 pixel posicionado em p1(x,y) = p1(10, 20) no *template* poderá fazer correspondência com um ponto p1(x,y) = p1(10, 200), no entanto a sua relação de distância indica que esta não será uma boa correspondência, já que, no eixo das ordenadas, existe uma grande diferença de posição 20 vs 200.

Com base nisto, é aplicado, no processo de reconhecimento, uma filtragem de correspondências através de um teste de relação da posição dos pontos de interesse. Esta abordagem está presente no Código 16, e passa por percorrer todos os *matches* encontrados e verificar para cada um deles, quais os pontos de interesse (candidato e modelo) e qual a relação de distância na região de interesse normalizada. Se esta não passar o limite, é aceite como uma boa correspondência. Definem-se também restrições quanto à posição em Y de cada um dos pontos interesse.

```
double thresholdDist = 0.25 * Math.Sqrt((double)(ROIs[i].Size.Height *
ROIs[i].Size.Height + ROIs[i].Size.Width * ROIs[i].Size.Width));
VectorOfVectorOfDMatch goodMatches = new VectorOfVectorOfDMatch();
for (int k = 0; k < matches.Size; ++k)
{
    for (int k2 = 0; k2 < matches[k].Size; k2++)
    {
   System.Drawing.PointF                          from                          =
_observeredKeypoint[matches[k][k2].QueryIdx].Point;
System.Drawing.PointF to = ModelKeypoints[matches[k][k2].TrainIdx].Point;
double dist = Math.Sqrt((from.X - to.X) * (from.X - to.X) + (from.Y - to.Y)
* (from.Y - to.Y));
if (dist < thresholdDist && Math.Abs(from.Y - to.Y) < 5)
{
goodMatches.Push(matches[k]);
                                        k2 = matches[k].Size;
         }
     }
}
```

Código 16 – Abordagem para filtragem de boas correspondências segundo teste de relação de distâncias

Depois de determinadas as semelhanças de boa qualidade, é realizada a comparação do número de *matches* detetados com o número mínimo de aceitação definido para o processo. Se o resultado for positivo, significa que o modelo que está a ser comparado corresponde ao candidato detetado, e o sistema despoletará a preparação da informação a disponibilizar para a camada superior.

O processo de preparação permite recolher informações sobre o sinal que é representado pelo modelo que passou na comparação de semelhanças, como detalhes gráficos a apresentar para a UI, mensagens de aviso e outras informações textuais.

Conclusão

O processo de deteção e de reconhecimento descrito nas secções anteriores procurou em primeiro lugar, determinar todas as circunstâncias inerentes à identificação de sinalização com o formato triangular.

Pela experiência adquirida, verifica-se que, para extensão da identificação aos restantes sinais, é necessário realizar o mesmo processo de análise e desenvolvimento, uma vez que, a forma, a cor e outras características necessitam de implementação de outros comportamentos.

De forma a apresentar o resultado final e transversal a cada uma das etapas, está presente na Figura 103, uma imagem capturada e submetida a cada um dos processos para identificação de sinalização de trânsito.

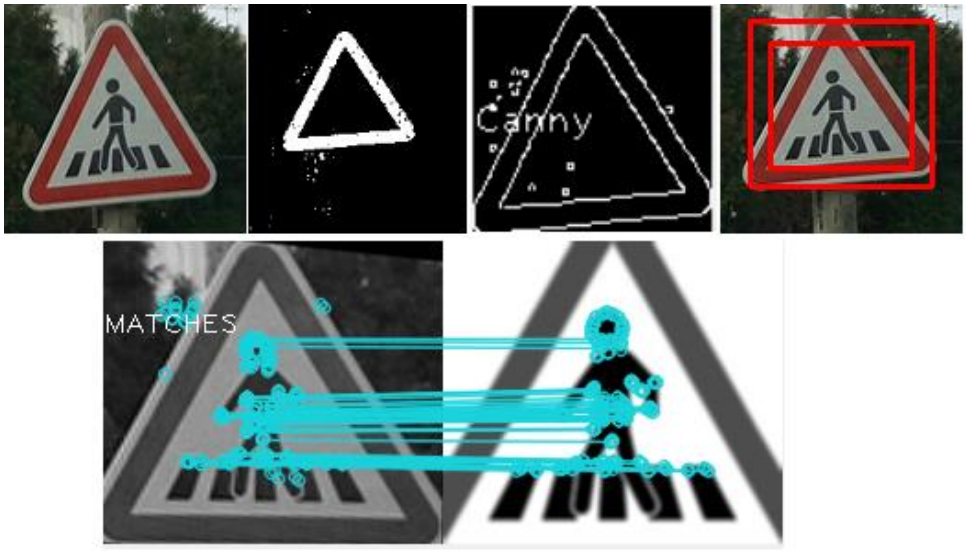


Figura 103 – Resultado das várias fases: Imagem original> Segmentação baseada em Cor> Segmentação baseada em Forma> Normalização> Reconhecimento.

5.4 Ameaças

O reconhecimento de sinalização de trânsito vertical através do processamento e computação de imagem acarreta um conjunto de dificuldades no processo de deteção. Após primeira análise a diferentes imagens capturadas, foi estabelecido um conjunto de ameaças que poderão afetar a eficácia do protótipo, agrupadas em três tipos, Condições de Sinalização, Condições de Luminosidade e Condições de Condução (Figura 104).

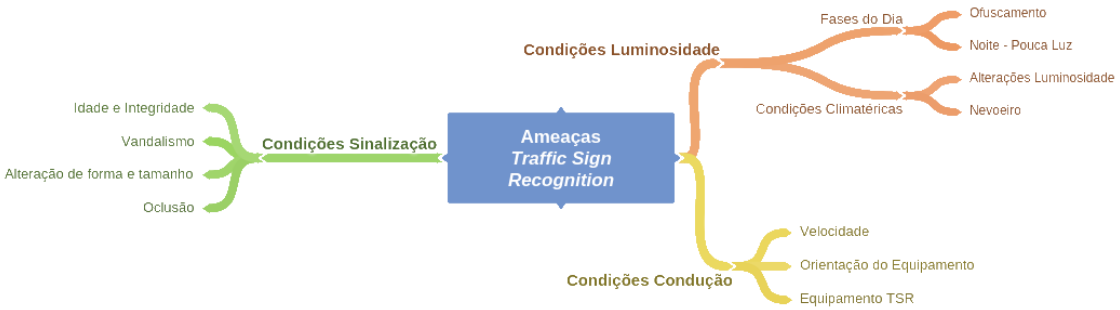


Figura 104 – Ameaças *Traffic Sign Recognition*

Relativamente às Condições da Sinalização, existe necessidade de prever os seguintes fenómenos:

- Idade e Integridade da sinalização – O tempo de exposição de um sinal vertical de trânsito a fatores climatéricos poderá ser fundamental para a manutenção das suas características principais, nomeadamente a cor dos elementos do sinal (Figura 105). Devido à falta de manutenção e substituição dos mesmos, a deteção de um sinal numa captura de imagem poderá falhar, por falta de atributos que satisfaçam os critérios do modelo *TSR*. Para minimizar o impacto destes fatores no processo de deteção e reconhecimento, são realizadas transformações de cor, luz e forma nas imagens capturadas, para torná-las mais significativas;

Figura 105 – Sinal (A23 – Trabalhos na via) deteriorado

- Vandalismo – O facto de a sinalização ser fisicamente alcançável pelos cidadãos, faz com que seja sujeita a atos de vandalismo e ações de publicidade, que alteram a visibilidade da mesma (Figura 106). Do ponto de vista computacional o problema é maior, pois deverão ser previstos casos “fora da regra”;

Figura 106 – Sinal (C1 – Sentido Proibido) vandalizado

- Alteração da forma e tamanho – A sinalização é também condicionada pela sinistralidade automóvel e outro tipo de acidentes que poderão provocar uma alteração de forma e tamanho (Figura 107);

Figura 107 – Sinal (A2a – Lomba) acidentado

- Oclusão – Um dos fenómenos mais difíceis de controlar pela computação e processamento de imagem é a oclusão da sinalização. A presença de objetos, como: outro tipo de sinalização; árvores; cartazes; e outros tipos de condicionantes, na via pública, poderá comprometer a visualização na íntegra da sinalização (Figura 108).

