1 2 9 0

UNIVERSIDADE Đ
COIMBRA

João Pedro Santos Ferreira

System for the detection of bottle misalignments for a bottle spraying system of a glass manufacturer

Dissertation within the scope of the Master's degree in electrical and computer engineering supervised by Professor Doutor Hélder de Jesus Araújo and presented to Department of Electrical and Computer Engineering, Faculty of Science and Technology, University of Coimbra

July of 2024

System for the detection of bottle misalignments for a bottle spraying system of a glass manufacturer

Dissertation in the context of the Master degree in Electrical and computer engineering (speciality of Robotics and automation)

Author
João Pedro Santos Ferreira

Supervisor
Hélder de Jesus Araújo

Jury

President	Professor Doutor **Pedro Manuel Gens de Azevedo de Matos Faia**
Supervisor	Professor Doutor **Hélder de Jesus Araújo**
Vowel	Professor Doutor **Cristiano Premebida**

Institutional Collaboration

INSTITUTE OF SYSTEMS AND ROBOTICS UNIVERSITY OF COIMBRA ISR

VIDRO MECANICA Vidromecânica

Coimbra, July, 2024

"Nothing happens unless first a dream."
Carl Sandburg

Acknowledgements

Accomplishing work of this magnitude required the help of several people. In this section I will only mention the most important ones. However, my sincerest appreciation goes to all the others who have contributed in some way and are not mentioned.

First and foremost, I would like to thank Professor Hélder Araújo for the trust he placed in me when he agreed to be my supervisor, and also for all the support he provided when I was accepted for the research scholarship, which later became the topic of my dissertation. The truth is that I was looking for a challenge, and professor never doubted my ability to carry it out. For his guidance, for the constructive criticism and opinions that allowed me to improve my work, as well as for his availability, I express my deepest gratitude. I would like to thank Engineers João Curado and Rui Gonçalves from Vidromecânica for all their help with the necessary tests to validate the accuracy of this dissertation. I would also like to express my gratitude to my colleagues at ISR, who were always available and ready to help, particularly to Professor João Marcos, for his availability and assistance whenever needed, and my Master's colleague André Neto, for all his help and the productive nights at ISR. Focusing on the Department of Electrical and Computer Engineering, I want to thank the professionals from the Network Management room, who were always available to assist me with the connection of the camera to the department's ethernet network, and to my coursemates and close friends who were a great help, both in the development of this work and during break times.

Above all, I must thank my parents, who are the reason for my life and, therefore, it is also thanks to them that this work exists. Moreover, for giving me the opportunity to study for five years in University of Coimbra and for all their support throughout my academic journey.

I also want to thank the rest of my family (grandparents, uncles, cousins), who always believed in my potential and provided me with great moments that mean a lot to me.

A special thanks to a very important person in my life: my girlfriend Francisca. For the right words at the right times, for always lifting me up when I thought I couldn't go on, for pulling me away from work when I needed a break to see things from a different perspective, for her affection and love, and for making my academic journey in Coimbra more significant and special to me.

I must also thank my friends, with whom have been harder to meet since the beginning of this journey, but who provide me with unforgettable moments in life whenever we manage to get together.

Abstract

The topic of this dissertation concerns a challenge faced by the equipment manufacturer for the glass industry, Vidromecânica. The growing demand for efficiency and innovation has driven the manufacturing industry to adopt advanced automation solutions. In today's competitive environment, where consumer expectations are increasingly high, companies are constantly seeking ways to optimize their processes and improve the quality of their products.

The field of computer vision emerges as a crucial tool in this new era of industry, where automated processes are becoming increasingly predominant, driving us toward Industry 4.0.

This project aims to automate a spraying process that requires an integrated control system. During this process, a substance is applied to the external surface of the bottles to give them their final coating. These bottles are arranged in parallel lines, with the sprayer moving through the empty spaces between these lines, spraying only the outside. At this stage, a control system is essential to check if the bottles are correctly positioned. Otherwise, the sprayer must be stopped before applying the substance, thus avoiding potential health issues for the end consumers.

Moreover, automating the spraying process promises a range of significant benefits. Among these, a substantial increase in operational efficiency is expected, resulting in higher productivity and cost reduction. The precision and consistency provided by the computer vision system will also contribute to improving the final product's quality, ensuring a uniform finish on all bottles. Finally, by minimizing the waste of the sprayed substance, the system will help optimize resources and the environmental sustainability of the manufacturing process.

This work employs a computer vision system capable of capturing images using an industrial camera with an Ethernet interface and detecting bottles frame by frame through methods to be studied, such as the Hough transform, adapted for circle detection. Subsequently, possible misalignments are detected to prevent the substance from being sprayed inside the bottles.

This dissertation was carried out as part of the VidroTec 4.0 project - Bottle Spraying Control System, funded by the company Vidromecânica.

Keywords: Industrial inspection, Computer vision, Estimating circles and ellipses.

Resumo

O tema desta dissertação diz respeito a um desafio enfrentado pelo fabricante de equipamentos para a indústria vidreira, a Vidromecânica. A crescente demanda por eficiência e inovação tem impulsionado a indústria de fabricação a adotar soluções avançadas de automação. No cenário atual, onde a competição é vigorosa e as expectativas dos consumidores cada vez mais altas, as empresas procuram, constantemente, formas de otimizar os seus processos e melhorar a qualidade dos seus produtos.

A área da visão por computador emerge como uma ferramenta crucial nesta nova era da indústria, onde os processos automatizados se estão a tornar cada vez mais predominantes, impulsionando-nos em direção à Indústria 4.0.

Este projeto visa automatizar um processo de pulverização que demanda um sistema de controle integrado. Durante o processo em questão, é aplicada uma substância na superfície externa das garrafas, de forma a conferir-lhes o acabamento final. Estas garrafas são dispostas em linhas paralelas, sendo que o pulverizador percorre os espaços vazios entre essas linhas, pulverizando apenas a parte externa. Nesta fase, é essencial um sistema de controle que verifique se as garrafas estão posicionadas corretamente. Caso contrário, o pulverizador deve ser interrompido antes de aplicar a substância, evitando, assim, possíveis problemas de saúde para os consumidores finais.

Além disso, a automação do processo de pulverização promete uma série de benefícios significativos. Entre eles, espera-se um aumento substancial da eficiência operacional, resultando numa maior produtividade e redução de custos. A precisão e consistência proporcionadas pelo sistema de visão por computador também contribuirão para a melhoria da qualidade do produto final, assegurando um acabamento uniforme em todas as garrafas. Por fim, ao minimizar o desperdício de substância pulverizada, o sistema ajudará a otimizar os recursos e a sustentabilidade ambiental do processo de fabricação.

O presente trabalho recorre a um sistema de visão por computador, capaz de capturar imagens por meio de uma câmera industrial, com interface Ethernet e detectar, frame a frame, as garrafas por meio de métodos a estudar, como a transformada de Hough, adaptada para detecção de círculos. Posteriormente, procede-se à deteção de possíveis desalinhamentos, de modo a evitar a ejeção da substância para dentro das garrafas.

Esta dissertação foi realizada no âmbito do projeto VidroTec 4.0 - Sistema de Controle do Sistema de Pulverização de Garrafas, financiado pela empresa Vidromecânica.

Palavras Chave: Inspeção industrial, Visão por computador, Estimativa de círculos e elipses.

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

1D One-Dimensional

2D Two-Dimensional

3D Three-Dimensional

ASCII American Standard Code for Information Interchange

CEC Cold end coating

cm centimeter

CNC Computer Numerical Control

CNN Convolutional Neural Network

COCO Common Objects in Context

CRC Cyclic Redundancy Check

CVD Chemical vapor deposition

DCS Distributed Control System

ERSCD Edge Region Selection via Color and Depth

FN False Negative

FOV Field of View

FP False Positive

FPS Frames Per Second

FTADSP Feature-based texture analysis and defect segmentation with post-processing

GPS Global Positioning System

HEC Hot end coating

HMI Human-Machine Interface

IDS Imaging Development Systems

IMU Inertial Measurement Unit

IOT Internet of Things

IP	Internet Protocol
ISR	Instituto de Sistemas e Robótica
IZA	Forschungsinstitut zur Zukunft der Arbeit
LGN	Lateral Geniculate Nucleus
mm	millimeter
numpy	Numerical Python
opencv	Open Source Computer Vision Library
OSHA	Occupational Safety and Health Administration
PAC	Programmable Automation Controller
PID	Proportional-Integral-Derivative
PLC	Programmable Logic Controller
POE	Power over Ethernet
R-CNN	Region-based Convolutional Neural Network
RGB	Red Green Blue
ROI	Region of Interest
RRPN	Rotated Region Proposal Network
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
sdk	Software Development Kit
SSD	Single Shot MultiBox Detector
TCP/IP	Transmission Control Protocol/Internet Protocol
TN	True Negative
TP	True Positive
UAV	Unmanned Aerial Vehicle
US	United States
WTMF	Wavelet transform multiscale filtering
YOLOv2	You Only Look Once version 2
YOLOv3	You Only Look Once version 3

1. Introduction

This project was carried out at the Department of Electrical and Computer Engineering, more specifically at ISR's Computer Vision laboratory but also, simultaneously, at the facilities of the company "Vidrala", (a client of Vidromecânica) based in Marinha Grande. Vidromecânica assured the necessary funding to ensure the proper development of this work, as well as provided all the necessary support during the integration of the system into the final product.

This chapter addresses themes such as context and motivation, with a contextualization of topics on industrial automation and computer vision systems. Following this, the problem and main goals, the state of art, which brings together some research and products that serve to conclude which are the most effective methods to be used, as well as a term of comparison of this product with what currently exists on the market. Finally the structure of the thesis will be mentioned, including a description of each chapter.

1.1. Context and motivation

1.1.1. Industrial automation's history

The term "industrial automation" means the replacement of human labor with eletromechanical machines, software, and specific equipment in dangerous, repetitive situations where significant physical effort is required. Typically, the emergence of this concept is dated to the mid-18th century, as it corresponds to the beginning of the Industrial Revolution, with the English invention of the steam engine by Thomas Savery (Bellis, 2019). Its advent is commonly associated with the century of the Industrial Revolution because it marked the birth of an entire industry on a global scale. However, factually and as previously mentioned, the term "automation" requires replacing a task usually performed by a person with a specific piece of equipment. According to this definition, it is possible to trace the term back to prehistoric times, when humans began using their intelligence to create mechanisms and inventions to assist in certain tasks, thereby reducing the necessary physical effort. Some examples that can be named include the wheel, used for moving loads, or even mills, which were used in the process of grinding grains with the help of wind power or animal force (Roggia and Fuentes, 2016).

Returning to the 18th century, with the creation of the steam engine, there was an evolution in the mode of production: humans began to produce goods on a large scale, aiming to increase productivity. With this industrial revolution, new technological innovations emerged, namely modern machines capable of greater precision and speed compared to manual labor, and new energy sources, such as steam (Roggia and Fuentes, 2016).

Later, in the 19th century, Henry Ford envisioned something he called the ”assembly line”. This concept represents the division of all the processes involved in manufacturing a product, where each employee or team focuses exclusively on their own stage. The idea of the ”assembly line”, along with the advent of electric power, initiated the second Industrial Revolution, which led to the development of new concepts in the sector, namely the notion of mass production.

Across the ages, the industry has been constantly evolving. More recently, with the creation of robotic systems and PLCs, the current industry features the automation of many processes to replace human labor in repetitive and dangerous situations that can be performed through automation processes.

1.1.2. Computer vision system

To implement an industrial automation system, eletromechanical machines are needed to process all data and control the entire system. For these functions, computers, PLCs (programmable logic controllers), DCSs (distributed control systems), or CNCs (computer numerical control) are used. In this research project, it will be used a computer vision system integrated into a computer, working side by side with other systems with computers and PLCs.

A computer vision system is a system that aims to reproduce the relationship between human eyes and the brain - the eyes capture information and the brain analyzes the provided data and acts according to its conclusions - which is extremely important in this project.

As humans, we rely heavily on vision to understand our surroundings. What we see through our eyes always depends on how our brain processes the captured information. When an image is detected, the light reflected from it passes to the eyes — directly to the retina (a layer located at the back of the eye that is sensitive to light), rods, and cones — as forms of light receptors. The rods, located at the periphery of the retina, process low levels of light and movement. On the other hand, the cones, which are predominantly located in the center of the retina, distinguish colors and details. After being captured, these signals travel to the brain through optic nerves, which are small cables of nerve fibers composed of a bundle of neuronal axons. The signals sent by the eyes are not directed straight to the brain. In other words, the signals captured by the right eye do not go to the right side of the brain, nor do the signals captured by the left eye go to the left side of the brain. Although the mentioned signals may initially follow a direct path (being sent from the right or left eye to the corresponding side of the brain), during their trajectory, there is a change in the direction of some axons at a point known as the optic chiasm. At this point, the nerves cross paths, allowing the primary cortex to receive information from both eyes. It is also where visual information is classified and divided for further processing. This seemingly simple process is fundamental, as it allows us a greater and more accurate perception of depth.

Following the previously mentioned process, the signals advance to a region of the brain called the thalamus, more specifically to its lateral geniculate nucleus (LGN). Here, visual data transmitted through the optic nerves is integrated along with the rest of the senses. Finally, this information is directed to the brain's visual cortex via axons or nerve fibers known as optic radiations. Finally, in the visual cortex, which has six distinct layers, the images received by the retina are processed, thus beginning the process of interpreting and recognizing what the human sees, where elements such as light, color, shape, shadow, depth perception, and movement are perceived. This entire process operates at the speed of light, leading to the processing of what we see (Chadwick Optical (nd), Baskin (nd)).

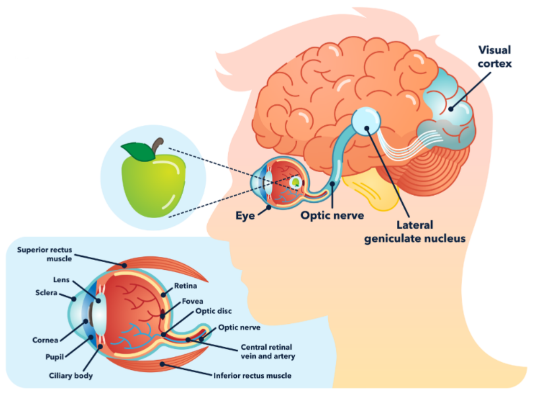


Figure 1.1: Process of processing and analyzing what the eyes capture (Baskin, nd)

1.1.3. Present situation and consequences

Currently, there are still some barriers that hinder the implementation of these systems, as well as their growth, which has been observed over the last few years. In this regard, various studies have been carried out, including one in particular from 2022 by Mordor Intelligence™ Industry Reports, which evaluated the market for factory automation and industrial control systems at 199,69 billion dollars (equivalent to 182,86 billion euros in Portugal) in 2024. This also estimates that, by 2029, it will reach 304,43 billion dollars (278,77 billion euros), denoting a growth of approximately 65.56% over a 5-year period (Mordor Intelligence, 2022).

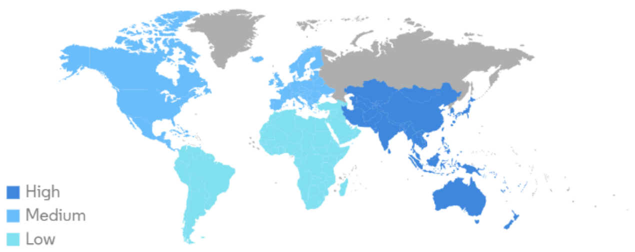


Figure 1.2: Global market for factory automation and industrial control - growth rate by region 2022-2027 (Mordor Intelligence, 2022)

It is true that industrial automation has many benefits, including: improving the cost/efficiency ratio; allowing 24-hour production, reducing production time; and also affecting workplace safety, as it decreases the likelihood of accidents. Regarding the last advantage listed, according to some studies conducted in the private sector in the US, workplace accidents have been decreasing between 2003 and 2017, a period that coincides with the intensification of automation in general (U.S. Bureau of Labor Statistics, 2018) (Figure 1.3). Another study conducted by the IZA (Forschungsinstitut zur Zukunft der Arbeit in German - Institute of Labor Economics in English) in 2020, explored the relationship between the insertion of industrial robots and automated systems and injuries predominantly related to industrial work, considering data provided by OSHA (Occupational Safety and Health Administration, US entity) related to American industry between 2005 and 2011. Based on the American data, it was possible to observe that an increase in the standard deviation in the insertion of robots and automated systems reduces work-related accidents by about 16% (Pichler et al., 2020).

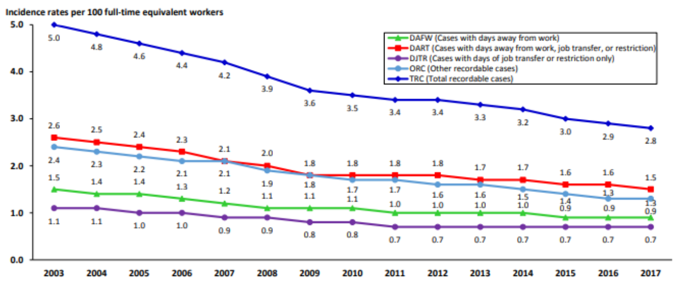


Figure 1.3: Incidence rates of non-fatal occupational injuries and illnesses by case type, private industry, 2003-17 (U.S. Bureau of Labor Statistics, 2018)

On the other hand, the process of automating industrial sectors is not as simple as previously thought. Factually, a significant initial investment is required, which yields significant returns in the long run. There is also the human factor, as it is imperative for employees to undergo a gradual and specialized learning process to correctly operate these systems. Therefore, a substantial investment, not only financial, is necessary to automate the industry and maintain competitiveness (Mitsubishi electric, 2022).

Nowadays, all companies and researchers seem to be working towards facing and overcoming these barriers to achieve the much-anticipated transition to Industry 4.0. The system addressed and implemented in this project may, in the future, be used according to these ideals, where all systems within a factory are connected according to the IOT (Internet of Things), and cyber-physical systems will be able to make autonomous decisions, avoiding delays during production. The purpose is that most of the computation will be done in the cloud to free up physical systems from this task, using simulations to predict potential failures. It is also intended that data processing will be carried out efficiently, being able to analyze and process large amounts of data from all systems within a factory.

