Bruno Daniel Horta Medinas

Licenciado em Engenharia Informática

InspFocus
Análise de Imagem para Inspeção Semi-Automática de Veículos Automóveis

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática

Orientador: Nuno Manuel Robalo Correia,
Professor Catedrático,
Faculdade de Ciências e Tecnologia
da Universidade Nova de Lisboa

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2017

InspFocus
Análise de Imagem para Inspeção Semi-Automática de Veículos Automóveis

Copyright © Bruno Daniel Horta Medinas, Faculdade de Ciências e Tecnologia, Universidade NOVA de Lisboa.
A Faculdade de Ciências e Tecnologia e a Universidade NOVA de Lisboa têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objetivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Aos meus pais Joaquim e Carolina, ao meu irmão André e à minha namorada Raquel. Sem vocês não me via a concretizar esta etapa da minha vida, a vossa ajuda e apoio em inúmeras situações permitiram-me ter força para ultrapassar todos os obstáculos.

Aos meus avós, que já não se encontram connosco, mas que teriam o maior prazer do mundo de me ver onde estou agora.

Agradecimentos

Em primeiro lugar obrigado ao Professor Nuno Correia pelo apoio no decorrer de todo o trabalho, aos professores do Departamento de Informática que me prepararam ao longo dos anos para poder realizar esta dissertação e à Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa por me acolherem durante o decorrer do curso. Gostaria também de agradecer a bolsa que me foi concedida. Um agradecimento também à equipa do Centro de Inspeções de Vila Nova de Poiares pela troca de informações e recolha de imagens do domínio.

Um grande obrigado aos meus colegas de curso, sem eles as inúmeras horas a trabalhar no departamento não tinham sido a mesma coisa. À minha família, que nunca me deixou faltar nada para me poder focar no trabalho. E por fim à minha namorada, porque sem os incentivos dela, não tinha sido possível chegar aqui.

Resumo

Com o aumento das exigências dos consumidores, as tradicionais técnicas de inspeção já não se adequam às necessidades de sistemas cada vez mais rápidos e eficientes. O processo tradicional de inspeção tem como base um elemento humano para comprovar visualmente se o produto está apto ou não. Este fator causa diversos problemas devido à ineficiência e falibilidade do ser humano.

É no seguimento deste problema que a inspeção automática ocorre. Ao utilizar sistemas de inspeção automática, remove-se o fator de erro humano proveniente de fatiga, da repetição de tarefas monótonas, da subjetividade, entre outros.

Esta dissertação surge de forma a contribuir para a resolução do problema através do estudo da viabilidade de certos algoritmos para a sua utilização numa aplicação de inspeção semi-automática, possibilitando a substituição do fator humano na fase de avaliação do funcionamento e condição do veículo. No futuro, esta aplicação receberá imagens e/ou vídeos captados por uma câmara colocada num braço robótico, processando de seguida a informação recolhida e devolve ao utilizador um relatório sobre se o veículo está aprovado, e no caso de não estar, devolver uma lista dos problemas.

Palavras-chave: Inspeção Automática, Machine Vision, Inspeção Automóvel, Processamento de Imagem/Vídeo

Abstract

Due to an increase in consumer demand, traditional techniques of inspection are no longer adequate for the necessities of systems that are increasingly faster and more efficient. The traditional process of inspection depends on a human element to prove visually if a product is okay or not. The presence of this element causes multiple problems, due to the inefficiency and fallibility of the human being.

It was due to the existence of this problem that automatic inspection appeared. Using automatic inspection systems, it's possible to remove human error that originates in fatigue, repetition of monotonous tasks, subjectivity, among others.

This dissertation aims at contributing for the solution of this problem by studying the viability of certain algorithms for their use in a semi-automatic inspection application, allowing the substitution of the human factor in the evaluation phase of the operation and condition of the vehicle. In the future, this application will receive images and/or video captured by a camera placed in a robotic arm, processing the collected information and finally returning a report to the user, containing the verdict of the inspection, either approved, or denied and listing all the problems occurred.

Keywords: Automatic Inspection, Machine Vision, Automobile Inspection, Image/Video processing

Índice

Lista de Figuras

Lista de Tabelas

Siglas

BF	Brute-Force.
BoW	Bag-of-words.
CLAHE	Contrast Limited Adaptive Histogram Equalization.
FLAAN	Fast Library for Approximate Nearest Neighbors.
ORB	Oriented FAST and rotated BRIEF.
SIFT	Scale-invariant Feature Transform.
SURF	Speed Up Robust Features.
SVM	Support Vector Machine.

Capítulo 1

Introdução

Devido à evolução da tecnologia, é hoje possível complementar o ser humano em funções como a de a inspeção visual de diferentes objetos. Tornar computadores capazes de interagir desta forma com o mundo é uma área chamada Visão por Computador, do inglês *Machine Vision*.

Visão por Computador é a área que engloba todos os processos e técnicas que permitem a sistemas visuais de inspeção automática analisar a tarefa em questão e produzir um resultado especifico de forma autónoma.

A utilização destas técnicas permite a criação de um sistema que processa a condição de um veiculo automaticamente, indicando as possíveis falhas que estejam presentes nele.

1.1 Motivação

Vivemos numa sociedade que está em constante mutação. Existe uma necessidade de tornar tudo cada vez mais eficiente, de forma a reduzir o tempo de produção de artigos ou até de reduzir o tempo de espera dos clientes.

No caso de sistemas de produção em linha ou noutros ramos como a inspeção automóvel, ter uma pessoa a realizar a inspeção, torna-se um obstáculo para a eficiência de todo o sistema.

A necessidade de um sistema de inspeção automática provém de o ser o humano não ser eficiente em tarefas de detetar sistematicamente problemas e da possibilidade de falhas causadas por subjetividade, fadiga ou outros.

A ideia será complementar o processo de inspeção com um sistema que permite a um inspetor realizar um serviço mais pormenorizado e rigoroso.

1.2 Descrição e contexto

Esta dissertação surge no contexto de um projeto inserido no programa Portugal 2020[1], em colaboração com a equipa do Centro de Inspeções de Vila Nova de Poiares.

A temática desta dissertação surge da necessidade de automatizar todo o processo da inspeção de automóveis, de forma a permitir resultados mais consistentes e corretos.

O objetivo deste projeto é especificar, desenvolver e avaliar técnicas de análise de imagem para inspeção automóvel. Será desenvolvida uma aplicação que permite receber imagens do domínio, para as poder analisar com o objetivo de, com base num conjunto de imagens treino, reconhecer peças e defeitos para poder relatar possíveis problemas com o automóvel.

Estas imagens serão captadas por um braço robótico com uma câmara que integra o sistema.

1.3 Principais contribuições

Nesta dissertação desenvolveu-se os seguintes tópicos:

Desenvolvimento e avaliação de técnicas de análise de imagem - identificação de *features* diferentes em imagens através do uso de técnicas de pré-processamento e manipulação de parâmetros de diferentes detetores.

Algoritmos e testes aos mesmos - algoritmos que permitem o reconhecimento de peças. Foram realizados testes com imagens do domínio, utilizando conjuntos de testes normalizados a estes algoritmos.

1.4 Organização do documento

Este documento irá seguir a estrutura apresentada de seguida:

Introdução: neste capitulo são descritos o tema desta dissertação, a sua motivação, descrição e contexto e por fim as principais contribuições previstas ao realizar esta dissertação.

Trabalho Relacionado: no segundo capitulo é feita uma introdução à temática de Inspeção Industrial, de Inspeção Automóvel e de Visão por Computador. É apresentado um estudo do que é processamento de imagem, indicando alguns algoritmos e técnicas relacionadas com a temática. Por fim são apresentados alguns sistemas relacionados para que se torne possível retirar algumas conclusões relativas a técnicas previamente aplicadas.

[1] https://www.portugal2020.pt/Portal2020/o-que-e-o-portugal2020

Especificação da solução: neste capitulo é feita uma descrição do trabalho que foi efetuado no decorrer da dissertação. É apresentada uma explicação das diferentes técnicas que foram aplicadas para a análise e comparação de imagens.

Resultados: os resultados apresentados no capitulo correspondem aos testes realizados com determinados conjuntos de dados de forma a obter métricas que permitissem avaliar a *performance* das diferentes técnicas.

Conclusões: por fim, é feita uma conclusão sobre as diferentes técnicas em relação à sua eficácia para o problema em questão. É também feita uma breve descrição do trabalho a realizar no futuro.

Capítulo 2

Trabalho relacionado

No presente capítulo é apresentada uma introdução à temática e um estudo de sistemas relacionados com o proposto por esta dissertação.

Nas secções 2.1, 2.2 e 2.3 são introduzidos conceitos sobre inspeção industrial, inspeção automóvel e visão por computador. Na secção 2.4 é apresentado um estudo de processamento de imagem no contexto de visão por computador e ainda apresentado técnicas e conceitos relacionados. Na secção 2.5 são descritos alguns sistemas relacionados, com o objetivo de retirar algumas conclusões sobre a utilidade de implementar determinadas técnicas num futuro sistema desenvolvido no decorrer desta dissertação.

2.1 Inspeção industrial

Uma inspeção consiste na avaliação realizada a um determinado objeto com o propósito de verificar se este se encontra dentro de parâmetros previamente definidos. Para este fim, são realizadas medições, testes e exames a um dado conjunto de características deste objeto para ser possível comparar as métricas retiradas destas operações com os requisitos da inspeção em causa. Por norma, o alvo de uma inspeção não sofre qualquer dano, sendo que são realizadas de uma forma não invasiva.

A inspeção pode classificar-se com base no método com que foi realizada. Segundo [1] estas classificações são:

- Inspeção visual.
- Inspeção por tecnologias de deteção [1].

[1] Tecnologias que permitem detetar características não visíveis ao olho humano

Ambos os métodos podem ser realizados de forma presencial, ou através de um meio de transmissão remota, permitindo a avaliação do objeto em condições que não é fisicamente possível um ser humano estar presente, ou em que é necessário reencaminhar para ser inspecionado noutro local.

Pode-se ainda subdividir a inspeção em **manual** e **automática**, como representado na figura 2.1, dependendo se é realizada por uma pessoa, ou por um sistema com capacidade de recolha e análise de imagens do objeto a ser inspecionado e posteriormente devolver o relatório com a conclusão da inspeção.

Manual [2]

Automático [3]

Figura 2.1: Exemplos de inspeções manuais e automáticas

2.2 Inspeção automóvel

A inspeção automóvel é realizada com o objetivo de verificar se os veículos se encontram em condições seguras para circularem na via pública, sem por em causa tanto a segurança do condutor e seus passageiros, como a de outras pessoas na via.

A entidade responsável pela regulamentação destas inspeções é o Instituto da Mobilidade e dos Transportes Terrestres, ou IMTT. A inspeção é realizada por inspetores, titulares de uma licença de atividade emitida pelo IMTT, em centros de inspeção (figura 2.2) autorizados pela instituição previamente referida.

Segundo o IMTT [5] as inspeções servem para o controlo das condições técnicas dos veículos e é um imperativo nacional e comunitário a sua realização. Têm por objetivo a melhoria das condições de circulação dos veículos, através da verificação periódica das suas características e das suas condições de segurança, com particular importância para a salvaguarda da segurança rodoviária.

Existem diferentes tipos de inspeção, sendo estas:

Periódicas - As inspeções periódicas visam confirmar, com regularidade, a manutenção das boas condições de funcionamento e de segurança de todo o equipamento e das condições de segurança dos automóveis ligeiros, pesados e seus reboques [6].

Figura 2.2: Centro de inspeções em Portugal [4]

Extraordinárias - As inspeções extraordinárias destinam-se a identificar ou confirmar ocasionalmente as condições de segurança dos veículos, em consequência da alteração das suas características construtivas ou funcionais, por acidente ou outras causas que comprometem a segurança do veículo [7].

Para Matrícula - As inspeções para atribuição de nova matricula têm por objetivo a identificação de veículos anteriormente matriculados, a verificação das respetivas caraterísticas e a confirmação das suas condições de funcionamento e de segurança [8].

Facultativas - As inspeções facultativas têm como finalidade a verificação das condições de funcionamento e de segurança de todo o equipamento e das condições de segurança dos veículos [9].

No fim de uma inspeção é apresentado um resultado, indicando se o veículo foi aprovado ou reprovado. Caso seja aprovado, é atribuído um certificado verde e o veículo pode circular até à próxima data de inspeção. Caso seja reprovado, é atribuído um certificado vermelho e o veiculo pode circular durante um curto período, apenas com o objetivo de reparar os problemas detetados pela inspeção.

2.3 Visão por computador

Visão por computador consiste nas tecnologias e métodos usados para executar inspeção automática baseada em imagens. Sistemas de visão por computador têm como principal uso inspeção automática e orientação de robôs industriais, sendo que existem

inúmeras aplicações dentro destas áreas, como por exemplo inspeção automática de PCB [2] e sistemas de controlo de qualidade.

Segundo Allan Oliveira et al. [15] o típico sistema de visão por computador consiste numa estrutura com a seguinte cadeia de processos:

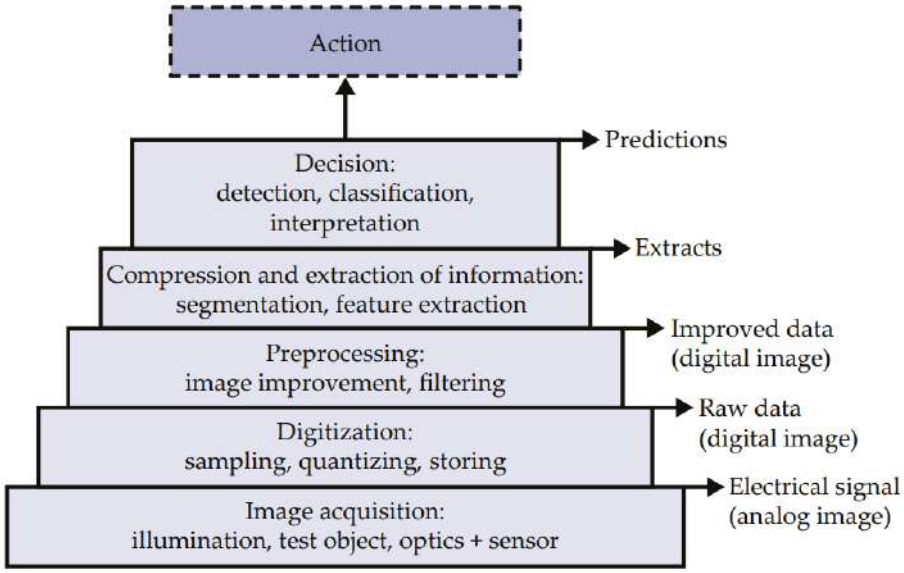


Figura 2.3: Cadeia de processos de um sistema de visão por computador [15]

1. **Aquisição de imagem: iluminação, objeto de teste, ótica e sensor** - Este passo é o mais importante do processo, visto que a qualidade da informação captada depende dele. Se alguma informação não for adquirida neste passo, é praticamente impossível recuperá-la em passos posteriores. É de salientar que bastantes sistemas de inspeção falham por não conseguir cumprir este requisito inicial.

2. **Digitalização: amostragem, quantização** [3]**, armazenamento** - A imagem é captada pelo sensor, onde é transformada num sinal elétrico. De seguida este sinal analógico é convertido numa medida discreta e é limitado com base no espaço e amplitude. Desta forma, é possível converter o sinal numa imagem digital para ser processado por um computador. O problema reside no facto de este sinal não conter exclusivamente dados relevantes, mas também ruído e incongruências.

