SAVONIA


THESIS – BACHELOR'S DEGREE
TECHNOLOGY, COMMUNICATION AND TRANSPORT

# QUALITY ASSURANCE WITH THE USE OF COMPUTER VISION METHODS

AUTHOR/S Vladislav Komlev

SAVONIA UNIVERSITY OF APPLIED SCIENCES THESIS

Abstract

| | |
|---|---|
| Field of Study<br>Technology, Communication and Transport | |
| Degree Programme<br>Degree Programme in Information Technology, Internet of Things | |
| Author<br>Vladislav Komlev | |
| Title of Thesis<br>Automatic Quality Assurance of The Smart Factory's Products with Use of Computer Vision Methods | |
| Date 28.05.2025 | Pages/Appendices 33/0 |
| Client Organisation /Partners<br>Savonia University of Applied Sciences | |


The advent of Industry 4.0 has revolutionized manufacturing by integrating automation and data-driven technologies into smart factories. Quality assurance (QA) remains a critical challenge, where manual inspection methods are often slow, error-prone, and inefficient. This thesis explores the application of computer vision to automate QA processes in a smart factory environment, leveraging Basler's cameras for high-precision image acquisition and OpenCV for real-time defect detection. The proposed system is designed to interface with Programmable Logic Controllers (PLCs) via TIA Portal, enabling seamless integration with existing industrial automation infrastructure.

A proof-of-concept (POC) system is developed to detect defects such as surface scratches, misalignments, or missing components on production lines. The workflow includes image capture via Basler cameras, preprocessing and defect classification using OpenCV-based algorithms, and triggering corrective actions through PLCs. The system's performance is evaluated based on accuracy, latency, and scalability, demonstrating its potential to reduce human intervention while improving QA efficiency

Keywords

Smart Factory, PLC, TIA Portal, OpenCV, Python, Computer Vision, Machine Vision

CONTENTS

LIST OF FIGURES

# 1 INTRODUCTION

The manufacturing industry is undergoing a transformative shift with the rise of Industry 4.0, where automation, data exchange, and smart technologies are reshaping production processes. One of the critical challenges in modern manufacturing is ensuring consistent quality assurance (QA) while maintaining efficiency and reducing costs. Traditional QA methods, which rely heavily on manual inspection, are prone to human error, slow processing times, and inefficiencies—especially in high-volume production environments. To address these limitations, computer vision has emerged as a powerful tool for automating defect detection and improving QA accuracy in smart factories.

This thesis explores the integration of computer vision-based QA systems into industrial automation frameworks, specifically using Basler cameras for high-resolution image capture, OpenCV for real-time image processing, and Programmable Logic Controllers (PLCs) programmed via TIA Portal for seamless control and decision-making. By combining these technologies, the proposed system aims to detect defects—such as surface imperfections, dimensional inaccuracies, or assembly errors—with greater speed and reliability than manual inspection methods.

# 2 LITERATURE REVIEW

The integration of computer vision into smart factory environments for quality assurance (QA) represents a convergence of multiple technological advancements in Industry 4.0, automation, and machine vision. This chapter reviews existing research on automated visual inspection, industrial camera systems, PLC-based control, and real-time defect detection, providing a foundation for the proposed system.

## 2.1 Smart Factories and Industry 4.0

The Fourth Industrial Revolution (Industry 4.0) has transformed traditional manufacturing into interconnected, data-driven ecosystems known as smart factories. These facilities leverage Cyber-Physical Systems (CPS), Industrial Internet of Things (IIoT), and advanced automation to achieve unprecedented levels of efficiency and quality control (Zheng, 2018). A key characteristic of smart factories is their ability to perform real-time monitoring and adaptive decision-making, where quality assurance (QA) processes are increasingly automated to minimize human intervention (Oto Haffner, 2024).

Despite these advancements, visual inspection remains a bottleneck in many production lines. Manual QA methods are not only labor-intensive but also inconsistent, with human inspectors typically achieving only 80-90% detection accuracy for subtle defects (Hütten, et al., 2024). This has spurred significant research into automated visual inspection systems that can match or exceed human performance while operating at production-line speeds.

### 2.1.1 Industry 4.0 Communication Architecture

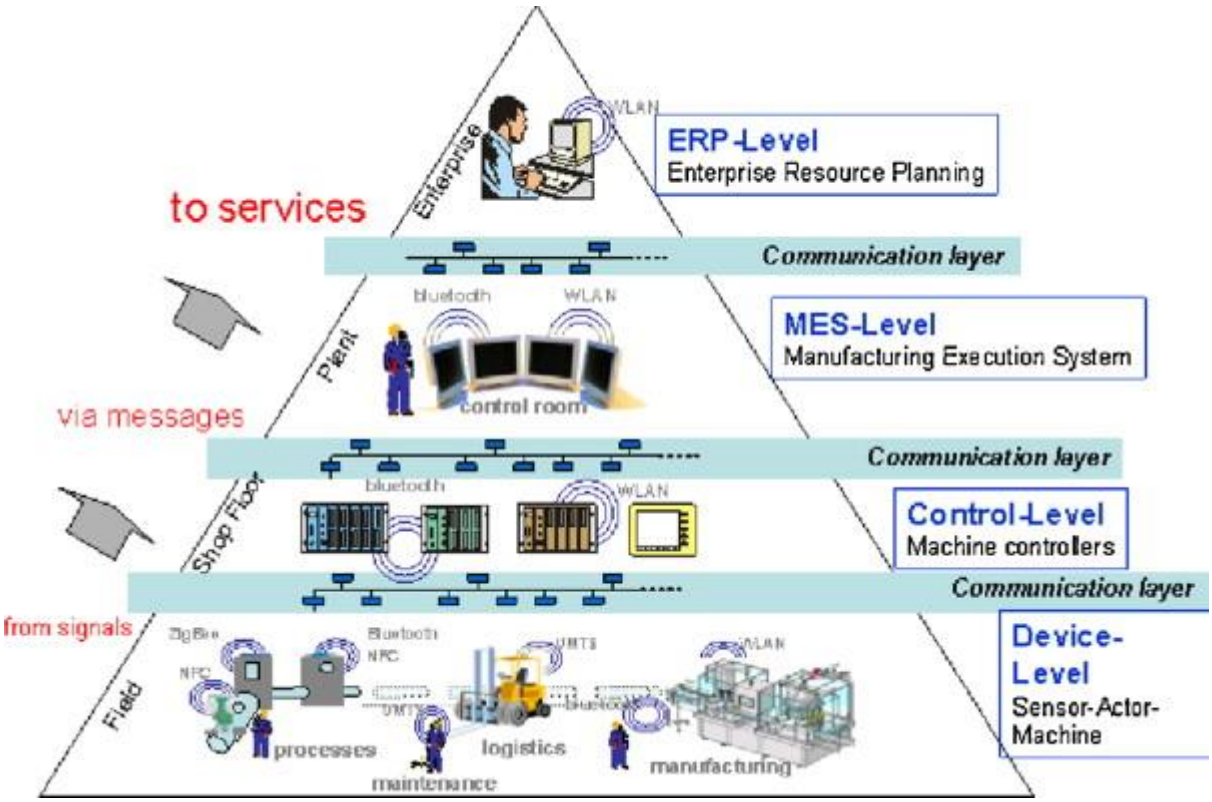



Figure 1. The structure of Smart factory

The hierarchical communication framework in smart factories forms the core principles of Industry 4.0, as illustrated in Figure 1. This architecture establishes seamless connectivity across all operational levels, from enterprise planning to machine-level control.

At the ERP level, enterprise resource planning systems coordinate high-level business processes, including logistics and maintenance, through standardized service-oriented messaging. The MES layer (Manufacturing Execution System) acts as the critical bridge between business and production domains, translating strategic plans into executable workflows while aggregating real-time performance data. The Control level is responsible for hardware management and data transmission to the MES layer. The original level is a Device level, which includes sensors to read and transmit data and actuators to perform an action based on the sensors' readings (Lasi, et al., 2014).

## 2.2 Computer Vision in Quality Assurance

Computer vision has emerged as a transformative technology for industrial QA, with applications ranging from simple presence/absence checks to complex surface defect classification. Traditional image processing techniques using OpenCV, such as edge detection (Canny, Sobel), contour analysis, and template matching, remain popular due to their computational efficiency and interpretability (Ghosh, et al., 2024). For example, in PCB inspection, OpenCV-based systems can achieve >95% accuracy in detecting missing components or soldering defects when proper lighting conditions are maintained (Zhang, et al., 2023)

More recently, deep learning approaches have demonstrated superior performance for complex defect detection tasks. Convolutional Neural Networks (CNNs) and object detection models like YOLO (You Only Look Once) and its upgraded versions such as DEW-YOLO can learn intricate defect patterns without explicit feature engineering (Li & Chen, 2024). However, these methods require substantial training data and computing resources, making them less suitable for resource-constrained edge deployments. Hybrid approaches that combine traditional computer vision preprocessing with lightweight neural networks for classification are gaining traction as balanced solutions. For instance, (Maaz, et al., 2022) developed EdgeNeXt, an efficient hybrid CNN-transformer architecture optimized for mobile vision applications. Similarly (Abdelkrim Mecharbat, et al., 2022) introduced HyT-NAS, aa hardware-aware neural architecture search framework that designs hybrid models combining convolution-based and attention-based components for vision tasks on edge devices.

## 2.3 Industrial Camera Systems

Capturing a proper and accurate image is challenging in normal conditions. To achieve a satisfactory result, the next parameters must be analyzed to select the best solution for case: field of view, working distance, resolution, depth of filed, sensor size, and primary magnification (https://www.edmundoptics.com/Knowledge-Center/application-notes/imaging, accessed 14.02.2025)

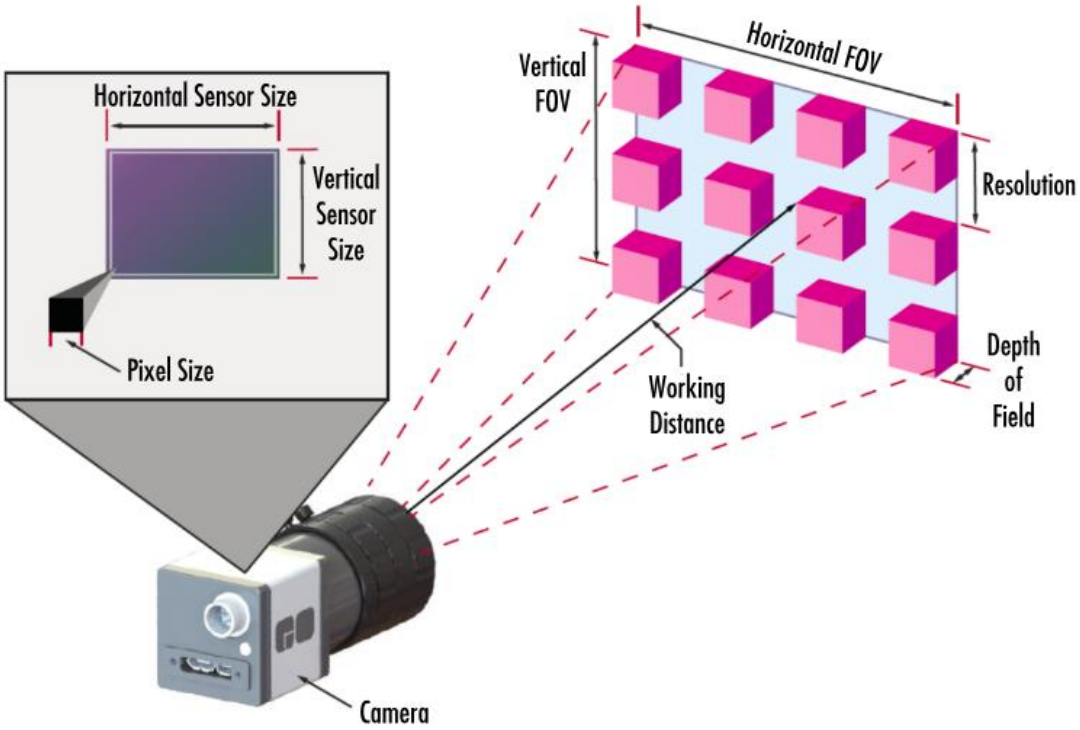



Figure 2. Illustration of the fundamental parameters of an imaging system (Hollows & James, n.d.)

