UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Faculty of Electrical Technology and Engineering

DEVELOMENT OF AN AUTOMATED PILL MEDICINE DEFECT SYSTEM USING CAMERA AND WEB APPLICATION

MUHAMMAD DENIAL BIN ANANG SYAFRIZAL ANWAR

Bachelor of Electrical Engineering Technology (Industrial Automation & Robotics) with Honours

2024

DEVELOMENT OF AN AUTOMATED PILL MEDICINE DEFECT SYSTEM USING CAMERA AND WEB APPLICATION

MUHAMMAD DENIAL BIN ANANG SYAFRIZAL ANWAR

A project report submitted
in partial fulfilment of the requirements for the degree of
Bachelor of Electrical Engineering Technology (Industrial Automation & Robotics)
with Honours

Faculty of Electrical Technology and Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

DECLARATION

I declare that this project report entitled “Development of an Automated Pill Medicine defect System Using Camera and Web Application” is the result of my own research except as cited in the references. The project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature :

Student Name : Muhammad Denial Bin Anang Syafrizal Anwar

Date : 27/2/2025

DEDICATION

This project is dedicated to my parents, family, and friends for their unwavering support and encouragement throughout this journey.

I also dedicate this work to my supervisor, whose guidance, patience, and expertise have been instrumental in overcoming challenges and shaping my approach to problem-solving. His invaluable mentorship has significantly enriched my learning experience.

This project reflects the passion, persistence, and teamwork I have been fortunate to experience along the way, as well as the dedication to seeing this endeavour through to completion.

Last but not least, I wanna thank me, I wanna thank me for believing in me, I wanna thank me for doing all this hard work, I wanna thank me for having no days off, I wanna thank me for, for never quitting.

Thank you for everything

ABSTRACT

Automated defect detection systems have revolutionized industrial quality control by enhancing efficiency and consistency. This project introduces an automated pill inspection system for pharmaceutical manufacturing, aiming to improve product quality and safety. The system integrates a camera-based inspection mechanism, a web application interface, a Raspberry Pi controller, and a DC motor-driven conveyor. By utilizing OpenCV, the system processes high-resolution images captured by a connected camera in real-time to detect defects such as size deviations, colour inconsistencies, and shape irregularities. Specifically, the system categorizes pills into three groups Pill A (Paracetamol) in the yellow-green category, Pill B (Aspirin) in the white-blue category, and Pill C (Adezio) in the red-black category. Initial tests demonstrated a defect detection accuracy of 86%, with the system inspecting up to 30 pills per test by continuously refilling the dispenser, which has a single-fill capacity of 15 pills. The web application enables real-time monitoring, data visualization, and defect categorization, fostering informed decision-making. A servo-controlled dispenser ensures defective pills are promptly removed from the production line. Preliminary results highlight the system's potential for practical implementation in pharmaceutical production, addressing industry demands for automation while contributing to sustainable manufacturing practices through reduced product waste.

ABSTRAK

Sistem pengesanan kecacatan automatik telah merevolusikan kawalan kualiti industri dengan meningkatkan kecekapan dan konsistensi. Projek ini memperkenalkan sistem pemeriksaan pil automatik untuk pembuatan farmaseutikal, yang bertujuan untuk meningkatkan kualiti produk dan keselamatan. Sistem ini mengintegrasikan mekanisme pemeriksaan berasaskan kamera, antara muka aplikasi web, pengawal Raspberry Pi, dan penghantar yang digerakkan oleh motor DC. Dengan menggunakan OpenCV, sistem ini memproses imej resolusi tinggi yang ditangkap oleh kamera yang disambungkan secara masa nyata untuk mengesan kecacatan seperti penyimpangan saiz, ketidakkonsistenan warna, dan keabnormalan bentuk. Secara khusus, sistem ini mengkategorikan pil kepada tiga Kumpulan Pil A (Paracetamol) dalam kategori kuning-hijau, Pil B (Aspirin) dalam kategori putih-biru, dan Pil C (Adezio) dalam kategori merah-hitam. Ujian awal menunjukkan ketepatan pengesanan kecacatan sebanyak 86%, dengan sistem memeriksa sehingga 30 pil setiap kitaran ujian melalui pengisian semula dispenser yang mempunyai kapasiti maksimum 15 pil. Aplikasi web membolehkan pemantauan masa nyata, visualisasi data, dan pengkategorian kecacatan, yang menyokong pembuatan keputusan yang berinformasi. Dispenser yang dikawal servo memastikan pil yang rosak dikeluarkan dengan segera daripada barisan pengeluaran. Hasil awal menunjukkan potensi sistem untuk pelaksanaan praktikal dalam pengeluaran farmaseutikal, memenuhi permintaan industri terhadap automasi sambil menyumbang kepada amalan pembuatan yang mampan melalui pengurangan sisa produk.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my supervisor, Ts. Aminurrashid Bin Noordin for the precious guidance, words of wisdom and patient throughout this project.

I am also deeply indebted to Universiti Teknikal Malaysia Melaka (UTeM) for providing the support that enabled me to accomplish this project

My highest appreciation goes to my parents and family members for their love and prayer during the period of my study.

Lastly, I extend my sincere thanks to my classmates, the faculty members, and everyone who has contributed, directly or indirectly, to the success of this project. Your support has been truly invaluable.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

FP	-	False Positive
3D	-	3 Dimensions
FN	-	False Negative
%	-	Percentage

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

The project centers on the development of an automated production line for pill medicine defect checking, leveraging a combination of camera technology and a web application interface. Traditional quality control methods in pharmaceutical manufacturing often rely on manual inspection processes, which can be labor-intensive and may be prone to human error. To address these challenges, the project aims to implement an automated system that streamlines defect detection processes and enhances accuracy.

The automated production line integrates a camera system, strategically positioned along the production line, to capture images of pill medicines in motion. These images are then analyzed using advanced image processing algorithms to identify defects such as size variations, colours discrepancies, and inconsistencies. The system also incorporates a web application interface, allowing users to monitor the defect detection process in real-time, manage system settings, and receive alerts promptly.

The project encompasses several phases, including planning and design, hardware setup, software development, integration and testing, and ongoing monitoring. In the planning and design phase, considerations will be made for the layout of the production line, camera placement, and system requirements. The hardware setup will involve the installation of the camera system and conveyor belt mechanism to facilitate the seamless transportation of pills for inspection.

Software development will focus on creating image processing algorithms and web application interface for data visualization and user interaction. Integration and testing will ensure the seamless operation of the system, including accurate detection and efficient communication between components. Ongoing monitoring and maintenance will involve regular system checks, updates, and optimization to ensure continued performance and reliability.

Overall, the project aims to revolutionize quality control methods in pharmaceutical manufacturing by automating the pill medicine defect checking process. By leveraging camera technology and a web application interface, the automated production line offers enhanced efficiency, accuracy, and real-time monitoring capabilities, ultimately improving product quality and patient safety. Figure 1.1 shows the overview of the project.

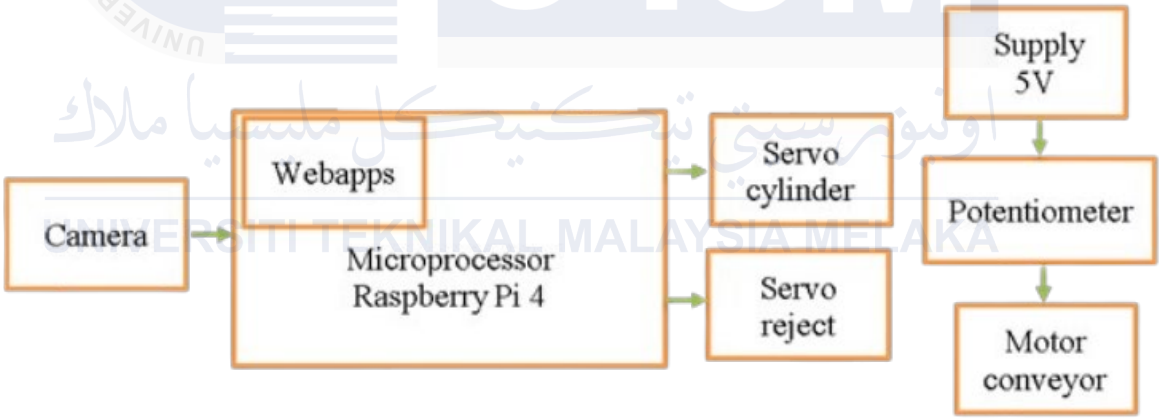


Figure 1.1 Overview of The Project

1.2 Problem Statement

In pharmaceutical manufacturing, reliance on manual inspection for quality control in pill production poses significant challenges, including inefficiencies, inconsistencies, and the potential for distributing defective products. These challenges not only increase production costs but also raise societal and global concerns regarding public health, as defective products can lead to adverse health outcomes and undermine trust in healthcare

systems. Ensuring medication safety and efficacy is crucial for protecting public health and maintaining societal well-being. Addressing these quality control challenges contributes to the SDGs, particularly Goal 3 (Good Health and Well-being).

To address these challenges, this project aims to develop an automated production line for pill medicine defect checking. Leveraging advanced technologies such as cameras and image processing algorithms, the system will enable seamless inspection of pills in motion. By integrating a conveyor belt with strategically positioned cameras, the system will capture high-resolution images for detailed defect analysis. Sophisticated image processing techniques will detect and classify various defects, including size discrepancies, colour variations, and structural abnormalities. This automated approach enhances defect detection accuracy and efficiency, minimizing the risk of defective products entering the market.

Furthermore, the project will include a user-friendly web application interface for comprehensive data monitoring. Operators will have access to detailed reports and analytics, enabling them to track production metrics and performance trends. This data monitoring capability empowers operators to make informed decisions and system adjustments, ensuring optimal performance and product quality. By streamlining the defect checking workflow and facilitating proactive decision-making, the automated production line will improve manufacturing efficiency and bolster product quality assurance, benefiting both pharmaceutical companies and consumers.

In addition to improving public health outcomes, the automated inspection system aligns with environmental sustainability goals. Reducing defective product waste and optimizing the production process contribute to more sustainable manufacturing practices. This project supports SDG 12 (Responsible Consumption and Production) by promoting efficient resource use and minimizing waste, thereby reducing the environmental footprint of pharmaceutical manufacturing.

1.3 Project Objective

The objectives of the project are as follows:

1. To design and implement automated pill medicine defect checking system.
2. To create webapps for data visualization for providing clear and comprehensive insights.
3. To develop and integrate an image processing algorithm capable of accurately analysing images captured by the camera

1.4 Scope of Project

The scope for the project is as follows:

1. Use Raspberry Pi 4 as the central processing unit for the system.
2. develop or integrate belt system to transport pills under the camera for continuous inspection.
3. Control the conveyor speed using a potentiometer.
4. Using 2 colour combination pill
5. Ensure a reliable power supply to the Raspberry Pi and other connected components.
6. Display key performance such as the number of pills inspected, the image of defective pills identified, and the types of pills defects detected.
7. Implement real-time monitoring features to showcase the inspection process, including visual displays of detected defects with corresponding images.
8. Only detect top surface of the pill only.
9. Limited to carrying only 14 pills in the dispenser.
10. Develop an image processing algorithm using OpenCV library.

11. Identify key defects to be detected, such as discoloration, and abnormal pill.
12. Ensure consistent and uniform lighting conditions to minimize the impact of shadows and reflections.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, a comprehensive literature review is conducted to explore previous developments in image processing techniques for defect detection in automated systems. By examining reliable sources such as websites, articles, and journals, this review provides insights into the methods, algorithms, and technologies that have been employed to enhance the accuracy and efficiency of defect detection processes. Additionally, it highlights key advancements in camera-based detection mechanisms, traditional image processing methods, and control algorithms that have contributed to the progress in this field.

2.2 Mechanism of Image Processing

The main concept of image processing is the systematic approach to transforming visual data into meaningful information. This process involves a series of steps that allow for the efficient analysis and interpretation of images for various applications, such as defect detection and object recognition. Image processing employs several techniques, each with specific roles and purposes, which work together to achieve the desired outcomes. These techniques are categorized into four key mechanisms:

1. Image Acquisition
2. Preprocessing
3. Segmentation
4. Object Recognition

2.2.1 Image Acquisition

Image acquisition is the initial step in image processing, involving the capture of images from a device or sensor such as a camera, scanner, or specialized imaging equipment. This process includes setting up appropriate lighting, focusing, and configuring the hardware to collect visual data and converting it into a digital format suitable for storage and analysis. It plays a crucial role in applications like quality control, medical imaging, robotics, and surveillance, providing the raw input for subsequent processing techniques. In an automated pill defect detection system, image acquisition involves capturing high-quality images of pills to enable accurate defect identification using traditional image processing methods. Figure 2.1 shows an example of the steps in image acquisition, starting from Acquisition Artifacts, which involve the Colour Filter Array (CFA), followed by post-processing to convert the captured data into a digital image, and lastly, the storage stage.

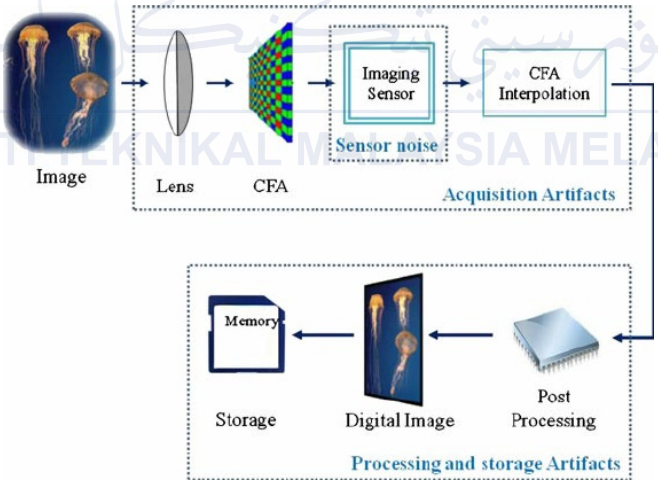


Figure 2.1 Image Acquisition

In recent advancements, Huitong Xu et al. [1] introduced an innovative approach to tunnel lining crack detection by integrating a mobile image acquisition system with a deep learning ensemble model. This research addresses the limitations of traditional manual inspections, which are often time-consuming and hazardous, by providing an efficient and accurate solution. The system leverages the YOLO-LD model, incorporating advanced

techniques such as large separable kernel attention and dynamic snake convolution to significantly enhance crack detection performance.

Similarly, Keelson et al. [2] conducted a study on the application of cardiac CT acquisition modes in dynamic musculoskeletal (MSK) imaging, demonstrating a significant reduction in motion artifacts compared to conventional cine modes. Using a rotating PMMA phantom, their research quantitatively evaluated the impact of various scanning techniques, revealing that the cardiac mode produced lower Jaccard distance values, indicating fewer motion artifacts. The findings underscore the advantages of cardiac CT acquisition for improving imaging quality in dynamic MSK applications.

Zhenjun Dai [3] carried out a study focused on optimizing the You Only Look Once (YOLO) algorithm for processing drone images, specifically for inspecting wind turbine blades. The research underscores the growing importance of wind energy as a renewable resource and the critical need for efficient maintenance and damage detection methods. The study involved preprocessing drone-captured images of blade damage using techniques like deblurring, noise reduction, and image enhancement to improve detection accuracy.

2.2.2 Preprocessing

Preprocessing in the context of image processing refers to the set of techniques applied to raw image data to enhance its quality or make it more suitable for analysis and further processing. It serves as a preparatory step to improve the efficiency and accuracy of subsequent tasks like feature extraction, classification, or object detection. Figure 2.2 show the example of the preprocessing.