3. **Pré-processamento: Melhoria de imagem, filtragem** - Neste passo, é realizado um processo de recuperação para tentar manter a informação relevante e compensar os dados irrelevantes no sinal.

4. **Compressão e extração de informação: Segmentação, extração de características** - Como resultado do pré-processamento, dados da imagem melhorados são obtidos,

[2] *Printed Circuit Boards*

[3] Quantização é o processo de atribuição de valores discretos para um sinal cuja amplitude varia entre infinitos valores.

sendo posteriormente separados em regiões com significado ou processados de forma a extrair os parâmetros relevantes à tarefa.

5. **Decisão: deteção, classificação, interpretação** - Por fim, podem ser tomadas decisões com base na informação processada. Essas decisões consistem em deteções (verificar defeitos no objeto), em classificações (reconhecer um objeto) ou interpretações (determinar parâmetros do objeto).

6. **Ação** - Por fim, é possível tomar uma ação com base na decisão tomada pelo sistema, dependo da tarefa em questão. Por exemplo, num sistema de deteção de defeitos, um resultado negativo (objeto com defeito) pode resultar na eliminação desse objeto.

2.4 Processamento de imagem

Processamento de imagem consiste em processar imagens usando operações matemáticas onde o *input* consiste em imagens ou vídeos. Como *output* é comum receber características ou parâmetros da imagem.

2.4.1 Extração de *features*

Um *feature* ou característica é uma propriedade de algo que se está a observar. A recolha de *features* bem distintos e com significado é extremamente importante para detetar padrões ou aplicar classificação. Em processamento de imagem, a extração de *features* consiste em retirar de uma imagem (ou um conjunto de imagens) características informativas e não redundantes. Esta seleção permite uma maior facilidade em aplicar algoritmos posteriormente para a tarefa em questão.

De acordo com [27], podemos dividir as técnicas de extração de *features* nas que tratam de extrair *low-level features* e as que tratam de *high-level features*.

2.4.1.1 *Low-level features*

Low-level features definem-se como as características básicas que podem ser extraídas automaticamente de uma imagem sem nenhuma informação de forma [23].

Um exemplo de uma técnica de extração é o *thresholding*, que consiste em substituir um pixel de uma imagem por um pixel preto caso a intensidade da imagem esteja inferior a um certo limite ou por um pixel branco caso contrario, resultando numa imagem binária. Processo exemplificado na figura 2.4.

O objetivo desta técnica é gerar imagem binárias, reduzindo efetivamente os dados a processar, mas mantendo a informação suficiente para a tarefa em questão.

Outra técnica bastante utilizada é denominada de *edge detection*. Esta tem como objetivo identificar pontos na imagem onde o brilho varia (figura 2.5).

Ao aplicar esta técnica, obtemos uma imagem com diversas curvas que delimitam objetos. Isto significa que ao aplicar esta técnica, reduzimos a quantidade de informação

Imagem original [10]

Imagem após aplicação de thresholding [11]

Figura 2.4: Exemplo de *thresholding*

Imagem original

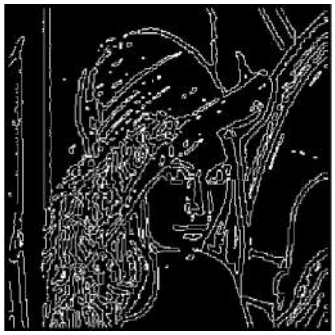


Imagem após deteção de *edges*

Figura 2.5: Exemplo de *edge detection*

que tem de ser processada posteriormente ao filtrarmos informação menos relevante, mas preservando a integridade estrutural da imagem.

Corner detection é outra técnica de extração de *low-level features*, consistindo em detetar pontos em que as linhas da imagem curvam muito acentuadamente. É possível observar o resultado da aplicação desta técnica na seguinte figura 2.6.

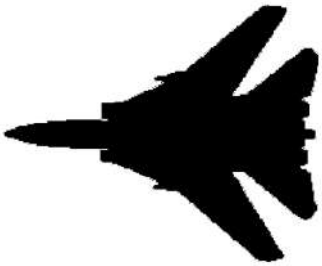


Imagem original

Imagem após deteção de cantos

Figura 2.6: Exemplo de *corner detection*

Tal como nas técnicas anteriores, o objetivo desta técnica é a redução dos dados de

forma a simplificar processamento da imagem posteriormente em etapas de classificação por exemplo.

Optical flow consiste numa técnica que permite descrever o movimento presente numa sequencia de imagens. Na figura 2.9 podemos observar o resultado da aplicação desta técnica. Os pontos mais claros correspondem à zona com o maior grau de movimento, tal como as mãos ou os pés.

Sequencia de imagens Imagem após deteção de cantos

Figura 2.7: Exemplo de *optical flow*

A informação que se retira do resultado desta técnica permite delinear um conjunto de pontos que se podem agrupar e permitir a extração de uma forma. No caso apresentado na figura 2.9, é possível extrair uma forma correspondente a uma pessoa a caminhar, sendo assim determinada a existência de um objeto na imagem, a pessoa.

Existem diversas outras técnicas para a extração de *low-level features*, sendo que apenas foram realçadas algumas das mais comuns.

2.4.1.2 *High-level features*

High-level features correspondem às características que permitem extrair formas em imagens. Estas características são utilizadas maioritariamente para tarefas de aprendizagem automática. Para realizar essa tarefa é necessário extrair os diversos componentes da imagem, ou seja, os olhos, as orelhas, o nariz, as características mais relevantes que permitem detetar a face de um ser humano. Para detetar essas características é possível reconhecer a sua forma, por exemplo o olho tem uma forma de elipse e a boca de duas linhas. Extração de formas implica encontrar a posição, orientação e tamanho de um objeto.

A extração destas características pode ser associada à forma de como um ser humano destingue o mundo à sua volta.

Neste tipo de extração é necessário encontrar propriedades invariáveis de forma a que o processo de extração não varie com base em diversas condições, como por exemplo a iluminação. Como tal, as técnicas que se aplicam a este tipo de extração têm de conseguir encontrar formas consistentemente, independentemente de parâmetros que possam alterar uma forma.

As invariantes mais importantes para as técnicas de extração de forma são (ordenadas da mais relevante para a menos):

1. Iluminação - É necessário ser capaz de detetar uma forma numa cena iluminada ou escura, desde que exista contraste suficiente com o fundo para a deteção da mesma.

2. Posição - É preciso ter a capacidade de detetar uma forma independentemente do local onde esta apareça.

3. Rotação - Independentemente da rotação do objeto, é preciso ter a capacidade de detetar a forma e questão.

4. Escala - Tenta-se determinar o objeto mesmo que este surja em tamanhos diferentes, por exemplo a ocorrência do posicionamento do mesmo perto ou longe da câmara, entre outras razões.

O problema consiste em que as imagens têm sempre ruído e que ao detetar formas, estas podem ocorrer múltiplas vezes na mesma imagem. Pode acontecer a oclusão de uma ou mais formas devido à sobreposição de outra forma.

Para extrair uma forma de uma imagem, é necessário identifica-la no meio de outros elementos. É possível faze-lo, por exemplo, aplicando técnicas que comparam os pixeis com base num modelo, ou considerando a informação de intensidade.

Algumas técnicas para este tipo de extração são:

- *Thresholding* e *background subtraction* [24]
- *Template matching* [25]
- *Hough transform* [26]

Thresholding consiste na técnica previamente referida. A diferença entre a aplicação para *low-level* e para *high-level* é que é necessário aplicar outra técnica previamente, devido à existência de ruído na imagem. Esse ruído aumenta a dificuldade de delimitar formas quando se tenta aplicar um certo limite para os pixeis brancos e pretos. Como tal, é aplicado *background subtraction* com o objetivo de separar o objeto do fundo da imagem, eliminando ruído que poderia afetar a extração. Na figura 2.8 podemos observar este processo.

Template matching consiste em corresponder um modelo (figura 2.9) a uma imagem (figura 2.9), onde o modelo é uma sub imagem que contém a forma que pretendemos encontrar. Para tal, é colocado o modelo num ponto da imagem e é verificada a quantidade de pontos do modelo que têm correspondência com os pontos da imagem. O processo repete-se pela imagem toda. O ponto com a melhor correspondência, corresponde a um ponto da forma que se pretendia encontrar.

Imagem original

Imagem após *background subtraction*

Imagem após *thresholding*

Figura 2.8: Exemplo de *background subtraction* seguido de *thresholding*

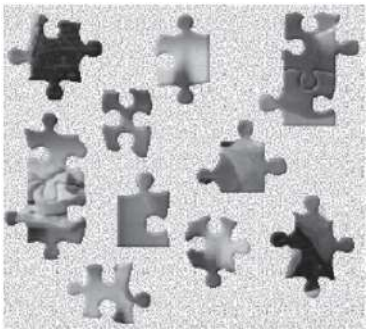


Imagem das formas

Modelo da forma alvo

Figura 2.9: Exemplo de *template matching*

Existe a possibilidade de aplicar técnicas como a binarização ou *edge detection* à imagem com o objetivo de reduzir a carga computacional ao executar o algoritmo de comparação. Isto ocorre quando o número de pontos a comparar é reduzido, sendo que no caso da binarização apenas existe a comparação de pixeis pretos e brancos, por oposição ao *edge detection* onde se comparam os pontos das arestas detetadas.

Hough transform é uma técnica desenhada para identificar formas em imagens, como podemos ver na figura 2.10. É utilizada para extrair linhas, círculos, elipses entre outras formas. A sua principal vantagem é conseguir chegar a um resultado equivalente ao de *template matching* mais rapidamente.

Imagem original | Aplicando *Hough transform*

Figura 2.10: Exemplo de *Hough transform*

Esta técnica reformula o processo da sua antecessora utilizando uma abordagem baseada em recolha de evidências, onde estas evidências são votos acumulados. A implementação desta técnica define um mapeamento dos pontos da imagem para um espaço de acumulação.

É uma técnica bastante popular devido à sua equivalência de resultados com *template matching*, garantindo neste caso uma melhor performance.

2.4.2 Classificação

Classificação consiste em determinar qual a categoria, de um conjunto de n categorias, a que pertence uma certa observação, com base num conjunto de dados de treino, o qual contém informação relativa a observações das quais se sabe a sua categoria. É considerada um caso de aprendizagem supervisionada, ou seja, um classificador aprende com base num conjunto de treino previamente definido, onde existem exemplos da pertença a certas categorias. A classificação é maioritariamente utilizada para o reconhecimento de padrões na área de processamento de imagem.

Segundo [27], a classificação consiste no processo de atribuir uma classe a um conjunto de medidas. Existem bastantes alternativas de abordagem de classificação. Uma delas são redes neuronais, nas quais os componentes computacionais tentam imitar propriedades dos neurónios no cérebro humano. Estas redes necessitam treino, de forma a minimizar

o erro. Com este treino, a rede tem a capacidade de reconhecer os dados introduzidos, sendo que é possível atribuir classes aos *outputs* da rede.

Outra abordagem comum são as Support Vector Machine (SVM). Esta abordagem tem a vantagem de ter uma capacidade de generalização excelente, tendo como valência a correta classificação de um conjunto de dados não presentes no conjunto de treino. SVMs são bastante úteis na classificação de por exemplo documentos, de imagens e caracteres escritos.

2.5 Sistemas relacionados

Nesta secção serão apresentados sistemas que tenham algum grau de semelhança com a presente dissertação e que são considerados úteis para uma melhor compreensão da mesma. Estes sistemas podem já existir ou ser apenas estudos.

Existem diversos artigos relacionados a inspeção automática, sendo que poucos poucos se focam na vertente de inspeção automóvel, só que estes apenas se concentram em algumas peças do veiculo, por exemplo travões ou retrovisores.

2.5.1 Inspeção automática de defeitos de distorção de espelhos curvos de veículos

Na industria automóvel existe a permanente procura pelo aumento do nível de segurança dos veículos de forma a salvaguardar a integridade dos passageiros. Espelhos curvos surgem neste contexto visto que permitem ao condutor ter um maior campo de visão e informação sobre o seu redor por oposição aos espelhos planos.

O problema surge na produção dos referidos espelhos curvos, nos quais podem surgir defeitos de distorção da reflexão provocados pela instabilidade da temperatura dentro dos fornos ou controlo inapropriado do processo de fusão. Devido à irregularidade da forma e fronteira dos defeitos de distorção, não é fácil determinar o quanto se encontra distorcido o espelho.

O artigo faz referência sobre o facto de que os métodos mais comuns de deteção de defeitos são inspeções realizadas por humanos. Este tipo de inspeções é falível devido a falsas conclusões provenientes da subjetividade do inspetor ou cansaço do mesmo. Apesar de mais consistente, um sistema computacional não é infalível. É necessário uma captura rigorosa das imagens a analisar, visto que a mínima modificação no padrão refletido, a diferença no ângulo da captura ou instabilidade na iluminação podem causar erro na analise.

Este estudo [21] propõe uma abordagem baseada num processo de pequenas mudanças de um padrão para inspecionar os defeitos de distorção da reflexão 2.11, tendo como objetivo criar um método de inspeção visual automática dos mesmos. O artigo salienta que a importância do sucesso desta inspeção deriva da possibilidade da distorção da informação ao redor do condutor levar o mesmo a tomar decisões erradas, podendo provocar

graves acidentes.

Retrovisor normal Retrovisor com distorção

Figura 2.11: Exemplo de retrovisor com e sem distorção [21]

De forma a quantificar a deformação, este estudo propõe a utilização de um padrão (figura 2.12) para refleti-lo num espelho capturando a imagem. Posteriormente, a reflexão do espelho é comparada com um espelho de controlo para que seja possível a determinação de quais os locais onde existe distorção. Neste processo são feito determinados os pontos de interseção do padrão e são medidas as distâncias desses pontos à origem, calculando a distância dos desvios dos pontos de interseção entre o espelho a testar e o de controlo.

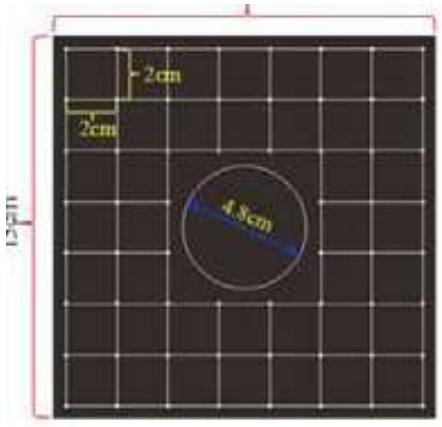


Figura 2.12: Especificações do padrão [21]

Para avaliar a proposta, o autor fez testes em retrovisores curvos de veículos. Foram selecionadas amostras aleatórias, fornecidas por uma empresa de fabrico de retrovisores. O algoritmo que permitiu testar estes retrovisores foi elaborado na linguagem Matlab e executado no ambiente interativo MATLAB (7ª versão). Com base no resultado dos testes, pode concluir-se que esta abordagem tem bastante sucesso, alcançando 98% de probabilidade de determinar corretamente se um retrovisor tem defeito ou não.

2.5.2 Método de inspeção de imagem para peças de travões

Este estudo [20] surge na necessidade do aumento da produtividade e eficiência da produção de peças para veículos, propondo um sistema automático de deteção de defeitos usando processamento de imagem. As peças alvo desta inspeção são chamadas de *MEDAMA*, apresentadas na figura 2.13 e o seu uso 2.13.

