HOCHSCHULE SCHMALKALDEN

UNIVERSITY OF APPLIED SCIENCES

Smart Modular Robotics for Precision Fruit Harvesting: A Vision-Based System Integrating AI and Soft End-Effectors in Agriculture 4.0

Master Thesis By:

Abhijith Balamurali
Matriculation number: 315603

August 26, 2025

1^{st} *Supervisor* : **Prof. Dr.-Ing. Frank Schrödel**
2^{nd} *Supervisor* : **Prof. Dr.-Ing. Maria Schweigel**

Department of Mechanical Engineering, Schmalkalden University of Applied Sciences

Declaration of Originality

I hereby declare that I am the sole author of the present thesis with the title: "Smart Modular Robotics for Precision Fruit Harvesting: A Vision-Based System Integrating AI and Soft End-Effectors in Agriculture 4.0" I did not submit this thesis for other examination purposes. To the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices.

Place, Date Signature

Acknowledgement

The successful completion of this thesis would not have been possible without the invaluable support and guidance I received throughout the journey. First and foremost, I would like to express my deepest gratitude to my supervisor, Prof. Dr.- Ing. Frank Schrödel, whose expertise, insightful feedback, and unwavering encouragement have been instrumental in shaping this project. His approach to problem-solving greatly enhanced my analytical thinking and significantly contributed to my growth in independent work and conceptual development. I am equally thankful to my co-supervisor, Prof. Dr.-Ing. Maria Schweigel, for her consistent support and for offering a fresh perspective that helped me approach the research challenges more effectively. I would also like to acknowledge the support of the faculty and staff of Schmalkalden University of Applied Sciences for creating an academic environment that inspired learning and innovation. My heartfelt appreciation goes to my parents, whose unconditional love, encouragement, and belief in my abilities have always been a source of strength. Lastly, I am truly grateful to my friends, whose constant support, motivation, and companionship made this academic journey enjoyable and fulfilling. Their presence was a vital part of my success.

Contents

Contents iii

List of Figures vi

Abbreviations viii

Abstract ix

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Problem Statement . . . 1
- 1.3 Addressing Goals . . . 2

2 Current State Analysis 3
- 2.1 Background . . . 3
  - 2.1.1 From Mechanisation to Autonomy: The Evolution of Agricultural Robotics 3
  - 2.1.2 Digital Transformation and the Emergence of Intelligent Agriculture . . . 4
  - 2.1.3 Application of YOLO and Custom-Designed Intelligent Teaching Aids in Robotic Arm-Based Fruit Classification and Grasping Instruction . . . 5
  - 2.1.4 Modular and Reconfigurable Robotic Architectures for Adaptive Agriculture . . . 6
  - 2.1.5 Advancements in Soft End-Effectors for Delicate Fruit Harvesting . . . 7
  - 2.1.6 AIoT Integration and the Future of Smart Agriculture . . . 7
  - 2.1.7 Design and Development of a Peduncle-Holding End Effector for Robotic Harvesting of Mango . . . 8
  - 2.1.8 Robot Farming: Transforming Agriculture with Advanced Techniques for Efficient Cultivation, Irrigation and Crop Monitoring . . . 9
  - 2.1.9 Semantic Obstacle Avoidance of Robotic Arm for Fruit Harvesting . . . 10
  - 2.1.10 Advancements in Agricultural Ground Robots for Specialty Crops . . . 10
- 2.2 Industry Landscape of Fruit-Picking Robots . . . 11
  - 2.2.1 Commercial Systems . . . 11
  - 2.2.2 Emerging & Specialized Systems . . . 14
  - 2.2.3 Market Trends & Challenges . . . 16
  - 2.2.4 Academic Research Highlights . . . 17
  - 2.2.5 Comparative Evaluation of This Project with Existing Systems . . . 18

3 Fundamentals 20

3.1 Introduction to Vision Systems in Agricultural Robotics . . . 20
3.1.1 Basic Principles of Vision-Based Perception . . . 20
3.1.2 Radiometric Considerations and Sensor Response . . . 21
3.1.3 Depth Estimation with Stereo or RGB-D Cameras . . . 21
3.1.4 Color Space Transformations . . . 21
3.1.5 Relevance to Agricultural Automation . . . 21
3.2 Software Selection . . . 22
3.3 Hardware Selection . . . 24