Figura 108 – Oclusão do sinal (G1 - Zona de estacionamento autorizado)

Outro conjunto de condições que foram alvo de cuidado no desenvolvimento da solução relacionam-se com a luminosidade presente na imagem capturada. A luminosidade é influenciada por diversos fatores, tais como:

- Fases do Dia – Dependendo da fase do dia em que a captura é realizada, o nível de luminosidade varia substancialmente. O caso mais díspar é a diferença de luminosidade entre a presença do sol (Figura 109) e a noite (Figura 110);

Figura 109 – Foto capturada durante dia

Figura 110 – Foto capturada após por do sol

- Ofuscamento – Durante o nascer do sol e o pôr-do-sol existem períodos em que a luz poderá provocar o ofuscamento (excesso de brilho capturado), resultando numa imagem escurecida em alguns pontos, sendo por vezes impossível a deteção da sinalização (Figura 111). Este fenómeno poderá acontecer com diversas fontes de luz, sejam elas diretas, como a luz emitida pelos faróis dos carros, ou indiretas, como os reflexos de luz em materiais espelhantes;

Figura 111 – Ofuscamento provocado pela luz do Sol

- Sombra – A falta de incidência de luz sobre a sinalização de trânsito provocada pela presença de objetos no cenário de captura, poderá dificultar e impossibilitar o processo de deteção da mesma.

Foram contemplados também, no desenvolvimento desta solução, os fatores relacionados com a condução automóvel, a velocidade do veículo, a orientação da captura e as características do equipamento utilizado para a captura.

A velocidade média do veículo poderá afetar o rácio de sinalização detetada. O intervalo de tempo disponível para captura e a focalização da mesma, uma vez que poderá não ser suficiente com velocidades mais altas.

A orientação do equipamento face ao plano de orientação da sinalização poderá afetar a correta leitura da posição da sinalização, resultando em transformações de perspetiva.

As características do equipamento, como a resolução da câmara fotográfica, os *fps* disponíveis, a capacidade de processamento *CPU,* e *GPU,* poderão ou não potenciar os resultados da solução.

5.5 Conclusão

De acordo com fase de desenvolvimento do protótipo Virtual Driver, são evidenciadas algumas conclusões sobre o trabalho realizado.

A utilização de uma metodologia *Inversion of Control* permite uma grande agilidade no processo de desenvolvimento, permitindo que o módulo de reconhecimento posso ser desacoplado e testado fora da solução principal.

A experiência no desenvolvimento da interface com o utilizador, através da utilização de Xamarin Forms, foi uma agradável surpresa, pela simplicidade na construção dos layouts e realização do *data binding.* No entanto, destaca-se como ponto negativo, o tempo de compilação e instalação no dispositivo para realizações de depuração.

Uma das principais dificuldades encontradas, esteve relacionada com a capacidade de o sistema ser responsivo na captura, pré-visualização e processamento de imagem em tempo real. Através da introdução da API Android, Camera 2, foi possível desenvolver um circuito capaz de processar pedidos em *background* de uma forma robusta.

Destaca-se também o longo processo de afinação da solução, em cada uma das fases da metodologia de reconhecimento de sinalização de trânsito, em que eram realizadas experiências em diferentes cenários. A biblioteca EmguCV apresentou-se como uma ferramenta poderosa para o processamento de imagem, abstraindo funções mais complexas do OpenCV.

Destaca-se que a *framework* EmguCV permite executar o mesmo código desenvolvido tanto em CPU como a GPU, verificando para isso se o dispositivo suporta OpenCL.

6 Avaliação

Neste capítulo, pretende-se descrever de forma sucinta, para o problema e objetivos definidos, quais foram as grandezas, experiências e resultados que representaram a base para avaliação do protótipo da solução *Virtual Driver*.

6.1 Introdução

De acordo com a magnitude do sistema Virtual Driver, o mesmo poderá ser dividido em duas partes, a interface com utilizador final e o motor de reconhecimento e a catalogação de sinalização.

Para avaliação do protótipo, foi dado foco na avaliação das grandezas que envolvem os processos de deteção e reconhecimento de sinalização, evidenciando resultados obtidos em diversos cenários (simulados ou reais) de acordo com os algoritmos adotados.

A avaliação da interface com utilizador foi realizada de acordo com o feedback de um conjunto amostra de utilizadores que foram submetidos à experiência Virtual Driver.

Nos subcapítulos seguintes, são detalhados: o problema e objetivos, as grandezas utilizadas para avaliação, experiências e resultados obtidos, testes de hipóteses e metodologia de avaliação.

6.2 Descrição Clara e Sucinta do Problema e Objetivos

Os sistemas avançados de assistência ao condutor (ADAS) assumem hoje um papel importante na redução dos erros Humanos ao volante, os mesmos, permitem alertar o condutor e atuar sobre o automóvel de forma a prevenir acidentes rodoviários.

A disponibilização destes sistemas ainda não proliferou, principalmente devido ao preço de aquisição, restrições do automóvel e restrições tecnológicas.

Pretende-se que o *Virtual Driver*, conceba uma aplicação móvel evolutiva capaz de ser executada em dispositivos de baixo custo, como o *smartphone, tablet* e outros microcomputadores, garantindo a independência do ano, marca, modelo, gama e tecnologia do automóvel. Numa primeira fase o sistema será dotado da componente de reconhecimento de sinalização de trânsito vertical denominada *Traffic Sign Recognition*, que esteja comprometida com uma boa taxa de identificação, garantindo assim a qualidade da solução que pretende responder ao problema.
Serão adotados alguns algoritmos na fase de deteção e reconhecimento com base na investigação realizada.
Pretende-se fazer um comparativo dos resultados obtidos, detalhando cada cenário de teste.

6.3 Grandezas Para Avaliação do Trabalho

Para avaliação da solução desenvolvida são evidenciadas diferentes dimensões, como a eficácia do processo de deteção/reconhecimento e a eficiência de cada processo, nomeadamente o tempo de execução.

Em primeiro lugar pretende-se avaliar a precisão da solução através do apuramento do rácio do número de sinais identificados face aos sinais existentes num determinado cenário, abordagem seguida por (Jatmiko and Prihatmanto, 2016). Pretende-se aplicar este cálculo a um conjunto de cenários (Tabela 23) de forma a obter outro nível de significância, realizando a média dos valores. Esses cenários deverão ser avaliados sobre o mesmo conjunto de condições, no caso da velocidade do veículo, deverá ser a mesma por uma questão de equidade da avaliação. O número de sinais a reconhecer dependerá da gama de sinais que a solução em forma de protótipo vai conseguir responder.

Cenário Real ou Simulado	Condições	Nº Sinais Possíveis	Nº Sinais Identificados	Nº Falsos Positivos
Cenário 1	Velocidade, Clima	A	A1	X
Cenário 2		B	B1	Y
Cenário 3		C	C1	Z
Cenário 4		D	D1	W

Tabela 23 – Exemplo de cenários possíveis de avaliação

A formula da *Accuracy* será dada por:

$$Accuracy = \frac{\sum_n \left\{\left(\frac{A}{A1}\right) + \left(\frac{B}{B1}\right) + \left(\frac{C}{C1}\right) + \left(\frac{D}{D1}\right)\right\}}{n} * 100$$

Importa referir a necessidade de quantificar das falsas identificações (falsos positivos) como medida para melhoria.

É realizada também, a avaliação da dimensão Tempo em cada uma das fases de execução do sistema. Pretende-se determinar o tempo de cada fase de acordo com pressupostos base, como

por exemplo a resolução de captura (Farhat *et al.*, 2016) e os algoritmos de deteção e descrição de descrição de características.

Resolução de captura	*Features Detector*	*Features Descriptor*	Deteção (ms)	Reconhecimento (ms)	Informação (ms)
1280 × 720 pixels	ORB	ORB	0.15	0.30	0.06
1280 × 720 pixels	FAST	BRIEF	0.10	0.25	0.06
…	…	…	…	…	…

Tabela 24 – Exemplo de avaliação de tempos de execução do processo *TSR*

Com base nestas grandezas de avaliação, foram realizadas algumas experiências e detalhados os seus resultados, que serão explicados no próximo subcapítulo.