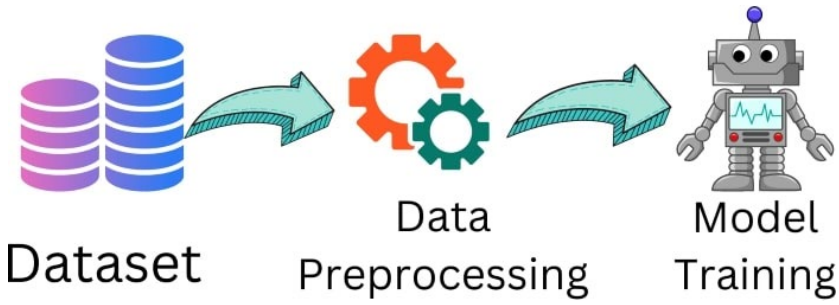


Figure 2.2 Preprocessing Technique

In [4], Dall Al-Alimi et al. proposed a novel framework to enhance forensic blood detection using hyperspectral imaging (HSI) and advanced preprocessing techniques. Their study addresses the limitations of traditional blood detection methods, which often suffer from low specificity and susceptibility to false positives. By utilizing molecular spectroscopy, the authors emphasize the importance of identifying the unique spectral fingerprints of blood derived from its molecular composition, to improve accuracy and reliability in forensic applications.

K.U. Ahamed et al. [5] explored a deep learning approach that leverages effective preprocessing techniques to detect COVID-19 from chest CT-scan and X-ray images. Their study highlights the critical role of preprocessing in improving the quality of medical images before they are processed by deep learning models. The authors utilized various image processing techniques, such as cropping and sharpening filters, to enhance the clarity and relevance of the images, which is essential for accurate diagnosis. The preprocessing steps involved cropping CT-scan images to focus solely on lung regions, thereby eliminating irrelevant parts of the images. Figure 2.3 shows the journal results using the technique applied.

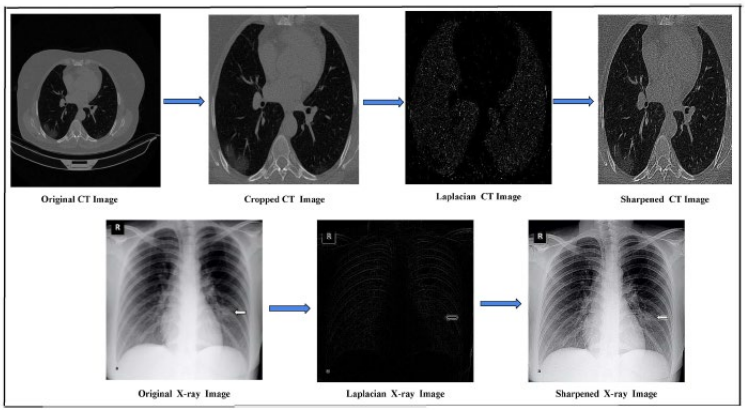


Figure 2.3 Results Tested using The Technique

2.2.3 Segmentation

Segmentation in image processing refers to the process of dividing an image into multiple regions or segments that represent distinct objects, areas, or features. This technique simplifies complex images by identifying and isolating specific components, making them easier to analyze and interpret. Segmentation can be categorized into various types, such as semantic segmentation, which classifies each pixel into different classes, for example people and vehicles, and instance segmentation, which distinguishes between individual objects of the same class. Additionally, region-based and edge-based segmentation methods are used to group pixels based on similarity in properties like intensity, colour, or texture. Figure 2.4 shows the technique of segmentation.

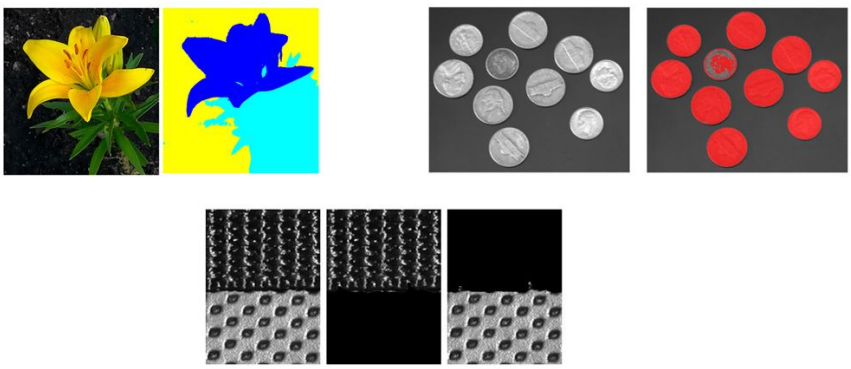


Figure 2.4 Segmentation of flowers, coins, and texture regions based on their textures, illustrating the effectiveness of texture-based methods.

In [6], Yinkai Fu et al. present a significant advancement in the field of soil pore segmentation through their proposed weakly supervised soil pore segmentation method (WSSPS), which integrates traditional segmentation algorithms to generate pseudo-labels for pre-training. Recognizing the challenges associated with extensive manual labelling in image segmentation tasks, particularly in soil research, the authors utilize weakly supervised learning (WSL) to reduce the dependency on expert-defined labels.

In [7], Jonathan S. Cárdenas-Gallegos et al. conducted a study aimed at improving the segmentation of plant pixels in hydroponic lettuce images captured under variable greenhouse illumination conditions. Their research focused on enhancing plant pixel identification, particularly in the face of inconsistent lighting that affects pixel uniformity. To achieve this, they compared three segmentation techniques: Otsu segmentation, a traditional method based on a single colour threshold; Random Forest classification, a machine learning approach that utilizes multiple features for pixel classification; and U-Net Convolutional Neural Network (CNN), an advanced deep learning model designed for semantic segmentation.

In their systematic review, Ghobadi et al. [8] investigates the challenges and solutions associated with deep learning-based automated liver segmentation. The authors categorize the challenges into five main groups: medical image-related challenges, such as limited training images and noise hardware-related challenges, including high computational costs liver structure and intensity-related challenges, which encompass ambiguous liver borders and variability in liver shape and size implementation-related challenges, such as information loss in network layers and other miscellaneous challenges. The review discusses various methods and techniques proposed in the literature to overcome these challenges, thereby enhancing the accuracy and efficiency of liver segmentation.

2.2.4 Object Recognition

Object recognition refers to the process of identifying and classifying objects within an image or scene. It involves detecting objects, categorizing them into predefined classes, and determining their location. The goal is to enable machines to understand and interpret visual data by recognizing and understanding the objects present. Object recognition can be achieved through various techniques, including traditional computer vision methods like feature-based approaches and advanced deep learning models such as convolutional neural networks (CNNs), which are capable of learning hierarchical features from raw images. Figure 2.5 shows an example using object recognition techniques applied in detecting items indoors, such as sofas, pillows, etc.

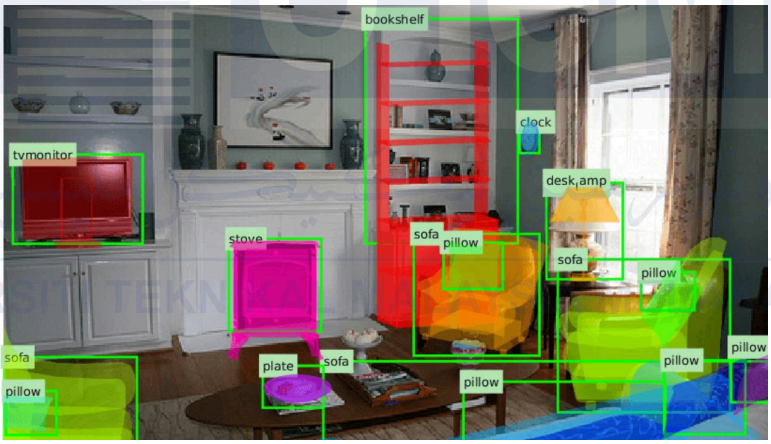


Figure 2.5 Object Recognition Apply in living Room

In Faseeh et al. [9], the authors present an advanced real-time object recognition system focused on enhancing short-range depth estimation. The study integrates temporal information from video sequences and attention mechanisms to improve depth estimation accuracy in dynamic and adaptive environments. Their research addresses the critical need for effective object recognition in military and emergency applications, filling a significant gap in current methodologies.

The paper by A.A.M. Muzahid et al. [10] is published, focuses on the intersection of neuroscience and computational methods, particularly in artificial intelligence and machine learning. It covers a wide range of topics, including deep learning and neural networks, emphasizing advancements in architectures and algorithms applicable to various problems, such as 3D object recognition. The journal explores the application of machine learning techniques across different domains, including computer vision, natural language processing, and robotics, highlighting innovative approaches and methodologies.

Qiang Zhang et al. [11] focused their research on flexible electronic devices, specifically in the areas of sensors and robotics. It covers various topics, including the development of electronic skin, piezoresistive and piezo capacitive sensors, and their applications in enhancing robotic systems. The research seeks to improve the sensory capabilities of robots, enabling them to interact more effectively with their environment and perform complex tasks, such as object recognition and grip strength measurement.

2.3 Techniques for Image Processing Using OpenCV

OpenCV, as described by [12], refers to the Open-Source Computer Vision Library, a powerful programming library designed for real-time computer vision applications. It is widely used for image processing, machine learning, and computer vision tasks. The library supports various programming languages, including Python, C++, and Java, and is known for its extensive capabilities in recognizing human beings, faces, and handwriting, among other features. Additionally, OpenCV is an essential tool for processing photographs and videos, making it a significant resource in today's technological landscape. In the context of my project, OpenCV is utilized for implementing advanced image processing techniques such as contour detection, edge detection, and image filtering which contribute to efficient defect detection and object recognition.

2.3.1 Contour detection

Contour detection in image processing is the process of identifying and outlining the boundaries of objects or regions within an image. It plays a crucial role in separating distinct objects from their backgrounds, allowing for better analysis and understanding of the image. Contours are curves that represent these boundaries, created by detecting areas where there are changes in intensity or color values. This is achieved using algorithms that trace and analyze the contours based on their properties, such as area, perimeter, and shape.

a) Cylinder Detection: Contour detection can be effectively used to identify cylindrical objects by focusing on the curved boundaries. Algorithms analyze the contour properties to match the expected shape, ensuring that areas with similar curvature are detected as cylinders. Techniques such as Hough Transform, or elliptical fitting can be employed to enhance the accuracy of cylindrical shape detection.

b) Rectangular Detection: For rectangular objects, contour detection focuses on straight edges and corners. Using edge-based methods and geometric properties (e.g., aspect ratio), rectangular shapes can be distinguished from other shapes. Algorithms like the Convex Hull or contour approximation techniques are used to precisely outline rectangular regions.

c) Square Detection: Square shapes can be detected by examining contours for equal side lengths and right angles. By analyzing the contour properties such as area, perimeter, and symmetry, a square can be distinguished from other quadrilateral shapes. Techniques like shape fitting or contour approximations can be used to accurately outline squares.

In practical terms, contour detection involves applying techniques such as edge detection to highlight boundaries, followed by contour extraction to outline these regions. Figure 2.6 shows the contour detection using Open CV to detect phone and pencils.

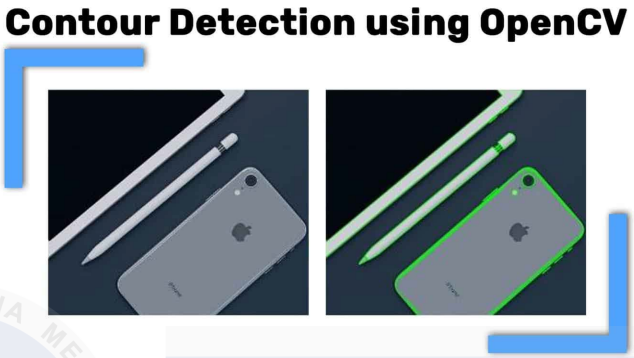


Figure 2.6 Contour Detection Technique Detect Phone and pencils

Contour Detection plays a crucial role in a variety of applications, including the study of phenomena such as subcooled boiling in microgravity environments. In the work by Xenophon Zabulis et al. [13], contour detection is employed to analyze and understand the dynamics of bubble shapes under unique fluidic conditions. This research is significant due to the impact of microgravity on fluid behaviours, where traditional fluid dynamics are altered, making accurate detection and measurement of bubble contours essential for understanding boiling processes in such environments. Additionally, contour detection is a key element in applications like feature extraction for biometric systems. The journal article by Rubab Mehboob et al. [14] highlights the development of a novel feature extraction method, BiRi-PAD, designed to enhance fingerprint liveness detection. Contour detection, through algorithms provided by libraries such as OpenCV, is critical in extracting meaningful features from images, ensuring that biometric systems can accurately distinguish live fingers from spoofed attempts.

The journal article by Ju Jian Lv et al. [15] emphasizes the integration of biomedical engineering with signal processing, highlighting the development and application of sophisticated algorithms for analyzing biomedical signals and images. A significant focus of this research is on processing medical images, including MRI, CT scans, and ultrasound images. Techniques such as segmentation, feature extraction, and image enhancement are explored to enhance diagnostic accuracy and provide more reliable interpretations. These methodologies are essential for improving the quality and precision of medical diagnostics, making them invaluable tools in modern healthcare.

2.3.2 Edge Detection

Edge detection is a fundamental technique in image processing used to identify the boundaries or transitions between different regions in an image. It highlights areas of rapid intensity change, which are often associated with the edges of objects, textures, or other features. This process helps in extracting the outline of objects, enabling tasks such as object recognition, shape analysis, and feature extraction. Edge detection is commonly employed in applications like object tracking, scene understanding, and image segmentation, where identifying clear boundaries between elements is crucial for accurate analysis and interpretation. Figure 2.7 example of edge detection using the Prewitt method. The original image (left) and the processed result (right) highlight edge features.

Figure 2.7 Edge Detection

Zhiyi Pan et al. [16] presents a comprehensive study on category-agnostic semantic edge detection, marking a significant advancement in the field of computer vision. It highlights the evolution of semantic edge detection methods, emphasizing the shift from traditional category-aware approaches, which rely heavily on detailed category annotations, to more flexible category-agnostic techniques. These techniques overcome the constraints of relying on specific categories, making edge detection systems more robust and adaptable to diverse scenarios. The journal article by Le Wang et al. [17] presents a comprehensive exploration of advanced edge detection techniques, specifically focusing on a novel scheme that integrates Fourier single pixel imaging with spiral phase contrast imaging. The study emphasizes the use of sinusoidal speckle patterns for illumination, allowing for the reconstruction of high-quality edge images from sub-Nyquist measurements, thereby significantly reducing the number of required measurements.

2.3.3 Image Filtering

Image Filtering refers to the process of applying a filter to an image to modify its appearance or enhance specific features. Filters are used to reduce noise, sharpen edges, enhance contrast, or smooth out certain areas. In essence, image filtering helps in improving the visual quality of an image or extracting meaningful information from it. Techniques such as Gaussian blur, median filtering, and sharpening filters are common examples used to perform image filtering. These methods are widely used in applications like object detection, medical imaging, and image restoration. Figure 2.8 illustrates the process of image filtering for flowers, starting from the original image filtering to noise removal, with different results filtered at r =5, r=10, r = 20. As r increases, the result becomes smoother.

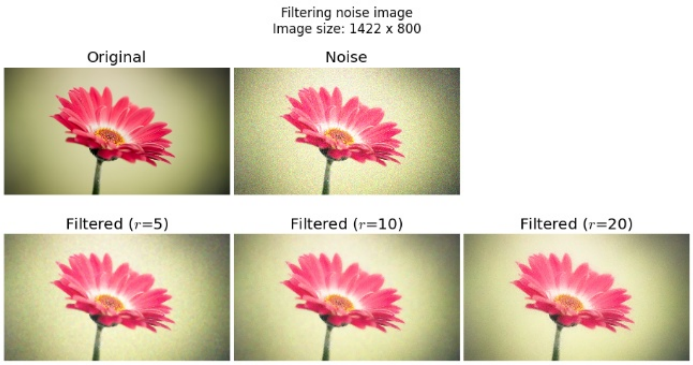


Figure 2.8 The Process of Image Filtering Applied to Flower Image

The study by Yongfei Yu et al. [18] presents a hybrid noise filtering algorithm for colour images, utilizing deep convolutional neural networks (CNNs) enhanced by evolutionary strategies. It addresses the issue of noise interference, which can degrade image quality and hinder accurate recognition and segmentation. By mapping noise points to feature space through nonlinear mapping and employing linear regression, the study filters noise while preserving image clarity. The journal by Shukla Mondal et al. [19] reviews advancements in content-based image filtering (CBIF) and retrieval systems, with a particular focus on techniques for obscene image detection. It highlights the rise in pornographic content online and associated risks for vulnerable populations like adolescents. The review covers methodologies, including feature extraction for human skin and body segmentation, essential for effective image classification.