MEDAMA Uso das peças

Figura 2.13: Peças alvo de inspeção [20]

Um bom sistema de deteção de falhas é uma necessidade visto que quando utilizadas peças com defeito e pelo facto de circular óleo no seu interior, poderá dar origem a uma fuga de óleo, resultando na possível perda de eficácia da travagem, podendo causar um acidente de viação.

Os autores discutem a existência de um método de deteção de falhas em superfícies metálicas usando uma imagem em escala de cinzento, de autoria dos mesmos [19], construindo a sua proposta com base nele. Esse método divide a superfície das peças em blocos de igual tamanho, extraindo e analisando esses blocos para deteção das falhas. Para essa deteção é utilizado um método que examina a diferença e distribuição da luminância dos pixeis no bloco. A proposta deste artigo faz referência a um método que melhora e a deteção de falhas usando blocos.

O método proposto segue os seguintes passos:

- Captura da imagem.
- Transformação em escala de cinzento.
- Redução do ruído na imagem usando *smoothing*.
- Divisão dos blocos (figura 2.14).
- Extração dos blocos candidatos a falha, baseada na diferença do valor da luminância entre os blocos divididos.
- Decisão dos blocos com falha com base na distribuição dos pixeis nos blocos candidatos.

- Extração da região candidata a falha pela conexão dos blocos com falha.
- Decisão da falha através da área de pixeis escuros na região candidata.
- Decisão da falha através da concentração de pixeis escuros na região candidata.
- Separação das peças dependendo se têm falha ou não.

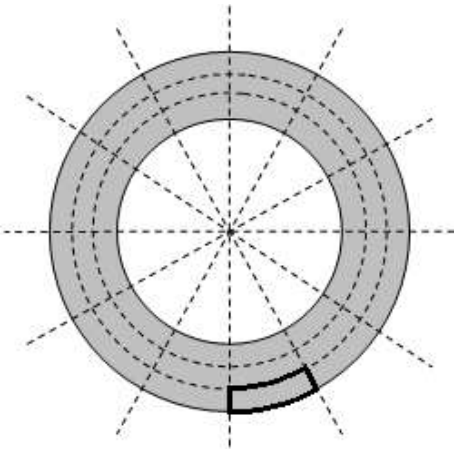


Figura 2.14: Divisão da peça em blocos [20]

Foi criado um sistema (figura 2.15) para realizar a inspeção automaticamente dividindo as peças defeituosas e as sem defeito. As peças entram na parte superior, onde um travão as imobiliza. Depois de inspecionada a peça atual, o travão abre e deixa uma nova peça cair para a zona onde é fotografada. A imagem é então enviada para um computador, onde é analisada e decidido se a peça é ou não defeituosa. Um travão é acionado com base na decisão e permite que a peça seja enviada para o local correto. Por fim a peça é deixada cair e o processo repete-se.

Este sistema foi considerado adequado para uma situação real de um local de produção através de testes realizados num prototipo, apresentado na figura 2.16.

2.5.3 Inspeção visual automática em problemas industriais

Este artigo [22] apresenta uma forma de detetar automaticamente defeitos em placas de cerâmica usando técnicas de processamento de imagem e de decisão automática. Foram estudadas formas de detetar três tipos de erros diferentes:

- Erro de impressão
- Bolhas
- Decalques

Este trabalho sugere um sistema de reconhecimento de padrões, tendo como *input* imagens das placas de cerâmica, sendo que pode ou não apresentar defeitos. O *output* do

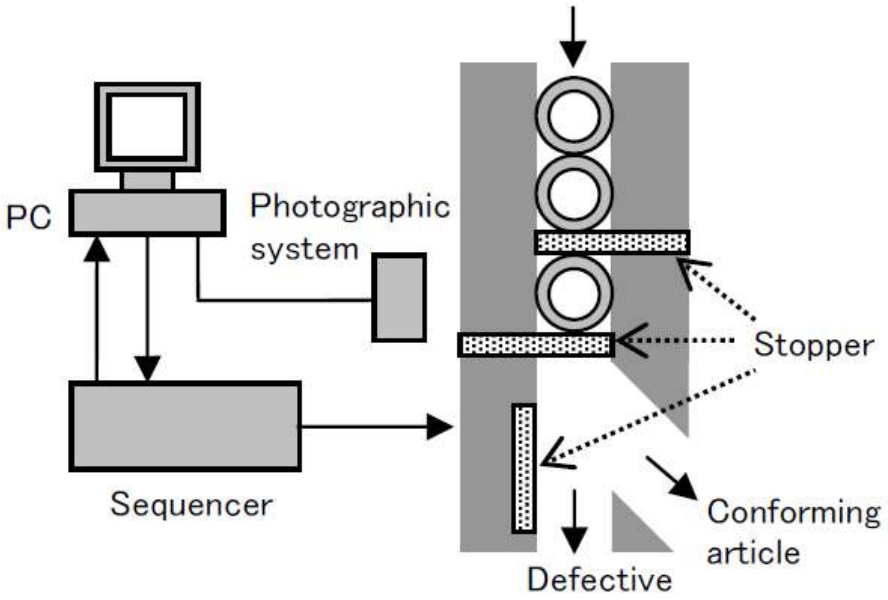


Figura 2.15: Arquitetura do sistema [20]

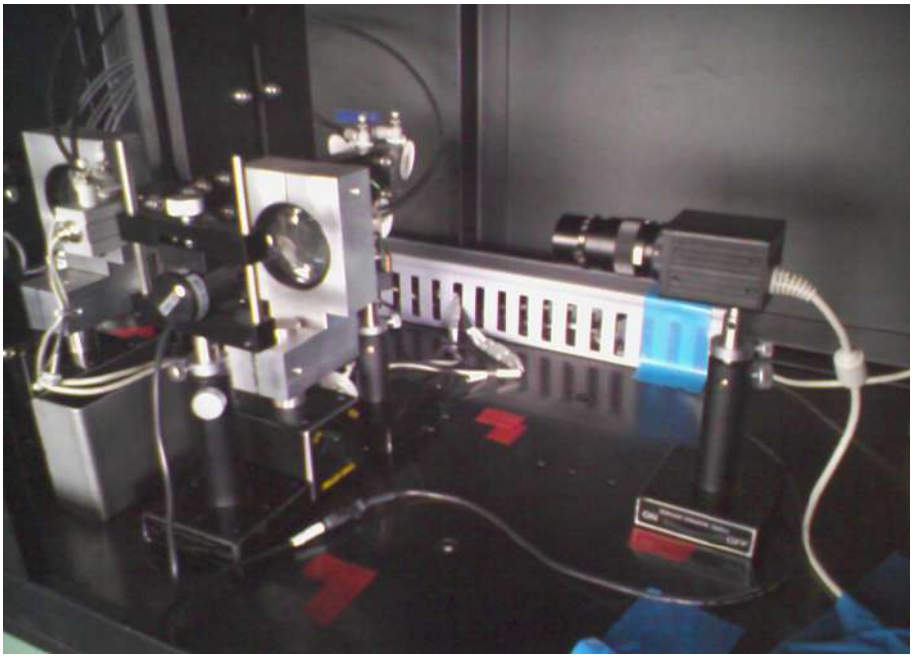


Figura 2.16: Prototipo do sistema [20]

sistema consiste nos resultados da identificação dos defeitos, indicando se uma placa em particular apresenta ou não esses mesmos defeitos. O sistema tem como base três pontos essenciais:

- Pré-processamento
- Extração de *features*
- Treino e identificação

O autor deste artigo comenta que um sistema automático como este, para resolver problemas industriais (como detetar defeitos), deve ter a capacidade de detetar diferentes tipos de defeitos, tendo em conta o tempo que este demora a realizar a tarefa e a necessidade de recursos computacionais. Além disso deve ser capaz de capturar imagens com qualidade suficiente para o problema em questão e ter em conta um numero grande de exemplos dos tipos de defeitos, de forma a ter uma maior taxa de sucesso.

Para o processamento das imagens, foram utilizadas técnicas como:

Binarização - Processo que permite transformar uma imagem RGB [4] numa imagem binária [5].

SIFT - Scale invariant feature transform. É um algoritmo que permite detetar e descrever características locais em imagens. Utilizado em reconhecimento de objetos, mapeamento, modelação e reconstrução 3D. É considerado dos melhores algoritmos para este tipo de problemas, tem como desvantagem o facto de ser um algoritmo lento.

SURF - Speeded Up Robust Features. É um detetor de características locais, utilizado em reconhecimento de objetos e reconstrução 3D. Este algoritmo foi criado com inspiração no SIFT, otimizando o tempo de execução do seu precedente.

A nível de técnicas para a automatização da decisão no sistema, o autor decidiu utilizar redes neuronais (figura 2.17), algoritmos inspirados na biologia do cérebro adquirindo conhecimento com base na experiência. Estes algoritmos são utilizados para a classificação de processos.

Foram criados dois protótipos (figura 2.18) para testar a captura de imagem no sistema. O importante a retirar das experiências feitas com os dois sistemas é o facto de a iluminação da imagem ser extremamente importante para conseguir imagens sem ruído e facilitar todo o processo. Uma má iluminação pode resultar numa falsa decisão por parte do sistema.

De acordo com o autor, a segmentação de imagem é um passo bastante importante em sistemas de inspeção automática. Técnicas de segmentação de imagem têm como objetivo identificar a posição de um objeto de interesse e isolar esse objeto de outros com menos

[4] Red, Green, Blue. Modelo de cores usado para a representação de imagens em sistemas eletrónicos
[5] Imagem digital que só tem dois valores possíveis para cada pixel

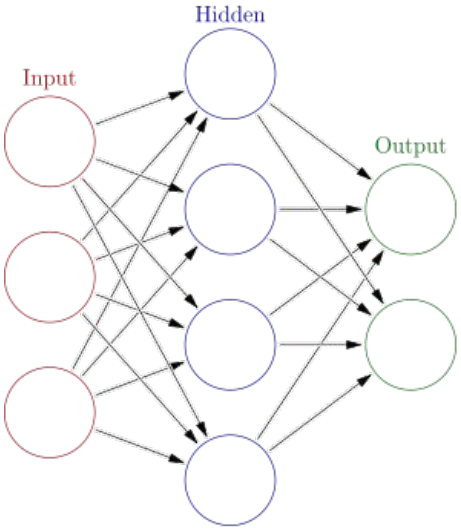


Figura 2.17: Diagrama de uma rede neuronal [12]

Figura 2.18: Protótipos do sistema de captura de imagem [12]

interesse, como o fundo da imagem, identificar regiões de interesse no objeto, tornando assim mais fácil a análise da a imagem.

Este trabalho foi realizado em MATLAB, devido à existência de bibliotecas com os algoritmos utilizados. Estas bibliotecas permitiram ao utilizador a manipulação dos dados que ele pretendia.

2.5.4 Comparação dos sistemas estudados

Todos os sistemas previamente referidos encontram-se numa fase de proposta, tendo sido realizados testes em alguns dos sistemas que já contam com protótipos funcionais.

O sistema descrito em 2.5.1 elaborou um algoritmo em MATLAB com a capacidade de testar se retrovisores contêm defeitos de distorção, comparando a sua reflexão com a reflexão de um retrovisor de controlo. Não foi construído nenhum protótipo do sistema

para realizar automaticamente este processo, foi apenas testado o algoritmo com imagens recolhidas para o propósito.

O sistema descrito em 2.5.2 criou um sistema com a capacidade de separar peças de travões com e sem defeito. Foi construído um protótipo com a capacidade de processar uma fila de espera de peças e dividi-las com base no resultado da inspeção. Este sistema foi testado em condições reais, numa fábrica de produção destas peças, sendo considerado adequado para uso.

Por fim, o sistema descrito em 2.5.3 introduz uma forma de detetar defeitos em placas de cerâmica. Foram utilizadas técnicas como a binarização e SURF para realizar o processamento das imagens. A decisão no sistema foi automatizada por meio de redes neuronais. Foram construídos dois protótipos com o objetivo de determinar qual o melhor sistema de captura de imagem. O algoritmo foi elaborado em MATLAB.

Capítulo 3

Especificação da solução

3.1 Descrição geral

O projeto, no qual esta dissertação se encontra inserida, tem como objetivo o desenvolvimento de uma aplicação que permite auxiliar na inspeção de veículos, automatizando parte dos processos associados à mesma. Para esse fim, esta dissertação consiste no desenvolvimento de algoritmos que permitam a automatização de tais processos, simplificando a tarefa de potenciais utilizadores.

Esta aplicação é composta por uma interface simples de linha de comandos, criada para a realização dos testes de validação dos diferentes algoritmos testados, sendo que não foi criada com o intuito de ser posteriormente utilizada por um utilizador, mas sim pela equipa de desenvolvimento.

O estudo desenvolvido nesta dissertação focou-se na extração de diferentes *features* de imagens, de forma a realizar *matching* ou classificação com base nos mesmos e verificar quais os que apresentam resultados mais favoráveis.

3.1.1 Processo de desenvolvimento

A aplicação desenvolvida para este estudo sofreu iterações incrementais com base na necessidade de testar os diferentes algoritmos estudados.

O ponto de partida corresponde ao estudo que abordou diferentes extrações de *features* e a realização de *matching* dos mesmos.

De seguida, avaliou-se novamente diferentes formas de extrair *features* e aplicar a algoritmos de classificação.

Para ambos os focos de estudo, foram realizados diferentes testes que permitiram obter métricas descritivas dos diferentes métodos utilizados.

3.2 *Matching* de imagens

Nesta secção são introduzidos os diferentes métodos de comparação de imagens estudados no decorrer desta dissertação. É feita uma descrição da utilização dos mesmos. Os resultados obtidos em cada método utilizado serão discutidos no capitulo seguinte.

3.2.1 Arestas

Existe uma imensa panóplia de métodos de deteção de arestas, tendo como objetivo a identificação de mudanças súbitas (formalmente, descontinuidades) no brilho entre um píxel e os da sua vizinhança. Estes pontos da imagem onde ocorrem estas mudanças organizam-se normalmente num conjunto de segmentos de reta, as arestas.

A deteção de arestas é uma ferramenta fundamental em processamento de imagem, em particular nas áreas de deteção e extração de *features*, visto que permite reduzir a quantidade de informação que se está a processar numa imagem, facilitando a filtragem de informação menos relevante inerente ao problema em questão, preservando ainda assim os dados importantes da estrutura de uma imagem.

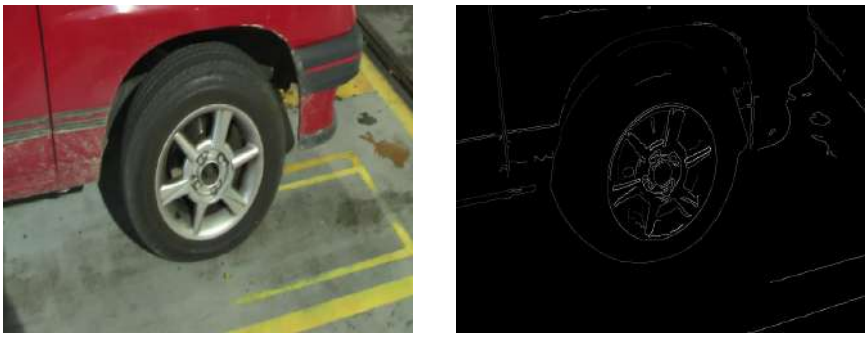
3.2.1.1 *Canny*

A primeira abordagem à extração desta *feature* foi a aplicação de um método de nome *Canny*. Na imagem 3.1 podemos observar o resultado da aplicação deste método a uma imagem.