6.4 Experiências e Resultados Obtidos

O protótipo Virtual Driver foi experienciado em dois tipos de cenários distintos, o cenário simulado, que se realizou em ambiente interior e o cenário real, em que o sistema é testado na estrada.

Descreve-se neste subcapítulo, que equipamento foi utilizado para testes, qual a configuração do sistema, quais os cenários, dificuldades e resultados obtidos.

6.4.1 Equipamento

Para realização dos testes foi utilizado o *smartphone* Letv 1S (Figura 112).

Figura 112 – *Smartphone* Letv 1S usado como teste[56]

Destacam-se algumas características (informação GSMARENA[57]) de acordo com os requisitos do sistema Virtual Driver:

[56] Imagem retirada de: https://www.gsmarena.com/leeco_le_1s-pictures-8055.php

[57] Informação presente em: https://www.gsmarena.com/leeco_le_1s-8055.phpimagem

- Resolução de ecrã de 1920x1080 pixels;
- Sistema Operativo Android 6.0;
- Chipset Mediatek MT6795 Helio X10;
- CPU Octa-core 2.2 GHz Cortex-A53;
- GPU PowerVR G6200;
- 3G RAM;
- Camera primária de 13 MegaPixels com autofócus.

O lançamento do equipamento é datado de Fevereiro de 2016, tendo um custo atual aproximado de 130€[58].

6.4.2 Configuração do Sistema

O processo de deteção e reconhecimento tem como base um conjunto de configurações que permitem definir limites que estipulam qual a aceitação de um determinado objeto, presente na imagem, como sendo um sinal de trânsito. Na Tabela 25, estão descritas as configurações base que serviram para a realização das experiências.

Parâmetro	Valores
Percentagem mínima de semelhanças (*good matches)*	10%
Resolução	1280x720 pixels
Percentagem Ratio Test	0.75
Limites de Cor para vermelho HSV	H = [0-20][130-179] S=[90-255] V=[60-255]
Área mínima de Blob segmentado	1000 pixels
Área máxima de Blob segmentado	100000 pixels
Percentagem máxima de diferença binária entre o candidato e o modelo	80%
Área mínima para aceitação de um contorno como forma	250 pixels
Percentagem máxima de diferença de distância entre os pontos de interesse do modelo com os do candidato	15%
FPS	5-30

Tabela 25 – Configuração do protótipo Virtual Driver

Os parâmetros relacionados com a captura restringem-se ao que está disponível no equipamento. Por defeito, foram utilizados os mecanismos de balanceamento de brancos e exposição automáticos, para a taxa de FPS de 5-30. Relativamente à focagem da câmara, é

[58] Informação presente em: https://www.ebay.com/

utilizado o mecanismo "*ContinuousPicture*"[59] que permite de forma agressiva e cíclica realizar focagem enquanto o sistema está a capturar.

6.4.3 Experiência em Cenário Simulado

Representou a primeira fase de experiência e testes ao protótipo Virtual Driver, e foi realizada em ambiente controlado. Foram montados alguns circuitos no interior do Departamento de Informática do Instituto Superior de Engenharia do Porto (Figura 113), onde se afixaram exemplares impressos dos sinais, de forma espaçada ao longo dos corredores.

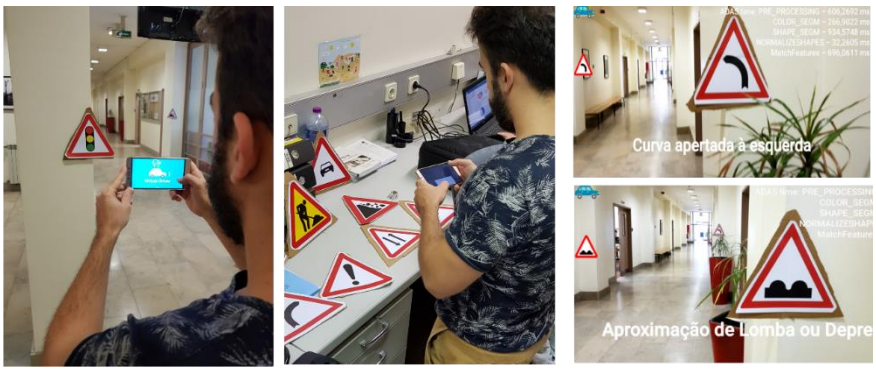


Figura 113 - Deteção em circuitos simulado – ISEP.

Esta etapa permitiu treinar e afinar os parâmetros de configuração do sistema, preparando-o para realização de testes no exterior.

A intensidade e variação da luz dentro de um cenário controlado, propicia a obtenção de melhores resultados, é, no entanto, imprescindível que o sistema seja submetido a este teor de experiências antes de avançar para testes reais.

6.4.4 Experiência em Cenário Real

Representa o cenário real de utilização do sistema, através da sua aplicação durante o exercício de condução em estradas portuguesas (Figura 114).

Foram realizados um conjunto de percursos por estradas nacionais e autoestradas, observando-se os resultados de identificação do Virtual Driver. Em cada sinalização presente, foi verificado se a mesma foi detetada e/ou reconhecida, e qual o tempo para cada um dos processos.

---

[59] Informação presente em: https://developer.android.com/reference/android/hardware/Camera.Parameters.html#FOCUS_MODE_CONTINUOUS_PICTURE

Figura 114 – Exemplificação da utilização do Virtual Driver em tempo real

6.4.5 Dificuldades

Devido à problemática relacionada com a captura de imagem em tempo real durante a condução, existem fatores que estão diretamente ligados à qualidade da imagem capturada e por consequente à taxa de reconhecimento. Identificam-se:

- Velocidade da marcha – Influencia no número de *frames* com qualidade disponível para análise;
- Fenómenos causados pela alteração e visibilidade da fonte de luz – Sombras, ofuscamento e falta de luz resultam em *frames* de fraca qualidade;
- Captura e processamento em tempo útil – A taxa de captura da camara poderá ser elevada, mas o sucesso do reconhecimento dependerá do tempo de resposta do motor, porque numa taxa de 30 FPS poderão ser "desperdiçados" mais de 80% dos *frames*;
- Resolução de Captura – O tamanho da resolução de captura está diretamente relacionado com o tempo de resposta do processo. Para melhores resultados, definiu-se como resolução 1280x720 pixels.

Os resultados obtidos minimizam relativamente aqueles que foram observados num cenário controlado.

6.4.6 Resultados

Nesta seção descrevem-se os resultados obtidos através da experiência do Virtual Driver num cenário real, dando foco ao processo de deteção e reconhecimento.

Como pressupostos para os testes, definiu-se que as experiências deveriam ser realizadas durante o dia, nomeadamente com céu limpo.

A aplicação foi submetida a validação de um conjunto de 51 sinais, permitindo calcular a sua eficácia. Ao longo do processo foram recolhidas estatísticas relacionadas com os tempos de processamento, e foram arquivadas as imagens relacionadas com o processo de identificação.

A metodologia desenvolvida nesta dissertação, permitiu que em 51 sinais de trânsito, fossem detetados 44 potenciais candidatos, e reconhecidos 38, através do algoritmo detetor e extrator ORB, como sendo os respetivos sinais de trânsito. Destaca-se também a geração de 3 falsos positivos (Tabela 26).

Algoritmo Seleção	Algoritmo Extração	Sinais Detectados	Sinais Reconhecidos	Falsos Positivos
ORB	ORB	44	38	3

Tabela 26 – Dados estatísticos de experiência real Virtual Driver.

Com base nos resultados obtidos foi possível verificar uma eficácia máxima de 75% (Figura 115).

Figura 115 – Taxa de reconhecimento Virtual Driver

Detalhando os tempos médios relacionados com cada fase da metodologia de deteção e reconhecimento de sinalização de trânsito (Figura 116), verifica-se que o processo de segmentação por cor apresenta um maior tempo de processamento, cerca de 105,89 milissegundos, porque é da sua responsabilidade fazer um rastreio da imagem, gerando Blobs que permitirão acelerar os restantes processos.

Totalizando todos os tempos, verifica-se um tempo médio de processamento sobre CPU de 224,7 milissegundos.