The H. Jia et al. [20] article presents a study on advancements in guided image filtering techniques, introducing the Weighted Guided Image Filter with Entropy Evaluation Weighting (EEW-WGIF). The authors explore modifications like the Weighted Guided Image Filter (WGIF), which uses edge-aware weighting to dynamically adjust regularization parameters.

2.4 Web applications

A "web app" (web application) refers to a software application that users access through a web browser over the internet, allowing them to perform various tasks without needing to install software on their local devices [21]. In the context of the smart drip irrigation system discussed in the document, the web app plays a crucial role by enabling users to monitor real-time data from sensors that measure soil moisture, temperature, and humidity. It also allows remote control of the irrigation process, such as activating or deactivating pumps based on the sensor data. Additionally, the web app facilitates the storage of this sensor data in the cloud, enabling users to analyze historical data and track the health of their crops over time. Overall, the web app enhances the functionality of the smart irrigation system by providing a user-friendly interface for effective management and monitoring of the irrigation process. Table 2.1 shows the comparison of applications based on AI techniques for image processing using OpenCV.

Table 2.1 Comparison Applications Based on AI Techniques for Image Processing Using OpenCV

AI Technique	Reference	Study Focus	Key Features	Weakness
Contour Detection	[13]	Bubble dynamics, feature extraction	Boundary outlining, object separation	Sensitive to noise, computationally intensive
	[14]	Fingerprint liveness detection	Contour-based feature extraction, enhanced accuracy	Sensitive to variations, dependency on preprocessing
	[15]	Medical imaging (MRI, CT, Ultrasound)	Segmentation, feature extraction, image enhancement	Needs precise preprocessing, computationally intensive
Edge Detection	[16]	Semantic edge detection	Flexible, adaptable to diverse scenarios, category-independent	May require more computational power for large-scale images
	[17]	High-quality edge reconstruction	Reduces number of measurements, high resolution from sub-Nyquist data	Computationally intensive, limited to specific imaging setups
Image Filtering	[18]	Noise reduction in colours images	Filters noise, preserves image clarity	May require extensive training data, slower processing
	[19]	Obscene image detection	Feature extraction for human skin/body segmentation	Limited by sensitivity to lighting variations
	[20]	Image enhancement, edge-preserving tasks	Edge-aware weighting, improves detail and structure	Complex parameter tuning, high computation cost

2.5 Summary

In image processing, image acquisition serves as the foundational step, capturing high-quality visual data essential for subsequent processing. Based on previous studies, this project will focus on developing an automated pill defect detection system using traditional image processing techniques. This process ensures that images, such as those used in automated pill defect detection systems, are optimized for accurate analysis. After acquiring the images, preprocessing techniques refine the data by improving clarity and eliminating irrelevant noise, preparing the images for tasks like feature extraction and object recognition. Segmentation then divides the images into distinct regions or objects, simplifying complex visuals for detailed examination. This step is critical in fields like medical imaging and dynamic musculoskeletal analysis.

Object recognition further enhances image processing by identifying and categorizing objects within the images. Using advanced deep learning models and traditional computer vision methods, object recognition enables precise detection and classification. OpenCV, as a versatile tool, facilitates various image processing tasks, including contour detection, edge detection, and image filtering, ensuring that images are suitable for both research and real-world applications. These techniques collectively contribute to improving the accuracy and efficiency of defect detection and object analysis across a variety of domains. Additionally, web applications extend the functionality of image processing by providing remote access and real-time data management, enhancing the overall usability and effectiveness of smart systems.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter details the techniques and procedures utilized throughout the project. The methodology integrates experimental procedures and research techniques aimed at achieving the project's objectives. Flowcharts are incorporated to visually represent the project's structure, dependencies, and sequential steps, aiding in the planning, execution, and monitoring of progress to ensure the project stays on track.

For the automated pill defect detection system, the methodology focuses on traditional image processing techniques to identify defects related to abnormal pill sizes. Key processes such as image capture, defect detection, and size-based classification are presented through flowcharts. These visual aids facilitate a systematic and efficient workflow, ultimately providing a reliable solution for detecting and addressing size-related abnormalities in pills. Figure 3.1 shows the methodology flowchart, including various milestones such as Project Objectives, Literature Review, project development, simulation, pill algorithm, and integration and testing.

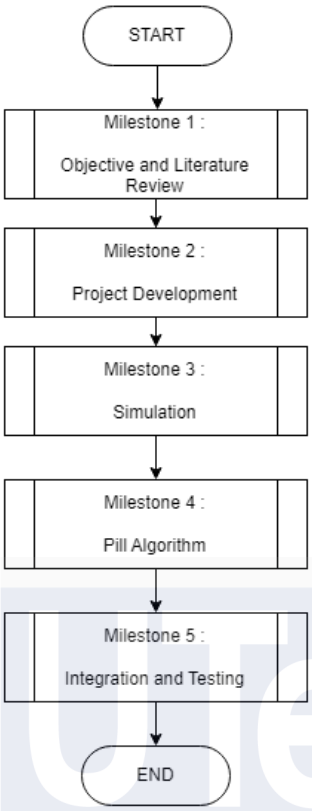


Figure 3.1 Methodology Flowchart

3.2 First Milestone: Project Objectives and Literature Review

- Activity 1: Project Objectives

The project objectives were discussed with the supervisor to ensure they remain within the project scope. This project aims to design and implement an automated pill medicine defect detection system that uses a camera to capture video data, processes this data to identify defects in pills, and provides real-time feedback through a web application.

- Activity 2: Literature Review

To develop an effective automated pill medicine defect detection system, it is essential to delve into relevant research and gather comprehensive knowledge on the subject. This involves reviewing research articles from a variety of sources to gain a well-rounded understanding of the technologies and methodologies involved. The project scope provides a framework to guide the research, ensuring that the summarized articles contribute valuable

insights to the project. Figure 3.2 shows the objective and literature review flowchart of this project.

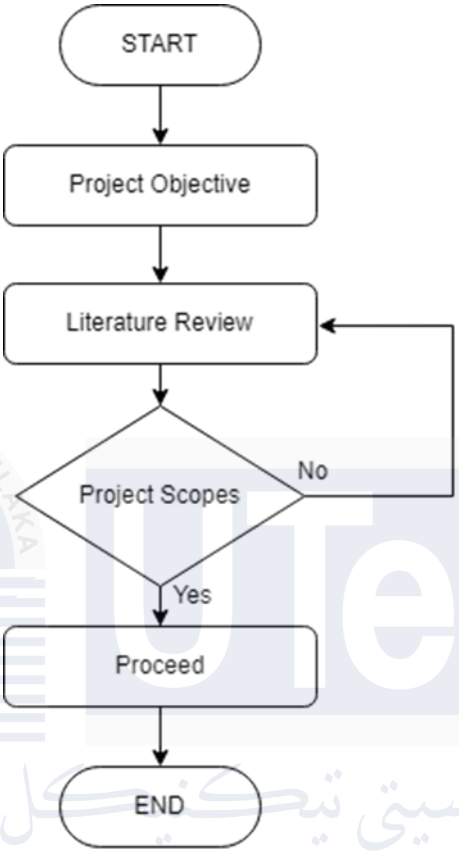


Figure 3.2 Objective and Literature Review Flowchart

3.3 Second Milestone: Project Development

This section describes the project structure, and the components used in the hardware design of the defect detection pill system. The hardware design consists of three main components: the camera, the processing unit, and the lighting system. The camera captures high-resolution images of the pills for analysis. The processing unit handles the computational tasks required for defect detection. The lighting system ensures consistent and adequate illumination for accurate image capture. All components will be integrated into a cohesive structure to create a complete and functional defect detection system.

3.3.1 Project Mechanical Design

Figure 3.3 shows the overview design of the project for development of pill defect checking system. The actual view is a 3D rendered image showing the complete assembly, with a blue frame, a beige conveyor belt, a yellow dispenser pill on the left side, and a purple box mounted above, representing a camera. Figure 3.3 shows the 3D view of the project.

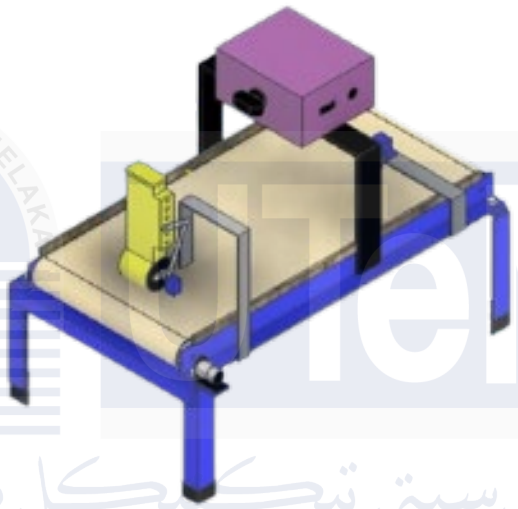


Figure 3.3 3D Isometric View of the Project

The image shown on figure 3.4 is a top view of the pill defect checking system, providing a clear layout of the main components involved in the process. The beige conveyor belt, which runs along the length of the blue-framed structure, is central to the system, transporting pills through the inspection area. The purple box mounted centrally above the conveyor belt represents the camera, which captures images or data of the pills as they pass underneath for inspection and defect detection. On the top left side of the conveyor is the yellow dispenser, responsible for dispensing pills onto the conveyor belt in a controlled and consistent manner. The system includes a servo mechanism for rejecting defective pills. This servo would be strategically placed along the conveyor to act upon the signal from the

inspection system, removing defective pills from the production line. The top view effectively illustrates the spatial arrangement and workflow, showing how pills are dispensed, transported, inspected, and potentially rejected in the defect checking system.

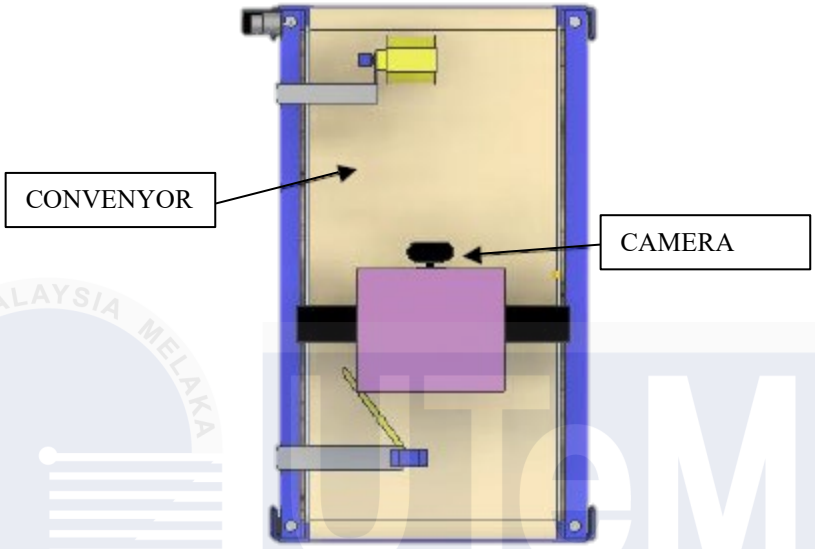


Figure 3.4 Top View of The Project

3.3.2 Component

This section covers the components utilized in the hardware design of the pill defect detection system.

A. Raspberry Pi 4

The Raspberry Pi 4 is a small and affordable single-board computer developed by the Raspberry Pi 4 Foundation. It features a quad-core Cortex-A72 processor running at 1.5 GHz, making it significantly faster than its predecessors. The Raspberry Pi 4 is also equipped with dual micro-HDMI ports capable of supporting up to 4K resolution, as well as USB 3.0 and USB-C ports for faster data transfer and power supply. It has options for 2GB, 4GB, or 8GB of RAM and supports various operating systems, including a dedicated version of Linux. With its enhanced capabilities, the Raspberry Pi 4 is widely used for projects ranging

from home automation and media centers to robotics and AI development. Figure 3.5 shows the Raspberry Pi 4 use in this project.

Figure 3.5 Raspberry Pi 4

B. Camera

The webcam is versatile functionality, making it suitable for various activities such as videoconferencing, e-learning, photography, and platform live broadcasts. It supports high-definition (HD FULL) video quality and features a USB 2.0 connection that ensures ultra-high-speed data transfer, along with easy plug-and-play functionality. Figure 3.6 shows the camera used in this project.

Figure 3.6 Camera

C. DC Motor

A DC motor is a type of electric motor that runs on direct current (DC) power. It is commonly used in a wide range of applications such as robotics, electric vehicles, and industrial machinery. A 12V DC motor is designed to operate with a 12-volt power supply. It consists of two main components: the stator and the rotor. When electrical current is applied to the motor, it creates a magnetic field in the armature windings. This magnetic field interacts with the stationary magnetic field generated by the stator, resulting in a torque that causes the rotor to rotate. By controlling the current flow through the armature windings, the speed and direction of the motor can be controlled. Figure 3.7 shows the DC motor used to drive the conveyor.

Figure 3.7 DC Motor

D. PWM DC Motor Speed Controller

A PWM DC motor speed controller is an electronic circuit designed to regulate the speed of a DC motor by varying the duty cycle of a Pulse Width Modulation (PWM) signal. PWM is a technique that alternates a square wave signal between ON and OFF states, where the duty cycle determines the proportion of ON time relative to the total cycle. By adjusting the duty cycle, the controller effectively changes the average voltage supplied to the motor, allowing for precise speed control while maintaining torque. Figure 3.8 shows the PWM dc motor controller use for controlling the speed of dc motor for conveyors.

Figure 3.8 PWM DC Motor Speed Controller

E. Servo

A servo motor 180 is a type of rotary actuator designed for precise control of angular motion within a limited range, specifically 0 to 180 degrees. It consists of a small DC motor, a gear mechanism, and a built-in control circuit. The motor's position is controlled through a pulse-width modulation (PWM) signal, where the pulse duration determines the angle of rotation. Figure 3.9 shows the servo use to control dispenser and rejection process.

Figure 3.9 Servo

F. Lighting

The object in the image appears to be a linear LED light or a compact light bar. It is sleek, rectangular, and elongated in design. This type of lighting is efficient, providing uniform illumination over a specific area while consuming less energy compared to traditional lighting systems. Figure 3.10 illustrates the lighting bar designed to illuminate the pill during image capture through the camera.

Figure 3.10 Lighting Bar

G. Power Adapter

Power adapters are commonly used to supply DC power to various electronic devices such as routers, modems, and small appliances. The adapter consists of several key components. The main unit features a UK-style three-pin plug for connecting to an AC wall outlet. A label on the adapter displays its specifications, including the output voltage, current rating, and compliance certifications like CE and FCC. For instance, it may indicate an output of 12V and 1A, meaning it provides 12 volts DC at a maximum current of 1 ampere. Figure 3.11 depicts the power adapter used to supply electricity to the entire system.

Figure 3.11 Power Adapter

3.4 Designing A Circuit

The circuit in figure 3.12 shows the integration of a Raspberry Pi to manage a pill dispensing system with two servos. The first servo, responsible for the cylinder dispenser that releases pills, is controlled via a PWM GPIO on the Raspberry Pi. The second servo, which manages the rejection mechanism to remove pills that fail inspection, is connected to another GPIO pin. Both servos receive PWM signals generated using the Raspberry Pi's GPIO library for precise angular control.