O processo de deteção do algoritmo passa por 4 passos:

1. **Aplicação de filtro gaussiano** - remoção de ruído.
2. **Calcular o gradiente de intensidade da imagem** - magnitude e direção do gradiente são calculados para cada ponto. Um gradiente alto corresponde a uma mudança brusca de brilho, correspondendo a uma aresta. Um gradiente baixo implica não ter existido mudanças substanciais, logo não se configura numa aresta.
3. **Supressão não máxima** - se um píxel não é máximo na sua vizinhança é suprimido. Isto permite reduzir o segmento de reta que representa a aresta a apenas uma linha de pontos.
4. ***Hysteresis thresholding*** - permite limitar as arestas sendo as mesmas consideradas com base em dois valores de *threshold* definidos, um máximo e um mínimo. Píxeis com um gradiente maior que esse máximo são considerados arestas. Caso apresentem um gradiente inferior ao mínimo, não são considerados. Os que se encontrem entre *thresholds* são considerados arestas ou não por meio de conectividade.

A métrica aplicada neste caso corresponde à distancia absoluta entre as duas listas de valores obtidas pela variação dos parâmetros do algoritmo, resultando num valor



Pré Canny Pós Canny

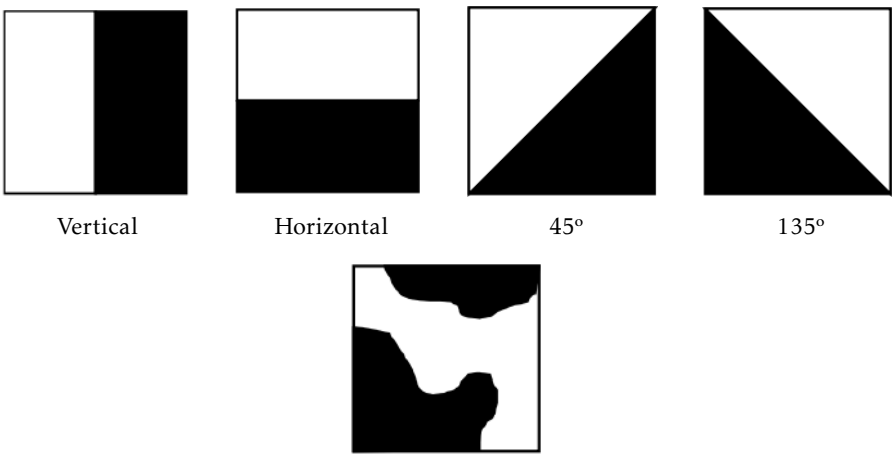
Figura 3.1: Resultado de aplicar *canny*

indicativo da semelhança da média e variância das duas imagens. A proximidade de zero, para cada um destes valores, indica um maior grau de semelhança entre as imagens.

3.2.1.2 *Edge histogram*

Um histograma de arestas corresponde à distribuição das arestas na imagem. Este algoritmo foi criado segundo o método descrito no artigo [29].

Começou-se por dividir as imagens em n blocos (16 no teste em questão, 4x4) calculando para cada um desses blocos a força das arestas em 5 orientações (3.2).



Vertical Horizontal 45º 135º

Sem direção

Figura 3.2: Orientação das arestas

Com base nas matrizes geradas pelo passo anterior, foram calculadas a média e a variância das mesmas, obtendo uma lista de 80 valores para cada uma. Este processo foi aplicado a uma imagem de referencia e a diferentes imagens de teste, obtendo a listagem de valores para ambas.

A métrica aplicada neste caso corresponde à distancia absoluta entre as duas listas de valores, resultando num valor indicativo da semelhança da média e variância das duas

imagens. A proximidade de zero, para cada um destes valores, indica um maior grau de semelhança entre as imagens.

3.2.1.3 *Gabor filter*

Os filtros de gabor têm aplicação tanto na deteção de arestas, como na análise de textura de imagens. A sua utilização é comum para encontrar padrões na escrita em documentos. Podem ser aplicados filtros com diferentes frequências e orientações para localizar padrões em imagens. São também utilizados em situações de reconhecimento facial ou reconhecimento de impressões digitais. Na figura 3.3 podemos ver o resultado da aplicação destes filtros a uma imagem. Neste caso é aplicado dois filtros com o parâmetro theta diferente, obtendo imagens que apresentam orientações distintas.

Figura 3.3: Resultado da aplicação de filtros gabor

O estudo efetuado com estes filtro consistiu na aplicação de filtros gerados com diferentes parâmetros de *kernel size*, sigma, theta, lambda, gamma e psi. Todos os parâmetros exceto o sigma e o theta foram fixos com base em valores referidos na documentação do algoritmo, sendo que estes últimos dois parâmetros sofreram variação.

A métrica aplicada nesta situação foi semelhante à referida em 3.2.1.2, onde foram geradas listas de valores médios e variância, posteriormente calculando a diferença entre os valores referentes a duas imagens.

3.2.1.4 *Wavelet transform*

A *wavelet transform* é uma forma de compressão de imagem, permitindo remover redundâncias da imagem, preservando no entanto as características originais da mesma. Na figura 3.4 podemos observar o resultado da decomposição de uma imagem utilizando este método.

Figura 3.4: Resultado da aplicação de *wavelet transform*

No estudo deste método foi utilizada a transformada discreta de wavelet para fazer uma decomposição da imagem em 4 regiões, uma de baixa frequência (LL), e 3 de alta frequência (LH, HL e HH – horizontal, vertical e diagonal) como representado na figura 3.5.

As métricas utilizadas para avaliar este método são novamente as previamente utilizadas em 3.2.1.2 e 3.2.1.3 utilizando a média e a variância obtidas da matriz representante de cada uma das diferentes regiões e uma métrica de semelhança desenvolvida por *Zheng Dezhong* e *Cui Fayi* no artigo *Face Recognition based on Wavelet Transform and Image Comparison* [16]. Esta métrica corresponde ao calculo de

$$\sum_{i=1}^{k} 255 - |m(i) - n(i)|$$

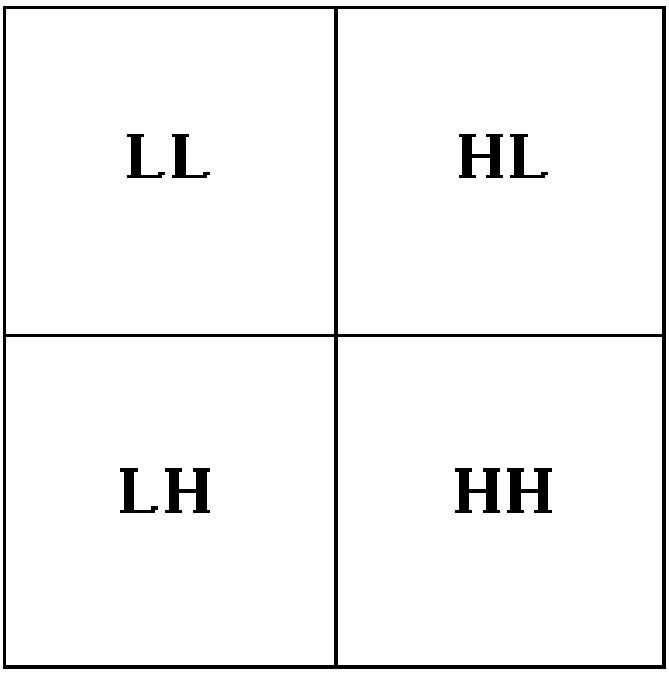


Figura 3.5: Distribuição da decomposição usando *wavelet transform*

ou seja, o somatório da diferença entre o valor máximo de um píxel numa imagem em grayscale, 255, e o valor absoluto da diferença entre os valores dos píxeis das imagens m e n. Esta formula permite obter uma percentagem, 0 a 100%, de semelhança entre as duas imagens a serem comparadas.

3.2.2 *Matching* de *features* e homografia

Feature matching consiste na comparação de descritores de *features* recolhidos de duas imagens através de um detetor como *SURF* ou *SIFT*. O descritor de um determinado *feature* de uma imagem é comparado com todos os outros *features* de outra imagem utilizando uma métrica de distância, retornando o que apresenta a menor distância (maior semelhança entre eles).

Existem diferentes *matchers* no OpenCV, sendo que os que foram utilizados no decorrer desta dissertação foram os seguintes:

- *Fast Library for Approximate Nearest Neighbors (FLAAN) matcher*
- *Brute-Force (BF) matcher*

Apesar de terem uma funcionalidade idêntica, a forma como o fazem difere de um para o outro. Como o nome indica, *brute-force matcher* experimenta todas as possibilidades existentes, retornando exatamente as melhores correspondências entre *features*. Por outro lado, o *flann matcher* funciona com base numa aproximação dos melhores *features* numa vizinhança, o que significa que retorna uma boa aproximação da melhor correspondência possível.

Por outras palavras, a utilização de *flann matching* permite maior velocidade de execução visto que não procura todas as possíveis combinações. Em contrapartida as correspondências encontradas podem não corresponder à melhor possível. Por sua vez, o *brute-force matcher* funciona de forma oposta. De forma a obter resultados de comparação mais robustos, troca a velocidade de execução pela obtenção do melhor resultado possível.

Homografia é a transformação que mapeia os pontos de uma imagem aos pontos correspondentes noutra imagem. A figura 3.6 permite ver os diferentes pontos correspondentes em ambas as figuras.

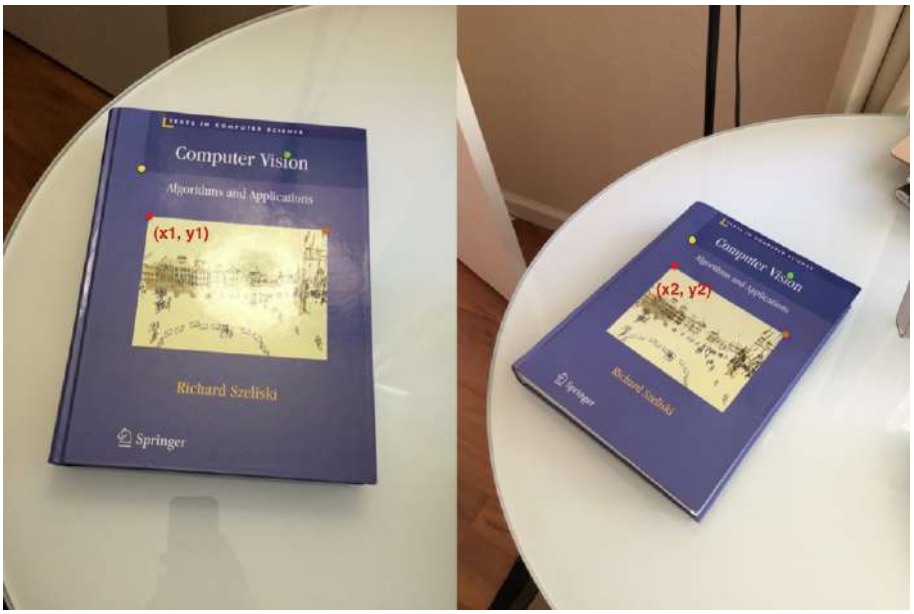


Figura 3.6: Homografia - livro colocado em posições diferentes [18]

A homografia é utilizada para diversos fins, como por exemplo alinhamento de duas imagens ou para a criação de fotos panorâmicas. No caso desta dissertação, foi utilizada para sinalizar a localização de uma peça de um automóvel numa dada imagem.

Extração de *features* No decorrer dos testes de *matching* foram utilizados os seguintes extratores de *features*:

- *Scale-invariant Feature Transform (SIFT)*
- *Speed Up Robust Features (SURF)*
- *Oriented FAST and rotated BRIEF (ORB)*

Este estudo foi realizado com o objetivo de verificar a possibilidade de detetar peças de automóveis contidas numa dada imagem.

O algoritmo criado para o teste começa por detetar os diferentes pontos de interesse nas duas imagens fornecidas. De seguida, esses pontos de interesse são comparados utilizando um *matcher*. De forma a selecionar apenas os melhores *matches*, é calculada a menor distância entre pontos de interesse, de forma a efetuar uma seleção com base na vizinhança desses pontos. Neste caso, foram apenas aceites os *matches* que tivessem um valor igual ou inferior a um múltiplo da distância mínima. Estes *matches* são desenhados nas imagens para visualizar a correspondência encontrada

Depois de encontrar os bons *matches*, é calculada a matriz de transformação da homografia, de forma a permitir calcular os cantos da imagem da peça dentro da imagem e desenhar uma linha que representa a localização dessa peça.

Na figura 3.7 podemos ver o resultado de este processo.

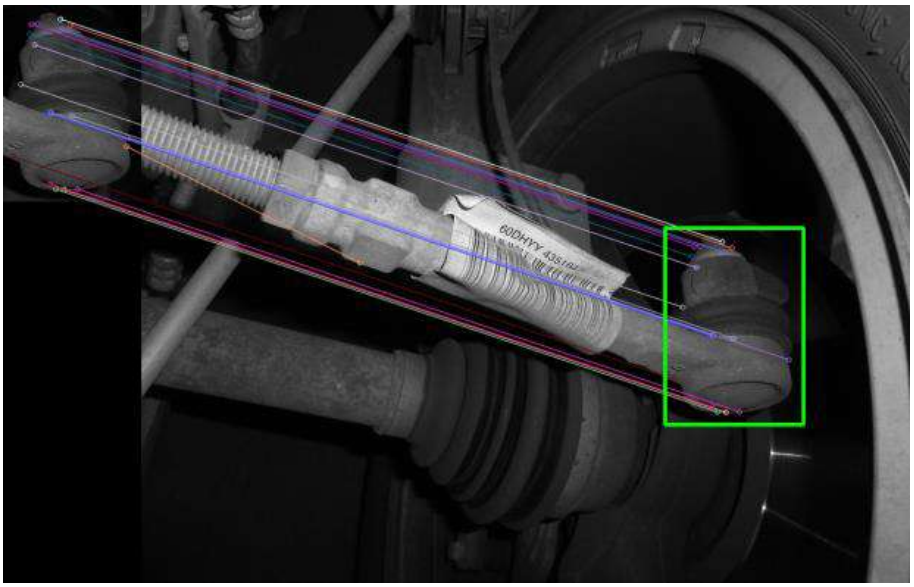


Figura 3.7: Resultado da procura de uma peça na cena

3.3 Classificação de dados

A solução proposta para a classificação utilizada nos testes que foram efetuados nesta dissertação partiu de uma proposta efetuada por um colega do professor Nuno Correia. Esta solução combina a utilização de *Bag-of-words (BoW)* para a criação de descritores de imagem que possam ser utilizados para treinar um classificador *SVM*.

BoW **[13]** *Bag-of-words* é um modelo utilizado em processamento de linguagem natural, sendo utilizado em situações de categorização de documentos. O objetivo é analisar e classificar os diferentes *bag-of-words* presentes nos documentos e ao comparar as diferentes categorias, identificar a que *bag* um determinado texto pertence.

Esta ideia pode não só ser aplicada a texto, como também ao conceito de visão por computador. Este modelo pode ter aplicação na área de classificação de imagens ao considerar

as diferentes *features* como palavras. Assim, em visão por computador, um *bag-of-visual-words* é um vetor de ocorrências de um vocabulário de *features*.