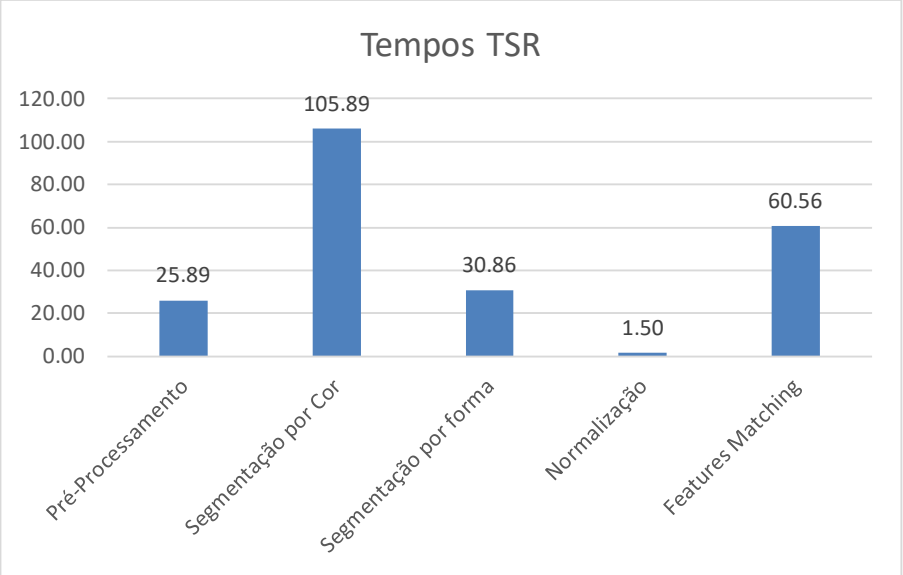


Figura 116 – Tempos médios de cada etapa de metodologia de reconhecimento.

As imagens recolhidas apresentam, no seu conteúdo, diferentes perspetivas da sinalização, diferentes tamanhos, diferentes iluminações e, em alguns casos, fenómenos de oclusão.

Identificaram-se como principais falhas no processo de deteção de candidatos os seguintes fenómenos:

- Oclusões provocados por objetos, impossibilitando o seu total reconhecimento(Figura 117);

Figura 117 – Fenómeno de oclusão

- Sinais desgastados que dificultam o processo de segmentação baseado em cor (Figura 118);

Figura 118 – Desgaste da sinalização

- Sinalização presente na imagem em tamanho reduzido. A resolução de captura está diretamente ligada à capacidade para reconhecimento de sinalização mais longínqua da câmara (Figura 119).

Figura 119 – Tamanho do sinal no frame

6.4.6.1 Comparação de algoritmos Features Matching

De acordo com a revisão da literatura descrita nesta dissertação, pretende-se evidenciar aqui o resultado dos testes, utilizando diferentes algoritmos de deteção e extração de características na imagem. Desta forma, foram realizados testes através das imagens registadas na experiência descrita na seção anterior.

Foram submetidos a teste, os algoritmos de deteção ORB, SURF e FAST e de extração ORB, BRIEF, BRISK e FREAK. Verificaram-se que algumas combinações de algoritmos não foram satisfatórias, porque, no processo inicial de deteção e processamento de características dos modelos em comparação, não foram capazes de detetarem um número satisfatório de *features.* A utilização dos algoritmos foi realizada com base nas parametrizações e valores por defeito inerentes.

Algoritmo Seleção	Algoritmo Extração	Sinais Detectados	Sinais Reconhecidos	Falsos Positivos
ORB	ORB	44	38	3
ORB	BRIEF	44	27	10
SURF	BRIEF	44	27	15
SURF	ORB	44	25	19
FAST	BRIEF	44	13	19
FAST	ORB	44	14	24
FAST	FREAK	44	10	14
FAST	BRISK	44	4	3

Tabela 27 – Resultados da deteção e reconhecimento para amostra de 50 imagens.

Analisando os resultados relativamente ao desempenho de cada algoritmo, verifica-se que em termos de eficácia de reconhecimento (Figura 120), existe uma diferença significativa (25%) do algoritmo ORB para as três combinações mais próximas (ORB/BRIEF, SURF/BRIEF e SURF/ORB). As restantes combinações apresentaram desempenhos muito abaixo do aceitável para a sua aceitação em trabalho futuro.

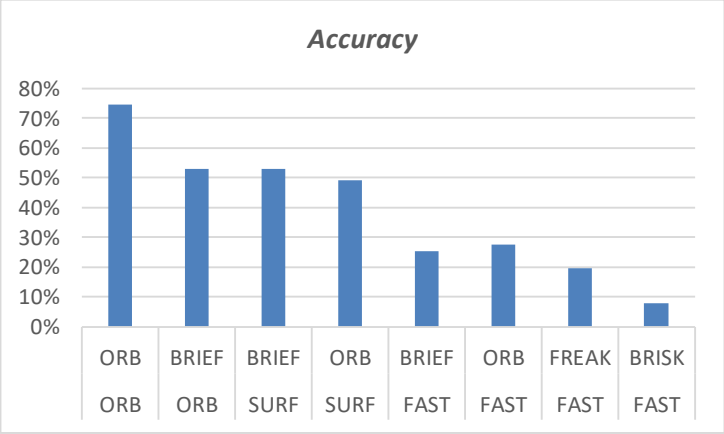


Figura 120 – Eficácia de Reconhecimento por combinação de algoritmos Features Matching

Analisando as 4 combinações mais eficazes na experiência anterior, verifica-se que o tempo que é dedicado à deteção, extração e *matching* de características para cada região de interesse é próximo, destacando a utilização SURF+BRIEF como a mais rápida (15,50 milissegundos).

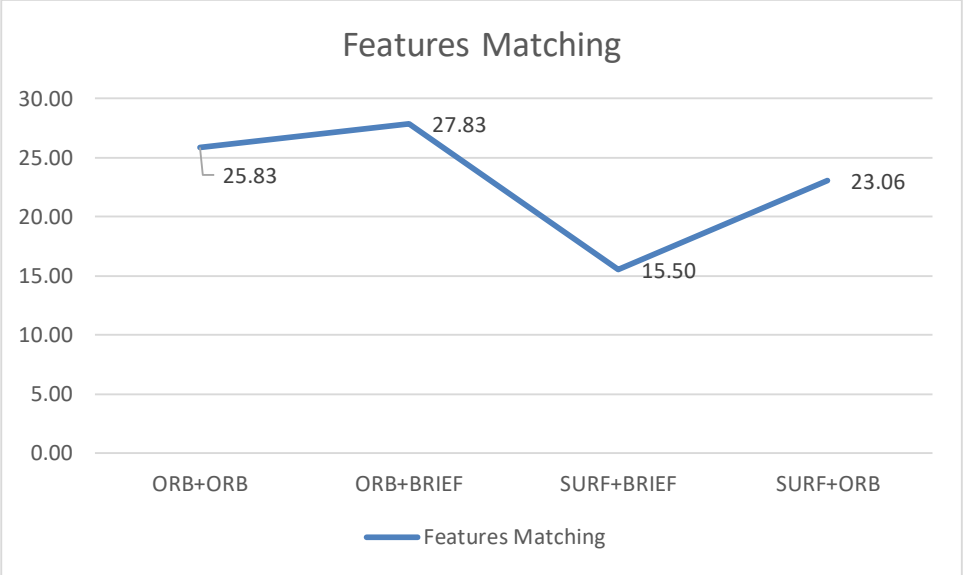


Figura 121 – Tempos Features Matching

6.5 Conclusões

Após análise dos resultados obtidos nas experiências simuladas e reais do Virtual Driver, verifica-se uma diferença significativa no reconhecimento da sinalização num cenário controlado e um cenário real, relacionada com dois motivos principais: o sinal a reconhecer é o próprio modelo que está computado no Virtual Driver e a luminosidade é controlada.

De forma a perceber realmente até onde o protótipo Virtual Driver poderia ir, o mesmo foi submetido a um cenário real para reconhecimento de 51 sinais em diferentes condições físicas, diferentes posições na via (incluindo oclusões) e o mesmo apresentou uma taxa de reconhecimento de 75% relativamente a uma amostra inicial. A eficácia demonstrada está abaixo daquilo que se deseja (>=95%). No entanto, retirando os casos de oclusões, ofuscamento e desgaste dos sinais é possível verificar uma taxa de reconhecimento de aproximadamente 83%. Aliado a isso, verifica-se que a resolução escolhida para efeitos de teste, apesar de compatível com a maioria dos dispositivos, não é suficiente para captação a distâncias maiores, que necessitam de maior detalhe (mais resolução).