One of the first considerations is the field of view, which determines how much of the scene can be seen by the camera. A wider field of view allows for broader coverage but may reduce the detail visible in the image. Another important factor is the working distance, which is the space between the lens and the object being observed. This distance influences how the system is physically arranged and can impact image clarity, especially in tight environments.

Resolution plays a central role in defining the system's ability to distinguish fine details. Higher resolution allows for more precise measurements and better identification of small features, making it critical in inspection tasks. Depth of field is also vital; it refers to the range over which objects remain in focus. A greater depth of field reduces the need to constantly adjust the focus when imaging objects at different distances.

The sensor size influences how much of the image is captured and how detailed it appears. Larger sensors can record more information but often require more space and compatible lenses. Finally, primary magnification relates the size of the object seen by the sensor to the actual field of view. It determines how large the image appears and affects how much detail is retained. Balancing all six parameters is essential for tailoring imaging systems to meet specific application requirements.

However, it becomes more complicated considering the extreme environment of the industrial factories. Environmental challenges, such as dust, noise and vibration must be considered during the choice of camera for the application. Therefore, for industrial cases special cameras such as Basler’s cameras are used because of their high frame rate, global shutter technology eliminating motion blur for precise defect capture, and seamless integration with Python making the development process of automated solutions making them a great choice for these tasks.

Studies have demonstrated that combining high-resolution cameras, such as Basler area scan models, with appropriate lighting techniques—like coaxial lighting for reflective surfaces and diffuse dome lighting for uniform illumination—can significantly enhance the detection of micro-scratches and other defects on metal surfaces (Bufi & Shehata, 2020).

## 2.4 PLC Integration and TIA Portal

The connection between vision systems and factory automation infrastructure represents a crucial component of industrial quality assurance. Classical portable computers are not suitable for this task because the code execution is not synchronized, and an event driven system is employed. Moreover, it could be complicated to connect other hardware such actuators and sensors for a proper execution of the task. On the other hand, PLCs are specifically built for real-time industrial control. (https://www.candtsolution.com/news_events-detail/what-is-a-plc-the-difference-with-pc/ , accessed 12.04.2025)

PLCs provide deterministic execution and are designed to seamlessly interface with actuators, sensors, and other industrial hardware. For example, Siemens SIMATIC PLCs, programmed using the TIA Portal, are widely used to manage quality assurance tasks in manufacturing by responding quickly and reliably to detected defects.

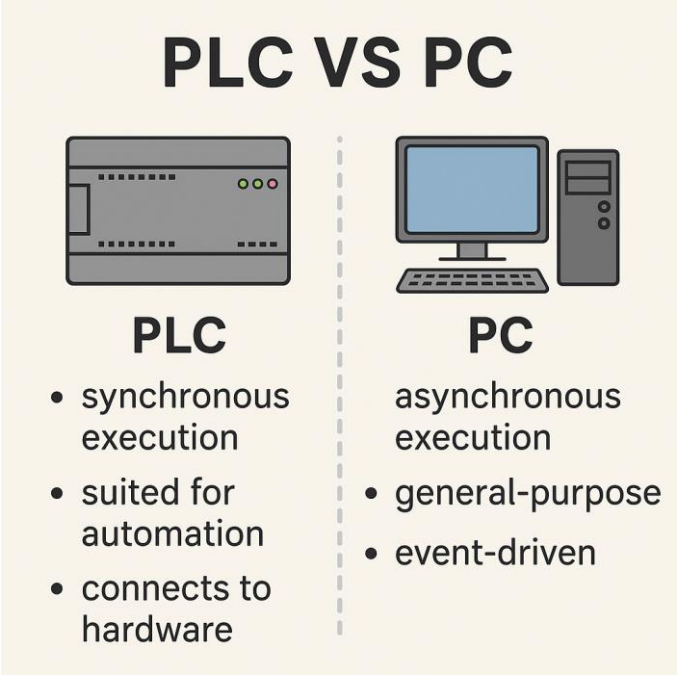



Figure 3. Key differences between PC and PLC

PLCs also support industrial communication protocols such as PROFINET for real-time data transfer with latencies below 1ms and OPC UA for standardized system integration, making them highly suitable for modern factory environments.

## 2.5 Current Challenges and Research Directions

Despite demonstrated successes, several challenges persist in implementing computer vision for industrial quality assurance. Edge deployment of vision systems remains problematic, with most research focusing on PC-based solutions rather than optimized implementations for industrial edge devices. Lighting robustness continues to be an issue, as many current systems perform poorly in environments with variable ambient light conditions.

# 3 TECHNICAL BACKGROUND

This chapter provides the technical foundation for implementing computer vision-based quality assurance in smart factory environments. The discussion focuses on three core components: vision system hardware, image processing algorithms, and industrial control integration.

## 3.1 Vision Systems Components

Industrial machine vision systems represent a solid integration of optical physics, sensor technology, and mechanical engineering. The Basler acA640-90um camera used in this project captures fine image detail. With installed monochrome CCD sensor for clear results and having 659 x 494 resolution allowing to catch minor defects such as scratches. Moreover, this camera has a global shutter to freeze the motion without blur techniques. This characteristic proves that Basler acA640-90um camera usage in the project is crucial because it can withstand the manufacturing environment challenges such as vibration and dust. (Basler, 2022)

The sensor architecture implements a global electronic shutter with a minimum exposure time of 1µs, effectively freezing motion for objects moving at speeds up to 5m/s when using standard 5MP resolution. (Coates & Juvan-Beaulieu, 2020) The cameras' 10GigE Vision interface leverages the IEEE 802.3 standard to support data transfer rates up to 1.25 GByte/s (Basler, n.d.), enabling high-throughput inspection of fast-moving production lines without compromising image quality.

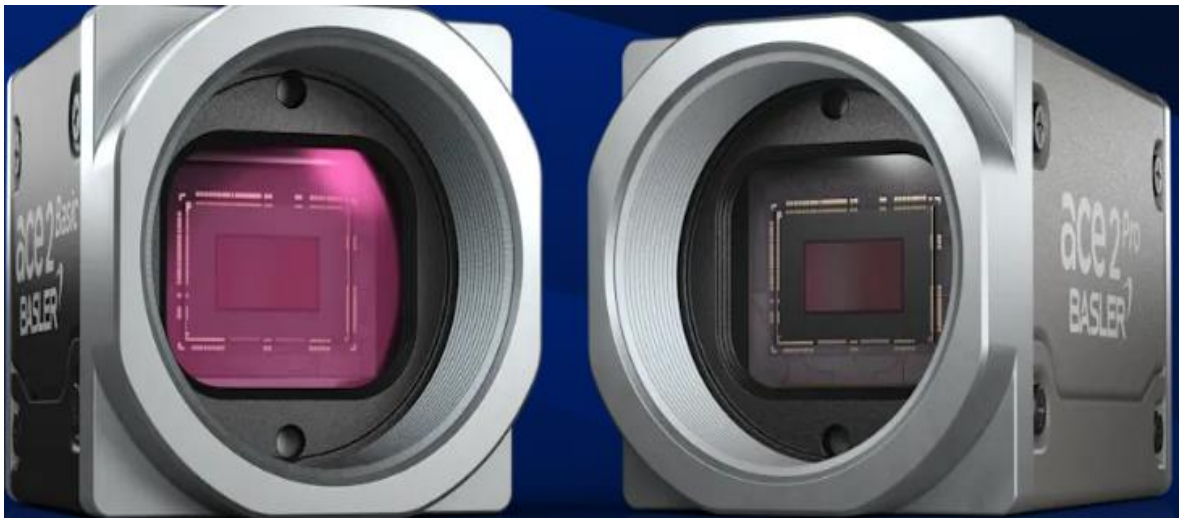



Figure 4. Basler ace 2 camera that could be used for image capturing

## 3.2 Digital Image Acquisition and Representation

Digital images are created through a sophisticated process that converts light into electronic data. When light enters a camera, it passes through a lens assembly that focuses the image onto a sensor composed of millions of light-sensitive elements called photosites. Each photosite corresponds to a pixel in the final image and functions as a microscopic light meter, measuring the intensity of incoming photons (Spring, et al., n.d.).

Modern industrial cameras use either CCD (Charge-Coupled Device) or CMOS (Complementary Metal-Oxide-Semiconductor) sensors. While CCD sensors were traditionally preferred for their high image quality, advances in CMOS technology have made them the dominant choice for industrial applications due to their lower power consumption, faster readout speeds, and better integration with processing electronics

(Turchetta, et al., n.d.). The sensor's photodiodes convert light into electrical charge, with the amount of charge being proportional to the light intensity and exposure time.

Then, an analog-to-digital converter (ADC) transforms these analog electrical signals into digital values. Industrial cameras typically use 8-bit to 12-bit ADCs, meaning each pixel can represent 256 to 4096 distinct intensity levels. This digitization process creates a two-dimensional array of numbers, where each number corresponds to the brightness of a specific point in the image.