SVM **[14]** Como muitos outros algoritmos de aprendizagem automática, *Support vector machines* utilizam um conjunto de dados já classificados, o conjunto de treino, e tentam prever a que classe pertence um dado conjunto de dados não classificados, o conjunto de teste.

As *SVM* tentam encontrar um híper-plano ótimo que consiga separar duas ou mais classes de um conjunto de dados. Se separáveis, existe um número infinito de híper-planos que permitem separar o conjunto de dados, como podemos ver na figura 3.9. O híper-plano ótimo é escolhido de forma a que a distância do mesmo ao ponto mais próximo das classes ponderadas seja máxima, como podemos ver na figura 3.9.

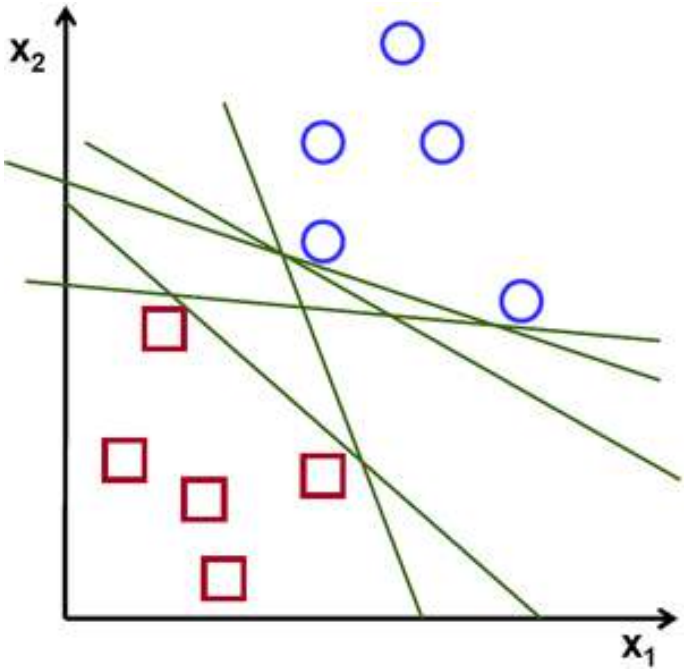


Figura 3.8: Híper-planos que separam duas classes distintas [28]

O nome deste algoritmo provém do facto de os pontos que se situam na margem do híper-plano ótimo serem os vetores de suporte.

As figuras representam um exemplo simples, apenas separando duas classes distintas para melhor compreensão. Na realidade, o nome híper-plano provém do facto deste plano separar conjuntos de dados em dimensões que a mente humana não consegue visualizar.

O algoritmo criado para os testes realizados passa pelas seguintes fases durante o treino do classificador:

1. Extração de todos os *features* de todas as imagens do conjunto.
2. Obter o descritor para cada um desses *features* (este passo e o anterior utilizam *SURF, SIFT ou DENSE SIFT*).

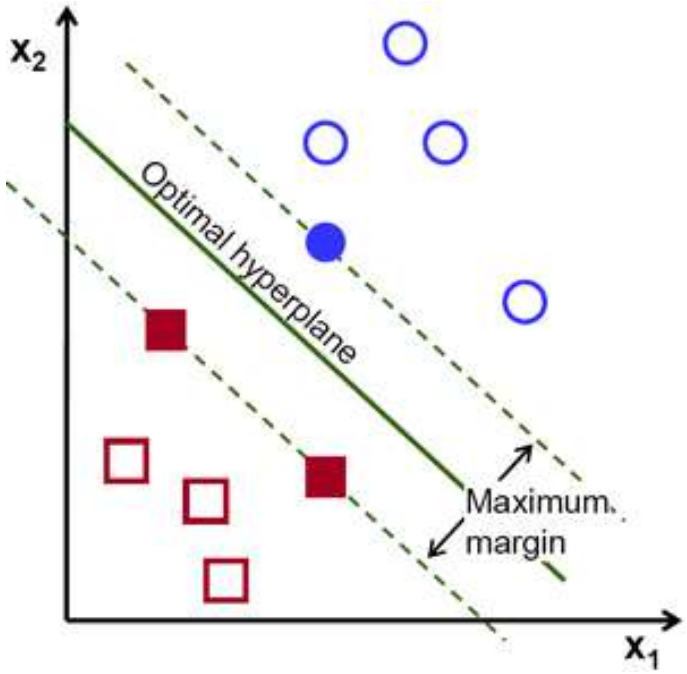


Figura 3.9: Híper-planos ótimo que separa as duas classes [28]

3. Obter um vocabulário realizando *clustering* do conjunto de descritores de *features* (utilizando a função de *cluster* presente na classe de *BoW* utilizada). Este *clustering* é efetuado com base no algoritmo *K-Means*.

4. Obter os descritores de cada imagem contida dentro das classe existentes, com recurso ao vocabulário gerado e criar um conjunto de treino. Este conjunto de treino consiste na *label* associada ao descritor de uma determinada imagem.

5. Treinar o classificador *SVM* com este conjunto de treino.

Por sua vez, para classificar as imagens de um dado conjunto de teste, o algoritmo executa os seguintes passos:

1. Extração dos *features* de uma dada imagem com recurso a *SURF, SIFT ou DENSE SIFT*.

2. Obtenção de descritores de *features* utilizando o vocabulário obtido previamente.

3. Utilização dos descritores na função de *predict* do classificador de forma a obter as *labels* previstas.

3.4 *Histogram equalization*

O método de *histogram equalization* permite aumentar o contraste global de imagens, especialmente se os valores de intensidade forem semelhantes. Ao aplicar este método,

as intensidades são distribuídas de forma mais homogénea, como podemos verificar nos histogramas presentes na figura 3.10, permitindo assim que ocorra uma melhoria nas zonas de baixo contraste, como podemos ver na figura 3.11.

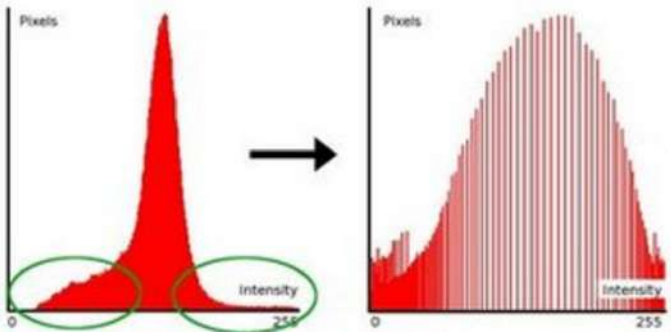


Figura 3.10: Representação gráfica do resultado de *histogram equalization* [17]

Antes Depois

Figura 3.11: Resultado da aplicação de *histogram equalization*

3.4.1 *CLAHE*

Contrast Limited Adaptive Histogram Equalization (CLAHE) consiste noutra forma de *histogram equalization*, mas ao contrário do método anterior que considera o contraste global da imagem, o *CLAHE* divide a imagem em blocos de pequenas dimensões. Posteriormente, estes blocos sofrem *histogram equalization*, garantindo um enaltecimento do contraste mais detalhado aos objetos presentes em cada bloco. A figura 3.12 representa o resultado de aplicar *CLAHE* à imagem utilizada na figura 3.11.

Como podemos ver, o *CLAHE* apresenta resultados com maior contraste em relação a *histogram equalization*.

O objetivo do uso destas técnicas de aumento de contraste, provém da necessidade de aumentar o número de *features* recolhidas pelos diferentes extratores. Ao aumentar o contraste, são criadas mais zonas de mudanças súbitas entre as diferentes regiões da imagem, facilitando a deteção de *features* na mesma. No caso do *matching* de features

Antes Depois

Figura 3.12: Resultado da aplicação de *CLAHE*

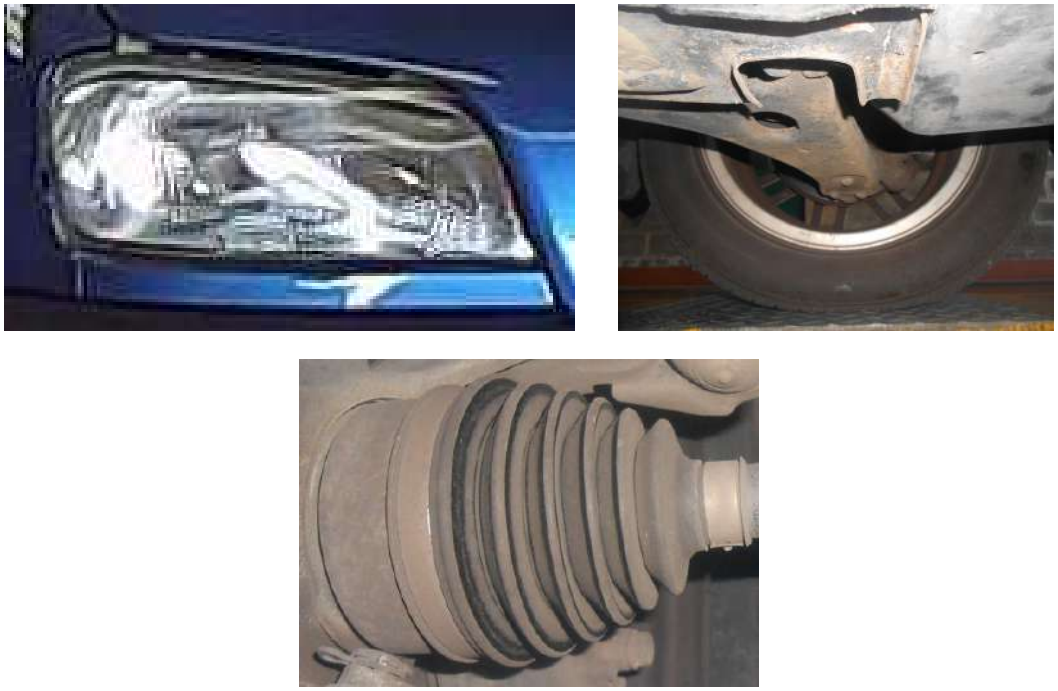


Figura 3.13: Exemplos de imagens fornecidas

3.2.2, o uso destas técnicas permitiu o aumento de bons *matches* quando comparado com o uso da imagem original.

3.5 Fontes de dados

Os dados utilizados no decorrer desta dissertação consistem em imagens recolhidas de diversas fontes na internet e conjuntos de imagens fornecidos pela equipa do Centro de Inspeções de Vila Nova de Poiares. Estas imagens correspondem a fotos de diferentes peças de automóveis, como as da figura 3.13.

O conjunto de imagens que foi fornecido aumentou ao longo do tempo, sendo que a equipa foi recolhendo fotos de diferentes veículos que deram entrada no centro de inspeções. Esta variação do conjunto disponível levou a um condicionamento de alguns testes realizados, sendo que testes realizados mais perto da data final da dissertação dispõem de melhores e maiores conjuntos de dados.

Capítulo 4

Resultados

Neste capitulo são apresentados diversos resultados de testes às diferentes técnicas referidas no capitulo 3.

4.1 *Matching* de imagens

Os primeiros testes realizados abordaram as técnicas de *Canny*, *Edge Histogram*, *Gabor Filter e Wavelet Transform*. Os mesmos utilizaram como imagens de teste as referidas na figura 4.1.

Figura 4.1: Imagens de teste utilizadas. À esquerda a imagem comparada com os outros conjuntos, em cima as imagens positivas, em baixo as imagens negativas

Utilizar um conjunto igual para os diferentes testes, permite efetuar comparações dos resultados obtidos pelas diferentes técnicas. Esta comparação não pode ser efetuada a nível de valores, visto que não existe uma relação direta entre valores de média e variância obtidos.

Estes testes permitem verificar se existe algum *threshold* nos valores obtidos que permita diferenciar as imagens que estão no conjunto positivo, das que estão no conjunto

negativo.

4.1.1 *Canny*

Os resultados obtidos ao utilizar *Canny* não permitem observar uma separação entre imagens positivas e negativas. Como podemos ver na figura 4.2, não é possível indicar uma clara distinção entre os valores apresentados para o grupo positivo e negativo, sendo que podemos observar imagens positivas com valores maiores do que as negativas e vice-versa. Isto ocorre para ambos os parâmetros utilizados (**Média** e **Variância**).

Por exemplo, a imagem Pos4 apresenta um valor de média maior do que a imagem Neg7. Este facto indica que o algoritmo determinou que uma imagem do grupo positivo tem um grau de semelhança inferior ao de uma imagem do grupo negativo.

Canny	Média	Variância
Pos 1	18.0323	18.7815
Pos 2	44.8693	45.2957
Pos 3	30.4286	35.6222
Pos 4	72.5484	79.7486
Pos 5	52.3987	63.6101
Pos 6	58.0845	62.9625
Pos 7	53.5335	58.9627
Neg 1	76.2062	83.7997
Neg 2	50.315	57.9428
Neg 3	53.0728	59.6857
Neg 4	42.7304	57.0603
Neg 5	92.5262	125.904
Neg 6	22.6423	20.0375
Neg 7	32.2088	34.5814

Figura 4.2: Resultados do teste com Canny

4.1.2 *Edge histogram*

Com a aplicação de *Edge histogram* foram obtidos resultados muito próximos do objetivo. Ainda assim, é possível observar na tabela 4.3 que existem imagens com valores muito próximos da separação de ambos os grupos, criando uma ténue linha onde não é possível determinar se uma imagem pertence ao conjunto positivo ou negativo.

Exemplos deste problema são as imagens Pos4 e Neg7, que apresentam valores que as colocariam em conjuntos opostos ao que na realidade elas pertencem.

Edge Histogram	Média	Variância
Pos 1	17.298	43.5413
Pos 2	24.02	52.0974
Pos 3	26.9289	65.1759
Pos 4	35.7109	79.8306
Pos 5	30.6312	68.1597
Pos 6	31.0992	69.8698
Pos 7	28.6638	63.9444
Neg 1	34.3419	75.3422
Neg 2	33.456	74.2008
Neg 3	31.8449	73.2997
Neg 4	34.1973	83.5129
Neg 5	41.5987	90.1854
Neg 6	30.5213	65.7492
Neg 7	29.9015	53.6157

Figura 4.3: Resultados do teste com Edge Histogram

4.1.3 *Gabor filter*

Ao aplicar *gabor filter* para obter resultados, verificou-se aleatoriedade entre os valores obtidos para cada imagem, impedindo, com base tanto na média como na variância, qualquer tipo de separação do grupo de imagens positivas e negativas.

Esta técnica apresentou os piores resultados das testadas com este conjunto de dados, apesar de diversas alterações nos parâmetros.

Se observarmos os valores da tabela 4.4, com base em qualquer um dos dois parâmetros, não é possível determinar nenhum *threshold* que separe os dois conjuntos.

Apesar de modificações aos parâmetros do algoritmo e a realização de diferentes testes, não foi possível obter resultados que se aproximassem aos dos outros três métodos aqui discutidos.

4.1.4 *Wavelet transform*

Os resultados obtidos na figura 4.5 provém da aplicação de *wavelet transform* às diferentes imagens. Como podemos observar nesses resultados, em qualquer um dos três parâmetros utilizados para a comparação, não existe um limite que separe as imagens de ambos os conjuntos, sendo que uma ou outra imagem apresenta valores que a colocaria no conjunto oposto.