Relativamente aos tempos de cada etapa da metodologia, verifica-se um tempo total de 224,7 milissegundos em execução sobre o CPU. Com o aumento da resolução para 2080x1560 pixels o tempo de processamento aumenta cerca de 50%. De forma a garantir a compatibilidade com resoluções maiores, deverá ser tido mecanismos de computação heterogénea.

7 Conclusões e Trabalho Futuro

Este capitulo pretende descrever que conclusões foram possíveis de determinar numa primeira iteração da solução evolutiva Virtual Driver. São resumidas as principais conclusões, quais os objetivos alcançados e qual o trabalho futuro.

7.1 Principais Conclusões

De acordo com os objetivos principais desta dissertação, foi realizada uma extensa investigação sobre sistemas inteligentes de transporte atualmente existentes no mercado, particularmente o subconjunto, sistemas avançados de assistência ao condutor e todos os conceitos inerentes a esta temática. Foram investigadas também, diferentes metodologias e abordagens para a deteção e reconhecimento de sinalização de trânsito, bibliotecas de processamento de imagem e bibliotecas de desenvolvimento para plataforma móvel.

Relativamente ao estado de arte relacionado com as soluções ADAS atualmente existentes no mercado, verifica-se uma grande taxa de adopção dos principais fabricantes de automóveis a este tipo de sistemas. As soluções oferecidas pelos fabricantes, apresentam-se como soluções robustas e bastante fiáveis, no entanto verifica-se que relativamente ao reconhecimento de sinalização de trânsito vertical, apresentam limitação à gama de sinalização disponível de deteção. Aliado a esta fator, verifica-se que a disponibilização destes sistemas ainda está intrínseca à aquisição de um determinado automóvel ou extra, limitando o alcance a automóveis mais datados. Verifica-se também que a nível de soluções disponíveis ao utilizador final por plataformas comuns como *Google Play*, *App Store*, tem pouca aplicabilidade na realidade portuguesa.

Através do desenvolvimento do protótipo Virtual Driver, foi possível verificar a aplicabilidade de algumas metodologias de deteção e reconhecimento de sinalização vertical de trânsito investigadas no processo de estado da arte.

A utilização da metodologia de deteção baseada em segmentação por cor em conjunto por segmentação por forma, apresenta-se como uma metodologia bastante simples e poderosa

para extração e filtragem de regiões de interesse numa imagem. No entanto, este tipo de metodologia só foi validado em horários de sol, pelo que o reconhecimento com baixa luz poderá causar o insucesso do método de deteção por cor. A componente de deteção de forma apesar de ter apresentado resultados bastante positivos, é, no entanto, sensível a oclusões ocorridas sobre a sinalização.

Relativamente ao processo de reconhecimento, a utilização dos algoritmos *features matching*, é uma alternativa viável para o reconhecimento da forma presente no interior de um sinal. No entanto, a variação do aspecto e forma de apresentação de cada sinalização presente nas estradas em conjunto com a utilização de baixas resoluções, despoleta a geração de alguns outliers no processo de deteção de *features* e por consequente más comparações. Foi verificado que o algoritmo com melhor desempenho na deteção e extração de características de imagem é o ORB. Foi registada uma taxa de reconhecimento de cerca de 75%. Para garantir o cumprimento com a qualidade do sistema, pretende-se a taxa se eleve para valores acima dos 95%, para isso em trabalho futuro deverão ser tratados alguns fenómenos como oclusões e cenários de baixa luz.

A utilização da *framework* Xamarin, permitiu acelerar o processo de desenvolvimento da interface com o utilizador com alta taxa de abstração para a linguagem nativa. A disponibilização das principais bibliotecas nativas sobre linguagem C#, permite ser uma grande vantagem, porque dessa forma não existe necessidade de aprendizagem de cada linguagem respectiva. Verifica-se que em cenários que seja necessário realizar a costumização de um controlo nativo, como foi o caso da utilização da pré-visualização da camara, a taxa de reutilização baixa significativamente para esse caso.

A total compatibilidade do Xamarin com a biblioteca de processamento imagem EmguCV, permitiu desenhar uma solução completamente integrada e extensível a novos módulos ADAS. EmguCV apresentou-se como uma biblioteca bastante robusta e mais amigável à programação abstraindo em alto nível a utilização de funções OpenCV, disponibilizando assim funções atómicas capazes de facilmente realizar operações em imagem. EmguCV apresenta também como principal vantagem a disponibilização de diversos algoritmos de deteção e extração de características, que facilitou o desenvolvimento de uma solução utilizar diferentes algoritmos com base em configuração.

7.2 Objetivos Alcançados

De acordo com os objetivos traçados para o plano desta dissertação, foi evidenciado um estudo do estado da arte que serviu de apoio ao desenvolvimento do protótipo Virtual Driver.

Relativamente ao processo de investigação, foram atingidos os principais objetivos, tendo sido realizada a investigação sobre os principais conceitos relacionados com sistemas ITS e ADAS, que soluções os fabricantes disponibilizam aos condutores e de que forma. Foram também investigadas metodologias de deteção e reconhecimento que estão presentes na literatura e que algoritmos existem para cada metodologia. Foi realizado uma análise comparativa de

bibliotecas de processamento de imagem e foram estudados as principais plataformas móveis e qual o melhor ambiente de desenvolvimento para multiplataforma.

No processo de investigação, não foi realizada uma análise das as tecnologias de TTS - *Text to Speech* existentes no mercado. O mesmo, deverá ser realizado em trabalho futuro. Visto que o Xamarin facilita a integração com este tipo de tecnologias, foi desenvolvido em protótipo um comportamento, que permite fazer uso da tecnologia TTS instalada e pré-definida no dispositivo.

A fase desenvolvimento permitiu conceber um protótipo capaz de reconhecer um conjunto restrito de sinalização em formato triangular de limites vermelho, representativo da categoria de sinalização de perigo e alguns presentes na categoria de cedência de passagem.

O protótipo foi alvo de experiências e avaliação, permitindo-se concluir uma taxa de deteção e reconhecimento de 75%.

7.3 Limitações e Trabalho Futuro

Durante a fase de desenvolvimento e avaliação do protótipo, verificaram-se algumas limitações que deverão ser alvo de reparo em trabalho futuro.

A utilização da camara do dispositivo móvel para pré-visualização e captura de frames para processamento de imagem em tempo real, necessita de um processo celere, responsivo e permita processar o maior número de frames dentro do intervalo de *FPS* disponível. Como trabalho futuro, pretende-se melhorar o processo de escalonamento do processamento desses frames, recorrendo sempre que possível a multi-tarefa e computação heterógena. A melhoria deste processo estará directamente ligada à melhoria da taxa de reconhecimento.

Para satisfazer o compromisso da qualidade do reconhecimento da solução, deverá ser elevada a taxa de reconhecimento do protótipo, tratando de uma melhor forma os problemas relacionados com a imagem, que pode ser capturada em pouca luz, com fenómenos de oclusão e desgaste da sinalização. Deverão ser avaliados algoritmos de aprendizagem automática que possam acrescentar mais valor neste processo.

Deverá ser alargada a panóplia de sinalização possível de reconhecer. Através da metodologia desenvolvida facilita e acelera o processo de introdução de novas sinalizações a reconhecer.

Será necessário, no futuro, o desenvolvimento de um portal, previsto no design da solução, que permita realizar a configuração central do sistema, através de disponibilização de novas sinalizações a reconhecer e a respectiva computação de características.

A solução Virtual Driver foi desenvolvida com base numa metodologia evolutiva de desenvolvimento de software de forma a responder ao requisito enunciado pelo Laboratório Multimédia do Departamento de Informática do Instituto Superio de Engenharia do Porto, que pretende que a mesma sirva de base para um sistema avançado de assistência ao condutor.

Desta forma, no futuro, pretende-se dotar a solução de sistemas inteligentes como deteção de sinalização horizontal, deteção de peões a atravessarem a estrada, deteção de sonolência, entre outros.