4 Concept and Methodology **29**
4.1 Methodology . . . 29
4.1.1 System Architecture . . . 29
4.1.2 Why YOLOv8 Was Chosen for Tomato Detection . . . 30
4.1.3 Dataset . . . 31
4.2 Arduino Code for Servo-Based Gripper Control . . . 34
4.2.1 Important Considerations for Servo Motor Setup . . . 35
4.2.2 Wiring Diagram . . . 36
4.2.3 Arduino Nano . . . 37
4.2.4 MG996R Servo Motor . . . 37
4.2.5 Intel RealSense Depth Camera . . . 37
4.2.6 Buck Converters . . . 37
4.2.7 12V Power Supply . . . 37
4.2.8 Ground (GND) . . . 37
4.3 V-Model Integration . . . 38
4.3.1 Relevance of the V-Model to This Project . . . 38
4.3.2 Custom V-Model Mapping for the Project . . . 38
4.3.3 Detailed Project Requirements . . . 39
4.3.4 Visual Representation . . . 40
4.4 Design Concept Generation . . . 41
4.4.1 **Initial Requirements and Design Goals:** . . . 41
4.4.2 Ideation and Concept Inspiration . . . 42
4.4.3 Component-Level Design and Redesign . . . 42
4.5 Intel RealSense L-Link . . . 48
4.5.1 Technical Features . . . 49
4.6 Technical Drawing and View Analysis . . . 49
4.6.1 Purpose and Significance of Technical Drawings . . . 49
4.6.2 Projected Views . . . 49
4.6.3 Orthographic Projection Views of the End-Effector Assembly . . . 49
4.6.4 Exploded View . . . 52
4.7 Manufacturing Process . . . 54
4.7.1 Additive Manufacturing: An Overview . . . 54
4.7.2 Material Selection . . . 54
4.7.3 Manufacturing 3D Printable Components . . . 54
4.7.4 Post-Processing Steps . . . 55
4.7.5 Commercially Acquired Components . . . 55
4.7.6 Print Settings and Process Optimization . . . 55
4.7.7 Types of 3D Printing Machines . . . 55
4.7.8 Toolpaths and Slicing Software . . . 56
4.7.9 Toolpaths and Slicing Workflow with Bambu Lab P1S . . . 57

5 Testing and Validation **59**
5.1 Prediction Testing Cycle . . . 59
5.1.1 Initial Tests on Static Images . . . 59
5.1.2 Transition to Real-Time Testing Using Webcam . . . 60
5.1.3 Cycle 1 – Live Detection of Ripe Tomato . . . 60
5.1.4 Cycle 2 – Real-Time Unripe Tomato Detection Using RealSense Camera . 61
5.1.5 Cycle 3 – Multi-Object Detection in Real-Time . . . 61
5.1.6 General Observations . . . 61
5.2 Precision-Recall Curve Analysis . . . 62
5.3 Understanding the Confusion Matrix . . . 63
5.3.1 Model Training Progress and Evaluation Metrics . . . 65
5.4 Understanding the F1-Confidence Curve . . . 66
5.4.1 F1 Score and Confidence Threshold Analysis . . . 66
5.4.2 Overall Model Performance . . . 67
5.4.3 Considerations for Real-World Deployment . . . 67
5.5 Requirements Validation . . . 68

6 Discussion **70**
6.1 Overview . . . 70
6.2 Vision Model Completion . . . 70
6.2.1 Additional Testing and Validation . . . 70
6.2.2 Robustness Improvements . . . 70
6.2.3 Real-Time Deployment Goals . . . 71
6.3 Completed Functional and Technical Objectives . . . 71
6.4 Hardware Assembly . . . 72
6.4.1 Mechanical Assembly Plan . . . 72
6.4.2 Servo Integration and Actuation Testing . . . 72
6.4.3 Full System Calibration . . . 72
6.5 Integrated System Goals . . . 72
6.5.1 End-to-End Pipeline Integration . . . 72
6.5.2 Agricultural Field Deployment . . . 73
6.6 Industrial and Research Extensions . . . 73
6.6.1 Modular Scalability . . . 73
6.6.2 Integration with Autonomous Rovers . . . 73
6.6.3 Potential for Cross-Domain Use . . . 73

7 Conclusion **74**
7.1 Conclusion . . . 74

8 Appendix **75**
8.1 Additive Manufacturing . . . 75
8.1.1 Understanding Toolpaths . . . 75
8.1.2 Slicing with Bambu Studio and Printer Integration . . . 76

Bibliography **77**

List of Figures

2.1 Apple Harvester in Action . . . 12
2.2 Strawberry picking in real time . . . 12
2.3 Strawberry picking in real time . . . 13
2.4 Automated Harvesting in real time . . . 13
2.5 Automated Harvesting in real time . . . 14
2.6 Automated fruit picking . . . 14
2.7 Automated harvesting . . . 15
2.8 Automated harvesting Robot . . . 15
2.9 Automated harvesting Robot . . . 16
2.10 Automated harvesting Robot . . . 17
2.11 Automated berry picking . . . 17

3.1 Custom figure showing the software stack — Source: Generated with AI . . . 22
3.2 Exploded view of the RealSense camera parts . . . 25
3.3 3D printed components fabricated with PLA for the end-effector system. . . . 28

