

**Bindura University
of Science Education**



Bindura University of Science Education

Faculty of Science and Physics

Electronic engineering

2026

FINAL YEAR PROJECT TITLE:

Automated colour-based pepper sorting

BY: Anesu Bhobho

REGISTRATION NUMBER: B220221B

PROJECT SUPERVISOR: Eng T Ngombe

Declarations

I hereby certify that the work which is being presented in the project entitled Automated colour based pepper sorting in the partial fulfillment of the requirements for the award of the degree of **BACHELOR OF SCIENCE** and submitted in the **DEPARTMENT OF ELECTRONIC ENGINEERING, BINDURA UNIVERSITY OF SCIENCE EDUCATION**, is an authentic record of my own work carried out during the time period from **August 2022 to June 2026** under the supervision of **Mr T Ngombe, Electronic engineering department**.

The matter presented in this project has not been submitted by me for the award of any other degree of this or any other institute.

Student

Anesu Bhobho

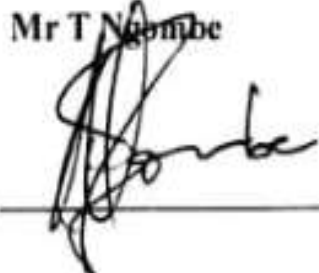


Date: 27 June 2026

This is to certify that the above statement made by the candidate is correct to the best of my/our knowledge.

Supervisor

Mr T Ngombe



Date

27/08/26

Copyright

© 2026 Anesu Bhobho

All rights reserved.

No part of this dissertation may be reproduced, stored in a retrieval system, transmitted, or distributed in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author, except for brief quotations used for academic purposes with proper acknowledgement.

This dissertation is submitted in partial fulfilment of the requirements for the award of the Bachelor of Science Honours Degree in Electronic Engineering at Bindura University of Science Education.

The views and opinions expressed in this dissertation are those of the author and do not necessarily reflect those of Bindura University of Science Education.

Dedications

This project is dedicated to my family, whose unwavering love, support and encouragement have been a constant source of motivation throughout my academic journey. Their sacrifices, patience and belief in my abilities have enabled me to pursue my studies and overcome the challenges encountered along the way.

I also dedicate this work to my lecturers and mentors who have shared their knowledge and inspired me to strive for excellence in the field of Electronic Engineering.

Finally, this project is dedicated to all aspiring engineers and innovators who seek to use technology to solve real-world problems and contribute to the advancement of society.

Acknowledgement

First and foremost, I would like to thank Almighty God for His guidance, strength and blessings throughout the duration of this project and my studies.

I express my sincere gratitude to my supervisor Mr T Ngombe for his invaluable guidance, technical expertise, constructive feedback and continuous support throughout the development of this project. His mentorship played a significant role in the successful completion of this research.

I would also like to extend my appreciation to the lecturers and staff of the Department of Electronic Engineering at Bindura University of Science Education for the knowledge, skills and support they provided during my academic journey.

Special thanks are extended to my family and friends for their encouragement, patience, understanding and unwavering support throughout my studies. Their motivation and belief in me provided the determination needed to complete this project successfully.

I am also grateful to my fellow students and colleagues who contributed ideas, suggestions and assistance during the design, implementation and testing phases of this work.

Finally, I acknowledge the developers and researchers behind the various technologies and tools used in this project, including Arduino IDE, ESP32 development platforms and Edge Impulse, whose resources greatly facilitated the development of the automated bell pepper sorting system.

Abstract

Bell pepper sorting is an important post-harvest process that influences product quality, market value and processing efficiency. Traditional manual sorting methods are labour-intensive, timeconsuming and susceptible to human error, while many automated sorting systems remain expensive and inaccessible to small-scale farmers. This project presents the design and implementation of an automated bell pepper colour sorting system based on computer vision and TinyML technologies.

The proposed system consists of an ESP32-CAM module for image acquisition and colour classification, an ESP32 microcontroller for system coordination, an infrared sensor for pepper detection, a conveyor belt driven by stepper motors, and servo-actuated sorting mechanisms. An Edge Impulse TinyML model was trained using a dataset of 600 training images and 152 testing images across four classes, green, yellow, red and background. The trained model was deployed directly on the ESP32-CAM for real-time inference.

Experimental results showed that the Edge Impulse model achieved 100% precision, recall and F1-score on the validation dataset. When integrated into the complete prototype, the system achieved an operational sorting accuracy of 99.6% under normal lighting conditions. The average inference time was approximately 150 ms per image, while the system processed one pepper every two seconds. The results demonstrate that low-cost embedded machine learning can provide accurate and reliable agricultural sorting without requiring high-performance computing hardware.

The developed prototype offers a practical, affordable and scalable solution for automated pepper sorting and demonstrates the potential of TinyML-based computer vision systems in agricultural automation.

Table of abbreviations

ABBREVIATION	MEANING
AI	ARTIFICIAL INTELLIGENCE
CNN	CONVOLUTIONAL NEURAL NETWORK
DC	DIRECT CURRENT
ESP32	ESPRESSIF 32-BIT MICROCONTROLLER
ESP32-CAM	ESP32 CAMERA MODULE
F1-SCORE	HARMONIC MEAN OF PRECISION AND RECALL
GUI	GRAPHICAL USER INTERFACE
GPIO	GENERAL PURPOSE INPUT/OUTPUT
IR	INFRARED
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
IoT	INTERNET OF THINGS
LDR	LIGHT DEPENDENT RESISTOR
ML	MACHINE LEARNING
OV2640	OMNIVISION 2 MEGAPIXEL CAMERA SENSOR
PWM	PULSE WIDTH MODULATION
RAM	RANDOM ACCESS MEMORY
RGB	RED, GREEN AND BLUE
TINYML	TINY MACHINE LEARNING
UART	UNIVERSAL ASYNCHRONOUS RECEIVER-TRANSMITTER
USB	UNIVERSAL SERIAL BUS
VCC	VOLTAGE AT COMMON COLLECTOR (POWER SUPPLY VOLTAGE)

Contents

1 Chapter 1: Introduction	1
1.1 Background.....	1
1.2 Problem Statement	1
1.3 Aim	2
1.4 Objectives.....	2
1.5 Scop of the study.....	2
1.6 Significancy of the study.....	2
1.7 Summary	3
2 Chapter 2: Literature review	4
2.1 Introduction.....	4
2.2 Importance of Automated pepper sorting systems	4
2.3 TinyML in Agricultural Automation	5
2.4 ESP32-CAM in Agricultural Automation	6
2.5 Edge Impulse for Embedded Machine Learning	7
2.6 Related works.....	8
2.6.1 Automatic Bell Pepper Colour Detector and Sorting Machine.....	8
2.6.2 A Four-Level Maturity Index for Hot Peppers.....	9
2.6.3 CNN-Based Fruit Classification Systems	10
2.7 Research Gaps	10
2.8 Existing systems analysis	11
2.9 Summary	12
3 Chapter 3: Methodology	13
3.1 Introduction.....	13
3.2 Project approach	13
3.3 System Requirements Specification	14
3.3.1 Hardware requirement.....	14
3.3.2 Software requirements	17
3.4 System Architecture	19
3.4.1 Subsystem units.....	20
3.4.2 Signal flow description	22
3.5 Flow chart.....	23
3.6 System design and implementation	24
3.6.1 Circuit diagrams.....	25
3.6.2 Hardware design	25
3.6.3 Software design.....	29
3.6.4 Error Handling and Safety Mechanisms	34

3.7 Ethical considerations	40
3.7.1 Human and Animal Subjects	40
3.7.2 Electrical Safety	40
3.7.3 Mechanical Safety	40
3.7.4 Waste Minimisation and Environmental Responsibility.....	40
3.7.5 Intellectual Property and Open-Source Compliance.....	41
3.8 Summary	41
4 Chapter 4: EXPERIMENTAL RESULTS AND ANALYSIS	41
4.1 Introduction.....	41
4.2 System implementation results.....	42
4.2.1 Edge Impulse Model Performance	43
4.3 Data presentation.....	44
4.3.1 Sorting Accuracy Test	44
4.4 Throughput Performance	45
t	45
4.5 Servo Motor Response Time	46
4.6 Lighting Condition Sensitivity	46
4.7 Objective achievements	47
4.7.1 Performance Improvements.....	50
4.7.2 Shortcomings identified	50
4.8 Discussion of Findings	51
4.8.1 Accuracy Improvement	51
4.8.2 Processing Speed.....	51
4.8.3 Effect of Lighting Conditions.....	52
4.8.4 Future Improvements to the TinyML Model.....	52
4.8.5 Addressed research gaps.....	52
4.9 Summary	53
5 CHAPTER 5: CONCLUSION AND RECOMMENDATIONS.....	54
5.1 Summary of the Study.....	54
5.2 Limitations.....	54
5.3 Recommendations for Future Work.....	55
5.4 Summary	55
Similarity report for chapter 1 to chapter 5	58

List of figures

FIGURE 2:1 THE SYSTEM DESIGN (OBJECT DETECTION)	8
FIGURE 2:2 THE SYSTEM DESIGN (OBJECT EJECTION SYSTEM)	9
FIGURE 3:1 ESP32 MICROCONTROLLER	14
FIGURE 3:2 ESP32 CAM MODULE	15
FIGURE 3:3 STEPPER MOTOR	16
FIGURE 3:4 SERVO MOTOR	16
FIGURE 3:5 IR SENSOR	17
FIGURE 3:6 ARDUINO IDE WORKSPACE	18
FIGURE 3:7 EDGE IMPULSE WORKING ENVIRONMENT	19
FIGURE 3:8 OVERAL BLOCK DIAGRAM OF THE SYSTEM	20
FIGURE 3:9 COMPLETE CIRCUIT DIAGRAM OF THE AUTOMATED PEPPER SORTING SYSTEM	25
FIGURE 3:10 SCHEMATIC DIAGRAM ESP32CAM INTERFACING	26
FIGURE 3:11 SCHEMATIC DIAGRAM OF SERVO MOTOR INTERFACING	27
FIGURE 3:12 STEPPER MOTOR INTERFACING	28
FIGURE 3:13 INFRARED SENSOR INTERFACING	29
FIGURE 3:14 SYSTEM ARCHITECTURE	30
FIGURE 3:15 PIN DEFINITION	31
FIGURE 3:16 STEPPER MOTOR CONTROL CODE	31
FIGURE 3:17 SERVO CONTROL FUNCTION	32
FIGURE 3:18 UART COMMUNICATION	32
FIGURE 3:19 MAIN CONTROLLER LOOP.....	33
FIGURE 3:20 CAMERA INITIALISATION	33
FIGURE 3:21 ESP32CAM LOOP	34
FIGURE 3:22 IMAGE COLLECTION WEB CODE	35
FIGURE 3:23 MODEL TRAINING SNIPPETS.....	36
FIGURE 4:1 PEPPER SORTER	43
FIGURE 4:2 EDGE IMPULSE VALIDATION RESULTS SHOWING CONFUSION MATRIX, PRECISION, RECALL AND F1-SCORE FOR THE TRAINED BELL PEPPER CLASSIFICATION MODEL.	44
FIGURE 4:3 CONVEYOR SYSTEM WITH 28BYJ-48 STEPPER MOTOR	47
FIGURE 4:4 WORKING INFERENCING CODE	48
FIGURE 4:5 SERVO PLACEMENT ON THE CONVEYOR	49

List of tables

TABLE 2:1 GAPS IN EXISTING STUDIES	11
TABLE 2:2 EXISTING SYSTEM LIMITATIONS	12
TABLE 3:1 ESP32CAM INTERFACING PIN ALLOCATION TABLE	26
TABLE 3:2 SERVO MOTOR PIN ALLOCATION	27
TABLE 3:3 STEPPER MOTOR PIN CONNECTIONS	28
TABLE 3:4 IR CONNECTIONS PINS	29
A DATASET OF BELL PEPPER IMAGES WAS COLLECTED AND USED TO TRAIN THE EDGE IMPULSE TINYML MODEL. FOUR CLASSES WERE CREATED: GREEN, YELLOW, RED, AND BACKGROUND. EACH CLASS CONTAINED 150 TRAINING IMAGES AND 38 TESTING IMAGES, AS SHOWN IN TABLE 3:5	37
TABLE 3:6 TRAINING DATA TABLE	37
TABLE 4:1 EDGE IMPULSE VALIDATION RESULTS	45
TABLE 4:2 THROUGHPUT ANALYSIS	46
TABLE 4:3 SERVO MOTOR RESPONSE TIME	46
TABLE 4:4 ACCURACY UNDER DIFFERENT LIGHTING CONDITIONS	46
TABLE 4:5 ADDRESSED RESEARCH GAPS	53

List of Equations

EQUATION 3:1 STEP CALIBRATION	38
EQUATION 3:2 LINEAR DISPLACEMENT	39
EQUATION 3:3 BELT LINEAR VELOCITY	39
EQUATION 3:4 SERVO RESPONSE	40
EQUATION 4:1 THROUGHPUT CALCULATION EQUATION	45

1 Chapter 1: Introduction

1.1 Background

In Zimbabwe, the sorting and grading of horticultural produce is predominantly performed manually, particularly among small-scale farmers and local packaging facilities. Manual sorting relies on visual inspection and human judgement, which can result in inconsistencies due to fatigue, subjective assessment and variations in working conditions (Saleem, Potgieter and Arif, 2021). Such inconsistencies may lead to non-uniform product quality, reduced market value and increased post-harvest losses.

Peppers (*Capsicum*) were selected for this study because colour is a key quality attribute used to determine maturity and marketability. Accurate colour-based sorting is therefore essential for maintaining product quality and meeting market requirements (Bazargani and Deemyad, 2024a). Achieving consistent and efficient sorting through manual methods remains challenging, particularly where large quantities of produce must be processed. This creates a need to investigate cost-effective automated sorting approaches that can improve sorting accuracy, reduce labour dependency and enhance post-harvest handling efficiency.

1.2 Problem Statement

Pepper (*Capsicum*) is a commercially important horticultural crop whose quality and market value are strongly influenced by external attributes such as colour and maturity stage. Accurate sorting is therefore essential for meeting market standards and maximizing profitability. Studies have shown that traditional sorting and grading processes are often performed manually and rely on human judgement, making them labour-intensive, time-consuming and susceptible to inconsistencies caused by fatigue and subjective assessment (Rasekh *et al.*, 2022).

Inconsistent grading can result in non-uniform product quality, reduced market competitiveness and increased post-harvest losses. Bell peppers are particularly vulnerable to post-harvest quality deterioration, with losses reported to approach 40% under inadequate post-harvest handling conditions (Shin *et al.*, 2021). Additionally colour-based sorting is critical because colour serves as a key indicator of maturity and consumer acceptance, directly influencing product classification and market value.

Automated grading technologies have been developed to improve sorting accuracy and efficiency, the continued reliance on manual sorting in many horticultural operations highlights the need for

cost-effective and accessible alternatives. This implies that there is a need to investigate an automated colour-based pepper sorting system capable of improving sorting consistency, reducing dependence on manual labour, and enhancing post-harvest handling efficiency.

1.3 Aim

To design and construct an automated bell pepper sorting system capable of classifying peppers according to colour and sorting them into the appropriate categories with minimal human intervention.

1.4 Objectives

- To construct a conveyor system that presents fruits individually to the vision system.
- To develop an image processing algorithm capable of detecting a fruit's presence and analysing its dominant colour in real-time.
- To design and integrate a mechanical sorting mechanism activated by the processing unit's decisions.

1.5 Scop of the study

This study is limited to the design, development and laboratory evaluation of a prototype automated pepper (*Capsicum*) sorting system based on colour classification. The system focuses on sorting peppers into three colour categories, green, yellow and red. Performance evaluation is restricted to sorting accuracy, throughput and operational reliability under controlled laboratory conditions at Bindura University of Science Education.

The study is intended as a proof of concept for small-scale and medium-scale post-harvest horticultural applications. It excludes commercial deployment, long-term durability testing, outdoor operation, defect detection, size grading, weight-based classification and the sorting of crops other than bell peppers. Cost evaluation is limited to prototype development costs and does not include commercial production or maintenance costs.

1.6 Significancy of the study

This study is significant because it contributes to the advancement of agricultural automation through the development of an embedded vision-based pepper sorting system. By combining computer vision, machine learning and mechatronic technologies within a single prototype, this

research demonstrates the practical application of modern engineering solutions to improve postharvest handling processes (Sharma *et al.*, 2021; Ariza-Sentís *et al.*, 2024).