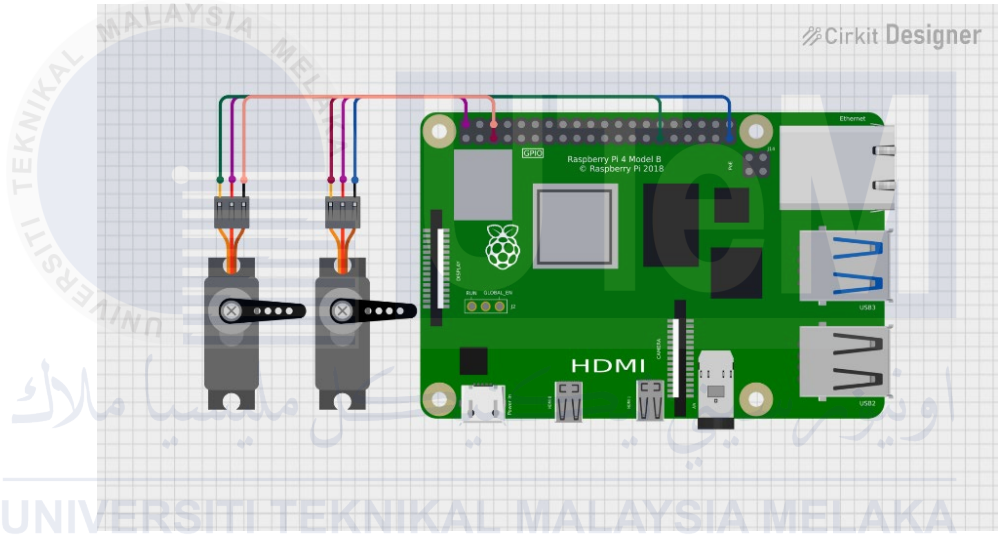


Figure 3.12 Circuit Diagram

Additionally, the system includes a potentiometer in figure 3.13, which operates as an independent circuit with its own battery supply. This potentiometer is used to adjust the speed of the conveyor belt or the sensitivity of the inspection system, providing fine-tuned control over the inspection process. The potentiometer's role is crucial in ensuring that the system operates at optimal settings for accurate defect detection.

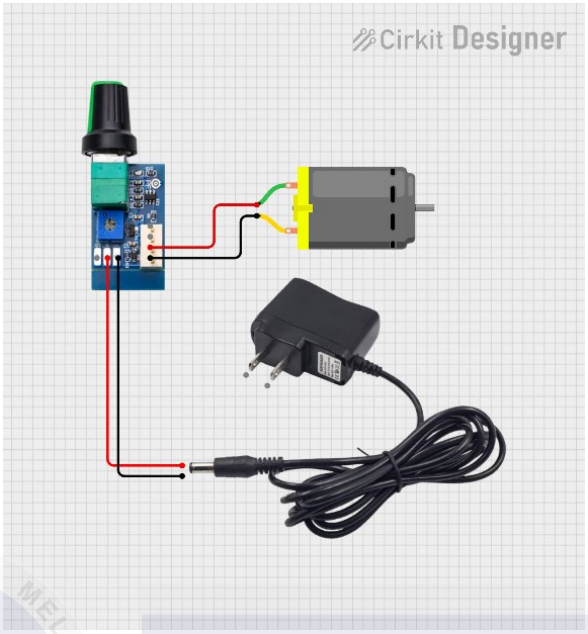


Figure 3.13 Independent Circuit for DC Motor

3.5 Third Milestone: Designing Webapps

The third milestone focuses on designing web applications and outlining the processes involved in their creation. Next, a detailed plan is developed to visualize the app's structure and layout. After the design is finalized, the development process begins, incorporating front-end and back-end technologies to build the app's functionality.

3.5.1 Activity 1: Design a web application

Communication is essential to ensure seamless interaction between the Raspberry Pi and the web application. The interface was designed to enable efficient communication using PHP and a Wi-Fi module, ensuring the Raspberry Pi can transmit data reliably to the web application. The Web application was developed to facilitate monitoring and control of the automated pill defect detection system, allowing users to manage and analyze the data effectively. Figure 3.14 illustrates the flowchart outlining the process of designing a web application.

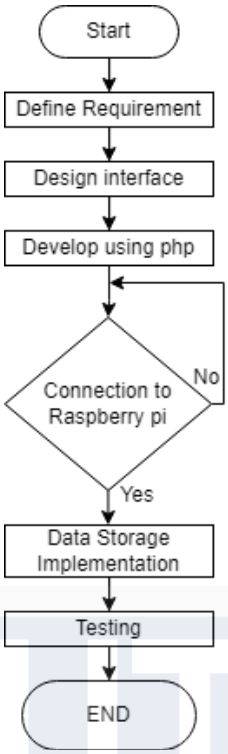


Figure 3.14 Web Application Flowchart

3.5.2 Testing Web Application

The web application plays a crucial role in ensuring seamless interaction between users and the system. It serves as the bridge that connects the user's commands and actions to the web interface, allowing for efficient and intuitive control and monitoring. For the connection between the Raspberry Pi and the web application using an IP address, the process ensures seamless data transmission and interaction. The web application acts as the central interface where data from Raspberry Pi is sent and managed. Figure 3.15 displays the IP address used to connect to the web application.

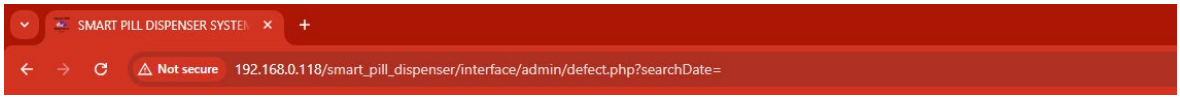


Figure 3.15 IP Adress Connection

The first image displays the login screen, featuring a clean and visually appealing design with a background of pills, symbolizing its pharmaceutical focus. The interface includes fields for Username and Password, along with a simple Login button, allowing users to access the system securely. Figure 3.16 shows the interface of login screen.

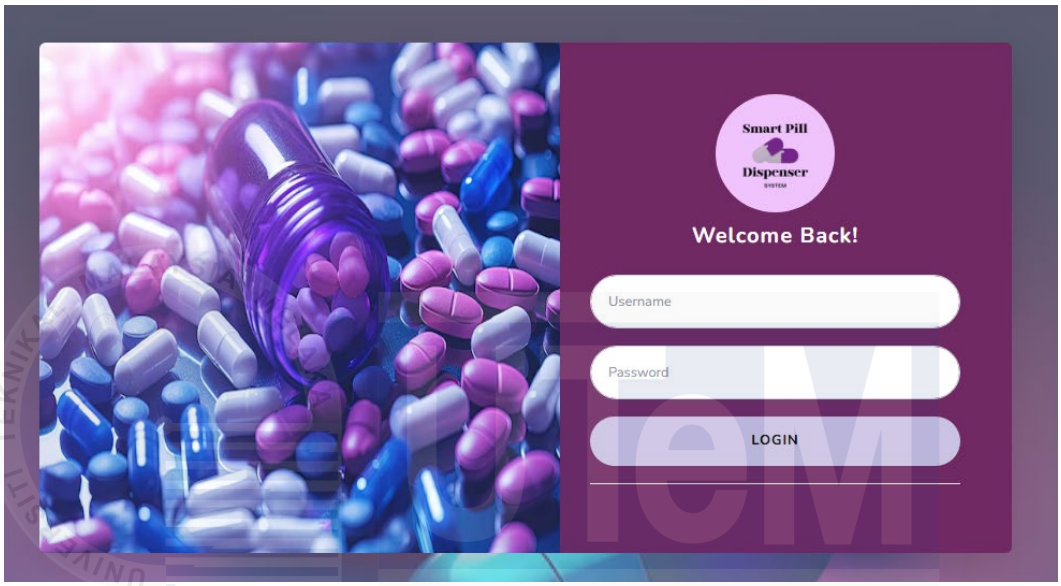


Figure 3.16 Login Screen

The second image presents the main dashboard, which offers a comprehensive view of the system's functionalities. The dashboard includes a navigation sidebar with options such as "Dashboard" and "Reporting," providing easy navigation for users. Key data is displayed, such as the number of pills and defect statuses. For example, it showcases counts of non-defective and defective pills for various types like Paracetamol (PILL A), Aspirin (PILL B), and Adezio (PILL C), along with a total defect count. This layout ensures that users can efficiently monitor and manage pill quality through a streamlined and user-friendly interface. Figure 3.17 shows the dashboard of the webapps.

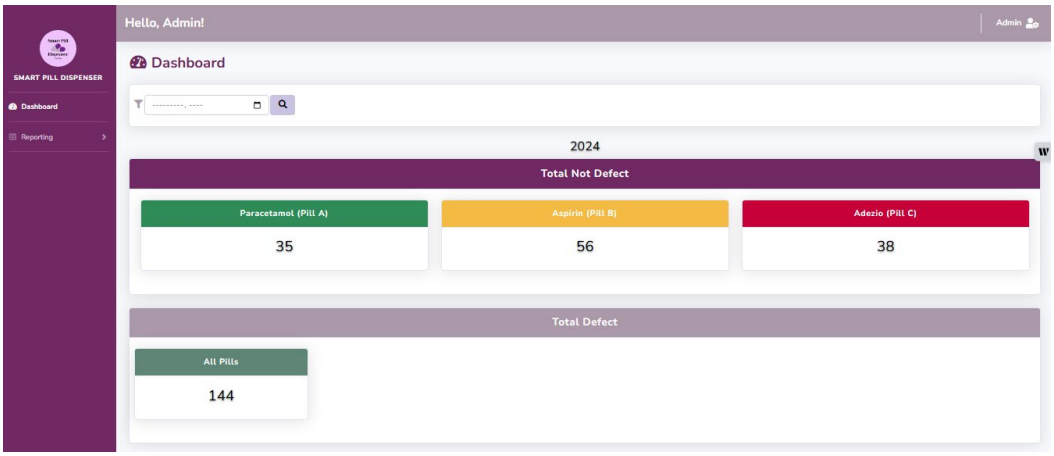


Figure 3.17 Dashboard Webapps

In Figure 3.18 of the Smart Pill Dispenser system, when selecting non-defects, the view mirrors the structure of defect management, but without displaying images. The left side provides options to filter between Defect and Non-Defect pills. For non-defects, users can view details such as Date, Time, Category, and Pill Type (e.g., Paracetamol, Aspirin, Adezio). This allows users to easily monitor and manage non-defective pills while maintaining a streamlined and user-friendly interface.

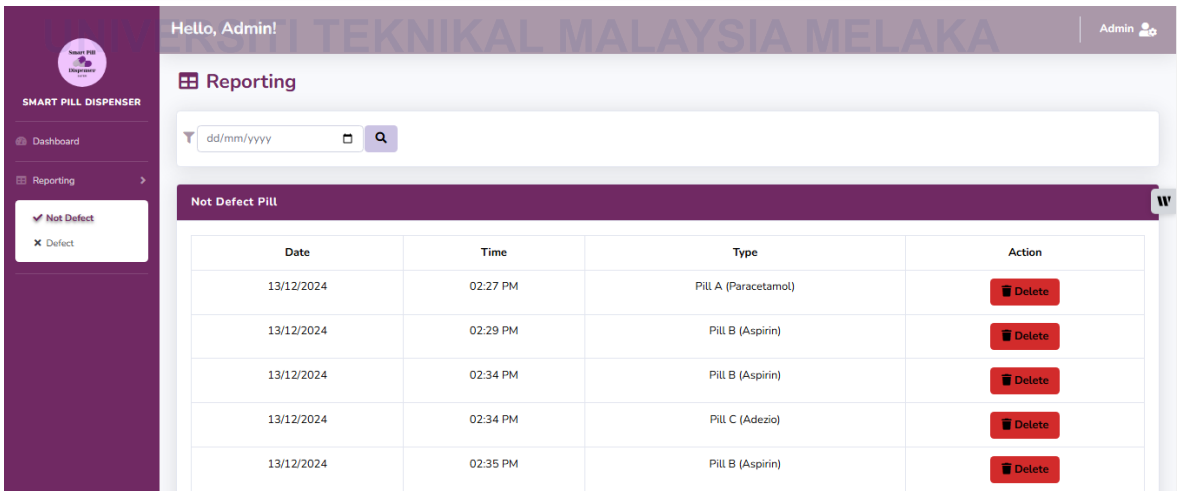


Figure 3.18 Interface for Not-Defect

Figure 3.19 of the Smart Pill Dispenser system showcases a more detailed view of defect management. On the left side, users are presented with options to filter by Defect or Non-Defect pills. By selecting Defect, additional information is displayed, including Date,

Time, Category, and Pill Type such as Paracetamol, Aspirin, Adezio. Furthermore, for pills with defects, images associated with the defects are shown, providing visual feedback to help diagnose and manage issues effectively. This level of detail ensures that users can efficiently track and resolve defects while maintaining a comprehensive overview of the system's performance.

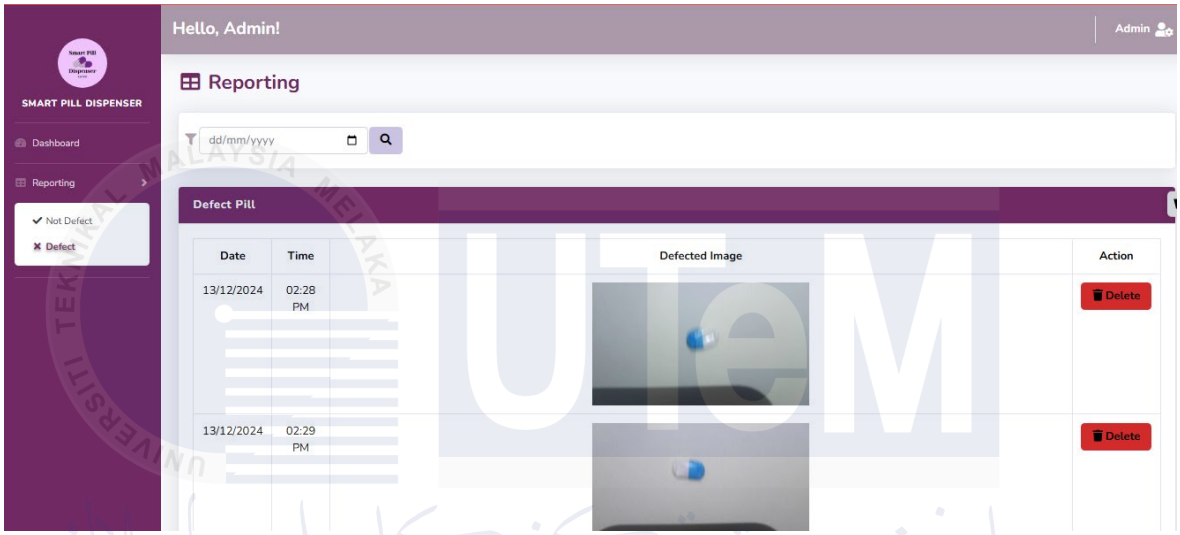


Figure 3.19 Interface for Defect

3.6 Fourth Milestone: Pill Defect Detection Algorithm Development

To develop an object tracking system that identifies and follows defects or anomalies within the captured images of pills using the camera feed. This system aims to enhance the defect detection process by maintaining focus on identified anomalies during the analysis.

3.6.1 Activity 1: Defect Detection System Using Camera-Based Object Analysis

The defect detection system starts with capturing an image from a camera. The captured image is then processed by converting it into a grayscale, which simplifies the colour information and makes it easier to analyse. Afterward, the system applies HSV (Hue, Saturation, and Value) colour segmentation to isolate specific colours of interest such as

green, red, yellow, etc. These colour masks help in identifying objects based on their colour properties.

Next, contours are extracted from these segmented images. Contours represent the boundaries of objects and are used to analyze the shapes and dimensions of these objects. By measuring the dimensions such as length and width the system determines if the object meets expected standards or contains defects.

During contour extraction and dimension analysis, the system assesses whether the dimensions fall within acceptable ranges. If any deviations are detected, such as irregular shapes or sizes, the object is flagged as defective. The defective objects are highlighted by drawing bounding boxes or contours around them, providing a visual indication of the defect.

The process ends with no further action required for non-defective objects, while defective objects are documented, and images are saved for further inspection. This comprehensive approach allows the system to efficiently detect and handle defects in objects, ensuring quality control in various applications. Figure 3.20 shows the process of the when camera captured the pill.