4.1 Pipeline of System Architecture — Source: AI generated . . . 29
4.2 Yolo models object detection curves . . . 30
4.3 Example of dataset used — Source: Custom collage of images made for model training . . . 32
4.4 Schematic wiring Diagram of Assembly, Source: AI Generated . . . 36
4.5 Formal Wiring Schematic of the End-Effector Control System, Source: AI Generated 38
4.6 Project V-Model, Source: AI Generated . . . 41
4.7 Clamp Technical drawing, source= generated from custom design . . . 43
4.8 Main Screw Technical drawing, source= generated from custom design . . . 43
4.9 Integrated finger Design — Source: Custom design through Fusion 360 . . . 44
4.10 Servo Horn Technical drawing, source= generated from custom design . . . 45
4.11 Base plate — Source: Custom design through Fusion 360 . . . 45
4.12 Servo motor Technical drawing, source= generated from custom design . . . 46
4.13 Servo motor mount Technical drawing, source= generated from custom design . . . 47
4.14 RealSense mount Technical drawing, source= generated from custom design . . . 47
4.15 RealSense Link Technical drawing, source= generated from custom design . . . 48
4.16 Orthographic projection views of the complete end-effector assembly. . . . 51
4.17 Exploded view of the modular end-effector assembly showing part-to-part integration. . . . 53

5.1 Collage of cycle 1 testing results — Source: adapted from custom-developed results 59

5.2 Collage of cycle 2 testing results — Source: adapted from custom-developed results 60
5.3 Precision-recall curve — Source: adapted from custom-developed results 63
5.4 Confusion Matrix — Source: adapted from custom-developed results 64
5.5 Evaluation Metrics — Source: adapted from custom-developed results 65
5.6 F1 Confidence Curve — Source: adapted from custom-developed results 66

8.1 View of my components in slicer getting prepped for printing 75

Abbreviations

Abbreviation	Meaning
AI	Artificial Intelligence
API	Application Programming Interface
AIoT	Artificial Intelligence of Things
AMS	Automatic Material System
AMR	Autonomous Mobile Robot
CNN	Convolutional Neural Network
CUDA	Compute Unified Device Architecture
CLI	Command Line Interface
CAD	Computer-Aided Design
DoF	Degrees of Freedom
DLP	Digital Light Processing
FDM	Fused Deposition Modeling
GPS	Global Positioning System
GPU	Graphics Processing Unit
IOT	Internet of Things
I/O	Input/Output
LiDAR	Light Detection and Ranging
MG996R	Servo Motor model MG996R
PLA	Polylactic Acid
PETG	Polyethylene Terephthalate Glycol
PWM	Pulse Width Modulation
RGB-D	Red Green Blue-Depth
SDK	Software Development Kit
SLA	Stereolithography
SLS	Selective Laser Sintering
TPU	Thermoplastic Polyurethane
TCP	Tool Centre Point
UR5e	Universal Robot model 5
UGV	Unmanned Ground Vehicle
UAV	Unmanned Aerial Vehicle
USB	Universal Serial Bus
WSL	Windows Subsystem for Linux
YOLOv8s	object detection model v8 small (You Only Look Once)

Abstract

This thesis presents the design, development, and fabrication of a modular robotic end effector integrated with advanced perception and actuation systems for use in precision harvesting and surface sampling tasks. The core objective is to enhance autonomous manipulation capabilities by leveraging computer vision, mechanical optimisation, and additive manufacturing. The end effector was designed for integration with the UR5e collaborative robotic arm, ensuring high repeatability and compatibility with industrial environments. Key contributions include the deployment of a trained YOLOv8-based deep learning model for real-time classification of ripe and unripe produce using an Intel RealSense camera and the mechanical realisation of a three-fingered claw gripper driven by an MG996R servo and an embedded gear-screw mechanism. Several components were prototyped using FDM 3D printing on the Bambu Lab P1S, which significantly reduced manufacturing time while maintaining mechanical fidelity. The design emphasises modularity, energy efficiency, and ease of assembly. A slicing workflow optimised using Bambu Studio was employed to validate toolpaths and ensure printability. The complete system architecture reflects principles of Industry 4.0 and 5.0, combining intelligent automation with human-centric design. This work demonstrates a scalable approach to developing intelligent robotic tools for modern agriculture and exploration, with potential for adaptation across various domains requiring soft-touch automation and modular deployment.

Chapter 1

Introduction

1.1 Motivation

Despite technological advancements in agriculture, the harvesting of delicate fruits such as tomatoes, kiwifruit, and mangoes still relies heavily on manual labour accounting for 60–70% of the overall process which makes it prone to inefficiencies, labour shortages, and high costs [1]. Traditional methods and even some mechanised approaches often result in bruising or injury to the produce, thereby reducing shelf life and market value [2]. Furthermore, current robotic solutions have yet to achieve widespread adoption: numerous reviews highlight persistent shortcomings in adaptability, damage rates, and cycle times, particularly in dynamic field environments [3].

The global agricultural landscape is undergoing a significant transformation. Traditional farming practices, once characterised by intensive manual labour and inconsistent outputs, are now giving way to smarter, more efficient systems. The confluence of Artificial Intelligence (AI), the Internet of Things (IoT), and robotics has ushered in a new era of precision agriculture, often referred to as Agriculture 4.0 [4, 5]. These innovations aim to address challenges such as declining labour availability, increasing food demand, and environmental sustainability [6, 7].

