FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

# Reversing ShopView analysis for planogram creation

**Telmo Domiciano Pereira Barbosa**


Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Dirk Christian Elias

Second Supervisor: João Nuno Castro Gonçalves

June 27, 2017

# Reversing ShopView analysis for planogram creation

**Telmo Domiciano Pereira Barbosa**

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Luís Filipe Pinto de Almeida Teixeira
External Examiner: José Maria Corte Real da Costa Pereira
Supervisor: Dirk Christian Elias

June 27, 2017

# Abstract

With the increasing care of retail shop owners in improving sales and costumer experience, there is a need to develop technology in order to optimize their goals. It's proven that a planned product placement can boost sales and improve costumer experience [CSDK07]. With this in mind, Fraunhofer Portugal came up with ShopView [RGC+16] solution to help retail shops extract, validate and manipulate planograms from high-resolution images of the real shelves in the store.

In this sense, this thesis focused on the creation of an algorithm using computer vision algorithms to extract information from high resolution images of retail shelves taken with the ShopView solution. Particularly, it was implemented pre-processing steps to improve the efficiency and accuracy of an OCR (Optical Character Recognition) engine in recognizing the text in the shelves products. These pre-processing algorithms comprise of denoising and segmentation techniques. The use of this OCR engine brings additional information about the products, this information is later used in clustering algorithms to automatically extract an accurate planogram from shelves photos.

The presented algorithm is capable of extracting relevant information from the shelves images, to identify the existing products and create a valid metadata about them and their location. With this metadata, it is possible to create, validate and modify the planogram in ShopView. The use of OCR on this algorithm has advantages over other available approaches due to its capability to differentiate products with minimal visual differences and its immunity to appearance changes on the products packaging. Moreover, the methodology proposed does not require any previous user interaction to work properly.

# Resumo

Com o aumento da preocupação dos retalhistas na melhora dos resultados das vendas e da experiência dos consumidores, existe uma necessidade de desenvolver tecnologia que auxilie a otimização desses objetivos. Está provado que uma correta colocação dos produtos nas prateleiras pode aumentar significativamente as vendas e a experiência dos consumidores [CSDK07]. Com isto em mente, a Fraunhofer Portugal desenvolveu o ShopView [RGC+16], uma solução que tem como objetivo ajudar os retalhistas a extrair, validar e manipular os planogramas a partir de imagens de alta resolução das lojas.

Desta forma, esta tese tem como foco a criação de um algoritmo, com auxilio a algoritmos de visão computacional, de forma a extrair informação de images de alta resolução de gondolas de supermercados obtidas pelo ShopView. Particularmente, foram implementados passos de pré-processamento de forma a melhorara a eficiência e precisão de um motor de OCR (Optical Character Recognition) no reconhecimento de texto nos produtos da gondola. Estes algoritmos de pré-processamento são compostos por técnicas de segmentação e remoção de ruido. O uso deste motor de OCR permite obter informações adicionais sobre os produtos, esta informação é posteriormente usada em algoritmos de agrupamento para extrair automáticamente um planograma preciso a partir das imagens.

O algoritmo apresentado é capaz de extrair informação relevante das imagens das gondolas, de forma a identificar os produtos existentes e criar metadados válidos sobre esses produtos e a sua localização. Com estes metadados é possível criar, validar e modificar o planograma no ShopView. O uso de OCR neste algoritmo oferece vantagens sobre as outras abordagens disponíveis devido à capacidade de diferenciar produtos com diferenças mínimas entre si, e a sua imunidade a alterações no aspeto das embalagens dos produtos. Além disso, a metodologia proposta não requer nenhuma interação prévia do utilizador para funcionar corretamente.

# Acknowledgements

I would like to thank to:

Fraunhofer AICOS Portugal for the opportunity to grow at professional level providing a suitable environment and the required tools to make this project.

Professor Dirk Elias, my supervisor at FEUP, for proposing this thesis and for the guidance provided to make this work possible.

João Gonçalves, my supervisor at Fraunhofer, and Filipe Soares, senior scientist at Fraunhofer, for all guidance, suggestions, knowledge exchange, support and patience during all this journey.

All my friends and colleagues for all the great moments.

Finally, my mother, father, sister, brother and Diana for all the patience, unconditional support and incentive along this months.

Telmo Domiciano Pereira Barbosa

*"We do not know what we want and yet we are responsible for what we are - that is the fact."*

Jean-Paul Sartre

# Contents

# List of Figures

# Abbreviations

| | |
|---|---|
| OCR | Optical Character Recognition |
| XML | eXtensible Markup Language |
| ROI | Region Of Interest |

# Chapter 1

# Introduction

## 1.1 Context

This document reports the work developed in the scope of the course "Dissertação" of the Integrated Master in Computer and Informatics Engineering of University of Porto. The work was proposed by Fraunhofer Portugal and was mainly developed at the Fraunhofer facilities in Porto, Portugal.

Retail shops aim to optimize the costumer experience, sales and minimize the operations cost. Studies found that correct planed product placement can significantly increase sales, improving retailer's earnings and customer satisfaction [CSDK07]. With this in mind, one of the responsibilities of marketing is to design planograms in order to plan an attractive presentation on the shelves, also brands often sign agreements with distributors in order to ensure that their products will be placed in specific locations and quantities. Since the planograms are built having a single store in mind, it is not always easy to follow in all the retailer chain stores due to changes in the store layout. Checking the placement of each product is nowadays carried out manually. Employees from the brands and retail shops are sent to inspect each shop and check agreement between the planogram and the actual placement of the products. Given the large number of shops, the different shop layouts between the same company and the number of products per shop, manual checking is extremely expensive and its far from an optimum solution. Also, smaller retail owners don't always have access to planogram creation tools, so they put aside the creation of a planogram, not getting the benefits of it.

Fraunhofer Portugal is developing ShopView [RGC+16] which is a solution to the problems that retailers have at the current time with planograms.

## 1.2 Motivation

A planed product placement can bring considerable benefits to retailers, as stated previously. With the awareness of retail shop owners and distributors of the importance and benefits of a planned product positioning, its relevant to develop solutions that aims the process of creating, maintain and validating the planograms. The most common solutions available on the market are mainly focus to only design the planogram manually. Some solutions that automatically create and validate the planograms from real shelf images are emerging, but they are not always a true complete and autonomous solution.

ShopView came to fight the problems that exists with the current solutions available. With the correct products recognition from the shelves images ShopView can build the planogram without the need of manually creating it in software, this is especially helpful for small retail shop owners since they don't always have access or knowledge to use the available solutions of planogram creation. After the planogram creation from meta-data of product positioning extracted from the pictures of the shelves, this solution is capable of checking the planogram compliance, also storing this meta-data and creation of a time-line of the product placement.

The aim of this thesis is to detect, locate and identify the products of the shelves from the high-resolution images taken from ShopView, involving several technical challenges. The different product package shapes and forms, the arrangement of products on shelves and the unconstrained environment are some of the challenges to be surpassed.
Will be explored the use of OCR in the identification of products, the development and adaptation of pre-processing techniques will help accommodate the OCR to this context. Is expected the ability to achieve high rates of product recognition, despite the very small visual differences between some products.

## 1.3 Overview of Dissertation

This document is organized in 5 chapters.

In the present chapter, introduction, is described the objectives of the work, it's context and motivations.

In chapter 2, is described the state of the art of the most relevant areas and some related work, solutions and methods are presented.

The chapter 3, addresses the state of the art of algorithms related to text processing, also related to the image processing.

In chapter 4, the work done is described in a detailed form in order to better understand the different approaches tried and the associated results.

In chapter 5 are explained and discussed the different approaches taken along the development of the algorithm.

Finally in chapter 6, are presented the main conclusions and reflections of the work, and the future work.

# Chapter 2

# Literature Review

In this chapter, will be presented some existing solutions related to this work. In the first section 2.1 is explained what is a planogram and its importance in retail and the associated challenges. In section 2.2 is presented some existing solutions to retail planograms problem and how they operate. In section 2.3 are presented some common techniques used in image processing that are useful in this thesis. The section 2.4 describes the use of features, some of them used in the implementation of this thesis. In section 2.5 is described the importance of the image segmentation on our context. Finally, the last two sections make a small introduction to the technologies used in the implementation.

## 2.1 Planograms in Retail

For the various reasons already stated, a planogram is a vital tool to organize the products in a store and archive better results on sales and costumers experience. A planogram is a retailer's blue print, which dictates how and where products are physically placed in the store shelves. Figure 2.1 shows an example of a planogram created manually with the aid of specific software.

Planograms specifies the absolute physical locations of the products, and the amount of space each type of product should occupy. Those planograms can be from the entire store, from an aisle (map of cookies aisle) or from a specific category (defines how one type of product is distribute though a store). Some of the porpoises for building planograms are:

- **Retail operations** — Helps the store employees to know where the products should be placed and to help locate them. It also should ensure sufficient inventory levels on the shelf taking into account the expected sales of each product.

- **Marketing** — To plan where new products should be placed and appraise how much the store should be charging the vendor for the product location. This is important for the store

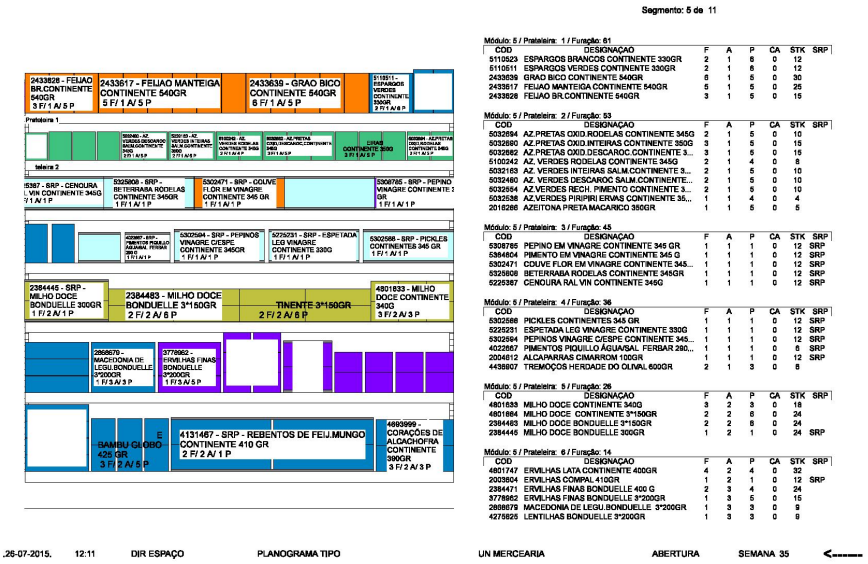



Figure 2.1: Example of a planogram

income being that certain places (such as shelves at the eye level) that generates more sales can be "rented" to vendors for high prices.

- **Costumer experience** — Placing associated products together will allow for an easier and satisfactory experience in the costumer purchasing.

### 2.1.1 Challenges of the Planograms

Having in mind all the stated reasons, the planograms should be planned in a detailed mode, resulting from a lot of thought, research and effort by the specialist team of the retailer (typically a marketing team). The planogram should be applied and maintained in the stores in a correct manner, the incorrect implementation could generate a lower income from sales and fines for not fulfil the agreements with the vendors that rent spots on the shelves. The problem is that planograms usually get outdated for many reasons: restocking, arrival of new products and seasonal changes in products and layouts. Due to the constant alterations, checking the planogram compliance is a necessary task to ensure that the current planogram is implemented correctly. This is a very time-consuming task for retailers, typically done by clerks in off-peak times. Manual checking is not an optimal solution, a more reliable and cost-effective solution must be used, this is discussed in section 2.2.

## 2.2 Retail Planogram Automated Extraction

There have been a few approaches to automate the planogram creation based on the real product placement [AO12, AA15, LLD+16, Tec, Cor, Yod, Tra]. A typical food store stocks an average of 40,000 items on their shelves [Foo15]. These items vary in form and size such as bagged, boxed or canned. In order to identify each item on a shelve using captured images isn't simple due to constant variations in the products, similarities between different products and when the same product has seasonal variations on its packaging, or simply makes changes to the design. All these approaches make use of image processing techniques in order to identify the products from real life pictures.

The available approaches can be divided into two types. Some try to identify all the products in the shelves by using product template images and verifies if they are according to the planogram 2.2.1. Others, try to extract the layout by detecting recurring patterns (without requiring product template images) identifying thereby the distribution of the products on the shelves 2.2.2.

It's worth noticing that none of these approaches have the ability to create a planogram on their own without any previous user interaction unlike the approach used in this thesis. They also struggle with products with similar packages or when a product has few or low-quality features.

### 2.2.1 Planogram compliance using template images

This is the most common approach, used for example in [AO12, AA15], uses object recognition analysis based on stored product images trained with artificial learning techniques (Deep Learning).
Deep Learning is a sub-field of machine learning and works with algorithms based in neural networks. These algorithms can be used to automatically extract features from images, in this case the features on product packages, in order to identify them. Yodiwo [Yod] makes use of this type of algorithms along with segmentation algorithms to identify the products on the shelves.

This approach does not always work as desired, especially in today's market since the same products don't always have the same appearance (may have a different appearance for seasonal promotions for example). These methods also require a large set of product images for training, this is very time consuming and requires constant updates to the database when new products arrive and when there are changes in the product packages. Also, this technique has trouble to distinguish different products when they have similar packaging characteristics.

### 2.2.2 Planogram compliance by detecting the layout

These methods are explored in [SHM15, LLD+16]. The problem with these methods is that they could lead to errors where different products with similar appearance are together, or when

the same product has some different features on the packaging it could lead to identify them as different products.
These techniques are unable to identify the products, they just locate the products and group the same products together.

## 2.3 Image Processing

Everything is a mixture of shapes, colours and textures. In order to make a machine interpret such information from an image or a set of images, image processing techniques can be used.

There are several levels of image processing algorithms, in this thesis all the levels are addressed. From lower levels with noise removal and feature detection, to intermediary levels like segmentation, to higher levels such as object detection and text recognition.

### 2.3.1 Product recognition

Identifying products using image processing is a relevant topic for development applied to different fields, mainly as assisted technology for blind people, to help them identify products in a store [Groa].

Nowadays multiple techniques of image processing are available to identify products:

- **Object Recognition based on attributes** — Items like vegetables and fruits generally don't carry brand logos or text, therefore, in order to identify them it's used an approach that takes in to consideration its size, colour, texture and other attributes.
  There is much work on this area to recognize everyday objects [SBF13, FEHF09]. Toshiba have developed an "object recognition scanner" focused on retail, which identifies the items only based on colour and pattern [Tos14].

- **Template and Feature Matching** — This technique is very time consuming to setup and use. Also, it does require a large database which contains attributes of all the products. In order to identify a product, an image of the item is captured and its attributes are extracted. Then this attributes are used to compare and find a matching template which has information about the product. There are some databases available online [Web, Grob], but for most cases they are not complete enough and can vary a lot from country to country and even store to store.

- **Optical Character Recognition (OCR)** — The approach used in this thesis. Identifying text found in images and converting it to machine-encoded text can help identify a product based on the brand name, flavours or some other description.
  The main challenge of this approach to recognize text in products, will be the recognition of different fonts, uneven backgrounds and text alignment. Work done in [BMB13] tries to fight this problems, however, their technique consists of aggressive noise cleaning which can lead to vanishing of some characters worsen the recognition.

### 2.3.2 Image Pre-processing

There are several steps in OCR algorithms, being the pre-processing step very relevant to the matter of this thesis. Most of OCR engines come with a lot of pre-processing techniques already available. The problem is that these techniques are optimized for scanner-captured images of documents. In order to recognize text in shelf products, its required different approaches from the ones required to recognize text in scanned documents. Some important explored techniques are:

- **Denoising** — The objective of an image denoising algorithm is to remove noise while preserving the edges. In natural images, this is one of the most important steps, being that it will help extract the text from the image removing the noise under the text. The work done in [KVV13], [KVV14] and [BMB13] rely highly in this step

- **Non-Linear image transformation** — This is required for products with non-flat surface, like a cylindrical box of cookies, this transformation will make those surfaces flat in order to optimize the recognition of characters, some similar work has been done to recognize curved lines in the center of textbooks [BGR07]. In this work curves are traced bounding the upper and lower borders of the text area and some lines are traced and evaluated between these curves, then with the help of interpolation equations its created perpendicular lines of the text.

- **Perspective correction** — Sometimes the products are non-orthogonal to the camera, creating a distortion. Experiments show that this distortion is harmful to the OCR process and some work has been done to solve this for images of documents [BGR07]. In this work is taken in example one image not acquired orthogonal to a text document, resulting in an area of the document not rectangle, but rather a trapezoid (or a parallelogram). The technique involves pointing out the quadrilateral containing the text manually, followed by calculations to normalize the text area.

### 2.3.3 Morphological operations

Image morphology is an important tool in image processing, normally performed on binary images. Morphology has been used to perform noise suppression, texture analysis, edge detection among others, for applications such as medical image processing and image compression. Mathematical morphology aids the development of operators for image processing, based around mathematical concepts from set theory applications and its functions. The basic morphological operations are erosion and dilation, but there are also others variant forms such as opening, closing, and gradient. It needs two inputs, an image to perform the operations, and a kernel which decides the nature of operation.

- **Erosion** — The main objective of erosion is to diminish the foreground, eroding its boundaries, making holes in this area larger. This makes the objects smaller by removing the outer

layer of pixels, as seen in Figure 2.2b. The kernel slides through the image (as in 2D convolution). For each iteration, the algorithm checks all the pixels under the kernel, if are all '1', that pixel in the picture stays at '1', otherwise it gets eroded - becomes '0' (black).

- **Dilatation** — The opposite of erosion. The main objective of dilation is to enlarge the foreground, as seen in Figure 2.2c. The way it works is, a pixel becomes '1' (white) if at least one pixel under the kernel is '1'. This makes the objects larger by gradually increase the boundaries of the objects and shrinking holes in those areas.


(a) Source (b) Erosion (c) Dilatation

Figure 2.2: Example of morphological operations

In cases such as noise removal, is commonly used the morphological opening, which defines as dilatation after erosion (the inverse is the close operation). The reason is that after erosion, used to remove the noise, the object becomes smaller. So, to recover the shrinked area, dilatation is used.

## 2.4 Image Features

There are techniques to extract visual features such as shape, colour and texture in order to characterize images. Features can be used for object recognition, image alignment, 3D reconstruction, motion tracking, etc. For object recognition, the main use in planogram scenarios, the extracted features are compared to those of the template images previously processed and stored in a database. With this comparison, the images are indexed according to its distance to the database template images. This can be used to identify the products on the shelves in retail scenarios. The main problem with this approach is the requirement of a large database and the need to constantly update the product images when the package is updated, also is very computer intensive. However this is the method which is common in the already available planogram checking solutions to identify the products [AO12, AA15, SHM15].

### 2.4.1 Defenition of a Feature

The definition of a feature is always relative to its application. A feature is a region in the image with an interesting characteristic - a point of interest. These features are mainly the initial point of image processing algorithms. Tuytelaars and Mikolajczyk [TM] define a local feature as "it is an

image pattern which differs from its immediate neighborhood". A good feature must be consistent over many images of the same scene, invariant in some transformations and immune to noise. Follows an example in Figure 2.3 of detected features in an image.


(a) Source image

(b) Detected features with FAST

Figure 2.3: Example of detected features in an image using FAST feature detector

### 2.4.2 What is Feature Detection?

The concept of feature detection refers to the low-level image processing operations that compute information at every pixel, deciding if there is or not a feature at that specific point. Typically, this is used to detected features for later examination. The main desirable property of a feature detector is repeatability, having capability of detecting the same features of the same target in different images.

There are many computer vision algorithms that use feature detection, so as a result, the has been a big development in this field resulting in a large range of feature detection algorithms. In the following subchapters are presented the common feature detection techniques.

#### 2.4.2.1 Edge Detection

Identifies points in images where the brightness has discontinuities. The points where these discontinuities happen are named edges and are typically organized into a set of curved line segments. The most commonly used are:

- **Canny algorithm** — The Canny edge detector developed by John F. Canny [Can] uses a multi-stage algorithm to detect a wide range of edges in images. The canny algorithm works as follows:

  1. Removes noise with a Gaussian Filter;
  2. Finds the intensity gradients of the image;
  3. Filter the edges with a non-maximum suppression;
  4. Apply a double threshold to determine potential edges;
  5. Track edges by hysteresis, suppressing the weak edges.

- **Deriche algorithm** — Developed by Rachid Deriche, it's a multistep algorithm used to obtain an edge detection in a discrete two-dimensional image. It's based on Canny's edge detector. This algorithm tried to improve the detection quality, accuracy and unambiguity.

- **Sobel–Feldman operator** — It's bases on an operator that calculates the finite differences, given an approximation of the gradient in the pixels of an image. At each point of the image, the output of Sobel's operator is a corresponding gradient vector or a norm of this vector.

- **Prewitt operator** — Developed by Judith M. S. Prewitt. In terms of operation it's similar to Sobel–Feldman operator but, this operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical directions. This algorithm is somehow inexpensive in terms of computer processing like Sobel–Feldman operator.

#### 2.4.2.2 Corner Detection

Corner detection is a popular research area in image processing and therefore many corner detectors have been presented. A corner can be defined as the intersection of two edges. Some of the most used are:

- **Harris detector** — The core idea of Harris detector is calculate the Eigen values and eigenvectors of a small region. Then, use the largest two Eigen values to calculate some functions. Finally, use the function value and a threshold to detect the corner.

- **Smallest Univalue Segment Assimilating Nucleus (SUSAN)** — The SUSAN algorithm works by counting the number of pixels having the same brightness as the center pixel. Comparing those detected pixels on the neighborhood of the center pixel (with a threshold), the detector can determine whether the centre pixel is a corner. This algorithm can work well with noise on the image.

- **Features from accelerated segment test (FAST)** — This is a relaxed version of the SUSAN corner criterion. Is very suitable for real-time video processing due to its speed of processing [RD06]. For any given point $p$ in the image, it can test whether is a corner by considering the 16 pixels at a radius of 3 pixels around $p$ and finding the longest uninterrupted sequence of pixels whose intensity is either greater than that of $p$ plus a threshold or less than of $c$ minus the same threshold. If the uninterrupted sequence is at least 12 pixels long, then $p$ is a corner. It's possible to make the algorithm faster by checking only some of the pixels of the $p$ radius. The Figure 2.3 shows an example of detected features using this algorithm.

#### 2.4.2.3 Blob Detection

One main reason to study blob detectors is to provide complementary information about regions, which is not obtained from corner or edge detectors. Maximally stable extremal regions and Laplacian of Gaussian are some examples of the most common ones. Blob detectors have two main types of methods:

- **Differential methods** — Based on derivatives of the function with respect to position.

- **Methods based on local extrema** — Based on finding the local maxima and minima of the function.

#### 2.4.2.4 Hough transform

The purpose of the Hough transform is to find instances of objects of a certain class of shapes within an image. Some relevant characteristics of this technique is that it can find the shapes even with noise and when they are not perfect, having gaps for example. The initial proposal of Hough transform [DH72] was concerned with the identification of lines in the image, later it has been generalized to identify other arbitrary shapes being the most common circles and ellipses.

The simplest form of Hough transform is to detect straight lines, and is the most relevant to this thesis. The form is:

$$r = x\cos\theta + y\sin\theta$$

where $r$ is distance from the origin to the closest point of on the straight line, and $\theta$ is the angle between the $x$ axis and the line connecting the origin with that closest point. It is therefore possible to associate with each line of the image a pair $(r,\theta)$. The $(r,\theta)$ plane is sometimes referred to as Hough space for the set of straight lines in two dimensions.
Some variations of the original Hough transform have been developed, follows some of the most important:

- **Standard Hough transform** — Consists of a voting stage, in which each feature point is mapped into a curve in a parameter space, and an exhaustive search for peaks. The parameter space is usually quantified and represented by an accumulator array [DH72].

- **Probabilistic Hough transform** — Has improved performance regarding the standard. Replaces the full-scale voting in the standard algorithm by a group of a small number of randomly selected points [KEB91].

- **An O(log n) Pyramid Hough transform** — Also with an improved performance regarding the standard algorithm. Is a pyramid-based Hough algorithm that finds clusters of Hough-space pixels arising from small blocks of the given image, and recursively merges these clusters [JR89].

### 2.4.3 Descriptors

For the recognition of the objects there are some descriptors available, they describe an object by the distribution of intensity gradients or edge directions. They divide an image in cells, and for each cell is calculated a histogram of gradient directions. All these histograms compose the descriptor. Some of the most robust and relevant feature descriptors are:

- **Scale-invariant feature transform (SIFT)** — SIFT works based on the appearance of the object at some interest points and makes use of difference of Gaussians. This is unaffected to changes to rotation, scale, illumination and small changes on perspective. From a technical point of view, it uses a vector of histograms of image gradients [Low].

- **Gradient location and orientation histogram (GLOH)** — Similar to SIFT but uses more spatial regions for the histograms [MS05].

- **Speeded up robust features (SURF)** — This descriptor describes the intensity of the pixels within the neighborhood of a specific point. It's also unaffected by changes in scale and rotation [BETVG08].

- **Histogram of oriented gradients (HOG)** — Similar to SIFT, but uses a dense grid of uniformly spaced cells and uses overlapping local contrast normalization. Its unaffected to geometric and photometric changes. [McC86].

- **Oriented FAST and rotated BRIEF (ORB)** — This descriptor is a fusion of FAST (Features from accelerated segment test) keypoint detector, uses corner detection, and BRIEF (Binary Robust Independent Elementary Features) descriptor [RRKB11].

- **KAZE Features** — Is a feature detection and description algorithm. Uses a novel mathematical framework called Fast Explicit Diffusion (FED) embedded in a pyramidal framework to speed-up dramatically the nonlinear scale space computation [ABD12].

## 2.5 Image Segmentation

Segmentation is a partition of the image into several coherent parts, for the porpoise of this thesis will help to partition the image into regions distinguishing uniform from non-uniform regions. This partitioning could help to recognize the shelf divisions and also help locate the individual products. A good example of image segmentation can be found in [XQ09] which takes images from the streets and thought segmentation techniques is able to identify the sky, ground, buildings and trees. Also in [STJ15] uses semantic segmentation to archive the similar results. In [GW15] is presented some segmentation techniques which allow to have decent accuracy in text location using segmentation.

## 2.6 OpenCV

To facilitate the image processing it's used the OpenCV library (Open Source Computer Vision Library) [ope] since its free for both academic and commercial use. The library has hundreds of optimized algorithms implemented in C++, which includes both classic and state-of-the-art computer vision algorithms. Today this library is used in a wide range of applications that ranges from interactive art, to mines inspection, stitching maps, detection of swimming pool drowning accidents.

The high number of available optimized algorithms with a wide range of applications and the large community makes this library suitable for the porpoise of this thesis.

## 2.7 C++

It's an imperative, object-oriented programming language, providing also low-level memory manipulation. Since performance, efficiency and flexibility of use as its design highlights this makes this language a must for this thesis.

## 2.8 Conclusions

There is not much work done in order to extract planograms from shelve images using OCR as an additional tool to retrieve more information, but rather using other visual techniques described. With the use of OCR is expected to improve the accuracy of the recognized products and without being crucial to create and maintain an updated database of images of the products. The already available OCR pre-processing techniques are a good starting point to recognize a good parcel of products, but it will take some adaptation from this scanner-captured images oriented algorithms to our case scenario.

# Chapter 3

# Text Processing

In this chapter, it's approached the text processing including text processing in images.
In section 3.1 is introduced the OCR and the relevant work done in this area. The state of the art of scene text location algorithms are presented in the section 3.2. In the chapters 3.3 and 3.4 are described two important steps for this thesis. The last chapter explains how the used OCR engine works.

## 3.1 Optical Character Recognition (OCR)

Optical Character Recognition (OCR) is used to convert text present in images into machine-encoded text. OCR is related to other fields of knowledge such as pattern recognition and digital image processing. An OCR system comprises of a series of processing steps. First, pre-processing techniques can be applied on the image to improve the chances of a successful recognition. Then, takes place the main OCR algorithm which outputs the characters recognized. Finally, post-processing techniques are used to fix some wrongly recognized characters, for example comparing the recognized words to the words of the language of the recognized text. This thesis focusses mainly on the pre-processing step, as well as doing some tweaks on the configuration of the other following two steps.

While modern off-the-shelf OCR engines show particularly high accuracy on scanned text, text detection and recognition in natural images still remain a challenging problem. This challenge comes from the different text fonts, languages, uneven backgrounds and uneven text alignment present in the images. The work done in [BMB13] does a good job in the recognition of the text, overcoming the problems of the real word images, however, as previously stated, sometimes this technique worsens the recognition of the text due to the aggressive noise cleaning. Regarding to overcome the uneven backgrounds, the work done in recognition of text in videos [Uch14, SKPB00, GZC15, MAP16] has some good results. The work done in [KVV13] and posteriorly in [KVV14] does a reasonable good job in the recognition of artistic fonts, unfortunately even

in the last article their proposed technique is not suitable for semi-arc text, and this is something common in brand logos.

## 3.2 Scene Text Localization

Scene text location is a problem with many interesting applications, ranging from ranging from helping the blind people, translation of text from real scenes, to index image and video databases by their textual content, etc... In the context of this work the detection and localization of text is a challenging task due to high variability of text appearance and the non-uniform backgrounds with high abundance of graphics.

There is been a big development in the detection of text in natural images with a wide range of approaches. Kumuda [KB15] method is based on texture feature extraction using first and second order statistics, his method has reported encouraging results in location of text but fails when the separation between the background and the text is not very clear. Other methods use connected components [KK13] or features like corners [FSZ15] or edges [CSJ16] also making use of maximally stable region algorithm to find text on images. Neumann has also done a relevant job in this area [NM16, NM13, NM13, NM12] with good results. In [NM12] Neumann proposes an algorithm with a real-time performance achieved by posing the character detection and segmentation problem as an efficient sequential selection from the set of Extremal Regions (ER). In this algorithm, the ER detector is robust, besides other things, to colour and texture variation.

In the first stage, is estimated the probability of each ER being a character using features calculated and only ERs with locally maximal probability are selected for the second stage, where the classification accuracy is improved using computationally more expensive features. A highly efficient clustering algorithm then groups ERs into text lines.

## 3.3 String Edit Distance

In some problems, it's required to find how similar are two words or strings, some examples are in spell correction, information extraction, speech recognition and DNA sequence alignment. Edit distance is a string metric for measuring the difference between two strings or words. The edit distance between two strings is the minimum number of operations required to transform one string into another. The common operations are insertion, deletion and substitution.

**Insertion:** cerel $\rightarrow$ cere**a**l

**Deletion:** cer**e**al $\rightarrow$ ceral

**Substitution:** cere**e**l $\rightarrow$ cere**a**l

Different definitions of edit distance allow different sets of string operations. For instance:

- **Levenshtein distance** - allows deletion, insertion and substitution. Being the most common metric, the Levenshtein distance is usually what is meant by "edit distance".

  The Levenshtein distance between "cereal" and "bebe" is 4. A minimal edit script that transforms the former into the latter is:

  1. cereal → bereal (substitution of "c" for "b")
  2. bereal → bebereal (substitution of "r" for "b")
  3. bebeal → bebea (deletion of "l" at the end)
  4. bebea → bebe (deletion of "a" at the end).

- **Longest common subsequence (LCS) distance** - only allows insertion and deletion, not substitution.

- **Hamming distance** - only applies to strings of the same length and only allows substitution of characters.

- **Damerau–Levenshtein distance** - allows insertion, deletion, substitution, and the transposition of two adjacent characters.

- **Jaro distance** - only allows transposition.

## 3.4 Clustering

Products have different faces, and it's very likely that in each face the product have similar words. Taking this fact into account, it's possible to take advantage of that to recognize the different faces of the products. Clustering is a technique that automatically groups data that are similar to each other. The similarities of words in our environment can lead to grouping the same faces of a product. For example, assuming the OCR algorithm recognizes the text that indicates the brand of the product, and since the brand is in most of the faces of products, this technique is a great tool to associate these different faces.

## 3.5 Tesseract

In order to recognize the text in images an optical recognition character engine is needed. Tesseract is a free open-source OCR engine developed in C/C++ [tes]. This is the OCR engine used in this thesis for numerous reasons being that it has Unicode (UTF-8) support, and can recognize more than 100 languages "out of the box", including Portuguese, it's possible to integrate with applications and in 2006 was considered one of the most accurate open-source OCR engines [Ltd, Wil].

### 3.5.1 Character Training

OCR software can easily read popular fonts like Arial and Times New Romans, but it struggles with more artistic or exotic fonts. At this point character training is required. The process of OCR model training consists of the following steps:

1. Preparation of the training data set;
2. Normalization;
3. Training the OCR model with the data set.

#### 3.5.1.1 Preparation of the training data set

To get the best recognition accuracy, the training character sets should be as similar as possible to those which will be provided for recognition. To obtain the sample characters a possibility is to extract them from the real product images. The ideal situation is to train using various real samples of the characters. However, this process could be exhausting. To get around this, it's possible to create a set of the same character modifying it using the flowing operations:

- Rotation;
- Dilatation and erosion;
- Shearing;
- Addition of noise.

#### 3.5.1.2 Normalization

Normalization resizes every character to a single size. This will allow OCR to recognize characters of various sizes, also reducing the amount of data used in character classification.

#### 3.5.1.3 Training the OCR model with the data set

After the creation of the training data set its required to train the OCR model. The OCR engine used on this thesis (Tesseract) comes with a training tool which makes use of neural networks. After this training, its outputted a file with the trained model.

# Chapter 4

# Methodology

In the chapter 2 and 3 was discussed the state of the art of the currently available solutions and methods. The objective of this chapter is to present more in depth the details of the implementation in order to better understand the work done. All the techniques used are presented, is explained what they do and why is beneficial to use them, beginning with a brief overview of the problem and the main algorithm.

## 4.1 The problem

The process starts with the ShopView solution, a cart passes thought the halls of a supermarket and takes high resolution images of the shelves with four camera positions (top, middle-top, bottom, middle-bottom), Figure 4.1 shows some examples of those images. Each set of images taken in each corridor is called a session, the number of pictures for each session depends of the corridor and shelves size.

The ShopView solution provides a XML for each session that contains information about the session and the images (paths for the images, physical-coordinates of each image, type of products in the corridor and other non-relevant information for this thesis). This XML file is a required parameter to execute our algorithm. For the development of this thesis we have access to a dataset of sessions, having each session about 50 to 150 images.

The objective of this thesis is to locally detect the products and recognize the key words that describe the product to create a meta-data that will be sent to a server for posterior processing by the ShopView solution. These detected words will be used by ShopView to identify the products with the help of a database that provides the products description (this database is already available in some retailers). If a product is not available in that database, it is possible to create the description of the product with the detected words. To process all the data, the algorithm is execute in a computer with an octa-core processor and at least 8GB of ram, all the tests and development were done in a similar computer.


(a) Top camera


(b) Middle-Bottom Camera


(c) Bottom Camera

Figure 4.1: ShopView Session Example Images

## 4.2 Overview of the algorithm

The diagram in the figure 4.2 describes the steps of the main algorithm.

The algorithm starts by loading the provided XML with the session information, from that information we load the paths to all the images. For performance purposes, the algorithm creates a number of processes equal to the number of cores the computer has (in our case is 8 process), giving to each process an equal number of paths of images to process, this will allow to process the images in parallel. Each process will execute the main image processing steps, for each iteration of each process an image is loaded and the pre-processing step starts (the details of this pre-processing will be latter addressed in 4.3). The pre-process step, among other things, includes the text location part in which returns the location of the segments of text in the image. After this, the text-processing part of algorithm takes place (described in detail in 4.4) in which the OCR engine recognizes the text in those text regions. This text is then clustered, the identical pieces of text are matched and also grouped together to the corresponding type of product. To finish, in the segmentation step 4.5, the product isolation algorithm is executed. Finally, after all the images have been processed, besides cleaning the memory, all the results are presented to the user or saved to a XML file for latter processing by the ShopView solution.

## 4.3 Image pre-processing

This step does the necessary operations to send the pieces of text to the OCR engine, as show in the image 4.3 it comprises of three steps. It starts by optimizing the image for the next algorithms, beginning with a resize operation to make the image smaller, this has no negative effects

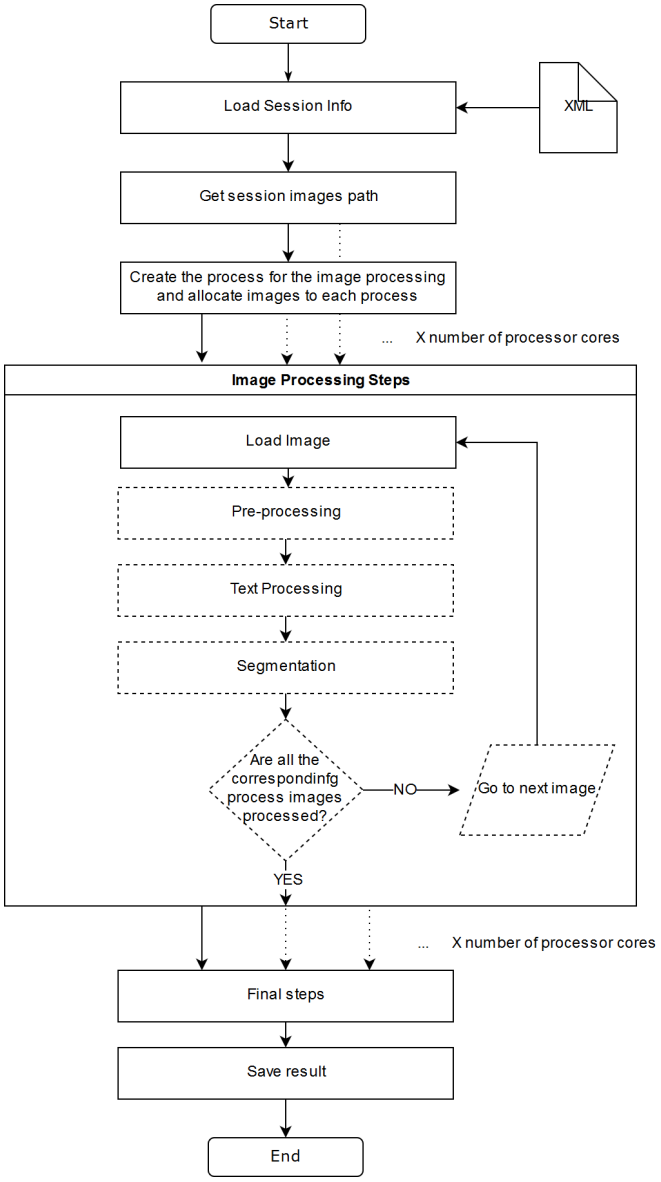



Figure 4.2: Main Algorithm Overview

in the overall algorithm but improves the performance and memory management. It is also done a transpose and flip operation to rotate the image to the correct orientation.

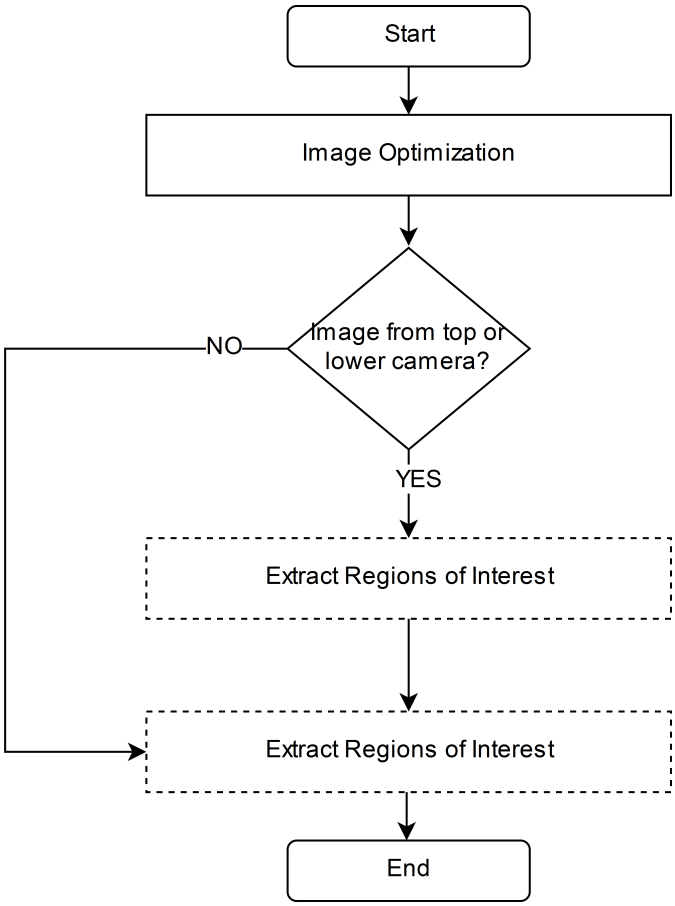



Figure 4.3: Pre-Processing Algorithm Overview

In the next step, the algorithm decides if the image requires to extract the regions of interest, this ROI's will be only calculated if the image comes from a top or bottom camera. Initially these ROI's were calculated for every image, but due to bad results (latter explained in the section 5.1) it was decided to use only on the top and bottom camera images. In the sub-chapter 4.3.1 are described the reasons for that.

Finally, the regions with text are extracted, this is explained more in depth in section 4.3.2.

### 4.3.1 Regions Of Interest (ROI) Extraction

In Figure 4.4 is shown an overview of the ROI exatraction algorithm.

The top and bottom camera photos usually have large regions with no products, as seen in Figure 4.5 and 5.1, that need to be discarded to optimize the algorithm in the text location (having less area to search). There are various types of regions with no interest such as empty spaces, the ground, ceiling, walls and shelves divisions. To identify the regions of interest it's used feature detection, assuming that a product is composed by a large region of features, these features came

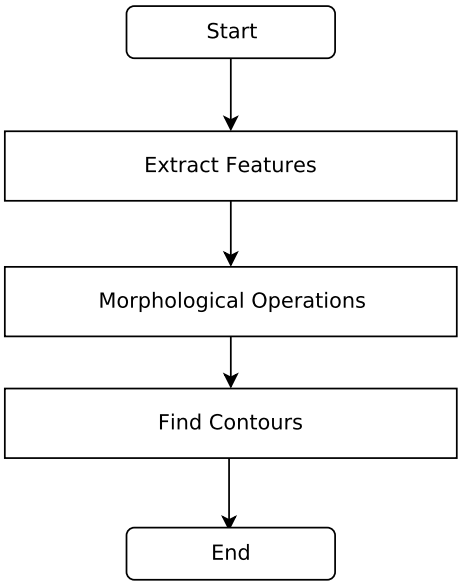
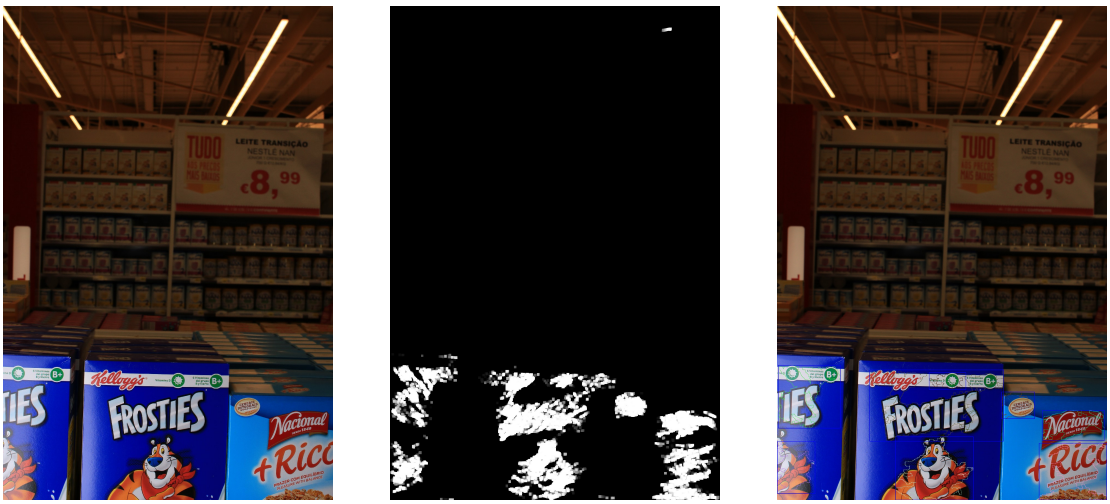



Figure 4.4: ROI Detection Algorithm Overview

from the graphics and words on the products packaging. On the other hand, the regions we want to discard have no (or minimal) features.




(a) Source (b) Features (c) Countours

Figure 4.5: Top camera image product isolation. As we can see, we perfectly isolate the featured products from the ones of the other corridor and the ceiling.

The used feature detector is the Canny algorithm. The recognized features are drawn as points in a black image of the same size of the original. With the image of the drawn features, we used the probabilistic Hough lines algorithm configured with custom parameters with the output filtered

to not include vertical and horizontal lines. Those detected lines were drawn in a black image of the same size of the original, so that this new binary image only contains the representation of the detected features.

In order to highlight the regions with features, creating better representation of the products location, we appeal to the use of morphological operators, this made these regions denser and suitable for the next step 4.5b.

Finally, its used an algorithm to find contours, this will create groups of high feature regions. It's expected that those groups will represent the areas with products, so we discard all the smaller areas of features 4.5c.

This algorithm ignores background objects, isolating the target ones. A successful example of the application of this technique to isolate the products from the other environmental objects is shown in the Figure 4.5 which takes a top camera image and isolates the featured products from the ceiling and the objects of other shelves.

### 4.3.2 Text Extraction

Correct detection of the text in the image is a fundamental part of the algorithm for further recognition and processing.

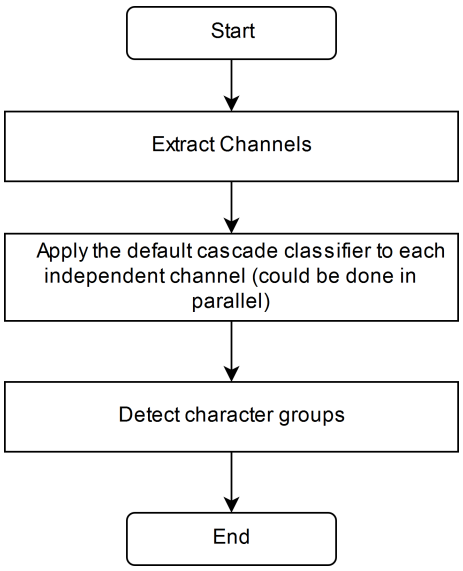



Figure 4.6: Text Extraction Algorithm Overview

A solution that brought good results with a decent processing time was the algorithm presented by Neumann in [NM12]. For the algorithm, was used a pre-trained classifier and only two channels of the source image for faster processing time, unlike the default approach of using all the channels.


Figure 4.7: Example of the text detection

In figure 4.6 we can see the overview of the Neumann's algorithm in use and in Figure 4.7 and example of the detected segments of text in an image.

## 4.4 Text Processing

After pre-processing the image, as a result we have a set of regions with a high probability of text. To recognize the text in those regions, we have to pass them though an OCR engine, in our case, we use Tesseract Open Source OCR Engine [tes].
Initially we use the Tesseract engine without any custom configurations, but for optimal results, we tested custom configurations for the allowed characters and dictionary.

### 4.4.1 Text Filtering

After the OCR, the filtration of the recognized text is required for many reasons:

- Almost all he sections used for the OCR have noise created by the product packaging due to their graphics and deformations. This noise results in recognition of isolated characters that don't really exist.

- Generic and small length words such as "a", "de", "para", "preço", etc... Because these words don't bring any additional information about the product and can be easily found in any product. This could also lead to false matches in text clustering 4.6 step.

- Depending of the allowed OCR charset, we could maybe have need to discard special characters and punctuation marks from our text since it does nor provide any additional info.

- Sometimes the recognition confidence of the OCR engine is too low to accept, so it's possible to discard the recognized text with a confidence under some specified value.

These filtrations also improve the overall speed of the implemented algorithm thanks to the smaller set of words. Follows an example of this filtration with the recognized text of the Figure 5.3: "ALTO TEOR EM FERRO e VITAMINAS: BL Ba Ba BEL 89 812 Ba". After filtration we have: "ALTO TEOR FERRO VITAMINAS". We can see that, the remaining words after filtration are closer to key-words and the others considered as trash were discarded.

## 4.5 Product Segmentation

Having for the hole image the text recognized and its specific location it's possible to have a rough idea of where there are products in the image. But, it's necessary to have a better (or exact) sense of the products location and their limits in order to associate all the dispersed segments of recognized text to a product. In other words, it's required to segment the products on the image.

Inspired in some character segmentation techniques [JSDK], it was implemented an algorithm with the help of the histogram of horizontal and vertical projections of the image. We segmented the image initially into shelves and then we took each of those selves and segment the products in them. The highlight of this algorithm regarding the common character segmentation techniques with the image projections is the adaptive threshold for the Canny algorithm.

An overview of the algorithm is shown in Figure 4.8.

First the algorithm calculates the horizontal projection of the image 4.9a, with that projection we apply the Canny algorithm to find the edges, this will dictate us the horizontal divisions that represent the shelves, as we can see in figure 4.10a. The algorithm starts by a pre-defined first and second threshold for the hysteresis procedure of Canny algorithm and changes them as needed. In the first iteration, if no division is found with canny, the algorithm changes both thresholds and tries again until the first threshold reaches 250 and the second threshold reaches 0.

Then, the algorithm takes each horizontal division (shelve) separately and does the same, but this time for the vertical histogram (products).

In this implementation, we discard boxes smaller than a desired size, this is a type of a filter to reduce the number of gaps (between products) to be further processed.
This has resulting in good segmentations even when the position of the products are not perfect as we can see in the examples of Figure 4.11.

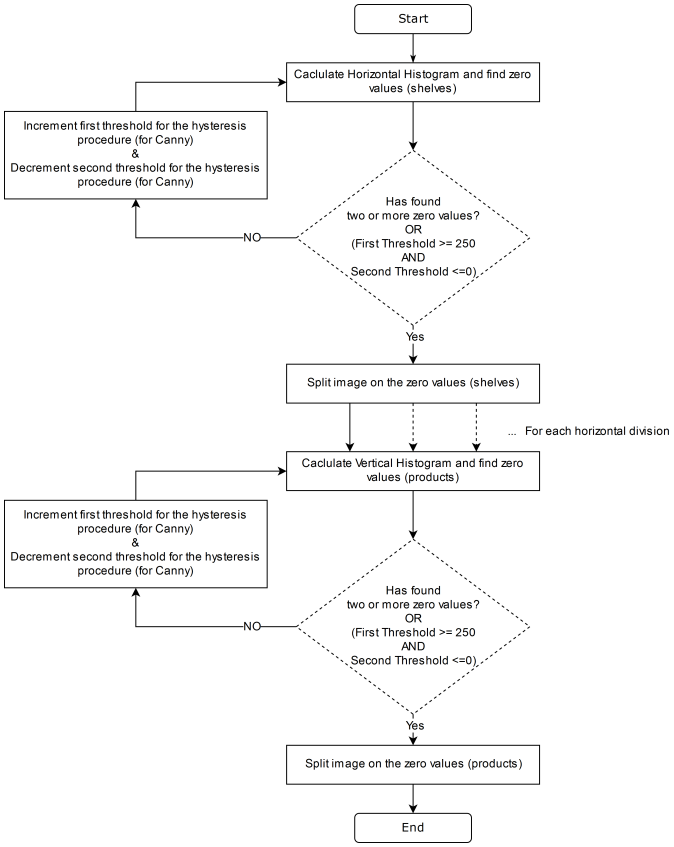
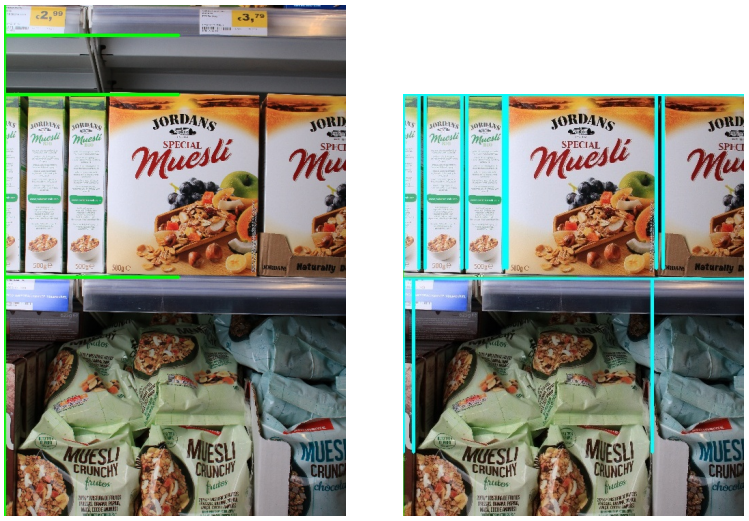



Figure 4.8: Overview of the Segmentation Algorithm

(a) Horizontal Projection

(b) Vertical Projections for each shelf

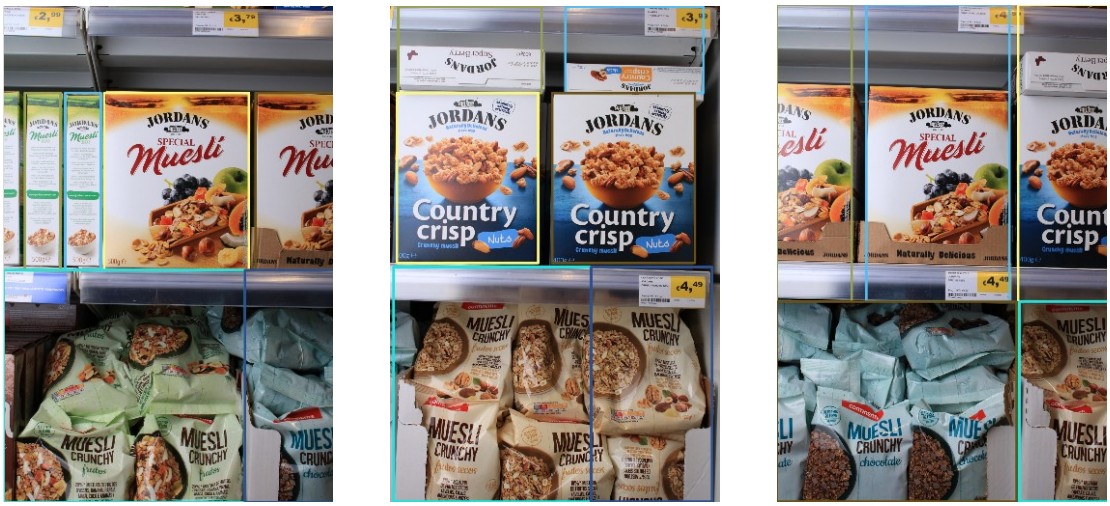
Figure 4.9: Image Projection




(a) Horizontal Projection

(b) Vertical Projections for each shelf

Figure 4.10: Output after Canny




(a) (b) (c)

Figure 4.11: Examples of segmentation

## 4.6 Text Clustering

After having each product segmented and the recognized segments of text (with their coordinates) we will associate all the segments of text to the corresponding products, grouping those segments into one for each product. This is done by a simple calculation, checking if the segment of text is inside the area of the product. But first we can relate the products by their text, this is done comparing the text segments of the products to each others. This comparison of text segments had one flaw, it could be easy if we had the exact text on each segment, but Tesseract has always an associated error in the recognition, resulting frequently in wrong character recognitions and unrecognized words/characters, even after the filtering explained in 4.4.1. To solve this problem we implemented an approximate string matching algorithm describe in 4.6.1. With this approximate string matching was possible to detect identical products in the image even with bad recognitions. This helped us to get a bigger bag of text associated to a type of product when more than one instance of a certain product is present on the image. Even with some text that we didn't recognize in one instance of the product we could have that text recognized in other instance. This is also helpful to match the same products even when they have different facing faces, for instance, we can match the side of a product to the front of another due to the word repetitions in all the faces of a product (usually flavors, brand names or slogans). Also, imagining we have three products in the image of the same type, at a specific position in the products on has the text "benena" as recognized but the other two have "banana" we can assume that "banana" is the correct recognized word for being the more frequent.

### 4.6.1 Approximate string matching (fuzzy string searching)

To match repeated segments of words taking into account the errors associated to the OCR, its required a mechanism that can relate those wrongly recognized words despite the small differences, finding approximate words rather than exactly. To compare the segments of words, first those segments are split into words and for each word we compare to the words of other segments using the Levenshtein distance. This approach was successful and a good example of this can be seen in the Figure 4.12.


Figure 4.12: Paired words

# Chapter 5

# Results and Discussion

In this chapter will be discussed the different approaches to the different sections of the main algorithm, referring the results and reasons for not using them. We start by referring the different approaches for the text extraction 5.1. In section 5.2, we explain the different approaches taken to improve the recognition of the text. In section 5.3 we explain the various ways we tried to segment the products. Finally, we show the results and the various attempts to improve the overall algorithm performance

## 5.1 Text extraction

Although in the final version of the algorithm we use the feature detection to identify regions of interest, i.e. to find the products in images with large areas that have no products (from top and bottom cameras), the initial approach was to use the features to detect the text in products. Since the source image is very large and with pieces of text at random locations, there was a need to find and extract the relevant regions for a faster location of the text. These regions of interest (ROI) were defined by areas with high probability of text. There are a series of the feature detection algorithms, as stated in the chapter 4. At first, we used the probabilistic Hough transform algorithm for line detection with some tweaking (defining the optimal line size range and ignoring horizontal lines so that the shelves separation was not included in the set of detected lines since this didn't affect anything more). With this we hope to find the most detected lines where the words were, but it turned out not to be the case, large regions of lines were detected though all the image.

Next thing we tried was to use other feature detectors such as FAST and Canny. The Canny edge detector seemed the most suitable because the number of features on the regions with products were higher than with FAST. Then, we applied the Hough algorithm and the morphological operations (identical to the process explained in section 4.3.1 as seen in Figure 5.1.

This approach did no bring advantages because a large number of regions had no text in it, and in some images most of the text was not associated to any of those regions as seen in Figure 5.2.

(a) Features detected with Hough Lines algorithm (after morphological operations) from the image of features found with Canny


(b) Detected Contours

Figure 5.1: Detected ROIS with feature detection

(a) Features detected with Hough Lines algorithm (after morphological operations) from the image of features found with Canny


(b) Detected Contours

Figure 5.2: Detected ROIS with feature detection with no text

The use of close edge elements technique is a known way to find text but it was discard for our case because it would result in many false positives due to the nature of the algorithm and the graphics present in the products packaging. This technique works better in cases of scanned documents or uniform backgrounds.

Being mandatory the text detection for the future text recognition, these approaches had a huge negative effect in the product recognition so they had discarded in the sense of finding regions with text.

The best solution found, and the one that's being used, is the Neuman's algorithm [NM12]. The problem found with this approach was that not all the text can be found, artistic fonts present in the brands name for instance are problematic. We hope that his problem can be solved by building a classifier with a more specific dataset of characters, although this solution will require to learn new characters for each product with very different artistic fonts.

Applying the text detection algorithm to the whole image although increasing the processing time, is better due to the higher detection of the text.

## 5.2 Text Recognition

In order to improve the text recognition we tried some approaches explained in the following sub-sections.

### 5.2.1 Different colors under a word or in his characters

After the regions with text are known, we tried to improve the OCR accuracy by isolating the text from the background. This was done by adding the different channels from the connected components used in the text location Neumann's algorithm. The output was a binary image of the text, but always had some noise associate to it. After various tests, the result of using this in the OCR engine was worse than using the source image. Since the OCR engine seem to deal better with the source image this step of isolating the characters was no longer used.

### 5.2.2 Custom Charset

When images had noise (that's the majority of cases) the OCR engine was having some issues to recognize the text. That noise in the image resulted in an output filled with random special characters such as dots, commas, interrogation and exclamation marks, etc...

In order to remove such noise, we decided to configure Tesseract to just allow a custom char-set of the most relevant characters for our context. After testing some sets of characters, we come up with the optimal set, the allowed characters are: "aáaãbcçdeêéfghiíjklmnoõópqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789:" After this new custom charset, the output was improved, with less noise and better recognitions. For the matter of finding the optimal charset for this problem, was also tried a set of only letters and no numbers (because the numbers where not

so important) or special characters, but it revealed to be an error prone solution because Tesseract tried to match the numbers in the image to the allowed letters.


(a) Source image (b) Binary image

Figure 5.3: Example image for OCR

For example, sending the binary image on the Figure 5.3 to Tesseract with the default charset gave the following output: "ALTO TEOR EM » “FERRO e VITAMINAS: B1, 82, B5, B6, Bg, B12, Ba.". The OCR engine matched the noise to """ and "»". With the custom charset it gave the output: "ALTO TEOR EM FERRO e VITAMINAS: BL Ba Ba BEL 89 812 Ba".

If the custom charset reveals to be problematic because it tries to match the special characters to other allowed characters (sometimes Tesseract matches an "o," to "a" because the comma is not allowed, so Tesseract tries to match that text so something allowed), another alternative is to allow all the characters but in the filtering step 4.4.1 we remove all the special characters (because they do not bring any additional information).

### 5.2.3 Custom Dictionary

The purpose of the text recognition is to recognize key-words for the retail - a distinct set of words that together let us identify and describe the products, such as: flavors, brands and quantities. Implementing a custom dictionary for the OCR engine seemed a good strategy since at first glance it would improve the performance of the OCR algorithm (because it had a smaller set of words to recognize) and in the output, was given only the relevant words that we wanted. A dictionary was strategically created and tested for a set of images, but the result was not satisfactory. The processing time had not significantly improved and the shortage of the set of words was creating an unexpected issue in the next step - having fewer recognized words was making more difficult to match products based on the text found in the packaging, for example, in descriptions of the product or slogans, since most times it does not contain retail specific words.

In the end, Tesseract was configured to use both Portuguese and English dictionaries.

## 5.3 Product Segmentation

The initial approach to segment the products was to match the segments of text to each other’s as seen in Figure 5.4 (and better explained in 4.6), this approach brought information about the

dimensions and the number of products calculating the distance between matches (identical pieces of text). This also helped to associate the dispersed pieces of text to a product since we had a rough idea of the area of the product in the image.


Figure 5.4: Paired words

This was not an optimal solution and we could have situations where the recognition of the text for one product was too bad (or very few words recognized), letting us with no enough information for this algorithm to work. Also, in cases where there is a sequence of different products we couldn't relate the text because they are all different, so we couldn't segment the products although the text was recognized.

Another try was using the Hough Lines algorithm to find the shelf tiers with custom parameters to find the larger lines. Next, the same algorithm was used, but for vertical lines (detecting products). This revealed to not be a good solution, there was too many false positives when rectangular products where together. When there was a separation between the shelves the algorithm was incapable of finding the shelf tires (because the separation made the lines of the shelves shorter

and could be confused with a product due to the size). It's also been equated and tested the use of superpixel segmentation [BBRG], but this options was quickly discarded because in the initial tests it didn't show any promising results. The segmentation appeared to be a bit random for our case due to the different graphics on the products packaging.

The better solution we came up was to use the vertical and horizontal projections of the image as explained in section 4.5. On the final solution, we tried to use a Gaussian Filter (additionally to the used in Canny) to blur the image in order to remove noise, preventing false detections. With this addition, no improvements were noticed. The Gaussian Filter used internally by the Canny algorithm was enough.

## 5.4 Results

For the testing was used a set of 20 images, having 118 products in total. It was analyzed the performance of the segmentation algorithm and the recognition of the keywords.

### 5.4.1 Segmentation

The segmentation phase was able to detect and segment the products with a reasonable success rate. We measured the success rate with two types of metrics:

- **Segmentation of each single product** 5.5a, had a success rate of 61% out of 118 products.
- **Segmentation of regions**, this is the most relevant metric for the purpose of this work, a region is well segmented if it is composed by one single product or multiple products of the same type 5.5b. It had a success rate of 79% out of 103 regions.

Most of the errors on the segmentation came from overlapping products 5.6a or deformed packages 5.6b.

### 5.4.2 Keywords

After segmenting each image by product type, resulting on 40 clusters, we recognized the words of those segments, being those words associated to a product type. The considered keywords are: brand name, product name, slogans, flavors or key-ingredients and dosages.

We were presented with the following results 5.7.

Of all the recognized words, we noticed that the most common keywords were the product name and flavors or key-ingredients, the brand name was the lest recognized keyword due to the artistic fonts 5.8.

We noticed that the main problem came from the artistic fonts, this caused difficulties on the location of that text and the recognition of its characters. A possible solution is to train the classifier with those artistic fonts, but it will go against the objective of this work, since it will require to train the classifier each time new fonts are introduced on the products package.


(a) Well segmented product


(b) Well segmented region

Figure 5.5: Example of segmentation


(a) Bad segmentation on overlaping products


(b) Bad segmentation on deformed packages

Figure 5.6: Example of bad segmentation

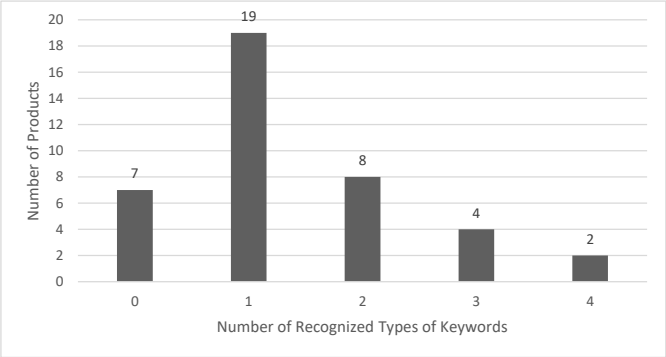



Figure 5.7: Number of recognized keywords

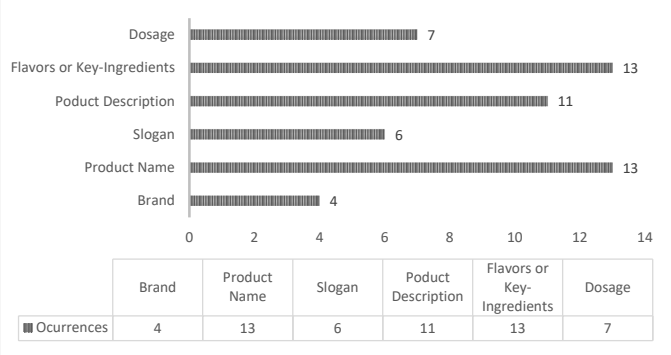



Figure 5.8: Number of recognized words by type

## 5.5 Performance improvements

Concerning to performance we did try to improve it by making some tweaks along the algorithm.

First, we implemented the algorithm in such way that it's possible to process the images in a concurrent form, this has done a drastically improvement on processing time by taking full advantage of the multi-core processor.

Being the images high resolution there was no need to optimize the image for processing, but we did resize the image to a lower dimension on the pre-processing step in order to occupy less memory and improve the time of image processing algorithms due to a lower number of pixels.

In the text location problem, the definition of the Neumann algorithm, all the channels of the image are used, we changed it to use only two channels and evaluated the output. The time of processing had reduced but there was no noticeable impact on the output.

Regarding to the OCR algorithm, as already pointed, there was done two things: creating a custom dictionary with less words and a smaller char-set. None of those optimizations brought satisfactory results in terms of performance.

After the text recognition, the filtering process shortened the number of words to process by the following algorithms which improved the performance time and memory occupation.

Also, we tried to change the algorithm by putting the segmentation before the text location algorithm. With this, the algorithm instead of finding text in the hole image, it searched for text product by product (like a divide and conquer strategy), but surprisingly it leads to a overall slower processing time and no real advantages over sending the whole image.

# Chapter 6

# Conclusions and Future work

It can be concluded that the objectives proposed on 20 February were reached by 23 June (19 weeks).

This work focused on the study and application of computer vision algorithms for the creation of retail planograms from high-resolution images of the real shelves. Several methods were studied and tested in order to get the best results of each step of the image processing.

The main algorithm is composed by several parts. Each one of the parts plays a crucial role in the recognition of the products. For each part, various configurations and techniques where tested and analyzed.

On the text extraction step, from the different approaches tried, the best solution was the Neuman's algorithm. Although the feature extraction algorithm was not able to detect the regions of text, we were able to take advantage of that implemented technique to discard regions of the image with no products. On the text recognition, we weren't able to make significant improvements for OCR despite the custom charset. For the product segmentation, from all the approaches, we came up with two different working solutions. The segmentation using the similar text was working but had some issues already mentioned. The optimal solution to this segmentation, with great results, was to segment the image using the vertical and horizontal projections with an adaptive threshold for the Canny algorithm.

As shown, it is possible to implement an algorithm to build planograms without the need of machine learning. Being thus, this solution is a better alternative regarding to the deep-learning based implementations because this approach doesn't need a large and constantly updated database of images (for the leaning). Also, it has immunity to product package and appearance changes and the ability to distinguish products with minimal differences between them. Therefore, the implemented technique leads to a better usability and practicality for not requiring additional and constant user interactions.

These characteristics are especially important to the real-world applications in terms of scalability and usability. On the daily basis, the retailers receive products with constant appearance changes, the use of deep-learning techniques would be impracticable due to the constant required updates of the database for the template matching - having to take pictures of the products throughout the entire life. The lack of a shared database between retailers of a country/market that centralizes the changes between products makes the deep-learning techniques impossible to support in terms of costs of operations and maintenance.

## Future work

The continuation of the development of this solution will pass by the integration of the output of the algorithm in ShopView. In the future, it should be developed a more reliable option to the text detection of the artistic fonts and the respective recognition (can be by training the classifier with artistic fonts used in retail for the current algorithm). Also, the work done does not consider products which are positioned in reverse, when this occurs it's impossible to detect or recognize the text due to the nature of the algorithms.
Besides knowing the words recognized in the products, it's left to be done the tests against a database of product descriptions (with ShopView) to know if the recognized text is sufficient to recognize exactly the products and build the planogram.
Finally, it must be done a series of tests in every type of products in the retail stores to see how the algorithm behaves.

# References


[AA15] Anne-Marie Tousch Adrien Auclair. Method for the automated extraction of a planogram from images of shelving, 2015. URL: http://www.google.com/Miscs/US9141886.

[ABD12] Pablo Fernandez Alcantarilla, Adrien Bartoli e Andrew J. Davison. *KAZE Features*, pages 214–227. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. URL: http://dx.doi.org/10.1007/978-3-642-33783-3_16.

[AO12] Fredrik Linaker Agata Opalach, Andrew Fano. Planogram extraction based on image processing, 2012. URL: http://www.google.com/Miscs/US8189855.

[BBRG] Michael Van den Bergh, Xavier Boix, Gemma Roig e Luc Van Gool. SEEDS: Superpixels extracted via energy-driven sampling. 111(3):298–314. URL: https://link.springer.com/article/10.1007/s11263-014-0744-2, doi:10.1007/s11263-014-0744-2.

[BETVG08] Herbert Bay, Andreas Ess, Tinne Tuytelaars e Luc Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, June 2008. URL: https://www.sciencedirect.com/science/article/pii/S1077314207001555, doi:10.1016/j.cviu.2007.09.014.

[BGR07] W. Bieniecki, S. Grabowski e W. Rozenberg. Image Preprocessing for Improving OCR Accuracy. In *2007 International Conference on Perspective Technologies and Methods in MEMS Design*, pages 75–80, May 2007. doi:10.1109/MEMSTECH.2007.4283429.

[BMB13] Sudipto Banerjee, Koustav Mullick e Ujjwal Bhattacharya. A Robust Approach to Extraction of Texts from Camera Captured Images. In *Camera-Based Document Analysis and Recognition*, pages 30–46. Springer, Cham, August 2013. DOI: 10.1007/978-3-319-05167-3_3. URL: http://link.springer.com/chapter/10.1007/978-3-319-05167-3_3.

[Can] J. Canny. A computational approach to edge detection. PAMI-8(6):679–698. doi:10.1109/TPAMI.1986.4767851.

[Cor] Intel Corporation. Planogram Compliance Automated with Image Recognition Technology. URL: http://www.intel.com/content/www/us/en/retail/solutions/ensure-product-placement.html.

[CSDK07] Chanjin Chung, Todd M. Schmit, Diansheng Dong e Harry M. Kaiser. Economic evaluation of shelf-space management in grocery stores. *Agribusiness*, 23(4):583–597, September 2007. URL: http://onlinelibrary.wiley.com/doi/10.1002/agr.20141/abstract, doi:10.1002/agr.20141.

[CSJ16] H. Cho, M. Sung e B. Jun. Canny Text Detector: Fast and Robust Scene Text Localization Algorithm. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3566–3573, June 2016. doi:10.1109/CVPR.2016.388.

[DH72] Richard O. Duda e Peter E. Hart. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Commun. ACM*, 15(1):11–15, January 1972. URL: http://doi.acm.org/10.1145/361237.361242, doi:10.1145/361237.361242.

[FEHF09] A. Farhadi, I. Endres, D. Hoiem e D. Forsyth. Describing objects by their attributes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1785, June 2009. doi:10.1109/CVPR.2009.5206772.

[Foo15] Food Marketing Institute. Supermarket Facts. http://www.fmi.org/research-resources/supermarket-facts, 2015.

[FSZ15] Y. Feng, Y. Song e Y. Zhang. Scene text localization using extremal regions and Corner-HOG feature. In *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 881–886, December 2015. doi:10.1109/ROBIO.2015.7418882.

[Groa] GroZi. A Grocery Shopping Assistant for the Visually Impaired . http://grozi.calit2.net/.

[Grob] GroZi-120 Database. GroZi-120 Database. http://grozi.calit2.net/.

[GW15] Jingrui Zhang Guan Wang. Recognizing characters from google street view images, 2015.

[GZC15] Guangyu Gao, He Zhang e Hongting Chen. A Robust Video Text Extraction and Recognition Approach Using OCR Feedback Information. In *Advances in Multimedia Information Processing – PCM 2015*, pages 507–517. Springer, Cham, September 2015. DOI: 10.1007/978-3-319-24075-6_49. URL: http://link.springer.com/chapter/10.1007/978-3-319-24075-6_49.

[JR89] Jean-Michel Jolion e Azriel Rosenfeld. An O(log n) pyramid hough transform. *Pattern Recognition Letters*, 9(5):343–349, June 1989. URL: http://www.sciencedirect.com/science/article/pii/0167865589900639, doi:10.1016/0167-8655(89)90063-9.

[JSDK] J. Jagannathan, A. Sherajdheen, R. M. Vijay Deepak e N. Krishnan. License plate character segmentation using horizontal and vertical projection with dynamic thresholding. In *2013 IEEE International Conference ON Emerging Trends in Computing, Communication and Nanotechnology (ICECCN)*, pages 700–705. doi:10.1109/ICECCN.2013.6528594.

[KB15] T. Kumuda e L. Basavaraj. Detection and localization of text from natural scene images using texture features. In *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pages 1–4, December 2015. doi:10.1109/ICCIC.2015.7435688.

[KEB91] N. Kiryati, Y. Eldar e A. M. Bruckstein. A probabilistic Hough transform. *Pattern Recognition*, 24(4):303–316, January 1991. URL: http://www.sciencedirect.com/science/article/pii/003132039190073E, doi:10.1016/0031-3203(91)90073-E.

[KK13] H. I. Koo e D. H. Kim. Scene Text Detection via Connected Component Clustering and Nontext Filtering. *IEEE Transactions on Image Processing*, 22(6):2296–2305, June 2013. doi:10.1109/TIP.2013.2249082.

[KVV13] Vishwanath C. Kagawade, C. S. Vijayashree e T. Vasudev. Transformation of Artistic Form Text to Linear Form Text for OCR Systems. In *Proceedings of International Conference on Advances in Computing*, pages 1135–1143. Springer, New Delhi, 2013. DOI: 10.1007/978-81-322-0740-5_138. URL: http://link.springer.com/chapter/10.1007/978-81-322-0740-5_138.

[KVV14] Vishwanath C. Kagawade, C. S. Vijayashree e T. Vasudev. Transformation of Artistic Form Text to Linear Form Text for OCR Systems Using Radon Transform. In *Emerging Research in Electronics, Computer Science and Technology*, pages 747–756. Springer, New Delhi, 2014. DOI: 10.1007/978-81-322-1157-0_76. URL: http://link.springer.com/chapter/10.1007/978-81-322-1157-0_76.

[LLD+16] S. Liu, W. Li, S. Davis, C. Ritz e H. Tian. Planogram compliance checking based on detection of recurring patterns. *IEEE MultiMedia*, 23(2), April 2016. doi:10.1109/MMUL.2016.19.

[Low] David G. Lowe. Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image. Classificação dos EUA 382/219, 382/220; Classificação Internacional G06K9/46, G06T7/00; Classificação Cooperativa G06K9/4671, G06T7/73; Classificação Europeia G06T7/00P1, G06K9/46R. URL: http://www.google.com/Miscs/US6711293.

[Ltd] Canonical Ltd. Canonical. URL: https://www.linux.com/news/googles-tesseract-ocr-engine-quantum-leap-forward.

[MAP16] V. N. Manjunath Aradhya e M. S. Pavithra. A comprehensive of transforms, Gabor filter and k-means clustering for text detection in images and video. *Applied Computing and Informatics*, 12(2):109–116, July 2016. URL: http://www.sciencedirect.com/science/article/pii/S2210832714000234, doi:10.1016/j.aci.2014.08.001.

[McC86] Robert K. McConnell. Method of and apparatus for pattern recognition, January 1986. U.S. Classification 382/170, 382/209; International Classification G06K9/48; Cooperative Classification G06K9/48; European Classification G06K9/48. URL: http://www.google.co.uk/Miscs/US4567610.

[MS05] K. Mikolajczyk e C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, October 2005. doi:10.1109/TPAMI.2005.188.

[NM12] L. Neumann e J. Matas. Real-time scene text localization and recognition. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3538–3545, June 2012. doi:10.1109/CVPR.2012.6248097.

[NM13] Lukáš Neumann e Jiří Matas. Scene text localization and recognition with oriented stroke detection. In *2013 IEEE International Conference on Computer Vision (ICCV 2013)*, pages 97–104, California, US, December 2013. IEEE. doi:10.1109/ICCV.2013.19.

[NM16] Lukáš Neumann e Jiří Matas. Real-time lexicon-free scene text localization and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 38(9):1872–1885, Sept 2016. doi:10.1109/TPAMI.2015.2496234.

[ope] OpenCV library. URL: http://opencv.org/.

[RD06] Edward Rosten e Tom Drummond. Machine Learning for High-Speed Corner Detection. In Aleš Leonardis, Horst Bischof e Axel Pinz, editors, *Computer Vision – ECCV 2006*, Lecture Notes in Computer Science, pages 430–443. Springer Berlin Heidelberg, May 2006. DOI: 10.1007/11744023_34. URL: http://link.springer.com/chapter/10.1007/11744023_34.

[RGC+16] L. Rosado, J. Gonçalves, J. Costa, D. Ribeiro e F. Soares. Supervised learning for out-of-stock detection in panoramas of retail shelves. In *2016 IEEE International Conference on Imaging Systems and Techniques (IST)*, pages 406–411, 2016. doi:10.1109/IST.2016.7738260.

[RRKB11] E. Rublee, V. Rabaud, K. Konolige e G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision*, pages 2564–2571, November 2011. doi:10.1109/ICCV.2011.6126544.

[SBF13] Yuyin Sun, L. Bo e D. Fox. Attribute based object identification. In *2013 IEEE International Conference on Robotics and Automation*, pages 2096–2103, May 2013. doi:10.1109/ICRA.2013.6630858.

[SHM15] A. Saran, E. Hassan e A. K. Maurya. Robust visual analysis for planogram compliance problem. In *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*, pages 576–579, May 2015. doi:10.1109/MVA.2015.7153257.

[SKPB00] K. Sobottka, H. Kronenberg, T. Perroud e H. Bunke. Text extraction from colored book and journal covers. *IJDAR*, 2(4):163–176, June 2000. URL: http://link.springer.com/article/10.1007/PL00021523, doi:10.1007/PL00021523.

[STJ15] Abhishek Sharma, Oncel Tuzel e David W. Jacobs. Deep Hierarchical Parsing for Semantic Segmentation. *arXiv:1503.02725 [cs]*, March 2015. arXiv: 1503.02725. URL: http://arxiv.org/abs/1503.02725.

[Tec] Vispera Information Technologies. Vispera information technologies. URL: http://vispera.co/.

[tes] Tesseract ocr. URL: https://github.com/tesseract-ocr/tesseract.

[TM] Tinne Tuytelaars e Krystian Mikolajczyk. Local invariant feature detectors: A survey. 3(3):177–280. URL: http://dx.doi.org/10.1561/0600000017, doi:10.1561/0600000017.

[Tos14] Toshiba. Object Recognition Scanner. http://retail-innovation.com/image-recognition-self-checkouts, January 2014.

[Tra] Trax. Trax - Real Time Retail Execution Solutions. URL: http://www.traxretail.com/.

[Uch14] Seiichi Uchida. Text Localization and Recognition in Images and Video. In David Doermann e Karl Tombre, editors, *Handbook of Document Image Processing and Recognition*, pages 843–883. Springer London, 2014. DOI: 10.1007/978-0-85729-859-1_28. URL: http://link.springer.com/referenceworkentry/10.1007/978-0-85729-859-1_28.

[Web] WebMarket Database. WebMarket an image database built for Computer Vision Research. http://yuhang.rsise.anu.edu.au/.

[Wil] Nathan Willis. Google's tesseract ocr engine is a quantum leap forward. URL: https://www.linux.com/news/googles-tesseract-ocr-engine-quantum-leap-forward.

[XQ09] J. Xiao e L. Quan. Multiple view semantic segmentation for street view images. In *2009 IEEE 12th International Conference on Computer Vision*, pages 686–693, September 2009. doi:10.1109/ICCV.2009.5459249.

[Yod] Yodiwo. Retail Business - Yodiwo. URL: https://www.yodiwo.com/solutions/yodigram/.