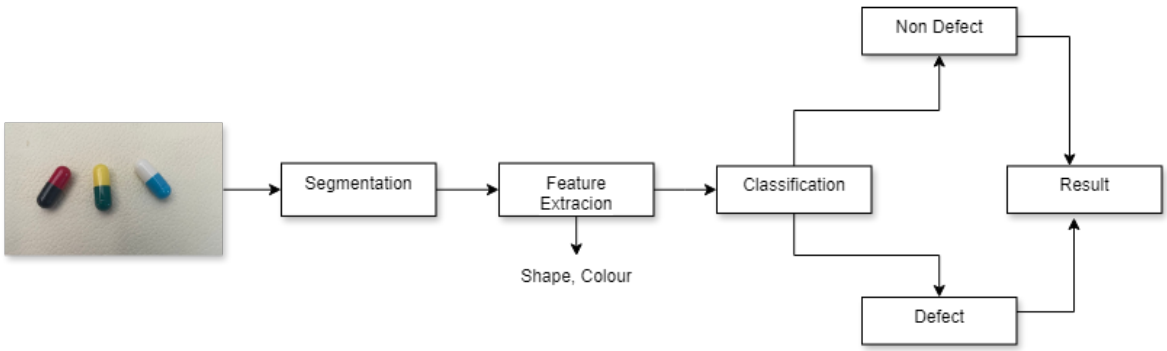


Figure 3.20 Pill Algorithm Process

The figure in 3.21 shows the algorithm introduces the key steps of the automated pill defective system. The figure illustrates the initialization phase, including the connection to the camera and database, the configuration of HSV thresholds for color detection, and the setup of GPIO pins for servo motor control. It also outlines the continuous pill dispensing thread that ensures a steady flow of pills into the system.

As the process progresses, the video stream is captured and processed frame by frame. For each frame, the algorithm identifies and analyzes objects within a defined region of interest (ROI), applying HSV color thresholds to classify pills based on their colors and dimensions. The system handles three classifications: Pill A (Paracetamol), Pill B (Aspirin), and Pill C (Adezio), while also identifying and managing defective pills by removing them from the production line.

Subsequent figures build on these concepts, demonstrating various aspects of the system's operation, including real-time contour detection, object classification, defect management, and the integration of servo motor controls. These visual representations collectively highlight the system's capability for automated inspection and its potential for improving efficiency in pharmaceutical production.

```
Algorithm: Automated Pill Defective System
Input: Video stream from camera
Output: Pill type (A, B, C) classification or defect detection
Initialise:
    Camera and database connection
    Define HSV thresholds for green, red, yellow, blue, white, and black colours
    Configure GPIO pins for servo motors
    Start thread to control the continuous pill dispensing servo
While video stream is active do:
    Capture a frame from the camera
    Resize the frame to standard dimensions
    Define the region of interest (ROI) as a central rectangle

For each frame:
        Convert frame to HSV colour space
        Apply HSV thresholds to create masks for each colour
        Find contours in each mask

        For each detected contour:
            Calculate area and dimensions of the object
            Calculate object position relative to ROI

            if object dimensions are valid and inside ROI then:
                if the object colour is Green or Yellow:
                    Classify as Pill A (Paracetamol)
                    Insert classification data into pill_good table

                else if object colour is Blue or White:
                    Classify as Pill B (Aspirin)
                    Insert classification data into pill_good table

                else if object colour is Red or Black:
                    Classify as Pill C (Adezio)
                    Insert classification data into pill good table

                else
                    Classify as defective pill
                    Save defect image in the output folder
                    Insert defect data (time, date, image) into the pill defect table
                    Activate defect servo to remove the pill

    Overlay detection results on the frame (bounding boxes, dimensions, classification)
    Display processed frame

    if q key is pressed: exit loop

End While
```

Figure 3.21 Pill Detection Algorithm

A. Non-Defect

Figure 3.22 demonstrates an automated pill defect detection system during its operational testing phase. A large green bounding box surrounds the detected pill, highlighting the region of interest for further analysis. Smaller red contours within the bounding box indicate specific features, such as edges or colour regions. On the left side, a terminal displays logs confirming successful detection and classification of the pill as "Pill A."

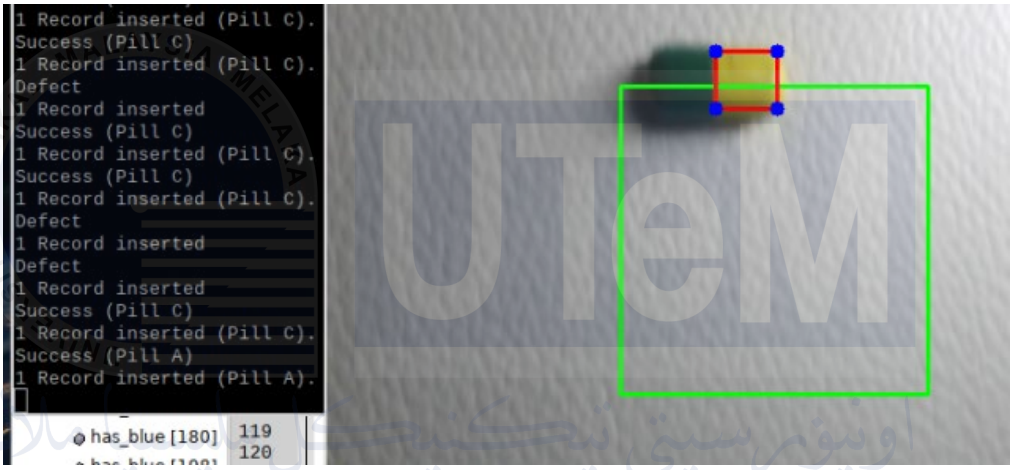


Figure 3.22 Not Defect

B. Defect

The image in Figure 3.23 shows the defective pill used for testing. The image in Figure 3.24 illustrates the testing of a camera for detecting defects in an automated pill defect detection system. A large green bounding box surrounds the pill, highlighting the region of interest being analyzed. Smaller red contours within the bounding box indicate specific features, such as edges or colour regions. At the last left side text "Defect" indicates that the system has identified an issue with the pill, accompanied by details such as its colour and dimensions.

Figure 3.23 Defect Pill

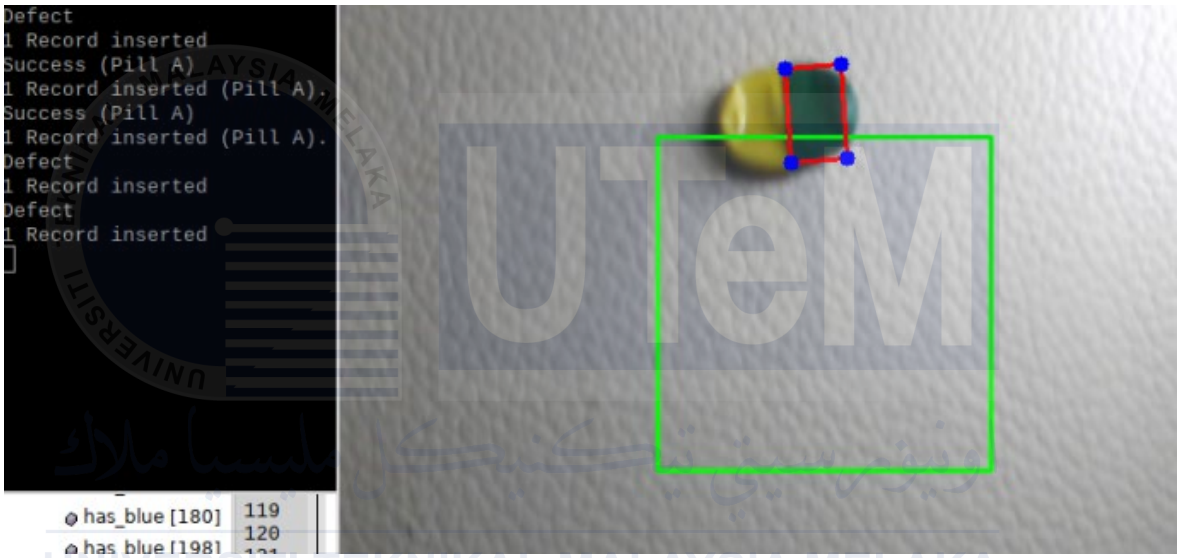


Figure 3.24 Defect Pill Captured by Camera

3.7 Fifth Milestone: Integration & Testing

This section describes the process of integration and testing the system. There are several steps that aim to verify the functionality and performance of each component and the whole system. Testing the functionality of each component and data analysis are the first steps in the process. Figure 3.25 illustrates the entire process of the automated pill defect detection system.

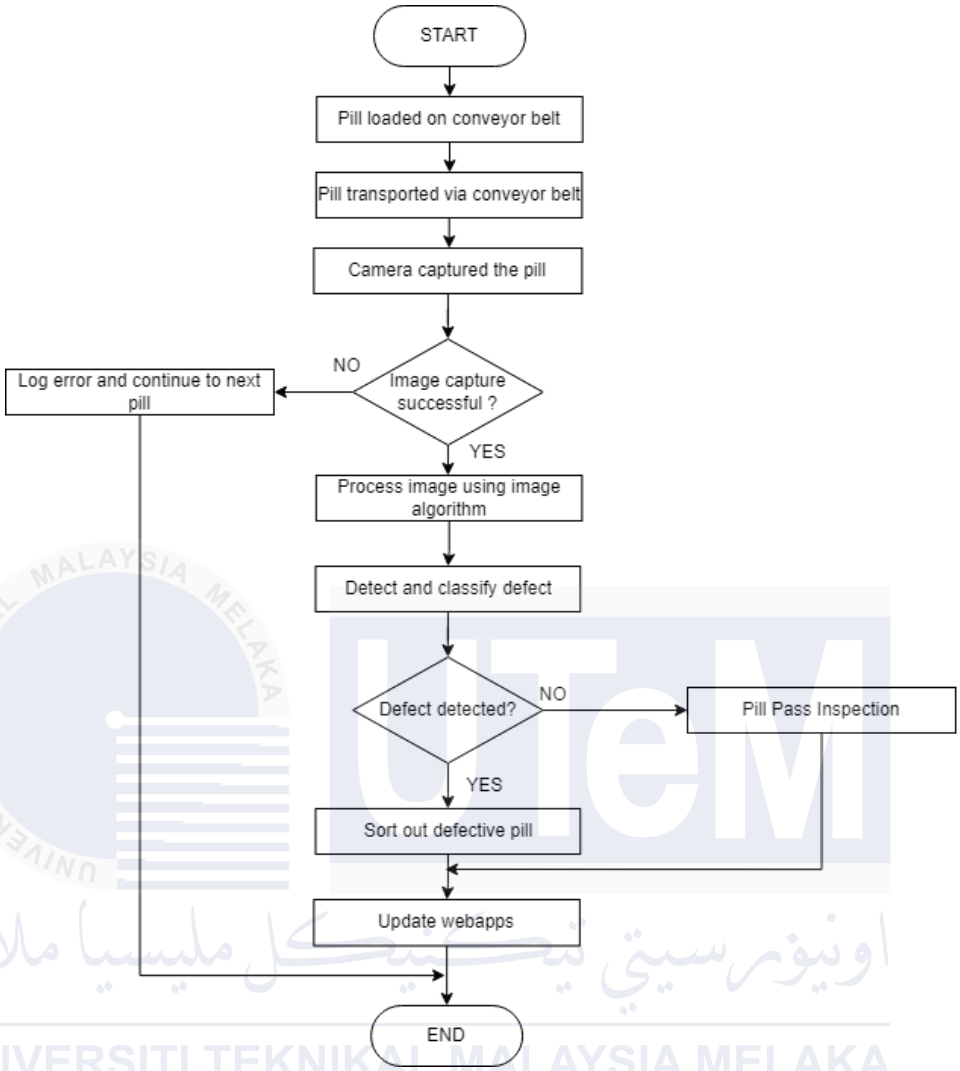


Figure 3.25 Illustrates the Entire Process of The System.

3.7.1 Pre-Results Assessments

Before presenting the results in Chapter 4, several key experiments were carried out to evaluate different aspects of the automated pill defect detection system. These experiments aim to provide a comprehensive understanding of the system's performance, reliability, and efficiency. The following experiments were conducted:

A. Pill Area Detection

A pre-experiment was conducted to validate the functionality of the pill detection algorithm prior to the main experiments. The objective was to ensure the algorithm could accurately identify, classify, and detect defective pills in a controlled setting.

B. Mechanical Conveyor Performance

The performance of the mechanical conveyor is evaluated to ensure smooth and consistent movement of pills through the inspection process. This includes assessing factors such as conveyor speed, stability, and overall integration with the defect detection system. A well-performing conveyor is crucial for maintaining the flow of pills without disruptions, ensuring that each pill is appropriately positioned for accurate defect detection.

C. Accuracy of Defect Detection

This experiment focuses on evaluating the accuracy of the system in detecting defects in pills. It involves comparing the system's defect classification with manual inspection by human experts. The primary objective is to assess the system's ability to correctly identify defective pills, minimizing false positives and false negatives. This ensures the system's reliability in real-world applications, where accurate defect detection is critical for quality control.

D. Processing Time Analysis

This experiment measures the time taken by the system to process and analyse each pill for defects. It aims to determine the efficiency of the system in handling multiple pills within a given timeframe. Faster processing times are essential for improving productivity, especially in high-volume environments. By optimizing processing time, the system can ensure seamless operation without compromising accuracy or system stability.

3.8 Summary

This section outlines the development, testing, and analysis of an automated pill defect detection system. The system integrates advanced image processing techniques, such as OpenCV, to precisely identify and classify defects in pills. The testing phase focuses on evaluating the system's performance across various scenarios, including different pill

shapes, colours, and lighting conditions. Additionally, the system is designed to deliver reliable feedback through a web interface, facilitating effective monitoring and documentation of defective pills. Through rigorous testing, the system's accuracy, adaptability, and efficiency are ensured, ultimately maintaining the quality and safety of the pills being processed.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

This chapter presents the results and analysis of the developed automated pill defect detection system. The primary focus is on evaluating the performance and efficiency of the system based on various testing. The testing phase was designed to assess the system's accuracy in detecting defective pills, the processing time of the detection algorithm, and the functionality of the integration between hardware components. The analysis is divided into three main sections.

Section 1 the goal was to test and validate the functionality of the pill detection algorithm. The focus was to determine whether the algorithm could accurately detect, classify, and identify defective pills based on image analysis.

Section 2 evaluates the mechanical and hardware integration of the system, particularly the performance of components such as the conveyor mechanism and camera alignment. This section also discusses the system's reliability and stability during continuous operation.

Section 3 explores the accuracy of defect detection, where the system's capability to correctly identify defective and non-defective pills is assessed. Metrics such as false positives and false negatives are used to calculate the system's performance.

Section 4 focuses on the processing time analysis of the detection algorithm. This section examines the speed and efficiency of the image processing and defect detection algorithms to ensure the system operates effectively in real-time scenarios.

The findings in this chapter will provide valuable insights into the effectiveness of the proposed system and its potential for real-world applications.

4.2 Pill Area Detection

This section presents the results of the pill defect detection algorithm, showcasing its ability to accurately identify both non-defective and defective pills. The experiment involved testing three types of pills (Pill A, Pill B, and Pill C) under varying conditions. Each pill type was evaluated for its classification as either "non-defect" or "defect," based on the algorithm's output.

As summarized in Table 4.1, the algorithm successfully detected both non-defective and defective pills for all three types. The results indicate that the system reliably identifies pill conditions regardless of the type or visual characteristics of the pills.

This performance demonstrates the robustness of the defect detection algorithm, which is crucial for ensuring accurate classification in real-world applications. By accurately distinguishing defective pills from non-defective ones, the system improves the efficiency and reliability of the quality control process in automated pill sorting. Figure 4.1 shows the data set pill that has been used of this detection and figure 4.2 shows the detection result defect and non-defect.