1.2 Problem Statement

Despite the advancements, several challenges remain. Data sparsity, connectivity limitations in rural areas, and high computational demands constrain the full-scale adoption of AI-based farming [4, 8]. Additionally, ensuring robust performance in varying lighting, weather, and crop conditions is an ongoing area of research [9, 10].

The COVID-19 pandemic further highlighted the vulnerabilities in global food supply chains and accelerated the shift toward automation and digital technologies in agriculture [11, 12]. As AI and IoT continue to converge, future agricultural systems are expected to achieve greater resilience, sustainability, and scalability [13, 14]. This project addresses these challenges by developing an intelligent, modular robotic harvesting system that combines real-time object detection (YOLO), a soft fluid-driven end-effector, and AIoT integration. The system aims to enable gentle, accurate, and autonomous fruit harvesting, reducing dependency on manual labour while increasing efficiency and crop handling precision in alignment with Agriculture 4.0 principles.

In summary, the integration of AI, robotics, and smart sensing technologies into agriculture holds transformative potential. As the field matures, multidisciplinary collaboration, pol-

icy support, and continued innovation will be essential for realising the vision of truly autonomous, precise, and sustainable farming systems.

1.3 Addressing Goals

Among the most promising advancements in this domain is the development of autonomous robotic systems equipped with vision-based algorithms, capable of performing complex tasks like fruit detection, classification, and harvesting [15]. These systems combine mechanical dexterity, real-time decision-making, and data-driven adaptability to execute tasks traditionally reliant on human labour. For instance, robots now play a crucial role in mitigating the impact of labour shortages and reducing the physical burden on farm workers [3, 16].

One of the critical challenges in robotic harvesting is ensuring minimal damage to delicate fruits during picking. This has led to the evolution of soft robotic end-effectors that utilize elastic materials and fluid-driven mechanisms for safe and effective grasping [17, 16]. For example, the elastic fluid-driven end-effector for tomato and kiwifruit harvesting demonstrated high gripping efficiency while preserving fruit integrity [18, 19]. The incorporation of soft fingers allows for adaptability to varying shapes and sizes, which is vital for crops like strawberries, cucumbers, and grapes [20, 17].

Meanwhile, computer vision algorithms, particularly the YOLO (You Only Look Once) family of detectors, have revolutionised object detection in agriculture. YOLO's ability to detect objects in real-time, its single-pass processing model, and its improved accuracy in recent versions have made it ideal for robotic applications [21, 22, 23]. Systems that integrate YOLO with robotic manipulators have proven effective in tasks such as green mango detection [24] and apple retrieval [25].

Furthermore, modular and reconfigurable robotic architectures offer flexibility in adapting to diverse farming environments and tasks. Modular robots, such as the Hefty system, enable the incremental assembly of functional units and reduce development costs [26, 27]. These robots can be customised with specific end-effectors, control systems, and mobility solutions to suit various crops and operational conditions.

The remainder of this thesis is organised as follows: **Chapter 2** provides a detailed analysis of the current state of technology in the industry, highlighting existing workflows and limitations. **Chapter 3** presents an overview of the fundamental basis of my project. **Chapter 4** describes in detail the concept and methodology and project execution flow. **Chapter 5** elaborates on the process followed to validate my model is ready for real-world functions. **Chapter 6** includes ideas and discussions about the project and its future scope. **Chapter 7** presents the conclusions drawn from the entire report.

Chapter 2

Current State Analysis

2.1 Background

Before delving into the specific technical aspects of this work, it is important to first establish a basic understanding of the underlying agricultural technologies and their evolution. This provides essential context for appreciating the significance and relevance of the methods and systems discussed in the subsequent sections.

2.1.1 From Mechanisation to Autonomy: The Evolution of Agricultural Robotics

Agriculture has historically relied on manual labour and animal-based traction systems. For centuries, crop production processes such as sowing, irrigation, weeding, and harvesting were physically demanding, time-consuming, and labour-intensive. The first wave of mechanisation in the 19th and early 20th centuries introduced machines like tractors and mechanical harvesters, which increased efficiency but still depended heavily on human operation. Early innovations included horse-drawn seed drills in the early 1700s, horse-drawn reapers from the 1830s, steam-powered threshing machines from the late 19th century, and the emergence of internal combustion tractors in the early 20th century [28, 29].

The evolution of autonomy in agricultural robotics has significantly transformed the landscape of modern farming. Early levels of autonomy, aligned with SAE J3016 standards, were achieved in the 1990s through the development of vision-based local navigation systems [30]. The integration of technologies such as GPS and LiDAR enhanced the autonomous capabilities of agricultural robots, allowing them to navigate complex environments more reliably [31].

Driverless tractors and autonomous vehicles became feasible as algorithms and AI matured, enabling machines to independently solve operational challenges [32]. Notably, vision systems advanced to distinguish crops from weeds with significant accuracy, as demonstrated in early weeding robots that used greyscale image processing and object detection techniques [33].