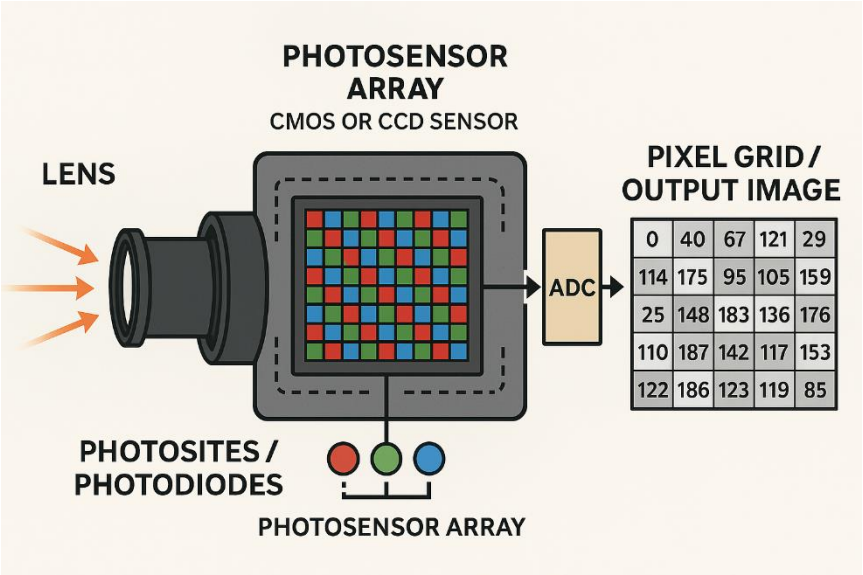



Figure 5. Image Acquisition Process (https://chatgpt.com, accessed 15.03.2025)

# 4 IMAGE PROCESSING TECHNIQUES

Despite the vast variety of processing methods and their parameters, each case requires a precise selection of the most appropriate and efficient techniques to extract the required features. In the case of scratch detection of 3D-printed object, the next methods are used: Fourier Transform, Canny's Edge Detection algorithm, Hough lines and contour filtering. Looking into the mechanism of these algorithms allows to get an understanding of the image processing pipeline.

## 4.1 Fourier Transform and Inverse Fourier Transform

The Fourier Transform is a commonly used method in digital image processing that converts an image from its spatial domain (pixel intensities) into the frequency domain, revealing hidden patterns and structures. In industrial quality inspection, it plays a crucial role in noise removal, texture analysis, and detection (Gonzalez, 2018). Because an image is represented as a 2-dimensional array, the 2-dimensional Fourier transform is applied to the processing with the next formula:

$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x.y)e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})}$,

where f(x,y) – pixel intensity at position (x,y), F(u,v) - Frequency domain representation, u,v – frequency variables. The inverse Fourier Transform provided the formula:

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

reconstructs the image from the frequency domain. In the case of this thesis, this method is used to detect and suppress peaks in frequency domain, highlighting the edges and scratches and deprecating the texture pattern of 3D-printing.

## 4.2 Canny Edge Detection

The Canny edge detector represents a mathematically rigorous approach to edge detection that combines careful noise reduction with precise edge localization (Gonzalez, 2018). The algorithm's effectiveness stems from its foundation in differential calculus and optimization theory, making it particularly suitable for industrial machine vision applications where robustness and accuracy are paramount. The Canny edge detection algorithm is mainly composed of 5 steps: noise reduction, gradient calculation, non-maximum suppression, double threshold, and edge tracking by Hysteresis. When noise reduction is provided by applying Gaussian blur to the original image, the other steps are not that obvious.

Edges correspond to a change of intensity of pixels detected by Sobel filter in both horizontal(x) and vertical(y) directions. It is implemented by applying convolution for the blurred image with Sobel kernels Kx and Ky respectively:

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, K_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

These kernels are used to determine the derivatives Ix and Iy with respect to the axes x and y.

Then, the magnitude G and the slope $\theta$ are calculated as follows:

$$|G| = \sqrt{I_x^2 + I_y^2}$$

$$\theta(x, y) = \arctan(\frac{I_y}{I_x})$$

The next step is transforming thick lines into thin. For this task non-maximum suppression, double thresholding, and Hysteresis algorithms are used. Non-maximum suppression functions by finding the pixel with the maximum intensity value of the edge. It requires splitting the 3x3 grid of pixels into 8 sections. Then, the check is performed if the gradient direction lies between the angle -22.5 and 22.5 to determine pixels to compare with the considered pixel. In figure below, the pixel *p* is considered. As can be seen, the pixels falling in between the appropriate angle are *q* and *r*. After comparing the intensity levels of pixel *p* with pixel values of *q* and *r* independently, the decision on keeping the value of *p* nor setting it to 0 is made (Liang, n.d.).

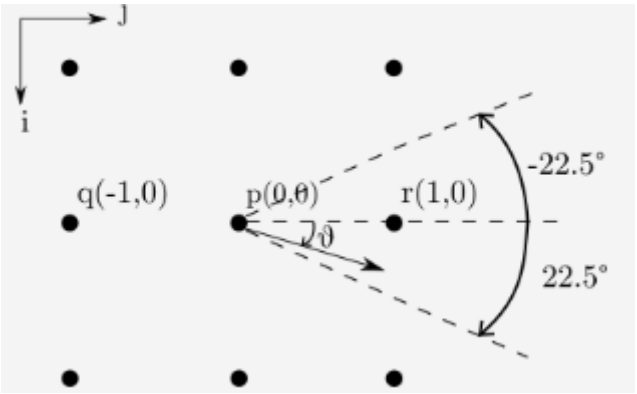



Figure 6. Non-maximum suppression

However, the result of the non-maximum suppression is not perfect. Some of the detected edges are not the actual edges, and there is noise in the picture. Therefore, double thresholding is used, which sets low ($L_{threshold}$) and high ($H_{threshold}$) thresholds. Pixels with value less than $L_{threshold}$ are set to 0; if the value satisfies the inequality $L_{threshold}$ < f(x,y) < $H_{threshold}$, where f(x,y) is the value of the pixel at the position (x,y), then these pixels are defined as weak edges; if the value is more than $H_{threshold}$, then these pixels are defined as strong edges.

The final step of Canny edge detector algorithm is the application of an edge tracking algorithm. Its core idea is keeping weak edges only if they are connected to strong edges.

## 4.3 Hough Transform

The Hough transform represents a powerful mathematical approach for detecting straight lines in digital images, with particular importance in industrial machine vision applications. This technique converts the problem of line detection in image space into a more manageable peak detection problem in parameter space, demonstrating robustness against noise and partial occlusion that commonly occur in factory environments (Dulari, 2024).

The fundamental mathematics behind the Hough transform begins with the normal representation of a line:

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

where ρ is the perpendicular distance from the origin to the line, and θ is the angle between the x-axis and this perpendicular (Simulink, n.d.). This parametric representation avoids the undefined slope issues encoun-

tered in the standard y = mx + b form, making it particularly suitable for algorithmic implementation. In practical applications, this means the algorithm can reliably detect both vertical and horizontal lines that commonly appear in manufactured parts.

Each edge pixel $(x_i, y_i)$ then votes in the Hough parameter space (ρ, θ) according to the equation above. In industrial practice, this voting process creates an accumulator array where local maxima correspond to likely lines in the original image. The precision of line detection depends directly on the discretization steps chosen for ρ and θ, with typical values of Δθ = 1° and Δρ = 1 pixel providing a good balance between accuracy and computational efficiency.

## 4.4 Contour Filtering

The contour filtering algorithm serves as a critical component in machine vision systems for industrial quality control, enabling precise extraction and analysis of object boundaries in digital images. This technique builds upon the fundamental concept that meaningful object contours exhibit distinct geometric and topological properties that can be mathematically characterized and separated from noise or irrelevant edges (Pietrzkiewicz, 2019).

At the core of contour filtering lies the mathematical representation of contours as ordered sequences of connected edge points. A contour C can be formally defined as:

$$C = \{p1, p2, \dots, p_n\}, p_i = (x_i, y_i) \in Z^2$$

with connectivity condition: $\forall i \in [1, n-1], p_i$ and $p_{i+1}$ are 8-connected neighbors (OpenCV, n.d.)

The contour area calculation provides a primary filtering criterion, computed using the shoelace formula:

$$A = \frac{1}{2}\left|\sum_{i=1}^{n} x_i y_{i+1} - x_{i+1} y_i\right|$$

This area measurement enables rejection of both excessively small contours (likely noise) and unexpectedly large contours (indicating multiple merged objects), with typical threshold values determined from statistical analysis of known good parts (Moon, n.d.).

Contour compactness, defined as:

$$K = \frac{P^2}{4\pi A}$$

where P is the perimeter length, serves as an effective shape regularity metric (Frejlichowski, 2002). For perfectly circular objects K = 1, while industrial components with complex geometries exhibit higher values. This measure proves particularly valuable for detecting deformed parts, where manufacturing defects often increase the compactness ratio beyond acceptable thresholds.

The convex hull operation represents another powerful filtering tool, mathematically defined as the smallest convex set containing all contour points. (OpenCV, n.d.) The difference between the original contour and its convex hull, known as the convexity defects, provides quantitative measures of surface irregularities. For a contour C and its convex hull H, the defect depth d at point p ∈ C is:

$$d(p) = \min_{h \in H} \|p - h\|$$

# 5 SYSTEM DESIGN AND METHODOLOGY

This chapter presents the comprehensive design framework and methodological approach for implementing the computer vision-based quality assurance system in smart factory environments. The discussion systematically addresses the system architecture, component selection criteria, and integration strategies while maintaining rigorous adherence to industrial standards.

## 5.1 System Architecture Overview

The architecture follows a distributed edge computing model, where a manufacturing-grade industrial PC serves as the primary processing unit. This setup ensures real-time performance while maintaining compatibility with existing factory control systems.

The system architecture is structured into three distinct functional layers, each designed to perform specialized tasks while maintaining seamless integration for robust industrial operation (https://macronet-services.com/what-are-the-benefits-and-challenges-of-edge-computing, accessed 15.04.2025).

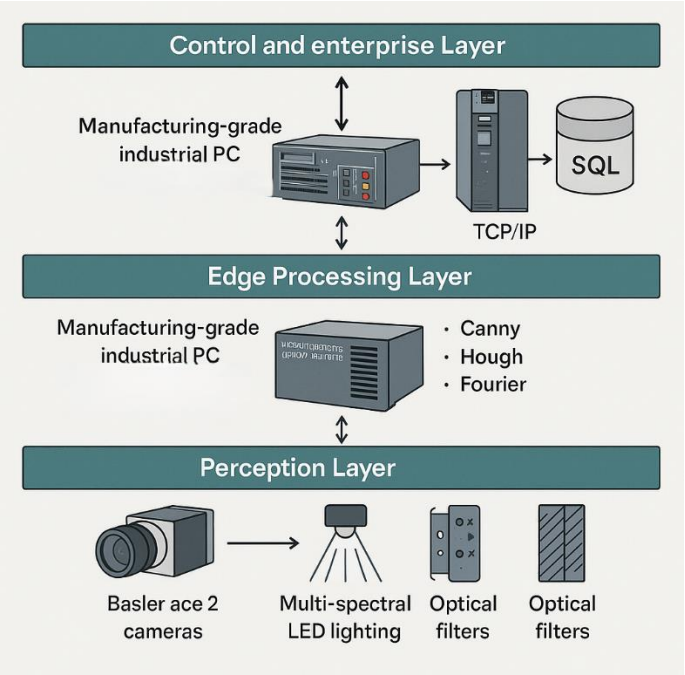



Figure 7. The structure of the system (https://chatgpt.com, accessed 15.03.2025)

The perception layer forms the foundation of the system, responsible for high-precision image acquisition. Basler ace 2 cameras are employed, utilizing encoder-synchronized triggering to ensure precise temporal alignment of captured images. Multi-spectral LED lighting is integrated to provide consistent and optimized illumination, enhancing defect detection capabilities.