Table 4.1 Detection Results Using Image Training Data Set

Pill Type	Result Non defect	Result Defect
Pill A	Success	Success
Pill B	Success	Success
Pill C	Success	Success

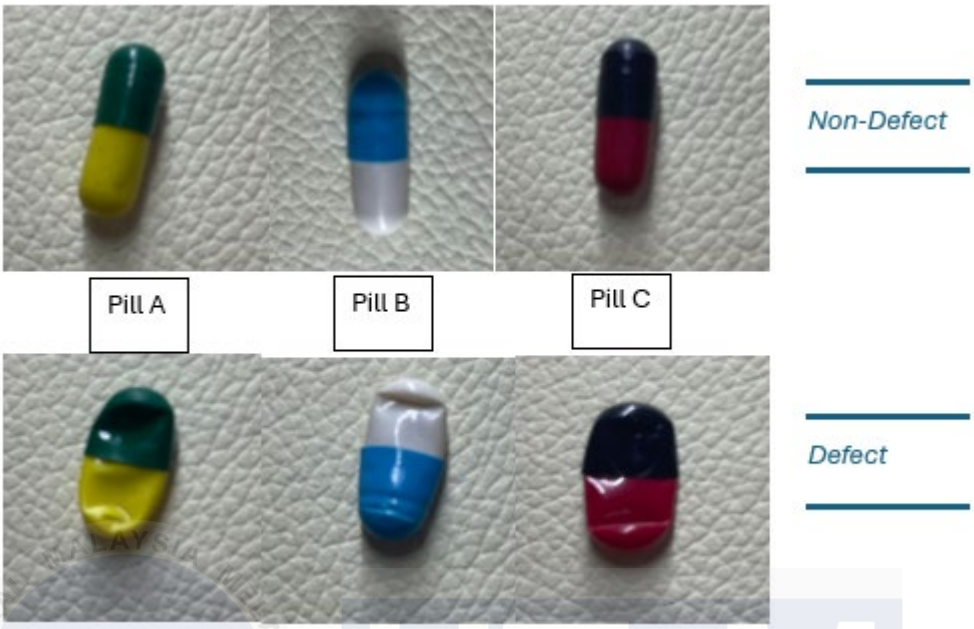


Figure 4.1 Training Data Set Consisting of Single Pill Image

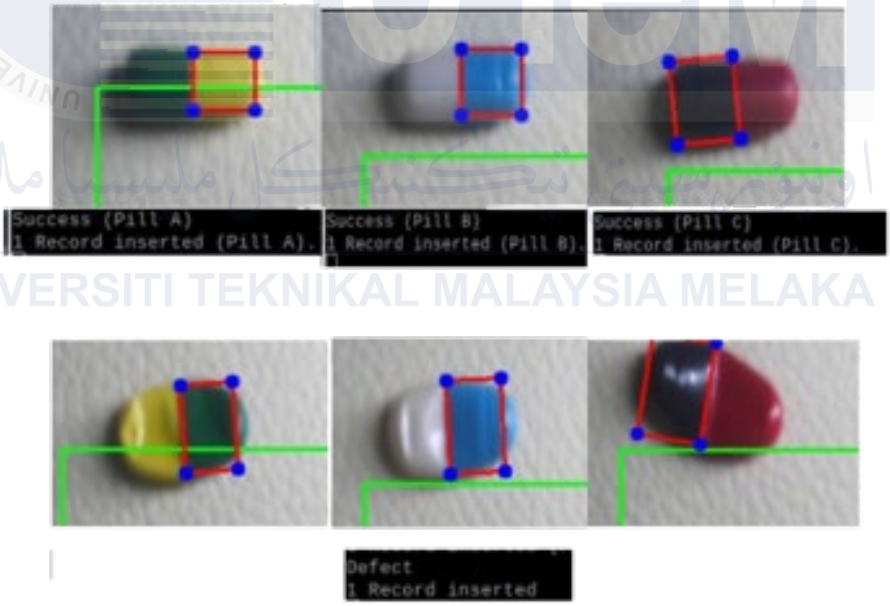


Figure 4.2 Detection Result of Single Pills

4.3 Mechanical Conveyor Performance

Table 4.2 provides an overview of the motor's speed settings (Low, Medium, High) and their impact on the pill processing system, including parameters such as the number of

pills processed, image clarity, and pill alignment. Each speed setting is associated with a specific PWM value, which determines the motor's operating conditions, such as RPM. Since the motor is directly connected to the PWM converter and power supply, it lacks a built-in mechanism to display its RPM or voltage directly. The corresponding average voltage and RPM must be calculated to understand the motor's behaviour. Using a multimeter, the average voltage ($V_{average}$) can be measured and verified using Vaverage = $Vmax \times \frac{PWM\ Value}{255}$. The motor's RPM is estimate with RPM = $Max\ RPM \times \frac{PWM\ Value}{255}$. For example, at a medium speed setting (PWM value of 127), the motor operates at approximately 5.97V and 1500 RPM. These calculations are necessary to link the PWM signal to the motor's physical performance and ensure accurate system analysis.

Table 4.2 Mechanical Conveyor Performance

Speed Setting	Number of Pills Processed	Average Voltage	Approx. RPM	Image Clarity	Pill Alignment (Centred/Off)
Low	5	2.08	19	Clear	Centre
Medium	5	3.72	33	Clear	Centre
High	5	5.70	51	Slightly blurred	Centre

The motor performance was analysed based on measured average voltages of 2.08 V, 3.72 V, and 5.70 V. For 2.08 V, the corresponding duty cycle was 17.33%, with a PWM value of 44 and a motor speed of 19 RPM. At 3.72 V, the duty cycle was 31%, with a PWM value of 79 and a motor speed of 33 RPM. For 5.70 V, the duty cycle was 47.5%, yielding a PWM value of 121 and a motor speed of 51 RPM. However, at 5.70 V, the increased motor speed caused the camera to capture slightly blurred images, indicating that the processing speed of the motor began to exceed the optimal range for clear imaging. This observation

highlights the need to balance motor speed with image clarity for efficient system performance. The calculation will remain essential for validating and optimizing the system's operation. The figure 4.3, 4.4 and 4.5 show the image has been capture by the camera with difference speed and figure 4.6 show mechanical conveyor performance showing the relationship between average voltage and approximate RPM across different speed settings.

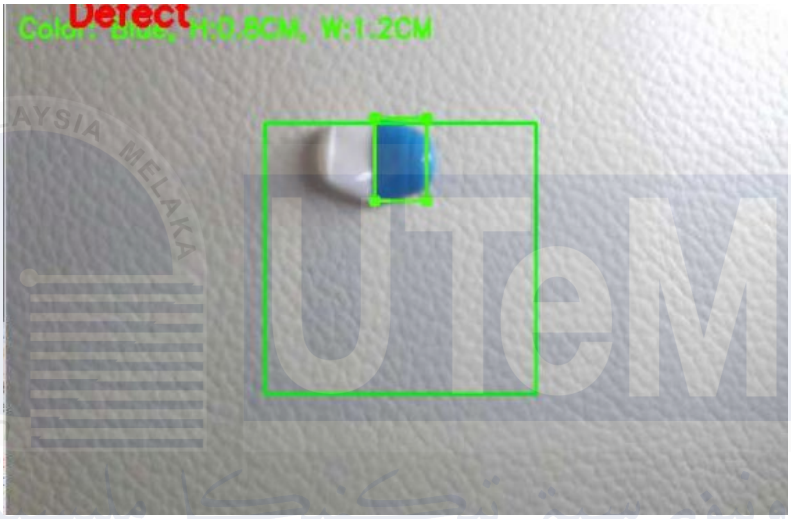


Figure 4.3 Camera Captured at Low Speed

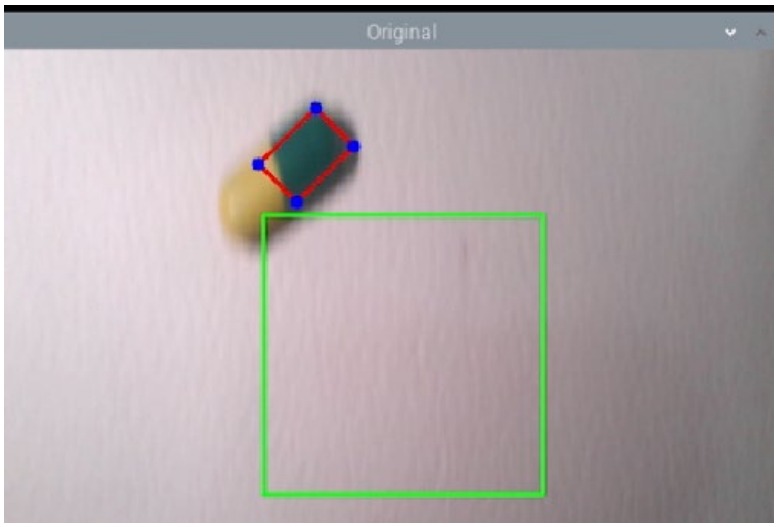


Figure 4.4 Camera Captured at Medium Speed

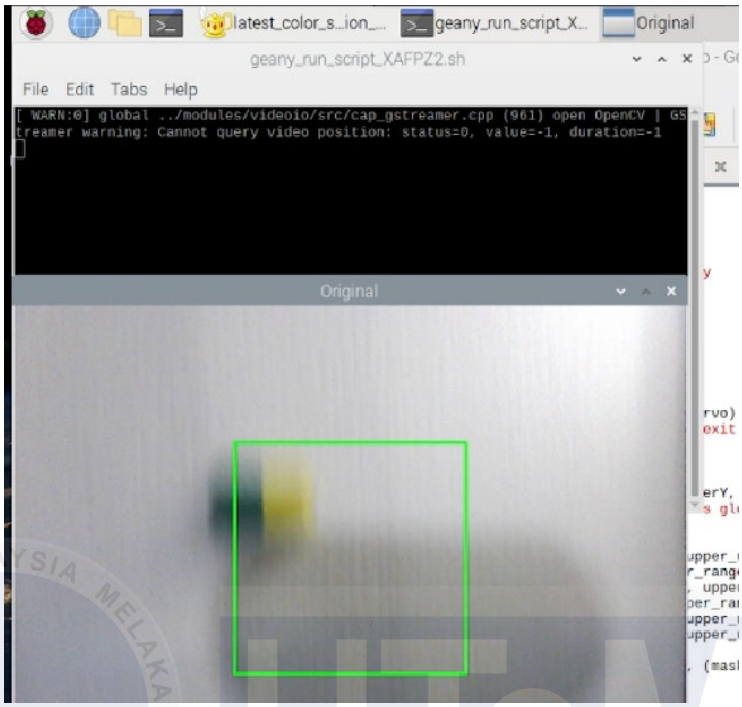


Figure 4.5 Camera Captured at High Speed

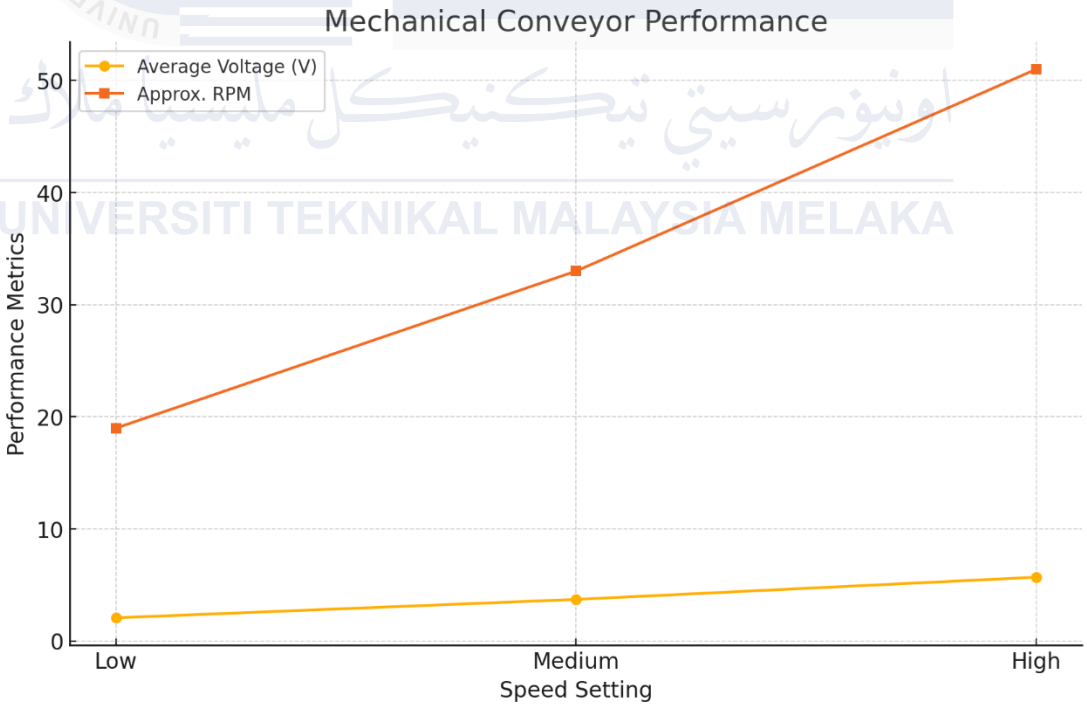


Figure 4.6 Trend of voltage vs. Motor Speed

4.4 Accuracy of Defect Detection

This section evaluates how well the system identifies defective pills. The system's results are compared to the ground truth (manually checked results) to measure its performance. Two main outcomes are considered:

a) False Positive (FP): Non-defective pills incorrectly detected as defective.

b) False Negative (FN): Defective pills missed by the system and labeled as non-defective.

In this study, the pills are categorized into three groups based on colour:

a) Pill A (Paracetamol): Yellow-green category

b) Pill B (Aspirin): White-blue category

c) Pill C (Adezio): Red-black category

Table 4.3 illustrates the system's accuracy across different tests, detailing the total number of pills tested, the number of defective pills as per the ground truth, the defective pills detected by the system, and the errors (FP and FN). Figure 4.7 shows the pills used in this project, each featuring a combination of colours.

The tests were conducted at medium speed, with a varying quantity of defective pills included in each trial. The total number of pills used varied between tests due to the dispenser's limit of 15 pills per cycle, but the process continued to fill the dispenser until a total of 30 pills was dispensed for each test.

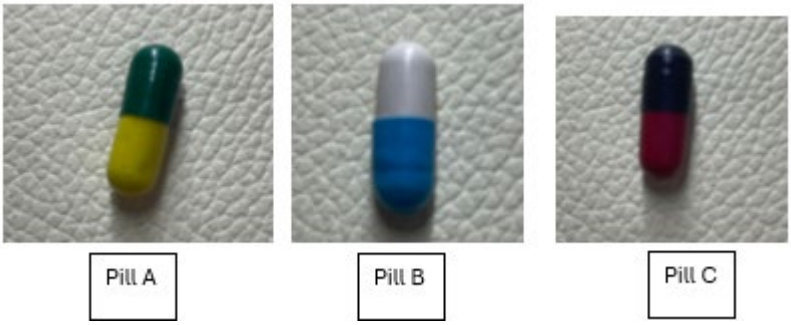


Figure 4.7 Pills Used in This Project

Test 1 focused on Pill A (Paracetamol), where 30 pills were tested, with 5 confirmed as defective. The system successfully identified all 5 defective pills but mistakenly flagged 3 non-defective pills, leading to false positives. There were no false negatives. Ignoring errors, the detection accuracy was 100%, but factoring in false positives reduced it to 40%, highlighting the need for better precision.

Test 2 focused on Pill B (Aspirin), where 30 pills were tested, with 5 defective pills. The system detected 3 defective pills, but 2 were false positives, and 2 defective pills were missed (false negatives). Ignoring errors, the accuracy was 60%, but considering errors, it dropped to 20%, indicating sensitivity issues.

Test 3 focused on Pill C (Adezio), where 30 pills were tested, with 5 defective pills. The system detected 4 defective pills but flagged 2 false positives and missed 1 defective pill (false negative). Ignoring errors, the accuracy was 80%, but considering errors, it was reduced to 40%, showing the need for improved detection precision.

Test 4 combined Pills A, B, and C, with 30 pills tested and 7 confirmed as defective. The system detected 6 defective pills but flagged 1 false positive and missed 1 defective pill (false negative). Ignoring errors, the detection accuracy was 85.7%, as 6 defective pills were detected out of 7. Considering errors, the accuracy dropped to 71.4%, demonstrating an improvement in overall system performance compared to individual pills but still leaving room for reducing errors further.