The urgency for autonomous solutions is further exacerbated by climate change, which demands innovations in sustainable agriculture. Climate-induced challenges, such as labour shortages and unpredictable weather, are driving new investments into the field, accelerating the development of high-autonomy systems [34]. Additionally, public interest in robotics surged during the COVID-19 pandemic, leading to increased venture capital funding towards agricultural robotics companies [11].

A growing global population continues to stress the agricultural supply chain, requiring increased food production efficiency [12]. Autonomous robots, such as fruit-picking systems equipped with RealSense cameras, promise to meet these demands by optimising yield and reducing labour dependency.

The development of robotic systems for agriculture began gaining momentum in the 1990s with vision-based navigation solutions [30]. As technology progressed, autonomous mobile robots (AMRs), unmanned ground vehicles (UGVs), and manipulators were introduced for tasks such as weeding, planting, and harvesting [35]. Robotic arms specifically designed for fruit harvesting became a major research focus due to the complexities involved in handling delicate biological objects without causing damage [3].

This literature review focuses primarily on autonomous agricultural machinery, including autonomous mobile robots (AMRs), driverless tractors, fruit-picking robots, and horticultural robots [36]. Automation of ancillary systems such as irrigation or storage remains outside the current scope. The development trend suggests that agricultural robots will become increasingly independent, although human supervision is still essential for troubleshooting unforeseen challenges.

Building on the advancements in autonomy, the integration of intelligent robotic systems into core agricultural operations has given rise to a new era of robot farming streamlining tasks such as cultivation, irrigation, and crop monitoring with unprecedented efficiency.

2.1.2 Digital Transformation and the Emergence of Intelligent Agriculture

With the growing global population and challenges such as labour shortages and climate variability, agriculture has entered a new era known as Agriculture 4.0. This phase leverages digital technologies, including AI, the IoT, big data, and robotics, to promote precision farming [4]. Unlike traditional approaches, Agriculture 4.0 enables real-time decision-making and data-driven interventions to optimise yields, reduce waste, and minimise environmental impact [5].

The growing global population, projected to reach nearly 10 billion by 2050, alongside a decreasing agricultural labour force, demands a more efficient and sustainable approach to global food supply chains [37]. AI offers transformative capabilities to tackle these challenges by enhancing productivity, ensuring precision, and improving decision-making across various agricultural processes [38].

Recent advancements in electronics and digital technologies have opened new avenues for innovation in agriculture, including robotic platforms, plant health sensors, unmanned aerial vehicles (UAVs), and advanced soil nutrient mapping systems [39]. These tools have enabled smarter, more targeted farming practices, thereby promoting sustainable crop production and resource management [40].

The application of AI in agriculture provides distinct advantages such as adaptability, efficiency, precision, and cost-effectiveness [41]. Expert systems and AI-driven decision support frameworks are being increasingly utilised to guide farmers through complex farming operations, from planting and fertilisation to harvesting and marketing [42]. These intelligent systems help optimise input use, forecast crop yields, and manage pests and diseases with higher accuracy.

Agriculture remains vital to global economic development, currently utilising about 37.7% of the Earth's land surface [37]. It plays a pivotal role not only in food security but also in the economic growth of both developed and developing nations. Enhancements in agricultural

productivity contribute significantly to rural development and broader economic transformation [7].

Despite these advancements, farmers must constantly adapt to numerous uncertainties such as seasonal weather variations, fluctuating prices of farming materials, soil degradation, crop failures, weed infestations, pest damage, and climate change [43]. The rising cost of agricultural labour further exacerbates these challenges. AI technologies, including machine learning algorithms and precision agriculture systems, offer tools to help farmers navigate these uncertainties, improving resilience and sustainability [5].

In conclusion, the integration of AI into agriculture is not merely an option but a necessity for addressing labour shortages, optimising resource use, and mitigating the risks associated with climate variability. As highlighted by Ashe and Mengistu [44], future farming will increasingly rely on intelligent systems to ensure food security and sustainable agricultural development. While AI continues to redefine agricultural decision-making and automation at a systems level, its practical embodiment is increasingly seen in the evolution of ground robots—especially those tailored for handling the unique challenges of speciality crop cultivation.

2.1.3 Application of YOLO and Custom-Designed Intelligent Teaching Aids in Robotic Arm-Based Fruit Classification and Grasping Instruction

A major advancement in robot intelligence came with the integration of computer vision. The "You Only Look Once" (YOLO) algorithm, introduced by Redmon et al., revolutionised object detection by providing high-speed, real-time identification of objects [21]. Its single-pass detection and classification framework allowed for its application in domains like autonomous driving, security, and precision agriculture [23]. Over the years, YOLO has evolved through multiple versions, becoming increasingly accurate while maintaining low computational latency [15, 24].