7.4 Considerações finais

O extenso processo de investigação desta dissertação permitiu adquirir uma série de conhecimentos importantes no ramo das tecnologias de visão computacional e desenvolvimento para recursos móveis.

Os conhecimentos de visão computacional adquiridos, permitem definir uma base embrionária para resolução futura de problemas deste teor. No entanto, o Universo de tecnologias de processamento de imagem e metodologias de reconhecimento, é bastante extenso, sendo que por isso deverão ser sempre avaliadas outras alternativas, nomeadamente sistemas baseados em aprendizagem máquina.

Para finalizar, salienta-se o quão gratificante foi a experiência de investigação que teve como base o desenvolvimento do Virtual Driver. Deseja-se que num futuro breve a solução passe a estar disponível para o utilizador final, garantindo assim missão originária.

Referências

Adaptive Vision (2017) 'Template Matching'. Available at: http://docs.adaptive-vision.com/current/studio/machine_vision_guide/TemplateMatching.html (Accessed: 25 February 2017).

AForge.NET (no date) *AForge.NET :: Framework Features*. Available at: http://www.aforgenet.com/framework/features/ (Accessed: 3 March 2017).

Alahi, A., Ortiz, R. and Vandergheynst, P. (2012) 'FREAK: Fast Retina Keypoint'. Available at: https://infoscience.epfl.ch/record/175537/files/2069.pdf (Accessed: 16 October 2017).

Allee, V. (2008) *VALUE NETWORK ANALYSIS: VALUE CONVERSION OF TANGIBLE AND INTANGIBLE ASSETS*, *Rotman Management. Fall2008, p42-47. 6p. 1 Chart.* Available at: http://eds.b.ebscohost.com/eds/detail/detail?sid=b5707778-7e0c-419a-abc7-579d5300adcb%40sessionmgr102&vid=0&hid=114&bdata=JkF1dGhUeXBlPWlwLGNvb2tpZSxzaGliLHVpZCZsYW5nPXB0LWJyJnNpdGU9ZWRzLWxpdmUmc2NvcGU9c2l0ZQ%3D%3D#AN=34229844&db=bth (Accessed: 30 January 2017).

Angrish, H. and Kaur, S. (2013) 'A Survey on Feature Description Techniques', *International Journal of Science and Research (IJSR) ISSN (Online Index Copernicus Value Impact Factor*, 14(5), pp. 2319–7064. Available at: https://www.ijsr.net/archive/v4i5/27041503.pdf (Accessed: 16 October 2017).

Apple (2017) *About the iOS Technologies*. Available at: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html#//apple_ref/doc/uid/TP40007898-CH1-SW1 (Accessed: 4 March 2017).

Auto Pi (2017) *AutoPi - IoT platform for your car, built on the Raspberry Pi*. Available at: https://www.autopi.io/ (Accessed: 7 October 2017).

Baro, X. *et al.* (2009) 'Traffic Sign Recognition Using Evolutionary Adaboost Detection and Forest-ECOC Classification', *IEEE Transactions on Intelligent Transportation Systems*, 10(1), pp. 113–126. doi: 10.1109/TITS.2008.2011702.

Bay, H. *et al.* (2007) 'Speeded-Up Robust Features (SURF)'. doi: 10.1016/j.cviu.2007.09.014.

Bradski, G. *et al.* (2008) *Learning OpenCV*. Edited by M. Loukides. O'Reilly Media, Inc. Available at: http://www.bogotobogo.com/cplusplus/files/OReilly Learning OpenCV.pdf (Accessed: 2 March 2017).

Business Model Fiddle (2017). Available at: https://bmfiddle.com/ (Accessed: 18 February 2017).

Calonder, M. *et al.* (2010) 'BRIEF: Binary Robust Independent Elementary Features'. Available at: https://www.cs.ubc.ca/~lowe/525/papers/calonder_eccv10.pdf (Accessed: 16 October 2017).

Canny, J. (1986) 'A Computational Approach to Edge Detection', *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, (6). Available at: https://pdfs.semanticscholar.org/55e6/6333402df1a75664260501522800cf3d26b9.pdf (Accessed: 10 October 2017).

Cernadas, E. *et al.* (2016) 'Influence of normalization and color space to color texture classification'. doi: 10.1016/j.patcog.2016.07.002.

Cowan, B. *et al.* (2016) 'A performance evaluation of detectors and descriptors for UAV visual tracking', in *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, pp. 1–6. doi: 10.1109/ICARCV.2016.7838649.

DNER (1999) 'Manual de Sinalização Rodoviária - DNER', *Manual de Sinalização Rodoviária*. Available at: http://www1.dnit.gov.br/arquivos_internet/ipr/ipr_new/manuais/ManualSinalizacaoRodoviaria.pdf (Accessed: 4 February 2017).

Dr. Jadranka Dokic, Dr. Beate Müller and Dr. Gereon Meyer (2015) *European Roadmap Smart Systems for Automated Driving*. Berlin. Available at: http://www.smart-systems-integration.org/public (Accessed: 9 February 2017).

Duda, R. O. and Hart, P. E. (1972) 'Use of the Hough Transformation To Detect Lines and Curves in Pictures'. Available at: https://www.cse.unr.edu/~bebis/CS474/Handouts/HoughTransformPaper.pdf (Accessed: 22 October 2017).

European Comission (no date) *Statistics – accidents data - European Commission*. Available at: https://ec.europa.eu/transport/road_safety/specialist/statistics_pt# (Accessed: 16 January 2017).

European Commission (2002) *European Commission : CORDIS : Projects Results Service : The best advanced driver assistance systems*. Available at: http://cordis.europa.eu/result/rcn/45209_en.html (Accessed: 25 January 2017).

European Commission (2016) *Road safety evolution in EU*.

Ezell, S. (2010) 'Intelligent Transportation Systems'.

Farhat, W. *et al.* (2016) 'Real-time hardware/software co-design of a traffic sign recognition system using Zynq FPGA', in *2016 11th International Design & Test Symposium (IDT)*. IEEE, pp. 302–307. doi: 10.1109/IDT.2016.7843059.

Fleyeh, H. and Dougherty, M. (2005) 'Advanced OR and AI Methods in Transportation ROAD AND TRAFFIC SIGN DETECTION AND RECOGNITION'.

Gil (2013) *A tutorial on binary descriptors – part 4 – The BRISK descriptor*, *Gil's CV blog*. Available at: https://gilscvblog.com/2013/11/08/a-tutorial-on-binary-descriptors-part-4-the-brisk-descriptor/ (Accessed: 16 October 2017).

Glasbey, C. A. (Chris A. . and Horgan, G. W. (Graham W. . (1995) *Image analysis for the biological sciences*. J. Wiley. Available at: http://www.bioss.ac.uk/people/chris/ch4.pdf (Accessed: 21 February 2017).

Google (2017a) *ART and Dalvik | Android Open Source Project*. Available at: https://source.android.com/devices/tech/dalvik/ (Accessed: 3 March 2017).

Google (2017b) *Platform Architecture | Android Developers*. Available at: https://developer.android.com/guide/platform/index.html?hl=pt#linux-kernel (Accessed: 3

March 2017).

Google (2017c) *Qualidade do aplicativo para Auto | Android Developers*. Available at: https://developer.android.com/develop/quality-guidelines/auto-app-quality.html (Accessed: 7 October 2017).

Google (2017d) *RenderScript | Android Developers*. Available at: https://developer.android.com/guide/topics/renderscript/compute.html (Accessed: 17 October 2017).

Gu, Y. *et al.* (2010) 'A new vision system for traffic sign recognition', in *2010 IEEE Intelligent Vehicles Symposium*. IEEE, pp. 7–12. doi: 10.1109/IVS.2010.5548005.

Hosang, J., Benenson, R. and Schiele, B. (2017) 'Learning non-maximum suppression'. Available at: http://arxiv.org/abs/1705.02950 (Accessed: 15 October 2017).

IHS Markit (2016) *Consumers Crave Advanced Technology in New Vehicle Purchases, IHS Markit Survey Finds - IHS Technology*, *IHS Press Release*. Available at: https://technology.ihs.com/581934/consumers-crave-advanced-technology-in-new-vehicle-purchases-ihs-markit-survey-finds (Accessed: 5 February 2017).

Instituto da Mobilidade e dos Transportes (IMT, I. P. . (no date) 'Sinalização Vertical Características'.