Ao observar apenas o parâmetro **Semelhança** conseguimos entender que esta métrica

Gabor Filter	Média	Variância
Pos 1	979.26	598.262
Pos 2	61.2968	1282.68
Pos 3	1214.48	1617.38
Pos 4	2738.84	2143.28
Pos 5	1615.16	1999.45
Pos 6	3358.99	1628.06
Pos 7	145.498	1618.78
Neg 1	493.569	1916.7
Neg 2	1177.82	1826.39
Neg 3	729.158	2035.06
Neg 4	1279.57	1912.01
Neg 5	3102.18	2553.32
Neg 6	2903.49	1628.75
Neg 7	189.391	1074.67

Figura 4.4: Resultados do teste com Gabor Filter

indica percentagens de semelhança bastante elevadas para imagens que apresentam poucas características semelhantes à imagem de comparação (por exemplo, a imagem Neg2).

4.1.5 Comparação resultados

Com base nos resultados obtidos nos métodos discutidos nas secções anteriores, podemos observar que o objetivo inicial de uma métrica que permitisse separar as imagens pertencentes a ambos os grupos não foi alcançado.

Como os resultados não cumpriram o objetivo, é possível dizer que nenhuma destas técnicas utilizadas tem vantagem em relação à outra. No entanto ao fazer uma aproximação de quais as que estiveram mais próximas de cumprir o objetivo, as técnicas discutidas em 4.1.2 e 4.1.4 apresentaram resultados superiores.

Com base nestas experiências, foi possível determinar que esta aproximação não é a melhor, visto que não foi possível obter resultados convenientes ao objetivo.

4.1.6 *Matching* de *features* e homografia

A realização de testes utilizando *matching* de imagens e homografia permitiu explorar a possibilidade de detetar imagens (neste caso, certos componentes de automóveis) noutra

Wavelet Transform	Média	Variância	Semelhança
Pos 1	3.34027	8.60059	77.5298
Pos 2	4.9292	12.2947	76.6896
Pos 3	6.08172	14.8571	85.3576
Pos 4	8.44917	19.7032	67.8501
Pos 5	6.99513	14.9142	77.6327
Pos 6	7.75376	17.3606	69.7971
Pos 7	6.50237	13.9218	64.0192
Neg 1	8.63191	17.8943	73.7937
Neg 2	9.18125	19.996	79.8925
Neg 3	8.92397	21.7348	71.0555
Neg 4	8.35241	20.8807	71.0718
Neg 5	11.3596	25.838	74.5228
Neg 6	8.16443	18.867	69.8404
Neg 7	5.94221	10.0605	64.2111

Figura 4.5: Resultados do teste com Wavelet Transform

imagem (normalmente veículos de diversos ângulos). Isto permite perceber se existe a possibilidade de identificar e destingir os diferentes componentes de um veículo entre si.

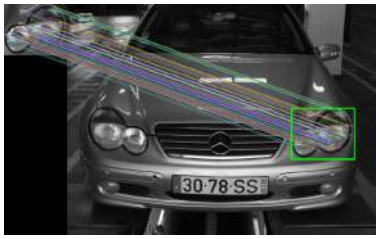
O primeiro teste utilizou a técnica referida em 3.2.2 na tentativa de identificar certas peças num conjunto de imagens de veículos. As imagens utilizadas correspondem a uma transformação para *greyscale* das imagens originais.

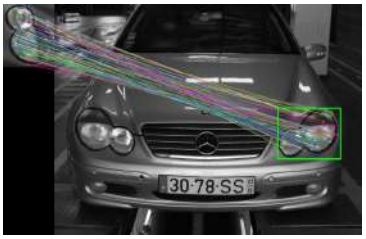
Foram obtidos resultados maioritariamente positivos. Como podemos observar na figura 4.6 ao fazer a comparação de uma imagem contida na cena, o algoritmo consegue identificar a sua localização na imagem corretamente. Podemos ainda observar que a imagem da peça pretendida, pode ter sofrido rotações ou ser alterada a escala, mostrando robustez a nível destes parâmetros.

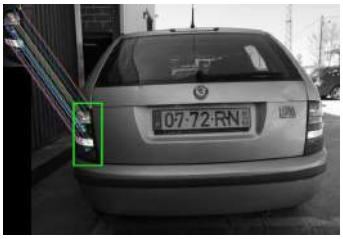
Por outro lado, ao comparar uma imagem que não se encontra contida na cena, como é feito na figura 4.7, o algoritmo consegue determinar que não existem bons pontos de referencia para desenhar o retângulo que representa a homografia, indicando que a peça em questão não está presente no veículo.

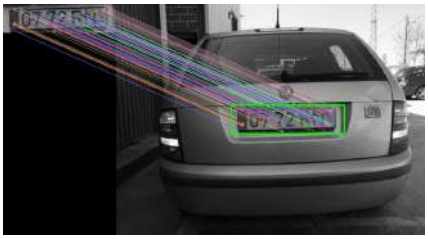
Apesar de bastante positivos, estes resultados não estão isentos de falhas. Como se pode ver na figura 4.8, a imagem que apresenta a comparação de um depósito no bloco do motor encontra diversos bons *matches* na região correta, mas é incapaz de desenhar o retângulo da homografia na posição certa. Este resultado foi um falso positivo, visto que apesar de desenhar a homografia na posição errada, passou por uma verificação que indica se esta é ou não boa.

Por fim, temos o exemplo representado pela outra imagem da figura, em que a peça que está claramente contida na figura, não apresenta quaisquer bons *matches*.


Farol dianteiro


Farol dianteiro rodado


Farol traseiro


Matricula traseira


Matricula dianteira

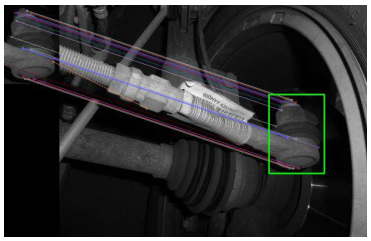

Rotula direção

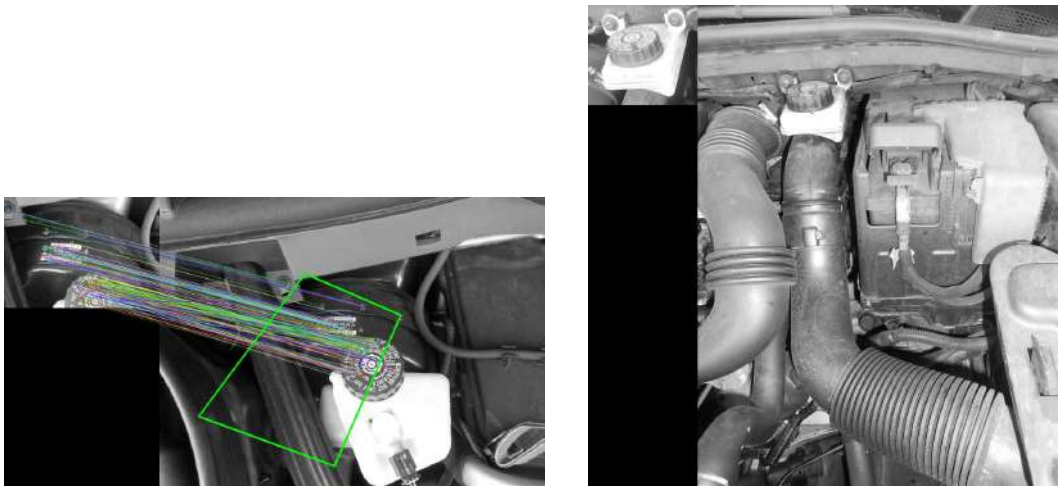
Figura 4.6: Resultados positivos da aplicação de *matching* com homografia


Carro e rotula direção


Rotula direção e farol dianteiro


Rotula direção e farol traseiro

Figura 4.7: Outros resultados positivos da aplicação de *matching* com homografia



Depósito Tampa

Figura 4.8: Resultados negativos da aplicação de *matching* com homografia

Como estes testes não tiveram a precisão necessária, foram realizados novos testes alterando diversos parâmetros. Foram aplicados os algoritmos referidos em 3.4.1 com o objetivo de melhorar o contraste da imagem e possibilitar um aumento de bons e distintos *features* disponíveis, sendo que estes *features* foram extraídos com a utilização de SIFT, SURF e ORB. Foram também utilizados outros métodos para a comparação dos diferentes *features* como os referidos em 3.2.2.

No anexo I pode ver-se os diferentes resultados dos parâmetros testados neste conjunto de imagens. Os resultados são bastante diversos, mas uma conclusão que se pode retirar é o facto de a forma mais eficiente de extrair *features* ser a utilização de *SIFT*. É o algoritmo mais consistente a obter bons *matches* em praticamente todas as imagens. Por outro lado, ao observar os resultados obtidos com *ORB* ou *SURF*, vemos que existe uma grande quantidade de *matches* incorretos (I.1 - *ORB*), ou simplesmente a ausência dos mesmos (I.2 - *ORB*).

Outra conclusão que podemos tirar é que modificar o histograma de cor da imagem, permite, para algumas imagens, aumentar o número de bons *features* recolhidos e por sua vez, o número de bons *matches* encontrados. Um exemplo disto é a imagem I.1, na parte associada a *SIFT*, podemos ver um incremento na quantidade de bons *matches*.

O inverso também se observa, na imagem I.4, na parte associada a SIFT, o número de *matches* não só reduz, como introduz um *match* incorreto.

Apesar de ser vantajoso na generalidade das imagens a aplicação de *histogram equalization* ou *CLAHE*, a melhoria de resultados não se observa em todas as imagens testadas.

Contudo, mesmo com a alteração de diversos parâmetros, continua a não ser possível a identificação de peças em certas imagens, como na figura I.5. O ruído existente do resto dos componentes do veículo torna difícil o reconhecimento das características correspondentes à peça que se tenta identificar.

4.2 Classificação de dados

Os testes de classificação realizados utilizaram um conjunto de imagens fornecidas pela empresa. As imagens correspondem a diferentes peças do domínio, como faróis ou rodas. Com o objetivo de reduzir o ruído em cada imagem, estas foram recortadas de forma a diminuir o aparecimento de outras peças situadas à volta da peça alvo.

As classes utilizadas para estes testes foram as seguintes:

Classe 1 - Disco

Classe 2 - Farol dianteiro

Classe 3 - Farol traseiro

Classe 4 - Fole transmissão

Classe 5 - Mola

Classe 6 - Roda

Classe 7 - Rótula direção

Classe 8 - Rótula suspensão

Classe 9 - Silenciador

Classe 10 - Fole direção

A figura 4.9 mostra um exemplo de imagens pertencentes a cada uma das classes.

Estes testes foram realizados com um conjunto de treino com 10 classes, sendo que cada classe tem 11 imagens. Portanto o conjunto de treino é representado por 110 imagens. O conjunto de teste contem 71 imagens, sendo que algumas são do conjunto de onde o treino foi retirado (mas distintas das utilizadas no treino) e outras retiradas de fontes alternativas.

Os resultados da classificação deste conjunto de imagens estão representados na forma de uma matriz de confusão e do calculo da *Precision*[1] e *Recall*[2] de cada classe.

Os resultados apresentados na figura 4.1 correspondem aos valores *default* dos diferentes algoritmos *SURF, SIFT e DENSE SIFT*.

Observando os resultados da tabela 4.1 e os presentes no anexo II, podemos verificar que o *DENSE SIFT* apresenta resultados irregulares, indicando um grande número de classificações na classe 1. Como tal, a sua *precision* e *recall* apresentam valores baixos (na sua maioria 0) exceto a classe 1, visto que classificou a grande maioria do conjunto de teste como 1. É provável que este erro de classificação seja proveniente do ruído presente nas imagens de treino. Ao criar uma rede de *keypoints* em toda a imagem com um espaçamento

[1] número de previsões positivas corretas dividido pelo total de previsões positivas (TP / TP + FP)
[2] número de previsões positivas corretas dividido pelo total de positivos (TP / TP + FN)

Figura 4.9: Exemplo de imagens de cada classe

Default		SURF	SIFT	DENSE		SURF	SIFT	DENSE	
		Recall	Recall	Recall		Precision	Precision	Precision	
Classe 1		0.857143	0.285714	1.000000		0.375000	0.285714	0.118644	
Classe 2		0.000000	0.222222	0.000000		0.000000	0.400000	0.000000	
Classe 3		0.166667	0.166667	0.000000		1.000000	1.000000	0.000000	
Classe 4		0.000000	0.333333	0.000000		0.000000	0.400000	0.000000	
Classe 5		0.555556	0.333333	0.000000		1.000000	0.600000	0.000000	
Classe 6		1.000000	0.833333	0.500000		0.387097	0.357143	0.545455	
Classe 7		0.142857	0.142857	0.000000		0.333333	0.333333	0.000000	
Classe 8		0.000000	0.200000	0.000000		0.000000	0.125000	0.000000	
Classe 9		0.428571	0.285714	0.000000		0.750000	0.666667	0.000000	
Classe 10		0.000000	0.333333	0.000000		0.000000	0.166667	0.000000	

Tabela 4.1: *Precision* e *Recall* para parâmetros *default*

predefinido, são criados *keypoints* em partes da imagem que correspondem a ruído (o fundo, peças que não são o alvo dessa classe, entre outros). Como tal, o vocabulário contem palavras relativas a este ruído, provocando a classificação errada das imagens.

Entre *SIFT* e *SURF* podemos observar que o *SIFT* tem resultados mais constantes, tendo em média, maior taxa de classificação correta em todas as classes. SURF, por sua vez, apesar de apresentar uma taxa de classificação correta maior em certas classes (1, 5, 6, 9), classifica incorretamente múltiplas outras classes. Para diferentes classes, ambos têm as suas vantagens.

A tabela 4.2 apresenta valores referentes aos diferentes parâmetros do algoritmo *SURF*.

Ao variar o parâmetro *hessian threshold* é possível reduzir o número de *keypoints* detetados (o critério de se um ponto é um *keypoint* ou não fica mais rigoroso). Por defeito, no OpenCV este valor é de 100, mas de acordo com a documentação, um valor entre 300 a 500 pode ter melhores resultados. O parâmetro *upright* determina se a orientação de cada *feature* é calculada ou não.

Ao observar os resultados com o *hessian* a 300, podemos ver que o facto de ter o parâmetro *upright* ativo não influencia os resultados. No nosso conjunto de imagens, provavelmente, as imagens a testar vão ter sempre a mesma orientação, sendo que não existe a necessidade de calcular a orientação dos *features* das mesmas. Isto permite reduzir bastante o tempo de execução do treino e classificação.

Podemos observar que os resultados são bastante dispersos e dependendo da classe, obtém-se melhores ou piores resultados, mantendo os valores bastante semelhantes entre os diferentes parâmetros. Existe o problema expresso, na classe 8, ao utilizar o *hessian* a 500, obtemos uma *precision* e *recall* igual a 0, sendo que por outro lado, na classe 10 ocorre uma melhoria para 0,66.

As tabelas presentes em 4.3 são referentes a resultados provenientes da variação de parâmetros do algoritmo *SIFT*.

SURF	Recall				Precision			
	Default	Hessian 300	Hessian 300 Upright True	Hessian 500	Default	Hessian 300	Hessian 300 Upright True	Hessian 500
Classe 1	0.857143	0.857143	0.857143	0.571429	0.375000	0.300000	0.300000	0.200000
Classe 2	0.000000	0.000000	0.000000	0.111111	0.000000	0.000000	0.000000	1.000000
Classe 3	0.166667	0.166667	0.166667	0.166667	1.000000	1.000000	1.000000	0.250000
Classe 4	0.000000	0.166667	0.166667	0.000000	0.000000	0.250000	0.250000	0.000000
Classe 5	0.555556	0.333333	0.333333	0.333333	1.000000	0.750000	0.750000	1.000000
Classe 6	1.000000	0.916667	0.916667	0.916667	0.387097	0.354839	0.354839	0.366667
Classe 7	0.142857	0.285714	0.285714	0.142857	0.333333	0.666667	0.666667	0.500000
Classe 8	0.000000	0.200000	0.200000	0.000000	0.000000	0.166667	0.166667	0.000000
Classe 9	0.428571	0.000000	0.000000	0.142857	0.750000	0.000000	0.000000	0.500000
Classe 10	0.000000	0.000000	0.000000	0.66666	0.000000	0.000000	0.000000	0.666667

Tabela 4.2: *Precision* e *Recall* para variação de parâmetros de *SURF*

Contrast threshold corresponde a um limite para filtrar *features* em regiões da imagem com baixo contraste. *Edge threshold* filtra *features* correspondentes a objetos semelhantes a arestas. A alteração do *contrast threshold* não proporcionou melhorias fora da classe 2, resultando em valores piores noutras classes. Este parâmetro é difícil de avaliar pois depende do contraste presente na imagem. Imagens diferentes apresentam contrastes diferentes, resultando numa diferença nos *keypoints* extraídos. A alteração do *edge treshold* para o valor 20, proporcionou resultados iguais ou melhores que os valores *default*, se não tivermos em conta a classe 8. Ao alterar o sigma, podemos observar resultados diversos. Com o valor de 2.4 melhorou-se os resultados em certas classes, piorando noutras. Em certos casos, a variação destes parâmetros aumentou a *precision*, mas reduziu o *recall*.

Por fim, os resultados da tabela 4.4 representam a variação da distância utilizada para adquirir a "rede"de *keypoints* a utilizar para *DENSE SIFT*.

Observa-se pouca variação nos valores obtidos. A alteração do intervalo a que os *keypoints* se encontram não garante melhores resultados, apenas valores diferentes. Isto deve-se ao facto de ao utilizar uma granularidade pequena, o enorme número de detalhes incluídos causa um aumento de ruído no vocabulário gerado.

O mesmo se verifica ao aumentar a granularidade, o número reduzido de detalhes incluídos no vocabulário faz com que qualquer imagem apresente características que a permitem ser classificada em qualquer classe.

A classificação das imagens pelos diferentes métodos utilizados não apresentou resultados muito positivos, sendo que muitas imagens foram incorretamente classificadas. Este resultado deveu-se a dois fatores, ao número insuficiente de imagens de treino e ao facto de algumas imagens de teste apresentarem veículos com grandes diferenças em relação aos presentes no treino.

Como tal, foi construído um novo conjunto de treino. Este conjunto apresenta novamente as 10 classes previamente referidas, mas com um total de 1114 imagens de treino, distribuídas pelas diferentes classes. O conjunto de teste foi ajustado com ajuda da empresa de forma a conter imagens que fizessem mais sentido dentro do domínio.

Além de tentar classificar esse conjunto, foi também classificado o conjunto de teste

SIFT Recall	Default	Contrast Threshold 0.08	Contrast Threshold 0.16	Edge Threshold 15	Edge Threshold 20	Sigma 0.8	Sigma 2.4
Classe 1	0.285714	0.000000	0.285714	0.428571	0.428571	0.142857	0.142857
Classe 2	0.222222	0.333333	0.444444	0.333333	0.111111	0.666667	0.222222
Classe 3	0.166667	0.333333	0.166667	0.166667	0.666667	0.166667	0.166667
Classe 4	0.333333	0.166667	0.000000	0.166667	0.166667	0.000000	0.166667
Classe 5	0.333333	0.333333	0.333333	0.555556	0.555556	0.555556	0.111111
Classe 6	0.833333	0.666667	0.500000	0.750000	0.750000	0.500000	1.000000
Classe 7	0.142857	0.000000	0.000000	0.000000	0.142857	0.142857	0.000000
Classe 8	0.200000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Classe 9	0.285714	0.285714	0.000000	0.142857	0.285714	0.428571	0.428571
Classe 10	0.333333	0.333333	0.666666	0.33333	0.66666	0.66666	0.33333

SIFT Precision	Default	Contrast Threshold 0.08	Contrast Threshold 0.16	Edge Threshold 15	Edge Threshold 20	Sigma 0.8	Sigma 2.4
Classe 1	0.285714	0.000000	0.125000	0.375000	0.375000	0.142857	0.071429
Classe 2	0.400000	0.750000	0.500000	0.428571	1.000000	0.545455	1.000000
Classe 3	1.000000	0.400000	0.500000	0.500000	0.800000	0.500000	1.000000
Classe 4	0.400000	0.200000	0.000000	0.200000	0.166667	0.000000	0.250000
Classe 5	0.600000	0.500000	0.500000	0.625000	0.625000	0.833333	1.000000
Classe 6	0.357143	0.275862	0.230769	0.360000	0.391304	0.461538	0.363636
Classe 7	0.333333	0.000000	0.000000	0.000000	0.500000	0.333333	0.000000
Classe 8	0.125000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Classe 9	0.666667	0.500000	0.000000	1.000000	1.000000	0.600000	1.000000
Classe 10	0.166667	0.250000	0.166667	0.100000	0.166667	0.105263	0.500000

Tabela 4.3: *Precision* e *Recall* para variação de parâmetros de *SIFT*

utilizado previamente, para poder verificar se o aumento do conjunto de treino proporciona melhores resultados.

Para estes testes, não foi utilizado *DENSE SIFT*, visto que o incremento no conjunto de imagens de treino levou a que não fosse possível executar esse treino na máquina onde os testes foram efetuados.

Para comparação, foram utilizados *SURF* e *SIFT* com os parâmetros *default* e com alterações a um dos seus parâmetros que proporcionou melhores resultados. Foi também testado se o pré-processamento das imagens com o uso de *CLAHE* resultava em melhor classificação.

A modificação do conjunto de treino levou a melhores resultados de classificação, como podemos ver nas tabelas 4.5 e 4.6. Enquanto que com parâmetros *default* o *SURF* previamente apenas classificou 28 imagens corretamente de um total de 71, com o novo

DENSE	Recall			Precision		
	Default (10 kp)	4 kp	25 kp	Default (10 kp)	4 kp	25 kp
Classe 1	1.000000	1.000000	0.857143	0.118644	0.120690	0.125000
Classe 2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Classe 3	0.000000	0.000000	0.166667	0.000000	0.000000	0.500000
Classe 4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Classe 5	0.000000	0.000000	0.111111	0.000000	0.000000	1.000000
Classe 6	0.500000	0.333333	0.416667	0.545455	0.444444	0.454545
Classe 7	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Classe 8	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Classe 9	0.000000	0.428571	0.000000	0.000000	1.000000	0.000000
Classe 10	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Tabela 4.4: *Precision* e *Recall* para variação de parâmetros de *DENSE SIFT*

treino verificou-se um aumento para 40 em 71. O mesmo aconteceu para o *SIFT*, tendo apenas classificado corretamente 25 em 71 e com o novo treino classificou 40 imagens corretamente.

Recall	SURF				SIFT			
	Sem CLAHE		CLAHE		Sem CLAHE		CLAHE	
	Default	Hessian 300	Default	Hessian 300	Default	Edge Threshold 20	Default	Edge Threshold 20
Classe 1	0.857143	0.857143	0.714286	0.714286	0.571429	0.714286	0.142857	0.285714
Classe 2	0.555556	0.666667	0.555556	0.555556	0.555556	0.555556	0.444444	0.555556
Classe 3	1.000000	1.000000	0.666667	0.833333	1.000000	1.000000	0.500000	0.500000
Classe 4	0.166667	0.166667	0.000000	0.000000	0.166667	0.333333	0.000000	0.000000
Classe 5	0.444444	0.444444	0.111111	0.222222	0.555556	0.444444	0.111111	0.111111
Classe 6	0.916667	1.000000	1.000000	0.916667	0.833333	0.833333	0.750000	0.750000
Classe 7	0.142857	0.285714	0.000000	0.000000	0.285714	0.285714	0.000000	0.000000
Classe 8	0.200000	0.200000	0.000000	0.000000	0.200000	0.000000	0.000000	0.000000
Classe 9	0.571429	0.571429	0.714286	0.571429	0.857143	1.000000	0.857143	0.714286
Classe 10	0.33333	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Tabela 4.5: *Recall* para variação de parâmetros de *SURF e SIFT*

Outra conclusão que se tirou é o facto da utilização de modificadores de intensidade da imagem, como o *CLAHE*, pioraram os resultados da classificação, sendo que o numero de imagens corretamente classificadas desceu dos 40 para os 32 no caso do *SURF* e de 40 para 24 no caso do *SIFT*.

Ao variar os parâmetros dos algoritmos, não se observou alterações significativas, sendo que o número de classificações corretas não variou mais do que uma unidade.

Por fim, foi utilizado *SURF* para classificar o outro conjunto de imagens. Neste caso não se aplicou *CLAHE* às imagens visto que a sua aplicação reduziu o número de classificações corretas. A tabela 4.7 apresenta os resultados da classificação deste conjunto de

Precision	SURF				SIFT			
	Sem CLAHE		CLAHE		Sem CLAHE		CLAHE	
	Default	Hessian 300	Default	Hessian 300	Default	Edge Threshold 20	Default	Edge Threshold 20
Classe 1	0.545455	0.600000	0.714286	0.555556	0.800000	0.833333	0.200000	0.400000
Classe 2	1.000000	1.000000	0.833333	1.000000	1.000000	1.000000	0.666667	0.714286
Classe 3	0.857143	0.857143	1.000000	1.000000	0.857143	0.857143	0.750000	1.000000
Classe 4	0.333333	0.500000	0.000000	0.000000	0.333333	0.500000	0.000000	0.000000
Classe 5	1.000000	0.800000	1.000000	1.000000	0.714286	0.571429	0.500000	0.500000
Classe 6	0.407407	0.400000	0.292683	0.282051	0.344828	0.370370	0.236842	0.225000
Classe 7	0.500000	0.500000	0.000000	0.000000	0.666667	0.400000	0.000000	0.000000
Classe 8	0.142857	0.333333	0.000000	0.000000	0.333333	0.000000	0.000000	0.000000
Classe 9	1.000000	1.000000	0.833333	0.571429	0.750000	0.875000	0.500000	0.500000
Classe 10	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Tabela 4.6: *Precision* para variação de parâmetros de *SURF e SIFT*

SURF	Recall		Precision	
	Default	Hessian 300	Default	Hessian 300
Classe 1	1.000000	1.000000	1.000000	1.000000
Classe 2	1.000000	1.000000	0.750000	0.750000
Classe 3	0.000000	0.000000	0.000000	0.000000
Classe 4	1.000000	1.000000	0.750000	0.750000
Classe 5	1.000000	0.500000	1.000000	0.500000
Classe 6	1.000000	1.000000	0.750000	0.600000
Classe 7	1.000000	1.000000	1.000000	1.000000
Classe 8	1.000000	0.666667	1.000000	1.000000
Classe 9	1.000000	1.000000	1.000000	1.000000
Classe 10	1.000000	1.000000	1.000000	1.000000

Tabela 4.7: Resultado da classificação do conjunto de 28 imagens

teste.

A utilização de um conjunto com imagens recolhidas pela empresa resultou numa grande melhoria no resultado da classificação. Como podemos ver na tabela 4.7, a *precision* e o *recall* apresentam valores mais elevados por oposição a testes anteriores. De um total de 28 imagens, o classificador conseguiu classificar corretamente 25 imagens com os parâmetros *default* e 23 com o parâmetro *hessian* a 300. Isto significa que o classificador teve uma taxa de sucesso próxima dos noventa porcento, uma grande melhoria quando comparado com os testes em que se utilizaram imagens recolhidas de outras fontes.

Capítulo 5

Conclusões

No decorrer desta dissertação, foram estudados diversos métodos para o reconhecimento de peças de automóveis e formas de processar imagens, com o objetivo de realçar detalhes e permitir uma melhor extração de características. Numa fase inicial analisou-se a possibilidade de utilizar *matching* direto para a identificação das diferentes peças, através do uso de diferentes algoritmos de extração de *features*, como *canny* ou *wavelet transform*.

De seguida, explorou-se a eficácia de utilizar *matching* e homografia para o reconhecimento de peças dentro de uma dada image. Recorreu-se a extratores de *features* como *SURF e SIFT* e a *matchers* como o *FLAAN e BF*.

Por fim, estudou-se o uso de classificação para identificar as diversas peças. Utilizou-se um híbrido de *BoW* e *SVM* para alcançar este objetivo. Com *BoW* foi possível a criação de descritores de imagens das diferentes classes, para fornecer um conjunto de treino ao classificador *SVM* e posteriormente classificar um dado conjunto de imagens.

É possível verificar que nem todos os métodos estudados conseguiram alcançar o objetivo desejado. Ambos os estudos efetuados com *matching* não obtiveram resultados com taxas de sucesso altas o suficientes para serem utilizados no contexto do trabalho desta dissertação.

O uso de classificação é uma alternativa mais robusta e que proporciona uma taxa de sucesso mais elevada, sendo capaz de alcançar resultados superiores a noventa porcento nos testes efetuados.

5.1 Trabalho futuro

Os diferentes tópicos abordados no trabalho futuro são objetivos que foram pensados, mas que devido a limitações de tempo não foi possível implementar no decorrer desta

dissertação.

De futuro, é necessário explorar não só o reconhecimento de diferentes peças, mas também o reconhecimento de diferenças entre as mesmas. Ou seja, ao ter duas imagens do mesmo tipo, é necessário identificar se uma delas apresenta algum defeito (gasto, estragos, entre outros). Este estudo não será trivial, visto que as diferenças entre uma peça normal ou uma com defeito serão bastante ténues. É possível que exista problemas com classificação incorreta das imagens (o classificador pode classificar uma peça com defeito incorretamente, como se de uma peça sem defeito se tratasse).

Outro foco futuro, é um estudo mais detalhado sobre os parâmetros dos diferentes algoritmos, para permitir adaptar extratores de *features* a pormenores como defeitos nas diferentes peças ou a introdução de outros pré-processamentos de imagem. No estado atual, é provável que peças com defeito sejam classificadas como uma peça normal.

Por fim, a elaboração de uma componente gráfica permitirá facilitar o uso dos diferentes métodos descritos nesta dissertação. Será interessante a implementação de formas de alterar os parâmetros dos diferentes algoritmos usados, uma forma de carregar as imagens a usar sem recurso à localização do sistema de ficheiros e um *feedback* visual dos resultados obtidos.

Bibliografia

[1] URL: https://en.wikipedia.org/wiki/Inspection (acedido em 31/01/2017).

[2] URL: https://img.blogs.es/circulaseguropt/wp-content/uploads/2017/01/DEKRA-SafetyCheck.jpg (acedido em 31/01/2017).

[3] URL: http://www.teslasistemas.com.br/automacao/images/imagens/tesla%20(33).jpg (acedido em 31/01/2017).

[4] URL: http://www.itvp.pt/storage/centro812775afd.jpg (acedido em 01/02/2017).

[5] URL: http://www.imtt.pt/sites/IMTT/Portugues/Veiculos/Inspecao/TiposInspeccoes/Paginas/TiposdeInspecoes.aspx (acedido em 01/02/2017).

[6] URL: http://www.imtt.pt/sites/IMTT/Portugues/Veiculos/Inspecao/TiposInspeccoes/Periodicas/Paginas/Home.aspx (acedido em 01/02/2017).

[7] URL: http://www.imtt.pt/sites/IMTT/Portugues/Veiculos/Inspecao/TiposInspeccoes/Extraordinarias/Paginas/Home.aspx (acedido em 01/02/2017).

[8] URL: http://www.imtt.pt/sites/IMTT/Portugues/Veiculos/Inspecao/TiposInspeccoes/Paramatricula/Paginas/Home.aspx (acedido em 01/02/2017).

[9] URL: http://www.imtt.pt/sites/IMTT/Portugues/Veiculos/Inspecao/TiposInspeccoes/Facultativas/Paginas/Home.aspx (acedido em 01/02/2017).

[10] URL: https://upload.wikimedia.org/wikipedia/commons/d/d6/Pavlovsk_Railing_of_bridge_Yellow_palace_Winter.jpg (acedido em 11/02/2017).

[11] URL: https://upload.wikimedia.org/wikipedia/commons/d/d4/Pavlovsk_Railing_of_bridge_Yellow_palace_Winter_bw_threshold.jpg (acedido em 11/02/2017).

[12] URL: https://en.wikipedia.org/wiki/Artificial_neural_network#/media/File:Colored_neural_network.svg (acedido em 08/02/2017).

[13] URL: https://docs.opencv.org/2.4/modules/features2d/doc/object_categorization.html (acedido em 15/07/2017).

[14] URL: https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html (acedido em 15/07/2017).

[15] J. Beyerer, F. P. León e C. Frese. *Machine vision: Automated visual inspection: Theory, practice and applications*. Springer, 2015.

[16] Z. Dezhong e C. Fayi. "Face Recognition based on Wavelet Transform and Image Comparison". Em: *2008 International Symposium on Computational Intelligence and Design*. Vol. 2. 2008, pp. 24–29. DOI: 10.1109/ISCID.2008.42.

[17] *Histogram Equalization*. URL: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/histogram_equalization/histogram_equalization.html (acedido em 23/09/2017).

[18] *Homography*. URL: https://www.learnopencv.com/homography-examples-using-opencv-python-c/ (acedido em 23/11/2017).

[19] Y. Koya e Y. Bao. "Metal Parts Surface Inspection Using Luminance Information of Division Blocks". Em: *SICE Trans. on Industrial Application vol.9 No.15*. IEEE. 2010, pp. 108–114.

[20] Y. Koya e Y. Bao. "An image inspection method for car brake parts". Em: *SICE Annual Conference (SICE), 2011 Proceedings of*. IEEE. 2011, pp. 37–42.

[21] H.-D. Lin e K.-S. Hsieh. "Automated distortion defect inspection of curved car mirrors using computer vision". Em: *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering e Applied Computing (WorldComp). 2015, p. 361.

[22] L. M. M. Martins. "Automatic Visual Inspection in Industrial Problems". Em: ().

[23] M. S. Nixon e A. S. Aguado. Em: *Feature extraction & image processing for computer vision*. Academic Press, 2012, pp. 115–117. ISBN: 978-0-12397-824-0.

[24] M. S. Nixon e A. S. Aguado. Em: *Feature extraction & image processing for computer vision*. Academic Press, 2012, pp. 184–185. ISBN: 978-0-12397-824-0.

[25] M. S. Nixon e A. S. Aguado. Em: *Feature extraction & image processing for computer vision*. Academic Press, 2012, pp. 186–189. ISBN: 978-0-12397-824-0.

[26] M. S. Nixon e A. S. Aguado. Em: *Feature extraction & image processing for computer vision*. Academic Press, 2012, pp. 196–197. ISBN: 978-0-12397-824-0.

[27] M. S. Nixon e A. S. Aguado. *Feature extraction & image processing for computer vision*. Academic Press, 2012.

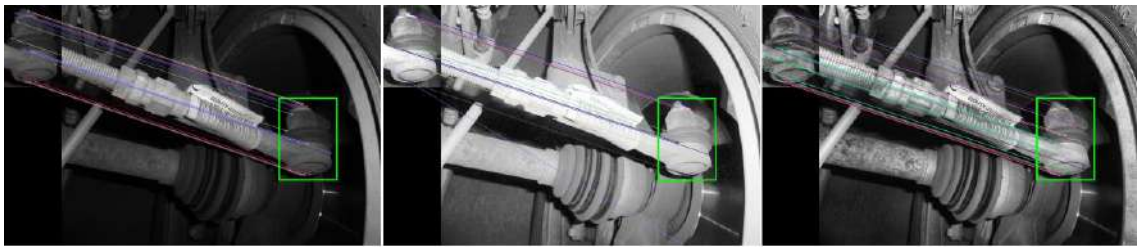
[28] *OpenCV SVM*. URL: https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html (acedido em 21/10/2017).

[29] D. K. Park, Y. S. Jeon e C. S. Won. "Efficient use of local edge histogram descriptor". Em: *Proceedings of the 2000 ACM workshops on Multimedia*. ACM. 2000, pp. 51–54.

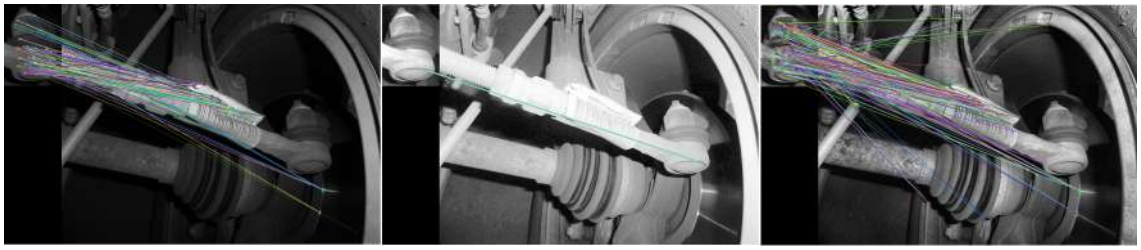
ANEXO I

Resultados *matching* e homografia com diferentes parâmetros

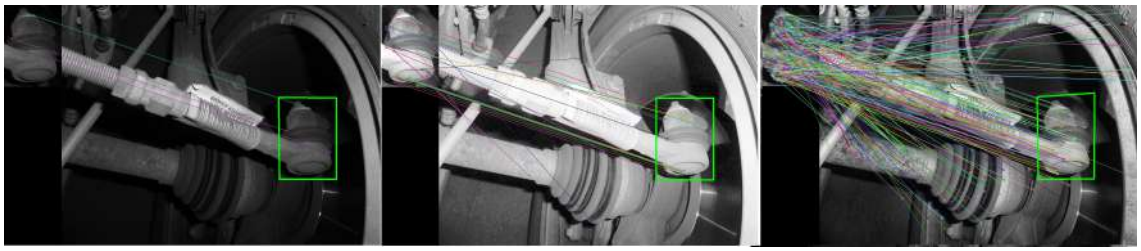
As seguintes figuras representam os resultados obtidos ao aplicar diferentes parâmetros a uma imagem. As *features* foram extraídas com recurso a *SIFT, SURF e ORB* e a intensidade da imagem foi alterada com base em *histogram equalization* e *CLAHE*. A primeira imagem corresponde à imagem original em *greyscale*, a segunda *greyscale* aplicando equalização e a terceira *greyscale* com *CLAHE*.



SIFT



ORB



SURF

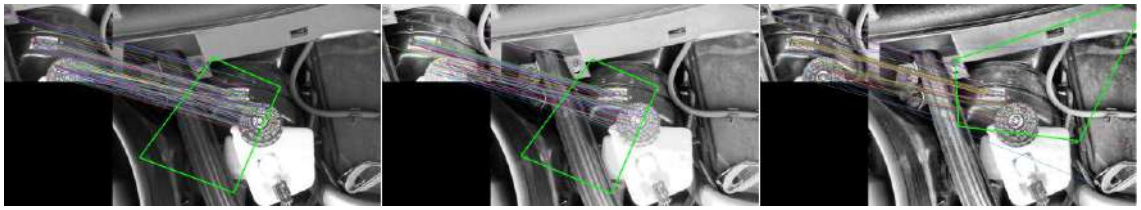
Figura I.1: Rotula de direção - teste diferentes parâmetros

SIFT

ORB

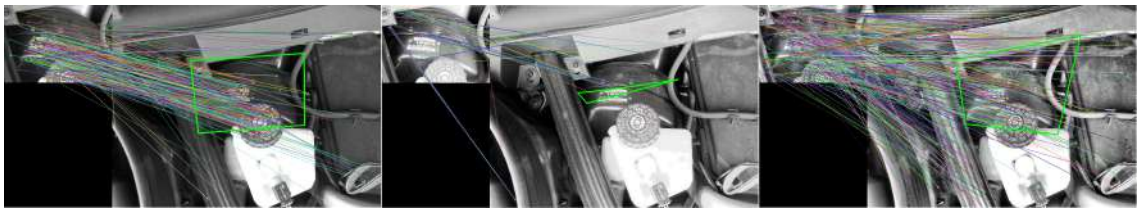
SURF

Figura I.2: Dianteira carro - teste diferentes parâmetros



SIFT

ORB



SURF

Figura I.3: Depósito - teste diferentes parâmetros

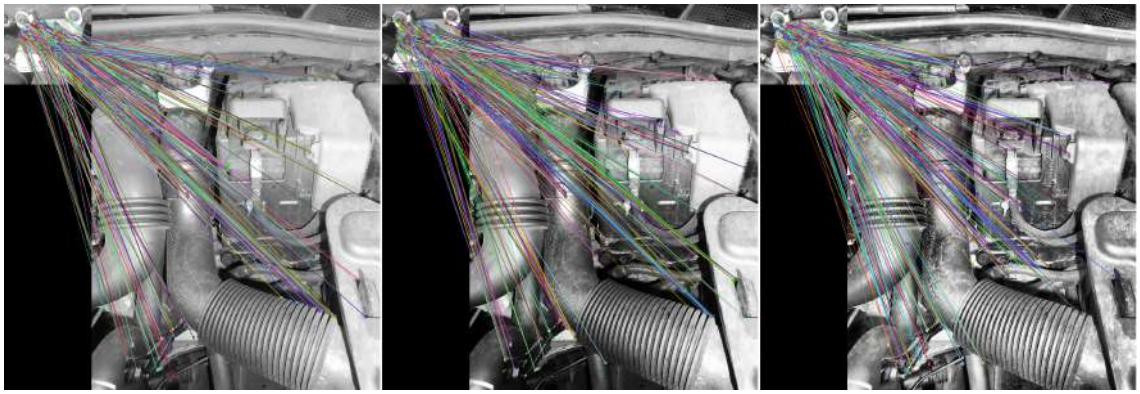
SIFT

ORB

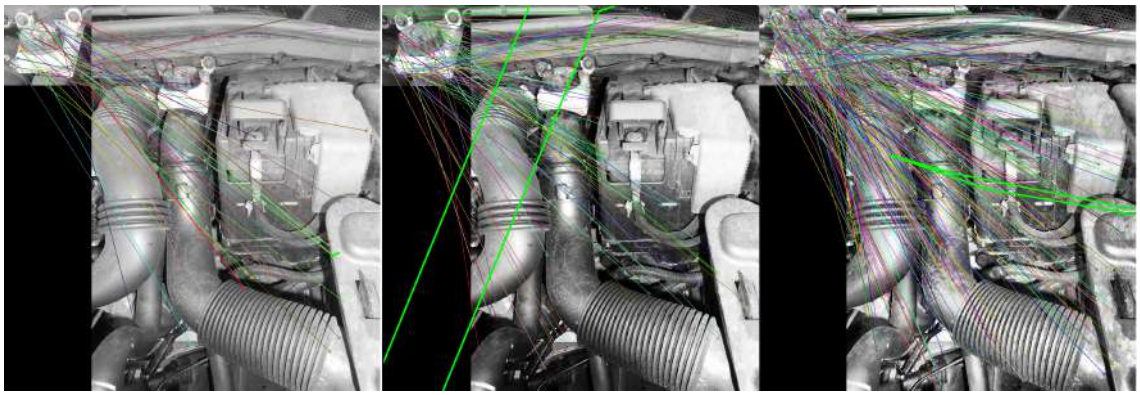
SURF

Figura I.4: Traseira carro - teste diferentes parâmetros

SIFT



ORB



SURF

Figura I.5: Tampa - teste diferentes parâmetros

ANEXO II

Matrizes de confusão - parâmetros *default*

Neste anexo estão representadas as tabelas com os valores referentes às matrizes de confusão obtidas nos testes de classificação das imagens com os algoritmos *SURF, SIFT e DENSE SIFT* nos seus parâmetros *default*.

Label	1	2	3	4	5	6	7	8	9	10		Total
Predicted 1	6	2	0	1	0	0	4	3	0	0		16
Predicted 2	0	0	1	0	0	0	0	0	0	0		1
Predicted 3	0	0	1	0	0	0	0	0	0	0		1
Predicted 4	0	2	0	0	0	0	0	0	0	0		2
Predicted 5	0	0	0	0	5	0	0	0	0	0		5
Predicted 6	1	3	4	0	3	12	1	2	2	3		31
Predicted 7	0	0	0	2	0	0	1	0	0	0		3
Predicted 8	0	0	0	3	1	0	1	0	2	0		7
Predicted 9	0	1	0	0	0	0	0	0	3	0		4
Predicted 10	0	1	0	0	0	0	0	0	0	0		1
Total	7	9	6	6	9	12	7	5	7	3		SURF

Tabela II.1: Matriz de confusão obtida na classificação utilizando *SURF* com parâmetros *default*

Label	1	2	3	4	5	6	7	8	9	10		Total
Predicted 1	2	0	0	1	1	0	2	1	0	0		7
Predicted 2	0	2	2	0	0	1	0	0	0	0		5
Predicted 3	0	0	1	0	0	0	0	0	0	0		1
Predicted 4	1	0	0	2	0	0	1	0	1	0		5
Predicted 5	0	0	0	1	3	0	1	0	0	0		5
Predicted 6	2	2	3	0	3	10	2	2	3	1		28
Predicted 7	0	0	0	1	1	0	1	0	0	0		3
Predicted 8	1	1	0	1	1	1	0	1	1	1		8
Predicted 9	0	1	0	0	0	0	0	0	2	0		3
Predicted 10	1	3	0	0	0	0	0	1	0	1		6
Total	7	9	6	6	9	12	7	5	7	3		**SIFT**

Tabela II.2: Matriz de confusão obtida na classificação utilizando *SIFT* com parâmetros *default*

Label	1	2	3	4	5	6	7	8	9	10		Total
Predicted 1	7	7	5	6	7	6	7	4	7	3		59
Predicted 2	0	0	0	0	0	0	0	0	0	0		0
Predicted 3	0	0	0	0	0	0	0	0	0	0		0
Predicted 4	0	0	0	0	0	0	0	0	0	0		0
Predicted 5	0	0	0	0	0	0	0	0	0	0		0
Predicted 6	0	1	1	0	2	6	0	1	0	0		11
Predicted 7	0	0	0	0	0	0	0	0	0	0		0
Predicted 8	0	1	0	0	0	0	0	0	0	0		1
Predicted 9	0	0	0	0	0	0	0	0	0	0		0
Predicted 10	0	0	0	0	0	0	0	0	0	0		0
Total	7	9	6	6	9	12	7	5	7	3		**DENSE**

Tabela II.3: Matriz de confusão obtida na classificação utilizando *DENSE SIFT* com parâmetros *default*