The integration of image recognition technologies with robotic arm systems plays a vital role in the advancement of smart agriculture. The study by Wang et al. [45] demonstrates a teaching framework that combines the YOLO object detection algorithm with robotic automation to enable real-world fruit classification and sorting. Through this hands-on approach, students develop practical skills in AI-driven agricultural solutions and strengthen their understanding of machine vision and robotics [10].

The YOLO (You Only Look Once) algorithm has been praised extensively in the field of computer vision for its real-time processing capabilities and end-to-end training efficiency [21]. Unlike traditional two-stage detectors, YOLO treats object detection as a single regression problem, mapping raw images directly to bounding box coordinates and class probabilities [22]. Its simple yet powerful architecture enables it to achieve both high speed and competitive accuracy, making it highly suitable for real-time applications such as autonomous driving, surveillance systems, and robotic fruit picking [23].

With continuous improvements over its versions, YOLO has achieved accuracy levels comparable to those of two-stage detection frameworks [46]. Specifically, YOLOv4 optimises feature extraction, introduces spatial pyramid pooling, and enhances generalisation through advanced data augmentation methods [47]. As a result, YOLOv4 combines superior detection accuracy with outstanding processing speeds, making it ideal for integration with robotic arms in real-world tasks [48].

Hands-on training projects like the one developed by Wang et al. [45] not only enhance students' technical skills but also foster teamwork, communication, and innovative problem-

solving abilities. By addressing real-world challenges such as fruit variability in shape, orientation, and surface characteristics, students gain critical insights into the application of AI and robotics in agriculture [49].

In conclusion, the application of YOLO algorithms in robotic fruit harvesting systems bridges the gap between theoretical learning and practical implementation. It equips future engineers and researchers with the essential skills required to develop intelligent, efficient, and scalable agricultural technologies. While intelligent vision systems and teaching aids enhance a robotic arm's ability to identify and interact with fruits, the effectiveness of the physical grasp still relies heavily on the design of specialised end effectors especially for delicate crops like kiwifruit that demand gentle handling.

2.1.4 Modular and Reconfigurable Robotic Architectures for Adaptive Agriculture

Modular robotics offer an adaptable and scalable solution for a range of agricultural tasks. Systems like Thorvald and Hefty exemplify modular design, where robots can be configured using interchangeable parts to meet different task requirements [27, 26]. Modularity enhances system maintainability, reduces costs, and allows easier integration of new tools such as specialised end-effectors [50].

The introduction of modular robots in agriculture addresses the growing need for flexible, adaptable, and cost-efficient robotic systems. Modular robots, as discussed by Guri et al. [26], offer multiple configurations and reduce overall costs by allowing users to incrementally acquire only the necessary modules required for specific agricultural tasks. This design strategy not only facilitates research advancements but also enhances technology transfer from laboratory environments to real-world agricultural applications [27].

One of the critical drivers for agricultural robotics adoption is the mitigation of labour shortages and the reduction of worker exposure to strenuous and hazardous tasks [3]. Moreover, the adoption of modular and reconfigurable robots aligns closely with the goals of "Agriculture 4.0", promoting automation-driven, environmentally friendly farming methods aimed at minimising waste and maximising yields [3]. Agricultural robotics equipped with modular designs can thus support a wide range of tasks such as targeted pesticide application, selective harvesting, and precise weeding [35].

Effective agricultural robot design demands an optimal integration of mechanical structures, computational hardware, sensory equipment, and sophisticated algorithms [40]. Guri et al. [26] emphasise that for successful deployment, agricultural robots must be capable of perceiving the environment, navigating through complex terrains, and delicately manipulating sensitive crops and plants. These functionalities are achieved through careful selection and arrangement of sensors like LiDAR, RGB-D cameras, and GPS modules, combined with robust control and planning algorithms [51].

The modularity and reconfigurability of robotic systems like Hefty refer to the independent design of mechanical, computational, and sensory components, allowing for easy rearrangement based on task requirements [50]. Reconfigurability ensures that robots can adapt to various crop types, field layouts, and task-specific demands, thereby enhancing operational versatility and extending the robot's useful lifespan. This approach not only reduces development and acquisition costs but also facilitates repairability and scalability, which are crucial for widespread adoption of robotic systems in agriculture.

In conclusion, the Hefty robot represents a significant advancement toward versatile and cost-effective agricultural robotics. By emphasising modularity and reconfigurability, it paves the

way for broader adoption of robotic solutions capable of meeting the diverse and evolving needs of modern agriculture. While Hefty showcases the potential of modular robotics in broad agricultural tasks, targeted innovations like specialised end effectors are essential for handling crop-specific challenges—such as the precise and damage-free harvesting of fruits like mangoes.

2.1.5 Advancements in Soft End-Effectors for Delicate Fruit Harvesting

Handling soft and irregularly shaped objects like fruits requires gentle and adaptive gripping mechanisms. Traditional rigid grippers often caused bruising or damage, prompting the development of soft robotic end-effectors. These systems employ materials such as silicone, along with pneumatic or fluid-driven actuation, to provide compliant and adaptive gripping [15, 52]. Notable designs, such as the elastic fluid-driven tomato gripper and the soft end-effector for kiwifruit harvesting, demonstrate that deformation-based gripping ensures better contact with complex geometries while significantly reducing injury risk [53].