1.2. Problem and main goals

This dissertation addresses an existing problem in the glass bottle manufacturing industry. With new regulations and food standards, the pressure on these processes is increasing. Dealing specifically with a cold surface treatment process where the bottles are sprayed with a substance in order to enhance the sliding between them and prevent scratching. The problem is that the sprayer moves along the middle of parallel lines of bottles, so that it sprays from the neck down, as shown in the figure 1.4. As a result of previous mechanical processes, some bottles may become misaligned with the rest of the line. Consequently, when the sprayer passes, it can knock over a bottle, or even spray inside it. Although the liquid is theoretically harmless, it can react with carbonated drinks, creating foam, which is undesirable according to current regulations.

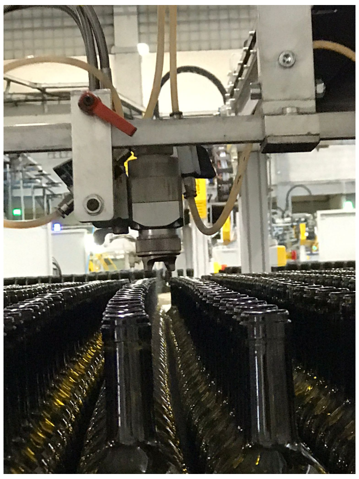


Figure 1.4: Image of the sprayer between 2 lines of bottles

Here there are two main aims. The first concerns to the entire development of the algorithm and its improvement, which is achieved through tests with image datasets, videos, and real-time tests to allow the algorithm to function fully and in a completely optimized manner when installed at the firm. The second goal is to proper integrate the system with the algorithm within the company's overall system, using computers and PLCs. This process ranges from positioning the different components of the system to the communication and data transfer between the algorithm and the rest of the company's systems.

1.3. State of art

1.3.1. Bottle detection

Bottle detection is a topic that has only recently gained the attention of the research world. With the technological evolution observed in recent years, product validation has become significantly important in industrial establishments, aiming to ensure that products are delivered to the user in perfect condition and that all safety requirements are met. Additionally, its increased relevance can be largely attributed to competition between companies, emerging as a consequence.

In the glass bottle manufacturing industry, defect detection systems have been developed to examine bottles, ensuring that only those in good condition are delivered to the end consumer. Ma et al. (2002) presented a glass bottle defect detection system that did not require physical contact with the objects. It used eight cameras installed alongside the production line to individually analyze the bottles, which captured images of the bottle's mouth, lip, and neck, and were capable of analyzing up to 20 bottles per minute. Simultaneously, two computers processed the captured images, later analyzing, recognizing, and communicating any defects to the system to ensure defective bottles did not continue down the production line. The recognition process was divided into three parameters: image orientation, defect extraction, and recognition of the same defects. For image orientation, the bottle's edges were delineated so that defect extraction could focus specifically on a small rectangle, which could correspond to the mouth and lip or the neck and shoulder of the bottle. Image binarization occurred at this stage, enabling the algorithm to scan the image to identify areas of pixels that might represent defects. Finally, object recognition was carried out, where the algorithm analyzed the characteristics of each area and distinguished the different types of defects.

In 2010, a group of students developed a glass bottle inspection system as part of their bachelor's dissertation. This system was designed to detect faults in the wall and mouth of the bottles using LabView, with defective bottles then being rejected by the conveyor. For this process, image processing techniques were applied, such as grayscale, pattern matching, edge detection, and object detection (An et al., 2010).

Zhou et al. (2019) published an article presenting a method for the automated inspection of glass bottle bottoms by analyzing different regions of this objects, such as the central region of the bottom, the annular bottom region, and the annular texture region. The proposed method involves multiple stages and detection algorithms. The location of the bottle bottoms is determined through Hough circle detection, which divides them into three parts. Then, saliency detection methods and multiscale filtering are used to detect defects. Finally, these defects are combined to assess the quality of the tested bottle bottom.

A group of researchers conducted a study aiming to propose a framework for detecting defects in glass bottle bottoms (Zhou et al., 2020). To achieve this, they used methods such as the ERSCD strategy (Edge Region Selection via Color and Depth), which combines color and depth information to identify edges in the bottle bottom, allowing for the location of regions of interest (ROI). Next, they used the FTADSP method (Feature-based Texture Analysis and Defect Segmentation with Post-processing), which consists of a series of processes to detect defects in the center of the bottle. On the other hand, for the annular texture region, they used the WTMF strategy (Wavelet Transform Multiscale Filtering), which separates texture information at different frequencies and then uses multiscale filters to highlight relevant features that can later be classified as defects. The results suggest that this approach may represent a significant advancement in the field of industrial product quality inspection.

Gizaw and Kebebaw (2022) proposed an algorithm capable of detecting defects in bottles during the water bottling process, specifically targeting defects involving the caps (such as improper cap placement, absence of caps, or the presence of foreign objects inside them). To achieve the success, the researchers started by using a vast dataset collected from various sources to locate the bottles with the YOLOv3 algorithm. Following this, they processed the images and proceed to analyze the bottle caps.

In addition to detecting the presence of the bottles, they also located them so that, later on, they could process only a subimage containing the bottle and use it in a convolutional neural network (CNN) model to detect possible cap defects. The authors also compared the use of the YOLO3 with other algorithms and concluded that this is the fastest and has the best overall performance.

Algorithm	Speed	Performance	Time
RetinaNet	Slow	High	3-15'
YOLO	Faster	Moderate	30s-8'
Tiny YOLO	Fast	Low	1-10'

Table 1.1: Comparison of algorithms in the detection of bottles based on the study (Gizaw and Kebebaw, 2022)

The development of bottle detection algorithms is a relatively recent occurrence, driven as a consequence of the complexity involved in detecting the object itself. This type of detection is used in different areas and is increasingly gaining attention from bottle manufacturers, manufacturers of machines to be used among bottle factories (such as Vidromecânica), and researchers in the field of computer vision.

Orachon and Intani (2012) proposed a system that uses a webcam to verify whether a crate of bottles is full or not through a detection process involving the bottle caps. The process begins with converting the image to grayscale, followed by the application of the Canny edge detection algorithm to identify the bottle caps. Finally, the exact location of the caps is determined through template matching with two-dimensional cross-correlation and a defined threshold. This results in a black image with white dots corresponding to the bottles, allowing their counting, and verification of crate filling. If there is a lack of bottles in the crate, it would be rejected through a PLC. With this study, the researchers concluded that, although the system is viable for practical implementation, it has some limitations due to the quality of the camera and inadequate lighting conditions.

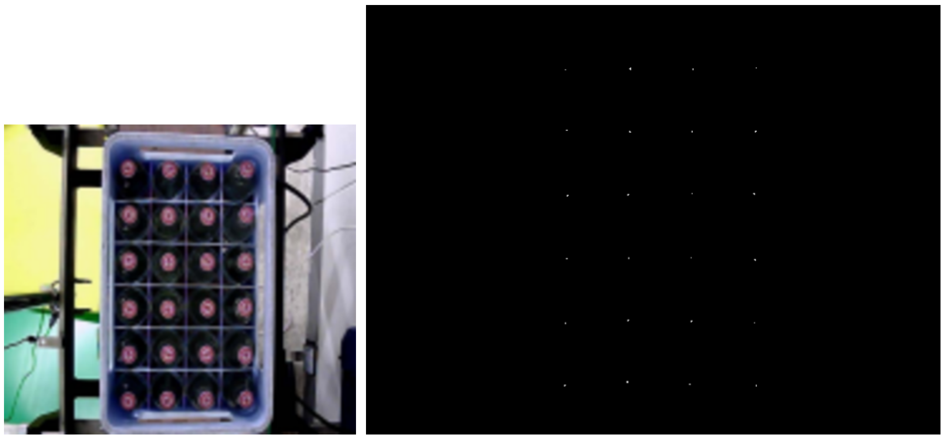


Figure 1.5: Crate image (left). Final image of bottle detection, with white dots identifying the bottles (right) (Orachon and Intani, 2012)

Sanver et al. (2017) conducted a study in which they explored bottle detection methods to ensure the quality of packages consisting of 6 or 20 mineral water bottles, as some could be defective. To do this, they proposed the use of a method that employs the Sobel filter to detect edges, and the Hough transform to identify the circles representing the bottle caps, in order to calculate the number of bottles present in each package. If any of the packages are defective, meaning if they contain a quantity of bottles or circles detected in the image that differs from what is expected (in this case, 6 or 20 bottles per package), they are promptly removed from the production process through a drive motor.

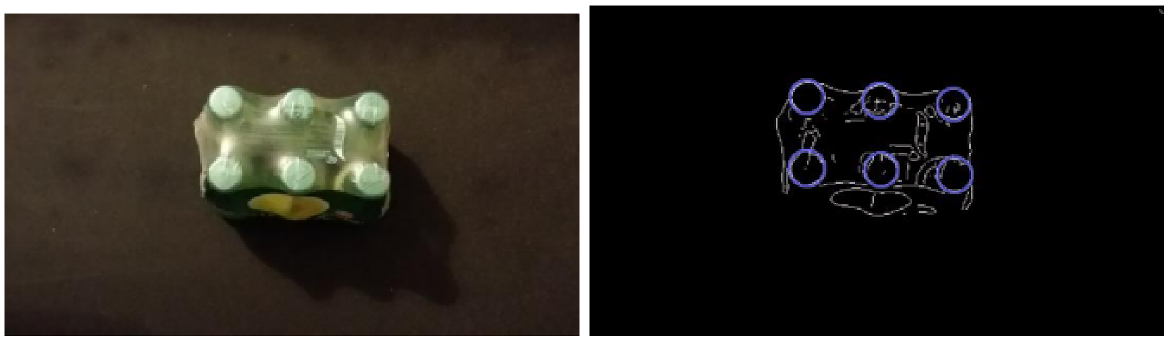


Figure 1.6: Pack image (left). Image when the Hough transform is applied (right), © 2017 IEEE. Reprinted, with permission, from (Sanver et al., 2017).

Wang et al. (2018) proposed a UAV (Unmanned Aerial Vehicles) image dataset called UAV-Bottle Dataset (UAV-BD), consisting of 25,407 images and a reference for object detection. The images were obtained at low altitudes, between 10 and 30 meters, focusing on the location of discarded plastic bottles in the wild. To develop this algorithm, the researchers evaluated several types of object detection algorithms using CNNs such as Faster R-CNN, SSD, YOLOv2, and RRPN, with modifications made to RRPN to use oriented bounding boxes. This approach provided superior results given the unique challenges in detecting bottles in UAV images, such as the small size of the bottles, complex backgrounds, arbitrary orientations, and bottle transparency. The use of oriented bounding boxes effectively addressed these challenges, surpassing conventional methods.

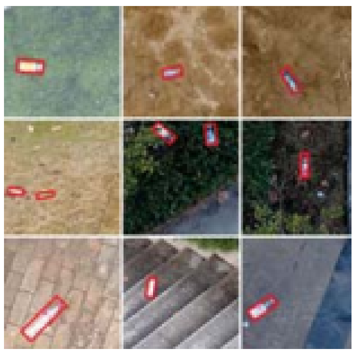


Figure 1.7: UAV images with detection using oriented bounding boxes (Wang et al., 2018)

Akbar et al. (2022) proposed a bottle detection algorithm, using the YOLOv3 algorithm, chosen for its high capacity for real-time object detection, and the COCO dataset. The image capture was performed by using a low-quality mobile phone, aiming to test the algorithm's effectiveness under suboptimal image conditions. The algorithm locates the bottle while also providing information about the values of the bounding box position, the number of bottles identified, and the class of objects being detected. As a result, this study reported an F1 score of 0.88, a precision of 0.96, and a recall of 0.81.

More recently, Din et al. (2022) presented a proposal for a mobile robot composed of a Kobuki base, a Raspberry Pi 3, a stereo camera, a robotic arm, a computer, a GPS, and an IMU (Inertial Measurement Unit), capable of identifying and collecting recyclable plastic bottles. This project is divided into three main processes: navigation control, image processing with computer vision, and the kinematics of the arm for bottle collection. The bottle detection mechanism uses convolutional neural networks (CNNs), specifically R-CNN (Region-based Convolutional Neural Network). The neural network is trained with a dataset that contains diverse images of bottles in different scenarios, allowing it to identify regions of interest in the image likely to contain plastic bottles. Furthermore, a depth map is generated using the stereo camera, enabling the determination of the distance to the bottle and, subsequently, calculating its coordinates. This allows the mobile robot to navigate towards the bottle and execute the collection task effectively. It is worth emphasizing that the system underwent testing in indoor and outdoor environments, both where the robot was able to successfully identify and collect the bottles.

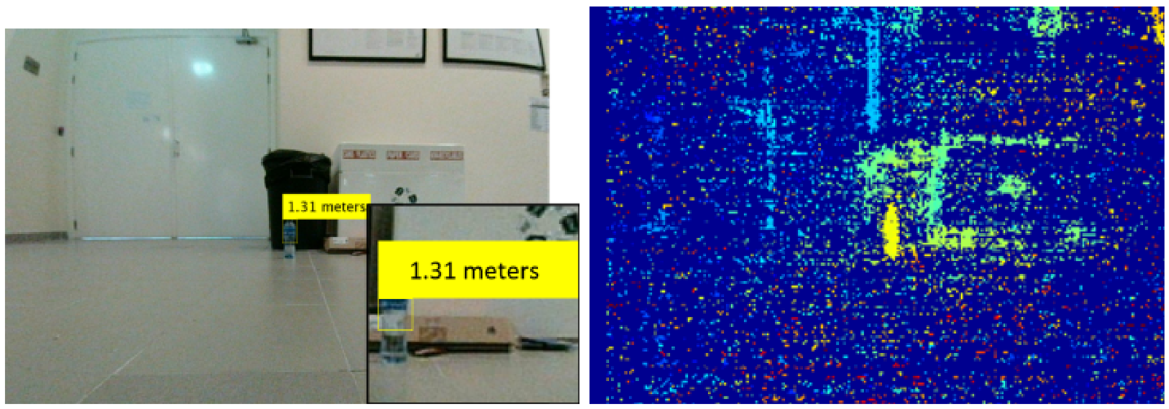


Figure 1.8: Image of the scenario (left). Disparity map with the bottle highlighted in yellow (right), © 2022 IEEE. Reprinted, with permission, from (Din et al., 2022).

Based on the research conducted in the current state of the art, it is possible to conclude that, since the analysis of this research project will be conducted from above the bottles and requires a high degree of precision, the Hough Transform emerges as an effective and appropriate method to be implemented. This is justified by the fact that, since detection is done from a top-down perspective of the bottles, the circles representing these objects are clearly noticeable. Thus, the Hough Transform is an ideal choice due to its remarkable ability to accurately detect circles, allowing and facilitating the subsequent calculation of their centers, which correspond to the centers of the circles. Although alternatives such as the use of convolutional neural networks for bottle detection could also be considered, the consistent arrangement of the bottles and the overhead analysis is done from above favor a simpler approach, such as the Hough

Transform, whereas resorting to CNNs would be a more beneficial option in scenarios where the bottles are positioned variably or in diverse environments, as evidenced in prior studies.

1.3.2. Cold surface treatment machines for bottles

The glass bottle manufacturing process has several stages, from the creation of the bottle itself to the phases of glass decoration, which improve the quality of the final product. First, the bottles are produced using molten glass in molds at high temperatures, followed by the surface decoration phase. This last one is divided into two distinct stages to ensure that the bottles can slip through each other and the glass obtains adequate resistance against scratches. In the first stage, a hot end coating (HEC) is applied - a metallic compound is deposited through chemical vapor deposition (CVD) on the newly created, hot glass bottles, to give the bottles a proper resistance. This process is carried out in a ”coating tunnel” to create a thin layer of metallic oxide. In contrast, the second stage concerns the cold end treatment (CEC) of the glassware which, according to Vidromecânica, consists of a spraying process with low-concentration polyethylene emulsions, creating a thin film on the glass surface that facilitates the articles’ slippage and reduces the likelihood of microcracks. This treatment can be carried out above the conveyor, with spraying from top to bottom in a downward direction. However, in exceptional cases, when the glassware has a wide mouth, the spraying is done from bottom to top, beneath the conveyor, to avoid the product entering the interior. The combination of HEC and CEC provides resistance to scratches and wanted slippage (Vidromecânica (2024), Clapper and Pfahl (2015)).

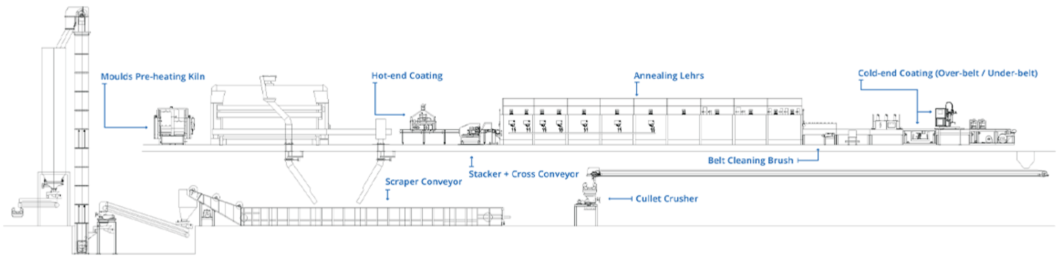


Figure 1.9: Bottle’s production pipeline at Vidromecânica (Vidromecânica, 2024)

Some researchers found that spray coating systems were not used in their country primarily due to high costs. For this reason, they developed a prototype of a system that facilitates spray coating, as well as its control, to maintain an exact temperature and fluid density. To achieve this, they used a PLC to carry out the process, SCADA to monitor the process, and a PID control to manage the temperature and density values. This way, they created an alternative system that is cheaper than the previous ones and can be used in the glass industry. Although it is not as sophisticated as other systems on the market, it also proved to be effective (Shaikat et al., 2019).

Clapper and Pfahl (2015) patented a method for applying a cold coating that includes horizontally arranged spray guns. The process takes place on a single line of bottles, which reach the spray zone through a conveyor belt. Here, the spraying is performed, which may be followed by a second spraying, both carried out by one or more sprayers, and which can be applied to selective areas of the containers, spraying the bottle uniformly.

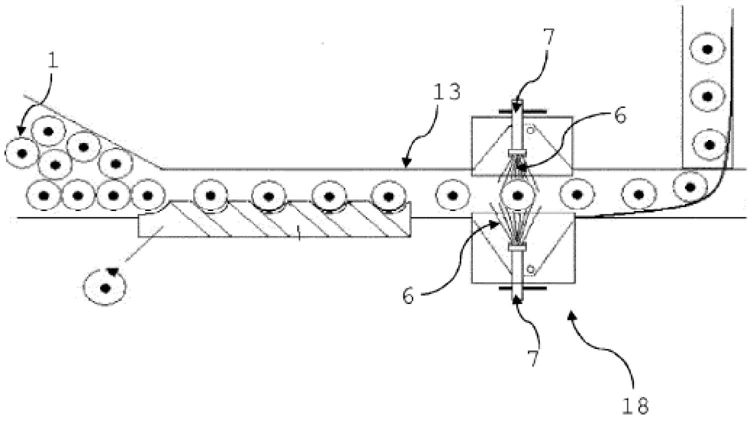


Figure 1.10: Top view image of the patented system (Clapper and Pfahl, 2015)

In the case of the Wagner Group, researchers developed a final coating system, consisting of a water-based paint. This product used atomizers to apply the electrostatic coating on a single production line, where the bottles are sprayed one by one. It is worth noting an interesting detail: the bottles are upside down throughout the process. This system can also be used in cold coating processes (Wagner Group, 2024).

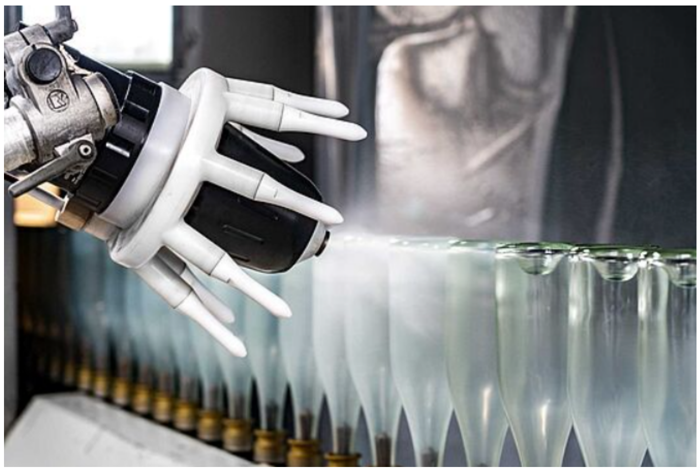


Figure 1.11: Image of the Wagner Group's electrostatic coating system (Wagner Group, 2024)

In addition to everything presented, the company Revimac also offers a cold spraying system, in which a sprayer moves from one side to the other of the plane containing the bottles arranged in horizontal lines, spraying line by line. This system integrates a cart assembly that carries the sprayer and contains spray nozzles, a vertical pneumatic cylinder to control the up and down movement, and a stepper motor used to control the lateral movement of the nozzles (Revimac, 2024).

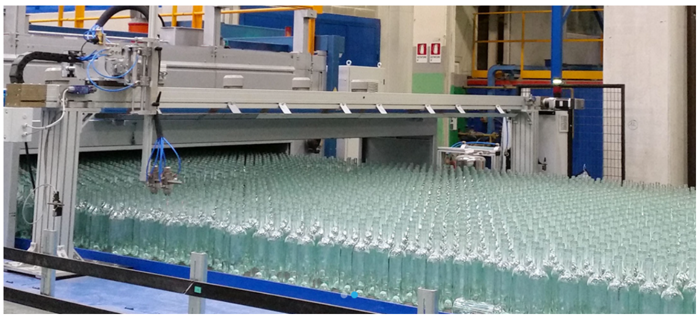


Figure 1.12: Image of Revimac's cold end spraying system (Revimac, 2024)

The current example from Vidromecânica is very similar to the system from Revimac, where the sprayer has two nozzles and, through a cart assembly, proceeds to spray parallel lines of bottles, line by line. The computer vision system proposed in this dissertation aims to work along with the current cold end coating process of Vidromecânica. By comparing it with other products on the market, it was possible to discern that there are two different approaches in these processes: if the sprayer is fixed, the bottles pass through, one by one, and are sprayed; on the other hand, if the sprayer is mobile, several bottles are sprayed in a short time. It can also be concluded that only the latter system truly requires a computer vision system, as in other cases, the sprayers are positioned horizontally instead of vertically, and thus there is no risk of the liquid entering the bottle. The last ones are not used in glass bottles because they can only treat a bottle each time, what can slow down the production.

In conclusion, based on the state-of-the-art research and according to Vidromecânica, it is noticeable that there is no cold bottle coating system on the market that integrates a system capable of detecting the alignment of the bottles when using a mobile sprayer, making the system being developed in this project the first of its kind.

1.4. Dissertation's overview

This document is divided into a few sections, starting with the theoretical basis, with detailed descriptions of the Hough transform and Modbus communication. Next comes the development of the algorithm, how it was integrated into the company's machine and a presentation of the results obtained. Finally are the conclusions with some comments about the future work that can be develop.

2. Theoretical basis

2.1. Hough transform

The Hough Transform, an innovative technique proposed by Paul Hough in 1962, is fundamental in detecting geometric shapes within images. Initially designed to identify lines, its applicability was later expanded to include the detection of circles and ellipses as well. This method operates by converting image elements from the image plane to the Cartesian or Hough space, using specific equations that vary according to the geometric shape. This transformation is facilitated by an accumulator matrix that records "votes" given to each point, reflecting the presence of a specific geometric shape.

One of the greatest strengths of the Hough Transform lies in its robustness, allowing the precise identification of shapes even in adverse conditions such as noise, occlusions, or distortions. This efficacy has turned the Hough Transform into an indispensable tool in contemporary computer vision systems, excelling in a wide range of practical applications.

To improve the detection of shapes before applying the Hough Transform, images typically undergo filtering processes. Filters such as Sobel or Canny are commonly used to detect edges in the image. Then, with the filtered image, the gradient of the image where the edges were detected is calculated. With this pre-processing, the images are ready to go through the Hough Transform, which will be able to identify and map the desired shapes more precisely (Nixon, 2013).

2.1.1. Lines

Starting with the detection of lines, the transformation of points is made according to a Cartesian parameterization through the following equation:

$$y = mx + b \tag{2.1}$$

where points in the image space with coordinates (x, y) are transformed into lines in the Cartesian space (m, b) through the Eq. (2.1), where m represents the slope and b the y-intercept.

In the image space, x and y are variables, and m and b are parameters. In the Cartesian space, the opposite is verified: x and y are parameters, while m and b are variables. When this transformation is carried out, a point in the image space is transformed into a line in the Cartesian space, or a line in the image space can be transformed into a point in the Cartesian space.

Looking at the example in Figure 2.1, four points in the image space are transformed into four straight lines in Cartesian space, characterized by the axes m and b. These four lines are calculated by swapping x and y in the equation while varying m to find several pairs of values (m, b), and in the end, the straight line passing through these points is drawn. In turn, the line in Cartesian space represents all the values of m and b for lines that pass through the point (x, y) in the image space (Nixon, 2013).

For example, when three points (A, B, and C) in the image space can be connected by a straight line, the result in Cartesian space will be three lines intersecting at a certain point (m, b), indicating the slope and y-intercept of the line passing through the three points in the image space, which is therefore the line being drawn, as shown in Figure 2.2. Regarding point D, it is transformed into a line in Cartesian space that does not intersect the other lines at the point (1,0) because it does not belong to the line formed by points A, B, and C. However it does intersect the other lines at different points because point D can be connected to another point in the image through straight lines.

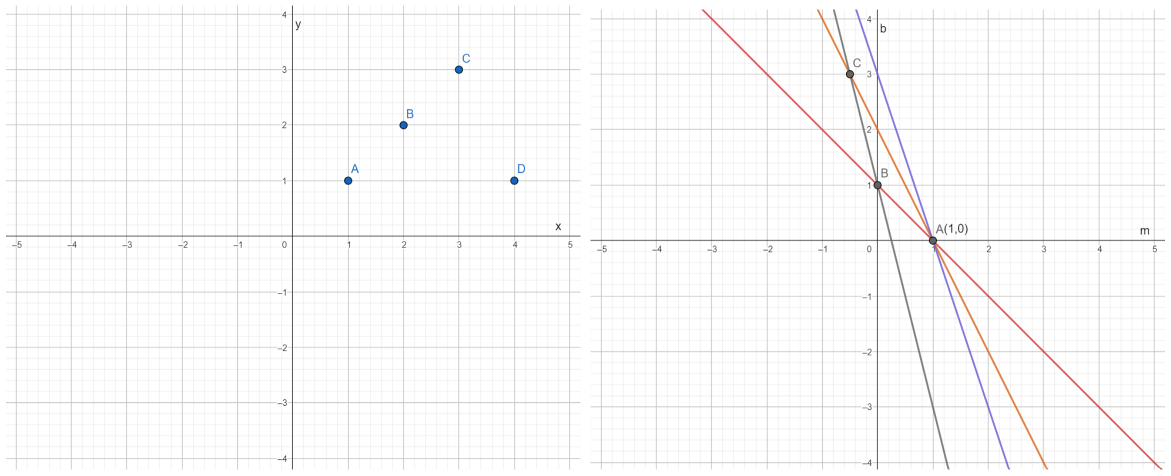


Figure 2.1: Image space with four points (left). Cartesian space with four intersecting lines (right)

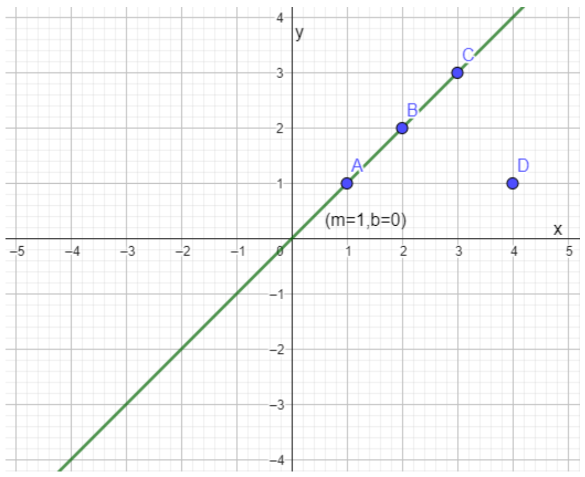


Figure 2.2: Image space representing the straight line that connects points A, B, and C

The complete process of the Hough Transform follows the following algorithm:

```
1. Quantize the parameter space (Cartesian) (m, b);

2. Create the accumulator matrix A(m, b);

3. Set A(m, b) = 0 for all (m, b);

4. For each image edge $(x_i, y_i)$:
        For each element A(m,b):
            If $(m, b)$ lies on the line $b = -mx_i + y_i$;
                $A(m, b) = A(m, b) + 1$

5. Find local maxima in A(m, b) (Araújo, 2023).
```

Looking at the algorithm, it is possible to understand that the transformation from image coordinate space to Cartesian space is performed first. Then, an accumulator matrix is created, where all elements are reset to zero in order to start the process of accumulating votes. This accumulator matrix is characterized by parameters m and b, so for each line transformed into Cartesian space, the m and b values of that line will be used in the matrix to accumulate votes. In other words, all m and b values of the line will accumulate one vote in each slot of the matrix containing these values. After this accumulation of votes is completed, the matrix is ready, and the process of finding local maxima can be used. A threshold is used to detect the important maxima. In the end, there may be one or more detected maxima that indicate m and b points, which, if transformed back to image space, characterize lines.

In cases where Cartesian parameterization is used, the value of m can have an infinite range of values since lines can vary from horizontal to vertical simply by changing it. As votes are placed in a discrete accumulator matrix, there can be issues of polarization. Regarding the parameter b, it can also take an infinite range of values, which would make the accumulation matrix huge, leading to higher memory usage and computational costs. To address this problem, two accumulator arrays can be used in the Hough Transform algorithm: one array for m values between -45° and 45°, where b does not take on high values, and another array for the remaining m values, where b can have higher values. This difference in the algorithm only partially resolves the problem, as it cannot guarantee that b will be small for any m value when it is between -45° and 45° (Forsyth and Ponce, 2012).

In order to completely avoid this type of problem when using Cartesian parameterization, an alternative parameterization called polar parameterization (Duda, 1972) is used, resorting to parameters (θ and ρ) instead of (m, b), according to the following equation:

$$xsin(\theta) - ycos(\theta) = \rho \tag{2.2}$$

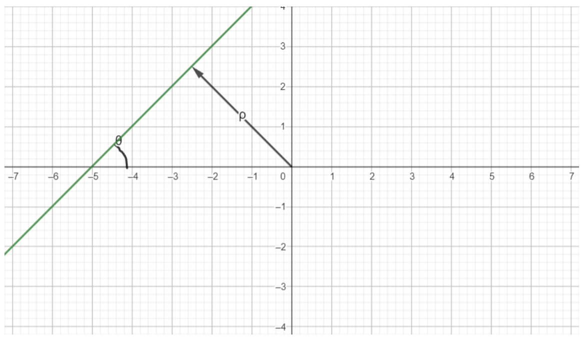


Figure 2.3: Definition of θ and ρ

Here, ρ indicates the perpendicular distance from the origin (point (0,0)) to the line, while θ is the angle between the line and the x-axis. Now, the accumulator space does not have to be enormous because θ and ρ are finite parameters, with θ ranging from 0 to π and ρ ranging from 0 to ρ_{max}, which will depend on the size of the image itself, solving the problem faced with Cartesian parameterization. With this parameterization, the space where the transformation will be carried out is called Hough space, where instead of lines, there are sinusoids (Nixon, 2013).

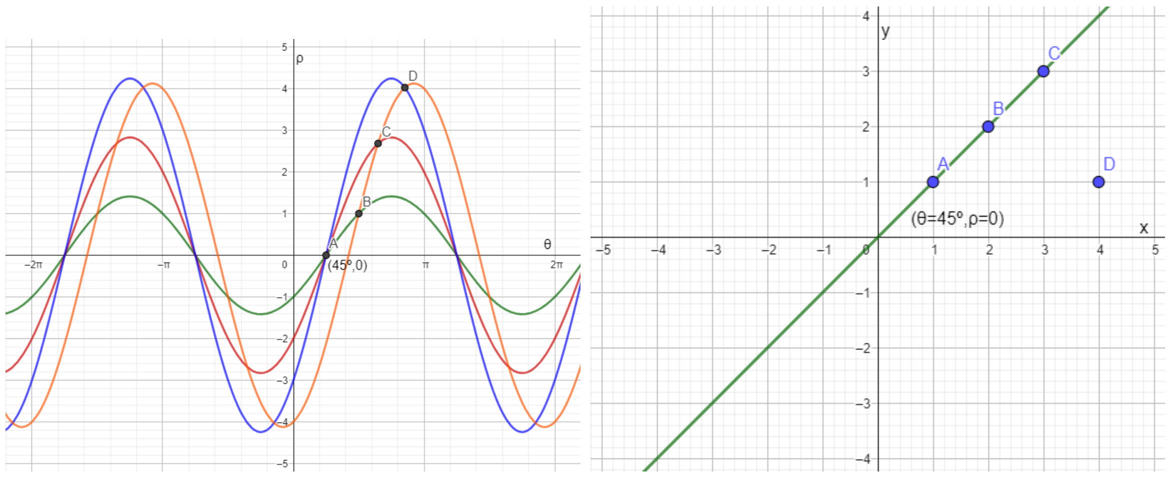
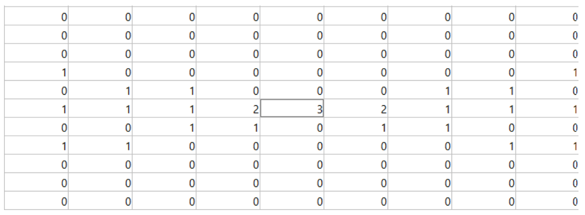


Figure 2.4: Hough space with four intersecting sinusoids (left). Image space representing the straight line that connects points A, B and C using polar parameterization (right)

Figure 2.4 demonstrates an example of a transformation of four points, in which three of them define a line, used in the figure 2.1(left). When transformed into Hough space, give rise to four sinusoids. Three of those intersect at a point A = (45º,0), figure 2.4(left) which defines the line in image space that connects the three points, figure 2.4(right), using the polar parameterization.

The accumulator matrix will have a set of 180 positions for the values of θ, from 0° to 180°, and the values of ρ will be between 0 and $\sqrt{N^2 + M^2}$, with N×M being the size of the image. Then, the points greater than a given threshold will be transformed into sinusoids, using the equation with θ values from 0 to 180, and calculating the value of ρ that must be within its limits, thereby discovering the pairs of values that constitute the sinusoid, which will be incremented in the accumulator matrix (Nixon, 2013).



0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1
0	1	1	0	0	0	1	1	0
1	1	1	2	3	2	1	1	1
0	0	1	1	0	1	1	0	0
1	1	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Figure 2.5: Part of the accumulator matrix from the previous case

Figure 2.5 represents a part of the accumulator matrix, showing the maximum point with a vote value of 3 in the center of the image.

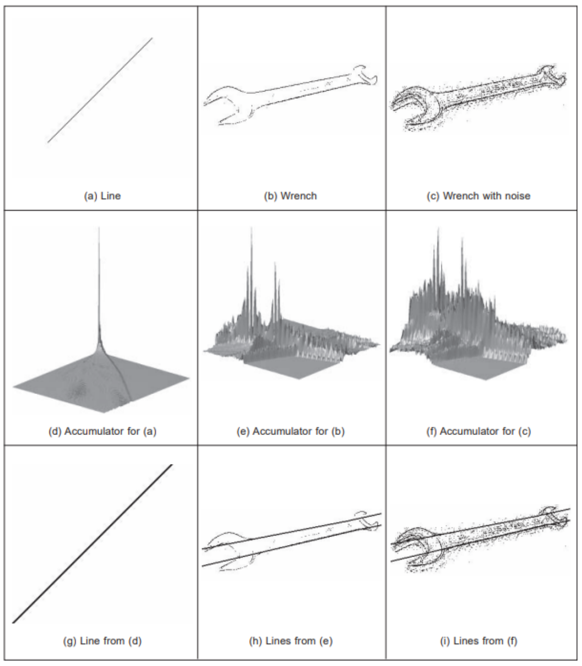


Figure 2.6: Example of the Hough Transform in three different images according to the equation $y = mx + b$, figure 5.7 on page 177 of the book "Feature Extraction and Image Processing" (Nixon, 2013)

In Figure 2.6, there are three examples of images with a line (a), a wrench (b), and a wrench with noise (c), along with the representations of their accumulator matrices (d, e, f) and the lines defined by the Hough Transform (g, h, i) when Cartesian parameterization is used.

Regarding the accumulator matrix (d), there is only one peak indicating the votes at the point (m, b) that defines the line (a); on the other hand in matrix (e), there are two peaks defining the two lines; finally in matrix (f), it is possible to observe two peaks, but there is more noise when compared to matrix (e) due to the extra number of points in the noisy image of the wrench. Since the two peaks in matrix (f) are in the same location as the peaks in matrix (e), this shows that the Hough Transform can tolerate noise, which is confirmed in image (i) where the two existing lines in the image are drawn.

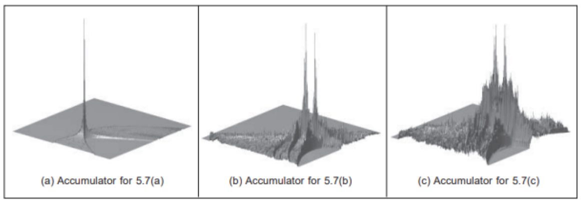

(a) Accumulator for 5.7(a)
(b) Accumulator for 5.7(b)
(c) Accumulator for 5.7(c)

Figure 2.7: Example of the accumulator matrices of the three previous images(a,b,c) according to the equation $xsin(\theta) - ycos(\theta) = \rho$, figure 5.9 on page 180 of the book "Feature Extraction and Image Processing" (Nixon, 2013)

In Figure 2.7, image (a) concerning the image of the line shows a single, well-defined peak; in images (b) and (c), accumulator matrices about the wrench and the wrench with noise, the peaks are more distinct when compared to the Cartesian parameterization. This concludes that polar parameterization is more practical and yields better results than the version using Cartesian parameterization.

2.1.2. Circles

To use the Hough Transform for detecting circles instead of lines, a simple modification is required. Previously, it was used the parameterization: $xsin(\theta) - ycos(\theta) = \rho$. Now, it will be used an equation that defines circles for parameterization, which is the equation (Nixon, 2013):

$$(x - a)^2 + (y - b)^2 = r^2 \tag{2.3}$$

Observing the equation, it is clear that a point will be transformed into a circle because the parameterization uses the equation of a circle.

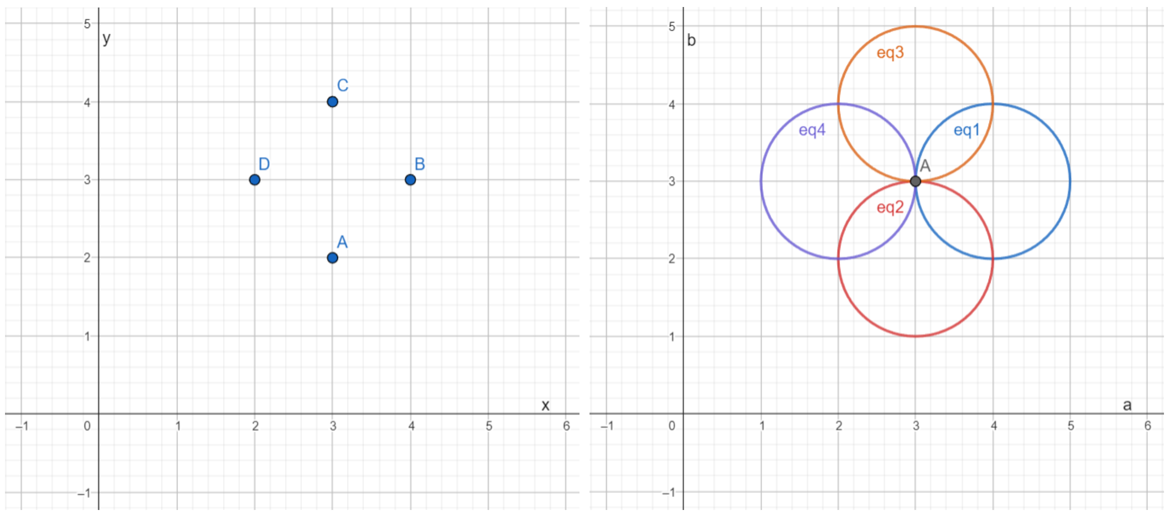


Figure 2.8: Image space with four points (left). Hough space with four intersecting circles (right)

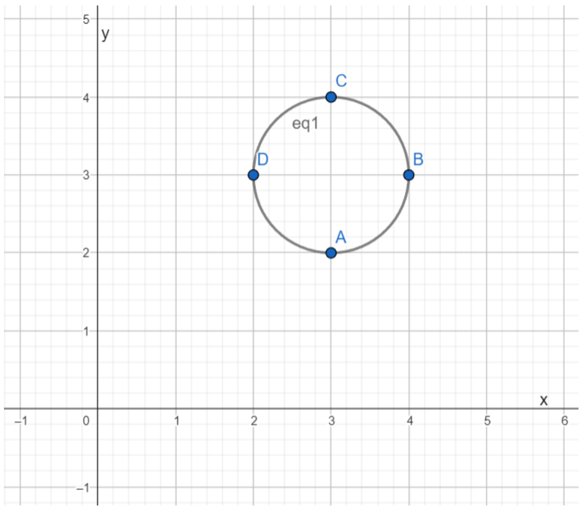


Figure 2.9: Image space with the four points and the detected circle

In this case, four points that can be joined by a circle are transformed into four circles with radius 1 that intersect at point A, which is the center of the circle that connects the four points in the first image, Figures 2.8 and 2.9.

Proceeding with the transformation from image space to parameter space, there are x and y as parameters and a, b, and r as variables. It can be seen now that there are three variables instead of two, as in the case of line detection. Here, there are two possibilities: if the radius is known, then there will only exist a and b as variables, and the accumulator matrix will be 2D as in Figure 2.8. But if the radius is unknown, then the accumulator matrix will have three dimensions (Nixon, 2013).

If the radius is unknown, a point in the image space will define a set of circles in the parameter space. These circles are all centered on the point captured in the image space and have different radii. This implies a 3D accumulator matrix where the points captured from the image space correspond to a cone of votes in the parameter space, i.e., a cone of possible circles with different radii, as shown in Figure 2.10. Once the matrix is complete, the peak values will indicate the center and radius of the detected circles in the image. This is possible because the peaks will be where the circles with the correct radius intersect, voting heavily at that point and radius. Compared to the Hough Transform for lines, the only difference in this case is that the votes are cast in cones in 3D accumulator matrices, and if the radius is known a priori, then the procedure is similar to the method for lines.

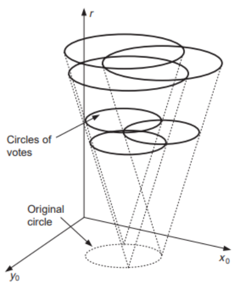


Figure 2.10: Hough space with a cone indicating different possibilities of radii, figure 5.10 on page 181 of the book ”Feature Extraction and Image Processing” (Nixon, 2013)

There is also a technique that requires a bit more work in the image space to extract more information. In this case, the information includes the location of the points, the value of r (parameters used previously), and a new value: the direction of the point (ϕ). Looking at Figure 2.11, it is possible to observe four points with known locations and directions, indicated by the vectors (e, u, v, w). In this way, the Hough Transform does not vote for a circle of points but rather for a single point with the direction established by (ϕ) and at a distance r from the point's location, as shown in Figure 2.11(right), where the vectors (a, b, c, d) all have the same length r and point towards point E. To find the point in the matrix where the vote will be incremented, the following equations are used:

$$a = x + rcos(\phi) \tag{2.4}$$
$$b = y + rsin(\phi) \tag{2.5}$$

Therefore, there is more work in the image space, while the work and computational effort to fill the accumulator matrix is considerably less in Cartesian space (Nixon, 2013).

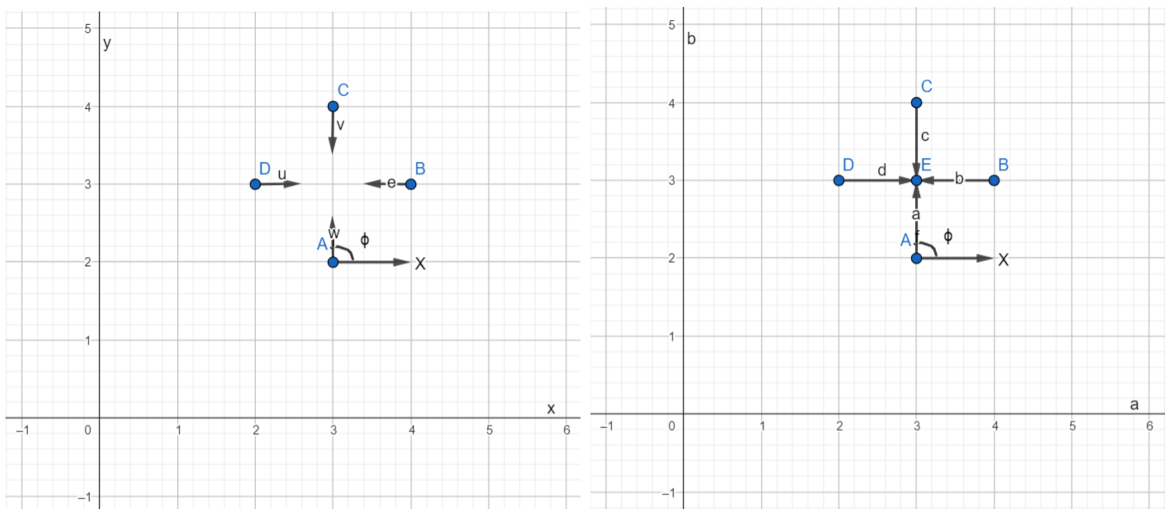


Figure 2.11: Four points with known direction (left). Four points voting in the center of the circle (right)

The Figure 2.12 illustrates the application of the Hough Transform for circles in three different images: one containing a synthetic circle (a), another containing a soccer ball (b), and the last containing the same soccer ball but with a higher level of noise (c). Then, the representations of the votes in the accumulator matrix (d, e, f) are shown, followed by the circle detected by the method (g, h, i). Regarding the first image, there is a peak in the vote representation (d), and there are some votes far from it that rise towards the center, but the peak is much higher than these votes. As for the other two images (e, f), the central peak remains much higher than the rest of the votes, even with more noise as in image (c). It is noted that the detections are well-executed because in (g, h, i), the traced circles are correct and there is also a great ability to tolerate occlusions and noise, similar to the detection of lines. Moreover, there are no parameterization problems because the circles are closed lines, so there are no issues with the dimensions of the parameters.

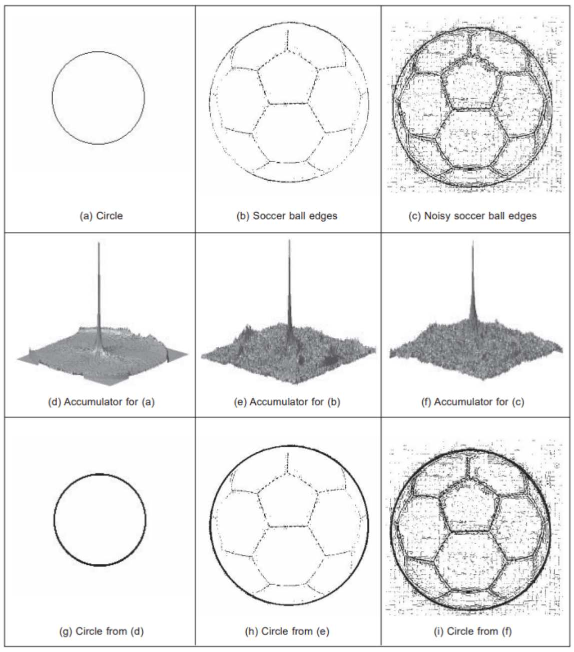


Figure 2.12: Example of Hough Transform applied to three different images for circle detection, figure 5.11 on page 183 of the book ”Feature Extraction and Image Processing” (Nixon, 2013)

2.2. Modbus communication

Modbus is a serial communication protocol commonly used between electronic devices, originally published by Modicon in 1979. This method is used to transmit information over serial lines between the client and the server, replacing the previous Master-Slave connection. In this protocol, clients are the devices requesting information, while servers are the devices providing that information. Additionally, the clients can write information to the server. This method is also referred to as a request-response protocol, where information is requested, and the response contains the requested information if available. Typically, the server can be a SCADA or HMI, while the clients are sensors, PLCs, or PACs.

This method is an open protocol, meaning it is free for manufacturers to use without paying royalties. Consequently, it is widely adopted by different manufacturers across various sectors to connect their industrial devices. This communication is often used to control systems where sensors acquire information, allowing the central system to act according to the data provided.

The Modbus protocol is divided into three existing versions: Modbus ASCII, Modbus RTU, and Modbus TCP/IP:

- Modbus ASCII: the first version created, where all clients respond only to a master, and messages can be up to 252 bytes long. It supports up to 247 addresses, and is based on serial communication.

- Modbus RTU: very similar to Modbus ASCII, but instead of encoding the message using characters, the RTU version uses bytes, increasing transmission capacity.
- Modbus TCP/IP: the last version created. It is based on Ethernet, allowing it to be used over long distances with higher transmission speeds when compared to other methods. Consequently, it is widely used by manufacturers today who need a communication network capable of transmitting over greater distances.

The transmission of information in Modbus uses databases or ranges of registers to store information that can be accessed later. These registers are divided into four different types: two store discrete boolean values (Coils and Input Contacts), while the other two store numeric values (Holding Registers and Input Registers). Two of these registers are read-only, while the remaining two are read-write. Each type of register has 9999 available registers, while the coils and contacts are 1-bit and the registers are 16-bit or 2-byte (Freitas, 2014).

Coil/Register Numbers	Data Addresses	Type	Table Name
1-9999	0000 to 270E	Read-Write	Discrete Output Coils
10001-19999	0000 to 270E	Read-Only	Discrete Input Contacts
30001-39999	0000 to 270E	Read-Only	Analog Input Registers
40001-49999	0000 to 270E	Read-Write	Analog Output Holding Registers

Table 2.1: Table about the registers (Schneider Electric, 2024)

These register numbers are considered locations of information. For instance, the first holding register, number 40001, has the address 0000.

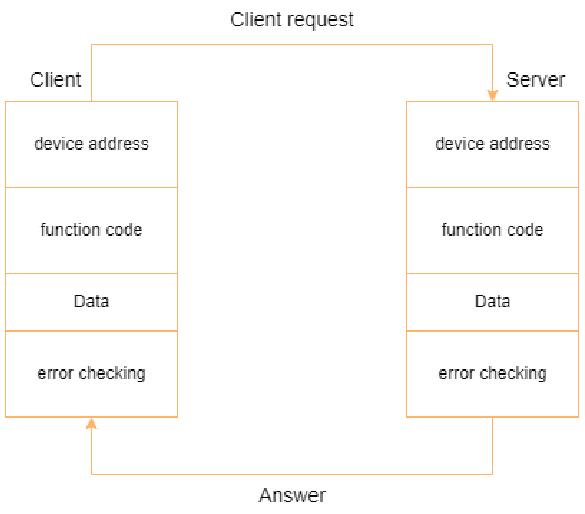
Messages transmitted through Modbus consist of a header and a payload. The header starts with the device address, which can range from 1 to 247, so the first byte is the server address. This allows the correct server to recognize that the message is intended for it, based on the address in the first byte. The second byte is the function code, indicating which type of registers one wants to access and whether the operation is for reading or writing (Freitas, 2014).

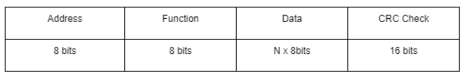
Function Code	Action	Table Name
01(01 hex)	Read	Discrete Output Coils
05(05 hex)	Write single	Discrete Output Coil
15(0F hex)	Write multiple	Discrete Output Coils
02(02 hex)	Read	Discrete input Contacts
04(04 hex)	Read	Analog Input Registers
03(03 hex)	Read	Analog Output Holding Registers
06(06 hex)	Write single	Analog Output Holding Register
16(10 hex)	Write multiple	Analog Output Holding Registers

Table 2.2: Table about the functions codes (Schneider Electric, 2024)

After the header, the payload contains either the data to be written, or the data received.

Finally, a CRC (cyclic redundancy check) is added, comprising two bytes used to detect errors. It consists of a calculated code based on the information in the message, computed by the sender and then by the recipient. If even a single bit of the message is received incorrectly, the CRCs will differ, indicating an error (Freitas, 2014).





Address	Function	Data	CRC Check
8 bits	8 bits	N x 8bits	16 bits

Figure 2.13: Messages structure (left). Length of the messages structure (right)

3. Algorithm Development

The proposed solution is a computer vision system capable of detecting several rows of bottles and identifying possible misalignments. This information is then transmitted to the PLC controlling the sprayer. In this way, when the sprayer passes through an area with misaligned bottles, it needs to adjust its height to avoid knocking them over and decide either to stop spraying or continue, depending on customer preferences and the intended contents of the bottles. As a result, some bottles will not be treated by the sprayer, but as they move along the production line and come into contact with treated bottles, they will receive the necessary treatment through touch.

The main aim of the algorithm is to detect the circles on the bottles and then, with some processing, determine if there are any bottles out of position, i.e. misaligned. To achieve this, it starts by using libraries such as ”opencv”, to use functions to detect the circles, and ”numpy”, for numerical computation.

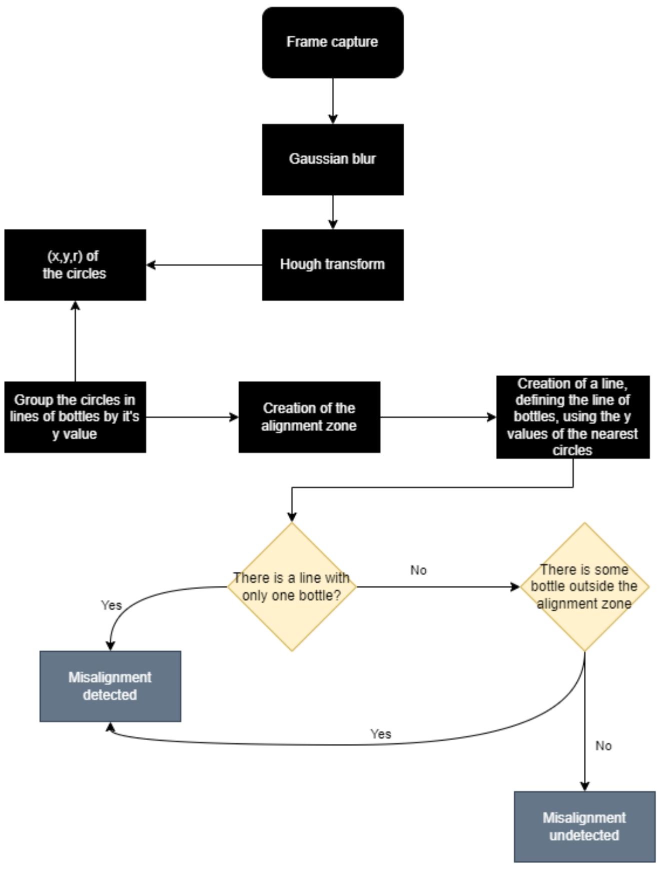

Frame capture
Gaussian blur
Hough transform
(x,y,r) of the circles
Group the circles in lines of bottles by it's y value
Creation of the alignment zone
Creation of a line, defining the line of bottles, using the y values of the nearest circles
There is a line with only one bottle?
No
There is some bottle outside the alignment zone
Yes
Misalignment detected
Yes
No
Misalignment undetected

Figure 3.1: Flowchart of the algorithm

3.1. Hough transform

The Hough transform to detect circles is implemented in a function in the ”opencv” library, with the following identifier:

cv.HoughCircles (image, circles, method, dp, minDist, param1, param2, minRadius, maxRadius)

To apply the method, the image is initially loaded and pre-processed. This involves converting it from RGB to grayscale format, if necessary, and applying a gaussian blur in order to reduce noise in the image, avoid detecting false circles and improve the algorithm’s capacity. Then, the cv.HoughCircles function is involved, to perform circle detection (cf. annex B).

There are two methods available, which have been created by opencv so as not to deal with the 3D parameter space explained in the section 2.1.2 (OpenCV, nd).

3.1.1. HOUGH_GRADIENT vs HOUGH_GRADIENT_ALT

The HOUGH_GRADIENT method uses the gradient information as discussed in section 2.1.2. Instead of voting on all the cells of the circle transformed according to different radii in a 3D accumulator, it focuses on voting on the cells in the direction of the gradient of each edge pixel, detecting the centers and then creating a 1D accumulator to find out the radius. This not only increases the processing speed but also, more importantly, eliminates the use of a 3D accumulator, which could lead to a lot of noise and yield unstable results (Bradski and Kaehler, 2008).

Firstly, the Canny edge detector is used to calculate the edge image, and the Sobel operator (derivatives in x and y) to calculate the gradient of each edge pixel. Then, the voting method is only done in the two directions of the gradient, selecting cells that are local maxima above a specified threshold.

With the candidate centers calculated, the best radius needs to be calculated, and now only a 1D accumulator is needed. For each candidate center, its distance to all edge pixels is calculated, and the radius values in the accumulator are incremented. The radius value that receives the highest number of votes will indicate the correct radius for that particular candidate center. Additionally, the computation window can be reduced by choosing the minimum and maximum radius constraints within the function parameters. This procedure is done for each candidate center, resulting in the identification of circle centers and their respective radii within the image.

The main distinction between this method and the one specified above, in section 2.1.2 lies in whether or not the radius is known, as in this case it is still necessary to calculate the radius (Kang and Atul, 2020).

The HOUGH_GRADIENT method is suitable in cases where the environment can be controlled and where there is good contrast between the circles and the background. The HOUGH_GRADIENT_ALT method is specifically designed to be used in situations where the previous method might fail, and can deal with uncontrolled environments without good lighting and noise. It also incorporates detection based on local intensities, without relying exclusively on the Canny algorithm. However, it operates slower and is only advisable when the standard method does not obtain satisfactory results (Bradski and Kaehler, 2008).

3.1.2. Choice of parameters

To use the ”cv.HoughCircles” function effectively, it is crucial to carefully select the proper parameters. In the case of this project, the image is the input image that will be captured by the camera that already transmits in grayscale. The method is HOUGH_GRADIENT, since it is the one recommended by books and the library opencv, because it is faster (essential in this project because it deals with several frames per second) and has shown promising initial results. Setting the dp to 0.5 ensures a higher-resolution accumulator, enhancing detection accuracy (Bradski and Kaehler, 2008).

The other parameters are the values that need to be tested in order to find the ideal values for the type of image being worked on. In this case, minDist can be a fixed value, since the bottles will be at a fixed distance (when they are next to each other). Param1 and param2 are values that can limit the function, either by making the algorithm hallucinate non-existent circles or by not detecting some real circles. Finally, minRadius and maxRadius help the algorithm computationally, as they constrain the search area for the ideal radius. All these values must undergo rigorous testing to ensure the function accurately detects all the desired circles and only these, without hallucinations.

3.1.3. Setbacks of the Hough transform

This latest version of the Hough transform to detect circles in the ”cv.Hough.circles” function offers numerous advantages, but also presents some setbacks in its implementation.

When the Sobel derivatives are used to calculate the gradients of the edge pixels, it can generate some noise in the result. Also, when the edge image is calculated, all pixels with non-zero energy can potentially become candidates for circle centers, depending on the threshold set, and other factors such as the minRadius and maxRadius parameter. This broad search area can lead to longer running times for the algorithm. Finally, there is the possibility of concentric circles (centered on the same point), which would mean that only one of the circles would be detected by the algorithm. However, in this case, that would not pose a significant problem, as one circle is detected per bottle (Bradski and Kaehler, 2008).

None of these setbacks are relevant to the detection carried out in this work, as it is not necessary to detect concentric circles. The parameters are intensively tested to obtain ideal values and the noise created by Sobel’s derivatives does not become substantial in this method, as we can see in figure 3.2, where the circles are all detected by the opencv function, with the HOUGH_GRADIENT method and the optimal parameters.

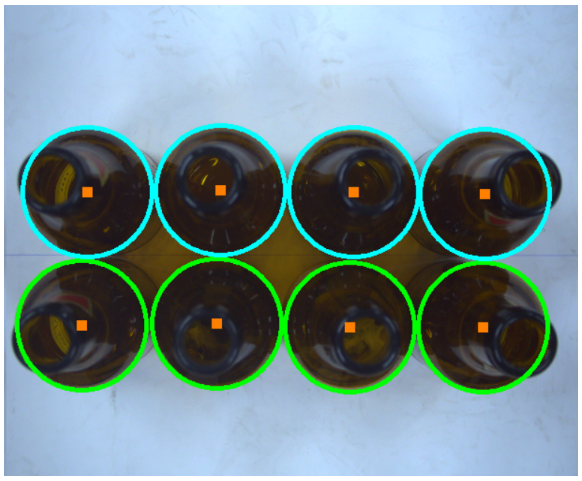


Figure 3.2: Circle detection using the cv.HoughCircles function

3.2. Alignment and misalignment detection

Once the circles have been detected, further processing is required to determine if the bottles are all aligned in rows, or if any are misaligned. To achieve this, two zones and a line are created. The bottles will be grouped into rows according to their y-value (x and y values are defined as the center of the circles, the orange squares in figure 3.2), meaning bottles with y-values within the same range of y-values will be placed in the same row. This range is determined using a threshold.

After the lines have been formed, misalignments are detected. If a bottle is alone in a line, then it means that it is misaligned, while in a line with more than two bottles an alignment sub-zone is calculated, meaning that if there is any bottle in a line outside this alignment zone then that bottle is considered misaligned from the rest. Subsequently, a line is also created of the most relevant y-averages of the bottles in a line, so that it can be seen in the frames how misaligned they are. Finally, these zones and lines are plotted on the image and it is calculated whether there is any misalignment, which is indicated by a Boolean variable.

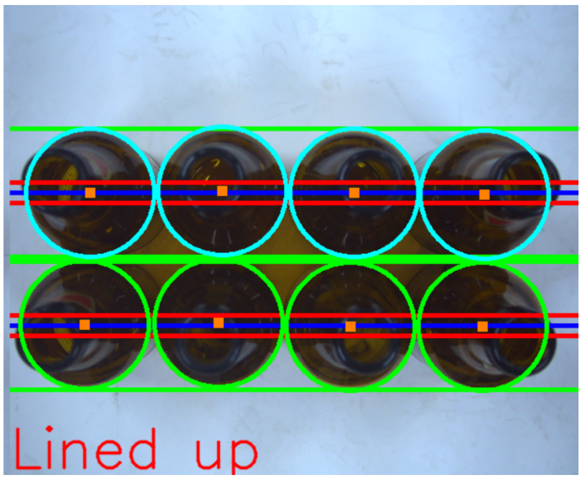


Figure 3.3: Detection of circles and misalignment of bottles

The final result is depicted in figure 3.3. The blue lines indicate the center of the bottle lines, while the red lines define the alignment zone. It is noticeable that all the bottle centers (represented by orange squares) lie within this alignment zone, and the green lines define the region of a bottle line that may or may not be aligned. Note that these zones provide total coverage of the area, since the green lines are consistently adjacent in the middle of the bottle rows, ensuring no space is left unanalyzed between them. The green circles belong to one line and the blue circles belong to another. The term ”lined up” indicates that there is no misalignment ensuring that, in this frame, the sprayer can move through without any issues, therefore not being necessary for it to go up and stop spraying.

There are two variables influencing the size of the two zones. Therefore, for different detection cases, such as the camera being further away from the bottles, these values will have to be different, something that also has to be tested until they are ideal. The definition of the alignment zone (marked by red lines) is particularly important and requires special attention, as it is this zone that determines whether the bottles are aligned correctly or not. It must undergo rigorous testing in an industrial environment and as requested by the client, ensuring it is robust enough to identify possible misalignments that the sprayer would hit.

3.3. Testing Datasets

Once the algorithm was complete, two datasets were emplyed - one containing images and the other with videos. Both datasets encompasses situations with one or multiple lines of bottles, which may or may not be aligned, and are situated at different distances from the camera. This lead to new values for the parameters of the Hough transform function, the variables that control the alignment zones, and the line definition variables. Moreover, the datasets include bottles of different colors such as brown, green, and transparent, reflecting diverse batches produced in the glass industry, as illustrated in figure 3.4. With the help of these datasets, it was possible to test the algorithm for different situations, concluding it performed very well. It is only needed to change the parameters when using frames at different distances from the camera.

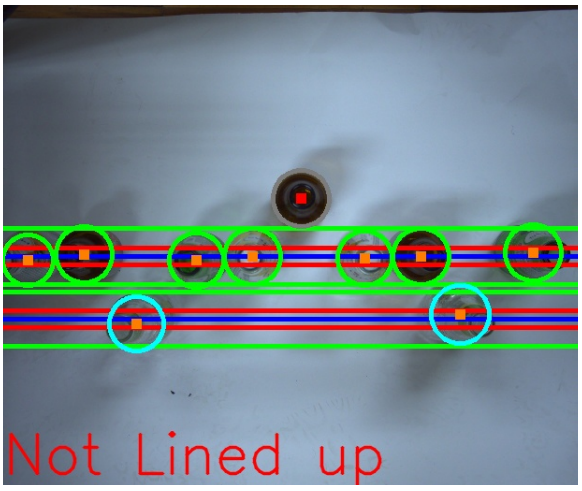


Figure 3.4: Detection of circles and misalignment of bottles with different colors and shapes

Focusing exclusively on the video dataset, it provided an evaluation of the algorithm in videos, frames one after the other, as happens in real time transmitted by a camera. It is possible to state this real-time simulation achieved a very robust performance, since the videos show that some bottles are being placed out of position, and as soon as this happens, the algorithm corresponds effectively.

In figure 3.4 a good analysis of the frame is depicted, showcasing bottles of different colors (brown, green and transparent). The image presents two lines of bottles circled in green and another in blue, with the uppermost bottle being visably misaligned from both previous lines and therefore circled in grey. This demonstrates the algorithm's great performance, even in situations where the bottles are different colors and shapes, and where the camera is further away from the scene. Some adjustments in the Hough transform parameters enables the algorithm to adapt effectively to different spatial configurations captured by the camera.

3.4. Camera configuration

Real-time frames of the industrial belt, featuring various bottle lines, are captured using a camera equipped with an Ethernet interface (i.e. the data is transmitted via Ethernet). To do this, both the computer and the camera must be connected to the factory's Ethernet network via Ethernet cables. The camera operates without an external power supply - the connection the camera has is the Ethernet cable, and it is through this cable that it will receive power (POE - power over ethernet), meaning that the camera will transmit the data of the frames it is capturing to the switch to which it is connected, and consequently to the Ethernet network.

Figure 3.5: Camera image from 3 different views (IDS GmbH, 2024a)

This camera already transmits images in grayscale, eliminating the need to convert the images from RGB to grayscale, as tipically required in the image and video datasets, since the use of the Hough transform requires grayscale images as input. The camera has some important specifications such as a resolution of 1600 x 1220 pixels, a frame rate of 35 frames per second in freerun mode. This high frame rate is particularly important for this project, as it is necessary to capture several frames within a second, as the camera goes from one side of the carpet to the other in a short time. Then a frame rate of 35 is ideal for capturing frames that can catch all the bottles in the rows of bottles to be analyzed (cf. annex A) (IDS GmbH, 2021).

3.4.1. Camera and python code connection

In order to establish a connection between the camera and the algorithm (written in Python code), a connection between these two elements is needed. Unlike some cameras that are able to transmit their data via a specific IP on the Ethernet network where it is located, this IDS camera does not work that way, necessitating the use of software and sdk (software development kit) provided by the company to reproduce this same connection.

3.4.2. Software development kit

The company provides a software development kit available on its website by accessing the support and downloads tab. Users can select the specific camera model they are working on, and proceed to download the necessary software to work with the camera. This includes documentation on the camera, software manuals, and the sdk in Python with the essential library. The sdk makes use of the ”pyueye” library, which is also crucial for this project, since it gives values to certain camera parameters, initializes and checks some functionalities. Once everything is checked and initialized properly, then accesses the camera data and broadcasts it on livestream.

The procedure to initialize and verify functionalities begins by initializing the drivers and establishing connection with the camera, setting the color mode to mono, meaning grayscale. It allocates memory for an image, making it active memory, turns on live video mode, and activates queue mode for existing image memory sequences. This procedure is used to establish the correct connection with the camera, where later it acquires image data through the allocated memory, updating the images allocated here based on what the camera transmits.

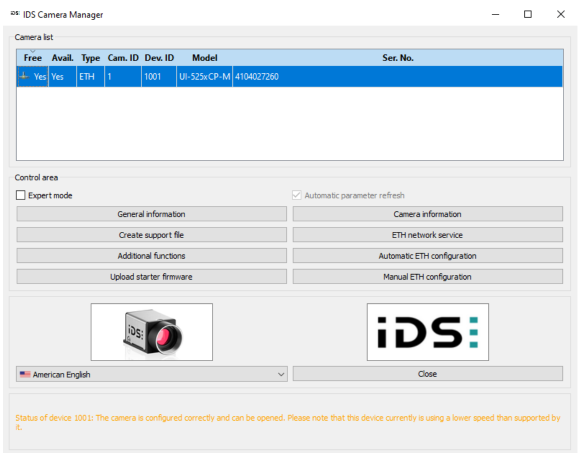
Within the commands that the library has, there are some specific to perform system checks, like those mentioned earlier when initializing and checking some features, but there are also commands free to adjust some parameters within the video captured by the camera. These commands include as the one that sets the color mode, and other ones to change other parameters such as the frame rate. This is also fundamental to this project, as the standard frame rate is a third of the maximum, and for this type of application a higher frame rate is required, close to 35 FPS.

3.4.3. Software

In terms of the software provided by IDS, there are two applications available: one to choose the appropriate network to read the data and the camera wanted to access from among others that may be transmitting to this network, and another application to open the camera’s transmission and change some streaming parameters if necessary.

IDS Camera Manager

This is one of the pieces of software needed to configure both the camera and the Ethernet connection. In this application there is a list of cameras that are connected to the network defined. Firstly, it accesses the network service where can be found the different networks available that the computer can access. Usually this involves Ethernet, Wi-Fi, USB ports, among others. In this case, the Ethernet network card is activated and all the others deactivated so that there is no interference between them. Once the network has been selected, all the cameras connected to that Ethernet card appear in the camera list. The configuration can be set automatically or manually. In this case it must be manually, as it needs a fixed IP address and not a range of IP addresses with a subnet IP address that is the same as the network it is connected to. Once completing these two steps, the camera should appear as available so that clicking on it opens the next application. To keep everything organized within the factory's Ethernet network, both the computer and the camera have specific IPs provided by the company, so as not to interfere with the IPs of other devices (IDS GmbH, 2024b).


IDS Camera Manager

Camera list

Free	Avail.	Type	Cam. ID	Dev. ID	Model	Ser. No.
Yes	Yes	ETH	1	1001	UI-525xCP-M	4104027260

Control area

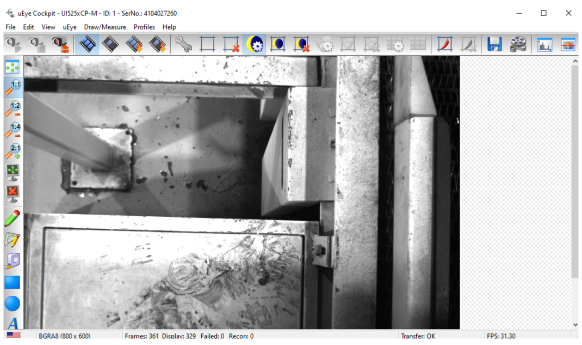
☐ Expert mode ☑ Automatic parameter refresh

Status of device 1001: The camera is configured correctly and can be opened. Please note that this device currently is using a lower speed than supported by it.

Figure 3.6: IDS Camera Manager software environment

uEye Cockpit

The following application demonstrates the functionality and performance of uEye cameras. In the uEye Cockpit it is possible to access the settings and functions of the uEye library, configure various parameters, see how they influence the image and save images or videos to a file on the computer. In the bottom corner of the image it provides some information such as the color mode, resolution and current frame rate. This software does not link directly to the algorithm, but there is a test area where users can change parameters such as the frame rate and see how they influence the video transmitted by the camera. It complements IDS Camera Manager because a camera is opened in this software, uEye Cockpit is launched to display the video or image stream transmitted by the camera at that moment. Typically, when the camera is available and operational in uEye Cockpit, then it will also work properly within the algorithm. However, it is important to note that it can only be connected to one of them, i.e. it will only be available to one of them, preventing conflicts between them (IDS GmbH, 2024b).


uEye Cockpit - UI525xCP-M - ID: 1 - SerNo.: 4104027260

BGRA8 (800 x 600) Frames: 361 Display: 329 Failed: 0 Recon: 0 Transfer: OK FPS: 31.30

Figure 3.7: uEye Cockpit software environment

3.5. Live tests with the camera

After implementing the sdk code in the algorithm and configuring the camera and the connection, performance tests were conducted to assess both the camera's quality and the algorithm's functionality with live video images. These tests took place in the ISR laboratory under controlled conditions, using a tripod setup to mimic the camera's position as it would be in a factory setting, and ensuring it could capture the bottles from above, in such a way that the algorithm could detect the circles in their structure. As an example, figure 3.8 presents a frame captured during one of these tests.

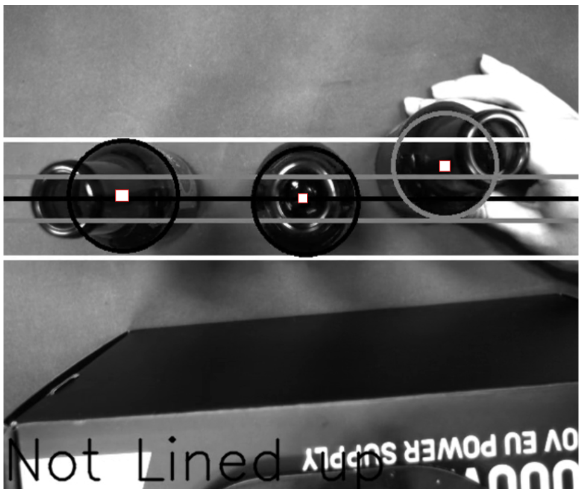


Figure 3.8: Frame from a test video using the camera's transmission

By analyzing the frame taken from one of the tests, which initially comprised three aligned bottles that were subsequently misaligned one by one manually, in order to check if the algorithm completed these misalignments, it is possible to observe how it works. Considering this specific frame, it is in grayscale because it was taken from the camera feed, where the bottles on the left and in the middle are aligned, because their centers are within the alignment zone, marked by gray lines. However, the bottle on the right has its center slightly above the alignment zone, leading the algorithm to conclude that there is misalignment, indicated by the phrase ”Not Lined up” displayed at the bottom of the frame. Throughout the rest of the test videos, the algorithm was able to accurately and promptly detect misalignments similar to this. Only some minor disturbance were observed in the circles drawn from frame to frame, which is normal as it is a set of consecutive frames and it was using a low frame rate.

4. Integration of the solution

The algorithm developed, aimed at analyzing frames, detecting the bottles contained in these frames, and identifying any misalignment between the various rows of bottles, is intended to be integrated into Vidromecânica's cold end coating machine. This machine comprises a conveyor belt that transports the bottles, and a carriage system that moves the sprayers across both sides of the belt, spraying the exterior of the bottles as it passes between the rows. Implementing the algorithm into this machine is straightforward. The industrial computer processes all the information transmitted to it from the camera, which is positioned on a support attached to the carriage system, allowing it to continuously analyze the bottles from one side of the belt to the other. The camera will be located a few rows behind the sprayers so that the frames are processed, the information is stored, and later, at the right moment, is communicated to the PLC that controls the sprayers.

This information will be shared through a certain number of addresses on the Modbus network. The data to be provided to the PLC consist of length intervals in millimeters where some misalignment is verified between the observed rows of bottles. With this data, the PLC will allow the sprayers to rise alightly in altitude, avoiding contact with any misaligned bottle. This prevents potential damage to the sprayers, avoids causing one or more bottles to fall, and prevent spraying the interior of the bottles. The length intervals are not highly precise, merely identifying if a row is misaligned or not, but also not entirely precise, as some intervals may contain a part that is aligned. This is justified by the fact that the intervals are selected according to the camera's field of view and whether there is any misalignment in that frame.

In this type of application, it is not strictly necessary for the intervals to be completely precise, containing only the misalignment. An interval can include a frame where the first half of the image shows a misalignment and the second half is aligned. This is acceptable due to the nature of the surface treatment process for the bottles. Bottles that have not been sprayed will still receive treatment as they slide against bottles with the product on their surface during transport on the conveyor belts.

4.1. Modbus communication

Communication between this system and the PLC responsible for bottle spraying is essential for the overall performance of the machine. As explained in section 2.2, this communication operates by sharing information through different registers between the client and the server. In this case, the server is implemented on the PLC, which has created various types of registers to facilitate communication with both the computer vision system and other PLCs within the factory. For this communication, only ”holding registers” are used, specifically the range of registers 40001-49999, which are read-write numeric values.

In this information sharing process, the algorithm will write the length intervals into a range of registers, but it will also require some information stored in other registers. Therefore, the algorithm will read some values from certain registers while writing the intervals into other registers (cf. appendix A).

These values are used to make the algorithm work properly, making it possible to adjust to different machines, with different dimensions.

4.2. Frame extraction process

The extraction of frames through the camera will be done at a considerably high speed, which complicates effective frame extraction. In this task, there are two alternatives: either analyze all frames captured by the camera as if it were a live video or analyze only certain strategically chosen frames to cover the entire conveyor belt.

Analyzing all frames captured by the camera would result in a large amount of information to process, much of which would probably be redundant due to a high overlap rate between frames. This means that intervals on the conveyor belt that could be analyzed with a single frame would instead be analyzed by multiple frames, unnecessarily using significant computational resources and time.

Instead of analyzing all the frames captured by the camera, a more intelligent method will be used. This approach saves time, computational capacity, and memory by using less information. The method employs strategic locations to capture frames, ensuring that the entire conveyor belt is covered from side to side without leaving any bottles out of the analysis.

4.2.1. FOV- field of view

To calculate these strategic points, it is first necessary to determine the camera’s field of view (FOV). This concept refers to the camera’s viewing area, which will be used to calculate how many millimeters of the conveyor belt the camera is able to capture in a single frame.

To achieve this, it is mandatory to calculate the camera’s viewing angle. Using this value along with trigonometry, it is possible to determine the length of the conveyor belt visible to the camera.

By observing figure 4.1, it is possible to clearly understand that the camera’s viewing angle (θ) can be calculated using the camera’s focal length (f) and the sensor size (h). In this case, since it needs to calculate the horizontal length, the relevant sensor size corresponds to its horizontal length. Thus, the following equation is obtained (Princeton Instruments, nd):

$$\frac{\theta}{2} = tan^{-1}(\frac{h}{2f}) \tag{4.1}$$

$$\theta = 2 \times tan^{-1}(\frac{h}{2f}) \tag{4.2}$$

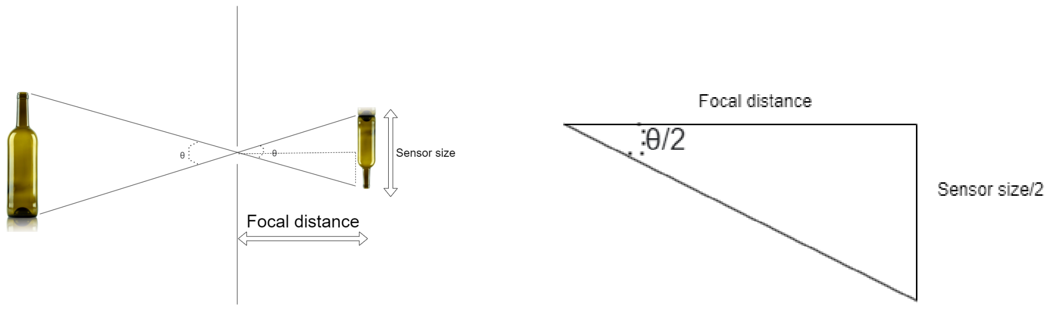


Figure 4.1: Representation of the camera's angle of view (left). Triangle used to calculate θ (right)

Once the camera's viewing angle is determined, along with the height at which the camera is positioned relative to the conveyor belt (a), it is possible to calculate the horizontal length of the conveyor belt that will be visible in any given frame.

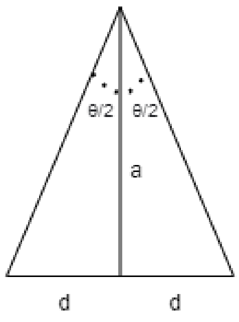


Figure 4.2: Representation of the horizontal length in relation to the height and the camera's viewing angle

By observing figure 4.2, it is possible to arrive at the equation capable of calculating the horizontal length of the conveyor belt that will be detected by the camera (Princeton Instruments, nd):

$$d = a \times tan(\frac{\theta}{2}) \tag{4.3}$$

With this value, it is now achievable to choose locations on the conveyor belt where frames should be extracted from, considering that the camera's projection is at the center of the frame. This means it detects bottles from its position - d until its position + d. Therefore, the camera's field of view is characterized by an angle of 48.46º and at a height of 70cm, each frame captures 630mm of the conveyor belt's horizontal length.

4.2.2. Overlapping

Knowing the camera's field of view allows to calculate the locations for capturing frames, however this alone is not sufficient; there are other considerations that must be taken into account.

Firstly, communication with the PLC was tested using an algorithm whose function was solely to connect to the server and read the register related to the position of the carriage system, hence the position of the camera and sprayers. The aim was to test the computer's capability to read this register and track every millimeter of the conveyor belt. It was found that the speed of the carriage system exceeded the computer's speed in reading the register, resulting in occasional jumps of a few millimeters up to a maximum of 50 millimeters. In other words, the computer did not consistently read every millimeter; it could read 300 mm in one instance and up to 350 mm in the next reading.

To ensure that frames were captured at the calculated locations, intervals were created corresponding to locations plus or minus 35 mm, creating a 70 mm interval where the algorithm could capture the frame. This ensured that within these 70 mm intervals, the algorithm would successfully capture at least one distance value, thereby capturing the frame at that point.

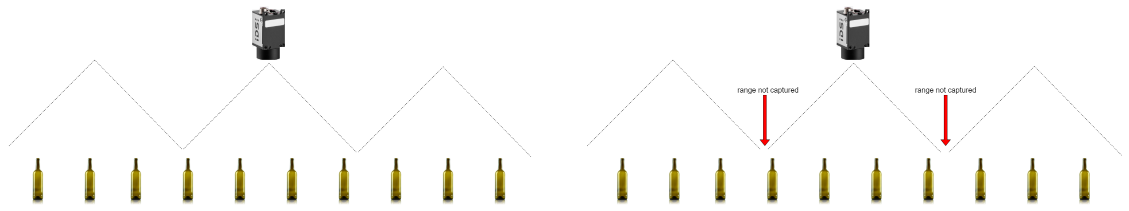


Figure 4.3: Frame capture without overlap at ideal points (left). Frame capture without overlap within intervals, but not exactly at the ideal points (right).

From figure 4.3, it is understood that these intervals can result in undetected zones. If frames are captured without overlap, just one being taken slightly above or below the ideal location can create an unanalyzed zone.

Another important aspect in this task is that in a frame captured, for example, from 400 mm to 800 mm, there could be a misaligned bottle near the edges of the frame, close to 400 mm or 800 mm. In this case, the interval sent would be from 400 mm to 800 mm, and there must be assurance that from 400 mm to 800 mm, the sprayers are safely positioned at the top, with no risk of colliding with a misaligned bottle or spraying its interior.

To address all these potential issues that could impact algorithm performance, frames will be captured with a 20% overlap, effectively mitigating these challenges. With intervals of 70 mm, there won't be any issues because the overlap ensures frames cover these intervals. If a bottle is misaligned at the left edge of a frame, it will also be captured by the preceding frame, thus ensuring the sprayer's safety by the time it reaches that point. Similarly, if a misaligned bottle is at the right edge of a frame, the next frame will detect it, ensuring the sprayer remains safe until the end of the following frame, assuming no further misalignments occur ahead.

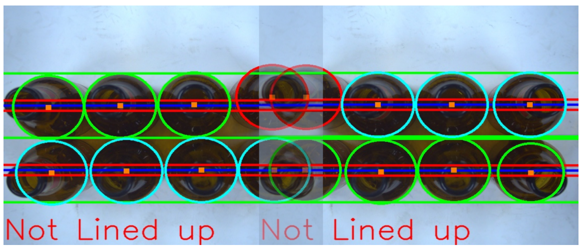


Figure 4.4: Two images analyzed with overlapping of 20%

Figure 4.4 depicts the process within the algorithm, where one frame is captured followed by the next positioned such that the initial 20% of the second frame aligns with the final 20% of the first frame. This arrangement extends the misalignment interval from the beginning of the first frame to the end of the second frame, instead of solely from the beginning to the end of the first frame. This ensures the sprayer descends only in a safe position, rather than at the end of the first frame where it might still collide with the misaligned bottle.

4.3. Type of frame analysis

After extracting the frames, the next step involves their analysis, specifically detecting the circles representing the bottles and subsequently assessing whether the bottle rows are aligned or not.

Due to the camera's altitude above the carpet, frames can detect multiple rows of bottles rather than just one. Consequently, it's possible to detect several rows at once or one row at a time. Both approaches are feasible, but they come with nuances. Analyzing multiple rows simultaneously means it may not be necessary to analyze frames for every movement across the carpet, as some rows can be detected in one pass that will repeat in subsequent movements. However, a significant challenge with this approach arises because while the camera moves in a single direction—either left or right — the carpet displaces the bottles perpendicular to the camera's movement. This means that as the camera moves from one end of the carpet to the other, the rows of bottles shift, potentially moving towards the top of the frame. If a row gets too close to the top, it may no longer appear within the camera's field of view and consequently may not be analyzed.

In conclusion, analyzing multiple rows in a single detection can lead to performance errors in the algorithm, as the uppermost row of bottles may disappear from view and evade analysis at any moment. Because this issue cannot be controlled, the process focuses solely on analyzing one row of bottles: specifically, the row directly under the camera at the start of its movement, which is positioned centrally in the frame. Therefore, in the initial frame, the algorithm seeks to analyze only the row of bottles closest to the frame's center, typically at an y-coordinate near $y = height/2$. In subsequent frames, the goal is to track this same row to ensure successful analysis.

This is achieved through an iterative approach where the algorithm searches through all frames, except the first, to identify the row of bottles closest in y-coordinate to the one detected in the previous frame. This iterative tracking method is effective because the carpet's displacement between consecutive frames is minimal enough to reliably follow the same row of bottles across frames.

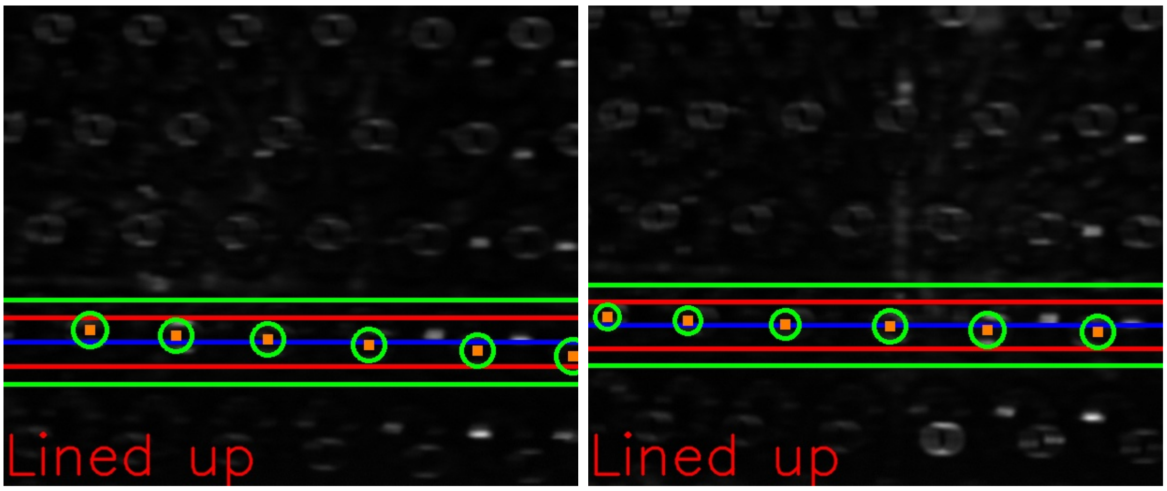


Figure 4.5: Frame taken near the beginning of the carpet (left). Frame taken near the end of the carpet (right)

Figure 4.5 confirms that the rows visible at the beginning of the carpet may no longer be visible at the end, highlighting the inability to analyze all rows of bottles present at the start. Instead, only the central row is analyzed to ensure it remains within the camera's field of view throughout the carpet's movement. This approach safeguards against losing visibility of the row being analyzed.

4.4. Structures used

Within all the logic needed for the system to work, it was necessary to create some particular structures such as a three-dimensional matrix, a buffer and two storages for distances.

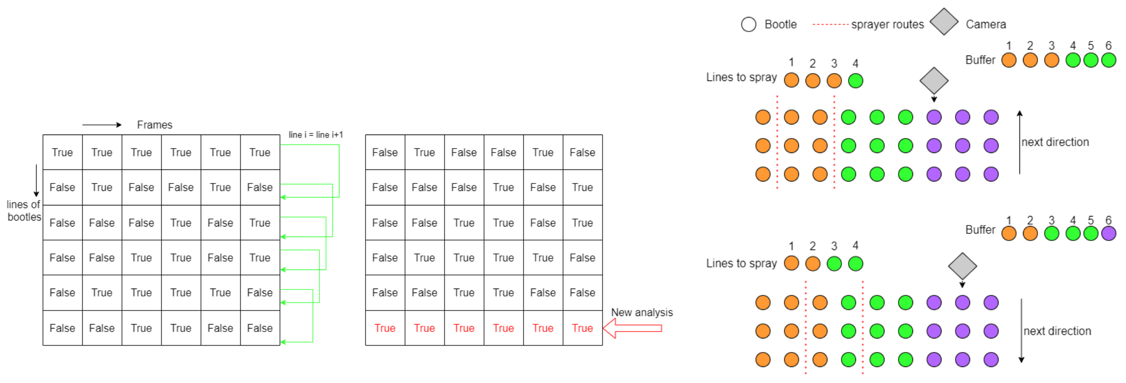
4.4.1. Three-dimensional matrix

During the frame capture process, it is essential to have a structure to store the frames. This process could simply save the frames as images in a computer folder and then access this folder later when necessary. In order to avoid this, a three-dimensional matrix was created to store all captured frames for later use. The x and y dimensions of this matrix correspond to the dimensions of the frames, which are constant, while the value of z will be the number of frames to be stored.

4.4.2. Buffer

The buffer mentioned consists of a matrix with dimensions of (number of frames x rows of bottles - 1). In this context, the number of frames corresponds to the frames that are captured during each pass of the camera from one end of the carpet to the other. On the other hand, "rows of bottles" refer to the number of rows of bottles that exist between the row of bottles at the first sprayer and the row directly beneath the camera. This buffer serves to store the analysis for all the frames pertaining to each bottle line between the first sprayer and the camera.

The buffer here utilizes (rows of bottles - 1) columns, functioning as follows: First, the buffer updates its data by updating the rows - meaning row i becomes row i+1 - allowing the last row to write the new analysis values made in the new pass. Only then the buffer is read on the lines to be sprayed and, consequently, analyzed to check for misalignment. This ensures that the first and second rows are sprayed by sprayer 1, while the fourth and fifth rows are sprayed by sprayer 2, figure 4.6.



True	True	True	True	True	True
False	True	False	False	True	False
False	False	False	True	False	True
False	False	True	True	False	True
False	True	True	True	True	False
False	False	True	True	False	False

False	True	False	False	True	False
False	False	False	True	False	True
False	False	True	True	False	True
False	True	True	True	True	False
False	False	True	True	False	False
True	True	True	True	True	True

Figure 4.6: Buffer update (left). Outline of how the buffer works (right)

4.4.3. Distance storages

There are two distance storages, one storage, with the dimensions equal to the buffer, that stores the ideal length values where the frames should be captured, and an array where it is stored the length intervals where the frames can actually be captured. As the algorithm has to work considering various mat sizes, the values in the arrays will be up to 10000mm, ensuring that no mat will need more intervals than the maximum, thus having values that will not be reached by any machine (since no machine manufactured by Vidromecânica will have 10000mm of mat). As the frames are captured, the matrix of ideal length values is replaced by the actual value of the distance where they were captured. This way, the matrix is updated so that it always provides information about how far away the capture occurred. This allows the matrix to have all the distances of the actual frames, and the previous ones taken in previous lines of bottles, so that when the algorithm has to communicate a misalignment interval, it will be exact interval, where that frame was taken.

4.5. Tasks in each location

In this type of system, the tasks to be performed can be carried out at the same time, capturing, analyzing the frames and all the necessary logic can all take place while the camera is traveling from the beginning to the end of the carpet and vice versa. This approach accommodates potential delays at the frame capture, since at certain limits, some frames might not be captured, which is why each task is carried out at different times.

4.5.1. Camera wandering

When the camera wanders from the beginning to the end of the carpet and vice versa, only the frames are captured at the calculated intervals and they are then stored within the three-dimensional matrix. If the camera moves from the beginning to the end of the carpet, the frames are stored sequentially from the beginning to the end of the matrix, but if it moves from the end to the beginning of the carpet, then the frames are stored from the end to the beginning of the matrix, so that the frames are always stored in the same order. Additionally, the ideal values of the distance matrix are also changed here, i.e. the ideal values where the frames should have been taken will be replaced by the distance values where the frames were actually captured. This adjustment provides precise information about the observed interval of carpet length within the frames. This interval will be written to the register if there is any misalignment.

4.5.2. Camera stop at the end of the carpet

At the end of the carpet, the camera and sprayers pause for a few seconds to allow the carpet to advance the next rows of bottles, both for spraying and analysis purposes. This moment is ideal for executing all logic and frame analysis. Initially, the buffer is updated where each row i becomes equal to row i+1, and the last row is updated only after the bottle analysis. Each frame is analyzed by a function that returns a value indicating alignment (True) or misalignment (False). Based on this analysis of each newly captured frame, the last row of the buffer is filled. Subsequently, the buffer is analyzed, focusing only on rows to be sprayed in the next traversal. If any misalignment is detected during this analysis, the distance values from the start to the end of the carpet observed in that frame are stored. The analysis continues with subsequent frames; if a True value is encountered, the stored values still the same, but if another False is detected, the end value is updated to encompass both frames, capturing both misalignments. The stored values are then written into predefined registers and subsequently read by the PLC, which then knows where to raise the sprayers and halt the liquid injection accordingly.

4.5.3. Camera stop at the beginning of the carpet

Just as when the camera is stopped at the end of the mat, at the beginning of the mat, the process follows the same sequence: the buffer is updated first, followed by frame analysis. Analysis results are then written into the last row of the buffer. Subsequently, the lines designated for spraying are analyzed from the buffer, and any detected misalignment intervals are recorded in the registers, if present. This systematic approach ensures that all necessary tasks are carried out efficiently and in the correct order, regardless of the camera's position on the carpet.

5. Tests and results

All tests were conducted with the camera fixed at a constant height above the conveyor belt, ensuring it captured the same length of the belt for each test, operating at 34 fps. The camera was positioned 70 cm above the belt, allowing it to capture 630 mm of it. However, with a 20% overlap, the points where the frames are captured are spaced 510 mm apart. The machine used for the tests has the conveyor belt starting at 50 mm and ending at 4300 mm, making the belt 4250 mm long. Initial trials revealed a high percentage of hallucinations in areas outside the belt, both at the beginning and the end. To mitigate this issue, the first and last frames must exclude the non-belt areas at the start and end of the machine. After making this adjustment, the following optimal points for frame capture were calculated: 350 mm, 860 mm, 1370 mm, 1880 mm, 2390 mm, 2900 mm, 3410 mm, and 3750 mm, resulting in each row being represented by 8 frames. The creation of line definition and alignment zones was calculated with distances above and below the average y-coordinate of the circles, using 2.6 $\times$ the average radius and 1.5 $\times$ the average radius, respectively.

5.1. Initial tests

Initially, tests were conducted using only natural light. After setting up the camera on the support, two lens parameters were adjusted: focus and diaphragm opening, allowing the image to become brighter or darker. These parameters were modified until the lens was focused on the bottle necks and the image had sufficient light to detect the bottles. It was concluded that an aperture setting between 80-100% was necessary for the bottles to be clearly visible in the frames. Subsequently, several frames were captured and tested to optimize the Hough transform parameters for detecting the neck circles (because they are the most visible), and consequently, the bottles. Once suitable parameter values were determined, the entire code was tested over a few rows.

During the camera's moving, images were captured and stored. At the beginning and end of the conveyor belt, the buffer was updated, the captured frames were analyzed, and the analyses were inserted into the buffer. Finally, the buffer analysis for rows 1, 2, 4, and 5 (rows 1 and 2 sprayed by sprayer 1 while rows 4 and 5 sprayed by sprayer 2) was performed to check for any misalignment, with the detected misalignment lengths subsequently sent to the PLC. It was possible to obtain good results, though there were still some detection errors, which led to the implementation of lighting systems to improve the frame quality and maintain consistent conditions throughout the capture process.

5.2. Tests with lighting systems

Among the lighting systems tested in the laboratory, one performed slightly better than the others. All the systems are from the company Infaimon, with the chosen model being the BDRL-W90/46 24VDC with white light. Since there was a system with the same dimensions but with red light, it was also tested in the factory, as the support built for the chosen lighting system could also accommodate this one. After extensive testing with both lighting systems and with the diaphragm aperture set to 100%, and exhaustively adjusting the Hough transform parameters, it was possible to conclude that using the lighting system, whether white or red, resulted in very poor bottle detection performance. This was due to many areas of the bottles and the conveyor belt reflecting the LED light excessively, creating numerous hallucinations and bottle detection failures.

5.3. Final results

Reaching the conclusion that using any lighting system with the lens opening set to 100% results in poor performance, further tests were conducted without artificial light, using both white and red lighting systems with different aperture settings. For each test, an optimal aperture value was determined to provide the best results. Without artificial light, an opening of f/1.4 was used, while tests with the lighting system used an aperture of f/2.8, where a larger number indicates less light entering the lens.

The tests involved running the code for each system (no light, white light, and red light). After determining the optimal Hough transform values, tests were conducted over 42 rows, 21 times from the beginning to the end of the conveyor belt and vice versa, resulting in 336 frames, with 8 frames captured during each camera pass. After capturing the frames, they were analyzed to extract true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). These metrics were used to calculate several performance indicators and typical error percentages, such as hallucinations, deviations, non-detection of bottles, and locations where errors were most prevalent.

	Actual Positive	Actual Negative
Predicted Positive	303	0
Predicted Negative	33	0

Table 5.1: Results without artificial light

	Actual Positive	Actual Negative
Predicted Positive	302	2
Predicted Negative	26	6

Table 5.2: Results with white light

	Actual Positive	Actual Negative
Predicted Positive	254	11
Predicted Negative	70	1

Table 5.3: Results with red light

The tables displays the confusion matrices for each of the tests. From these results, the following conclusions can be drawn: without any lighting system, no misalignments were detected either correctly or incorrectly. On the other hand, this setup achieved good results similar to the second test with white light. Only the tests using red light performed worse, resulting in 70 false negatives and 11 false positives.

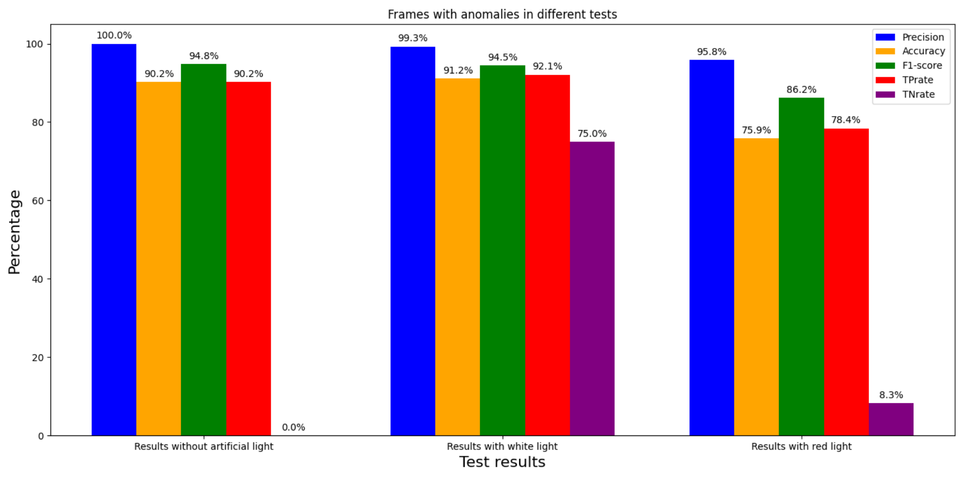


Figure 5.1: Metrics for the 3 tests

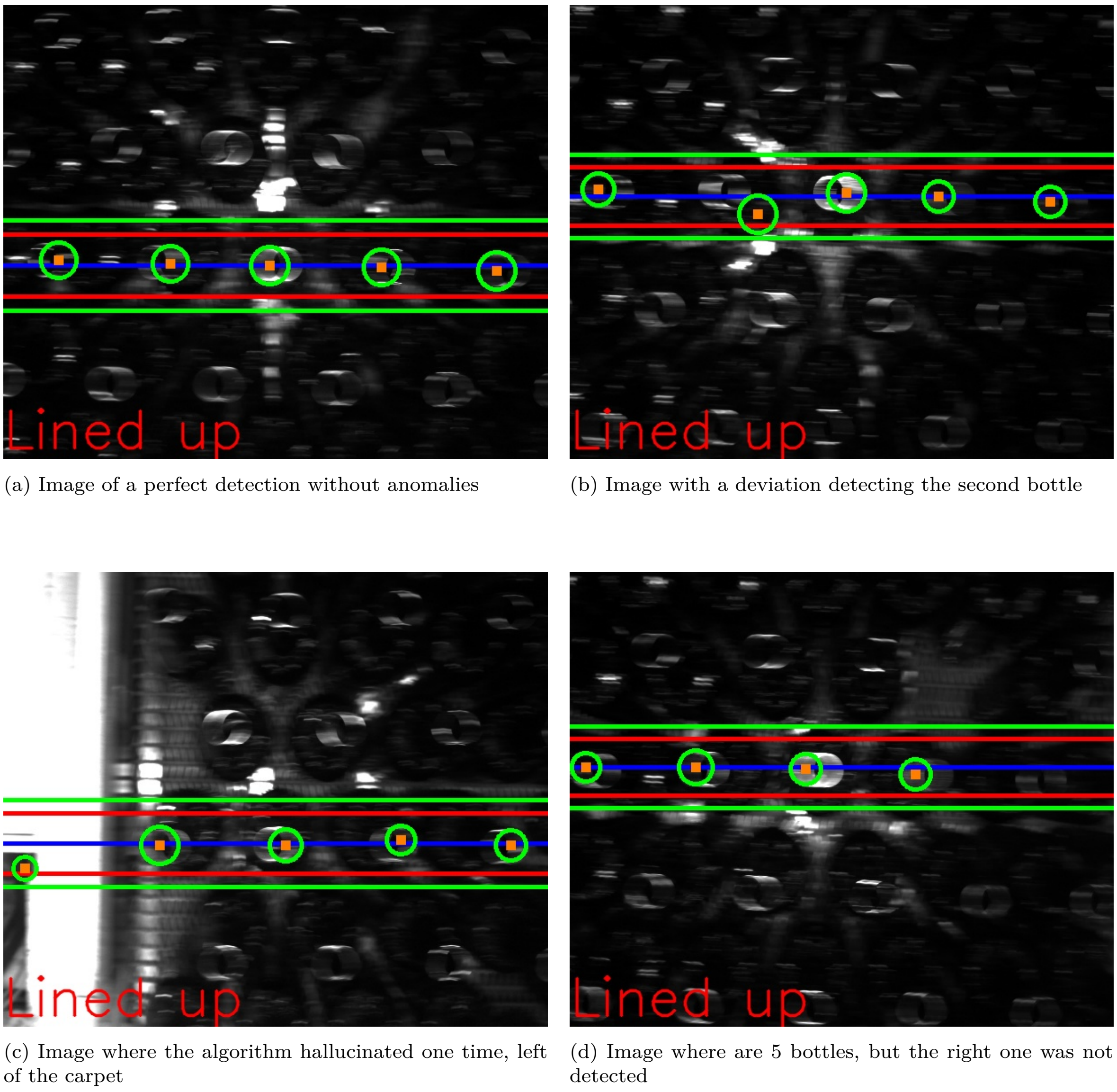
Regarding the metrics, they reveal to be appropriate but cannot be considered entirely valid, as there were no misalignments in the tests without light, and even the misalignments present in the other tests were few in number. However, since this is something unable to be controlled in the factory, the metrics (precision, accuracy, F1-score, TPrate, TNrate) are shown in the graph in Figure 5.1. It is possible to observe that the metrics for the first two tests are very similar and adequate, being always above 90%, except for the TNrate, which is 0 without artificial light since there were no misalignments. On the other hand, there were 8 misalignments using the white light, 6 of which were correctly detected. Finally, regarding the red light, the values are slightly lower than those of the other tests, with a very low TNrate, as it only detected one misalignment out of 12.

An important aspect of this work is the definition of a true positive. In this context, there is a ground truth indicating whether the lines are aligned or not, and the algorithm's prediction also indicates the alignment. For a true positive, the line in the frame must be aligned, and the algorithm must detect this alignment. This algorithm's detection can be 100% accurate or may contain anomalies such as minor deviations in bottle detection, hallucinations (detecting a bottle where none exists), or failing to detect a bottle in the line (e.g., the line has 5 bottles, but the algorithm detects only

4). These detections still qualify as true positives but include anomalies that do not affect the correct analysis of the frame.

	Results without artificial light	Results with white light	Results with red light
True positives	303-90,2%	302-89,9%	254-75,6%
True positives without anomalies	246-73,2%	249-74,1%	180-53,6%
True positives with anomalies	57-17%	53-15,8%	74-22%

Table 5.4: True positive values with and without anomalies



(a) Image of a perfect detection without anomalies

(b) Image with a deviation detecting the second bottle

(c) Image where the algorithm hallucinated one time, left of the carpet

(d) Image where are 5 bottles, but the right one was not detected

Figure 5.2: Examples taken from the test using white light

In the table, true positives are represented with and without anomalies. It is concluded that the number with anomalies is not significant, with a large percentage of true positives being completely correct.

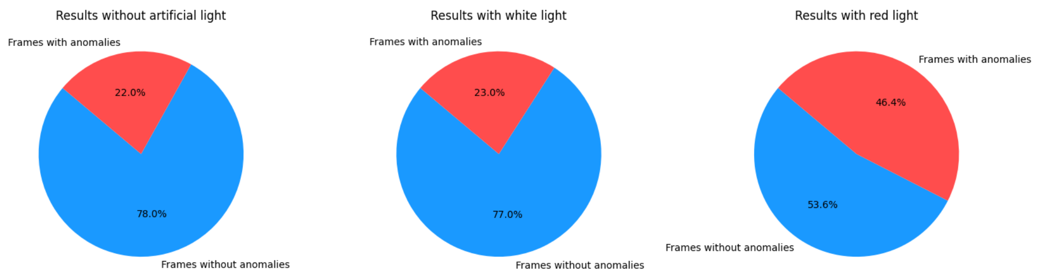


Figure 5.3: Percentage of frames with and without anomalies for the 3 tests

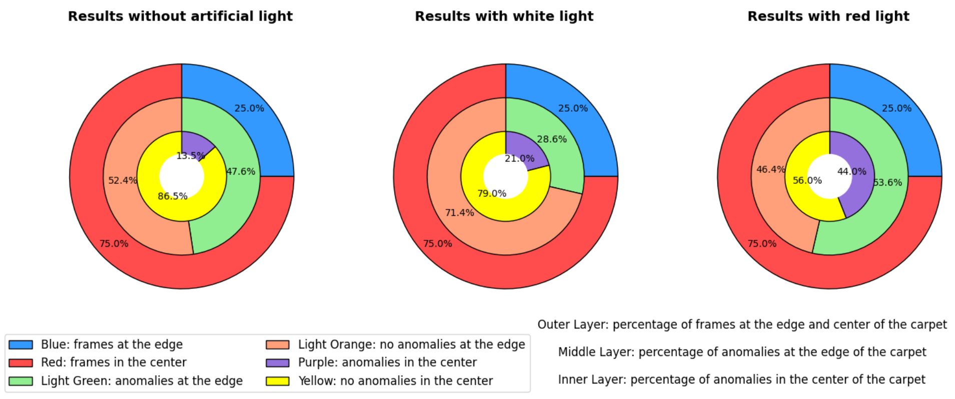


Figure 5.4: Percentage of anomalies at the edge and center of the carpet for the 3 tests

The previous figures contain pie charts analyzing the anomalies. Figure 5.3 represents the percentage of frames, out of the 336 frames, that have anomalies or not. It is evident that there is a small percentage of these anomalies, most of which are true positives and do not affect the correct analysis. Only in the tests with red light is it clear that almost half of the frames have anomalies, which is double the number of anomalies found in the other tests.

Figure 5.4 shows where these anomalies are more prevalent, either in the middle of the conveyor belt or at the ends. At each end, only one frame is captured, accounting for 25% of the frames at the ends of the belt. With no artificial light, there is a significant number of frames with anomalies captured at the ends, due to poor illumination. On the other hand, using white light, the anomalies at the ends were fewer thanks to more homogeneous lighting, but there were more anomalies in the middle of the belt due to reflections on both the belt and the bottles. Finally, with red light, there is a higher number of frames with anomalies at the ends compared to those without anomalies, and the same is verified in the middle of the belt, where there is a high rate of anomalies.

The last graph (figure 5.5) characterizes the different types of anomalies in 336 frames across various tests. These anomalies include failures to detect one, two or more bottles, hallucinations occurring once, twice or more times, and deviations in bottle detection, ranging from one and two to higher instances.

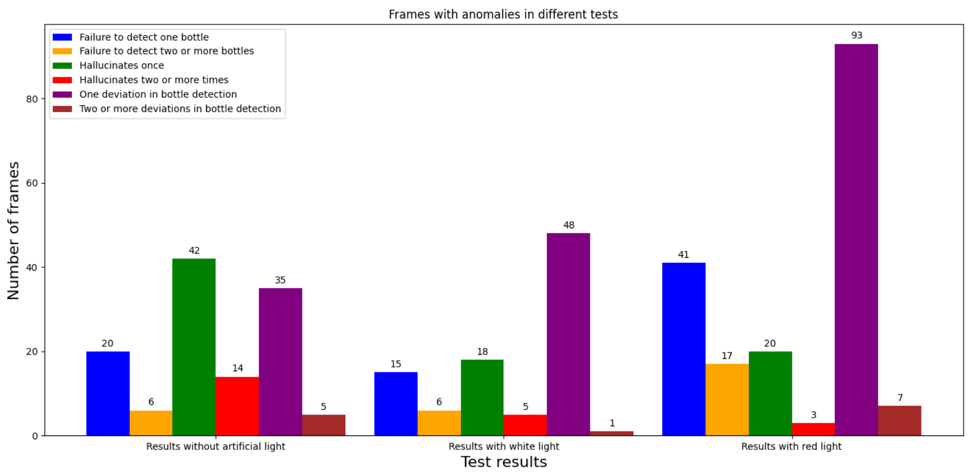


Figure 5.5: Frames with anomalies for the 3 tests

Regarding the tests with no artificial light, a higher number of hallucinations and deviations were observed. This is primarily justified by the fact that most hallucinations occurred at the ends of the conveyor belt, where side lighting from the machine negatively interfered. The environment is brighter here, whereas it is darker in the middle of the belt. In the tests using white light, hallucinations decreased significantly, partly due to reduced anomalies at the belt's ends. However, deviations increased because some light reflected off the belt or bottles, causing shifts in bottle center detection. Therefore, tests with red light showed fewer hallucinations compared to white light tests but had more deviations and failures to detect bottles. It was observed that red light is dimmer than white light, with greater intensity in the frame center than at the edges, resulting in numerous anomalies at the frame edges.

As it can be understood, the worst-performing tests are the ones with red light, standing out negatively from the others. Tests without artificial light and those using white light yielded very similar results, but the last ones slightly outperformed in mitigating anomalies, providing a more controlled environment. Considering that all tests were conducted during the day, with both factory and natural light from outside, achieving good performance without a lighting system is understandable. However, performance would drastically decrease if tested at night.

In conclusion, the system using white light is the best candidate to work in conjunction with the algorithm, due to its strong performance, consistent lighting conditions, and adaptability throughout different times of the day and year. While the white light system is the optimal choice among candidates, there are opportunities to further enhance its performance, which will be discussed in the future work section of the study.

5.3.1. Calculation of misalignment intervals and writing in the registers

As an example of the rest of the algorithm operating, the details regarding bottle line 11 are presented bellow. In this context, white light was used, and the code was executed. Since line 11 is an odd-numbered line, this means it was analyzed from the beginning to the end of the conveyor belt (on the other hand, if it were an even-numbered line, the analysis would be performed in the opposite direction). During this process, 8 frames were captured. At the end of the belt, the buffer was updated, and the 8 captured frames were analyzed. The analysis results were then written to the last line of the buffer, which ended up looking like this:

True	True	True	True	True	True	True	True
True	True	True	True	True	True	True	True
True	True	True	True	True	True	True	True
False	True	True	True	True	True	True	True
True	True	True	True	True	True	False	False
True	True	True	True	True	True	True	False
True	True	True	True	True	True	False	True
True	True	True	True	True	True	True	True
True	True	False	True	True	True	True	True
False	True	True	True	True	True	True	True

Table 5.5: Buffer after updating and writing new analyses

Observing the buffer above, it is possible to notice the detection of various misalignments, notably one in the first frame of the previous analyses. Before the sprayers and the camera return to the beginning of the belt, it is necessary to inform the PLC of possible misalignments in lines 1 and 2 sprayed by sprayer 1, and lines 4 and 5 sprayed by sprayer 2. As it is understandable, there is no misalignment in lines 1 and 2, but there is one in the first frame of line 4 and two in the last two frames of line 5, creating two intervals where the sprayers need to stop spraying and move up (both move up since when one does, the other also changes his position, as they are attached to the same support). With this buffer and the distances where the frames were captured, the algorithm wrote 4 values into 4 different registers:

Distance	Register
90	301
660	302
3148	303
4062	303

Table 5.6: Distances wrote on the registers

In this way, the odd registers get the start of the misalignments and the even ones the end, and there can be several distance intervals written on the next registers. The algorithm then calculated and wrote in the registers the 90-660mm interval containing the misalignment of the fourth line and 3148-4062mm, which encompasses the two consecutive misalignments of line 5.

6. Conclusion

The main aim of this dissertation was to demonstrate the feasibility of creating an algorithm capable of detecting bottles and calculating their misalignment intervals in a cold bottle coating machine manufactured by Vidromecânica. Until now, there have been no reports of any machine on the market with this functionality, and this work has proven that it is possible to develop an algorithm to address this issue.

Using the Hough transform, it is viable to detect the different circles representing bottle necks, thereby detecting the bottles, and it has been shown to be an effective method for this task. Through experiments conducted on the machine in an industrial setting with the aid of lighting systems, it was concluded that the results can vary significantly depending on the type of lighting used. Red light, for instance, was found to significantly decrease the algorithm's performance, even more so than when no artificial light was used. Conversely, white light provided superior algorithm performance, as it is stronger and helped correct some anomalies, providing a consistent luminous environment.

In conclusion, the algorithm has demonstrated that such a system can achieve high-performance results suitable for industrial implementation on Vidromecânica machines. However, improvements in lighting and machine materials could further enhance algorithm performance. Additionally, certain functionalities discussed further below, could be implemented to streamline machine operation, enabling continuous 24-hour operation without the need for pauses to adjust algorithm settings. In conclusion, a system has been developed capable of industrial operation with Vidromecânica machines — a system that was necessary but previously unavailable in the market.

6.1. Future work

The work presented in this dissertation demonstrates the feasibility of using an algorithm capable of detecting bottles through images captured by a camera aided by a lighting system, in a cold bottle coating machine. The algorithm is prepared to be implemented on any machine of this kind manufactured by Vidromecânica, accommodating potential variations in conveyor belt dimensions or machine sizes without compromising its performance. The only requirement is that the camera be positioned at a height of 70cm from the conveyor belt. As evidenced by the obtained results, the algorithm performs effectively with the white light system. However, in my opinion, its performance could be enhanced with certain modifications, such as using a conveyor belt made of a material and color that minimizes light reflection. Additionally, installing a fixed lighting system above the entire inspection area, positioned slightly higher than 70cm, that could ensure uniform illumination and prevent reflections that might affect bottle detection. This setup would create a more controlled environment with consistent lighting, potentially leading to improved performance.

Regarding additional functionalities that the algorithm could incorporate, a trigger system can be considered. The current machine triggers the carriage system to move across the conveyor belt using an infrared beam cell. While this method functions adequately, errors can occur if the beam cell detects any misalignment and triggers prematurely. This could lead to misalignments where the sprayers are positioned directly over bottles rather than the empty spaces between bottle rows.

Furthermore, the algorithm has been tested with various bottles of different heights and shapes through multiple trials to optimize the Hough transform values. For an industrial machine of this nature, transitioning between bottle types must be straightforward, achievable even by non-specialized operators who simply input relevant parameters into the PLC, which then communicates them to the algorithm. Therefore, there should be a method to train the algorithm or compute optimal Hough transform values, simplifying the transition from one production run to another and accommodating any type of bottles manufactured in the factories.

Bibliography

Akbar, F. S. P., Ginting, S. Y. P., Wu, G. C., Achmad, S., and Sutoyo, R. (2022). Object detection on bottles using the yolo algorithm. In *2022 4th International Conference on Cybernetics and Intelligent System (ICORIS)*.

An, M., Cruz, R., Ferrer, G., and Sy, L. (2010). Design and implementation of an automated bottle sorting system.

Araújo, H. (2023). Visão por computador. Slide presentation, Electrical and Computer Engineering Course, University of Coimbra. pages 31-96.

Baskin, K. (n.d.). The visual pathway: From the eye to the brain.

Bellis, M. (2019). Thomas savery and his steam engine.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media. pages 153-162.

Chadwick Optical (n.d.). The eye-brain connection.

Clapper, D. L. and Pfahl, D. P. (2015). System and method for automated bottle sorting.

Din, U. A., Mukhopadhyay, S., and Tariq, U. (2022). A robotic system for automatic identification and collection of recyclable plastic bottles. In *2022 5th International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*.

Forsyth, D. A. and Ponce, J. (2012). *Computer Vision: A Modern Approach*. Prentice Hall. pages 452-455.

Freitas, C. M. (2014). Protocolo modbus: Fundamentos e aplicações. Embarcados, url = https://embarcados.com.br/protocolo-modbus/.

Gizaw, A. and Kebebaw, T. (2022). Water bottle defect detection system using convolutional neural network. In *2022 International Conference on Information and Communication Technology for Development for Africa (ICT4DA)*.

IDS GmbH (2021). Ui-5250cp-m-gl (ab00252) datasheet. Technical Document.

IDS GmbH (2024a). Ids software suite download page. IDS Imaging Development Systems GmbH.

IDS GmbH (2024b). Ids software suite ueye manual. IDS Imaging Development Systems GmbH.

Kang and Atul (2020). Hough circle transform using opencv (the ai learner). The AI Learner Blog.

Ma, H.-M., Su, G.-D., Wang, J.-Y., and Ni, Z. (2002). A glass bottle defect detection system without touching. In *Proceedings. International Conference on Machine Learning and Cybernetics*, volume 2.

Mitsubishi electric (2022). Challenges in the industrial automation.

Mordor Intelligence (2022). Global factory automation and industrial controls market - industry report.

Nixon, M. S. (2013). *Feature Extraction in Computer Vision and Image Processing.* Academic Press. pages 184-195.

OpenCV (n.d.). Feature detection and description (opencv documentation). OpenCV Documentation.

Orachon, T. and Intani, P. (2012). Verify the number of soda bottles in the casket by using computer vision. In *2012 12th International Conference on Control, Automation and Systems.*

Pichler, S., Siegloch, F., and Zweimüller, J. (2020). The impact of automation on employment: Just the usual structural change? pages 1-26.

Princeton Instruments (n.d.). Field of view and angular field of view. Princeton Instruments Website.

Revimac (2024). Cold end treatment unit.

Roggia, L. and Fuentes, R. C. (2016). Automação industrial. pages 15-20.

Sanver, U., Yavuz, E., and Eyupoglu, C. (2017). An image processing application to detect faulty bottle packaging. In *2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus).*

Schneider Electric (2024). How to troubleshoot an m251 plc with serial communication via putty? Schneider Electric FAQ.

Shaikat, A. S., Tasnim, R., Khan, M., Imam, H., Rocky, S. A., and Alam, M. (2019). Development of plc and scada based spray coating system for application in glass bottle manufacturing industries of bangladesh. In *2019 Asia Pacific Conference on Research in Industrial and Systems Engineering (APCoRISE).*

U.S. Bureau of Labor Statistics (2018). Employer-reported workplace injuries and illnesses – 2017. pages 1-4.

Vidromecânica (2024). Vidromecânica.

Wagner Group (2024). Customer references: Wiegand-glas.

Wang, J., Guo, W., Pan, T., Yu, H., Duan, L., and Yang, W. (2018). Bottle detection in the wild using low-altitude unmanned aerial vehicles. In *2018 21st International Conference on Information Fusion (FUSION)*.

Zhou, X., Wang, Y., Xiao, C., Zhu, Q., Lu, X., Zhang, H., Ge, J., and Zhao, H. (2019). Automated visual inspection of glass bottle bottom with saliency detection and template matching. *IEEE Transactions on Instrumentation and Measurement*, 68(11).

Zhou, X., Wang, Y., Zhu, Q., Mao, J., Xiao, C., Lu, X., and Zhang, H. (2020). A surface defect detection framework for glass bottle bottom using visual attention model and wavelet transform. *IEEE Transactions on Industrial Informatics*, 16(4).

Appendix A. Table of registers used

Register	Definition	Read/Write
40106	Actual position	Read
40110	Starting position of the mat	Read
40108	Final position of the mat	Read
40122	Rows from sprayer one to the camera	Read
40300+	Misalignment intervals (even for start, odd for end)	Write

- Register 40106: Has the value of the actual position of the car system. Used to know if the camera is at the beginning, the middle or the end of the carpet;
- Register 40110: The starting position of the mat, to know when the camera is stopped in that position;
- Register 40108: The final position of the mat, to know when the camera is stopped in that position;
- Register 40122: Number of rows between the first sprayer and the camera. Is used to create the buffer;
- Register 40300+: From the register 40300, the intervals of misalignment are written. In the even registers is written the start and in the odd registers is written the end of the intervals;

Annex A. Camera specifications

Manufacturer: IDS Imaging Development Systems GmbH

Model: UI-5250CP-M-GL (AB00252)

Resolution: 1600 x 1220 pixels

Lenses: 8mm

Power supply: POE, 6-pin Hirose connector (HR10A-7R-6PB): 12 V - 24 V

Device temperature during operation 0 °C - 55 °C / 32 °F - 131 °F

Device temperature during storage -20 °C - 60 °C / -4 °F - 140 °F

Humidity (relative, non-condensing) 20 % - 80 %

Annex B. Opencv function cv.HoughCircles

cv.HoughCircles (image, circles, method, dp, minDist, param1, param2, minRadius, maxRadius)

- Image- 8-bit, single-channel, grayscale input image;
- circles- Output vector with 3 elements (x,y,radius) of the circles found;
- method- There are two methods for detecting circles: HOUGH_GRADIENT and HOUGH_GRADIENT_ALT;
- dp- Inverse ratio of the accumulator resolution to the image resolution. With dp=1, the accumulator and the image have the same resolution. If the accumulator's resolution is higher, there will be an increase of the computational cost, but it will enhance the ability to detect small circles;
- minDist- Minimum distance between the centers of the detected circles. If it is too small, there may be ”hallucination”, i.e. it may detect circles that aren't there. On the other hand, if it is too big, it might not detect some circles due to their close proximity;
- param1- First specific parameter of the method. In the case of HOUGH_GRADIENT, it represents the minimum intensity difference between the edge pixels. For HOUGH_GRADIENT_ALT, the threshold value is higher because it uses the Scharr algorithm;
- param2- Second method specific parameter. For HOUGH_GRADIENT, this is the minimum threshold of votes to accept a possible circle. The lower it is, the more false circles can be detected. The circles corresponding to the largest values in the accumulator will be returned first. In the case of HOUGH_GRADIENT_ALT, this is the measure of the circle's ”perfection”, i.e. the closer it is to 1, the better formed the circles selected by the method will be;
- minRadius- Minimum radius of the circles to be detected;
- maxRadius- Maximum radius of the circles to be detected.