At the edge processing layer, computational tasks are executed in real time to facilitate immediate defect detection. A manufacturing-grade industrial PC, such as the Beckhoff CX2040, serves as the processing unit, running classical computer vision algorithms including Canny edge detection, Hough transform, and Fourier analysis. This approach eliminates dependencies on deep learning, ensuring deterministic performance and reduced computational overhead, which is a strict parameter due to the limit computational power of the unit. Local buffering and preprocessing mechanisms are implemented to minimize latency, enabling rapid defect identification without compromising system responsiveness.

The control and enterprise layer manages higher-level system functions, interfacing with industrial automation components and enterprise data systems. Siemens S7-1500 PLCs are utilized to execute rejection mechanisms and process control actions based on defect detection outputs. Communication across factory networks is facilitated through TCP-based protocols, ensuring compatibility with legacy infrastructure. Additionally, SQL database integration is employed to log defect statistics, enabling comprehensive quality trend analysis and long-term process optimization.

This architectural design emphasizes robustness and simplicity, avoiding unnecessary complexity while ensuring reliable operation in demanding industrial environments. By leveraging proven technologies and deterministic processing methods, the system achieves high performance without compromising stability or maintainability.

## 5.2 Hardware Selection and Implementation

Imaging subsystems are responsible for accurate images capturing minimizing noise effects caused by side illumination and industrial noise. To achieve that Basler ace 2 camera is used allowing to capture high resolution images at 75 fps with 5.1 MP matrix resolution and global shutter. Encoder synchronization ensures images are captured at precise conveyor positions.

For edge computing the manufacturing computational unit is used due to its resilience to harsh environments and connectivity options.

## 5.3 Software Architecture

The image processing pipeline is structured to ensure robust defect detection while maintaining computational efficiency. The initial stage involves preprocessing, where flat-field correction is applied to compensate for any non-uniformities in lighting across the captured images. This step ensures consistent illumination, which is critical for accurate defect identification. Additionally, Fourier transform techniques are employed to eliminate high-frequency noise, such as sensor artifacts or electrical interference, further enhancing image clarity.

Following preprocessing, the pipeline transitions to defect detection, where a combination of classical computer vision algorithms is utilized. Canny edge detection is applied to identify abrupt intensity variations, which often correspond to defects such as cracks or surface irregularities. The HoughLines transform is then used to detect straight-line anomalies, including scratches and fractures, by analyzing geometric patterns in the image. To minimize false positives, contour filtering is implemented, where detected features are evaluated based on predefined shape and size criteria, ensuring only relevant defects are flagged.

The final stage of the pipeline involves decision logic, where a deterministic approach is favored over machine learning for reliability and interpretability. Threshold-based classification is employed to categorize defects based on severity, while statistical process control (SPC) rules monitor inspection results over time, flagging any abnormal trends that may indicate process deviations.

For system integration, TCP sockets serve as the communication protocol, facilitating real-time transmission of inspection results to the programmable logic controller (PLC). This ensures seamless coordination between defect detection and subsequent control actions, such as part rejection or process adjustments. The pipeline is designed for industrial robustness, prioritizing speed, accuracy, and compatibility with existing manufacturing systems.

# 6 IMPLEMENTATION

This part provides a deep exploration of the work made step by step.

## 6.1 Hardware Implementation

### 6.1.1 Conveyor belt

The smart factory used for this thesis is a smart factory produced by Lucas-Nülle (https://www.lucas-nuelle.us/, accessed 25.01.2025) for educational projects. At the factory, the product is assembled with the following components: bottom piece, top piece, and a bolt going through the pieces to fix their position relative to each other.

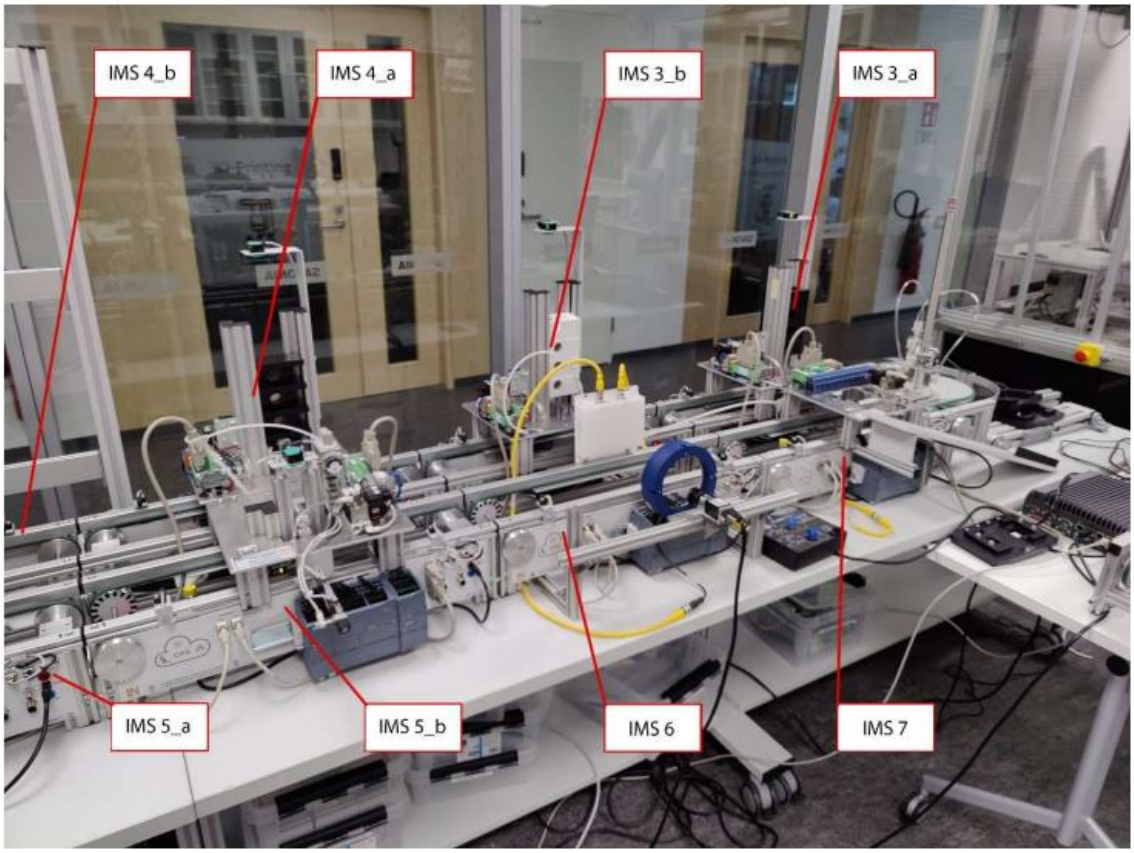



Figure 8. The smart factory with labeled stations. Station labels: IMS3 – bottom part of the box; IMS4 – top part of the box; IMS5 – bolt for the box; IMS6 – quality control; IMS7 – removing the finished product from the production line (Martikainen, 2022)

A particular station is fixed to the top of each of the several conveyor belts that make up the manufacturing hardware. The top and bottom pieces are black and white, while the bolt pieces are either red plastic or metal. This makes a total of six stations for various pieces. Additionally, one station (IMS6) is utilized for quality control, while another removes the final product from the production line.

This thesis specializes in quality control, therefore IMS6 unit was isolated and modified for the proof of concept, making it unnecessary to modify the complex code structure of the factory and remove the need of integration. The conveyor belt has a programmable logic controller, a motor, two magnetic detection sensors at each end of the belt, an RFID reader, a camera, light bars, and an industrial grade computational unit for software running.

### 6.1.2 Lighting installation

One of the most crucial steps in receiving a high-quality image is the choice of the illumination method. Because the object is inspected on having a scratch defect, the off-axis lighting scheme has been chosen. In detail, for scratch detection the light source should be placed in the dark field region to highlight the scratches on the resulting image for more accurate detection. (Advanced illumination, n.d.)

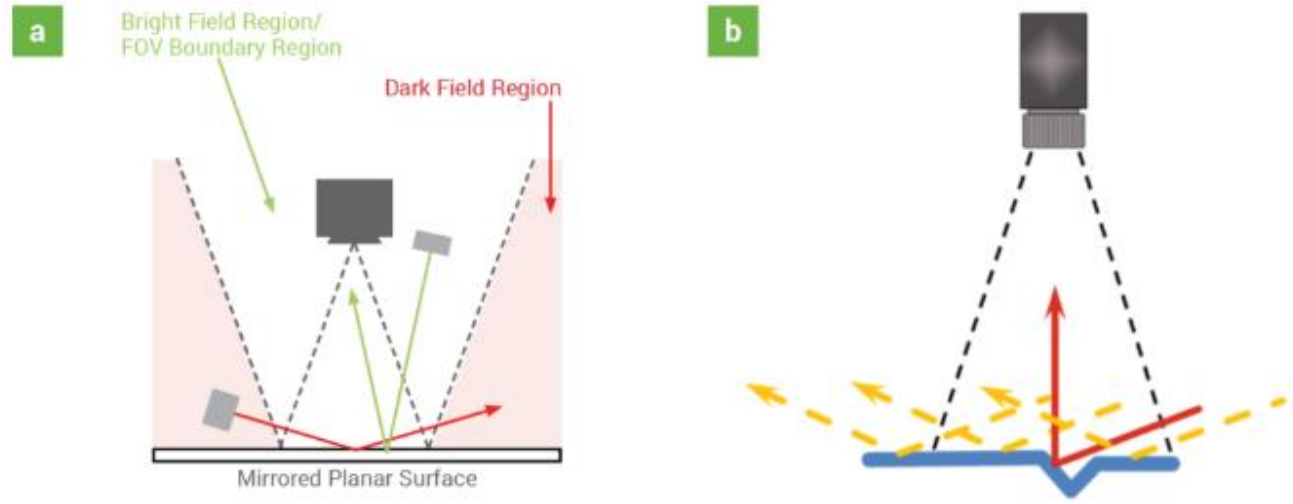



Figure 9. The light beam geometry. Picture b demonstrates how the surface defect affects the beam geometry highlighting the scratches on the image (Advanced illumination, n.d.)

However, the room light interrogates with an installed light system, therefore the decision on making a dark isolated area was made. For this purpose, laser-cut wooden walls with attached light bars were used.

### 6.1.3 Camera installation

A camera used in this case is Basler acA640–90um (Basler, n.d.), commonly used in machine vision tasks. The camera was also selected because of the compatibility with different types of lenses making it convenient to swap them depending on the task. Because of the distance between the object and the camera, the lenses must have a wide angle and short focal length. Therefore, based on the characteristics and users' cases Tamron 1/1.2 16mm M112FM16 (Tamron, n.d.)was chosen for this task.


Figure 10. Basler acA640-90um camera (Basler, n.d.)


Figure 11. Tamron 1/1.2 16mm M112FM16 lens (Tamron, n.d.)

## 6.2 Software Implementation

Software for quality control is split into two parts: the first part is computer vision software responsible for image processing, frame capturing and quality analysis, while the second part is PLC ladder logic to get the result from the computer through TCP connection and to stop the conveyor belt if too many scratches are detected.

### 6.2.1 Quality Assurance Software

The quality assurance software is implemented with Python and its computer vision and data analytics packages such as OpenCV, matplotlib, pypylon, and pandas.

Before any scratch detection can take place, image data must be acquired from the production line. This is often done using a Basler industrial camera, which is commonly used in smart factories and automated inspection systems. To interface with a Basler camera in Python, the pypylon library is used. This is Basler's official SDK wrapper for Python that allows full control over image acquisition, exposure settings, trigger modes, and more.

```
grab_result = 0
while 1:
    camera = pylon.InstantCamera(pylon.TlFactory.GetInstance().CreateFirstDevice())
    camera.Open()
    camera.StartGrabbingMax(1)
    grab_result = camera.RetrieveResult(5000, pylon.TimeoutHandling_ThrowException)
```

Figure 12. Usage of pypylon library for continuous image acquisition

Frame capturing typically starts by importing the relevant modules from pypylon.pylon and initializing the camera interface. A TLFactory object is used to detect connected Basler devices. Once a camera is selected (usually the first available one), it is opened and configured. The StartGrabbingMax(1) command instructs the camera to capture a single frame. After grabbing, RetrieveResult retrieves the image buffer. If the image is valid, the raw image data is extracted using grab_result.Array. This NumPy array can then be passed to OpenCV for further processing, such as blurring, FFT, or scratch highlighting.

The camera is typically closed or reused in a loop to continuously monitor products on a conveyor system. This live acquisition is essential in real-time inspection environments.

Then, it is a crucial step in image processing to convert it to grayscale to simplify processing. Grayscale conversion reduces computational complexity by eliminating color channels, which are unnecessary for edge or scratch detection.

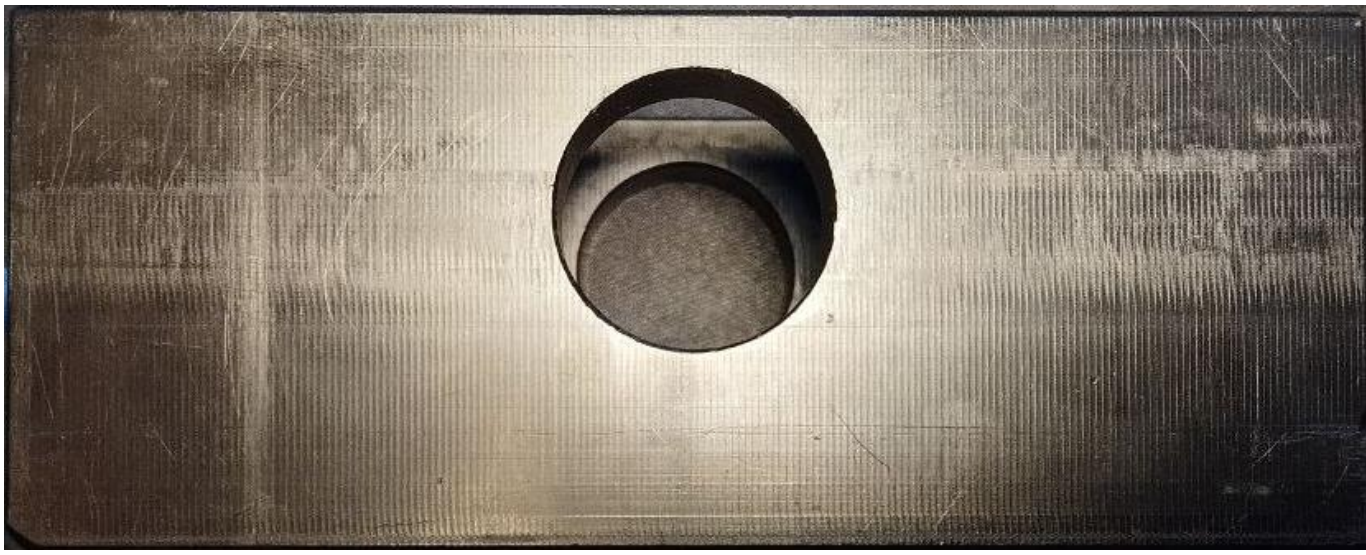


Figure 13. The original image of the object

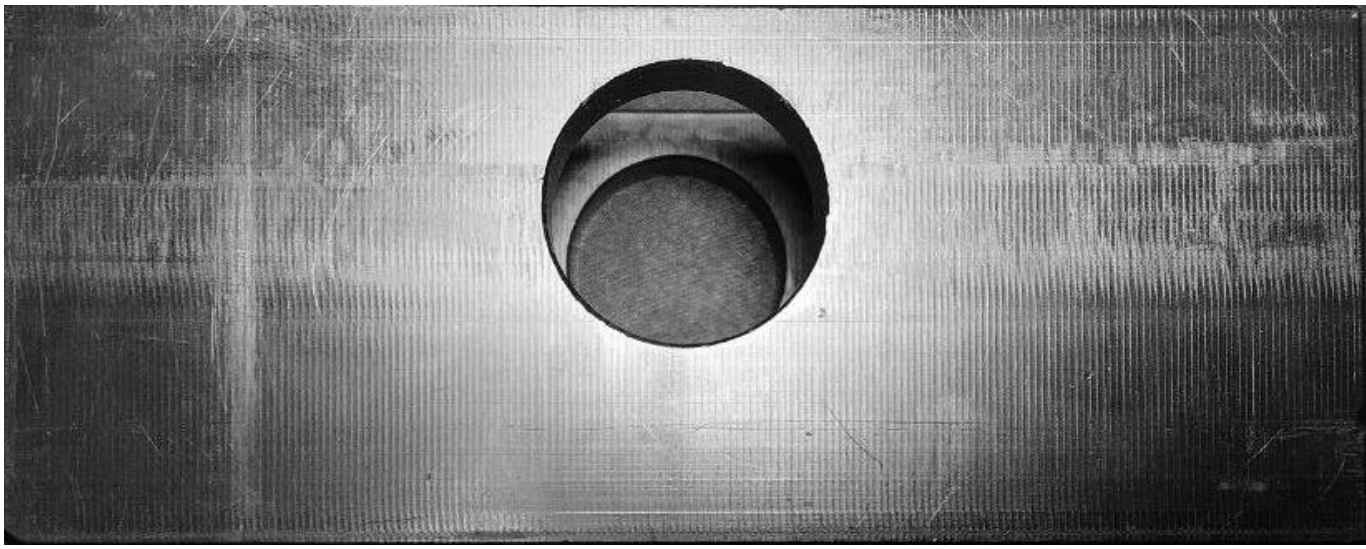


Figure 14. The image of the object in grayscale

Once the image is in grayscale, a Gaussian blur is applied to minimize image noise. This step is essential in reducing the impact of minor pixel-level fluctuations that may otherwise be occasionally misinterpreted as scratches. The blur uses a kernel size of (5,5), which defines the intensity of the smoothing operation.

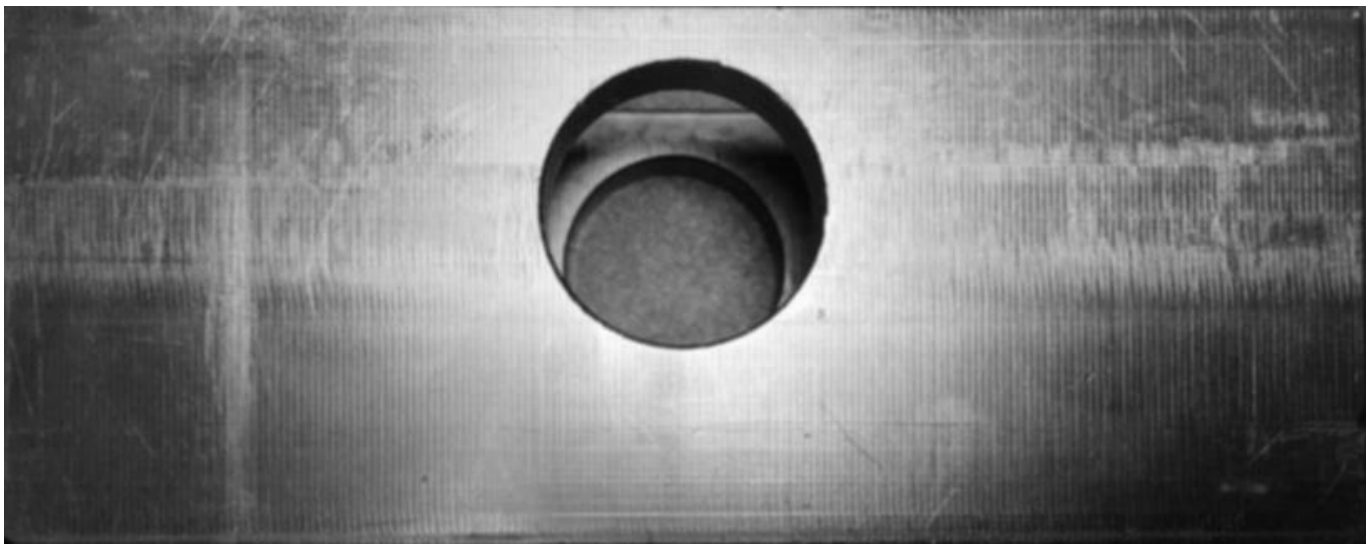


Figure 15. The grayscale image after applying Gaussian blur

After preprocessing the image using grayscale conversion and Gaussian blur, the next major technique employed in the code is frequency filtering using the Fast Fourier Transform (FFT).

This transformation helps identify and suppress repetitive patterns that might otherwise mask or resemble actual scratches.

```
def remove_repetitive_textures(image_gray):
    dft = cv2.dft(np.float32(image_gray), flags=cv2.DFT_COMPLEX_OUTPUT)
    dft_shift = np.fft.fftshift(dft)

    plt.plot("dft", np.abs(dft_shift))

    # Create a mask to remove repetitive patterns
    rows, cols = image_gray.shape
    crow, ccol = rows // 2, cols // 2
    mask = np.ones((rows, cols, 2), np.uint8)
    mask[crow - 30:crow + 30, ccol - 30:ccol + 30] = 0  # Suppress repetitive patterns including high frequency pea

    # Apply mask and inverse transform
    fshift = dft_shift * mask
    f_ishift = np.fft.ifftshift(fshift)
    img_back = cv2.idft(f_ishift)
    img_back = cv2.magnitude(img_back[:, :, 0], img_back[:, :, 1])

    return cv2.normalize(img_back, None, 0, 255, cv2.NORM_MINMAX).astype(np.uint8)
```

Figure 16 . The code responsible for Fast Fourier Transform and inverse transformation back into the spatial domain

In this code, cv2.dft() is used to perform a 2D FFT on the blurred grayscale image. The result is then shifted to center the low-frequency components using np.fft.fftshift(). The magnitude spectrum of the transformed image is computed to assess the strength of different frequency components.

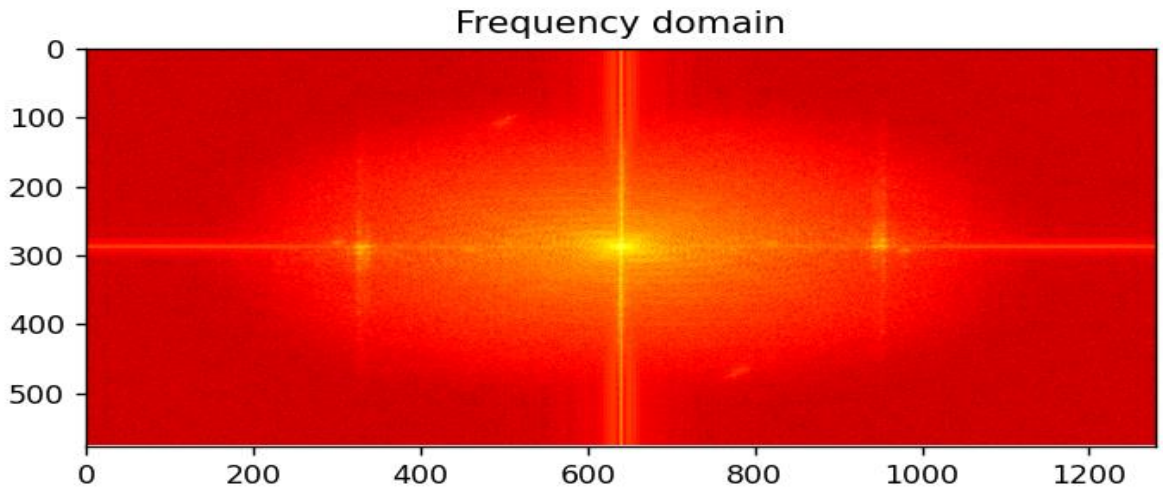



Figure 17. The magnitude spectrum of the transformed image

The thresholding technique selects a percentile (98.7% in this case) to isolate and suppress the highest peaks in the frequency spectrum. These peaks represent repetitive textures or lighting artifacts rather than meaningful surface defects. A binary mask is created to retain only those frequency components below the chosen threshold, effectively filtering out high-frequency noise or texture patterns.

The filtered frequency representation is then inverse transformed back to the spatial domain using np.fft.ifft2() and np.fft.ifftshift(). The resulting image, now with diminished periodic artifacts, enhances the visibility of irregular, non-repeating patterns such as scratches.

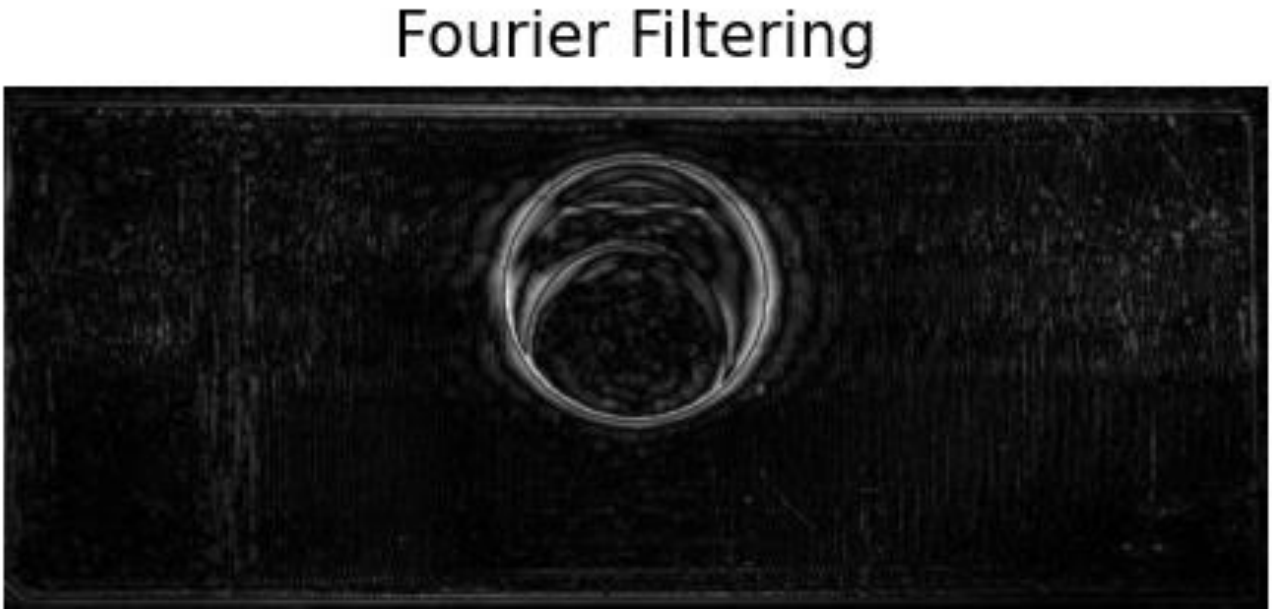



Figure 18. The image after suppressing high frequency peaks to remove repetitive patterns

After frequency filtering and preprocessing, the image is further analyzed for visible linear features that are characteristic of scratches. The next major step in the pipeline uses Canny edge detection, a well-known algorithm that identifies sharp changes in intensity — i.e., edges — which often correspond to surface discontinuities such as scratches.

In the code, cv2.Canny() is called with two threshold values: 20 and 150. These thresholds help define what constitutes a strong or weak edge. The algorithm performs gradient analysis and edge tracking to produce a binary image, where detected edges are marked in white.

To make these edges more coherent and structurally connected, morphological dilation is applied using a 3x3 kernel. Dilation thickens the edges and closes small gaps, which helps consolidate fragmented scratch segments into continuous shapes. This operation is crucial for reliable contour detection later.

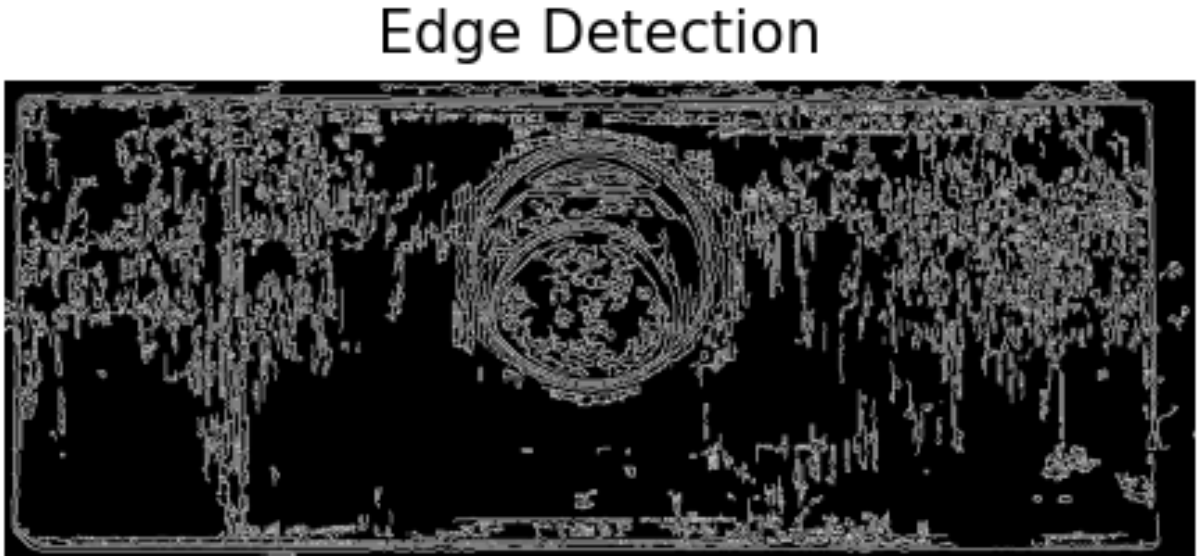
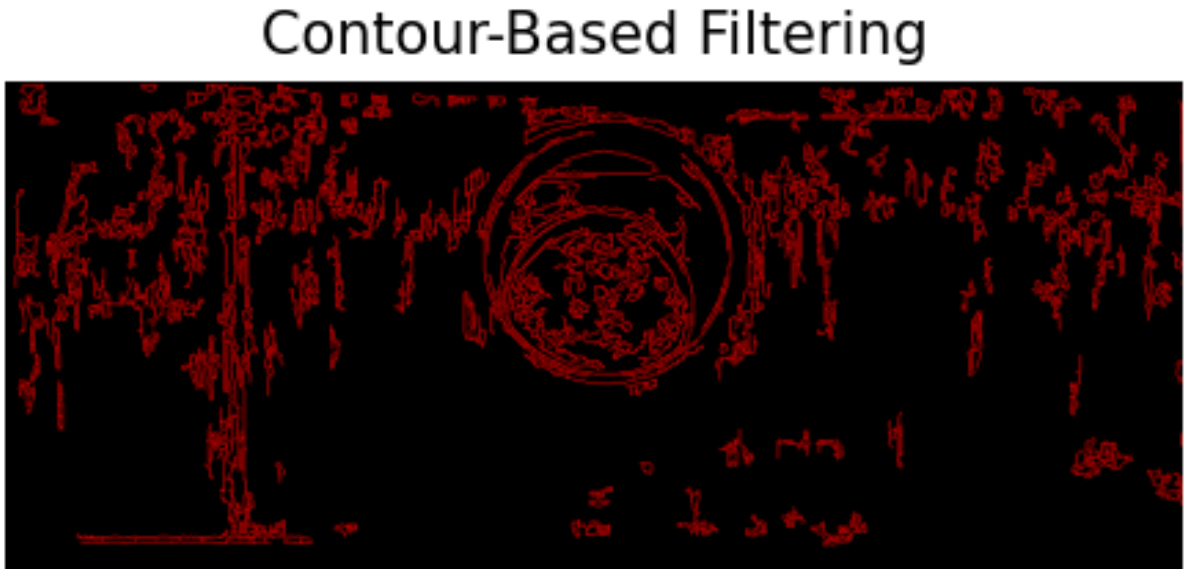



Figure 19. The result of applying Canny edge detection algorithm

The next step uses cv2.findContours() to detect contours, or closed boundaries around detected edge areas. However, not all contours are defects. Many might be noise, texture, or lighting artifacts. Therefore, the code implements contour filtering based on two main properties: area – small contours with area less than 50 pixels are discarded, and aspect ratio – long horizontal or vertical scratches are kept, while overly elongated contours (aspect ratio > 5) or near-square ones are filtered out.




d

Figure 20. The result of contour based filtering with set parameters

This selective process ensures that only meaningful defect-like regions — likely scratches — are passed forward.

Additionally, a more advanced enhancement involves using the Hough Line Transform, which detects straight lines from the edge image. This can be particularly useful in confirming whether the defects follow linear trajectories, which are typical for scratches caused by friction or abrasion.

```
line_mask = np.zeros_like(image)
if lines is not None:
    for line in lines:
        x1, y1, x2, y2 = line[0]
        angle = np.arctan2(y2 - y1, x2 - x1) * 180 / np.pi
        if not (-90 < angle < -70 or 70 < angle < 90) :  # Remove vertical 3D print lines
            cv2.line(line_mask, (x1, y1), (x2, y2), (0, 255, 0), 2)
```

Figure 21. Code snippet demonstrating the filtering out vertical lines because of the texture pattern

The HoughLinesP function helps verify the orientation and presence of scratches, especially when contour filtering is not conclusive.