In response to global labour shortages and the growing demand for mechanised agriculture, robots have emerged as vital tools to facilitate intelligent operations and mitigate the impacts of population decline [15]. Soft end-effectors, in particular, offer promising solutions for delicate fruit harvesting, as they reduce reliance on manual labour while ensuring high-quality yields.

Despite these advancements, significant challenges persist in agricultural robotics research, especially regarding the development of stable robotic arms and end-effectors with robust operational performance [54, 47]. Designing systems that balance strength and gentleness remains critical for minimising fruit damage and achieving efficient harvesting [47].

Xu et al. demonstrated that the incorporation of a grasp-and-separate technique significantly improves harvest success rates. Their citrus harvesting robot, equipped with a gripper and blade combination, achieved a success rate of 95.23% [54]. This highlights the value of combining gripping and cutting mechanisms in robotic end-effectors.

Soft fingers fabricated from elastic materials provide a transformative advantage in agricultural harvesting. By interacting with fruits through soft and deformable surfaces, they substantially lower the risk of mechanical injuries [55]. The end-effector developed by Zhou et al. [15] exemplifies this approach: an elastic fluid-driven gripper capable of harvesting tomatoes without causing damage at an inflation pressure of 28.293 kPa.

The elastic fluid-driven end-effector by Xu Wang et al. [47] extends these concepts to kiwifruit harvesting. Its soft fingers, inflated with controlled air pressure, provide both a strong grip and high compliance, protecting fruits from bruising while maintaining efficiency. The compact structure and adaptability of the design allow stable grasping even for elliptical fruits like kiwifruit, which pose challenges for traditional rigid grippers [53].

In summary, the development of soft-bodied, fluid-driven end-effectors represents a critical advancement in agricultural robotics. These systems not only ensure gentle handling of delicate fruits but also improve operational stability and harvesting efficiency, aligning with the future goals of non-destructive, high-throughput, and intelligent agricultural automation.

2.1.6 AIoT Integration and the Future of Smart Agriculture

AI and IoT play central roles in enabling predictive, real-time, and autonomous control over agricultural operations. AI models analyse sensor data to optimise decisions like irrigation

timing, pest control, and crop health monitoring [56]. IoT systems connect field devices such as cameras, moisture sensors, and GPS modules to centralised platforms for remote monitoring and control [13]. Together, AIoT frameworks enhance resilience, reduce inputs, and increase productivity [14].

The evolution of agriculture from traditional practices to the Agriculture 4.0 era highlights the pivotal role of technologies like Wireless Sensor Networks (WSN), the IoT, and AI [3]. This technological shift has enabled real-time decision-making, automation, and data-driven optimisation in agricultural practices [57]. Despite these advancements, smart agriculture faces several challenges, including data quality issues, network connectivity limitations, and storage and processing complexities [4]. Farmers traditionally relied on trial-and-error methods for fertiliser application, irrigation, and pest management, often resulting in suboptimal yields [58]. Integrating AI into farming management systems offers solutions by optimising chemical inputs based on spatial and temporal parameters, thereby improving food quality and reducing waste [59].

Spatial data remains crucial for precision agriculture, influencing machinery operations and decision-making processes. However, the complexities associated with spatial data introduce additional challenges in data processing and interpretation [60]. AI systems must navigate sparse and noisy datasets, increasing the complexity of decision-making frameworks [8].

The COVID-19 pandemic severely impacted agriculture, leading to pest outbreaks and crop diseases that underscored the limitations of traditional farming approaches [6]. Consequently, there has been a paradigm shift toward adopting Industry 4.0 technologies, including AI and IoT, to bolster the resilience of agricultural systems [56]. Post-pandemic, AI solutions are seen as vital for addressing labour shortages and ensuring food supply chain stability. The synergistic relationship between AI and IoT extends to application, implementation, infrastructure, and management, enhancing operational efficiency, providing robust risk management, and improving scalability [13]. The transformative potential of AI in agriculture is substantial, offering opportunities for sustainable and highly efficient farming practices [14].

In conclusion, while significant challenges exist, the integration of AI and IoT in precision agriculture presents transformative opportunities. Addressing data management, connectivity, and cybersecurity concerns will be essential for realising the full benefits of smart agricultural systems. Building upon the connectivity and intelligence offered by AIoT, the incorporation of real-time vision models like YOLO, combined with custom-designed teaching aids, enables robotic arms to perform precise fruit classification and grasping—paving the way for smarter, task-adaptive harvesting systems.

2.1.7 Design and Development of a Peduncle-Holding End Effector for Robotic Harvesting of Mango

The efficient harvesting of fruits such as mangoes requires delicate handling to minimise mechanical injuries, bruising, and sap-induced infections. Ranjan et al. [61] developed a peduncle-holding end effector capable of simultaneously cutting and grasping mango peduncles, thereby preserving fruit integrity and marketability. Their design underwent conceptual evaluation of three configurations, with the final design operating using a single servo motor for both cutting and grasping actions [62].