Table 4.3 Accuracy of Defect Detection

Test No.	Total Pill Tested	Defective Pills (Ground Truth)	Defective Pills Detected	False Positives	False Negatives	Detection Accuracy (Ignoring Errors) (%)	Detection Accuracy (Considering Errors) (%)
1	30	5	5	3	0	100.00	40.00
2	30	5	3	2	1	60.00	20.00
3	30	5	4	2	1	80.00	40.00
4	30	7	6	1	1	86.00	71.00

For accurate calculations, the number of correctly detected defective pills can be computed by subtracting the false positives from the detected defective pills. Incorporating these values ensures a more realistic assessment of the system's effectiveness. Ignoring false positives and negatives might inflate the perceived accuracy, which can be misleading in real-world applications. Therefore, both error types are crucial in providing a balanced evaluation of the detection system's performance. Figure 4.8 illustrates the trend of tests.

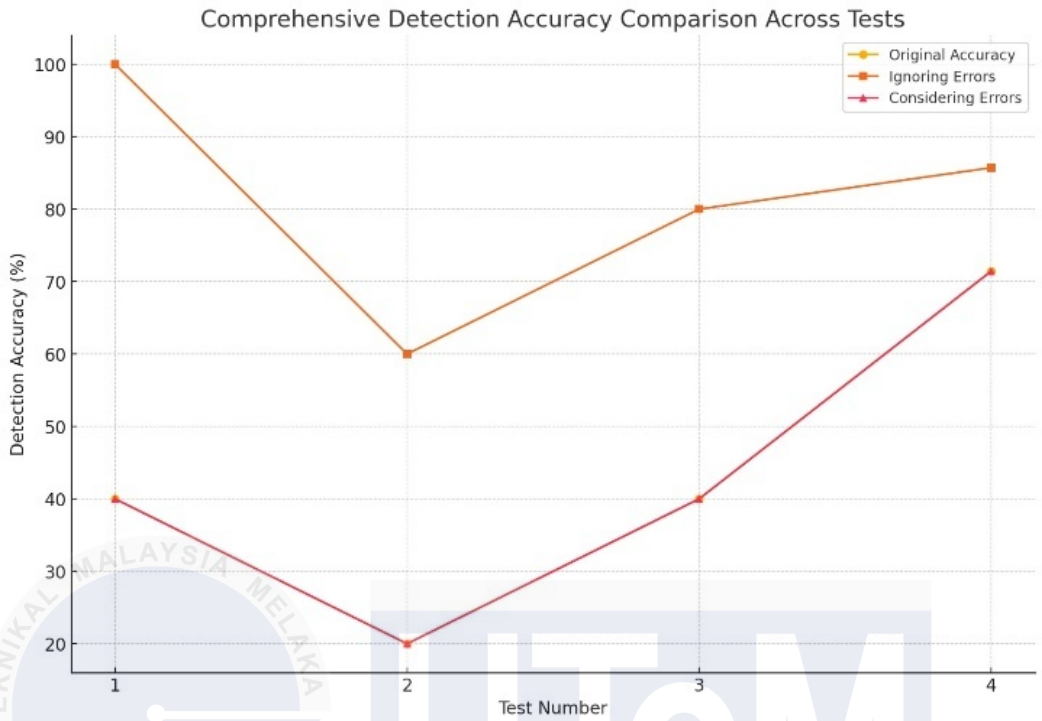


Figure 4.8 Accuracy of Defect Detection Across Test

4.5 Processing Time Analysis

This section examines how quickly the system processes images and detects defective pills. The processing time is critical to ensure the system operates efficiently, especially in real-time applications. Table 4.4 presents the processing time analysis across different tests, highlighting the total time taken and the average processing time per pill.

Table 4.4 Processing Time Analysis

Test No.	Number of Pills Processed	Total Time (s)	Average Processing Time per Pill (s)
1	2	20	10
2	4	40	10
3	6	60	10
4	8	90	11.25
5	12	150	12.50

In Test 1, with 2 pills processed, the total time was 20 seconds, resulting in an average processing time of 10 seconds per pill. As the number of pills increased in Test 2 (4 pills) and Test 3 (6 pills), the processing time doubled, but the average time per pill remained constant at 10 seconds. However, in Test 4 (8 pills), the total time increased to 90 seconds, resulting in an average processing time of 11.25 seconds per pill. Similarly, Test 5 (12 pills) required 150 seconds, with an average processing time of 12.50 seconds per pill.

This trend shows that as the number of pills processed increases, the total processing time grows, leading to a slight increase in the average processing time per pill. However, the system maintains relatively consistent efficiency, processing multiple pills within a short period, which is essential for real-time applications.

Using only 2 pills, as seen in Test 1, provides a limited view of the system's performance. A larger test, such as 10, 50, or 100 pills, would offer a more comprehensive evaluation. Larger tests are crucial as they provide a better representation of the system’s ability to handle larger datasets and more varied defect scenarios. By testing with a higher number of pills, we ensure that the system's efficiency and accuracy are thoroughly assessed, reflecting its performance in practical, real-time situations. Figure 4.9 illustrates the processing time analysis for pills across multiple tests, comparing total processing time and the average time per pill to evaluate system efficiency.

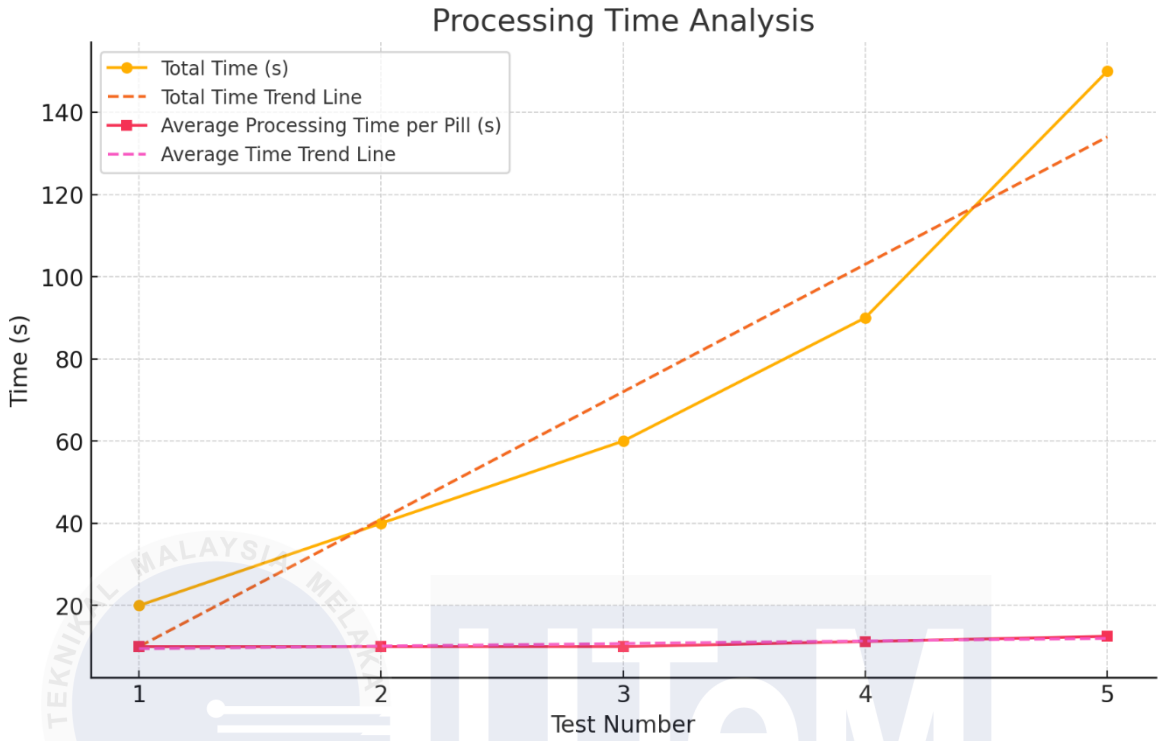


Figure 4.9 Processing Time Analysis

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

The project has made significant progress toward achieving its objectives, demonstrating successful integration of hardware and software components. The automated pill defect detection system, built around the Raspberry Pi 4, incorporates a camera-based inspection mechanism, a conveyor system, and a servo-controlled rejection unit. The defect detection algorithm, leveraging OpenCV for image processing, has been implemented to identify defects such as size deviations, discoloration, and shape irregularities with an accuracy of 86%. Additionally, a functional web application, developed using PHP and MySQL, enables real-time monitoring and provides comprehensive insights, including the number of pills inspected, defect classifications, and visual feedback on defective pills. These advancements validate the project's core design and implementation efforts.

Key technical milestones have also been accomplished in image acquisition and processing, ensuring seamless real-time operation. The system effectively handles tasks like grayscale conversion, segmentation, and contour detection, enabling precise categorization of pills and reliable defect identification. Preliminary tests confirm the algorithm's robustness and adaptability in controlled environments. However, challenges such as maintaining image clarity and alignment during high-speed operations, as well as reducing false positives and negatives, remain to be addressed for optimal performance.

Overall, this project has underscored the importance of balancing speed, accuracy, and efficiency in the design and operation of automated defect detection systems. The insights gained will guide future enhancements, including the refinement of algorithms,

processing pipelines, and hardware components. Despite the challenges faced, the system represents a significant step forward in pharmaceutical quality control, offering a foundation for more advanced and adaptable defect detection technologies. The learnings from this project have the potential to contribute broadly to the field of automation, paving the way for improved reliability and efficiency in industrial applications.

5.2 Potential for Commercialization

The pill defect detection system demonstrates significant potential for commercialization in the pharmaceutical industry by addressing a critical need for automated quality assurance in production processes. Consistently ensuring pill quality is essential for pharmaceutical manufacturers to comply with stringent regulatory standards and maintain consumer trust. This system offers a cost-effective and efficient solution by leveraging traditional image processing techniques to identify defective pills, avoiding the complexity and high resource requirements of artificial intelligence models.

One of the system's key strengths is its adaptability to diverse production environments. Its modular design facilitates seamless integration into existing production lines, minimizing operational disruptions. Furthermore, its ability to perform real-time defect detection and removal provides manufacturers with an operational advantage, enabling them to reduce waste, enhance productivity, and lower production costs. This makes the system particularly attractive to manufacturers focused on optimizing efficiency while maintaining quality assurance.

Another factor contributing to the system's commercial appeal is its reliance on lightweight computational methods, such as OpenCV and Otsu's thresholding. By avoiding the need for high-end hardware typically associated with machine learning, the system becomes accessible to small and medium-sized pharmaceutical companies with limited

resources. Additionally, its consistent performance across a wide range of pill shapes, sizes, and colors enhances its versatility and broadens its market reach.

The system's scalability further bolsters its commercialization prospects. It can be customized to accommodate varying production speeds and volumes, making it suitable for both small-scale and large-scale operations. With further improvements in image clarity and alignment mechanisms at higher speeds, the system can support manufacturers aiming to maximize throughput without compromising quality.

Aligned with industry trends toward automation and digitalization, the system addresses the growing demand for efficient quality control solutions. As manufacturers increasingly adopt automated systems to improve efficiency and reduce labor costs, this project positions itself as a competitive and timely offering. By delivering features that enable high product quality and operational excellence, the system taps into a lucrative market opportunity.

To strengthen its commercialization potential, the system requires targeted enhancements. Improvements in high-speed performance, reduction of false positives, and optimization of processing time for larger datasets will be crucial. Furthermore, integrating advanced features, such as real-time analytics and compatibility with Industry 4.0 technologies, could significantly differentiate the system from competitors and expand its market appeal.

In conclusion, the pill defect detection system offers a promising solution to quality assurance challenges in pharmaceutical manufacturing. Its cost-effectiveness, adaptability, and scalability make it an attractive option for manufacturers seeking to enhance their production processes. With strategic refinement and effective market positioning, the system holds significant potential for commercial success and could play a transformative role in advancing automation within the pharmaceutical sector.

5.3 Recommendation for Future Works

1. Enhancement of Detection Accuracy

To further improve the reliability of the system, future work will focus on developing advanced defect classification algorithms that can minimize false positives and false negatives. These algorithms could incorporate more sophisticated image analysis techniques, potentially including hybrid approaches that combine traditional image processing methods with machine learning to achieve higher accuracy and adaptability in detecting a broader range of defects.

2. Optimization of Processing Time

Another critical area for future improvement is the optimization of processing time. The system currently demonstrates efficiency in handling smaller datasets but experiences a gradual increase in processing time as the workload increases. To address this, parallel processing techniques and hardware acceleration will be explored. Additionally, refining the existing codebase to reduce computational overhead will help ensure faster processing, even with larger datasets and higher production volumes.

3. Improvement in High-Speed Operations

At higher operational speeds, the system encounters challenges such as slightly blurred images and off-centered pill alignment, which negatively impact detection accuracy. Future enhancements will include upgrading the camera system to handle higher frame rates and ensure consistent image clarity. Furthermore, mechanical alignment mechanisms will be redesigned to maintain pill centering and stability during high-speed operations, improving the system's overall reliability.

4. User Interface Improvements

To enhance usability, the system's interface will be redesigned to provide a more intuitive and user-friendly experience. Future versions will include customizable settings,

allowing operators to adjust parameters such as speed, detection sensitivity, and output reporting to suit specific production requirements. Providing detailed visual feedback and system diagnostics will also improve operational transparency and ease of use.

5. Integration with Automated Sorting Systems

Finally, integrating the system with automated sorting and packaging solutions will enhance its functionality within production workflows. Future iterations will focus on developing mechanisms for the immediate segregation of defective pills based on detection results. This will streamline the quality assurance process and enable compatibility with existing sorting and packaging systems, contributing to a more efficient and automated manufacturing environment.

REFERENCES

[1] H. Xu, M. Wang, C. Liu, F. Li, and C. Xie, "Automatic detection of tunnel lining crack based on mobile image acquisition system and deep learning ensemble model," Tunnelling and Underground Space Technology, vol. 154, p. 106124, Oct. 2024, doi: 10.1016/j.tust.2024.106124.

[2] B. Keelson et al., "The use of cardiac CT acquisition mode for dynamic musculoskeletal imaging," Physica Medica, vol. 104, pp. 75–84, Nov. 2022, doi: 10.1016/j.ejmp.2022.10.028.

[3] Z. Dai, "Image acquisition technology for unmanned aerial vehicles based on YOLO - Illustrated by the case of wind turbine blade inspection," Systems and Soft Computing, vol. 6, p. 200126, Aug. 2024, doi: 10.1016/j.sasc.2024.200126.

[4] D. Al-Alimi and M. a A. Al-Qaness, "Enhancing Forensic Blood Detection Using Hyperspectral Imaging and Advanced Preprocessing Techniques," Talanta, vol. 283, p. 127097, Oct. 2024, doi: 10.1016/j.talanta.2024.127097.

[5] K. U. Ahamed et al., "A deep learning approach using effective preprocessing techniques to detect COVID-19 from chest CT-scan and X-ray images," Computers in Biology and Medicine, vol. 139, p. 105014, Nov. 2021, doi: 10.1016/j.compbiomed.2021.105014.

[6] Y. Fu, Z. Huang, Y. Zhao, B. Xi, Y. Zhao, and Q. Han, "A weakly supervised soil pore segmentation method based on traditional segmentation algorithm," CATENA, vol. 249, p. 108660, Dec. 2024, doi: 10.1016/j.catena.2024.108660.

[7] J. S. Cardenas-Gallegos, P. M. Severns, P. Klimeš, L. N. Lacerda, A. Peduzzi, and R. S. Ferrarezi, "Reliable plant segmentation under variable greenhouse illumination conditions," Computers and Electronics in Agriculture, vol. 229, p. 109711, Dec. 2024, doi: 10.1016/j.compag.2024.109711.