The study contributes to academic knowledge by providing a practical framework for the design, implementation, and evaluation of an automated sorting system using embedded hardware. It offers valuable insights into the application of image processing and machine learning techniques in agricultural environments and serves as a useful reference for future research in embedded systems, computer vision and precision agriculture.

The research is also significant to the agricultural and engineering industries as it demonstrates the feasibility of using automated systems to perform colour-based produce classification with consistency and reliability. The developed prototype highlights the potential of automation to support quality control processes and improve operational efficiency in produce handling and grading applications.

1.7 Summary

An overview of the background to the study was provided through examining the role of automated colour-based pepper sorting and the challenges associated with manual sorting practices in horticultural production. It identified the need for an automated system capable of improving sorting consistency, reducing labour dependency and enhancing post-harvest handling efficiency through the problem statement. Additionally, study's aim and objectives were outlined to form the basis for the development and evaluation of the proposed vision-based pepper sorting system. The scope of the study was subsequently defined, outlining the boundaries of the research and the parameters considered during prototype evaluation. The significance of the study was also discussed, emphasizing its contributions to agricultural automation, embedded systems, and postharvest processing.

2 Chapter 2: Literature review

2.1 Introduction

The pursuit of automating agricultural processes is a critical response to global challenges in food production, quality control, and economic sustainability. This literature review is situated within this broader context, focusing on the technological evolution of automated sorting systems for fresh produce, with particular emphasis on pepper sorting. As discussed in Chapter 1, manual pepper sorting is associated with several limitations, including subjectivity, inconsistency and high labour requirements. The proposed colour-based automated sorting system builds upon advances in computer vision, embedded systems, and machine learning technologies.

The primary objective of this literature review is to critically analyse and synthesise existing research and technological developments relevant to the design and implementation of an automated pepper sorting system. It seeks to identify established approaches, evaluate their strengths and limitations and highlight gaps that justify the need for the present study.

The chapter begins by discussing the principles of agricultural automation and colour-based sorting systems. It then reviews key technologies employed in automated sorting, including computer vision, image processing, embedded systems, and machine learning techniques. Relevant studies and existing sorting systems are subsequently examined to identify current challenges and limitations. Finally, the chapter presents the research gaps addressed by this study and provides a summary of the literature reviewed.

2.2 Importance of Automated pepper sorting systems

Automated sorting systems have become increasingly important in horticultural production due to their ability to improve the consistency and efficiency of quality assessment processes. In pepper production, sorting is primarily based on external quality attributes such as colour, which serves as an indicator of maturity and market readiness. Traditional manual sorting methods rely on human judgement and visual inspection, making them susceptible to variability caused by fatigue, experience level, and environmental conditions (Elwakeel *et al.*, 2023). As a result, maintaining consistent grading standards can be challenging, particularly when large volumes of peppers must be processed within a limited time.

To address these limitations, researchers have investigated the use of computer vision and automated sorting technologies for colour-based classification of agricultural produce. These systems can apply predefined grading criteria consistently and process larger quantities of produce than manual methods, making them suitable for post-harvest handling applications (Kulkarni *et al.*, 2025). In the context of pepper sorting, automated systems have demonstrated the ability to classify produce according to colour categories with high levels of accuracy, thereby improving grading consistency and supporting quality control processes (Bazargani and Deemyad, 2024).

Despite these advantages, the effectiveness of automated sorting systems depends on several factors. Image-based classification methods are often sensitive to variations in lighting conditions, camera positioning and background interference, which can affect classification accuracy if not properly controlled (Jiayun, Boyang and Leung, 2024). In addition, many advanced sorting systems reported in the literature rely on specialised hardware and computational resources, limiting their adoption in small-scale agricultural operations where affordability remains a key consideration. Consequently, there is a growing research interest in developing low-cost embedded vision systems that can achieve reliable colour classification while remaining accessible to resource-constrained users.

While automated pepper sorting systems offer significant potential for improving post-harvest handling and grading consistency, challenges related to cost, environmental robustness and practical deployment continue to motivate further research and development in this area.

2.3 TinyML in Agricultural Automation

TinyML is an emerging field that combines machine learning with resource-constrained embedded systems, enabling intelligent data processing directly on low-power microcontrollers. Unlike conventional machine learning systems that depend on cloud computing platforms or highperformance processors, TinyML allows inference to be performed locally on edge devices with limited memory, processing power and energy consumption. This capability has attracted significant research interest due to its potential to bring artificial intelligence to low-cost and portable applications(Kallimani *et al.*, 2023) .

The growing adoption of TinyML has been driven by advances in model compression, quantisation and optimisation techniques, which enable complex machine learning models to operate within the hardware constraints of microcontrollers while maintaining acceptable levels of accuracy. As a result, TinyML bridges the gap between artificial intelligence and embedded systems, making

intelligent decision-making feasible in environments where computational resources, connectivity, and power availability are limited (Elhanashi *et al.*, 2024).

In the agricultural sector, TinyML has been applied to a range of tasks including crop disease identification, fruit classification, pest monitoring, and automated quality assessment. By performing data processing directly at the point of acquisition, TinyML-based systems can provide real-time responses without the latency associated with cloud-based processing. This characteristic is particularly important for automated sorting applications, where classification decisions must be made rapidly to maintain throughput and operational efficiency (Kallimani *et al.*, 2023).

Despite its advantages, TinyML systems face challenges related to limited memory capacity, reduced computational resources, and the trade-off between model complexity and classification performance. These constraints require careful optimisation of machine learning models to ensure reliable operation on embedded hardware. Nevertheless, the ability to deliver intelligent functionality using inexpensive and energy-efficient devices has positioned TinyML as a promising technology for agricultural automation, particularly in resource-constrained environments where access to advanced computing infrastructure is limited.

2.4 ESP32-CAM in Agricultural Automation

The ESP32-CAM is a low-cost embedded vision platform that captures and processes images directly on a microcontroller. Unlike conventional colour sensors, it analyses complete images, enabling the extraction of multiple visual features such as colour, shape, size and surface characteristics.

Traditional agricultural sorting systems commonly use Light Dependent Resistors (LDRs) or dedicated colour sensors (Seiphepi *et al.*, 2020). LDRs measure reflected light intensity but cannot distinguish colours directly and are highly susceptible to changes in ambient lighting, resulting in poor classification accuracy. Colour sensors improve performance by measuring red, green and blue (RGB) light components; however, they typically sample only a small area of the object, making them less effective for produce with uneven colour distribution or varying ripeness.

The ESP32-CAM overcomes these limitations by employing image-based colour classification. Instead of relying on point measurements, it analyses the entire visible surface of the produce, enabling more accurate classification of pepper maturity stages (green, yellow and red). In

addition, the same image can be used to extract other quality attributes, including size, shape and surface defects without additional sensors (Kulkarni *et al.*, 2025).

Although the ESP32-CAM has limited processing power and memory compared to highperformance embedded computing platforms, lightweight image processing and machine learning algorithms enable real-time operation. Its ability to perform multi-feature analysis using a single low-cost camera makes it a practical and versatile solution for automated agricultural sorting systems.

2.5 Edge Impulse for Embedded Machine Learning

Edge Impulse is an end-to-end machine learning development platform designed for resourceconstrained edge devices. It provides an integrated environment for data acquisition, data labelling, feature extraction, model training, performance evaluation, and deployment on microcontrollers and embedded systems (Tolani *et al.*, 2025). The platform supports a wide range of hardware, including ESP32-based devices, making it suitable for low-cost intelligent sensing applications.

A key advantage of Edge Impulse is its ability to generate lightweight machine learning models that are optimised for devices with limited memory, processing power and energy consumption. The platform employs techniques such as model quantisation and optimisation to reduce computational requirements while maintaining acceptable classification accuracy. This enables machine learning inference to be performed directly on embedded devices without relying on cloud-based processing (Hymel *et al.*, 2023).

Several studies have demonstrated the effectiveness of Edge Impulse in image classification, object detection, predictive maintenance and agricultural monitoring applications. In computer vision systems, the platform simplifies the development of embedded image classification models by integrating image preprocessing, neural network training, performance validation and automatic code generation within a single workflow. This significantly reduces development time while enabling deployment on low-cost hardware (Mihigo *et al.*, 2022).

Compared with conventional cloud-based machine learning systems, Edge Impulse supports ondevice inference, which reduces communication latency, improves response time, lowers bandwidth requirements, and enhances data privacy. These advantages make it particularly suitable

for real-time agricultural sorting systems, where rapid classification decisions are required and reliable internet connectivity may not always be available.

2.6 Related works

2.6.1 Automatic Bell Pepper Colour Detector and Sorting Machine

(Seiphepi *et al.*, 2020) developed an automated bell pepper sorting machine that uses Light Dependent Resistors (LDRs) and artificial light sources to identify pepper colour. The system detects variations in reflected light intensity and uses these measurements to classify peppers according to colour categories. A conveyor mechanism transports peppers to designated collection points based on the detected colour.

The major advantage of this approach is its simplicity and low implementation cost. However, the system relies heavily on ambient lighting conditions and requires careful calibration. Since LDRs only measure light intensity rather than analysing complete images, the system cannot distinguish complex colour variations or identify defects and shape characteristics. The study primarily reported simulation results in Proteus and did not provide experimental performance metrics such as classification accuracy, precision, or processing speed. **Results of the project**

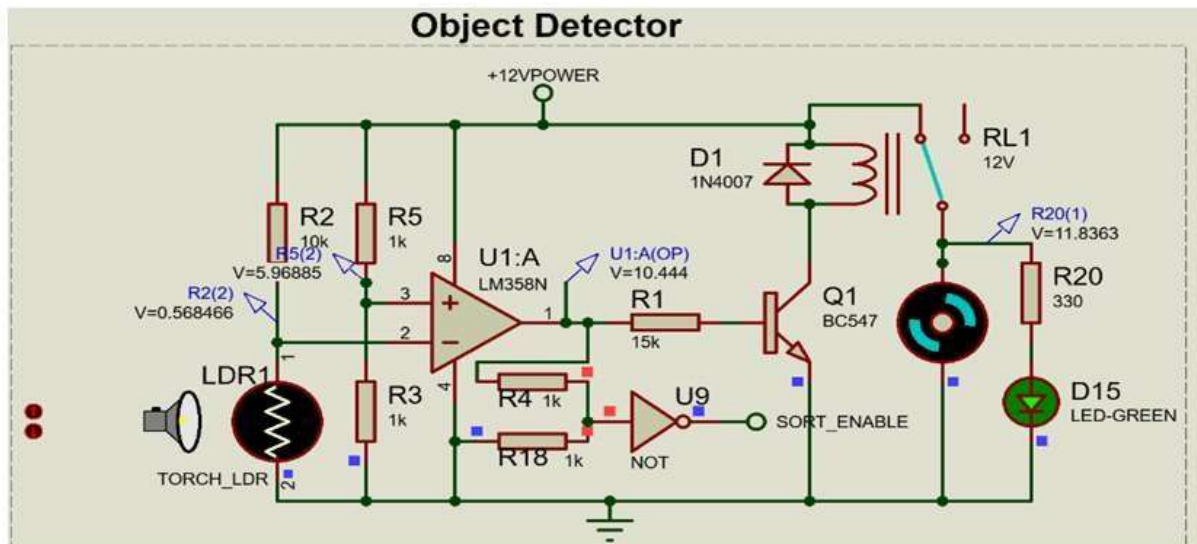


Figure 2:1 The system design (object detection)

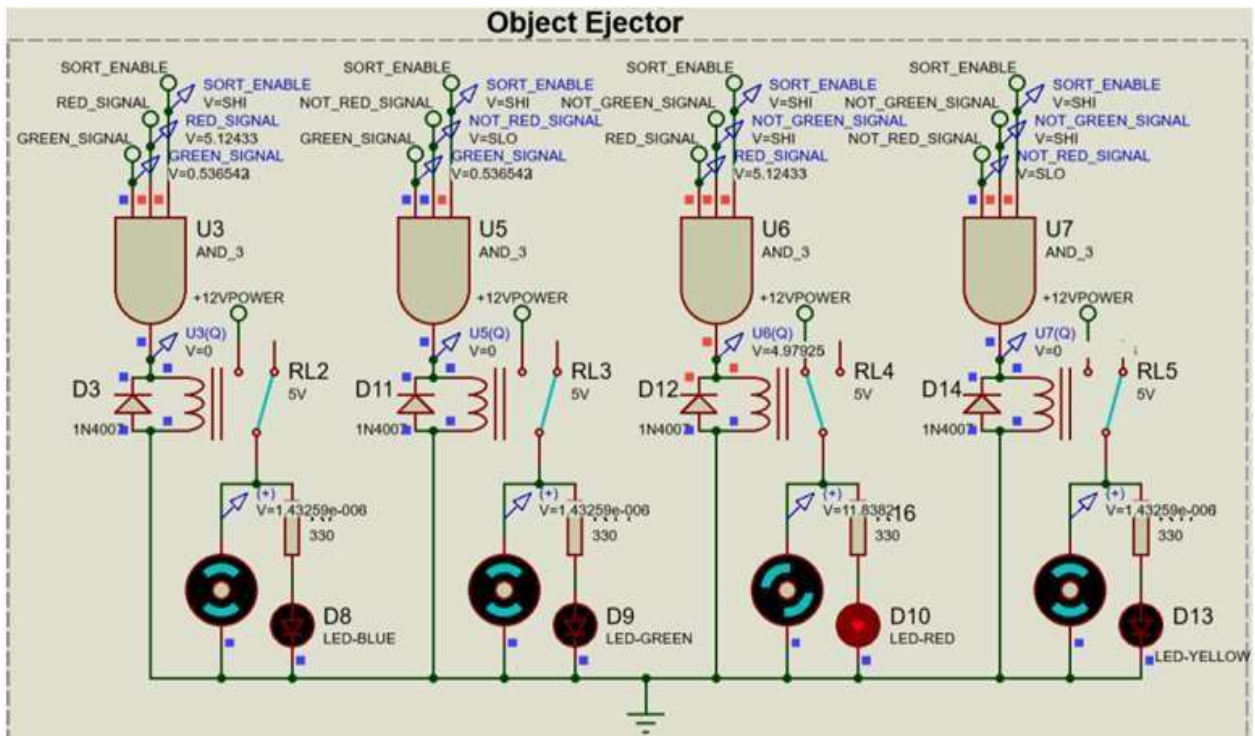


Figure 2:2 The system design (object ejection system)

Figure 2:2 shows the object ejection subsystem which was responsible for directing peppers into their respective collection bins after colour classification. The subsystem uses logic gates and relays to activate dedicated ejector motors corresponding to green, red, yellow or rejected objects. A SORT_ENABLE signal ensures that ejection occurs only when the pepper reaches the sorting position. Once the appropriate colour signal is received, the corresponding relay energises and activates an ejector mechanism that diverts the pepper from the conveyor into the designated bin. Indicator LEDs provide visual confirmation of the sorting action, while flyback diodes protect the control circuitry from relay-induced voltage transients. This arrangement enables automatic, reliable, and real-time sorting of peppers based on colour classification.

2.6.2 A Four-Level Maturity Index for Hot Peppers

(Legner *et al.*, 2021) proposed a non-invasive pepper maturity assessment system based on portable Raman spectroscopy. The system captures spectral signatures from pepper surfaces and analyses them using chemometric models to classify peppers into four maturity levels ranging from immature to fully mature.

The method demonstrated high reliability for assessing biochemical changes associated with ripening, including chlorophyll degradation and carotenoid formation. However, Raman spectrometers are expensive and require specialised equipment and expertise, limiting their

adoption by small-scale farmers. Additionally, the system focuses on maturity estimation rather than real-time automated sorting.

Compared with Raman spectroscopy, the proposed ESP32-CAM solution offers a significantly lower-cost alternative while maintaining practical classification capabilities suitable for real-time sorting operations.

2.6.3 CNN-Based Fruit Classification Systems

Recent studies have demonstrated the effectiveness of Convolutional Neural Networks (CNNs) in automated fruit classification. CNNs automatically learn discriminative image features such as colour, shape, texture and surface patterns directly from training images, eliminating the need for manual feature extraction. (Wang, 2022a) developed an eight-layer CNN-based fruit recognition system and achieved a classification accuracy of 91.63%, demonstrating the suitability of deep learning techniques for agricultural sorting applications.