Addressing the global shortage of agricultural labour and the high costs associated with traditional fruit harvesting methods necessitates advanced technological developments [63]. Robotic harvesting systems have emerged as viable solutions, offering efficiency and scalability [64]. The end effector is critical in these systems, serving as the direct interface between

the crop and the robotic manipulator. Its efficiency directly impacts harvesting cycle time and fruit quality [65].

Effective fruit detection plays an integral role in autonomous harvesting. Xu et al. [24] employed a lightweight YOLOv3 model to detect green mangoes in complex environments, demonstrating that precise object recognition significantly improves harvesting success rates. Similarly, Rong et al. [66] advanced robotic harvesting of watermelons using a vision system based on an improved YOLOv5s model, emphasising the importance of integrated perception systems.

Careful handling during robotic harvesting is crucial to maintain post-harvest fruit quality. Studies have shown that improper handling results in mechanical injuries such as bruising, cuts, and compression damage [17, 18, 19]. Therefore, designing end effectors that minimise direct contact or pressure on the fruit body is essential. Peduncle-holding mechanisms, like the one developed by Ranjan et al., effectively prevent direct fruit contact, minimising damage risks [61].

End effectors that hold fruit stems are particularly well-suited for fruits and vegetables with slim, elongated stems, such as strawberries, cucumbers, and grapes [67]. Hayashi et al. [20] and Feng et al. [68] previously introduced designs featuring synchronised cutting and gripping mechanisms, successfully applied in strawberry and tomato harvesting. These designs provided inspiration for the compact and effective mango end effector.

In conclusion, the development of a peduncle-holding end effector represents a significant step forward in robotic fruit harvesting technologies. By ensuring careful fruit handling and reducing post-harvest losses, such designs can significantly contribute to sustainable agricultural practices and address critical labour shortages. While specialised hardware like peduncle-holding end effectors addresses the physical aspects of crop harvesting, unlocking the full potential of precision agriculture increasingly depends on the seamless integration of intelligent, connected systems—ushering in the era of the Artificial Intelligence of Things (AIoT).

2.1.8 Robot Farming: Transforming Agriculture with Advanced Techniques for Efficient Cultivation, Irrigation and Crop Monitoring

Modern agriculture faces several key challenges, including inadequate chemical application, pest and disease infestation, improper irrigation and drainage, weed control, and inefficient yield forecasting [69]. These challenges have significantly accelerated discussions on the necessity of automating agricultural practices. The integration of robotics and AI into agriculture has led to the emergence of highly innovative robotic farming technologies aimed at addressing these problems [35]. A critical driver for agricultural automation is the projected need to double agricultural productivity by 2050 to meet the demands of a growing global population expected to reach 9 billion [70]. Compounding this is the environmental strain caused by traditional farming, which necessitates a 25% increase in agricultural efficiency with limited resources like land and water [70]. As a response, modern techniques that minimise labour dependency and optimise resource usage are gaining importance [71]. Agricultural robotics represents a rapidly expanding market, projected to reach $11.58 billion by 2025 [72]. This sector combines robotics, machine learning, and expert systems to automate tasks such as crop monitoring, precision irrigation, and harvesting. Robots equipped with AI-driven sensory technologies offer a solution to labour shortages and help ensure sustainable food production by enhancing productivity and reducing costs [73]. Despite these promising advancements, challenges remain in the effective integration and deployment of sophisticated technologies within agricultural systems. Seamless adoption requires not only technological improvements

but also strategic investment in infrastructure and training [74]. Nevertheless, innovations like unmanned aerial vehicles for field mapping [75] and autonomous ground robots for crop monitoring [76] are pushing the frontier of precision agriculture. In conclusion, the integration of AI and robotics into agriculture is revolutionising the sector by offering solutions to long-standing inefficiencies. Robot farming enhances cultivation, irri- gation, and crop monitoring, paving the way for more efficient, sustainable, and productive agricultural systems. While robot farming enhances large-scale field operations, precision tasks like fruit harvesting demand advanced perception capabilities—particularly for navigat- ing complex environments, which brings us to the importance of semantic obstacle avoidance in robotic arm systems.

2.1.9 Semantic Obstacle Avoidance of Robotic Arm for Fruit Harvesting

Fruit harvesting remains one of the most labour-intensive and physically demanding tasks in agriculture, particularly for crops such as apples and strawberries. Tanda [77] highlights that the seasonal nature of harvesting creates a labour demand that often exceeds the available workforce, leading to significant challenges for large-scale farms. Additionally, the repetitive and physically strenuous nature of fruit picking contributes to workforce shortages, making automation a necessary and urgent solution [25]. Several studies underline that harvesting costs are among the highest expenses in fruit agriculture, further motivating the shift toward automated solutions [78]te. Robotic harvesting systems aim to address this by reducing reliance on labour and lowering operational costs [71]. