



**Universidad de Valladolid**



**ESCUELA DE INGENIERÍAS  
INDUSTRIALES**

**UNIVERSIDAD DE VALLADOLID**

**ESCUELA DE INGENIERIAS INDUSTRIALES**

**Grado en Ingeniería Electrónica Industrial y Automática**

**Computer Vision system for rock  
classification using Artificial Neural Network**

**Autor:**

**Bernal Martínez, Antonio**

**Eusebio de la Fuente**

**UC Leuven - Limburg**

**Valladolid, Julio de 2021.**

**TFG REALIZADO EN PROGRAMA DE INTERCAMBIO**

---

**TÍTULO: Computer Vision system for rock classification using Artificial Neural Network**

**ALUMNO: Antonio Bernal Martínez**

**FECHA: 6 de julio de 2021**

**CENTRO: Faculty of Engineering Technology**

**UNIVERSIDAD: UC Leuven - Limburg**

**TUTOR: Wim Claes**

UC LEUVEN - LIMBURG

BACHELOR THESIS

---

**Computer Vision system for rock  
classification using Artificial  
Neural Network**

---

*Author:*

Antonio BERNAL

*Supervisors:*

Roel CONINGS

Cédric HAUBEN

Wim CLAES

*A thesis submitted in fulfillment of the requirements  
for the degree of Electronics - ICT*

*in the*

ACRO Research Group  
Faculty of Engineering Technology

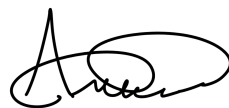
July 6, 2021

# Declaration of Authorship

I, Antonio BERNAL, declare that this thesis titled, "Computer Vision system for rock classification using Artificial Neural Network" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:



Date:

06/07/2021

UC LEUVEN - LIMBURG

# *Abstract*

Faculty of Engineering Technology

Electronics - ICT

## **Computer Vision system for rock classification using Artificial Neural Network**

by Antonio BERNAL

This project has been proposed by the Nijst company, located in Belgium. This company works with natural stone tiles and cobblestones. One of its activities is the purchase of large quantities of old road stones at very low prices, however the problem is that these stones, come in batches with different types mixed. At the moment, there are two operators in charge of the manual sorting, and the objectives are to reduce sorting time and costs.

This project explores the design of a computer vision system that is able to classify between the different types of cobblestones on a conveyor belt and separate them into different boxes. To carry out the recognition part, a computer vision system with a camera and lighting inside a black box, was designed. Additionally, a python script based on the OpenCV library was written, to detect the cobblestone and track it. After that, a Artificial Neural Network (ANN) was trained with histogram features such as, weighted mean, skewness, and kurtosis to classify the different types of cobblestones. Finally, for the separation part, a script was made with PLC programming so that when the rock passes next to its corresponding box, an actuator would push it inside.

Overall, it is concluded that due to the high classification accuracy, the correct performance of the system, and the extraordinary profitability, this project that has been carried out has been a success, heralding the promising future of artificial neural networks for classification tasks.

**Keywords** – Neural Network, Computer Vision, Classification, Cobblestone

## *Acknowledgements*

I would like to thank my supervisors Roel CONINGS, Cédric HAUBEN and Wim CLAES, for their guidance throughout this project.

Also thank my family and friends, for the support they have given me.

# Contents

<b>Declaration of Authorship</b>	<b>i</b>
<b>Abstract</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>List of Figures</b>	<b>vi</b>
<b>List of Tables</b>	<b>viii</b>
<b>List of Abbreviations</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Research Group and Company . . . . .	1
1.2 Motivation . . . . .	1
1.3 Objectives . . . . .	3
1.4 Thesis Outline . . . . .	3
<b>2 Background</b>	<b>4</b>
2.1 Computer Vision . . . . .	4
2.1.1 Image Thresholding . . . . .	5
2.1.2 Sobel filter . . . . .	7
2.1.3 Canny Edge Detector . . . . .	7
2.1.4 Morphological Transformations . . . . .	8
2.2 Artificial Neural Network . . . . .	10
<b>3 Methodology</b>	<b>12</b>
3.1 Overview . . . . .	12
3.2 Rocks . . . . .	12
3.3 Tools . . . . .	14
3.3.1 Hardware . . . . .	14
3.3.2 Software . . . . .	17
3.4 Computer Vision . . . . .	17

3.4.1	Object Detection . . . . .	18
3.4.2	Object Tracking . . . . .	22
3.5	Classification . . . . .	25
3.5.1	Image Acquisition . . . . .	26
3.5.2	Feature Extraction . . . . .	28
3.5.3	Dataset Creation . . . . .	32
3.5.4	Artificial Neural Network . . . . .	34
<b>4</b>	<b>Results and Implementation</b>	<b>37</b>
4.1	Classification Accuracy . . . . .	37
4.2	Implementation of the System . . . . .	38
<b>5</b>	<b>Conclusion</b>	<b>41</b>
5.1	Budget . . . . .	41
5.2	Profitability Study . . . . .	43
5.3	Summary . . . . .	44
5.4	Future Work . . . . .	45
<b>A</b>	<b>Additional Data</b>	<b>46</b>
	<b>Bibliography</b>	<b>52</b>



# List of Figures

1.1	Nijst Natuursteen company . . . . .	1
1.2	Outline of the 2020 project . . . . .	2
1.3	Types of rock faces . . . . .	2
2.1	Digitization process . . . . .	4
2.2	RGB color model channels . . . . .	5
2.3	Otsu's thresholding . . . . .	6
2.4	Global and adaptive thresholding . . . . .	6
2.5	Sobel filter . . . . .	7
2.6	Canny edge detector . . . . .	8
2.7	Erosion . . . . .	8
2.8	Dilation . . . . .	9
2.9	Opening . . . . .	9
2.10	Closing . . . . .	9
2.11	Neuron of a neural network . . . . .	10
2.12	Fully connected neural network . . . . .	11
3.1	Cobblestone types . . . . .	13
3.2	Cobblestone shapes . . . . .	13
3.3	Opaque box and conveyor belt setup . . . . .	14
3.4	Conveyor belt . . . . .	15
3.5	Lighting . . . . .	16
3.6	Camera . . . . .	16
3.7	Color image . . . . .	18
3.8	Gray scale Image . . . . .	19
3.9	Edges detected . . . . .	19
3.10	Image closed . . . . .	20
3.11	Image opened . . . . .	20
3.12	Detected rock . . . . .	21
3.13	Rotated rock . . . . .	21
3.14	Reduced rock . . . . .	22
3.15	Object tracking example . . . . .	24

3.16	Intermediate frame stripe for sorting . . . . .	25
3.17	Position to capture the rock in the frame . . . . .	25
3.18	Nijst equipment setup for database creation . . . . .	27
3.19	Rock washing and drying process . . . . .	28
3.20	Image histogram . . . . .	29
3.21	Skewness types . . . . .	30
3.22	Kurtosis types . . . . .	31
3.23	Dataset boxplot . . . . .	33
3.24	Grouped representation of the neural network . . . . .	35
3.25	Real representation of the neural network . . . . .	36
4.1	Confusion matrix . . . . .	37
4.2	Normalized confusion matrix . . . . .	38
4.3	Global implementation of the system (blue) . . . . .	39
4.4	Classification regardless of the face (blue) . . . . .	40
5.1	Asphalt in blue stone . . . . .	45
A.1	Global implementation of the system (gres) . . . . .	46
A.2	Global implementation of the system (porphyry) . . . . .	47
A.3	Classification regardless of the face (gres) . . . . .	48
A.4	Classification regardless of the face (porphyry) . . . . .	49
A.5	Asphalt in porphyry . . . . .	50

# List of Tables

3.1	Image features . . . . .	30
5.1	Salaries . . . . .	41
5.2	Cost of material equipment . . . . .	42
5.3	Stages for budget . . . . .	42
5.4	Profitability study . . . . .	43
A.1	Features table for ANN training . . . . .	51

# List of Abbreviations

<b>ACRO</b>	<b>A</b> utomation <b>C</b> omputer vision <b>R</b> obotics
<b>AI</b>	<b>A</b> rtificial <b>I</b> ntelligence
<b>ANN</b>	<b>A</b> rtificial <b>N</b> eural <b>N</b> etwork
<b>API</b>	<b>A</b> pplication <b>P</b> rogramming <b>I</b> nterface
<b>CMOS</b>	<b>C</b> omplementary <b>M</b> etal- <b>O</b> xide- <b>S</b> emiconductor
<b>CNN</b>	<b>C</b> onvolutional <b>N</b> eural <b>N</b> etwork
<b>CPU</b>	<b>C</b> entral <b>P</b> rocessing <b>U</b> nit
<b>DLL</b>	<b>D</b> ynamic- <b>L</b> ink <b>L</b> ibrary
<b>FPS</b>	<b>F</b> rames <b>P</b> er <b>S</b> econd
<b>GPU</b>	<b>G</b> raphics <b>P</b> rocessing <b>U</b> nit
<b>HDF5</b>	<b>H</b> ierarchical <b>D</b> ata <b>F</b> ormat <b>5</b>
<b>HSV</b>	<b>H</b> ue <b>S</b> aturation <b>V</b> alue
<b>IRR</b>	<b>I</b> nternal <b>R</b> ate of <b>R</b> eturn
<b>MLP</b>	<b>M</b> ulti- <b>L</b> ayer <b>P</b> erceptron
<b>NPV</b>	<b>N</b> et <b>P</b> resent <b>V</b> alue
<b>OpenCV</b>	<b>O</b> pen <b>S</b> ource <b>C</b> omputer <b>V</b> ision <b>L</b> ibrary
<b>PI</b>	<b>P</b> rofitability <b>I</b> ndex
<b>PLC</b>	<b>P</b> rogrammable <b>L</b> ogic <b>C</b> ontroller
<b>PNG</b>	<b>P</b> ortable <b>N</b> etwork <b>G</b> raphics
<b>PWM</b>	<b>P</b> ulse- <b>W</b> idth <b>M</b> odulation
<b>RGB</b>	<b>R</b> ed <b>G</b> reen <b>B</b> lue
<b>SL</b>	<b>S</b> upervised <b>L</b> earning
<b>UL</b>	<b>U</b> nsupervised <b>L</b> earning
<b>VDC</b>	<b>V</b> olts <b>D</b> irect <b>C</b> urrent

*Dedicated to my sister, Teresa, to encourage her to  
do an Erasmus...*

# Chapter 1

## Introduction

### 1.1 Research Group and Company

My bachelor thesis was made at ACRO (Automation, Computer vision and Robotics), a research group that belongs to UC Leuven-Limburg and KU Leuven as part of the Faculty of Engineering Technology. At ACRO, researchers as well as future PhD's are working on projects for Flemish companies.

This project has been proposed by the Nijst Natuursteen company (FIGURE 1.1), which works with natural stone tiles and cobblestones. It is one of the most competitive companies in its sector in Belgium. One of its activities is the purchase of large quantities of old road stones at very low prices. However, the challenge is that these stones, come in batches with different types mixed, and many man hours are required to sort them.



(A) Nijst logo.



(B) Factory in Munsterbilzen, Belgium.

FIGURE 1.1: Nijst Natuursteen company.

### 1.2 Motivation

In the old road stone lots, rocks of different types and sizes are mixed together. In 2020, a project was carried out to classify rocks by size (FIGURE 1.2). This project consisted of measuring the rocks at the beginning of the

conveyor belt with a light barrier, and depending on the size, a signal is sent to a Programmable logic controller (PLC), to push it to the corresponding box. There are 10 boxes for each of the 10 possible sizes.

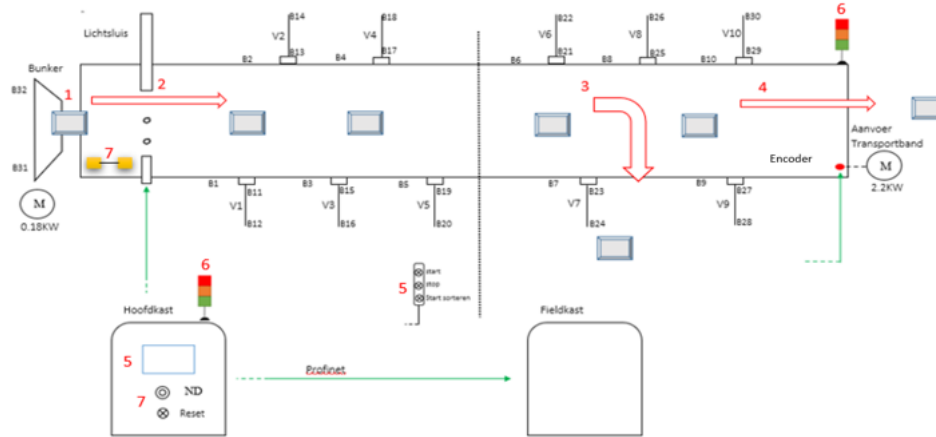
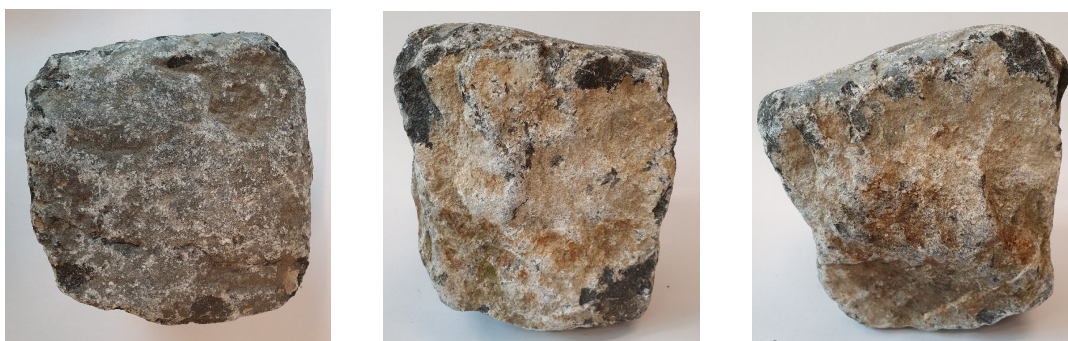


FIGURE 1.2: Outline of the project to classify rocks by size.

There is a drawback, and it is that the measurement of the cobblestone has to be taken on its visible face<sup>1</sup> (FIGURE 1.3). To place it in this position, it is necessary for an operator to do it manually, so it is not a completely automatic process.



(A) Visible face.

(B) Hidden face.

(C) Hidden face.

FIGURE 1.3: Difference between visible and hidden faces.

Immediately before the conveyor belt on which cobblestones are classified by size, there is the belt on which they are classified by type. Currently, this classification task is carried out manually by two operators. For this reason, and due to the desire of the Nijst Natuursteen company to reduce costs and increase the sorting speed, the motivation for this project (Computer Vision system for rock classification using Artificial Neural Network) arises, which will be developed starting from scratch, by Antonio BERNAL.

<sup>1</sup>The visible face is the surface that is exposed once the cobblestone has been placed.

## 1.3 Objectives

The **main objective** of this project is to design a system that automatically classifies between 3 types of rock, from the visible face of the rocks, therefore, an operator would have to manually place the rock so that a camera capture the rock visible face.

In addition, some **secondary objectives** were established, without which the project would be a success, but if achieved, would provide great added value. These objectives are:

- Classify regardless of the face of the rock that the camera captures. In this way, costs would be further reduced, since the operator, in charge of placing the rock with its visible face, would be saved.
- Obtain the measurements of the rocks, to potentially replace the project of classification by size, and have both classifications (by size and type) together in the same system.

## 1.4 Thesis Outline

**Chapter 1** provides an overview of the motivation and goals of the project. **Chapter 2** provides the theoretical background on which the project is based: computer vision and artificial neural networks. **Chapter 3** describes the methods that have been used to carry out the project. **Chapter 4** evaluates the results obtained and presents the global implementation of the system. **Chapter 5** summarizes the findings of the thesis and proposes paths for future research.



## Chapter 2

# Background

This chapter 2, aims to provide a better understanding of the fundamental concepts necessary to understand the thesis. This project is based on two pillars: computer vision (2.1) and artificial neural networks (ANN) (2.2).

### 2.1 Computer Vision

Computer vision is the set of techniques that deal with extracting relevant information from the physical world from images and video, using a computer system.

An image is a visual representation of a scene. And for it to be processed by a computer, it needs to be transformed into a digital image. This transformation is called digitization (FIGURE 2.1), and roughly speaking, it consists of two parts: sampling and quantification. Sampling (FIGURE 2.1b) consists of dividing the image into small samples or segments, and quantification (FIGURE 2.1c) consists of assigning a numerical value to each of these samples.

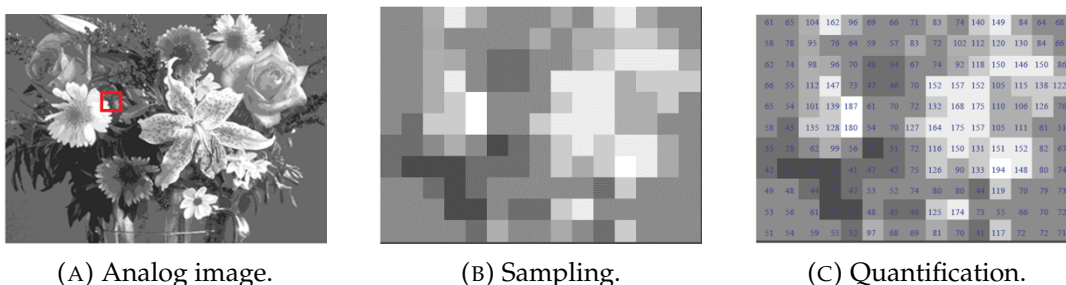


FIGURE 2.1: Digitization process.

A digital image is represented by a matrix. Each element of the matrix is called a picture element (pixel) and has a discrete value, called a gray level,

which represents its brightness or lightness. There are 256 gray levels, ranging from 0 (black) to 255 (white).

Real-world images are in color, and in them each pixel has 3 different gray levels (FIGURE 2.2), corresponding to the 3 channels of the RGB color model (red, green and blue). That is, the value of each pixel can have 256 different levels of red, 256 of green and 256 of blue, and combining them approximately 16 million of different colors are obtained.

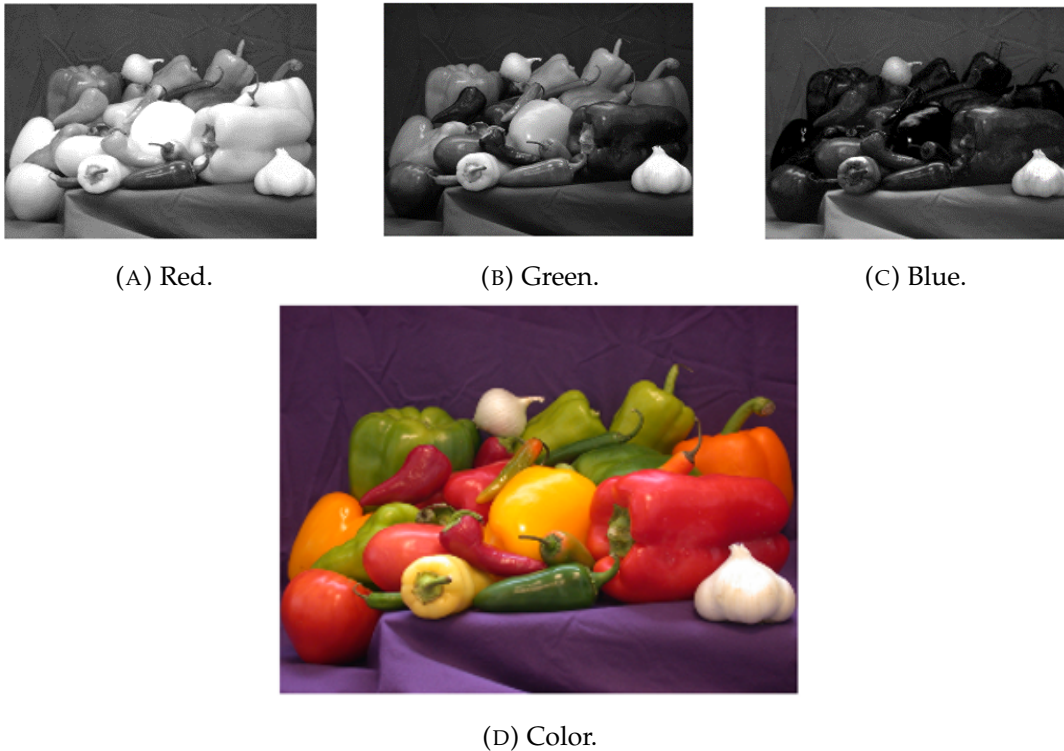


FIGURE 2.2: RGB color model channels. Source: [1].

In order to extract information from the images, it is necessary to apply image processing techniques, explained in the following sections (2.1.1, 2.1.2, 2.1.3 and 2.1.4).

### 2.1.1 Image Thresholding

The simplest method of image segmentation<sup>1</sup> is thresholding. From a gray scale image, binary images are created by applying this segmentation method [2]. There are several types of thresholding, including global thresholding, Otsu's thresholding and adaptive thresholding.

---

<sup>1</sup>Image segmentation is the process of dividing a digital image into several segments or sets of pixels. The goal is to simplify the image, so that it is easier to analyze [2].

In global thresholding (FIGURE 2.4b), for each pixel, the same threshold value is applied. If the pixel value is less than the threshold, this value is set to 0; whereas, if it is greater than the threshold, it is set to 1.

Otsu's thresholding (FIGURE 2.3), is used for automatic thresholding of an image. The algorithm returns a single threshold that separates the image pixels into two classes. This threshold is calculated by minimizing the within-class intensity variation or, in other words, by maximizing the between-class variation [3].

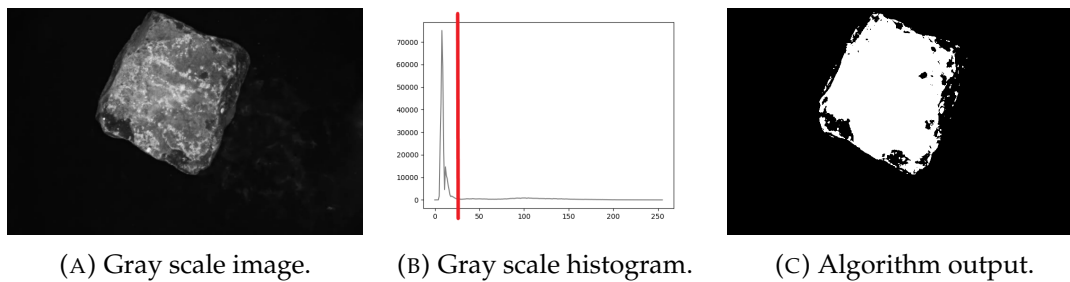


FIGURE 2.3: Otsu's thresholding. The threshold is the red line in FIGURE 2.3b.

Finally, the adaptive thresholding method (FIGURE 2.4c), is used when the image has different illumination levels. The algorithm determines the threshold of a pixel based on a small region around it. Therefore, different thresholds are obtained for different regions of the same image [4].

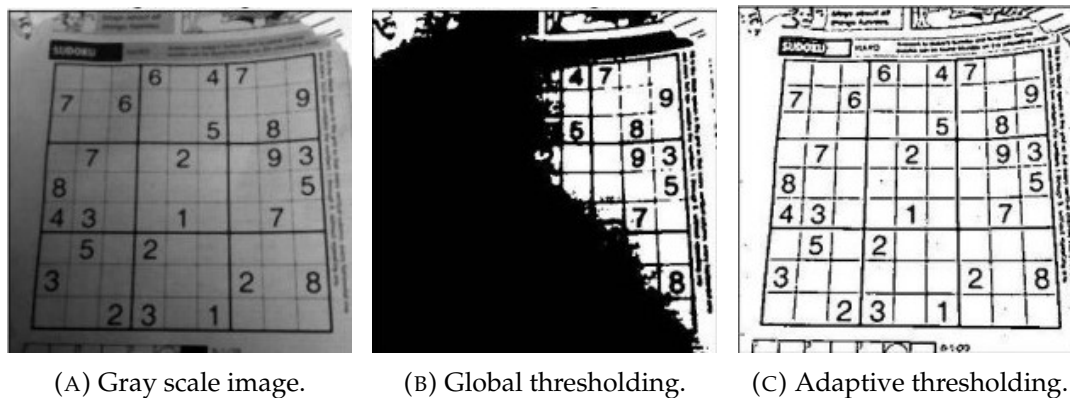


FIGURE 2.4: Comparison between global and adaptive thresholding. Source: [4].

## 2.1.2 Sobel filter

The Sobel filter or Sobel operator (FIGURE 2.5), used in computer vision for edge detection, is a discrete differential operator that calculates the gradient<sup>2</sup> of the intensity of an image, for each pixel. That is, for each pixel, it will give us information about the largest possible change (corresponding to the edges), and its direction and sense. In addition, it can specify the direction of the derivatives, vertical (FIGURE 2.5b) or horizontal (FIGURE 2.5c) [6].

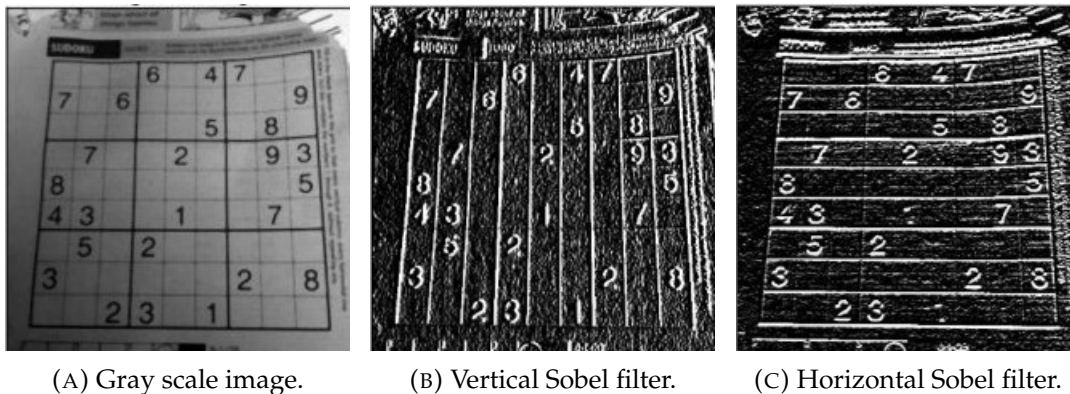


FIGURE 2.5: Sobel filter. Source: [7].

## 2.1.3 Canny Edge Detector

The Canny edge detector (FIGURE 2.6) is, a multi-stage edge detection algorithm, capable of detecting a wide range of edges [8]. It is divided into 5 stages:

- First, a Gaussian filter<sup>3</sup> is applied to remove noise and smooth the image.
- Second, the intensity gradients of the image are calculated.
- Third, apply lower bound cut-off suppression or gradient magnitude thresholding, to get rid of spurious response to edge detection.
- Fourth, apply double thresholding to detect possible edges.
- Fifth, apply hysteresis for thresholding. Suppressing the remaining weak edges and those not connected to strong edges.

<sup>2</sup>The gradient is a vector, which indicates the direction in which a scalar varies more rapidly [5].

<sup>3</sup>This filter uses a Gaussian function to blur an image.

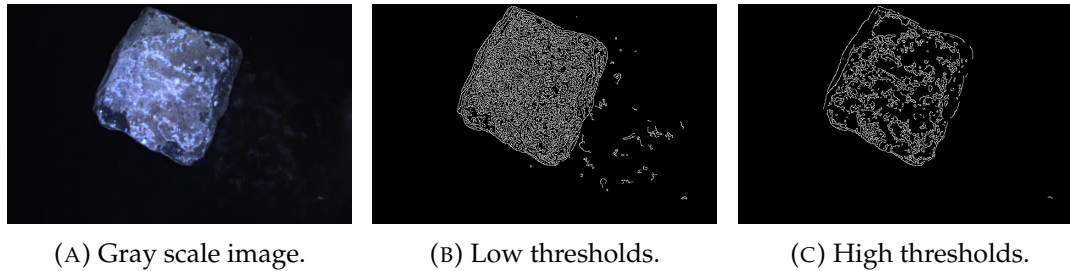


FIGURE 2.6: Canny edge detector.

### 2.1.4 Morphological Transformations

Morphological transformations are simple operations based on the shape of the image. A structuring element or kernel<sup>4</sup> decides the nature of the operation. The two basic morphological operations are: erosion and dilation [9].

The **erosion** (FIGURE 2.7) aims to reduce the size of an object. The operation is as follows: the kernel will go through all the pixels in the image, when all the pixels on the kernel have a value of 1 (white pixel), a 1 will be assigned to that pixel. Otherwise, if any of the pixels on the kernel is 0 (black pixel), the pixel will be assigned a 0 [9].

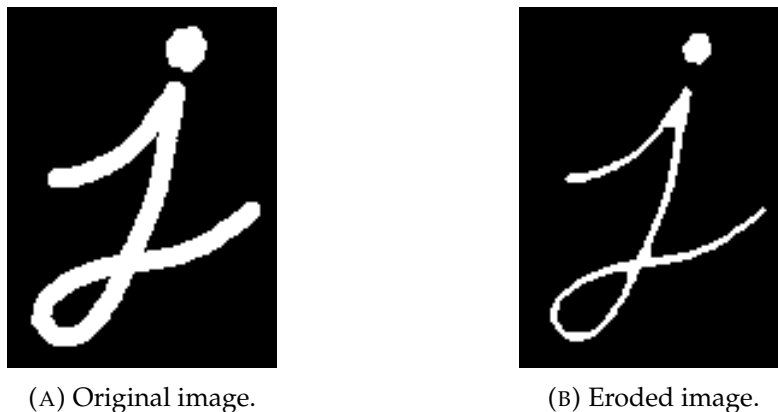


FIGURE 2.7: Erosion. Source: [9].

The **dilation** (FIGURE 2.8), contrary to erosion, serves to increase the size of an object. As in erosion, the kernel will go through all the pixels in the image, when all the pixels on the kernel have a value of 0, a 0 will be assigned. Otherwise, if any of the pixels on the kernel is a 1, will assign a 1 [9].

Once the two basic morphological operations (erosion and dilation) have been defined, it is now possible to define two more useful and complex operations: opening and closing. **Opening** (erosion + dilation) is used to remove

<sup>4</sup>The structuring element or kernel can have different shapes (rectangle, circle...).

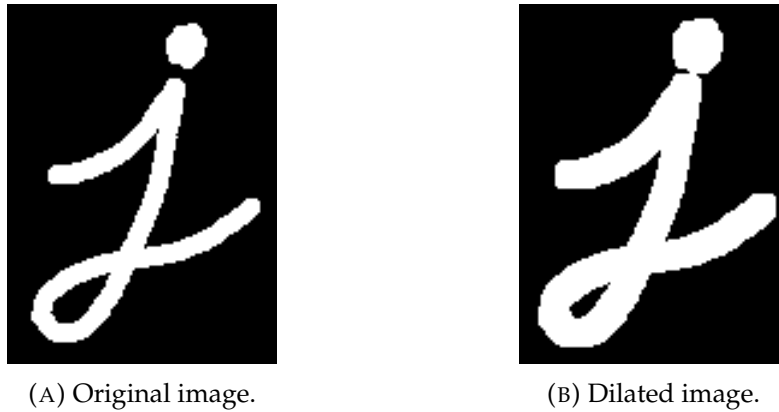


FIGURE 2.8: Dilation. Source: [9].

noise from the image (FIGURE 2.9). **Closing** (dilation + erosion) is useful for filling small holes that an object may have (FIGURE 2.10) [9].

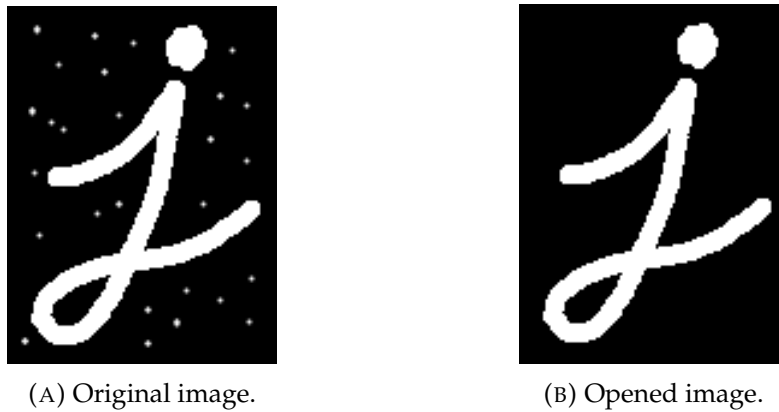


FIGURE 2.9: Opening. Source: [9].

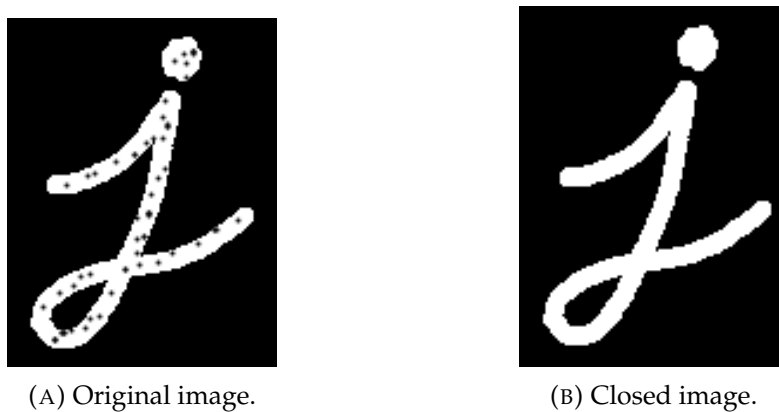


FIGURE 2.10: Closing. Source: [9].

## 2.2 Artificial Neural Network

In recent years, the resurgence of the neural network has revolutionized fields such as speech recognition, image recognition, natural language processing... A neural network, called artificial neural network (ANN) or multi-layer perceptron (MLP), is a supervised<sup>5</sup> machine learning model that has the ability to represent complex nonlinear relationships in the input data. The model was originally developed to copy the biological brain, but has since diverged to optimize classification and regression tasks<sup>6</sup>.

The basic unit of the neural network is the neuron (FIGURE 2.11), which takes a weighted average of the input values and then applies a nonlinear activation function<sup>7</sup> which returns a scalar value.

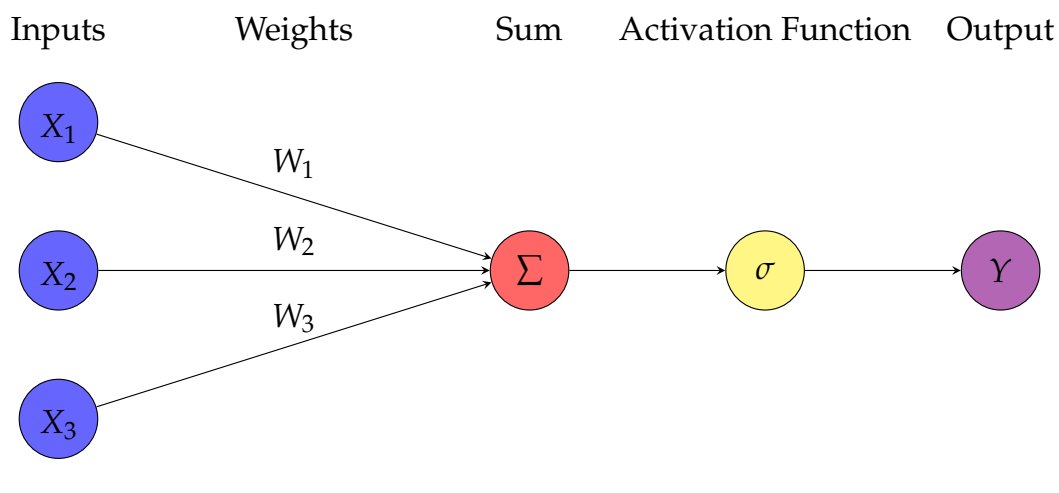


FIGURE 2.11: Neuron of a neural network.

Neural networks are often made up of layers, which in turn are made up of neurons. These layers, called hidden layers, are those between the input and output layers. When all neurons in a layer are connected to all neurons in the previous layer, the layer is called dense or fully connected (FIGURE 2.12). Neural networks with at least one hidden layer and with a non-linear activation function are universal approximators, which means that they can, a priori, approximate any continuous function of arbitrary complexity. In

<sup>5</sup>Supervised learning (SL) is the machine learning task of learning a function that maps an input to an output based on example input-output pairs [10].

<sup>6</sup>Analysis used to predict the outcome of a categorical variable, based on the predictor variables [11].

<sup>7</sup>The activation function of a neuron, defines the output of that node given an input or set of inputs. The most common activation function is the sigmoid, defined as:  $f(x) = \frac{1}{1+e^{-x}}$ , which returns an output value between 0 and 1 [12].

practice, more layers often provide more flexibility and power of representation, allowing the network to represent increasingly abstract relationships in the data.

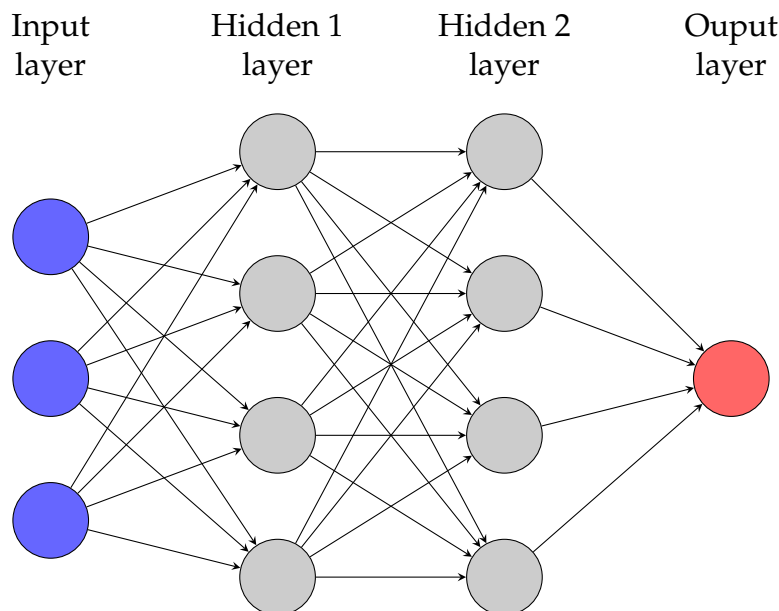


FIGURE 2.12: Fully connected neural network with two hidden layers.

For a neural network to learn, the developer must define a loss function<sup>8</sup> to evaluate the performance of the predictions of the network. During training, the neural network iteratively searches for a set of weights that minimize the loss function through a gradient-based optimization algorithm, such as gradient descent<sup>9</sup>. A process called back-propagation<sup>10</sup> is used to determine the contribution to the error of each weight and therefore how much you must change each weight in each iteration of the optimization.

<sup>8</sup>Loss functions are used to determine the error between the output of the model and the target value [13].

<sup>9</sup>Gradient descent is an iterative optimization algorithm for finding a local minimum of a function [14].

<sup>10</sup>Back-propagation (backward propagation of errors) is the most used method for calculating derivatives in a ANN [15].



## Chapter 3

# Methodology

### 3.1 Overview

The rocks are going to pass through a conveyor belt that is located outdoors, so the most viable and cheapest possible option is to install a camera at the beginning of the conveyor belt to obtain an image of the rock. With this image, the type of rock will be determined and a signal will be sent to a PLC to activate an actuator, in charge of pushing the rock to its corresponding box.

The conveyor belt is made of black rubber, which facilitates the work of detecting the rock. If a metal belt was used, there would be a lot of reflections due to lighting or ambient light, and it would be more difficult to classify the rocks.

The camera and lighting will be put inside an opaque box to always have the same illumination. If this was not done, as the ambient light varies throughout the day, sorting problems would be presented. In the opaque box, there will be two openings at the base, one for entry and one for exit, through which the rocks will pass.

### 3.2 Rocks

The types of cobblestones which will be classified are 3: Belgian gres, Belgian porphyry and Belgian blue stone. From now on, for simplicity, they will be referred to as: gres, porphyry and blue, respectively.

- Gres cobblestones (FIGURE 3.1a), have a smooth and unspckled face, and can have any plain color within the whole range.

- The porphyry cobblestones (FIGURE 3.1b), have a rough surface with black and white speckles and can have any color.
- The blue cobblestones (FIGURE 3.1c), like the gres, have a smooth and unspckled surface, and are usually blue or gray in color.

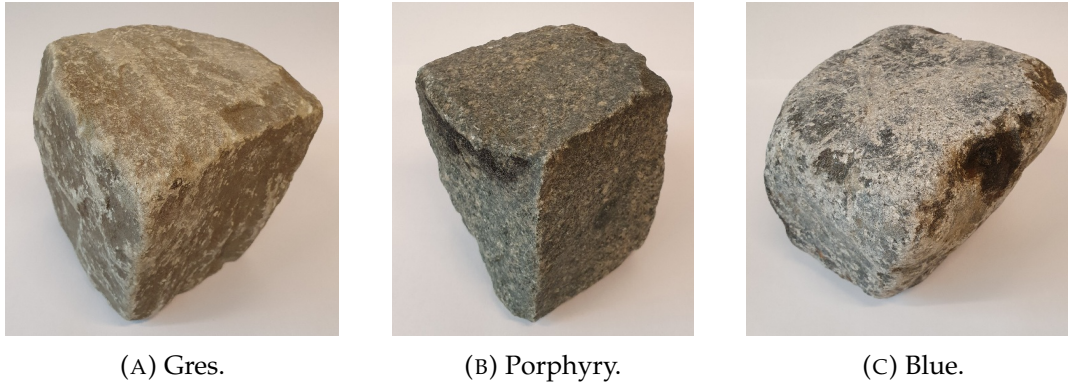


FIGURE 3.1: Cobblestone types.

As for the shape, the face of the cobblestone will be square (FIGURE 3.2a) or rectangular (FIGURE 3.2b), with the exception of any that may have been damaged, which will have a trapezoidal shape (FIGURE 3.2c).



FIGURE 3.2: Cobblestone shapes.

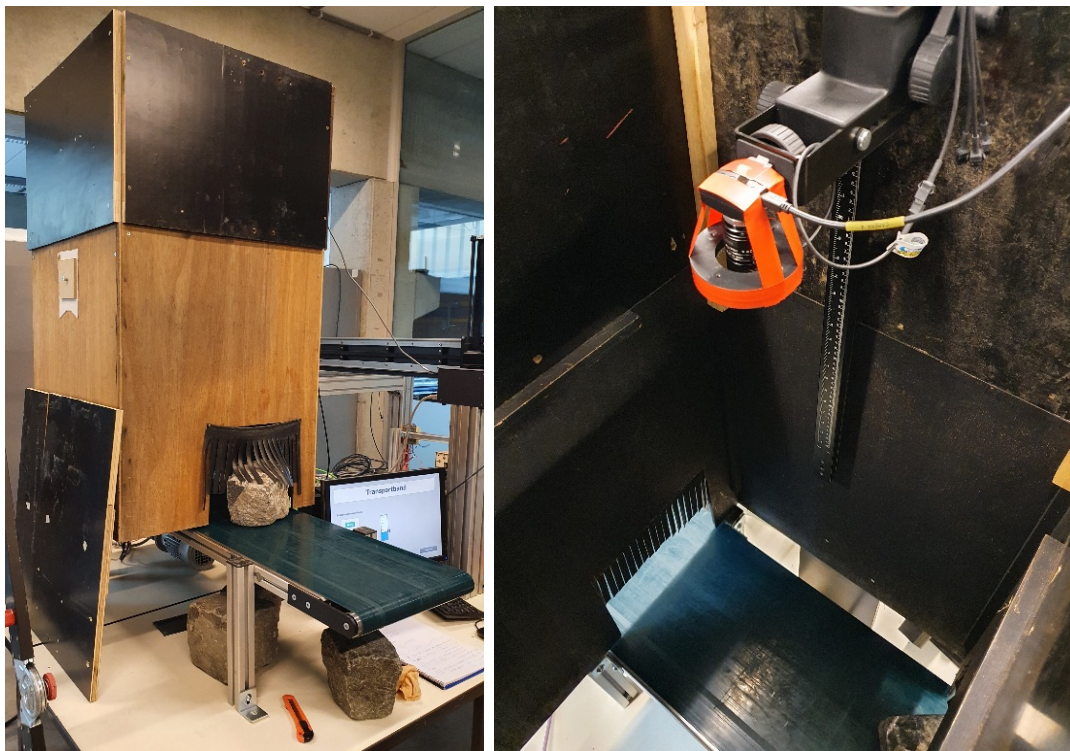
Regarding the size, there is a lot of variability. Square cobblestones can measure from 6x6 to 20x20 centimeters. And the rectangular ones, will have 5 possible measures: 10x20, 12x20, 13x20, 14x24 and 15x24, all of them in centimeters. And as for the height or depth of these, it can vary from 6 to 30 centimeters.

## 3.3 Tools

### 3.3.1 Hardware

The project was to be developed in the laboratories of the ACRO research group, so the setup, that would be used when the system would be in production at Nijst, was reproduced.

For this purpose, a prototype of an opaque box was built with a support to place it on top of the conveyor belt (FIGURE 3.3a), both the box and the support were built with recycled wood. The dimensions of the opaque box are 62x58x115 centimeters, enough to house the camera and lighting, at a height of 95 centimeters above the surface of the conveyor belt. The camera was mounted on a specific support for artificial vision, which was attached to the opaque box. The LED light ring is attached to the camera with duct tape<sup>1</sup> (FIGURE 3.3b).



(A) Opaque box and conveyor belt.

(B) Opaque box interior.

FIGURE 3.3: Opaque box and conveyor belt setup.

A 36x130 centimeter conveyor belt was used, driven by a motor with a gear-box (FIGURE 3.4). The speed of the belt could be varied through an interface,

<sup>1</sup>Since this is a prototype, the camera and LED can be attached like this, but the final system will require a structure on which to place the LED.

connected to a PLC with controllers, in order to adapt it to the speed of the Nijst belt (0.4 m/s).

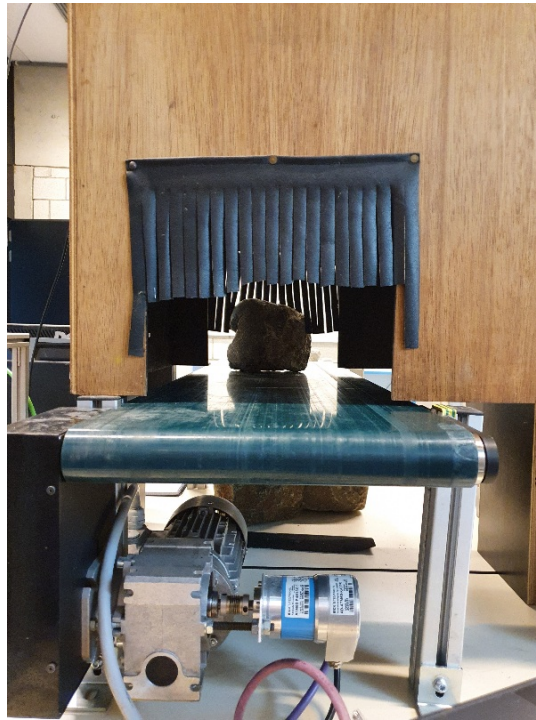


FIGURE 3.4: Motor-driven conveyor belt, and opaque box openings.

The LED ring light (FIGURE 3.5a) is the LDR2-90SW2, from CCS. It provides a white light, with a maximum power consumption of 14 Watts, provided that the operating temperature is between 0 and 40°C. The supply voltage of 24 VDC (Volts Direct Current) is provided by the power supply (FIGURE 3.5b) PD2-3024-2(A) (also from CCS) through a SMP-03V-BC cable. The power supply has a pulse-width modulation<sup>2</sup> (PWM) power supply current controller, at 62.5 kHz, with coarse and fine tuning. Both settings have 16 steps, and for the project in question, step 10 of the coarse and 16 of the fine were used.

The camera consists of the sensor (FIGURE 3.6a) and the lens (FIGURE 3.6b). The CMOS<sup>3</sup> sensor is the UI-1225-LE-C from IDS and was connected to the computer with a USB 2.0 cable. This sensor captures 752x480 pixel color

<sup>2</sup>PWM is a technique in which the duty cycle of a periodic signal is modified to control the amount of power sent to a load [16].

<sup>3</sup>The Complementary metal-oxide-semiconductor (CMOS) sensor is used to capture images and convert them to digital format. It consists of numerous photodiodes (light-sensitive semiconductor), one for each pixel, which produce an electric current that varies depending on the intensity of light received [17].



(A) LED ring light.



(B) Power supply.

FIGURE 3.5: Lighting.

images, has a frame rate of 30 frames per second (FPS) and an exposure time of 33 milliseconds. For this project, and for the height of the camera above the conveyor belt (95 cm), the resolution of the images is  $0.64 \times 0.625$  mm/pixel. The lens is the Kowa model LM8JC1MS, with a focal length<sup>4</sup> of 8 mm and a variable focal ratio<sup>5</sup> of  $f/1.4$  to  $f/16$ . For video capture, an  $f/1.4$  and a focus distance of 0.35 meters were used, both of which are adjustable on the lens.



(A) CMOS sensor.



(B) Lens.

FIGURE 3.6: Camera.

<sup>4</sup>Distance between the optical center of the lens and the focus (point where the light rays converge) [18].

<sup>5</sup>The f-number, or focal ratio, expresses the diameter of the aperture of a lens in relative terms with respect to its focal length [19].

### 3.3.2 Software

This project is implemented in Python 3.9. For the computer vision part, multiple functions from the open source computer vision library (OpenCV) library will be used [20]. The libraries, Numpy and Pandas, are used very frequently throughout the program for computational operations and data storage. The Matplotlib library is used for data visualization. Scikit-learn library is used for data preprocessing<sup>6</sup>. Keras, a high-level neural network application programming interface (API), is used for neural network modeling with TensorFlow<sup>7</sup> as back-end<sup>8</sup>.

For the training of neural networks, Google Colab was used [22], a cloud service, based on Jupyter Notebooks<sup>9</sup>, which allows free use of Google graphics processing units (GPU). However, since the computational load of this project was not excessive (since the database is relatively small, with a size of 30 megabytes), the models could have been trained locally with a GPU, or even with a central processing unit (CPU).

To process the images captured by the camera in Python, it was necessary to download the UEye API (a file with a DLL<sup>10</sup> extension).

## 3.4 Computer Vision

The input to the program is video, so it has to be treated frame by frame. First, the size of the frame is adapted to the one captured by the camera (752x480 pixels). Then, an infinite while loop is created, in which the successive frames will be captured recurrently. To stop this while loop, a manual exit was established by pressing the letter *q* on the computer keyboard.

In the first phase of object detection, the successive frames captured by the camera will be treated independently. And then, in the object tracking phase,

---

<sup>6</sup>Is a preliminary step during the data mining process. It is any type of processing that is done with raw data to transform it into data that has formats that are easier to use.

<sup>7</sup>TensorFlow is an open source end-to-end platform for machine learning, allowing developers to easily compile and deploy applications powered by artificial intelligence [21].

<sup>8</sup>In the computer world, the back-end refers to any part of a website or software program that users do not see. It contrasts with the front-end, which refers to a program's or website's user interface.

<sup>9</sup>Jupyter Book is an open source project for building beautiful, publication-quality books and documents from computational material.

<sup>10</sup>The Dynamic-link library (DLL) extension refers to the files with executable code that are loaded on demand of a program by the operating system [23].

the frames will be treated with an overview, in order to be able to track the object in the video.

### 3.4.1 Object Detection

The camera used in this project captures color images (FIGURE 3.7), and in order to effectively detect objects, a binary image is needed, in which each pixel can only take two values, 0 (black) or 1 (white). To carry out this transformation, it is necessary to perform an intermediate step which is to convert the color image to gray scale (FIGURE 3.8), in which each pixel will have a single gray level (0 - 255). This first transformation is performed with the function *cvtColor*<sup>11</sup>.

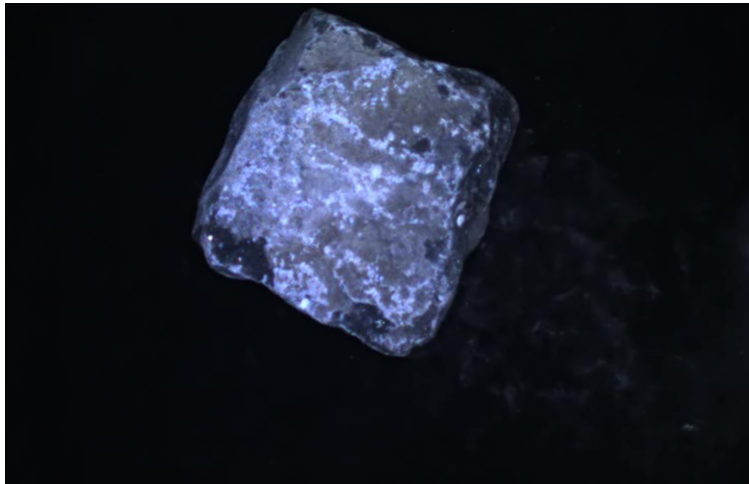


FIGURE 3.7: Color image.

Now it is necessary to perform a thresholding or binarization, i.e., that each pixel has only 2 possible values, 0 or 1. This step is very important because, with it, it will be possible to locate the rocks inside the conveyor belt. It must be taken into account that the rocks will arrive slightly dirty on the conveyor belt and that, with the time they will stain the conveyor belt. These stains can be a problem, as they can be mistaken for rocks. To minimize the occurrence of these stains, numerous binarization methods were explored, from the simplest to the most complex: global binarization, Otsu's method, adaptive binarization (explained in section 2.1.1), Sobel filters (explained in section 2.1.2) and Canny edge detector (explained in section 2.1.3).

The global, adaptive and Otsu binarization methods were not valid because when there was no rock in the frame, the program is much more sensitive to

---

<sup>11</sup>OpenCV function to transform a color image to a gray scale image

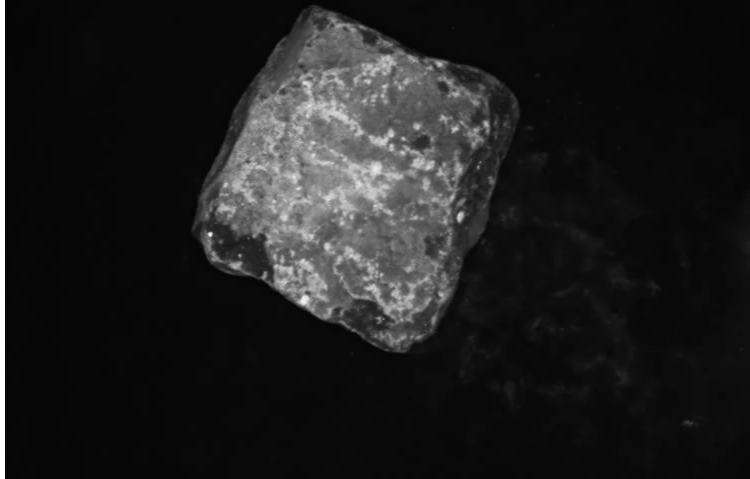


FIGURE 3.8: Gray scale Image.

detect stains on the conveyor belt as rocks. Therefore, these three methods were discarded. Finally, the Canny edge detector (FIGURE 3.9) was chosen, as it was notably more robust than the Sobel filter, for this particular case.

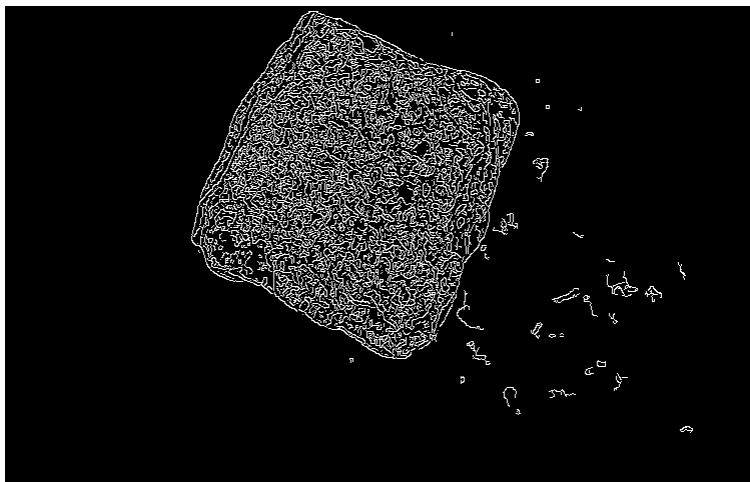


FIGURE 3.9: Edges detected by the Canny edge detector.

Once we have the edges of the rock as white pixels, we need the interior of the rock to be white as well. As rocks have a relief and a roughness, the Canny algorithm in the chosen configuration also detects inner parts of the rock. This is not a problem, on the contrary, because it will help us to fill the interior of the rock.

In order to carry out this task, a closing and a subsequent opening of the image (explained in section 2.1.4) is performed, with kernels with geometric shapes and sizes chosen by trial and error. The purpose of the closing is



to fill the gaps inside the rock (FIGURE 3.10). And the purpose of opening, is to remove the stains on the conveyor belt that may have been detected, however, larger stains are still going to be detected (FIGURE 3.11).



FIGURE 3.10: Image closed.



FIGURE 3.11: Image opened.

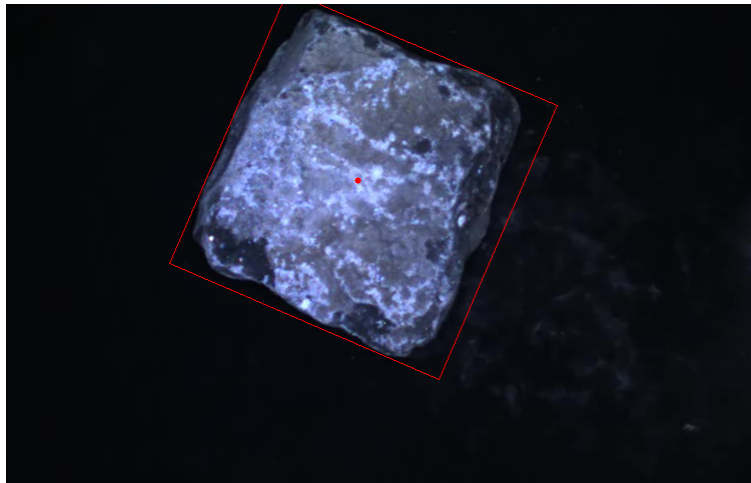
Once we have the image with the part of the rock in white and the conveyor belt in black, except for some spots, it is time to apply an algorithm to find the contours<sup>12</sup>. This algorithm will provide us with a list of the objects it detects in the image, as well as a series of very useful characteristics of these (size, position, centroid<sup>13</sup>...).

<sup>12</sup>The curve joining all the continuous points (along the object boundary), having same intensity. The contours are a useful tool for object detection and shape analysis [24].

<sup>13</sup>The centroid or geometric center of a plane figure, is the arithmetic mean position of all the points in the figure [25].

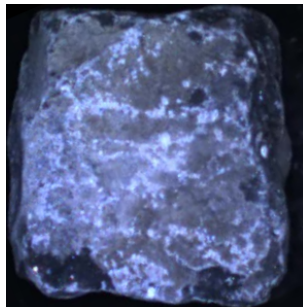
To completely eliminate the problem of stains on the tape, detected as rocks, all objects whose area is less than 10,000 pixels are eliminated, since there will be no rocks with an area smaller than this.

Now with only the rocks in the image, you get the bounding boxes of these. There is a problem, and it is that the rocks may not be aligned with the frame that the camera captures, and it needs to be aligned to obtain the image of the rock and to be able to classify it. So, to solve it, the minimum area bounding box is obtained (FIGURE 3.12) and with this, applying a perspective transformation technique [26], the aligned rock is obtained (FIGURE 3.13).



---

FIGURE 3.12: Detected rock.



---

FIGURE 3.13: Rotated rock.

Finally, the image is reduced by 20% in both width and height, to make sure that only what appears in the image belongs to the rock (FIGURE 3.14). The main reason for this step is that, although most rocks are square or rectangular, there are some rocks that are not (FIGURE 3.2c) and if this was not done, the captured image (which is necessarily square or rectangular) would show

a part of the conveyor belt, which would alter the characteristics of the rock at the time of classification.

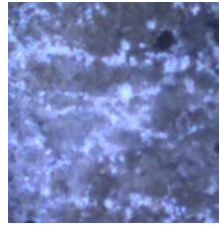


FIGURE 3.14: Rock reduced in size.

### 3.4.2 Object Tracking

So far, the video frames have been treated independently. But it is necessary to give a global approach to the set of frames that the camera will capture, because the same rock will appear in a large number of consecutive frames and only requires classification once. Classifying each rock only once, in one of the frames, will help greatly reduce the execution time of the algorithm.

It should also be taken into account that two or more rocks can appear in the same frame at the same time, if they have been loaded on the conveyor belt very close together. Therefore, the program has to be able to keep track of multiple objects at the same time.

Object tracking is a very useful technique in video processing, which is used to identify the trajectories of objects in a video. There are many object tracking techniques, but we will use one based on the Euclidean distance [27], which is the distance between two points in a Euclidean space, which is deduced by applying the Pythagorean theorem<sup>14</sup>. For the two-dimensional space in which work is done, the distance between two points  $P_1$  and  $P_2$ , of Cartesian coordinates  $(x_1, y_1)$  and  $(x_2, y_2)$  is calculated with the EQUATION 3.1.

$$d_E(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (3.1)$$

The input for this object tracking algorithm will be the coordinates of the centroids of the detected objects in each frame. Then, from one frame to the next, the distances between the centroids of the first frame with respect to the

<sup>14</sup>It states that the area of the square whose side is the hypotenuse, is equal to the sum of the areas of the squares on the other two sides ( $x^2 + y^2 = h^2$ ) [28].

second will be calculated. If this distance is less than 25 pixels, it will be the same object, otherwise it will be a new object and it will be assigned a new identifier number [29].

An example of how object tracking works is shown in the FIGURE 3.15.

Each rock will enter the opaque box through the opening and exit at the opposite side. This means that when the rock is entering the camera will only capture a part of the whole rock, and the same happens when it leaves the box. Therefore the instant, at which the image of each rock will be captured for classification, will be the one at which the centroid of the rock is in the center of the frame. But it cannot assume that all the centroids of the rocks will be found in one of the frames in which they appear, in the middle pixel<sup>15</sup>. Therefore, it is necessary to determine a range of middle pixels in which it is certain that the centroid will appear.

To calculate this middle range, it is necessary to know the number of pixels that a centroid moves from one frame to the next. To do this, starting from data such as the frame rate ( $30 \frac{\text{frames}}{\text{second}}$ ), the conveyor belt speed ( $0.4 \frac{\text{meters}}{\text{second}}$ ) and the resolution ( $0.64 \frac{\text{millimeters}}{\text{pixel}} = 0.00064 \frac{\text{meters}}{\text{pixel}}$ ), first the frame rate and resolution are inverted (EQUATION 3.2), to facilitate unit conversion, and finally we operate (EQUATION 3.3).

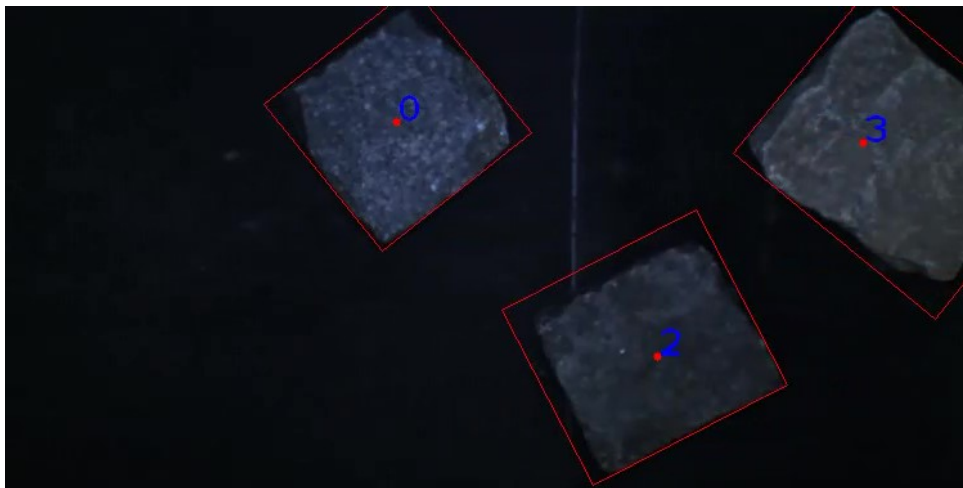
$$\frac{1}{\text{FrameRate}} * \text{Speed} * \frac{1}{\text{Resolution}} \quad (3.2)$$

$$\frac{1\text{second}}{30\text{frames}} * \frac{0.4\text{meters}}{1\text{second}} * \frac{1\text{pixel}}{0.00064\text{meters}} = 20.83 \frac{\text{pixels}}{\text{frame}} \approx 21 \frac{\text{pixels}}{\text{frame}} \quad (3.3)$$

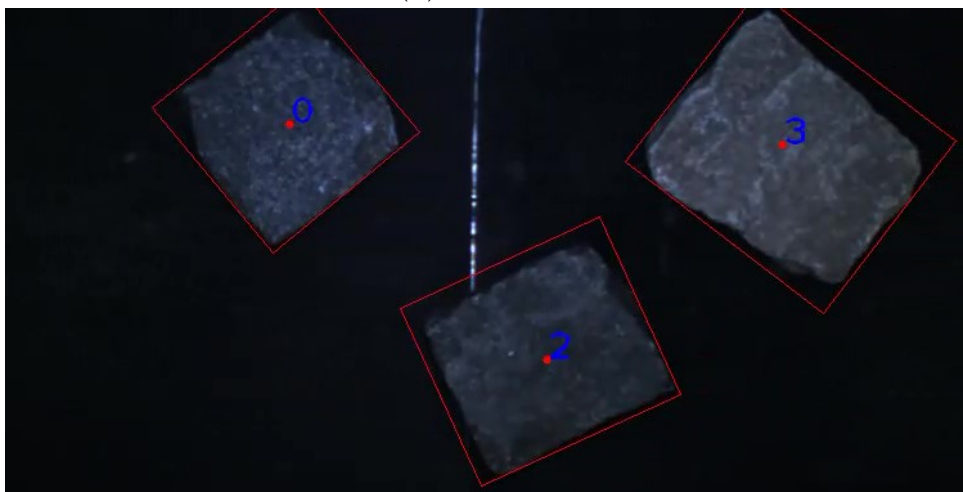
The centroid of a rock will travel 21 pixels, from one frame to the one immediately following it. So with this in mind, an intermediate stripe (FIGURE 3.16) is set in the frame of 64 pixels wide, so that at least each centroid is detected 3 times.

If the rock was not captured when it is in the middle stripe of the frame, there would be a high probability that it would not appear in its totality (FIGURE 3.17), which would give problems of classification since probably, all the characteristics of the rocks would not be taken into account.

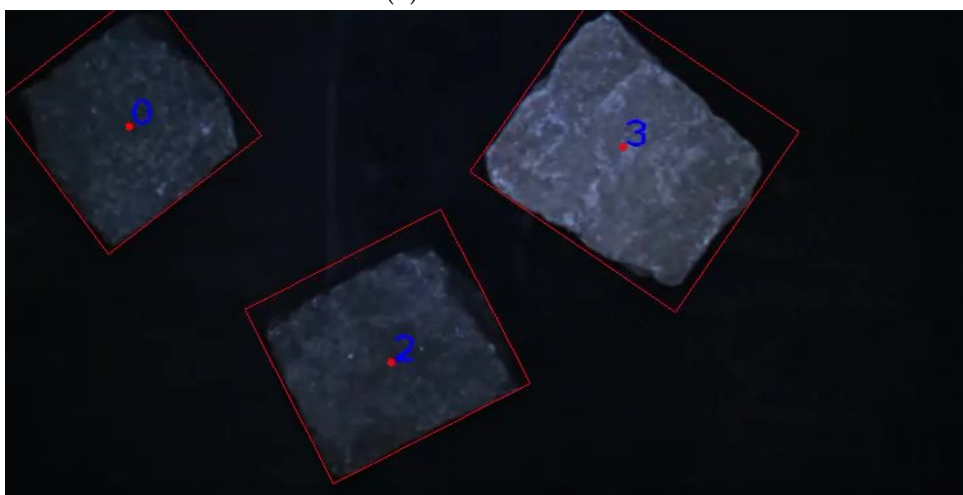
<sup>15</sup>The middle pixel, for the width of the frame (752 pixels) would be half, i.e., pixel 376.



(A) First frame.



(B) Second frame.

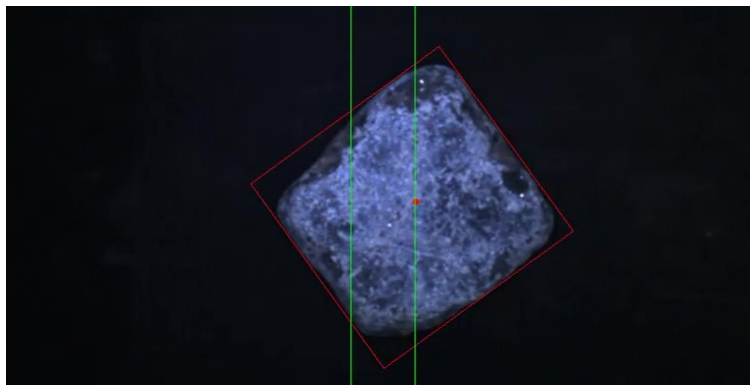


(C) Third frame.

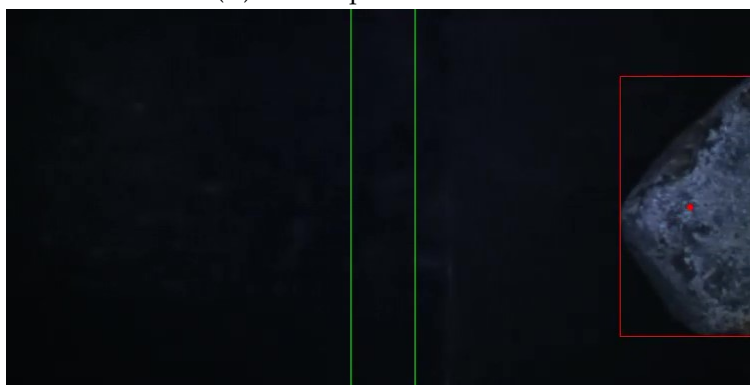
FIGURE 3.15: Object tracking example, in 3 frames separated by 2 seconds.



FIGURE 3.16: Intermediate frame stripe (space between the green lines), in which the rock classification will be performed.



(A) Total capture of the rock.



(B) Partial capture of the rock.

FIGURE 3.17: Comparison between capturing the entire rock in the center of the frame, and partially capturing the rock on the right of the frame

### 3.5 Classification

Once the rock is located on the conveyor belt, it has to be classified. The information available for classification is the image obtained after applying the methods described in section 3.4. This image is in color, and its dimensions are variable depending on the size of the rock, being the smallest ones in the order of 50x50 pixels, up to 260x260 the largest ones.

Various classification methods were explored, starting from the simplest, to the most complex and novel. The former were simple classification methods based on average color, both in the RGB and hue, saturation, value (HSV) color models. Then, an unsupervised learning method<sup>16</sup> was tested, with the K-Means algorithm<sup>17</sup>, but it was very slow. The predictive power of these two methods was very weak, being below 50 %, therefore, it was decided to use the ANN for classification.

### 3.5.1 Image Acquisition

The first step for training an ANN is to have a sufficiently large database, so that it can learn to classify rocks. Since no database existed, it had to be created.

To create it, images of the rocks had to be taken in the same lighting conditions that would be used when the system was put into production, otherwise the model would not work correctly. The opaque box, camera and lighting mentioned in the section 3.3.1 were used. It is worth mentioning that the lighting was kept constant while all images were captured.

To speed up and facilitate the process of capturing images of the rocks, a program was created, adding a small modification to the one already created for the rock detection part (3.4.1). The input to this program is a video, which is shown on the screen when it is executed. The operation is as follows, the user has to press the *s* key, so that an image is automatically saved when a rock appears in the video. These images are saved with number names (1, 2, 3...) and with portable network graphics (PNG) extension. In addition, another modification was also added, which is that if the user pressed the *l* key, images are saved with number names, but also including the prefix *S* (S1, S2, S3...).

This last modification, was introduced to be able to distinguish between the visible face of the rocks (saved by pressing *s*) and the rest of the faces (saved by pressing *l*), in order to study the possibility of fulfilling the secondary objective (section 1.3), which consists in classifying regardless of the rock face captured by the camera.

---

<sup>16</sup>Unsupervised learning (UL) is a type of algorithm that learns patterns from untagged data [30].

<sup>17</sup>K-means clustering is a method of classification, that aims to partition  $n$  observations into  $k$  clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid) [31].

In order to make the videos that the previous program works with, the whole equipment had to be moved to the Nijst company for one day (FIGURE 3.18), since there was a large amount of rocks there. With the help of an expert worker, representative samples of the 3 types of rocks varying in size, color, shape and texture were chosen. Videos of approximately 135 rocks were captured. A total of 3 videos were recorded, one for each rock type.



FIGURE 3.18: Nijst equipment setup for database creation, including opaque box, lighting, camera, power supply and computer.

Before being recorded, the rocks were washed with pressurized water and dried (FIGURE 3.19), as this same procedure will be carried out with the system in production. Washing the rocks helps to eliminate dust and dirt that they may have, in order to capture their real characteristics with the camera, eliminating the noise generated by the dirt; and drying them helps to avoid reflections created by the water that may have remained on their surface.

The videos consist of a person who puts a rock in the opaque box, takes his hands out (so that these do not cause interference with the rock detection algorithm (3.4.1)), waits 1 second, and puts his hands in again to rotate each rock 5 times, so that the 6 faces of the rock appear in the video. And this same procedure was repeated with all the rocks.

After processing these videos with the program, 796 images of rocks were obtained, with the following composition: 188 of blue, 375 of gres and 233 of porphyry. More rocks of the gres type were recorded, since it was the type with the most variations of colors and textures.





FIGURE 3.19: Rock washing and drying process.

### 3.5.2 Feature Extraction

Once the database was built, two ways of solving the classification problem were studied: Convolutional Neural Network (CNN) and ANN. CNN's are a special type of ANN, designed to recognize visual patterns directly from the pixels of an image with almost no preprocessing, while ANN's are the basic architecture of neural networks.

In order not to increase the computational load of the program too much, it was decided to use an ANN, since the inputs to a CNN are all the pixels of an image (on average the rock images were 150x150, which means 22500 inputs), while the inputs to an ANN will be an infinitely smaller number, depending on the features we want to use.

There are many features that can be obtained from an image: shapes, color, textures... However, not all of these features provide relevant information for the classification. Therefore, it is important to properly choose a limited number of characteristics, which help us to classify. It was decided to choose features of the color histograms<sup>18</sup> of the image, because it has been shown that training an ANN with those features, provides better classification accuracy [32].

---

<sup>18</sup>The histogram of an image, is a graphic representation of the distribution of the different tones of an image.

Color histograms indicate the number of pixels in the image that have a certain level of gray (0 - 255). With the help of the function *calcHist*<sup>19</sup> 3 histograms will be calculated, corresponding to the colors of the RGB model (red , green and blue) (FIGURE 3.20).

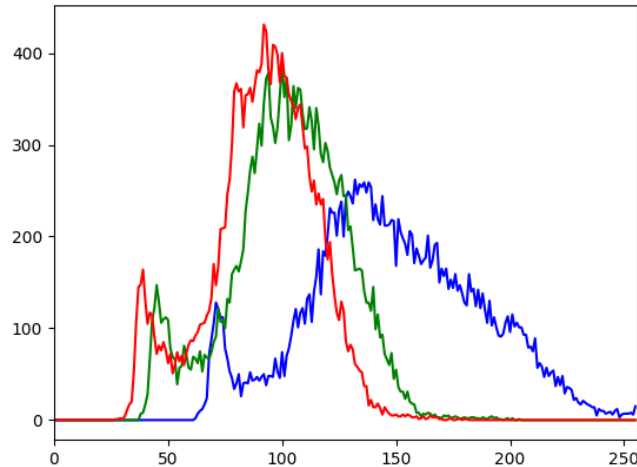


FIGURE 3.20: Image histogram of the 3 colors of the RGB model.

For example, if we have an image of size  $i * j$ , each pixel will have a gray level or intensity  $f$ , which will vary between 0 and 255 (from 0 to  $L - 1$ ). Therefore, the histogram  $H(f)$  will be equal to the number of pixels that have that particular gray level.

Once the histograms have been calculated, it is necessary to obtain from them features to describe their shape and to be able to differentiate between the histograms of the 3 types of cobblestones. Three features will be extracted from each histogram: weighted mean, kurtosis and skewness. Therefore, as 3 histograms (R, G and B) are obtained from each image, in total we will obtain 9 features per image (TABLE 3.1) with which it will be possible to train our network for classification.

The weighted average of the intensities of each histogram is calculated in such a way that greater weight is given to the higher intensities. This is calculated with the formula 3.4:

<sup>19</sup>The function *calcHist* belongs to the OpenCV library, and with it the histogram of a color image can be calculated.

	Weighted Mean	Skewness	Kurtosis
<b>R</b>	$\overline{f_{WR}}$	$S_R$	$k_R$
<b>G</b>	$\overline{f_{WG}}$	$S_G$	$k_G$
<b>B</b>	$\overline{f_{WB}}$	$S_B$	$k_B$

TABLE 3.1: Features extracted from each rock image.

$$\overline{f_W} = \frac{\sum_{f=0}^{L-1} f * H(f)}{\sum_{f=0}^{L-1} f} \quad (3.4)$$

The skewness is a measure of the asymmetry of the histogram with respect to its mean. This measure can be zero, negative or positive. A zero-valued skewness (FIGURE 3.21a), means that the tails on both sides of the mean are balanced. Negative skewness (FIGURE 3.21b), when the left tail is longer, and therefore, the mass of the histogram is concentrated on the right. In contrast, if the right tail is longer, the skewness is positive (FIGURE 3.21c), and the histogram mass is concentrated on the left [33].

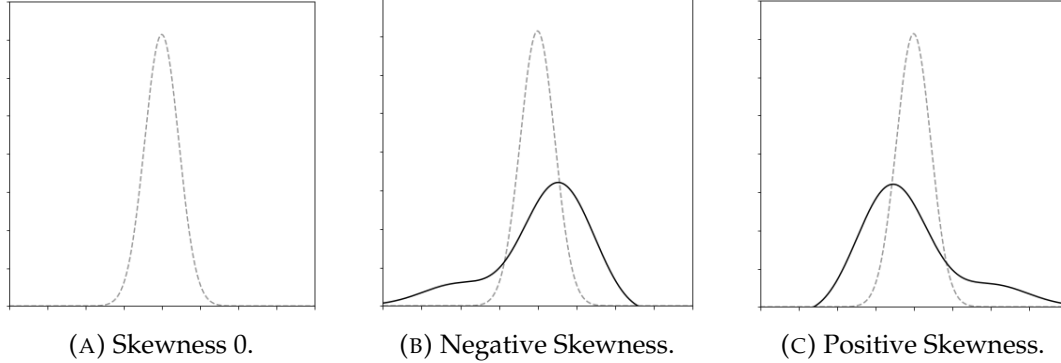


FIGURE 3.21: Skewness types.

To calculate the skewness, we need to first calculate the mean (EQUATION 3.5) and variance (EQUATION 3.6) of the histogram.

$$\bar{f} = \frac{\sum_{f=0}^{L-1} f * H(f)}{i * j} \quad (3.5)$$

$$v = \frac{\sum_{f=0}^{L-1} (f - \bar{f})^2 * H(f)}{i * j} \quad (3.6)$$

Once the mean and variance are obtained, we can calculate the skewness with the EQUATION 3.7.

$$S = \frac{\frac{1}{i*j} * \left( \sum_{f=0}^{L-1} (f - \bar{f})^3 * H(f) \right)}{(\sqrt{v})^3} \quad (3.7)$$

Kurtosis is a measure of shape that quantifies how steep or flattened the histogram is. This coefficient indicates the amount of data that is close to the mean, so the higher the kurtosis, the steeper the shape of the histogram [33].

This measure can be zero, negative or positive. A kurtosis with value 0 (FIGURE 3.22a), means that there is a normal concentration of values around its mean. The kurtosis is negative (FIGURE 3.22b), when there is a low concentration of data around the mean, and therefore, the shape of the histogram is flattened. In contrast, if there is a high concentration of data around the mean, the kurtosis is positive (FIGURE 3.22c), and the shape of the histogram is steep [33].

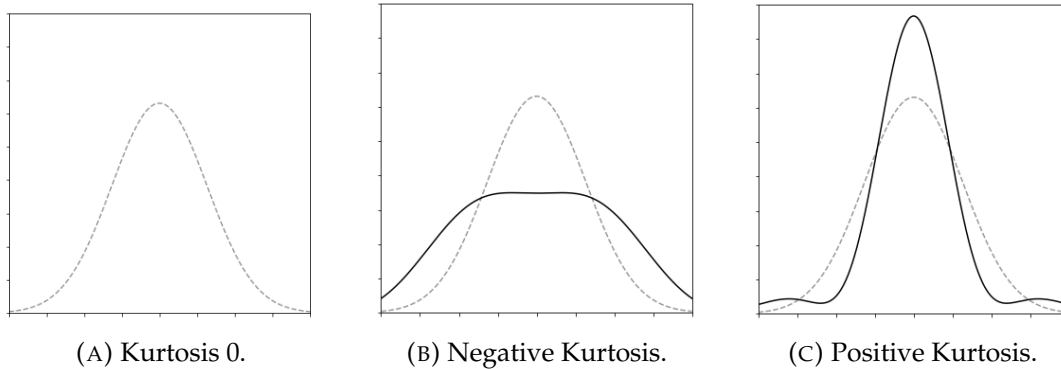


FIGURE 3.22: Kurtosis types.

To calculate the kurtosis (EQUATION 3.8), we need to previously calculate the mean (EQUATION 3.5) and variance (EQUATION 3.6) of the histogram.

$$k = \frac{\frac{1}{i*j} * \left( \sum_{f=0}^{L-1} (f - \bar{f})^4 * H(f) \right)}{(\sqrt{v})^4} \quad (3.8)$$

To generate the table with the 9 features of each rock and its type, necessary to train the ANN, a program was created from the image database. The database was composed of 3 subfolders, one for each rock type, in which all the images were stored. The output of the program was an Excel file with a

table with 10 columns and 796 rows (one row for each rock image). The first column for the rock type coded<sup>20</sup> and the remaining columns corresponding to the 9 extracted features (TABLE A.1 in APPENDIX A).

### 3.5.3 Dataset Creation

Once we have the 9 features of each image and its corresponding type, we have to prepare these data to serve as input to the neural network.

The first thing to do with the dataset is to represent its boxplot (FIGURE 3.23). This provides statistical information such as mean, standard deviation, minimum, maximum, first quartile (25% of the data is below this point), second quartile (50% of the data is below this point) and third quartile (75% of the data is below this point). Thanks to this, it can be seen how the value of each of the 9 features chosen to train the model varies. As it can be seen (FIGURE 3.23), individually, the features are quite variable, especially the weighted means. If this variability did not exist, it would be more difficult to achieve high prediction percentages.

Secondly, we are going to perform a test to see if it is a balanced dataset or not, since it has been shown the extreme importance of having a balanced dataset for classification<sup>21</sup>. To do this, Shannon entropy<sup>22</sup> is used as a balancing measure. For a dataset with  $n$  instances and  $k$  classes of size  $c_i$ , it computes:

$$H = - \sum_{i=1}^k \frac{c_i}{n} \log \frac{c_i}{n} \quad (3.9)$$

This entropy (EQUATION 3.9), will be equal to 0 when there is only 1 class ( $k$ ), in other words, it will tend to 0 when the dataset is not balanced; and it will be equal to  $\log k$ , when all classes have the same number of instances. Thus, when applying the EQUATION 3.10 we will get a 0 for unbalanced datasets or a 1 for balanced ones [36].

<sup>20</sup>The reason why the rock type column has to be coded and have the values [0, 1, 2] instead of [Blue, Gres, Porphyry], is because the neural networks are trained with numerical data.

<sup>21</sup>When we have an unbalanced dataset, in training, the network tends to minimize the loss, learning to classify the high occurrence classes very well and ignoring the classes that hardly appear [34].

<sup>22</sup>Shannon entropy measures the uncertainty of an information source [35].

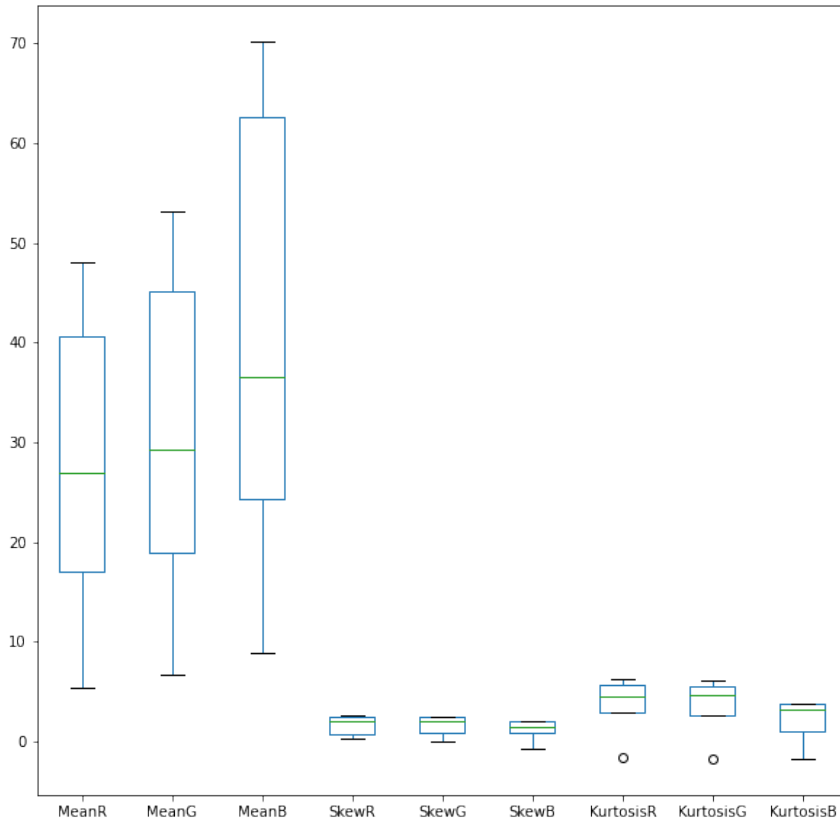


FIGURE 3.23: Dataset boxplot.

$$Balance = \frac{H}{\log k} = \frac{-\sum_{i=1}^k \frac{c_i}{n} \log \frac{c_i}{n}}{\log k} \quad (3.10)$$

The EQUATION 3.10 was applied to the dataset and a result of 0.9603 was obtained, therefore it is concluded that the dataset is balanced and no additional preprocessing is necessary.

Once these two statistical tests have been performed, it is necessary to divide the dataset into 3 groups: training, validation and test. This division of the dataset is usual when we want to train neural networks. The first group, as its name indicates, is used to train the model; the second is for validation, to help choose between different models; and the last is for testing, to measure the accuracy with which the model classifies.

Typically, 60% of all data will be for training, 20% for validation and the remaining 20% for testing [37]; however, since unlimited data are not available for training, this amount was increased to approximately 68%, thereby reducing the validation group to 12%.

Regarding the above data splitting, two important aspects must be taken into account: the splitting is random and stratified. Random, because the data are divided by chance, and stratified, because the same proportion of samples from each class is maintained in the 3 groups into which the data are divided (24% blue, 47% gres and 29% porphyry).

The ANN is going to be trained, providing the 9 features extracted from the images as input variables, and the coded rock types as output variables. As we had these input and output variables together in the same table, they were separated into two different matrices, the feature matrix and the target or label matrix. This separation was done for the 3 subsets of data (training, validation and test).

### 3.5.4 Artificial Neural Network

Once the data preprocessing (3.5.3) is done, an ANN will be trained to classify the rocks by type from the 9 input features.

For the neural network design, a combination of hyperparameters<sup>23</sup> or hidden layers had to be chosen to provide acceptable classification accuracy. Since the database is small (30 megabytes) and the number of inputs to the network is small, the training time never exceeded 20 seconds on average. Therefore, a manual search for hyperparameters was performed, and finally a configuration consisting of 6 layers was defined:

- Fully connected<sup>24</sup> **initial layer** (9 nodes).
- Fully connected **hidden layer** (64 nodes) with sigmoid activation<sup>24</sup>.
- Fully connected **hidden layer** (32 nodes) with sigmoid activation.
- Fully connected **hidden layer** (16 nodes) with sigmoid activation.
- Fully connected **hidden layer** (8 nodes) with sigmoid activation.
- **Dropout layer**<sup>25</sup> (10%).
- Fully connected **output layer** (3 nodes) with softmax activation<sup>26</sup>.

---

<sup>23</sup>Parameters which are used to control the learning process of a neural network.

<sup>24</sup>Explained in section 2.2.

<sup>25</sup>This layer randomly sets the input units to 0 with a frequency (10% in this case) at each step, during the training time, helping to avoid over-fitting (modeling error that occurs when a function corresponds too closely to a particular data-set).

<sup>26</sup>Activation function used to normalize the output of a network to a probability distribution over predicted output classes.

The input layer will have 9 nodes, one per feature, and the final layer has 3 nodes, one for each rock class.

In the FIGURE 3.24 shows a grouped representation of the network, while in the FIGURE 3.25 shows the actual structure of the network, with all of its nodes.

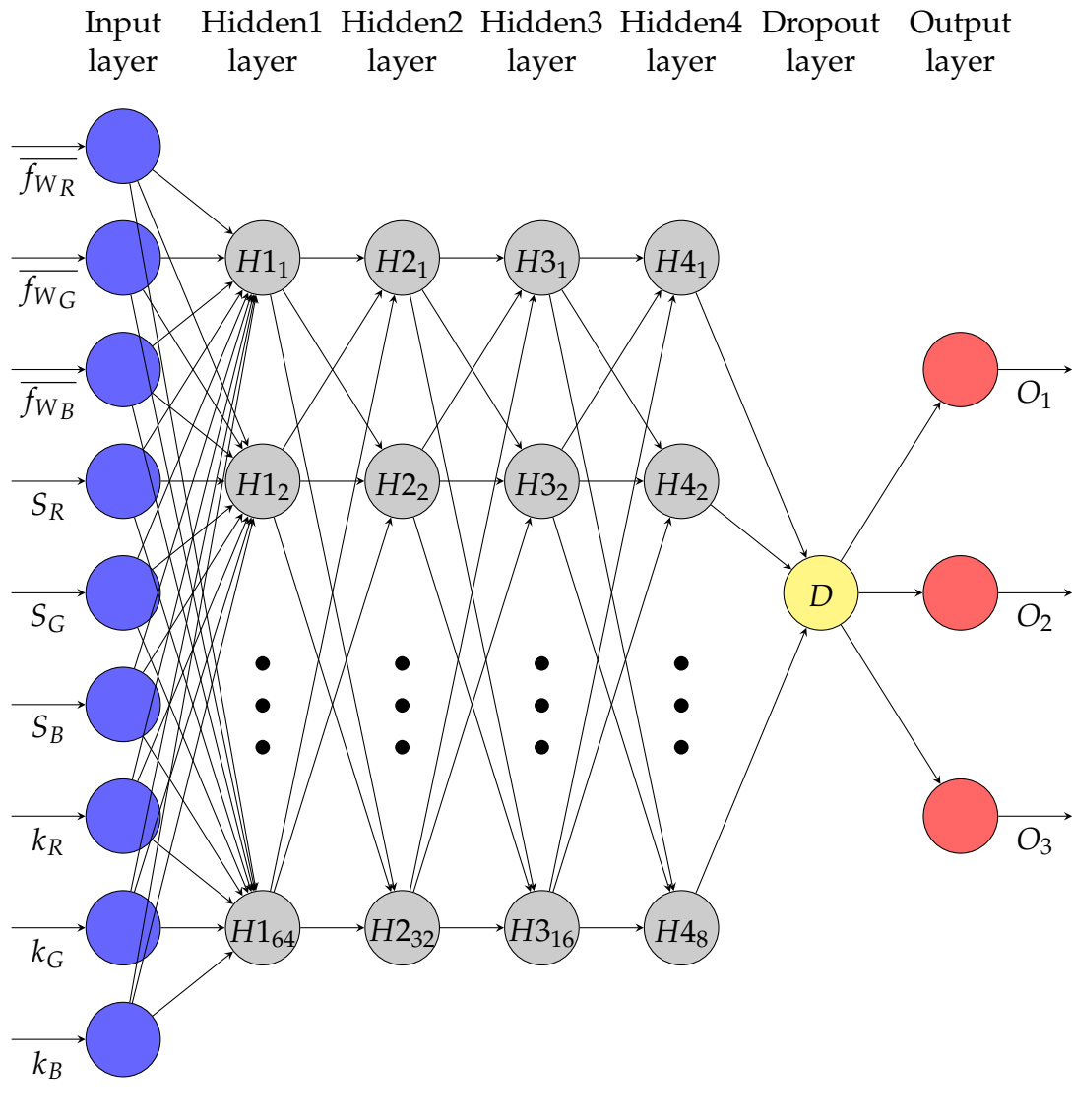


FIGURE 3.24: Grouped representation of the neural network.

Once the model was built, the compilation parameters were chosen:

- Adam, as an optimization algorithm (explained in section 2.2).
- Sparse categorical cross entropy, as a loss function (explained in section 2.2).



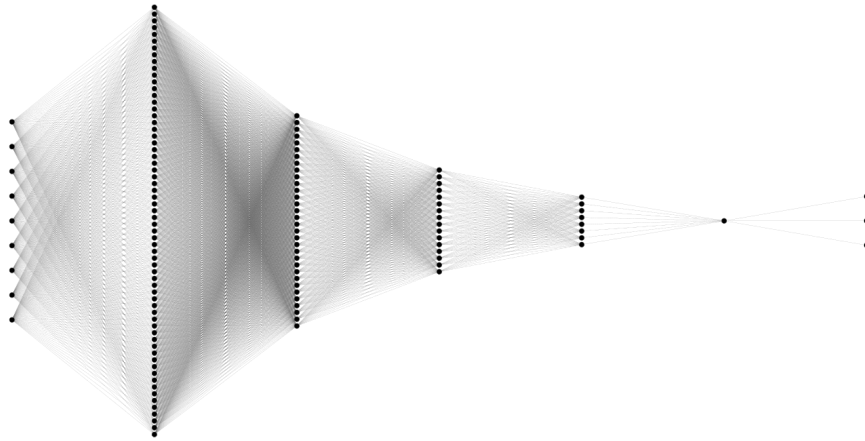


FIGURE 3.25: Real representation of the neural network.

- 300 epochs<sup>27</sup>.

Once the ANN was trained, it was saved in the H5 format<sup>28</sup>. It should be mentioned that the network had 3.411 hyperparameters.

<sup>27</sup>An epoch is one complete cycle through the full training dataset.

<sup>28</sup>Hierarchical data format 5 (HDF5) (.h5) is a file format for storing structured data, it is not a model by itself. Keras saves the models in this format as it can easily store the weights and model configuration in a single file [38].

## Chapter 4

# Results and Implementation

### 4.1 Classification Accuracy

The final trained model obtained an accuracy of 95% in the test<sup>1</sup>.

In the FIGURE 4.1, the confusion matrix<sup>2</sup> of the results of the test data can be seen.

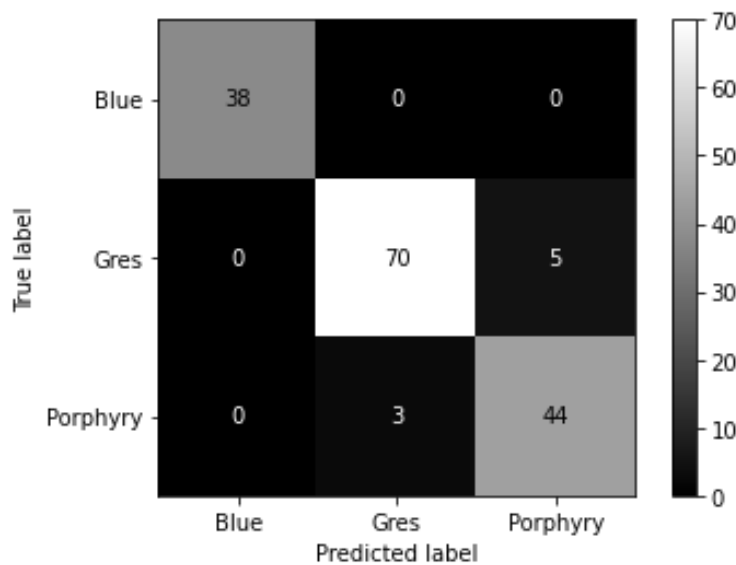


FIGURE 4.1: Confusion matrix of the test dataset.

Since the number of images of each type of rock in the test sample is not the same, the confusion matrix is normalized and expressed as a percentage. As we observe in the FIGURE 4.2, the ANN classifies with 100% accuracy the blue type rocks, with 93% accuracy the gres type rocks, and with 94% accuracy the porphyry type rocks.

<sup>1</sup>With rock images from a dataset with which the ANN has not been trained.

<sup>2</sup>Tool to visualize the performance of an algorithm used in supervised learning.

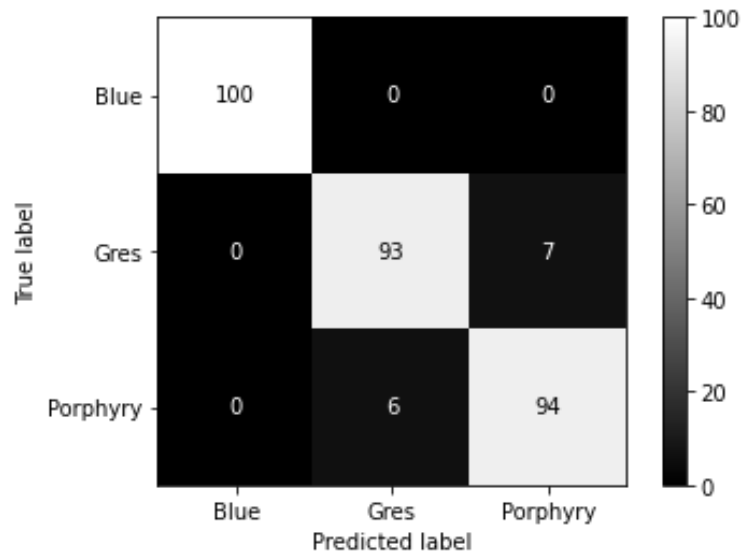


FIGURE 4.2: Normalized confusion matrix of the test dataset (expressed in percentages).

## 4.2 Implementation of the System

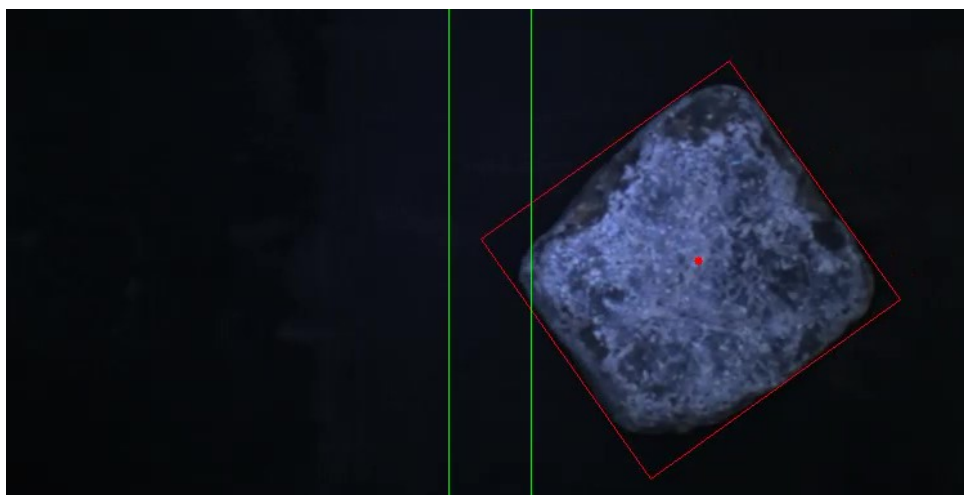
Once all the stages in which the project is divided (object detection and tracking (3.4), and classification (3.5)) work separately, it is time to put them all together in the same program.

In this way, the video captured by the camera will be the input to the program and in it, the rocks will appear on the conveyor belt. These rocks will be detected, tracked and, when their centroid is within the intermediate green stripe, they will be classified. As can be seen in the FIGURE 4.3, in the upper left corner of the frame, the ANN prediction will appear, as well as the probability (in %) that this prediction is correct.

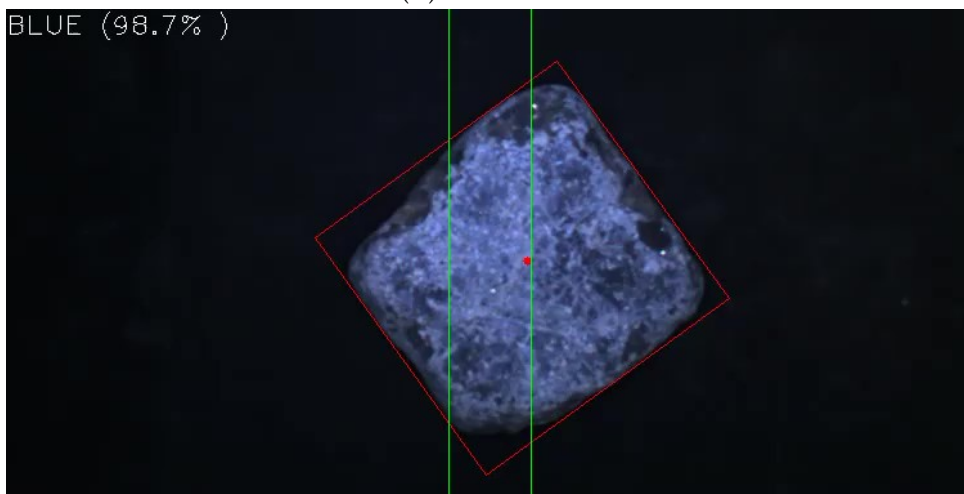
In the APPENDIX A, there are examples of global implementation of the system for rocks of gres type (FIGURE A.1) and porphyry (FIGURE A.2).

In addition to the overall system working correctly, as the ANN was trained with images of the 6 faces of each rock, the secondary objective of classifying regardless of the rock face captured by the camera (1.3) has also been fulfilled. As can be seen in the FIGURE 4.4, the system recognizes a rock of type blue, regardless of the face shown.

In the APPENDIX A, examples of classification regardless of the face are found for rocks of type gres (FIGURE A.3) and porphyry (FIGURE A.4).

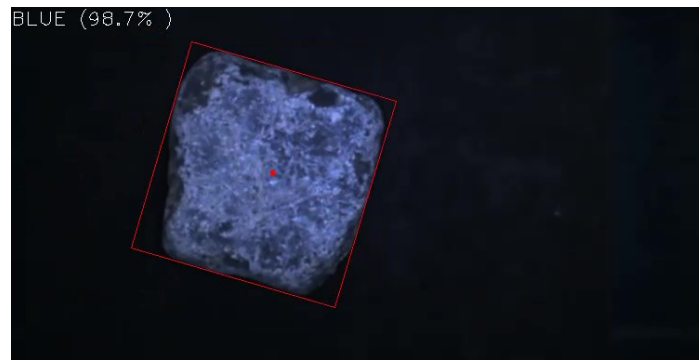


(A) First frame.



(B) Second frame.

FIGURE 4.3: Example of global implementation of the system for a rock of the blue type, in 2 consecutive frames separated by 2 seconds.



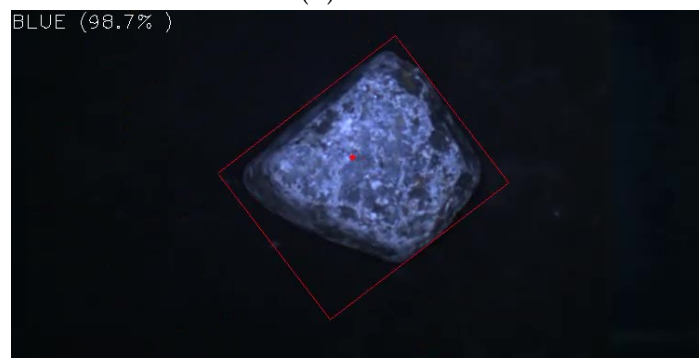
(A) Face 1.



(B) Face 2.



(C) Face 3.



(D) Face 4.

FIGURE 4.4: Examples of classification regardless of the face captured by the camera for a rock of type blue. As can be seen, the ANN prediction is always blue, but the probability (in %) that this prediction is correct may vary, since different faces are present.

## Chapter 5

# Conclusion

### 5.1 Budget

In this section, an estimation of the cost of this project is made based on the value of the software development and the price of the material means necessary to implement the device that will perform the classification of the rocks automatically.

For this purpose, it is determined that the basic human team for the development of the software must be composed of an artificial intelligence (AI) developer and a programmer, who will assist him in the creation of the program. In addition, it is considered that the advice of a senior engineer expert in PLC programming is necessary, who will provide technical support and supervise the execution. The cost of the human resources, based on average salaries, is calculated in TABLE 5.1.

	€ / Month	€ / Hour
<b>Junior AI developer</b>	4.260 €	26,63 €
<b>Junior programmer</b>	3.180 €	19,88 €
<b>Senior PLC programmer</b>	5.748 €	35,93 €

TABLE 5.1: Salaries. The sources from which the data in the first column (€ / Month) have been obtained are as follows: junior AI developer [39], junior programmer [40], and senior PLC programmer [41].

As for the equipment and materials necessary for the development of the device, their cost is estimated in TABLE 5.2.

Finally, the TABLE 5.3 breaks down the project into three phases or stages to arrive at the total budget, which amounts to 19,914.25 €.

	Cost
<b>Ring light (LDR2-90SW2)</b>	2.756 €
<b>Power supply (PD2-3024-2)</b>	110 €
<b>Camera (UI-1225-LE-C)</b>	165 €
<b>Lens (LM8JC1MS)</b>	207 €

TABLE 5.2: Material equipment. Prices have been obtained from the following sources: ring light [42], power supply [43], camera [44], and lens [45].

	Hours	Cost
<b>Information gathering and preliminary studies</b>		
Junior AI developer	150	3.993,75 €
Junior programmer	100	1987,50 €
Senior PLC programmer	20	718,50 €
<b>Technical development and code generation</b>		
Junior AI developer	120	3.195,00 €
Junior programmer	170	3.378,75 €
Senior PLC programmer	25	898,13 €
<b>Test performance</b>		
Junior AI developer	50	1.331,25 €
Junior programmer	50	993,75 €
Senior PLC programmer	5	179,63 €
<b>Material costs</b>		3.238,00 €
<b>Total budget</b>		<b>19.914,25 €</b>

TABLE 5.3: Stages for budget. For each of the 3 stages, junior AI developer, junior programmer and senior PLC programmer hours were required. The cost of each of these is obtained by multiplying the hours by the €/Hour of the TABLE 5.1. Finally, the total budget is obtained by adding the material cost (TABLE 5.2).

## 5.2 Profitability Study

The profitability study aims to demonstrate the economic interest, for the Nijst company, of facing the cost of the development of this project. It will be obtained by estimating the difference between the cost of the manual sorting activity with two operators (currently performed) and the automatic classification proposed in this project.

The total cost budgeted in section 5.1, amounting to 19,914.25 €, is considered as the initial investment. The savings generated annually by avoiding the labor costs of 2 operators are taken as savings. Each operator, has a remuneration of 40  $\frac{\text{euros}}{\text{hour}}$  and works 8  $\frac{\text{hours}}{\text{day}}$ , 200  $\frac{\text{days}}{\text{year}}$ <sup>1</sup>. Therefore, the total annual savings are obtained with the EQUATION 5.1.

$$\cancel{2\text{operators}} * \frac{40\text{euros}}{\cancel{\text{hour}} * \cancel{\text{operator}}} * \frac{8\cancel{\text{hours}}}{\cancel{\text{day}}} * \frac{200\cancel{\text{days}}}{\text{year}} = 128.000 \frac{\text{euros}}{\text{year}} \quad (5.1)$$

To simplify the profitability calculation, it is considered that the useful life of this project is 5 years, and that the residual value at the end of these years is zero, assuming that this system could become obsolete after this period, due to possible technical advances in this field.

The TABLE 5.4, will be used to calculate indicators of the profitability of the investment.

<i>t</i>	Investment (I)	Savings	Balance (V)
<b>Year 0</b>	- 19.914,25 €		- 19.914,25 €
<b>Year 1</b>		128.000,00 €	128.000,00 €
<b>Year 2</b>		128.000,00 €	128.000,00 €
<b>Year 3</b>		128.000,00 €	128.000,00 €
<b>Year 4</b>		128.000,00 €	128.000,00 €
<b>Year 5</b>		128.000,00 €	128.000,00 €
<b>Total</b>	- 19.914,25 €	640.000,00 €	620.085,75 €

TABLE 5.4: Table for calculating profitability indicators. Evolution of savings and total balance in the first 5 years.

<sup>1</sup>Data provided by the company Nijst



After 5 years and a annual discount rate ( $k$ ) of 10 %, the net present value (NPV), corresponding to the value of the net cash flows (income - expenses) originated by the initial investment investment, will be **423.005,87 €** ( EQUATION 5.2 ).

$$\text{NPV} = \sum_{t=1}^n \frac{V_t}{(1+k)^t} - I_0 \quad (5.2)$$

The internal rate of return (IRR) which is a method of calculating an investment's rate of return, will be **642,73 %** ( EQUATION 5.3 ).

$$\text{NPV} = \sum_{t=0}^n \frac{V_t}{(1+\text{IRR})^t} = 0 \quad (5.3)$$

The profitability index (PI), a ratio of payoff to investment of a project, used to quantify the amount of value created per unit of investment, is **22,24** ( EQUATION 5.4 ).

$$\text{PI} = 1 + \frac{\text{NPV}}{|I_0|} \quad (5.4)$$

And finally, the **project payback** will be **less than 2 months** (EQUATION 5.5).

$$\frac{1 \text{ year}}{128.000 \text{ euros}} * \frac{12 \text{ months}}{1 \text{ year}} * 19.914,25 \text{ euros} = 1,86 \text{ months} \quad (5.5)$$

In view of these indicators, it can be concluded that the project is extraordinarily profitable.

In addition to reducing costs, the automatic sorting system will **increase production speed by 265 %**, from 15.3 to **40.6  $\frac{\text{tons}^2}{\text{hour}}$** .

### 5.3 Summary

In view of the high classification accuracy (4.1), the correct global performance of the system (4.2), and the extraordinary profitability (5.2), this project has been a remarkable success, heralding a promising future for artificial neural networks for classification tasks.

---

<sup>2</sup>Estimate obtained from: average weight of a rock (11.72 kg), speed of the conveyer belt ( $0.4 \frac{m}{s}$ ) and rocks present in each meter tape ( $\sim 2.4$ ).

Although it is true that the system works correctly, in order for it to continue to do so when the system is in real production, the same conditions of illumination, resolution, distance from camera to rock, that have been used during the development of the project, will have to be rigorously respected.

## 5.4 Future Work

As the rocks come from old roads, some may contain asphalt. This presents a problem, as the Nijst company cannot commercialize rocks that contain asphalt in the same way as rocks that do not contain asphalt. Therefore, one possible path for future work would be to detect the percentage of asphalt in the rocks (FIGURE 5.1).

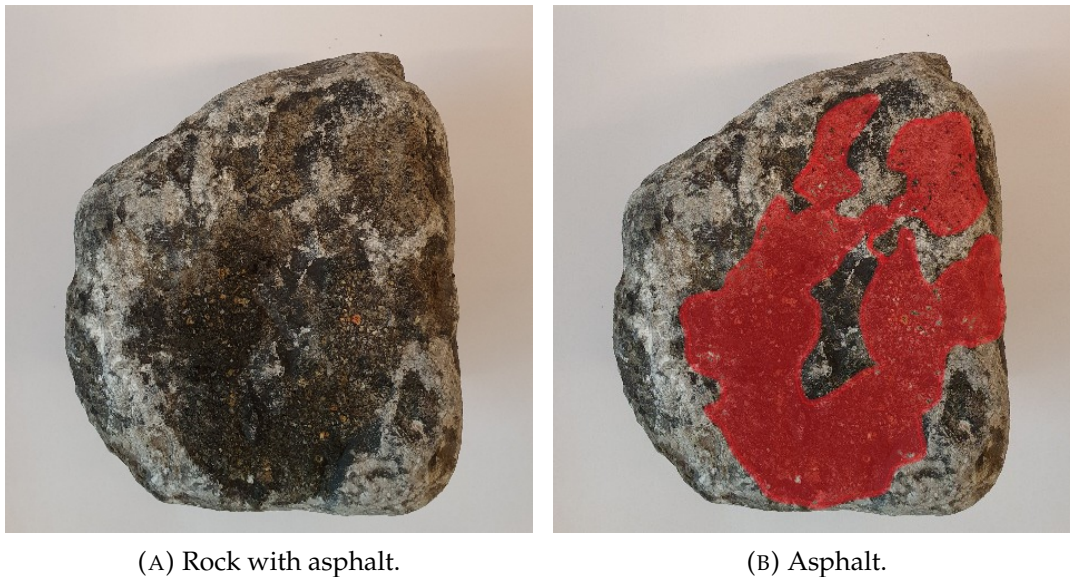


FIGURE 5.1: Asphalt in blue stone.

In the APPENDIX A, another example of asphalt is found, in a rock of the porphyry type (FIGURE A.5).

## Appendix A

### Additional Data



(A) First frame.



(B) Second frame.

FIGURE A.1: Example of global implementation of the system for a rock of the gres type, in 2 consecutive frames separated by 2 seconds.

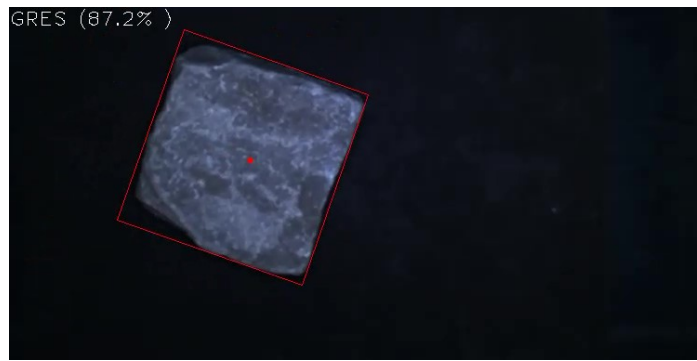


(A) First frame.



(B) Second frame.

FIGURE A.2: Example of global implementation of the system for a rock of the porphyry type, in 2 consecutive frames separated by 2 seconds.



(A) Face 1.



(B) Face 2.

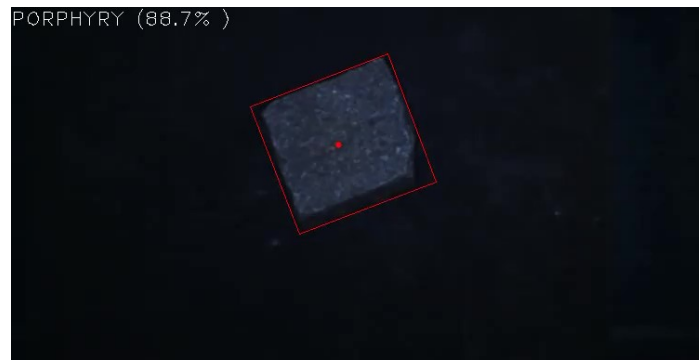


(C) Face 3.



(D) Face 4.

FIGURE A.3: Examples of classification regardless of the face captured by the camera for a rock of type gres. As can be seen, the ANN prediction is always gres, but the probability (in %) that this prediction is correct may vary, since different faces are present.



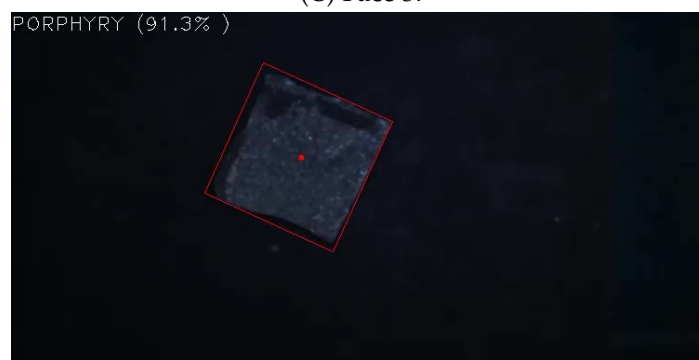
(A) Face 1.



(B) Face 2.



(C) Face 3.



(D) Face 4.

FIGURE A.4: Examples of classification regardless of the face captured by the camera for a rock of type porphyry. As can be seen, the ANN prediction is always porphyry, but the probability (in %) that this prediction is correct may vary, since different faces are present.



(A) Rock with asphalt.

(B) Asphalt.

FIGURE A.5: Asphalt in porphyry.

	RockType	MeanR	MeanG	MeanB	SkewR	SkewG	SkewB	KurtosisR	KurtosisG	KurtosisB
<b>0</b>	0	39.21	46.45	70.54	2.2	2.04	1.55	3.76	3.12	1.43
<b>1</b>	0	120.13	135.82	198.69	1.42	1.16	0.48	0.54	-0.18	-1.28
<b>2</b>	0	96.16	110.63	157.79	1.94	1.8	1.26	2.47	1.91	0.16
<b>3</b>	0	77.56	85.03	114.85	2.4	2.39	1.77	4.47	4.44	1.7
<b>4</b>	0	69.98	74.65	94.87	2.34	2.16	1.54	4	3.15	0.74
...	...	...	...	...	...	...	...	...	...	...
<b>791</b>	2	12.35	14.46	19.52	3.65	3.59	3.07	13.8	13.39	9.56
<b>792</b>	2	19.69	22.36	29.68	2.09	2.07	1.67	2.99	2.96	1.47
<b>793</b>	2	15.76	18.44	24.31	4.03	4.03	3.72	17.41	17.44	14.62
<b>794</b>	2	17.33	19.79	26.34	3.07	3.08	2.62	8.9	8.94	5.99
<b>795</b>	2	13.97	16.04	21.71	3.51	3.52	3.12	12.46	12.5	9.48

TABLE A.1: Features table for ANN training. This table contains: in its first column, the rock type coded; and the remaining columns, corresponding to the 9 extracted features. To show the table as an example, the value of the features was rounded to 2 decimal places, however, for the ANN training 6 decimal places were used.



# Bibliography

- [1] Matlab. (2021). "Rgb2gray," [Online]. Available: <https://nl.mathworks.com/help/matlab/ref/rgb2gray.html>. Accessed: 16/06/2021.
- [2] L. Shapiro and C George, "Stockman g: Computer vision," in *Prentice Hall*, 2002.
- [3] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
- [4] OpenCV. (2021). "Adaptive thresholding," [Online]. Available: [https://docs.opencv.org/master/d7/d4d/tutorial\\_py\\_thresholding.html](https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html). Accessed: 14/06/2021.
- [5] Wikipedia contributors, *Gradient — Wikipedia, the free encyclopedia*, [Online; accessed 14-June-2021], 2021. [Online]. Available: <https://en.wikipedia.org/w/index.php?title=Gradient&oldid=1024885933>.
- [6] I. Sobel, R Duda, P Hart, and J. Wiley, "Sobel-feldman operator,"
- [7] OpenCV. (2021). "Image gradients," [Online]. Available: [https://docs.opencv.org/master/d5/d0f/tutorial\\_py\\_gradients.html](https://docs.opencv.org/master/d5/d0f/tutorial_py_gradients.html). Accessed: 16/06/2021.
- [8] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.
- [9] OpenCV. (2021). "Morphological transformations," [Online]. Available: [https://docs.opencv.org/master/d9/d61/tutorial\\_py\\_morphological\\_ops.html](https://docs.opencv.org/master/d9/d61/tutorial_py_morphological_ops.html). Accessed: 14/06/2021.
- [10] S. Russell and P. Norvig, "Artificial intelligence: A modern approach," 2002.

- 
- [11] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013, vol. 398.
- [12] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017.
- [13] K. Janocha and W. M. Czarnecki, "On loss functions for deep neural networks in classification," *arXiv preprint arXiv:1702.05659*, 2017.
- [14] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.
- [15] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*, Elsevier, 1992, pp. 65–93.
- [16] M. Barr, "Pulse width modulation," *Embedded Systems Programming*, vol. 14, no. 10, pp. 103–104, 2001.
- [17] Wikipedia contributors, *Active-pixel sensor — Wikipedia, the free encyclopedia*, [Online; accessed 7-June-2021], 2021. [Online]. Available: [https://en.wikipedia.org/w/index.php?title=Active-pixel\\_sensor&oldid=1018753557](https://en.wikipedia.org/w/index.php?title=Active-pixel_sensor&oldid=1018753557).
- [18] ———, *Focal length — Wikipedia, the free encyclopedia*, [Online; accessed 7-June-2021], 2021. [Online]. Available: [https://en.wikipedia.org/w/index.php?title=Focal\\_length&oldid=1024363691](https://en.wikipedia.org/w/index.php?title=Focal_length&oldid=1024363691).
- [19] ———, *F-number — Wikipedia, the free encyclopedia*, [Online; accessed 7-June-2021], 2021. [Online]. Available: <https://en.wikipedia.org/w/index.php?title=F-number&oldid=1025188269>.
- [20] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.
- [21] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.
- [22] E. Bisong, "Google colab," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, Springer, 2019, pp. 59–64.

- [23] Wikipedia contributors, *Dynamic-link library* — *Wikipedia, the free encyclopedia*, [Online; accessed 7-June-2021], 2021. [Online]. Available: [https://en.wikipedia.org/w/index.php?title=Dynamic-link\\_library&oldid=1017853698](https://en.wikipedia.org/w/index.php?title=Dynamic-link_library&oldid=1017853698).
- [24] OpenCV. (2021). "Contours : Getting started," [Online]. Available: [https://docs.opencv.org/master/d4/d73/tutorial\\_py\\_contours\\_begin.html](https://docs.opencv.org/master/d4/d73/tutorial_py_contours_begin.html). Accessed: 15/06/2021.
- [25] M. H. Protter and C. B. Morrey, *College calculus with analytic geometry*. Addison-Wesley, 1977.
- [26] jdhao. (2019). "Cropping rotated rectangles from image with opencv," [Online]. Available: [https://jdhao.github.io/2019/02/23/crop\\_rotated\\_rectangle\\_opencv/](https://jdhao.github.io/2019/02/23/crop_rotated_rectangle_opencv/). Accessed: 24/04/2021.
- [27] M. Murugavel. (2019). "Object tracking — referenced with the previous frame using euclidean distance," [Online]. Available: <https://manivannan-ai.medium.com/object-tracking-referenced-with-the-previous-frame-using-euclidean-distance-49118730051a>. Accessed: 24/05/2021.
- [28] Wikipedia contributors, *Pythagorean theorem* — *Wikipedia, the free encyclopedia*, [Online; accessed 11-June-2021], 2021. [Online]. Available: [https://en.wikipedia.org/w/index.php?title=Pythagorean\\_theorem&oldid=1027893764](https://en.wikipedia.org/w/index.php?title=Pythagorean_theorem&oldid=1027893764).
- [29] S. Canu. (2021). "Object tracking with opencv and python," [Online]. Available: <https://pysource.com/2021/01/28/object-tracking-with-opencv-and-python/>. Accessed: 24/05/2021.
- [30] H. B. Barlow, "Unsupervised learning," *Neural computation*, vol. 1, no. 3, pp. 295–311, 1989.
- [31] Wikipedia contributors, *K-means clustering* — *Wikipedia, the free encyclopedia*, [Online; accessed 7-June-2021], 2021. [Online]. Available: [https://en.wikipedia.org/w/index.php?title=K-means\\_clustering&oldid=1021685301](https://en.wikipedia.org/w/index.php?title=K-means_clustering&oldid=1021685301).

- [32] D. Jadhav, G. Phadke, and S. Devane, "Colour and texture feature based hybrid approach for image retrieval," in *Advances in Computer Science, Engineering & Applications*, Springer, 2012, pp. 101–111.
- [33] S. Brown, "Measures of shape: Skewness and kurtosis," *Retrieved on August*, vol. 20, p. 2012, 2011.
- [34] V. López, A. Fernández, and F. Herrera, "On the importance of the validation technique for classification with imbalanced datasets: Addressing covariate shift when data is skewed," *Information Sciences*, vol. 257, pp. 1–13, 2014.
- [35] Wikipedia contributors, *Entropy (information theory) — Wikipedia, the free encyclopedia*, [Online; accessed 7-June-2021], 2021. [Online]. Available: [https://en.wikipedia.org/w/index.php?title=Entropy\\_\(information\\_theory\)&oldid=1024571966](https://en.wikipedia.org/w/index.php?title=Entropy_(information_theory)&oldid=1024571966).
- [36] S. (https://stats.stackexchange.com/users/2719/simone), *A general measure of data-set imbalance*, Cross Validated, (version: 2016-10-13). [Online]. Available: <https://stats.stackexchange.com/q/239982>.
- [37] I. Guyon *et al.*, "A scaling law for the validation-set training-set size ratio," *AT&T Bell Laboratories*, vol. 1, no. 11, 1997.
- [38] S. (https://stackoverflow.com/users/349130/dr\_snoopy), *What is h5 model in keras*, Cross Validated, (version: 2018-01-18). [Online]. Available: <https://stackoverflow.com/questions/48320588/what-is-h5-model-in-keras>.
- [39] Salaryexplorer. (2021). "Artificial intelligence developer average salary in belgium 2021," [Online]. Available: <http://www.salaryexplorer.com/salary-survey.php?loc=21&loctype=1&job=12098&jobtype=3>. Accessed: 14/06/2021.
- [40] S. explorer. (2021). "Developer / programmer average salary in belgium 2021," [Online]. Available: <http://www.salaryexplorer.com/salary-survey.php?loc=21&loctype=1&job=783&jobtype=3>. Accessed: 14/06/2021.

- [41] Paylab. (2021). "Plc programmer. electrical power engineering," [Online]. Available: <https://www.paylab.com/be/salaryinfo/electrical-power-engineering/plc-programmer>. Accessed: 14/06/2021.
- [42] Octopart. (2021). "Omron ccs-ldr2-90sw2 pricing and available inventory.," [Online]. Available: <https://octopart.com/ccs-ldr2-90sw2-omron-71332513>. Accessed: 14/06/2021.
- [43] Ebay. (2021). "One ccs pd2-3024-2 led light source controller," [Online]. Available: <https://www.ebay.com/itm/ONE-CCS-PD2-3024-2-LED-light-source-controller-DC24V-2-output-/283288537781>. Accessed: 14/06/2021.
- [44] ebay. (2021). "Ids ui-1225le-c-hq camara usb 2.0," [Online]. Available: <https://www.ebay.es/itm/154089823902?hash=item23e0781e9e:g:~mUAAOSwy~NfYe3w>. Accessed: 14/06/2021.
- [45] Kowa. (2021). "Lm8jc1ms | 2/3" 8mm 2mp c-mount lens," [Online]. Available: <https://www.kowa-lenses.com/en/lm8jc1ms-2mp-industrial-lens-c-mount>. Accessed: 14/06/2021.