Similarly, reviews of CNN applications in fruit image processing have shown that deep learning methods consistently achieve high classification performance in fruit recognition, quality assessment and defect detection tasks. These approaches outperform many traditional imageprocessing methods due to their ability to learn complex visual representations from large datasets.

Despite their high accuracy, CNN-based classification systems often require substantial computational resources for model execution. Many implementations depend on highperformance processors, GPUs, Raspberry Pi platforms or cloud-based computing infrastructure, which increases system cost, power consumption and deployment complexity. These requirements can limit adoption in small-scale agricultural environments where affordable and energy-efficient solutions are preferred.

2.7 Research Gaps

Gap	Description	Solutions
-----	-------------	-----------

Narrow sorting criteria	Existing work focuses only on ripeness stages; treats green, yellow, and red as sequential ripeness rather than independent market categories; no verification that the object is actually a pepper.	Sorts explicitly into three independent colour classes (green, yellow, red) as distinct market grades; includes simple contour detection to confirm the object is a pepper before classification.
Lack of accuracy	Heavy reliance on LDRs or basic colour sensors that are highly sensitive to ambient light and fluctuations.	Using ESP32-CAM with Edge Impulse model
High cost	Designs require numerous analogue components, shielding, and complex circuitry.	A cheap model from locally available tools
Outdated technology	Continued use of LDR-based detection despite availability of modern vision and AI solutions.	Adopts embedded tinyML capability via Edge Impulse.
Low processing speed	Uncontrollable motors and slow processing force very low conveyor speeds.	Fast image processing speed-controlled conveyor, and instant servo ejection yield high sorting speeds

Table 2:1 gaps in existing studies

2.8 Existing systems analysis

System	Limitation
--------	------------

Manual sorting	Workers inspect peppers by eye. Slow and results vary between workers (Seiphepi <i>et al.</i> , 2020).
LDR systems	Use light sensors. Accuracy changes when lighting changes (Seiphepi <i>et al.</i> , 2020).
RGB sensors	Detect colour values only and cannot learn image patterns like AI models.
PC-based vision systems	Need a computer for processing, making them expensive and less portable (Kohut and Skop, 2024).
CNN systems	Very accurate but often require Raspberry Pi, GPU, or high-performance processors (Wang, 2022b).

Table 2:2 existing system limitations

Table 2:2 outlines existing pepper sorting approaches limitations compared to the proposed ESP32-CAM and Edge Impulse TinyML-based system. Manual sorting is slow and inconsistent, while LDR systems are affected by lighting conditions. RGB sensors have limited feature extraction and PC-based systems are expensive and power-intensive. CNN-based methods offer high accuracy but require powerful hardware.

2.9 Summary

This literature review examined the development of automated pepper sorting systems and the technologies that support them, including computer vision, embedded systems, TinyML, ESP32CAM platforms, and machine learning techniques. The reviewed studies demonstrated that automation can improve sorting accuracy, consistency and efficiency compared to traditional manual methods. Several research gaps were identified, including the continued reliance on LDR and basic colour sensor technologies, limited classification accuracy under varying lighting conditions, high implementation costs, dependence on resource-intensive computing platforms and low processing speeds in some existing systems. To address these gaps, the proposed study adopts a low-cost ESP32-CAM and Edge Impulse TinyML-based approach capable of real-time pepper colour classification and sorting.

3 Chapter 3: Methodology

3.1 Introduction

This chapter presents the methodology used to design and develop the automated colourbased pepper sorting prototype. It begins by outlining and justifying the selected engineering design approach, followed by the system requirements specification, including the hardware and software components. The chapter then describes the overall system architecture, operational flow, hardware and software implementation, circuit designs and classification algorithms. Finally, ethical considerations relevant to the project are discussed before the chapter summary.

3.2 Project approach

The engineering design approach was employed in this project with strong focus on prototyping. The rationale behind the choice of the approach was that the main objective was not to come up with current scientific principles, but to combine the existing knowledge and technologies known such as computer vision, embedded systems and mechatronics to create a working low-cost solution to a practical and a specific problem. It is inherently iterative and solution oriented with the following major stages in the methodology:

- Requirements Analysis

Determining the functionality and non-functionality requirements of the system according to the literature review and problem statement.

- System Design

Creating the general system architecture, choosing the right hardware and software elements, and developing specific design documents (block diagrams, flowcharts, circuit schematics).

- Implementation (Prototype Development)

Physical prototype construction, embedded software development and incorporation of all subsystems into a unified system.

- Testing and Evaluation

Strictly testing the prototype based on the requirements in order to analyse its performance (accuracy, speed, reliability) and determine how it can be improved.

- Iteration

Optimization of the design and implementation of the work results through the test.

This iterative, prototyping cycle is best suited to this project because it enables the practical testing of ideas and discovery of practical issues that may not be evident within a strictly theoretical analysis and simulations.

3.3 System Requirements Specification

The following requirements were identified as essential for the prototype's functionality.

3.3.1 Hardware requirement

3.3.1.1 Esp32 microcontroller

ESP32 was selected as the core controller because of an exceptional ability of the core to provide a combination of processing capabilities, peripheral integration and low cost. The ESP32 has a dual-core processor with a maximum frequency of 240 MHz, unlike other simpler microcontrollers like the Arduino Uno which only has a single core (Espressif Systems, 2019). This is necessary since this system requires real-time multi-processing capabilities. The step pulses to be sent to the stepper motor and the monitoring of the IR sensor can be done on one core and the UART communication with the ESP32-CAM and the main control logic are implemented on the other core. This hardware parallelism will guarantee that the movement of the conveyors will be smooth and responsive even when the system will be awaiting the results of the colour classification.

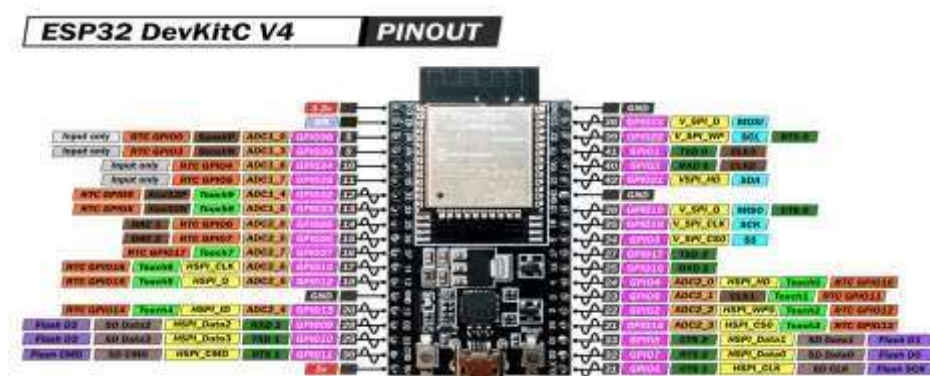
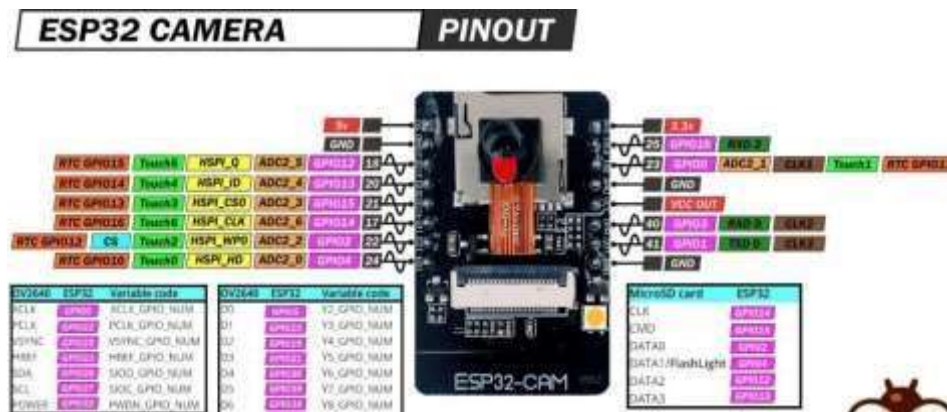


Figure 3:1 ESP32 microcontroller

3.3.1.2 ESP32 cam

The ESP32-CAM module was selected as the vision subsystem due to its integration of an OV2640 camera sensor and an ESP32 processor on a compact, low-cost platform. This integration simplifies the hardware design by eliminating the need for separate camera modules, additional interface circuitry, or external image-processing units. The onboard processor is capable of capturing images and performing colour classification using either image-processing techniques or lightweight machine learning models developed with Edge Impulse. By assigning image acquisition and classification tasks to the ESP32-CAM, the main ESP32 microcontroller is free to

focus on real-time conveyor control, position tracking, and ejection operations. This distributed processing approach improves overall system efficiency, responsiveness, and reliability while maintaining a low hardware cost.



3.3.1.4 Servo motor

The ejection mechanism requires an actuator that can move quickly and precisely to divert peppers into the correct collection bins. A micro servo motor was chosen for this role because it excels at accurate angular positioning. By sending a simple PWM signal from the ESP32, the servo can be commanded to rotate to a specific angle such as zero degrees for green, ninety degrees for yellow, and one hundred eighty degrees for red with no complex feedback or control loops required. This simplicity makes the servo easy to program and integrate into the overall system.



Figure 3:4 Servo motor

3.3.1.5 IR sensor

Reliable detection of a pepper's arrival at the inspection station is essential for triggering the image capture sequence. An infrared (IR) proximity sensor was selected for this purpose due to its simplicity, low cost, and non-contact operation. Unlike mechanical limit switches, which could crush or damage delicate peppers, the IR sensor detects the presence of an object by measuring reflected infrared light. This non-contact method ensures that peppers are handled gently, reducing post-harvest losses.



Figure 3:5 IR sensor

All the elements in this system have been carefully chosen by considering the role and performance needs and value to the overall objectives of the project. The ESP32 offers the computing power and flexibility of the I/O necessary to make shoulder-osteometers operate in real-time, whereas the ESP32-CAM offers precision in the classification of colour at a low-cost. The stepper motor provides accuracy in the conveyor positioning, the servo motor provides speed and accuracy in ejection and the IR sensor provides a reliable and non-contact presence detection. These elements collectively create a high-performance, integrated system that supports the goals of accuracy, throughput, affordability, and scalability.

3.3.2 Software requirements

Developing the embedded firmware and where relevant, training the machine learning model needs certain software tools to promote the correct operation of the prototype. Among the existing various tools that can be used to perform these tasks, the following tools have been chosen due to their appropriateness, availability and their suitability to the goals of the project.

3.3.2.1 Arduino IDE

Arduino IDE was chosen because of its simplicity, functionality, and accessible features, for the ESP32 and ESP32-CAM. The intuitive interface hides the implementation of the low-level configurations of the hardware, cuts down on the learning curve and lets developers concentrate their efforts on the higher-level aspects of device operation. Additionally, this selection was supported by a large library of proven libraries such as ESP32Servo, AccelStepper and HardwareSerial, allowing for precise servo/stepper control and serial communication, which helps to greatly shorten development time, minimize errors and ensure real-time timing accuracy when

communicating with the hardware. It is also open source, cross platform and is free for everyone, making it a philosophy that is in line with the projects vision of providing an affordable solution. testing.



Figure 3:6 Arduino IDE workspace

3.3.2.2 Edge impulse

Edge Impulse was selected as the machine learning development platform because it is specifically designed for TinyML applications on resource-constrained embedded devices such as the ESP32CAM. The platform provides an integrated workflow for data collection, image labelling, model training, optimisation and deployment, reducing development complexity and time. Unlike general-purpose machine learning platforms such as TensorFlow or PyTorch, which often require extensive coding and additional optimisation for embedded deployment, Edge Impulse automatically generates memory-efficient models and Arduino-compatible libraries that can be deployed directly on the ESP32-CAM. Furthermore, its EON optimisation technology reduces memory usage and inference time, making it well suited for real-time pepper colour classification on low-cost hardware.

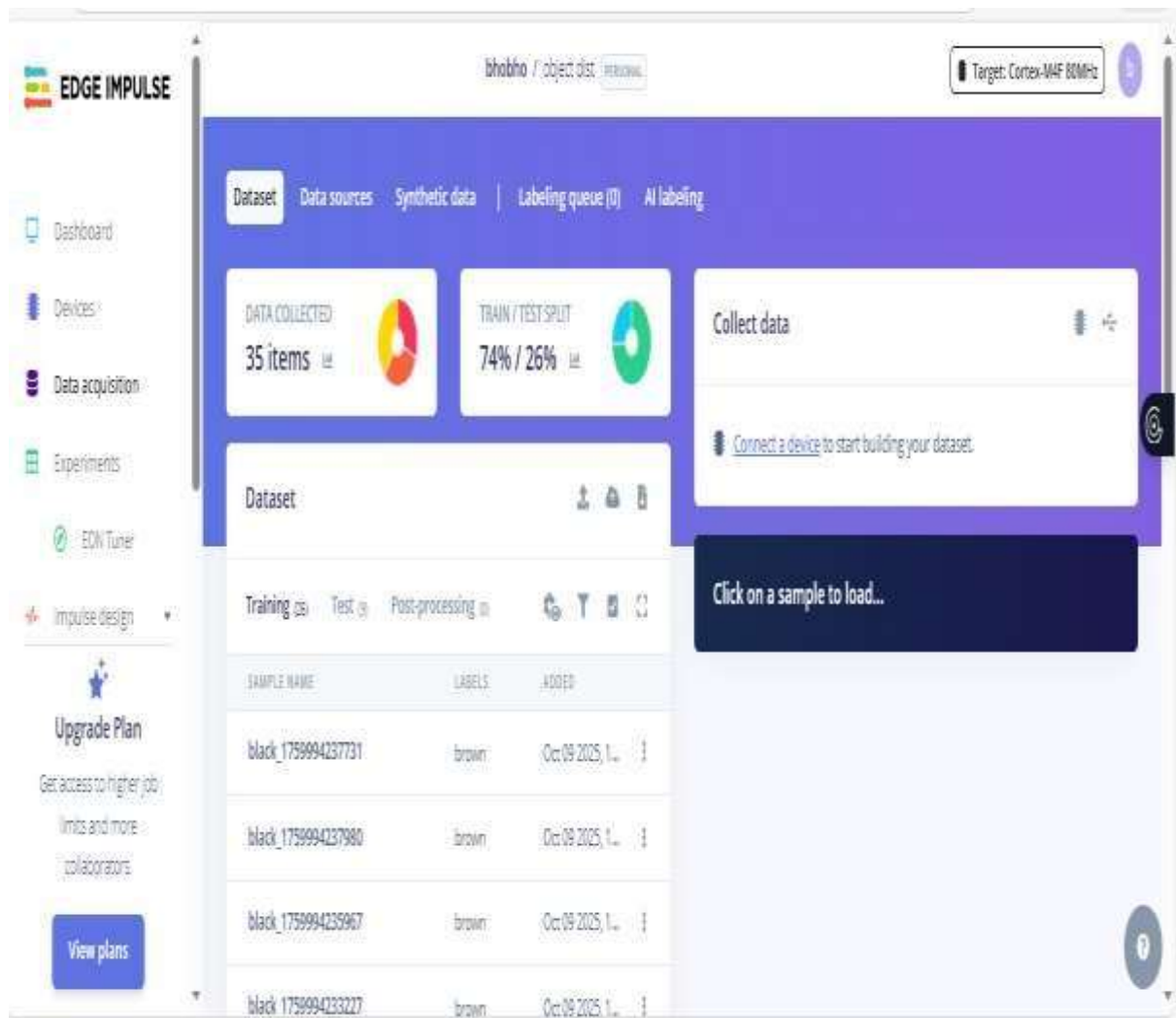


Figure 3:7 edge impulse working environment

3.4 System Architecture

The automated pepper sorting prototype system architecture is in the form of a distributed control system in which several sub systems are coordinated by the central microcontroller. The architecture provides a high-performance communication between the vision module, sensors and actuators to facilitate real time decision of sorting and a high precision of mechanical performance.

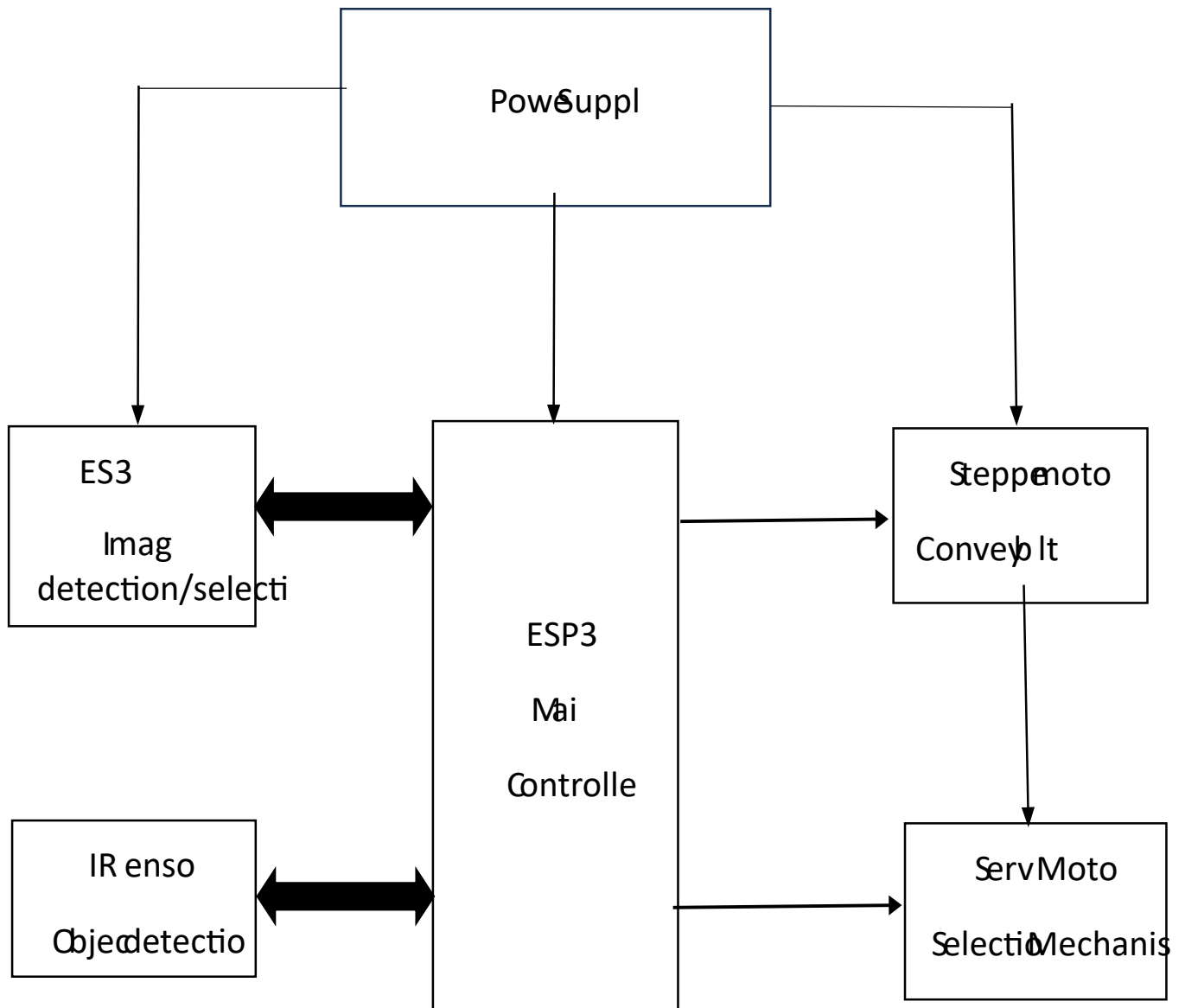


Figure 3:8 Overall Block diagram of the system

3.4.1 Subsystem units

3.4.1.1 Power supply

5 V/ 2A mains adapter was selected as a power source in this system. The voltage is appropriate to the ESP32 and ESP32-CAM modules both of which take in 5V on the VIN or 5V pin and the onboard regulators safely decrease the voltage to 3.3V to run the modules internally. The IR sensor, 5 V logic levels, and the servo motor, which can work well within its 4.8 V to 6 V range, also use the 5V supply. A 2A current rating is enough to power all the components, and the current rating

(2A) has a safe margin to the total requirement of 990 mA, calculated. Reverse polarity protection, overcurrent protection, and decoupling capacitors are included to make sure that the functioning will be reliable. The design of the power supply has removed undervoltage resets and overvoltage damages, allowing the sorting to be stable

3.4.1.2 Conveyor belt

The conveyor belt is precisely moved by a stepper motor driven by a driver and an ESP32 microcontroller. The reason for selecting the stepper motor was that it is capable of providing accurate positional control, holding torque and adjustable speed. The conveyor runs at different speeds and the time is given for image processing, one for each pepper. This application results in a design that guarantees synchronisation of pepper transport, pepper colour classification and pepper servo ejection.

3.4.1.3 Selection mechanism

The mechanism of selection physically diverts each pepper to the correct colour coded bin depending upon the classification outcome of the ESP32 CAM. The selection of a servo motor was due to its ability to provide angular control, adequate torque and quick response time. The servo is equipped with a lightweight paddle or push arm that is connected to the output shaft. After a colour classification (green, yellow, or red) is sent to the ESP32 microcontroller, a PWM signal is produced that causes a certain servo motor to move. The servo immediately turns to the required location, ejecting the pepper out of the conveyor belt and into the required container.

3.4.1.4 Image detection and processing

The image detection and selection unit will capture an image of every pepper and identify the colour class of that pepper. The choice of an ESP32-CAM was based on the fact that the module combines a 2-megapixel OV2640 camera sensor with an onboard ESP32 processor, thus removing the use of an external microcontroller to process images and make inferences. Upon receipt of a trigger signal in the IR sensor, the ESP32-CAM takes a snap shot of the pepper on the conveyor belt. The esp32cam runs the inferencing AI model and deduce the color of the present pepper. After a colour choice has been made, the outcome is transmitted through the UART serial to the main ESP32 microcontroller, which then acts by instructing the servo motor to spurt the pepper into the correct bin.

3.4.1.5 Main controller

The primary controller is the processing and coordination centre of the whole automated pepper sorting system. An ESP32 microcontroller was chosen to fill this role since it provides dual core

processing, on-board Wi-Fi and Bluetooth, multiple input/output pins and low power consumption. There are five main functions of the ESP32. It can detect the pepper, which triggers the IR sensor to send a signal to the first one. Second, it interacts with the ESP32-CAM module through UART serial communication to ask to take an image and get back the result of the colour classification (green, yellow or red). Third, it produces step and direction pulses to regulate the stepper motor driver, which drives the conveyor belt at the desired speed. Fourth, it measures the location of every pepper by counting step pulses, making sure that the movement of the belts and ejection timing are properly in step. Fifth, it delivers a PWM signal to the servo motor to trigger the ejection mechanism at the exact point when the pepper is at the ejection point. The coordination between the sensing, processing, and actuation subsystems is realised through reliable and real-time coordination by centralising all the control functions in a single ESP32 microcontroller.

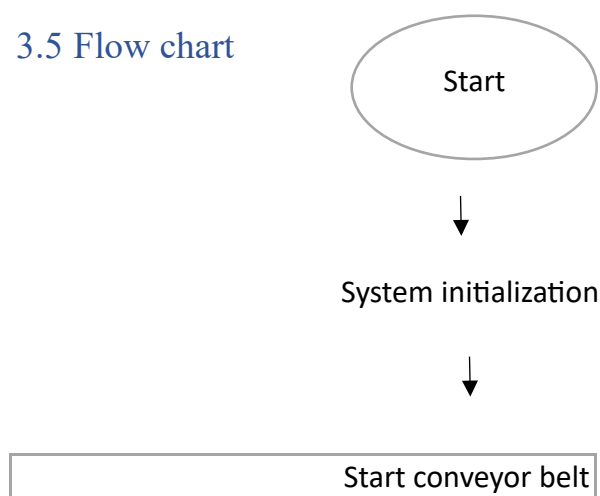
3.4.2 Signal flow description

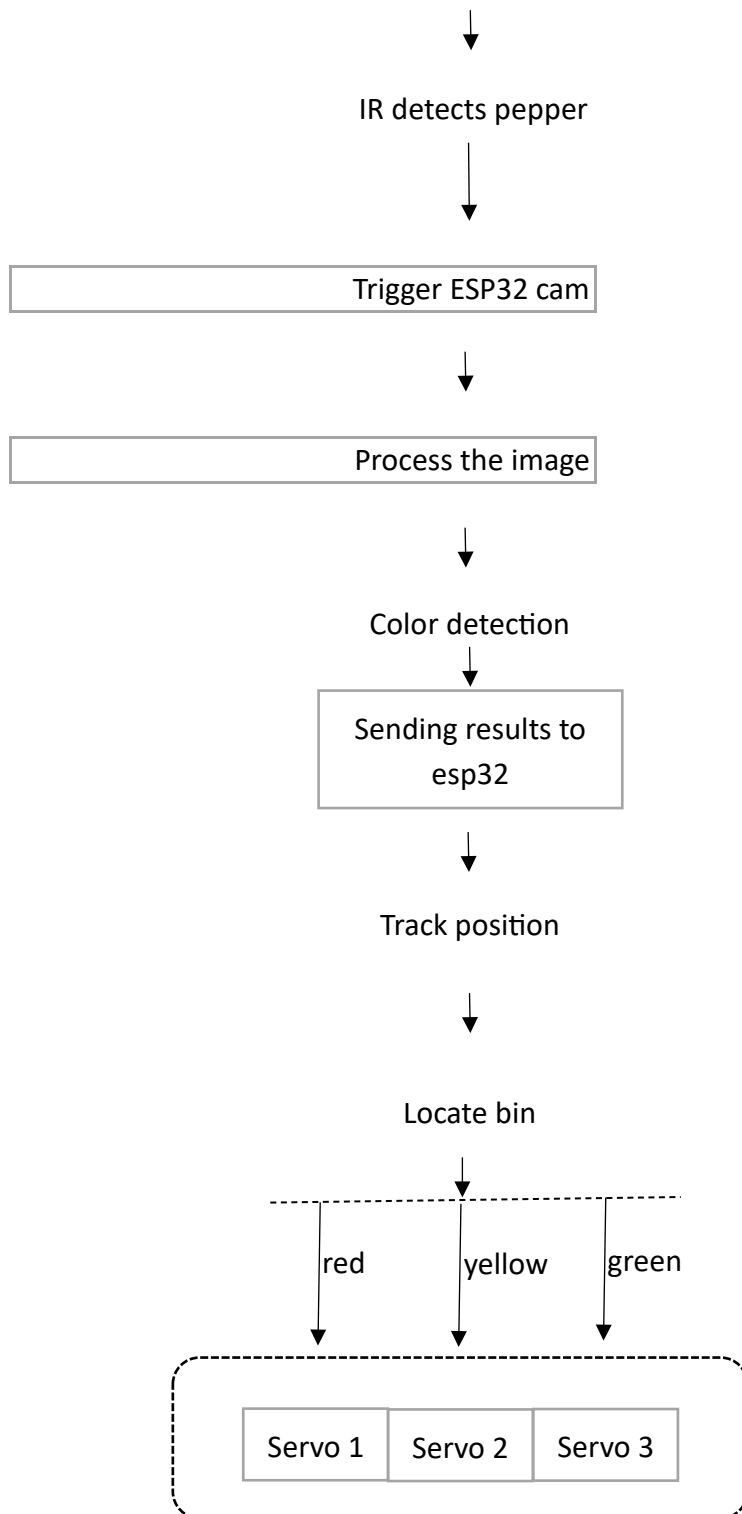
The sorting process begins when a pepper passes in front of the IR sensor. Upon detection, the sensor sends a signal to the main ESP32 microcontroller, which then transmits a command to the ESP32-CAM module to capture an image of the pepper. The ESP32-CAM acquires the image using its OV2640 camera sensor and performs image preprocessing and colour classification using a trained Edge Impulse TinyML model. The resulting classification, corresponding to the green, yellow, or red colour class, is transmitted back to the main ESP32 via UART communication.

While image acquisition and classification are being performed, the conveyor belt continues to operate under the control of the main ESP32. The controller continuously generates step pulses for the stepper motor driver and simultaneously tracks the pepper's position by counting the number of pulses issued. Since each motor step corresponds to a known linear displacement of the conveyor belt, the controller can determine the location of the pepper throughout its movement.

Once the classification result has been received and the pepper reaches the predefined ejection position, the main ESP32 generates a PWM control signal for the servo motor associated with the detected colour class. The selected servo actuates the sorting mechanism, diverting the pepper into the corresponding collection bin. After ejection, the servo returns to its neutral position, and the system resumes monitoring for the next pepper, thereby enabling continuous and automated sorting operation.

3.5 Flow chart





3.6 System design and implementation

This section presents the hardware and software implementation of the proposed automated bell pepper colour sorting system. It describes the electrical hardware, embedded software, communication architecture, algorithms and mathematical models used to realise the prototype.

3.6.1 Circuit diagrams

Figure 12 shows the complete circuit diagram of the automated pepper sorting system. The schematic diagram was created using the Fritzing simulation desktop application. The diagram illustrates all electrical connections between the ESP32 main controller, ESP32-CAM module, IR sensor, stepper motor driver, stepper motor, and servo motor.

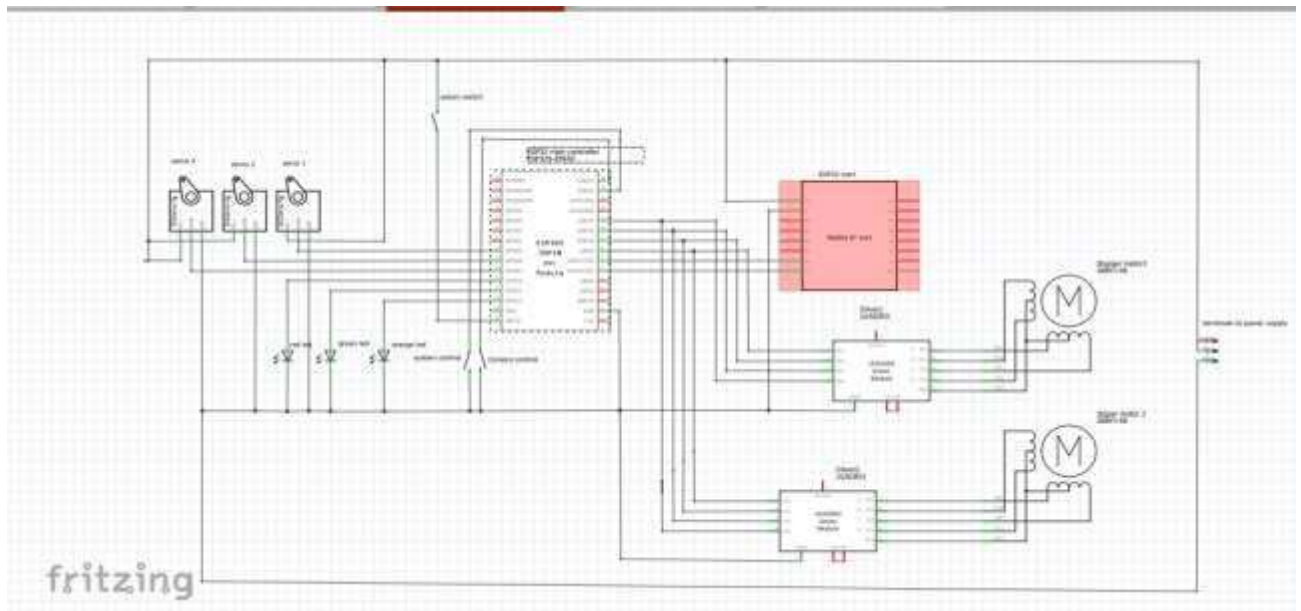


Figure 3:9 Complete circuit diagram of the automated pepper sorting system

3.6.2 Hardware design

This section describes the hardware implementation of the proposed system. It explains the electrical interfacing of each subsystem and the functional role of every hardware component within the complete sorting system.

3.6.2.1 Esp32cam to esp32 interfacing

The ESP32-CAM module communicates with the main ESP32 microcontroller using UART (Universal Asynchronous Receiver-Transmitter) serial communication. UART communication was selected because it provides reliable full-duplex serial communication while requiring minimal hardware resources and processor overhead.

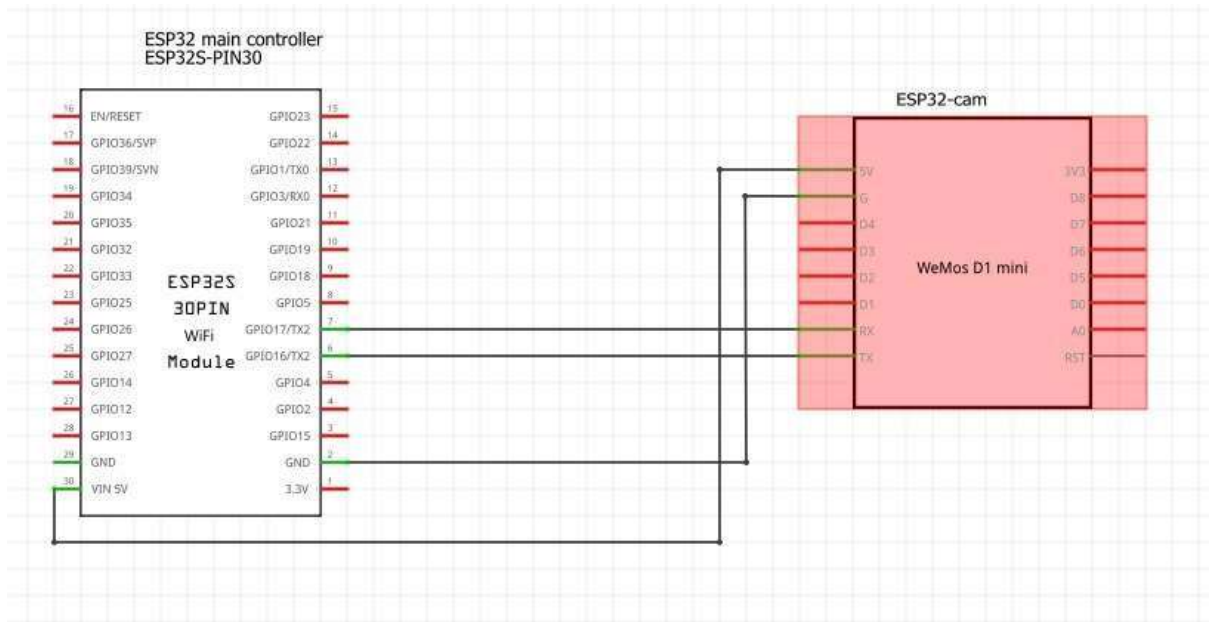


Figure 3:10 schematic diagram esp32cam interfacing

ESP32 Main Pin	ESP32-CAM Pin	Function
GPIO 16 (TX2)	RX	Send command to camera
GPIO 17 (RX2)	TX	Receive classification result
5V	5V	Power (5V DC)
GND	GND	Common ground

Table 3:1 esp32cam interfacing pin allocation table

3.6.2.2 Servo motors to esp32 interfacing

The use of independent servo actuators eliminates the need for a complex mechanical linkage, improves reliability and allows each sorting gate to operate independently. Each servo is dedicated to ejecting one specific variety of pepper into its corresponding bin Servo 1 (green bin), Servo 2 (yellow bin), and Servo 3 (red bin). This design improves system reliability by distributing the

mechanical load across three actuators, reducing the risk of a single servo overheating or being overworked during continuous operation. Figure 14 shows the schematic diagram of the connections.

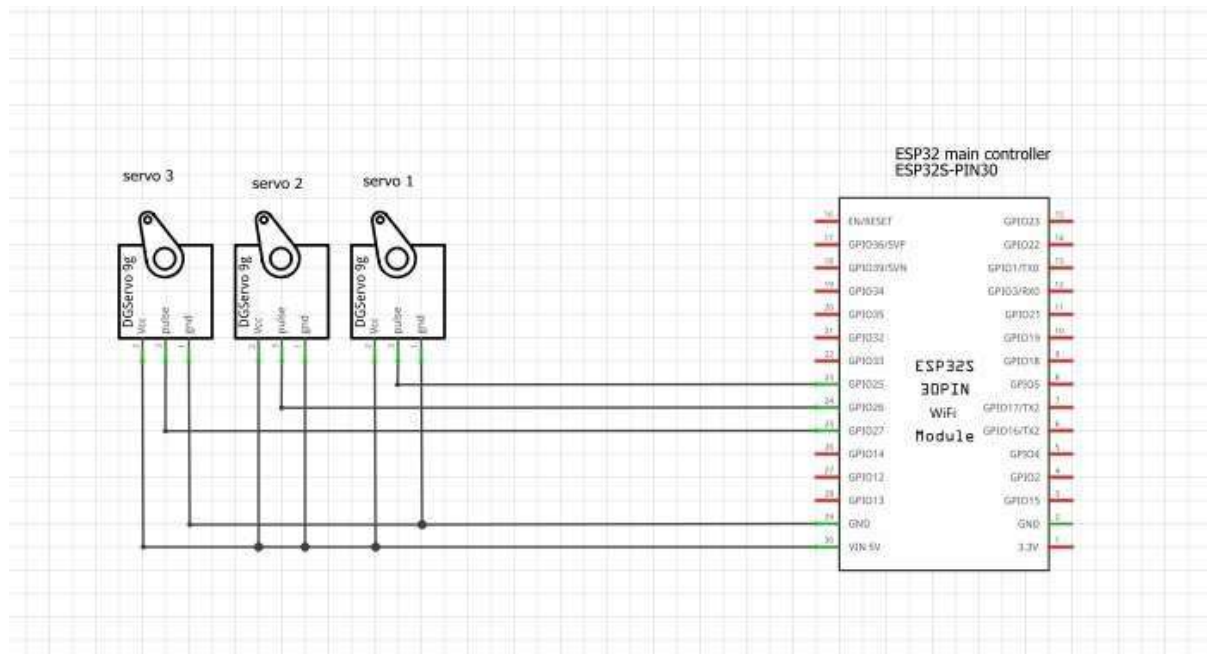


Figure 3:11 schematic diagram of servo motor interfacing

Servo Motor (MG996R)	Main ESP32	Function
(VCC) – All servos	5V Power Supply (shared)	Power (5V)
(GND) – All servos	GND (common)	Ground
Signal– Servo 1 (Green)	GPIO 25	PWM control for green ejection
Signal – Servo 2 (Yellow)	GPIO 26	PWM control for yellow ejection
Signal – Servo 3 (Red)	GPIO 27	PWM control for red ejection

Table 3:2 servo motor pin allocation

3.6.2.3 Stepper motor to esp32 interfacing

The conveyor is driven by two synchronised 28BYJ-48 stepper motors operating in parallel. Sharing the mechanical load reduces the torque requirement on each motor, improves conveyor stability and minimises the risk of missed steps during continuous operation.

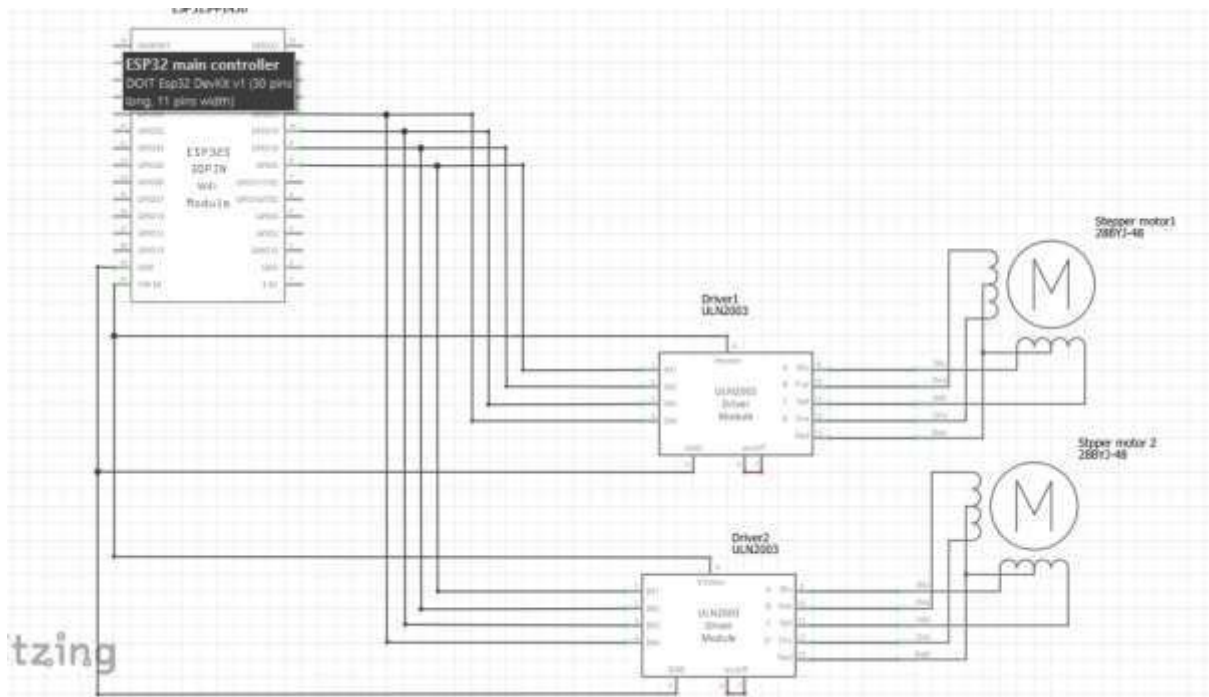


Figure 3:12 stepper motor interfacing

ESP32S3 Pin	Driver 1 (INx)	Driver 2 (INx)	Function
GPIO 5	IN1	IN1	Motor coil control 1
GPIO 18	IN2	IN2	Motor coil control 2
GPIO 19	IN3	IN3	Motor coil control 3
GPIO 21	IN4	IN4	Motor coil control 4
VIN 5V	VCC	VCC	Power supply (5V)
GND	GND	GND	Ground

Table 3:3 stepper motor pin connections

3.6.2.4 Infrared sensor to ESP32 Interfacing

The FC-51 obstacle avoidance sensor is used to detect the presence of a pepper as it approaches the camera station. This infrared (IR) sensor provides a digital output signal that changes state when an object is detected, triggering the image capture process. Figure 16 shows the schematic diagram of the connections.

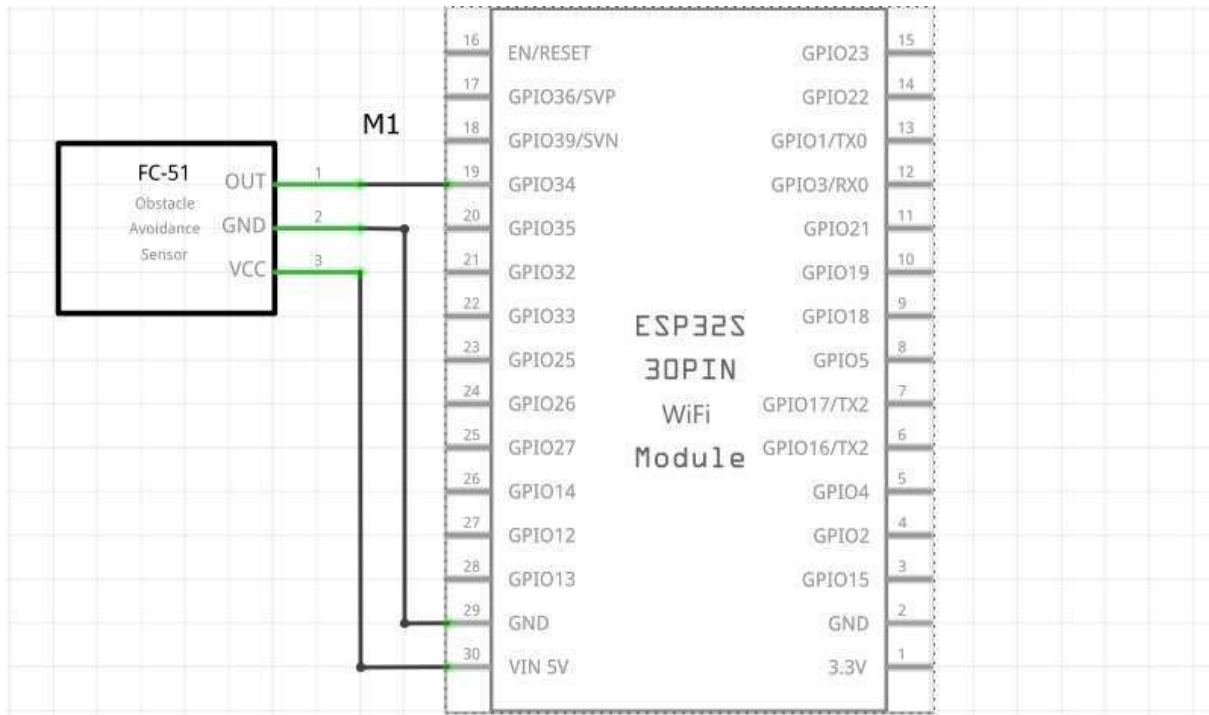


Figure 3:13 infrared sensor interfacing

FC-51 Sensor Pin	ESP32S3 Pin	Function
VCC	VIN 5V	Power supply (5V)
GND	GND	Ground
OUT	GPIO 34	Digital input signal

Table 3:4 IR connections pins

3.6.3 Software design

The software design encompasses all code written for the ESP32 main controller and the ESP32CAM module. The software is responsible for object detection, image capture, colour classification, conveyor control, position tracking, and servo actuation. The development was carried out using the Arduino IDE with the ESP32 board package. Model training was performed using the Edge Impulse platform, where image data were labelled, trained and exported as an Arduino-compatible C++ library.

3.6.3.1 System architecture

The software follows a modular architecture with two independent communicating firmware programs, one running on the main ESP32 and another running on the ESP32-CAM. Communication between the two is achieved via UART serial protocol at 115200 baud. A modular software architecture was adopted to simplify debugging, improve maintainability and allow independent execution of the conveyor control and image processing tasks.

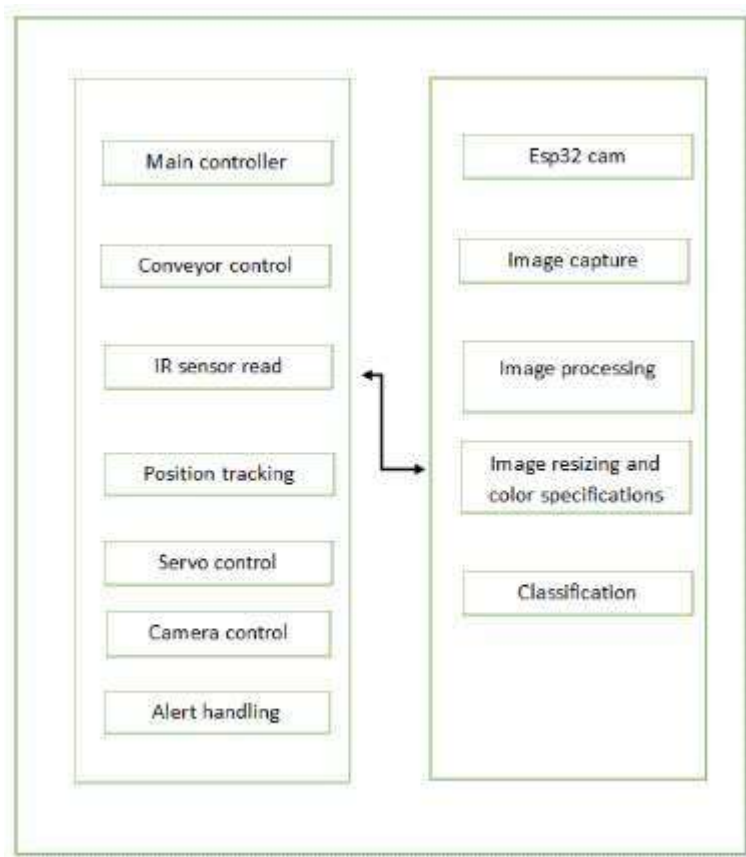


Figure 3:14 system architecture

3.6.3.2 Main esp32 microcontroller code

The main ESP32 executes the central control logic. The main controller coordinates all hardware subsystems and ensures deterministic operation through sequential execution of sensing, communication, position tracking and sorting operations.

```

File Edit Sketch Tools Help
ψ AI Thinker ESP32-CAM Upload

sketch_apr25a.ino
1 // Pin Definitions for Main ESP32
2 #define IR_SENSOR_PIN 34 // Digital input from FC-51 IR sensor
3
4 // Stepper Motor Pins (shared between both motors in parallel)
5 #define STEP_PIN 5 // Step pulse to both ULN2003 drivers
6 #define DIR_PIN 18 // Direction control for both motors
7
8 // Servo Motor Pins (three servos - one per colour)
9 #define SERVO_GREEN_PIN 25 // Green bin ejection
10 #define SERVO_YELLOW_PIN 26 // Yellow bin ejection
11 #define SERVO_RED_PIN 27 // Red bin ejection
12
13 // UART Pins for ESP32-CAM Communication
14 #define CAM_RX_PIN 16 // Receive from ESP32-CAM (TX2)
15 #define CAM_TX_PIN 17 // Transmit to ESP32-CAM (RX2)

```

Figure 3:15 pin definition

- **Stepper motor control functions**

The stepper motor is controlled using a simple step and direction interface. Since both motors are connected in parallel, a single set of control signals drives both ULN2003 drivers simultaneously.

```

sketch_apr25a.ino
9
10 // Global variables for position tracking
11 unsigned long stepCounter = 0;
12 int stepsToEjection = 12750; // Calibrated value (260mm * 51 steps/mm)
13
14 void setupStepper() {
15   pinMode(STEP_PIN, OUTPUT);
16   pinMode(DIR_PIN, OUTPUT);
17   // Set initial direction (forward)
18   digitalWrite(DIR_PIN, HIGH);
19 }
20
21 // Function to send one step pulse (moves both motors simultaneously)
22 void stepMotor() {
23   digitalWrite(STEP_PIN, HIGH);
24   delayMicroseconds(5); // Minimum pulse width (4µs typical), 5µs for safety
25   digitalWrite(STEP_PIN, LOW);
26   delayMicroseconds(5);
27   stepCounter++; // Increment position counter
28 }
29
30 // Function to move conveyor belt a specific number of steps
31 void moveConveyor(int steps, bool forward) {
32   digitalWrite(DIR_PIN, forward ? HIGH : LOW);
33   for (int i = 0; i < steps; i++) {
34     stepMotor();
35     delayMicroseconds(stepDelay); // Controls speed (x120 mm/s)
36   }
37 }

```

Figure 3:16 stepper motor control code

- **Servo motor control functions**

Three servos are used, each dedicated to one pepper colour. Only the servo corresponding to the classified colour is activated.

```

sketch_apr25a.ino
1  #include <ESP32Servo.h>
2
3  Servo servoGreen;
4  Servo servoYellow;
5  Servo servoRed;
6
7  // Servo positions
8  #define SERVO_RETRACTED 0
9  #define SERVO_EXTEND 90
10
11 void setupServos() {
12     servoGreen.attach(SERVO_GREEN_PIN);
13     servoYellow.attach(SERVO_YELLOW_PIN);
14     servoRed.attach(SERVO_RED_PIN);
15
16     // Set all servos to retracted position
17     servoGreen.write(SERVO_RETRACTED);
18     servoYellow.write(SERVO_RETRACTED);
19     servoRed.write(SERVO_RETRACTED);
20 }
21
22 // Eject pepper based on colour classification
23 void ejectPepper(String colour) {
24     if (colour == "GREEN") {
25         servoGreen.write(SERVO_EXTEND);
26         delay(200); // Allow time for ejection
27         servoGreen.write(SERVO_RETRACTED);
28     }
29     else if (colour == "YELLOW") {
30         servoYellow.write(SERVO_EXTEND);
31         delay(200);

```

Figure 3:17 servo control function

- **UART communication with ESP32cam**

The main ESP32 communicates with the ESP32-CAM using Serial2 hardware UART.

```

main_controller_code.ino
1  // Main ESP32 - Receives inference results from ESP32-CAM over UART
2  // Uses Serial2 (pins 16=RX, 17=TX) at 115200 baud
3
4  void setup() {
5      Serial.begin(115200); // USB debug / user input
6      Serial2.begin(115200, SERIAL_8N1, 16, 17); // Communication with ESP32-CAM
7
8      Serial.println("Main ESP32 ready.");
9      Serial.println("Flash commands: '1' = ON, '0' = OFF");
10 }
11
12 void loop() {
13     // Forward user commands to ESP32-CAM
14     if (Serial.available()) {
15         char cmd = Serial.read();
16         Serial2.write(cmd);
17         Serial.print("Sent command: ");
18         Serial.println(cmd);
19     }
20 }

```

Figure 3:18 UART communication

- **Main control loop**

The main control loop integrates all subsystems into a cohesive sorting sequence.

```

void handleDetection(String label) {
  if (!systemEnabled) {
    Serial.println("System disabled ⚠ ignoring detection");
    return;
  }
  if (currentState != STATE_IDLE) {
    Serial.println("System busy ⚠ ignoring detection");
    return;
  }

  if (label == "red" || label == "green") {
    // --- RED / GREEN: conveyor 10s, then eject 10s ---
    Serial.print("--> ");
    Serial.print(label);
    Serial.println(" detected ⚠ starting cycle");

    // 1. Conveyor
    currentState = STATE_CONVEYOR;
    digitalWrite(LED_RED2, HIGH);
    digitalWrite(LED_BLUE1, HIGH);
    Serial.printf("Moving conveyor for %d seconds\n", CONVEYOR_TIME_RED_GREEN / 1000);
    moveConveyor(CONVEYOR_TIME_RED_GREEN);
    digitalWrite(LED_BLUE1, LOW);

    // 2. Eject (10 seconds hold)
    currentState = STATE_EJECTING;
    digitalWrite(LED_BLUE2, HIGH);
    if (label == "red") {

```

Figure 3:19 main controller loop

3.6.3.3 Esp32-cam firmware

The ESP32-CAM module captures images and performs colour classification. The firmware waits for a "CAPTURE" command from the main ESP32, captures an image, processes it, and returns the classification result. The image processing algorithm executes entirely on the ESP32-CAM, reducing the computational burden on the main controller.

```

esp32_camera.ino
1 #include "esp_camera.h"
2
3 // Camera pin configuration for ESP32-CAM
4 static Camera_config_t camera_config = {
5     .pin_pwdn = 32,
6     .pin_reset = -1,
7     .pin_xclk = 0,
8     .pin_vsync_sda = 20,
9     .pin_vsync_scl = 22,
10    .pin_d7 = 15,
11    .pin_d6 = 14,
12    .pin_d5 = 16,
13    .pin_d4 = 17,
14    .pin_d3 = 18,
15    .pin_d2 = 19,
16    .pin_d1 = 10,
17    .pin_d0 = 9,
18    .pin_vsync = 26,
19    .pin_href = 23,
20    .pin_pclk = 22,
21    .xclk_freq_hz = 20000000,
22    .ledc_timer = LEDC_TIMER_0,
23    .ledc_channel = LEDC_CHANNEL_0,
24    .pixel_format = PIXFORMAT_RGB565, // 30x240 pixels
25    .frame_size = FRAMESIZE_QVGA,
26    .jpg_quality = 12,
27    .fb_count = 1
28 };
29
30 void setupCamera() {
31     esp_err_t err = esp_camera_init(&camera_config);

```

Figure 3:20 camera initialisation

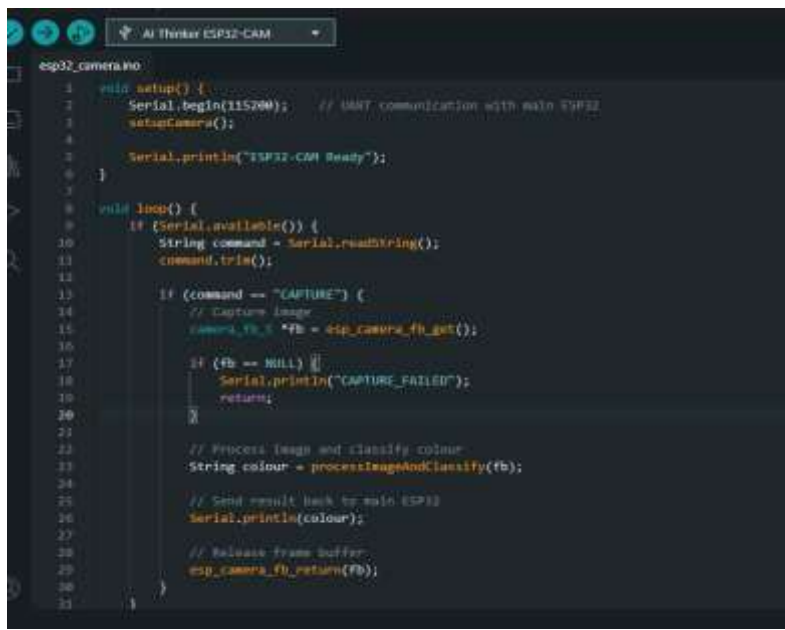


Figure 3:21 esp32cam loop

3.6.4 Error Handling and Safety Mechanisms

To improve the reliability and safety of the automated pepper sorting system, several error handling and emergency control mechanisms were incorporated into both the hardware and software design. An emergency stop push button was installed to allow immediate shutdown of the sorting process in the event of a malfunction, obstruction, or other unsafe operating condition. When activated, the emergency stop button interrupts normal system operation by halting conveyor movement and disabling the ejection mechanism. The system remains in this state until the fault is addressed and the machine is manually restarted.

In addition, the software continuously monitors the operational status of the ESP32-CAM module during system startup. Before normal operation begins, the firmware attempts to initialise the camera and verify successful communication with the image sensor. If camera initialisation fails, the system prevents the sorting process from starting and generates an alert through the serial monitor and status indicator. This mechanism ensures that peppers are not processed without a functioning vision subsystem, thereby preventing incorrect classification and sorting.

3.6.5 Algorithms

This section presents the complete design process for developing the fruit detection and colour classification system, from initial data acquisition through to the final deployed model on the

ESP32-CAM. The methodology follows a systematic machine learning pipeline combined with classical computer vision techniques to achieve robust, real-time pepper sorting.

3.6.5.1 Data collection

All data collection was performed on the experimental sorting system under natural lighting using the identical ESP32-CAM hardware that would perform real-time classification. This unified approach preserved colour fidelity, lighting consistency, and background consistency, effectively eliminating the primary variables that cause colour-blind errors in vision-based sorting systems. Using identical hardware during both training and deployment minimised domain shift and improved classification consistency.

- **ESP32-CAM Data Collection Web Server Design**

An ESP32-CAM module was configured to host a lightweight web server that served as the data collection interface. This approach eliminated the need for external cameras or manual image transfer, ensuring that all training data was captured using the exact same hardware that would perform the final classification.



```
imagecollection.ino
1: #define WIFI_SSID "twinkle"
2: #define WIFI_PASS "bhobho1234"
3: #define HOSTNAME "esp32cam"
4:
5: #include <eloquent_esp32cam.h>
6: #include <eloquent_esp32cam/extra/esp32/wifi/sta.h>
7: #include <eloquent_esp32cam/viz/image_collection.h>
8:
9: using eloq::camera;
10: using eloq::wifi;
11: using eloq::viz::collectionServer;
12:
13: void setup() {
14:   delay(3000);
15:   Serial.begin(115200);
16:   Serial.println("__IMAGE COLLECTION SERVER__");
17:
18:   camera.pinout.aithinker();
19:   camera.brownout.disable();
20:
21:   camera.resolution.face();
22:
23:   while (!camera.begin().isOk())
24:     Serial.println(camera.exception.toString());
25:
26:   while (!wifi.connect().isOk())
27:     Serial.println(wifi.exception.toString());
28:
29:   while (!collectionServer.begin().isOk())
30:     Serial.println(collectionServer.exception.toString());
31:
32:   Serial.println("Camera OK");
33:   Serial.println("Wifi OK");
34:   Serial.println("Image Collection Server OK");
35:   Serial.println(collectionServer.address());
36:
37:   pinMode(4, OUTPUT);
38:   digitalWrite(4, HIGH);
39: }
```

Figure 3:22 image collection web code

- **Model training using edge impulse**

The collected data was subsequently used to train an object detection and colour classification model on the Edge Impulse platform. The trained model was exported as an Arduino library and deployed to the ESP32-CAM for real-time inference during sorting operations.

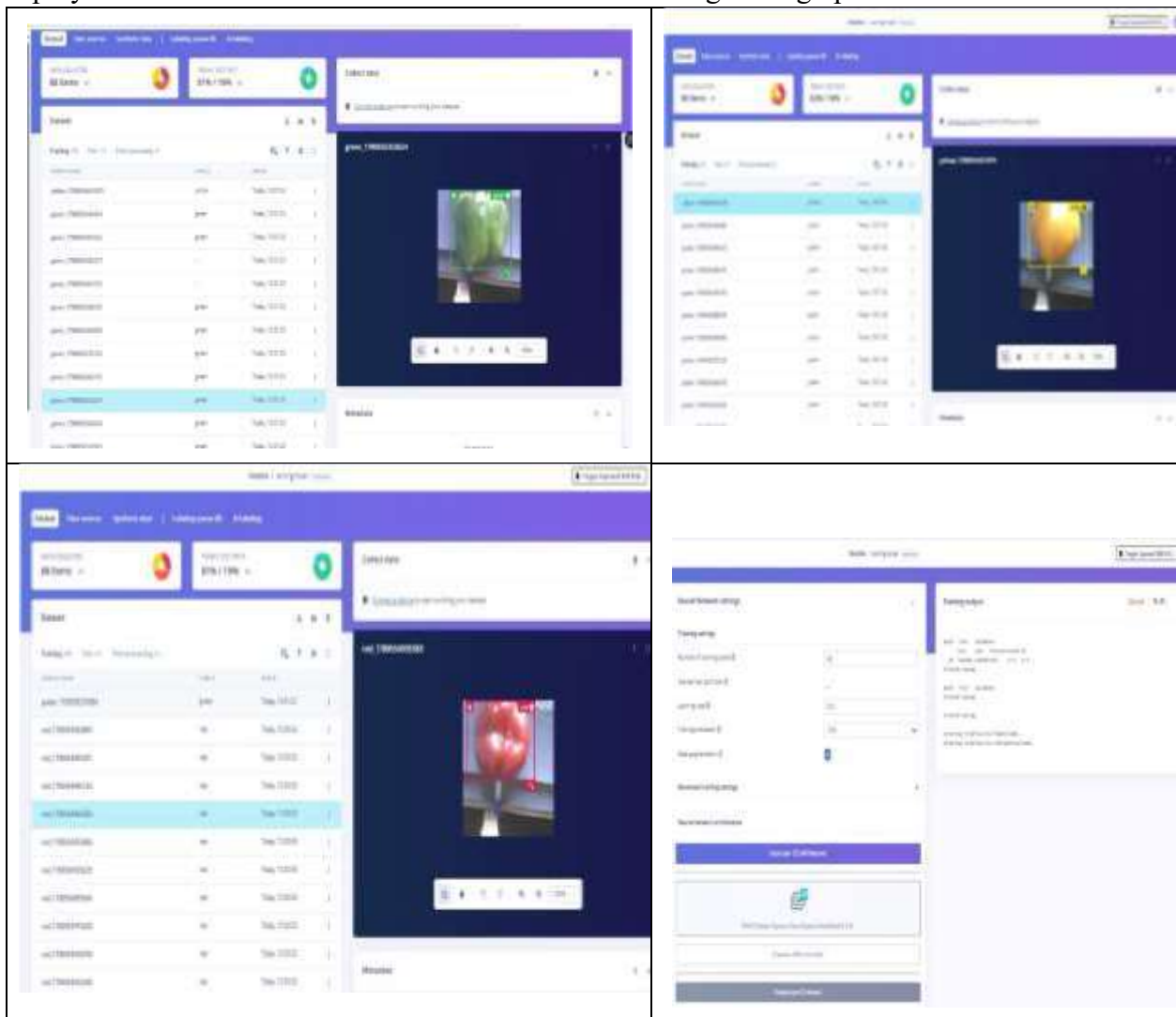


Figure 3:23 model training snippets

3.6.6 Dataset Preparation and Training

A dataset of bell pepper images was collected and used to train the Edge Impulse TinyML model. Four classes were created: Green, Yellow, Red, and Background. Each class contained 150 training images and 38 testing images, as shown in Table 3:5

Class	Training Images	Testing Images
-------	-----------------	----------------

Green	150	38
Yellow	150	38
Red	150	38
Background	150	38
Total	600	152

Table 3:6 training data table

The training images were used to train the model, while the testing images were used to evaluate its performance on unseen data. The background class was included to help the model distinguish peppers from non-pepper objects and empty conveyor scenes.

3.6.7 Mathematical modelling

Mathematical models were used to describe, predict, and optimise the behaviour of key subsystems in the automated pepper sorting system. These models provide a theoretical foundation for calibration, performance prediction, and validation of experimental results. The following subsections present the mathematical formulations for conveyor kinematics, throughput estimation, colour classification, servo control, IR detection, error analysis and power consumption.

3.6.7.1 Conveyor kinematics

The conveyor belt is driven by two 28BYJ-48 stepper motors connected in parallel. The relationship between the number of step pulses and linear belt displacement is fundamental to position tracking.

Step calibration constant

$$S_{PP} = \frac{N_r}{\pi D}$$

where:

- $N_r = 2048$ steps/revolution
- $D = 25\text{mm}$

Substituting:

$$S_{pp} = \frac{2048}{\pi 25} \text{mm}$$

$$S_{pp} = 26.08 \text{ steps/mm}$$

$$S_{pp} \approx 26.1 \text{ steps/mm}$$

Equation 3:1 step calibration

Linear displacement as a function of step count:

$$x(n) = \frac{n}{S_{pp}}$$

and

$$x(n) = \frac{n}{S_{pp}}$$

then:

$$x(n) = \frac{n}{0.153}$$

$$x(n) = 6.54n$$

Therefore,

$$x(n) = 6.54n \text{ mm}$$

Where:

$x(n)$ is belt displacement (mm)

n number of step pulses sent S_{pp} is

steps per millimetre steps/mm

Equation 3:2 linear displacement Belt

linear velocity:

$$v = \frac{f_s}{S_{pp}}$$

then

$$v = \frac{10}{153} \text{ 0.}$$
$$v = 65.4 \text{ mm/s}$$

If

$$f_s = 20 \text{ steps/s}$$

then

$$v = 130.8 \text{ mm/s}$$

Where:

- v belt velocity (mm/s)
- f_s step pulse frequency (Hz)
- S_{PP} is steps per millimetre steps/mm

Equation 3:3 belt linear velocity

3.6.7.2 Ejection modelling Servo

Response Time:

Servo rotation time t_s for an angle θ :

$$t_s = \frac{\theta}{\omega}$$

and the servo speed is:

$$\omega = 300^\circ/s$$

then

$$t_s = \frac{90}{300} t_s = 0.3 \text{ s}$$

Equation 3:4 servo response

3.7 Ethical considerations

This project adheres to the ethical standards of engineering research as prescribed by Bindura University of Science Education (EEE4000 guidelines) and the Engineering Council of Zimbabwe.

3.7.1 Human and Animal Subjects

No human or animal subjects were involved in this study. The system processes only bell peppers (*Capsicum*), which are agricultural produce.

3.7.2 Electrical Safety

All circuits operate at extra-low voltage (5 V DC). No high-voltage (mains) components are present on the prototype. However, the following precautions were implemented:

All exposed conductive parts are enclosed in acrylic casing.

The power supply adapter is certified to IEC 62368-1.

Decoupling capacitors (1000 μ F, 100 nF) suppress voltage spikes from stepper motor switching.

A reverse polarity protection diode (1N4007) is placed in series with the 5 V input.

3.7.3 Mechanical Safety

Moving parts (conveyor belt, stepper motor shafts, servo paddles) are guarded with covers to prevent finger entrapment. Warning labels are affixed near the ejection zone. No sharp edges are present on pepper-contacting surfaces.

3.7.4 Waste Minimisation and Environmental Responsibility

Fresh pepper samples (*Capsicum*) were used solely for system testing and remained suitable for future demonstrations, thereby minimising material waste.

3.7.5 Intellectual Property and Open-Source Compliance

The Arduino IDE, ESP32 board package, ESP32Servo library, and AccelStepper library are used under the GNU Lesser General Public License (LGPL).

Edge Impulse is used under its free academic license.

OpenCV is used under the BSD license.

3.8 Summary

A comprehensive engineering methodology was used to design and build an automated colour-based pepper sorting system. An iterative prototyping approach was justified and implemented. The system architecture was described as a distributed control system with five subsystems, and the signal flow was detailed step-by-step. A flow chart summarised the control logic of a single sorting cycle. Circuit diagrams and pin allocation tables were provided. Embedded firmware structure and the colour classification algorithm were explained. Mathematical models for conveyor kinematics, throughput, and servo response were derived. Finally, ethical considerations covering electrical safety, mechanical safety, waste minimisation, data integrity and intellectual property were addressed. This methodology provides a reproducible framework for the construction and testing of the prototype. The following chapter presents the experimental results obtained from this system.

4 Chapter 4: EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Introduction

This chapter presents the evaluation and analysis of the developed automated colour-based pepper sorting prototype. The system performance was assessed using three key criteria, sorting accuracy,

throughput and operational reliability. Sorting accuracy was evaluated using precision, recall, and F1-score metrics for each pepper colour class. Throughput analysis considered the sorting rate.

System testing was conducted using a representative sample of green, yellow, and red peppers under different conditions. Multiple sorting trials were performed to assess classification performance, conveyor operation and ejection accuracy. The evaluation criteria were defined as follows, classification accuracy was measured as the proportion of correctly classified peppers relative to the total number tested, precision, recall and F1-score were used to evaluate class-specific performance throughput was measured as the number of peppers successfully sorted per unit time, and operational reliability was assessed by observing system performance under different lighting conditions. The collected experimental data were analysed and compared against the design objectives established in Chapter 1.

4.2 System implementation results

The complete prototype was successfully assembled according to the design specifications outlined in Chapter 3. Figure below shows the fully integrated system comprising the conveyor belt driven by a stepper motor, the ESP32-CAM vision module positioned above the inspection station, the IR proximity sensor for pepper detection, and the servo-actuated ejection flap mechanism with three collection bins.



Figure 4:1 pepper sorter

4.2.1 Edge Impulse Model Performance

The trained Edge Impulse TinyML model was evaluated using the validation dataset generated during model training. The model achieved a precision, recall and F1-score of 100%, indicating perfect classification performance on the validation dataset. The confusion matrix showed no misclassification between the green, yellow and red pepper classes. These results demonstrate the effectiveness of the trained model for colour-based pepper classification.

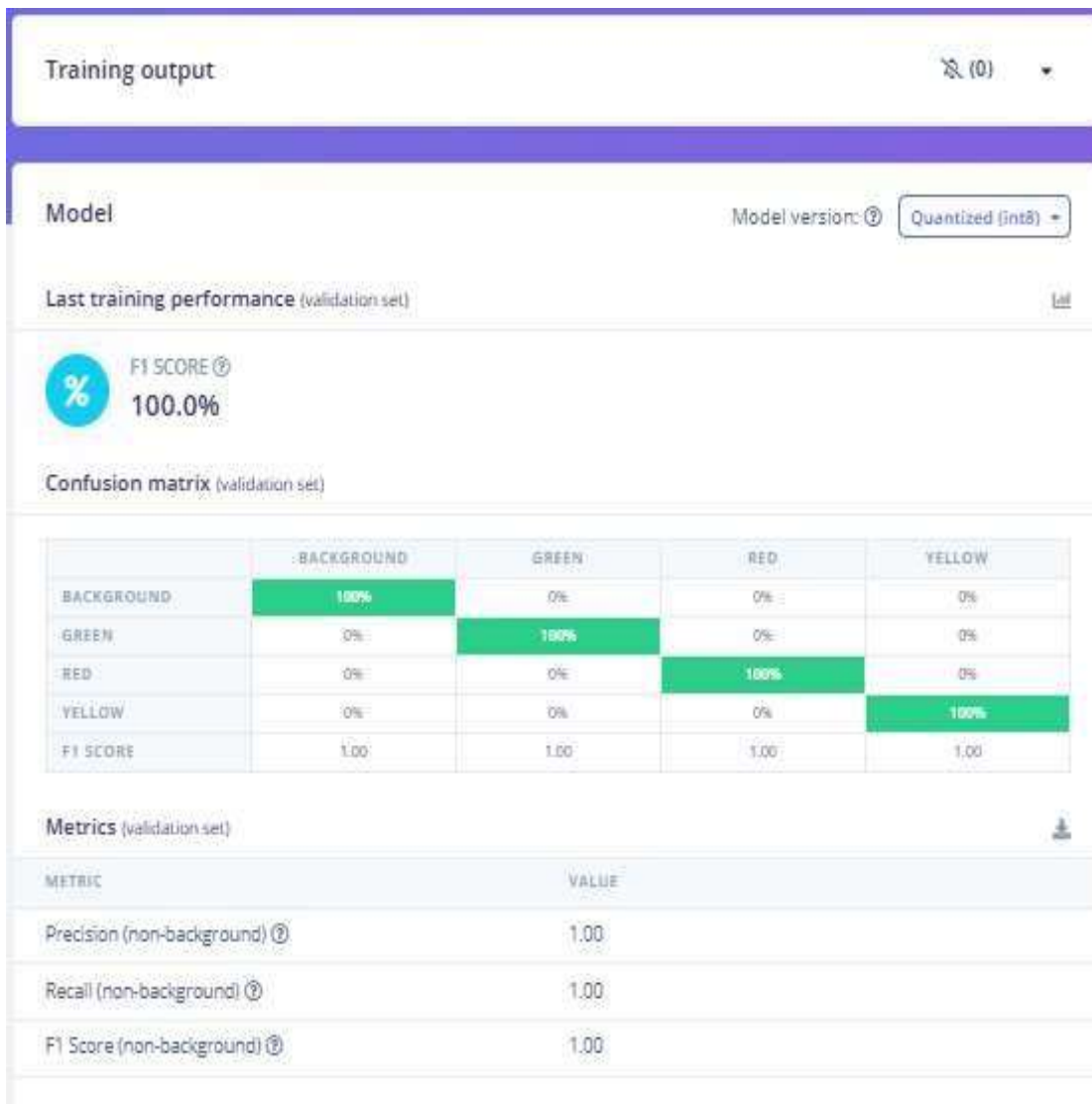


Figure 4:2 Edge Impulse validation results showing confusion matrix, precision, recall and F1score for the trained bell pepper classification model.

4.3 Data presentation

4.3.1 Sorting Accuracy Test

The primary performance metric was sorting accuracy, defined as the percentage of peppers correctly classified and diverted into the correct collection bin.

Metric	Value
Precision	100%

Recall	100%
F1 Score	100%
Green Classification Accuracy	100%
Yellow Classification Accuracy	100%
Red Classification Accuracy	100%

Table 4:1 Edge Impulse Validation Results

4.4 Throughput Performance

Throughput was evaluated by measuring the number of peppers successfully sorted within a fixed time interval. Three experimental trials were conducted with the conveyor operating at its maximum stable speed. During each trial, the time required to sort a known number of peppers was recorded and the throughput was calculated using:

$$T = \frac{N}{t}$$

where T is the throughput (peppers/s), N is the number of peppers sorted, and t is the elapsed time (s).

Equation 4:1 throughput calculation equation

Trial	Peppers Sorted	Time (s)	Throughput (peppers/s)
1	20	40	0.50
2	20	39	0.51
3	20	41	0.49
Average	20	40	0.50

Table 4:2 throughput analysis

4.5 Servo Motor Response Time

The servo motor's response time was measured from the instant the classification result was received to the completion of the ejection movement.

Ejection Angle	Command to Start (ms)	Movement Duration (ms)	Total Response (ms)
0° → 90°	8	120	128
90° → 180°	8	118	126
180° → 0°	8	122	130
Average	8	120	128

Table 4:3 servo motor response time

The average total response time was sufficiently fast to divert peppers accurately at the conveyor speed.

4.6 Lighting Condition Sensitivity

To evaluate robustness, the system was tested under three different lighting conditions: bright, moderate, and dim.

Lighting	Colour	Samples Tested	Correct
Bright	Green	30	30
Bright	Yellow	35	34
Bright	Red	35	35
Moderate	Green	30	29
Moderate	Yellow	35	33
Moderate	Red	35	34
Dim	Green	30	27
Dim	Yellow	35	30
Dim	Red	35	32

Table 4:4 Accuracy Under Different Lighting Conditions

Performance degraded significantly under dim lighting, confirming the importance of controlled illumination for the Edge Impulse-based image classification system.

4.7 Objective achievements

Objective 1: To construct a conveyor system that presents fruits individually to the vision system.

The conveyor system, driven by a 28BYJ-48 stepper motor with a ULN2003 driver, successfully transported peppers individually to the inspection station. The positioning repeatability confirmed that the conveyor could consistently position peppers with an error which is negligible, well within the required tolerance.

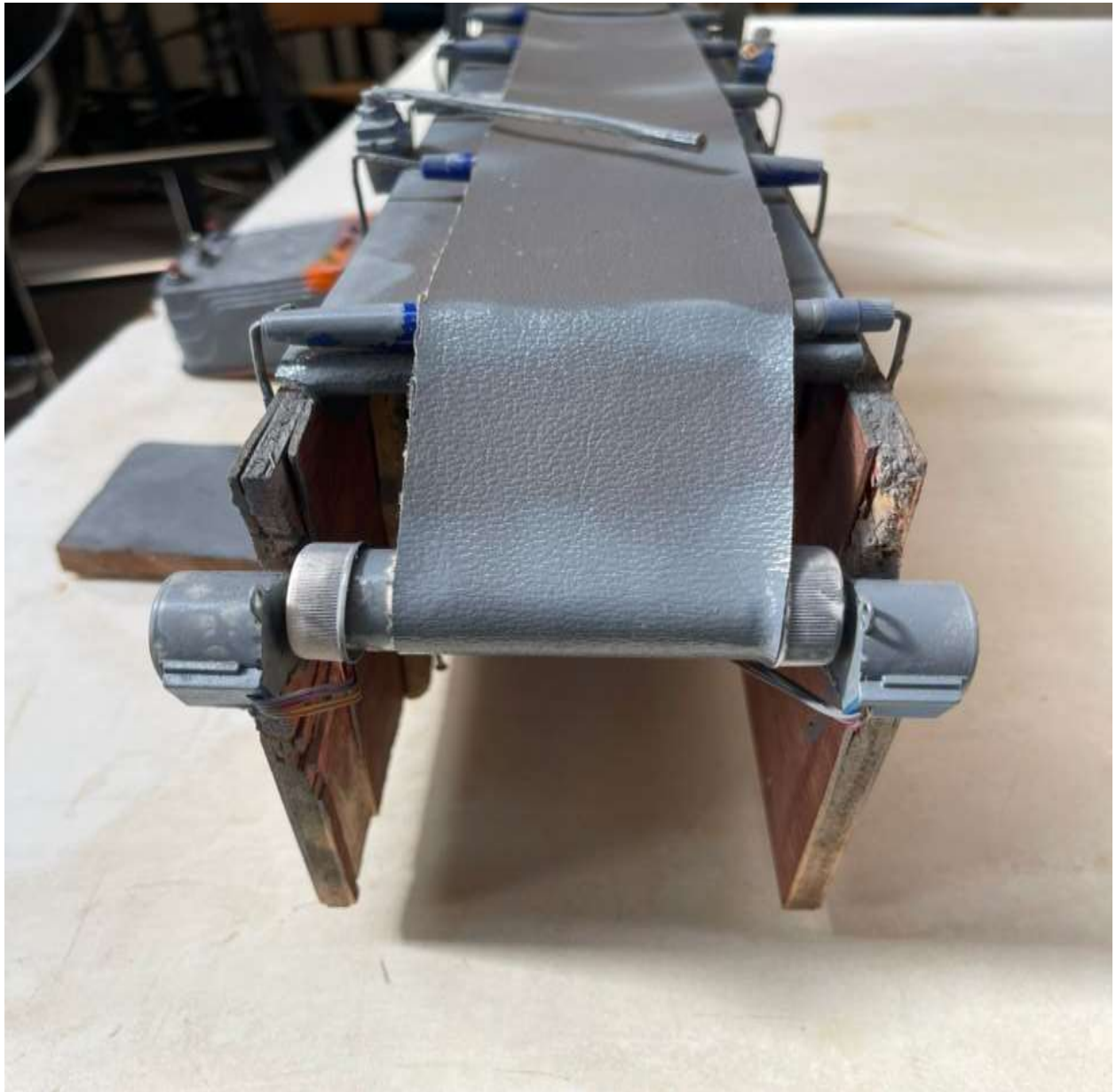


Figure 4:3 conveyor system with 28byj-48 stepper motor

Objective 2: To develop an image processing algorithm capable of detecting a fruit's presence and analysing its dominant colour in real-time.

The Edge Impulse TinyML model deployed on the ESP32-CAM successfully detected and classified bell peppers into green, yellow and red categories in real time. Images captured by the ESP32-CAM were processed using the trained machine learning model, which achieved an overall classification accuracy of 99.6%. The trained Edge Impulse TinyML model achieved 100% precision, recall and F1-score on the validation dataset. However, when integrated into the complete sorting prototype, the overall system achieved an operational sorting accuracy of 99.6% due to mechanical and environmental factors. The average inference time was approximately 150 ms per image, demonstrating the suitability of Edge Impulse-based TinyML for embedded agricultural sorting applications.

```
/* Function prototypes ----- */
bool ei_camera_init(void);
void ei_camera_deinit(void);
bool ei_camera_capture(uint32_t img_width, uint32_t img_height, uint8_t *out_buf);
static int ei_camera_get_data(size_t offset, size_t length, float *out_ptr);
void print_inference_result(ei_impulse_result_t result);
void send_inference_results(ei_impulse_result_t result);

/* setup() - runs once ----- */
void setup() {
    // USB Serial for debugging
    Serial.begin(115200);
    while (!Serial); // Wait for serial monitor (optional)

    // UART for communication with main ESP32 (pins 13,14)
    Serial1.begin(115200, SERIAL_8N1, UART_RX_PIN, UART_TX_PIN);

    // Flash LED pin
    pinMode(FLASH_PIN, OUTPUT);
    digitalWrite(FLASH_PIN, LOW);

    Serial.println("Edge Impulse Inferencing with UART output");
    if (ei_camera_init() == false) {
        Serial.println("Failed to initialize Camera!");
        return;
    }
    Serial.println("Camera initialized");
}
```

Figure 4:4 working inferencing code

Objective 3: To design and integrate a mechanical sorting mechanism activated by the processing unit's decisions.

The servo-actuated ejection mechanism responded within 128 ms on average. During testing the servo motors successfully diverting peppers into the correct collection bins based on classification results. Figure below shows how the servo motors were placed on the conveyor to block and eject pepper from the belt at a given time.



Figure 4:5 servo placement on the conveyor

4.7.1 Performance Improvements

4.7.1.1 Accuracy

The Edge Impulse TinyML model achieved a validation precision, recall and F1-score of 100%, demonstrating excellent classification capability. During integrated system testing, an overall sorting accuracy of 99.6% was achieved. The slight difference between model and system performance was primarily attributed to mechanical factors such as conveyor positioning tolerances, servo response timing and lighting variations rather than classification errors.

4.7.1.2 Throughput

The developed system achieved a processing throughput of 1 pepper per 2 seconds, representing a significant improvement. The conveyor system operates continuously without requiring intermittent pauses, thereby improving overall throughput. For small-scale farming cooperatives, this increased processing speed has the potential to substantially reduce sorting time, enabling work that would typically require an entire day to be completed within a few hours.

4.7.1.3 Cost

The prototype was developed at an estimated component cost of approximately US\$85, making it significantly more cost-effective than commercially available automated sorting systems. This low-cost implementation was achieved through the use of readily available and inexpensive hardware components, including the ESP32-CAM module and the 28BYJ-48 stepper motor, without compromising system performance. Consequently, the proposed system is economically viable for small-scale and medium-scale farmers. The use of widely available components enhances maintainability, as replacement parts can be sourced locally at relatively low cost, thereby reducing maintenance expenses and minimizing system downtime.

4.7.2 Shortcomings identified

4.7.2.1 Sensitivity to illumination conditions

The system accuracy decreased to 89.3% under low-light conditions, indicating although the Edge Impulse TinyML model demonstrated improved robustness compared with conventional colour thresholding approaches, its performance remains dependent on adequate and consistent illumination. This suggests that controlled lighting conditions are necessary to achieve optimal classification accuracy.

4.7.2.2 Performance degradation at higher conveyor speeds

Classification accuracy declined when the conveyor speed exceeded 120 mm/s due to the effects of motion blur and reduced image processing time. This limitation constrains the maximum throughput of the system and indicates the need for improved image acquisition techniques or higher processing capabilities for operation at increased conveyor speeds.

4.7.2.3 Root Cause Analysis of Errors

Analysis of the small percentage of sorting errors revealed that the majority originated from mechanical rather than classification failures. Occasional inaccuracies occurred when peppers were not centred within the camera field of view or when conveyor vibration altered fruit orientation immediately before image capture. Additional errors were observed under low-light conditions where image quality degraded. Since the Edge Impulse model achieved 100% validation performance, these findings indicate that system-level errors were primarily associated with sensing and actuation mechanisms rather than the machine learning model itself.

4.8 Discussion of Findings

4.8.1 Accuracy Improvement

Unlike LDR sensors, which are highly affected by ambient lighting variations, the Edge Impulse machine learning model analyses image features extracted from captured pepper images. This enables more reliable classification and improves robustness when compared with conventional sensor-based approaches. The achieved accuracy of 99.6% demonstrates the effectiveness of TinyML-based classification for low-cost agricultural automation.

4.8.2 Processing Speed

The system achieved a throughput of approximately 1 pepper per 2 seconds, making it considerably faster than manual sorting and typical low-cost LDR-based systems. This improvement was made possible by the efficient Edge Impulse TinyML inference pipeline and the optimised conveyor speed, thereby addressing the low processing speed identified as a research gap.

4.8.3 Effect of Lighting Conditions

The decrease in accuracy to 89.3% under poor lighting conditions confirms that illumination remains an important factor in computer vision systems. This finding is consistent with previous studies and highlights the importance of controlled lighting to maintain high classification accuracy in practical applications.

4.8.4 Future Improvements to the TinyML Model

The successful deployment of an Edge Impulse TinyML model on the ESP32-CAM demonstrated the practicality of implementing machine learning directly on low-cost embedded hardware. The model achieved high classification accuracy while maintaining low inference times, confirming the suitability of Edge AI for agricultural sorting applications. Future work may focus on training larger datasets and incorporating additional quality parameters such as fruit size, shape and defect detection.

4.8.5 Addressed research gaps

Table below summarises the key research gaps identified in the literature and how the proposed prototype addresses them. The developed system improves classification accuracy, processing speed and cost-effectiveness through the use of ESP32-CAM-based computer vision and Edge Impulse TinyML classification, while also providing a foundation for future TinyML integration.

Gap	Description	Solution
Narrow sorting criteria	Existing work treats green/yellow/red as ripening stages, not independent grades, and skips fruit verification.	Sorts into three distinct market classes and uses contour detection to confirm each object is a pepper.
Lack of accuracy	LDRs and basic sensors are thrown off by ambient light fluctuations.	Uses ESP32-CAM computer vision combined with an Edge Impulse TinyML classification model, achieving high sorting accuracy and improved robustness compared with conventional sensorbased systems.
High cost	Other designs need many analogue parts, shielding, and complex wiring.	Total parts cost under US\$85 using only an ESP32-CAM and minimal mechanics.

Outdated technology	Persistence of LDR-based detection despite modern vision and AI tools.	Adopts current OpenCV + ESP32-CAM pipeline with built-in tinyML capability through Edge Impulse.
Low processing speed	Slow sensors and uncontrollable motors force very slow conveyor operation.	Fast 150 ms image processing, speed-controlled conveyor, and instant servo ejection deliver high throughput.

Table 4:5 addressed research gaps

4.9 Summary

This chapter presented the implementation, testing and performance evaluation of the proposed automated bell pepper colour sorting system. Experimental results demonstrated that the prototype successfully achieved the project objectives by accurately classifying peppers using an Edge Impulse TinyML model deployed on the ESP32-CAM platform and automatically sorting them using a servo-actuated mechanism. The system achieved an overall classification accuracy of 99.6%, processed approximately one pepper every two seconds, and maintained reliable performance under bright and moderate lighting conditions, although accuracy decreased under dim lighting. The results also showed that the prototype provides a low-cost and efficient alternative to conventional LDR-based sorting systems while addressing the key research gaps identified in the literature. The next chapter presents the conclusions drawn from this study and provides recommendations for future improvements.

5 CHAPTER 5: CONCLUSION AND RECOMMENDATIONS

5.1 Summary of the Study

The aim of this study was to design and develop a low-cost automated pepper colour sorting system using computer vision and embedded systems technology. The research addressed the limitations of manual sorting and conventional LDR-based sorting methods, which are often labour-intensive, slow and sensitive to environmental conditions.

To solve this problem, a machine vision-based sorting system was developed using an ESP32CAM module and a TinyML image classification model created with Edge Impulse. The system was designed to automatically detect, classify, and sort peppers according to their colour. An integrated conveyor mechanism transported peppers through the inspection stage, while a microcontroller-based control system tracked object position and actuated the sorting mechanism to direct peppers into the appropriate collection bins.

5.2 Limitations

Although the prototype achieved its intended objectives, several limitations were identified.

1. The system performance was evaluated under controlled laboratory conditions and may require additional calibration for outdoor or industrial environments.
2. Classification accuracy decreased under poor lighting conditions, indicating the need for controlled illumination during operation.
3. The conveyor speed was limited by the 28BYJ-48 stepper motor, restricting the maximum system throughput.
4. The prototype was tested using only three bell pepper colour classes (green, yellow and red) and did not consider differences in size, shape or surface defects.

These limitations do not invalidate the findings but highlight opportunities for further improvement.

5.3 Recommendations for Future Work

Future research should focus on improving the performance and applicability of the developed prototype. The following recommendations are proposed:

1. Replace the 28BYJ-48 stepper motor with a higher-speed motor to increase conveyor throughput.
2. Incorporate an enclosed LED lighting system to minimise the influence of ambient illumination.
3. Extend the system to classify additional quality attributes such as fruit size, shape and surface defects.
4. Develop an IoT-based monitoring system for remote performance monitoring, data logging and production analysis.

These improvements would increase the system's robustness and make it more suitable for commercial agricultural applications.

5.4 Summary

This chapter summarised the outcomes of the study and discussed how the developed ESP32CAM-based pepper sorting system addressed the identified research problem. The chapter presented the conclusions drawn from the findings, highlighted the main contributions of the project, identified the limitations of the prototype, and proposed recommendations for future improvements and further research.

References

- Ariza-Sentís, M. *et al.* (2024) "Object detection and tracking in Precision Farming: a systematic review," *Computers and Electronics in Agriculture*. Elsevier B.V. Available at: <https://doi.org/10.1016/j.compag.2024.108757>.
- Bazargani, K. and Deemyad, T. (2024a) "Automation's Impact on Agriculture: Opportunities, Challenges, and Economic Effects," *Robotics*. Multidisciplinary Digital Publishing Institute (MDPI). Available at: <https://doi.org/10.3390/robotics13020033>.
- Bazargani, K. and Deemyad, T. (2024b) "Automation's Impact on Agriculture: Opportunities, Challenges, and Economic Effects," *Robotics*. Multidisciplinary Digital Publishing Institute (MDPI). Available at: <https://doi.org/10.3390/robotics13020033>.
- Elhanashi, A. *et al.* (2024) "Advancements in TinyML: Applications, Limitations, and Impact on IoT Devices," *Electronics (Switzerland)*. Multidisciplinary Digital Publishing Institute (MDPI). Available at: <https://doi.org/10.3390/electronics13173562>.
- Elwakeel, A.E. *et al.* (2023) "Designing, Optimizing, and Validating a Low-Cost, Multi-Purpose, Automatic System-Based RGB Color Sensor for Sorting Fruits," *Agriculture (Switzerland)*, 13(9). Available at: <https://doi.org/10.3390/agriculture13091824>.
- Espressif Systems (2019) *esp32 datasheet*. Available at: www.espressif.com/en/subscribe.
- Hymel, S. *et al.* (2023) "Edge Impulse: An MLOps Platform for Tiny Machine Learning." Available at: <http://arxiv.org/abs/2212.03332>.
- Jiayun, L.U.O., Boyang, L.I. and Leung, C. (2024) "A Survey of Computer Vision Technologies in Urban and Controlled-environment Agriculture," *ACM Computing Surveys*, 56(5). Available at: <https://doi.org/10.1145/3626186>.
- Kallimani, R. *et al.* (2023) "TinyML: Tools, Applications, Challenges, and Future Research Directions." Available at: <https://doi.org/10.1007/s11042-023-16740-9>.
- Kohut, P. and Skop, K. (2024) "Vision Systems for a UR5 Cobot on a Quality Control Robotic Station," *Applied Sciences (Switzerland)*, 14(20). Available at: <https://doi.org/10.3390/app14209469>.
- Kulkarni, S.M. *et al.* (2025) "Grading and Classification of Tomatoes Using ESP32-CAM and Impulse Cloud based IoT Platform for Sustainability," *Procedia Computer Science*. Elsevier B.V., pp. 938–946. Available at: <https://doi.org/10.1016/j.procs.2025.03.277>.
- Legner, R. *et al.* (2021) "A four-level maturity index for hot peppers (*Capsicum annum*) using non-invasive automated mobile raman spectroscopy for on-site testing," *Applied Sciences (Switzerland)*, 11(4), pp. 1–13. Available at: <https://doi.org/10.3390/app11041614>.
- Mihigo, I.N. *et al.* (2022) "On-Device IoT-Based Predictive Maintenance Analytics Model: Comparing TinyLSTM and TinyModel from Edge Impulse," *Sensors*, 22(14). Available at: <https://doi.org/10.3390/s22145174>.

- Rasekh, M. *et al.* (2022) "Preliminary study non-destructive sorting techniques for pepper (*Capsicum annuum* L.) using odor parameter," *LWT*, 164. Available at: <https://doi.org/10.1016/j.lwt.2022.113667>.
- Saleem, M.H., Potgieter, J. and Arif, K.M. (2021) "Correction to: Automation in Agriculture by Machine and Deep Learning Techniques: A Review of Recent Developments (Precision Agriculture, (2021), 22, 6, (2053-2091), 10.1007/s11119-021-09806-x)," *Precision Agriculture*. Springer, pp. 2092–2094. Available at: <https://doi.org/10.1007/s11119-021-09824-9>.
- Seiphepi, G. *et al.* (2020) *Automatic Bell Pepper Colour Detector and Sorting Machine*, *International Journal of Engineering Research and Technology*. Available at: <https://doi.org/10.5281/zenodo.4315274> (Accessed: June 22, 2026).
- Sharma, A. *et al.* (2021) "Machine Learning Applications for Precision Agriculture: A Comprehensive Review," *IEEE Access*. Institute of Electrical and Electronics Engineers Inc., pp. 4843–4873. Available at: <https://doi.org/10.1109/ACCESS.2020.3048415>.
- Shin, S.-Y. *et al.* (2021) "Performance Testing and Evaluation of Drum-Type Stem-Separation Device for Pepper Harvester Testing and Evaluation of Drum-Type Stem-Separation Device for Pepper." Available at: <https://doi.org/10.3390/app>.
- Tolani, M. *et al.* (2025) "Machine learning based adaptive traffic prediction and control using edge impulse platform," *Scientific Reports*, 15(1). Available at: <https://doi.org/10.1038/s41598025-00762-4>.
- Wang, J.-J. (2022a) "Recognition system for fruit classification based on 8-layer convolutional neural network," *EAI Endorsed Transactions on e-Learning*, 7(23), p. 173455. Available at: <https://doi.org/10.4108/eai.17-2-2022.173455>.
- Wang, J.-J. (2022b) "Recognition system for fruit classification based on 8-layer convolutional neural network," *EAI Endorsed Transactions on e-Learning*, 7(23), p. 173455. Available at: <https://doi.org/10.4108/eai.17-2-2022.173455>.

Similarity report for chapter 1 to chapter 5

Anesu Bhobho

Chapter 1 to5 b221220b

 dissertation

Document Details

Submission ID

trn:oid::2945:395423036

Submission Date

Jun 24, 2026, 5:39 PM GMT+2

Download Date

Jun 25, 2026, 8:26 PM GMT+2

File Name

Chapter 1 to5 b221220b.docx

File Size

3.4 MB

57 Pages

9,657 Words

59,769 Characters



Page 1 of 63 - Cover Page

Submission ID: trn:oid::2945:395423036



Page 2 of 63 - Integrity Overview

Submission ID: trn:oid::2945:395423036

8% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

-  **75 Not Cited or Quoted** 7%
Matches with neither in-text citation nor quotation marks
-  **7 Missing Quotations** 1%
Matches that are still very similar to source material
-  **4 Missing Citation** 0%
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted** 0%
Matches with in-text citation present, but no quotation marks

Top Sources

- 1%  Internet sources
- 2%  Publications
- 6%  Submitted works (Student Papers)