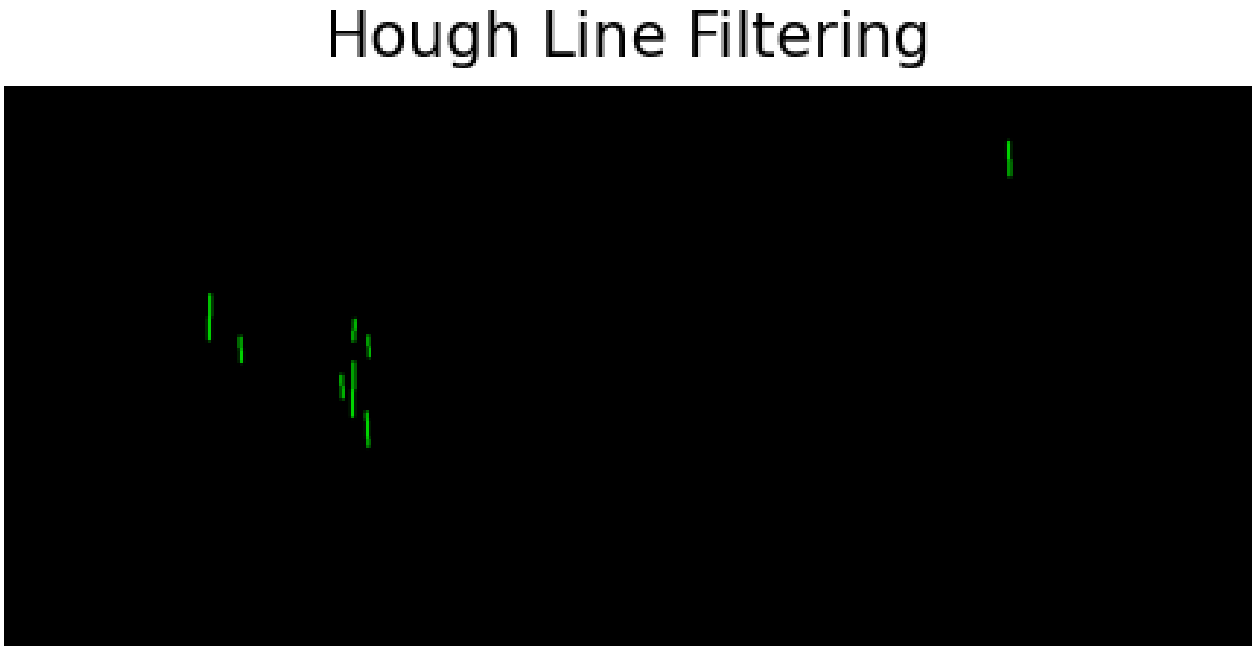



Figure 22. The vertical lines remained after Fourier filtering, and to be suppressed for further analysis

Finally, once valid scratch regions are identified, a red mask overlay is created using cv2.drawContours() and merged with the original image via cv2.addWeighted(). This produces an annotated image where scratches are highlighted — which can either be saved or displayed on a factory UI.

The quantitative output, total_defect_area, is compared to a maximum threshold (e.g., 20% of the product's surface area). If exceeded, the part is marked as defective. This final decision can be logged or sent to a PLC via TCP, closing the loop between machine vision and automation control.

At the end of the image processing and contour filtering stages, the system needs a quantitative measure of how damaged the inspected surface is to define the flag sent to the conveyor belt PLC. This is achieved with the usage of the variable total_defect_area, which adds up the area of all relevant contours that passed the filtering criteria. Each contour represents a candidate scratch region that meets size and shape conditions suggesting it is a real defect and not just texture or noise.

```
total_defect_area = sum(cv2.contourArea(c) for c in contours if cv2.contourArea(c) > 30)
max_possible_area = image.shape[0] * image.shape[1] * 0.2  # 20% of image area assumed as threshold

score = min(100, int((total_defect_area / max_possible_area) * 100))
print(f"Defect score: {score}")
```

Figure 22. Defining the total_defect_area variable. By iterating through all the detected contours, filtering out any small contours (area ≤ 30) to reduce false positives, and adds up the area (in pixels) of all valid defect regions

To contextualize this number the code calculates a maximum acceptable defect area, often defined as a percentage of the total image surface area. This score gives a percentage value representing how much of the product's surface is scratched. If the score is higher or equal to 20, then the part is rejected; if the score is less than 20, then the part is passed. This numerical scoring system allows for flexible, threshold-based quality control decisions. It also enables fine-tuning of sensitivity depending on product type, surface expectations, and tolerance for cosmetic defects. For example, the score of the considered image is 34 meaning

that the object don't pass the threshold value because of scratches it has, and must be labeled as damaged for further manual inspection.

Once the image has been analyzed and defects (e.g. scratches) have been identified, the system needs to send this result to a Programmable Logic Controller (PLC). This communication is done over TCP/IP, where the Python-based vision system acts as a client or server that exchanges data packets with the PLC's Ethernet module. In a typical setup, the vision system plays the client role. After computing the total_defect_area, the system decides whether the product is defective (e.g., defect area exceeds a set threshold). This decision is then encoded as a string and sent to the PLC's listening IP address and port.

```
import socket

HOST = '188.210.0.101'  # IP of the PLC or its gateway
PORT = 5025             # must match PLC socket configuration

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((HOST, PORT))
    decision = "FAIL" if score > 20 else "OK"
    s.sendall(decision.encode())
```

Figure 23. Code snippet for sending a status code to the PLC

### 6.2.2 PLC programming

A PLC used in this project is Siemens Simatic S7-1200. It is used to control the conveyor belt functionality with motor's power width modulation and two magnetic sensors detecting the carrier. These sensors and an actuator are connected to the input and output channels of the PLC, so that the logic programmed in the PLC can be used to start/stop the motor, control its power, and read inputs from the magnetic sensors. Each manufacturer has its software solution to program the provided PLC; Siemens provides totally integrated automation Portal (TIA Portal) to create and modify the logic of Siemens' PLCs.

Programming for PLCs (Programmable Logic Controllers) is quite different from writing standard software. One of the key differences is the importance of timing. In PLC programming, it's critical that the main program loop runs smoothly and completes without delays. This means avoiding any internal loops or operations that could slow down or block the cycle. In traditional software development, it's common to pause execution using loops to wait for input or other events but doing this in a PLC program could halt the entire production process, something that must be avoided at all costs. Therefore, instead of waiting for the input or event, the code checks data once per loop, but the running of this software does not depend on it. It is achieved by tags – an analogy to variables in other programming languages. The tags typically store data in binary format and can be accessed from any part of the software.

Because synchronization and timing are crucial for PLC programming, the original architecture of the station used in the project was stored on a flash drive as a backup, and completely new ladder logic was made to avoid conflicts with the preinstalled code.

The PLC main organization block consists of five networks, which are an analogue to code lines in a standard programming language. The first two networks are responsible for carrier movement, meaning they send a start signal

to the motor until receiving a signal from one of magnetic sensors when a carrier is close to the end of conveyor belt.

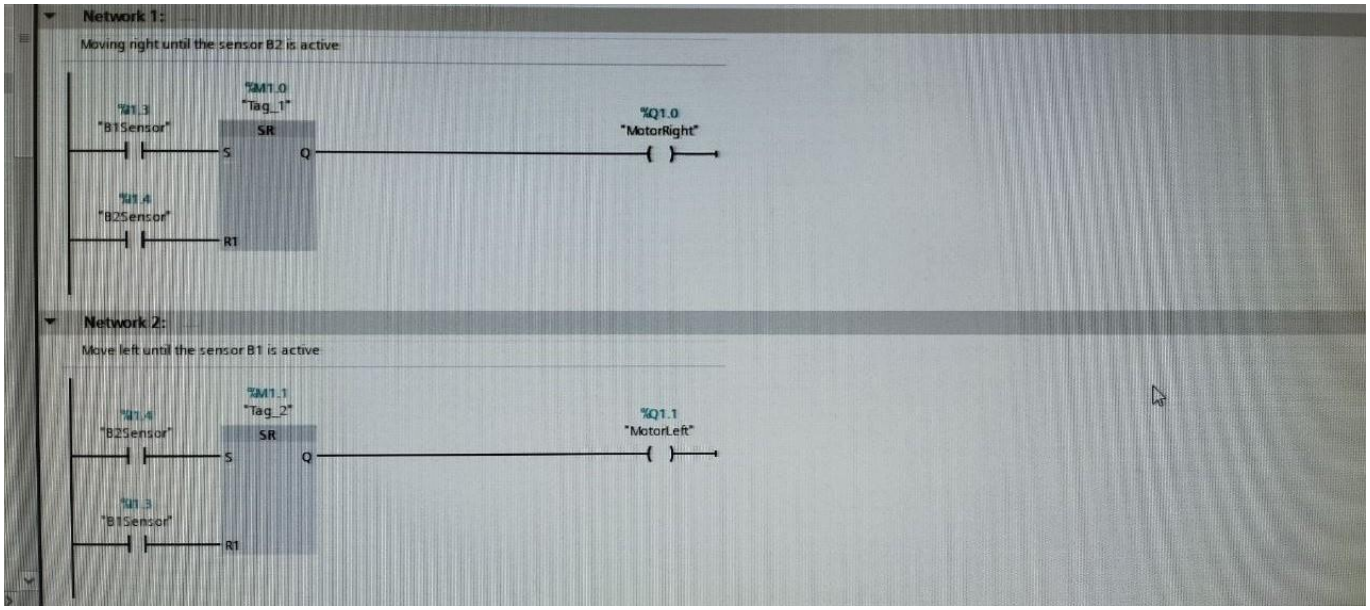



Figure 24. Networks 1&2 responsible for carrier movement

The third and fourth networks are responsible for establishing a TCP connection with a client (industrial computer), receiving a status code which depends on the quality of the object (the number of scratches it has over the surface), and confirming no error has occurred during connection establishment. If the status code represents a high score of scratches, then the PLC's code throws an exception and stops (see FIGURE 25); otherwise, the process continues, and the object moves to the next station.

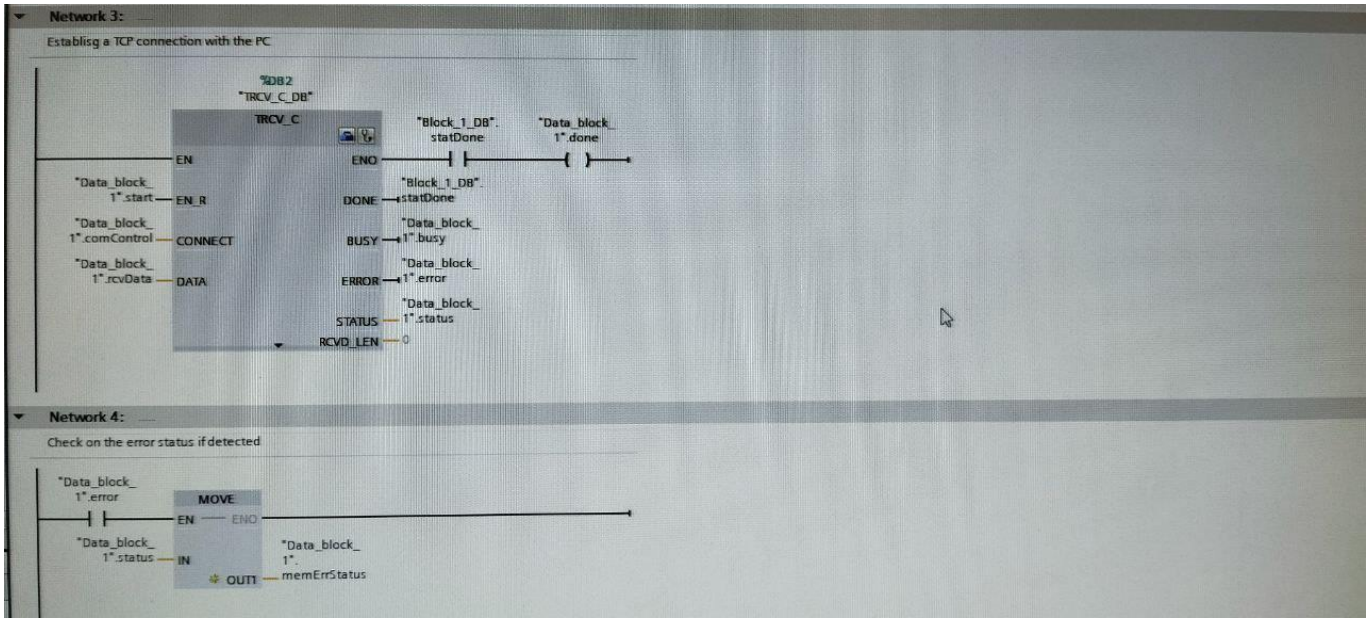



Figure 25. TCP connection establishment

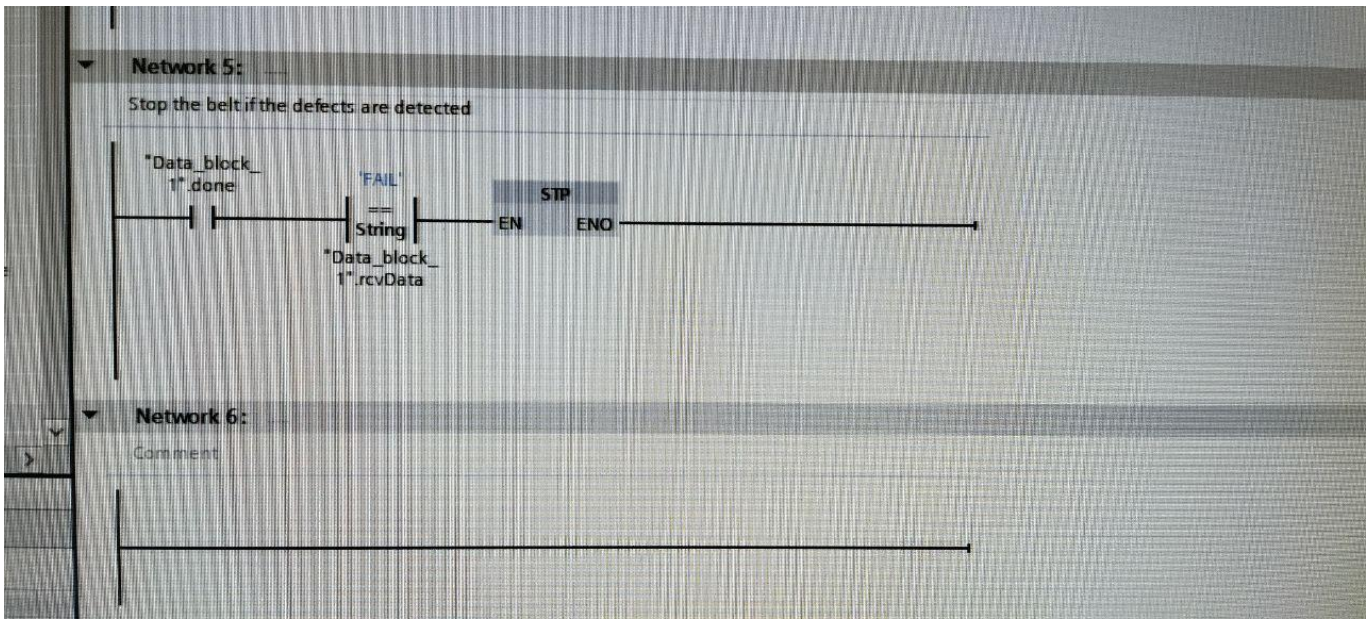



Figure 26. Network 5; Comparing the received message with the status code

# 7 CONCLUSION AND DISCUSSION

This thesis demonstrates the efficiency of combination of classical computer vision approaches with industrial grade equipment. A working proof-of-concept system has been established and verified using industrial-grade PLCs, Basler cameras, and traditional computer vision techniques. Through minimizing the high computational requirement of deep learning and implementing a deterministic processing pipeline, the system can detect faults such as surface defects and scratches in real-time.

Moreover, lightweight AI models could be added to the solution keeping the balance between computation complexity and real-time performance.

# 8 REFERENCES


Abdelkrim Mecharbat, L., Benmeziane, H., Ouarnoughi, H. & Niar, S., 2022. *HyT-NAS: Hybrid Transformers Neural Architecture Search for Edge Devices.* [Online]
Available at: https://arxiv.org/abs/2303.04440
[Accessed 3 8 2024].

Advanced illumination, n.d. *A Practical Guide To Machine Vision Lighting.* [Online]
Available at: https://advancedillumination.com/a-practical-guide-to-machine-vision-lighting/
[Accessed 2 23 2025].

Basler, A., 2022. *Product documentation.* [Online]
Available at: https://www.baslerweb.com
[Accessed 4 2 2024].

Basler, A., n.d. *Basler 10GigE Interface Card, 1 Port.* [Online]
Available at: https://www.baslerweb.com/en/shop/basler-10gige-interface-card-1-port/?srsltid=AfmBOoqwrKJMLj54zhCMqsu_hSGFZ-_8kCExm5Z4hOssRYh7pv0T4U6n
[Accessed 5 2 2025].

Basler, A., n.d. *Basler Ace acA640-90um.* [Online]
Available at: https://machinevisionstore.com/catalog/details/802
[Accessed 10 2 2025].

Bufi, M. & Shehata, R., 2020. *SYSTEM AND METHOD FOR AI VISUAL INSPECTION.* s.l. Patent No. 12243216.

Coates, C. & Juvan-Beaulieu, I., 2020. *Rolling Shutter vs Global Shutter sCMOS Camera Mode.* [Online]
Available at: https://andor.oxinst.com/learning/view/article/rolling-and-global-shutter
[Accessed 5 2 2024].

Dulari, 2024. *A Complete Guide on Hough Transform.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2022/06/a-complete-guide-on-hough-transform/
[Accessed 10 3 2025].

Frejlichowski, D., 2002. Methods for Shape Analysis of Two-Dimensional Closed Contours. *Electronic Journal of Pathology and Histology .*

Ghosh, A., Singh, R. & Kaur, H., 2024. Applications of Machine Learning and Computer Vision in Industry 4.0. *Applied Sciences.*

Gonzalez, W., 2018. *Digital Image Processing.* 4th ed. New York: Pearson.

Hakaja, J., 2022. *Hyvä, parempi, valmis: Opinnäytetyöopas ammattkorkeakouluille.* Helsinki: Gaudeamus.

Hirsjärvi, S., Remes, P., Sajavaara, P. & Sinivuori, E., 2009. *Tutki ja kirjoita.* 15. uud. p. ed. Helsinki: Tammi.

Hütten, N., Hölken, F., Alves Gomes, M. & Andricevic, K., 2024. Deep Learning for Automated Visual Inspection in Manufacturing and Maintenance. *MDPI.*

Junwei Wang, S. X. X. G. X., 2019. Intelligent decision making for service and manufacturing industries. *Journal of Intelligent Manufacturing,* Volume 1, p. 2089–2090.

Kagermann, H. W. W. &. H. J., 2013. *Recommendations for implementing the strategic initiative INDUSTRIE 4.0,* s.l.: s.n.

Lasi, H. et al., 2014. Industry 4.0. *Springer Nature,* Volume 6, pp. 239-242.

Liang, J., n.d. *Canny.* [Online]
Available at: https://justin-liang.com/tutorials/canny/
[Accessed 6 3 2025].

Li, J. & Chen, M., 2024. DEW-YOLO: An Efficient Algorithm for Steel Surface Defect Detection. *Applied Sciences.*

Maaz, M., Shaker, A., Cholakkal, H. & Khan, S., 2022. EdgeNeXt: Efficiently Amalgamated CNN-Transformer Architecture for Mobile Vision Applications.

Moon, G., n.d. *How Calculate Polygon Area in a Software.* [Online]
Available at: https://www.deepblock.net/blog/polygon-area
[Accessed 16 3 2025].

OpenCV, n.d. *Contour Features.* [Online]
Available at: https://docs.opencv.org/4.x/dd/d49/tutorial_py_contour_features.html
[Accessed 15 3 2025].

OpenCV, n.d. *Contours : More Functions.* [Online]
Available at: https://docs.opencv.org/4.x/d8/d1c/tutorial_js_contours_more_functions.html
[Accessed 14 3 2025].

Oto Haffner, E. K. D. R., 2024. Applications of Machine Learning and Computer Vision in Industry 4.0. *Applied Sciences.*

Pietrzkiewicz, P., 2019. Contour Analysis. In: *Image Analysis Techniques for Industrial Inspection Systems.* s.l.:s.n., pp. 73-83.

Simulink, M. &., n.d. *Hough Transform.* [Online]
Available at: https://se.mathworks.com/help/images/hough-transform.htm
[Accessed 10 3 2025].

Spring, K., Fellers, T. & Davidson, M., n.d. *Introduction to Charge-Coupled Devices (CCDs).* [Online]
Available at: https://www.microscopyu.com/digital-imaging/introduction-to-charge-coupled-devices-ccds
[Accessed 1 3 2025].

Tamron, n.d. *Lens, Tamron, M112FM16, 16 mm, 1/1.2".* [Online]
Available at: https://en.ids-imaging.com/store/lens-tamron-m112fm16-16-mm-1-1-2.html
[Accessed 15 2 2024].

Turchetta, R., Spring, K. & Davidson, M., n.d. *Introduction to CMOS Image Sensors.* [Online]
Available at: https://evidentscientific.com/en/microscope-resource/knowledge-hub/digital-imaging/cmosimagesensors
[Accessed 1 3 2025].

Wang, S. W. J. Z. D. L. D. &. Z. C., 2016. Computer Networks. In: *Towards smart factory for Industry 4.0: A self-organized multi-agent system with big data-based feedback and coordination.* s.l.:s.n., pp. 158-168.

Zhang, Z., Zhang, W., Zhu, D. Z. & Xu, Y., 2023. Printed circuit board solder joint quality inspection based on lightweight classification network. *IET Cyber Systems and Robotics.*

Zheng, P. W. H. S. Z. Z. R. L. Y. L. C. &. X. X., 2018. Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives. *Frontiers of Mechanical Engineering,* Volume 13, p. 137–150.