Instituto Nacional de Estatística (2015) *Statistics Portugal - Indicador Idade Média Veículo de Passageiros Motorizado*. Available at: https://www.ine.pt/xportal/xmain?xpid=INE&xpgid=ine_indicadores&indOcorrCod=0007246&contexto=bd&selTab=tab2 (Accessed: 25 January 2017).

Instituto Nacional de Estatística (2016) *Statistics Portugal*. Available at: https://www.ine.pt/xportal/xmain?xpid=INE&xpgid=ine_indicadores&indOcorrCod=0007246&contexto=bd&selTab=tab2 (Accessed: 23 January 2017).

Jatmiko, D. A. and Prihatmanto, A. S. (2016) 'Traffic signs text recognition and error correction', in *2016 International Conference on Frontiers of Information Technology (FIT)*. IEEE, pp. 123–127. doi: 10.1109/FIT.2016.7857550.

Joyce, A. and Paquin, R. L. (2016) 'The triple layered business model canvas: A tool to design more sustainable business models', *Journal of Cleaner Production*, 135, pp. 1474–1486. doi: 10.1016/j.jclepro.2016.06.067.

Kala, R. and Warwick, K. (2015) 'Intelligent Transportation System with Diverse Semi-Autonomous Vehicles', *International Journal of Computational Intelligence Systems*, 8(5), pp. 886–899. doi: 10.1080/18756891.2015.1084710.

Karthiga.PL, S. M. M. R. e K. (2016) 'TRAFFIC-SIGN RECOGNITION FOR AN INTELLIGENT VEHICLE/DRIVER ASSISTANT SYSTEM USING HOG', *Computer Science & Engineering: An International Journal (CSEIJ)*, 6(1). doi: 10.5121/cseij.2016.6102.

Koen, P. *et al.* (2001) 'Research-Technology Management Providing Clarity and A Common Language to the Fuzzy Front End', *Research-Technology Management*, 44(2), pp. 46–55. doi: 10.1080/08956308.2001.11671418.

Kurt Demaagd, Anthony Oliver, Nathan Oostendorp, K. S. (2012) *Practical Computer Vision with SimpleCV*. O'Reilly.

Lawrence D. Miles (2015) *Techniques of Value Analysis and Engineering*. Available at: https://books.google.pt/books?hl=pt-PT&lr=&id=yCf0CQAAQBAJ&oi=fnd&pg=PT16&dq=Techniques+of+value+analysis+and+engineering&ots=vUwygHMHzx&sig=mv-WBCnrQeIiAkDDHaU_Ekt4HhQ&redir_esc=y#v=onepage&q=Techniques of value analysis and engineering&f=false (Accessed: 25 February 2017).

Leutenegger, S., Chli, M. and Siegwart, R. Y. (2011) 'BRISK: Binary Robust invariant scalable keypoints', in *2011 International Conference on Computer Vision*. IEEE, pp. 2548–2555. doi: 10.1109/ICCV.2011.6126542.

Lindgreen, A. and Wynstra, F. (2005) 'Value in business markets: What do we know? Where are we going?' doi: 10.1016/j.indmarman.2005.01.001.

Lowe, D. G. (1999) 'Object Recognition from Local Scale-Invariant Features', *Corfu*. Available at: http://www.cs.ubc.ca/~lowe/papers/iccv99.pdf (Accessed: 4 March 2017).

Lowe, D. G. (2004) 'Distinctive Image Features from Scale-Invariant Keypoints', *International Journal of Computer Vision*.

M. Hassaballah, A. A. A. and H. A. A. (2016) 'Image Features Detection, Description and Matching'. Springer International Publishing Switzerland 2016.

Maldonado-Bascon, S. *et al.* (2007) 'Road-Sign Detection and Recognition Based on Support Vector Machines', *IEEE Transactions on Intelligent Transportation Systems*, 8(2), pp. 264–278. doi: 10.1109/TITS.2007.895311.

Martin Fowler (2004) *Inversion of Control Containers and the Dependency Injection pattern*. Available at: https://www.martinfowler.com/articles/injection.html (Accessed: 8 October 2017).

Matas, J. *et al.* (2002) 'Robust Wide Baseline Stereo from Maximally Stable Extremal Regions'. Available at: http://cmp.felk.cvut.cz/~matas/papers/matas-bmvc02.pdf (Accessed: 15 October 2017).

Mazda, B. (no date) 'What Does the Mazda Traffic Sign Recognition System Do?' Available at: http://www.mazdaoflodi.com/blog/what-does-the-mazda-traffic-sign-recognition-system-do/ (Accessed: 25 February 2017).

McKinsey & Company (2016a) 'Advanced driver-assistance systems: Challenges and opportunities ahead', *Advanced driver-assistance systems: Challenges and opportunities ahead*. Available at: http://www.mckinsey.com/industries/semiconductors/our-insights/advanced-driver-assistance-systems-challenges-and-opportunities-ahead (Accessed: 16 January 2017).

McKinsey & Company (2016b) 'Capturing the advanced driver-assistance systems opportunity'. Available at: http://www.mckinsey.com/industries/automotive-and-assembly/our-insights/capturing-the-advanced-driver-assistance-systems-opportunity (Accessed: 6 February 2017).

Michael Mattsson (1999) *Object-Oriented Frameworks*, *ResearchGate*. Available at:

https://www.researchgate.net/publication/2238535_Object-Oriented_Frameworks (Accessed: 8 October 2017).

Microsoft (2017) *Portable Image and Video processing with help from AForge.NET and Accord.NET | Coding4Fun Blog | Channel 9*, *Coding4Fun Blog*. Available at: https://channel9.msdn.com/coding4fun/blog/Portable-Image-and-Video-processing-with-help-from-AForgeNET-and-AccordNET (Accessed: 3 March 2017).

Microsoft (no date) *Intro to the Universal Windows Platform - UWP app developer | Microsoft Docs*. Available at: https://docs.microsoft.com/pt-pt/windows/uwp/get-started/universal-application-platform-guide (Accessed: 4 March 2017).

Miksik, O. and Mikolajczyk, K. (2012) 'Evaluation of Local Detectors and Descriptors for Fast Feature Matching'. Available at: http://epubs.surrey.ac.uk/806146/1/miksik2012icpr.pdf (Accessed: 17 October 2017).

Mobileye (2017a) 'About Us - Mobileye'. Available at: http://www.mobileye.com/about/ (Accessed: 20 February 2017).

Mobileye (2017b) *Our Technology - Mobileye*. Available at: http://www.mobileye.com/our-technology/ (Accessed: 20 February 2017).

Mobileye (no date) *The Evolution of EyeQ - Mobileye*. Available at: http://www.mobileye.com/our-technology/evolution-eyeq-chip/ (Accessed: 20 February 2017).

OpenCV (2015) *OpenCV: Contours Hierarchy*. Available at: https://docs.opencv.org/3.1.0/d9/d8b/tutorial_py_contours_hierarchy.html (Accessed: 10 October 2017).

Osterwalder, A. *et al.* (2010) *Business model generation: A handbook for visionaries, game Changers, and challengers (portable version)*.

Osterwalder, A. and Pigneur, Y. (2003) 'Modeling value propositions in e-Business', in *Proceedings of the 5th international conference on Electronic commerce - ICEC '03*. New York, New York, USA: ACM Press, pp. 429–436. doi: 10.1145/948005.948061.

Paul, R., Niewoehner, R. and Elder, L. (2006) 'THE THINKER'S GUIDE TO Engineering Reasoning'.

Peter A.Koen, Greg M.Ajamian, S. B. *et al.* (2002) *Fuzzy Front End: Effective Methods, Tools, and Techniques*. The PDMA T. Edited by N. Y. J. W. & Sons.

Peterson, S., Jaret, P. E. and Schenck, B. F. (2005) *Business Plans Kit For Dummies® , 2nd Edition*. Edited by I. Wiley Publishing. Wiley Publishing, Inc. Available at: www.wiley.com (Accessed: 12 February 2017).

Porter, M. E. (1980) *Competitive Strategy*. New York, New York, USA: THE FREE PRESS, A Division of Simon & Schuster Inc 1230 Avenue of the Americas. Available at: http://www.vnseameo.org/ndbmai/CS.pdf.