[8] V. Ghobadi, L. I. Ismail, W. Z. W. Hasan, H. Ahmad, H. R. Ramli, N. M. H. Norsahperi, A. Tharek, and F. A. Hanapiah, “Challenges and solutions of deep learning-based automated liver segmentation: A systematic review,” Computers in Biology and Medicine, vol. 185, p. 109459, Jan. 2025. [Online]. Available: https://doi.org/10.1016/j.compbiomed.2024.109459.

[9] M. Faseeh, M. Bibi, M. A. Khan, and D.-H. Kim, “Deep Learning assisted real-time object recognition and depth estimation for enhancing emergency response in adaptive environment,” Results in Engineering, vol. 24, p. 103482, Nov. 2024, doi: 10.1016/j.rineng.2024.103482.

[10] A. A. M. Muzahid et al., “Deep learning for 3D object recognition: A survey,” Neurocomputing, vol. 608, p. 128436, Aug. 2024, doi: 10.1016/j.neucom.2024.128436.

[11] Q. Zhang et al., “Highly Robust Electronic Skin for Object Recognition of Robot Hand based on Carbon Nanomaterials and SEBS,” Sensors and Actuators a Physical, vol. 379, p. 115922, Oct. 2024, doi: 10.1016/j.sna.2024.115922.

[12] S. V. A. Kumer, P. Kanakaraja, S. Areez, Y. Patnaik, and P. T. Kumar, “An implementation of virtual white board using open CV for virtual classes,” Materials Today Proceedings, vol. 46, pp. 4031–4034, Jan. 2021, doi: 10.1016/j.matpr.2021.02.544.

[13] X. Zabulis et al., “Advances on the detection and measurement of bubble contours during subcooled boiling in microgravity,” Measurement, vol. 222, p. 113644, Oct. 2023, doi: 10.1016/j.measurement.2023.113644.

[14] R. Mehboob, H. Dawood, and H. Dawood, “An encoded histogram of ridge bifurcations and contours for fingerprint presentation attack detection,” Pattern Recognition, vol. 143, p. 109782, Jun. 2023, doi: 10.1016/j.patcog.2023.109782.

[15] J. J. Lv et al., “Contour extraction of medical images using an attention-based network,” Biomedical Signal Processing and Control, vol. 84, p. 104828, Mar. 2023, doi: 10.1016/j.bspc.2023.104828.

[16] Z. Pan, P. Jiang, Q. Zeng, G. Li, and C. Tu, “Category-agnostic semantic edge detection by measuring neural representation randomness,” Pattern Recognition, vol. 156, p. 110820, Jul. 2024, doi: 10.1016/j.patcog.2024.110820.

[17] L. Wang, Z. Liu, Q. Xue, W. Zhu, and S. Zhao, “Isotropic and anisotropic edge detection based on Fourier single pixel imaging,” Optics & Laser Technology, vol. 179, p. 111300, Jun. 2024, doi: 10.1016/j.optlastec.2024.111300.

[18] Y. Yu and Y. Yan, “Color image hybrid noise filtering algorithm based on deep convolution neural network,” Systems and Soft Computing, vol. 6, p. 200120, Jul. 2024, doi: 10.1016/j.sasc.2024.200120.

[19] S. Mondal, A. K. Pal, and S. H. Islam, “CBIF-M: A content-based image filtering and retrieval scheme using multi-classifier filtering framework,” Computers & Electrical Engineering, vol. 118, p. 109450, Jul. 2024, doi: 10.1016/j.compeleceng.2024.109450.

[20] H. Jia, Q. Yin, and M. Lu, “Weighted guided image filtering with entropy evaluation weighting,” Computers & Graphics, vol. 117, pp. 114–123, Oct. 2023, doi: 10.1016/j.cag.2023.10.022.

[21] R. K. Jain, “Experimental performance of smart IoT-enabled drip irrigation system using and controlled through web-based applications,” Smart Agricultural Technology, vol. 4, p. 100215, Mar. 2023, doi: 10.1016/j.atech.2023.100215.

APPENDICES

Appendix A Programming Code

```
import os
import cv2
import numpy as np
import imutils
import mysql.connector
import time
from time import sleep
import threading
import RPi.GPIO as GPIO
from scipy.spatial import distance as dist
from imutils import perspective

# Output folder
output_folder = "/var/www/html/smart_pill_dispenser/defected_image"
os.makedirs(output_folder, exist_ok=True)

mydb = mysql.connector.connect(
  host="localhost",
  user="pi",
  password="root",
  database="pill_dispenser"
)

# Function to calculate midpoint
def midpoint(ptA, ptB):
    return ((ptA[0] + ptB[0]) * 0.5, (ptA[1] + ptB[1]) * 0.5)

# Initialize camera
cap = cv2.VideoCapture(0)

# Distance from camera to object in centimeters
distance_to_object = 9

# Focal length of the camera
focal_length = 500  # distance between the lens and the image sensor in mm

# Colour detection parameters (HSV model)
lower_range_green = np.array([77, 80, 56])
upper_range_green = np.array([121, 148, 92])

lower_range_red = np.array([173, 148, 112])
upper_range_red = np.array([179, 205, 188])
```

```
lower_range_yellow = np.array([15, 114, 140])
upper_range_yellow = np.array([28, 191, 189])

lower_range_blue = np.array([98, 120, 125])
upper_range_blue = np.array([112, 226, 197])

lower_range_white = np.array([106, 12, 188])
upper_range_white = np.array([124, 36, 230])

lower_range_black = np.array([97, 34, 26])
upper_range_black = np.array([168, 64, 249])

# Initialize pixelsPerMetric
pixelsPerMetric = None

# Setup GPIO for servos
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(3, GPIO.OUT)  # to send electrical signals
GPIO.setup(5, GPIO.OUT)
pwm = GPIO.PWM(3, 50)  # 50Hz servo defect
pwm.start(0)

pwm_pill = GPIO.PWM(5, 50)  # 50Hz servo pill
pwm_pill.start(0)

# Function to set servo angle
def SetAngle(angle, pwm_channel):
    duty = angle / 18 + 2
    pwm_channel.ChangeDutyCycle(duty)
    sleep(1)
    pwm_channel.ChangeDutyCycle(0)

# Function to control the servo in a separate thread
def control_defect_servo():
    SetAngle(100, pwm)  # Activate servo
    sleep(5)
    SetAngle(0, pwm)  # Reset servo

# Thread function to run the pwm_pill servo continuously
def run_pill_servo():
  while True:
     SetAngle(0, pwm_pill)
     sleep(2)
     SetAngle(100, pwm_pill)
     sleep(1)

# Start the pill servo thread
pill_servo_thread = threading.Thread(target=run_pill_servo)
pill_servo_thread.daemon = True  # Daemonize thread to exit when the main program exits
```

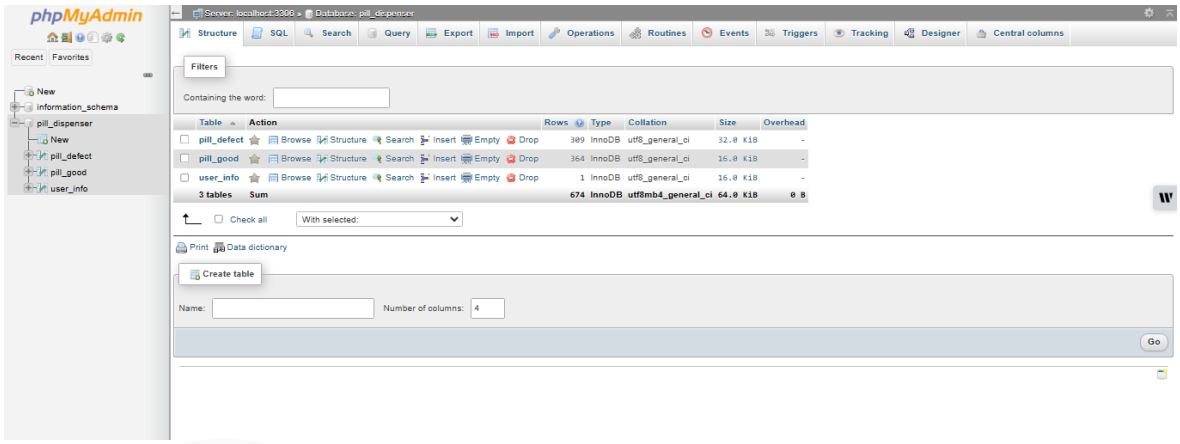
```
pill_servo_thread.start()

# Function to detect object color
def detect_color(frame, orig, frame_centerX, frame_centerY, threshold):
    global pixelsPerMetric  # Declare pixelsPerMetric as global

    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    mask_green = cv2.inRange(hsv, lower_range_green, upper_range_green)
    mask_red = cv2.inRange(hsv, lower_range_red, upper_range_red)
    mask_yellow = cv2.inRange(hsv, lower_range_yellow, upper_range_yellow)
    mask_blue = cv2.inRange(hsv, lower_range_blue, upper_range_blue)
    mask_white = cv2.inRange(hsv, lower_range_white, upper_range_white)
    mask_black = cv2.inRange(hsv, lower_range_black, upper_range_black)

    masks = [(mask_green, "Green"), (mask_red, "Red"), (mask_yellow, "Yellow"),
(mask_blue, "Blue"), (mask_white, "White"), (mask_black, "Black")]

    objects_info = []  # To store color information

    for mask, color_name in masks:
        _, mask_thresh = cv2.threshold(mask, 254, 255, cv2.THRESH_BINARY)
        cnts, _ = cv2.findContours(mask_thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_NONE)

        for c in cnts:
            area = cv2.contourArea(c)
            if 1000 < area < 12000:  # Rectangular object
                # Calculate object dimensions and draw bounding box
                box = cv2.minAreaRect(c)
                box = cv2.boxPoints(box) if imutils.is_cv2() else cv2.boxPoints(box)
                box = np.array(box, dtype="int")
                box = perspective.order_points(box)
                cv2.drawContours(orig, [box.astype("int")], -1, (0, 255, 64), 2)

                for (x, y) in box:
                    cv2.circle(orig, (int(x), int(y)), 5, (0, 255, 64), -1)

                (tl, tr, br, bl) = box
                (tltrX, tltrY) = midpoint(tl, tr)
                (blbrX, blbrY) = midpoint(bl, br)
                (tlblX, tlblY) = midpoint(tl, bl)
                (trbrX, trbrY) = midpoint(tr, br)

                # Calculate object center
                object_centerX, object_centerY = midpoint((tltrX, tltrY), (blbrX, blbrY))

                # Check if object center is within the central rectangle
                if (frame_centerX - threshold < object_centerX < frame_centerX + threshold)
and (frame_centerY - threshold < object_centerY < frame_centerY + threshold):
                    # Calculate dimensions in centimeters
```

```
            lebar_pixel = dist.euclidean((tltrX, tltrY), (blbrX, blbrY))
            panjang_pixel = dist.euclidean((tlblX, tlblY), (trbrX, trbrY))
            if pixelsPerMetric is None:
                pixelsPerMetric = (lebar_pixel + panjang_pixel) / 2  # in pixel
            lebar_cm = (lebar_pixel / focal_length) * distance_to_object
            panjang_cm = (panjang_pixel / focal_length) * distance_to_object

            # Append object information to the list
            objects_info.append((color_name, panjang_cm, lebar_cm, object_centerX,
object_centerY))

    return objects_info

while True:
    ret, frame = cap.read()
    if not ret:
        break

    frame = cv2.resize(frame, None, fx=1, fy=1, interpolation=cv2.INTER_AREA)
    resized_frame = cv2.resize(frame, (640, 480))
    orig = resized_frame.copy()

    # Calculate the center of the frame
    frame_centerX, frame_centerY = resized_frame.shape[1] // 2, resized_frame.shape[0] //
2
    threshold = 110  # Define a threshold for the central region

    # Draw central rectangle (region of interest)
    cv2.rectangle(orig, (frame_centerX - threshold, frame_centerY - threshold),
                  (frame_centerX + threshold, frame_centerY + threshold), (0, 255, 0), 2)

    # Detect if there's any object
    gray = cv2.cvtColor(resized_frame, cv2.COLOR_BGR2GRAY)
    _, thresh = cv2.threshold(gray, 200, 255, cv2.THRESH_BINARY)
    cnts, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    # Color detection
    objects_info = detect_color(resized_frame, orig, frame_centerX, frame_centerY,
threshold)

    # Initialize variables to track color status
    has_green = False
    has_yellow = False
    has_blue = False
    has_white = False
    has_red = False
    has_black = False

    # Variable to check if any object is detected
```

```
    object_detected = False

    info_lines = []  # Store lines of information for each object
    for color, height, width, centerX, centerY in objects_info:
        object_detected = True  # Set this to True when any object is detected

        if (frame_centerX - threshold < centerX < frame_centerX + threshold) and
(frame_centerY - threshold < centerY < frame_centerY + threshold):
            if color == "Green" and 0.5 < height < 1.1 and 0.5 < width < 1.1:
                has_green = True
            if color == "Yellow" and 0.5 < height < 1.1 and 0.5 < width < 1.1:
                has_yellow = True
            if color == "Blue" and 0.5 < height < 1.1 and 0.5 < width < 1.1:
                has_blue = True
            if color == "White" and 0.5 < height < 1.1 and 0.5 < width < 1.1:
                has_white = True
            if color == "Red" and 0.5 < height < 1.1 and 0.5 < width < 1.1:
                has_red = True
            if color == "Black" and 0.5 < height < 1.1 and 0.5 < width < 1.1:
                has_black = True

        # Display object details
        info_line = f"Color: {color}, H:{height:.1f}CM, W:{width:.1f}CM"
        info_lines.append(info_line)
        cv2.putText(orig, info_line, (10, 30 + 30 * len(info_lines)),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 64), 2)

    if object_detected:
        mycursor = mydb.cursor()

        rounded_date = time.strftime('%Y-%m-%d')
        rounded_time = time.strftime('%H:%M:%S')

        if has_green or has_yellow or (has_green and has_yellow):
            print("Success (Pill A)")

            pill_name = 'A (Paracetamol)'

            sqlQueryA = "INSERT INTO pill_good (good_date, good_time, pill_type)
VALUES (%s, %s, %s)"
            valInsert_A = (rounded_date, rounded_time, pill_name)
            mycursor.execute(sqlQueryA, valInsert_A)
            mydb.commit()

            print(mycursor.rowcount, "Record inserted (Pill A).")

        elif has_blue or has_white or (has_blue and has_white):
            print("Success (Pill B)")

            pill_name = 'B (Aspirin)'
```

```
            sqlQueryB = "INSERT INTO pill_good (good_date, good_time, pill_type)
VALUES (%s, %s, %s)"
            valInsert_B = (rounded_date, rounded_time, pill_name)
            mycursor.execute(sqlQueryB, valInsert_B)
            mydb.commit()

            print(mycursor.rowcount, "Record inserted (Pill B).")

        elif has_red or has_black or (has_red and has_black):
            print("Success (Pill C)")

            pill_name = 'C (Adezio)'

            sqlQueryC = "INSERT INTO pill_good (good_date, good_time, pill_type)
VALUES (%s, %s, %s)"
            valInsert_C = (rounded_date, rounded_time, pill_name)
            mycursor.execute(sqlQueryC, valInsert_C)
            mydb.commit()

            print(mycursor.rowcount, "Record inserted (Pill C).")

        else:
            cv2.putText(orig, "Defect", (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            print("Defect")

            threading.Thread(target=control_defect_servo).start()
            image_name = f"{rounded_date}_{rounded_time.replace(':', '-')}.jpg"
            image_path = os.path.join(output_folder, image_name)
            cv2.imwrite(image_path, frame)  # Save the captured image

            sqlQuery = "INSERT INTO pill_defect (defect_date, defect_time, defect_image)
VALUES (%s, %s, %s)"
            valInsert = (rounded_date, rounded_time, image_name)
            mycursor.execute(sqlQuery, valInsert)
            mydb.commit()

            print(mycursor.rowcount, "Record inserted")

        # Introduce a 5-second delay before continuing to the next detection
        sleep(5)

    cv2.imshow('Original', orig)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
GPIO.cleanup()
```

Appendix B phpMyAdmin Database Configuration



Appendix C User Registration Interface