Portugal, I. de (no date) *Início | Infraestruturas de Portugal*. Available at: http://www.infraestruturasdeportugal.pt/rede/rodoviaria/seguranca-rodoviaria/normas-de-sinalizacao (Accessed: 23 February 2017).

Quinlan, J. R. (1986) 'Induction of Decision Trees', *Machine Learning*, 1, pp. 81–106. Available at: http://hunch.net/~coms-4771/quinlan.pdf (Accessed: 15 October 2017).

Ren, F. *et al.* (2009) 'General traffic sign recognition by feature matching', in *2009 24th International Conference Image and Vision Computing New Zealand*. IEEE, pp. 409–414. doi: 10.1109/IVCNZ.2009.5378370.

República Portuguesa (1998) 'Diário da República - Decreto Regulamentar nº 22-A/98', *Diário da República*. Available at: http://www.ansr.pt/SegurancaRodoviaria/RegulamentoSinalizacaoTransito/Pages/default.aspx (Accessed: 4 February 2017).

Research and Markets (2016) *Global Automotive Traffic Sign Recognition System Market 2016-2020 - Price Fall in ADAS Subsystems*. Available at: http://www.prnewswire.com/news-releases/global-automotive-traffic-sign-recognition-system-market-2016-2020----price-fall-in-adas-subsystems-to-help-oems--consumers---research-and-markets-300293916.html (Accessed: 28 January 2017).

Rosten, E. and Drummond, T. (2005) 'Fusing Points and Lines for High Performance Tracking'. Available at: https://www.edwardrosten.com/work/rosten_2005_tracking.pdf (Accessed: 15 October 2017).

Rosten, E. and Drummond, T. (2006) 'Machine learning for high-speed corner detection'. Available at: https://www.edwardrosten.com/work/rosten_2006_machine.pdf (Accessed: 15 October 2017).

Rosten, E., Porter, R. and Drummond, T. (2010) 'Faster and better: a machine learning approach to corner detection'. Available at: https://www.edwardrosten.com/work/rosten_2008_faster.pdf (Accessed: 15 October 2017).

Rublee, E. *et al.* (2011) 'ORB: an efficient alternative to SIFT or SURF'. Available at: http://www.willowgarage.com/sites/default/files/orb_final.pdf (Accessed: 16 October 2017).

Saaty, T. L. and Vargas, L. G. (Luis G. (2012) *Models, methods, concepts & applications of the analytic hierarchy process*. Springer.

Scott Krig (2014) *Interest Point Tuning*, *Computer Vision Metrics*. Available at: https://link.springer.com/content/pdf/10.1007%2F978-1-4302-5930-5_6.pdf (Accessed: 16 October 2017).

Shi, S. (2013) *Emgu CV Essentials.* Packt Publishing. Available at: https://www.packtpub.com/application-development/emgu-cv-essentials (Accessed: 4 March 2017).

Smithers Group (2014) *Growth of Image Sensors in Automotive Market | News | Smithers Apex*. Available at: https://www.smithersapex.com/news/2014/february/significant-growth-potential-for-image-sensors (Accessed: 27 January 2017).

Susana Nicola, Eduarda Pinto Ferreira, J. J. P. F. (2012) 'Conceptual Model for Decomposing the Value for the Customer', *3rd Industrial Engineering and Management Symposium*.

Sussman, J. M. (Joseph M. (2005) *Perspectives on intelligent transportation systems (ITS)*. Springer Science+Business Media.

Thomas L. Saaty (1990) 'How to make a decision: The Analytic Hierarchy Process', *European Journal of Operational Research*, 48, pp. 9–26. Available at: https://www.ida.liu.se/~TDDD06/literature/saaty.pdf (Accessed: 19 February 2017).

Tuytelaars, T. and Mikolajczyk, K. (2008) 'Local Invariant Feature Detectors: A Survey', *Computer Graphics and Vision*, 3(3), pp. 177–280. doi: 10.1561/0600000017.

Union, E. and Commission, E. (2014) *CARS 2020 2014 Report*, *CARS 2020 Report on the state of play of the outcome of work of the High Level Group*. Available at: http://ec.europa.eu/growth/tools-databases/newsroom/cf/itemdetail.cfm?item_id=8085 (Accessed: 11 January 2017).

Value Networks, L. (2011) 'Value Networks Tutorial – Healthcare Scheduling'.

Verna Allee (2000) *Reconfiguring the Value Network*, *Published in Journal of Business Strategy, Vol 21, N 4, July-Aug 2000*. Available at: http://www.sveiby.com/articles/Allee-ValueNets.htm (Accessed: 30 January 2017).

Verna Allee (2008) 'Value network analysis and value conversion of tangible and intangible assets', *Journal of Intellectual Capital*, 9(1), pp. 5–24. Available at: http://dx.doi.org/10.1108/14691930810845777%0D.

Volvo Trucks (2013) 'European Accident Research and Safety Report 2013'.

Waite, S. and Oruklu, E. (2012) 'FPGA-Based Traffic Sign Recognition for Advanced Driver Assistance Systems'. doi: 10.4236/jtts.2012.*****.

WHO (2016) 'Global status report on road safety 2015', *WHO*. World Health Organization.

Woodall, T. (2003) 'Conceptualising "Value for the Customer": An Attributional, Structural and Dispositional Analysis', *Academy of Marketing Science Review*, volume 200.

Xamarin (2017a) *An Introduction to Xamarin.Forms - Xamarin*. Available at: https://developer.xamarin.com/guides/xamarin-forms/getting-started/introduction-to-xamarin-forms/ (Accessed: 18 October 2017).

Xamarin (2017b) *Building Cross Platform Applications Overview*. Available at: https://developer.xamarin.com/guides/cross-platform/application_fundamentals/building_cross_platform_applications/part_0_-_overview/ (Accessed: 4 March 2017).

Xamarin (2017c) *Controls Reference - Xamarin*. Available at: https://developer.xamarin.com/guides/xamarin-forms/user-interface/controls/ (Accessed: 8 October 2017).

Yu, L., Yu, Z. and Gong, Y. (2015) 'An Improved ORB Algorithm of Extracting and Matching Features', *International Journal of Signal Processing Image Processing and Pattern Recognition*, 8(5), pp. 117–126. doi: 10.14257/ijsip.2015.8.5.12.

ANEXO 1 : Sinalização Suportado

Imagens recolhidas através do web site:
http://www.infraestruturasdeportugal.pt/rede/rodoviaria/seguranca-rodoviaria/normas-de-sinalizacao


A1a - Curva à direita


A1b - Curva à esquerda


A1c - Curva à direita e contracurva


A1d - Curva à esquerda e contracurva


A2a - Lomba


A2b - Depressão


A2c - Lomba ou depressão

A3a - Descida perigosa

A3b - Subida de inclinação acentuada


A4a - Passagem estreita


A4b - Passagem estreita


A4c - Passagem estreita


A5 - Pavimento escorregadio


A6 - Projeção de gravilha


A7a - Bermas baixas


A7b - Bermas baixas

A8 - Saída num cais ou precipício

A9 - Queda de pedras

A10 - Ponte móvel

A11 - Neve ou gelo

A12 - Vento lateral

A13 - Visibilidade insuficiente

A14 - Crianças

A15 - Idosos

A16a - Passagem de peões

A16b - Travessia de peões

A17 - Saída de ciclistas

A18 - Cavaleiros

A19a - Animais

A19b - Animais selvagens

A20 - Túnel

A21 - Pista de aviação

A22 - Sinalização luminosa

A23 - Trabalhos na via

A24 - Cruzamento ou entroncamento

A25 - Trânsito nos dois sentidos

A26 - Passagem de nível com guarda

A27 - Passagem de nível sem guarda

A28 - Interseção com via onde circulam veículos sobre carris

A29 - Outros perigos

A30 - Congestionamento

A31 - Obstrução da via

A32a - Local de passagem de nível sem guarda

A32b - Local de passagem de nível sem guarda com duas ou mais vias

B9a - Entroncamento com via sem prioridade

B9b - Entroncamento com via sem prioridade

B9c - Entroncamento com via sem prioridade

B9d - Entroncamento com via sem prioridade

B7 - Aproximação de rotunda

B8 - Cruzamento com via sem prioridade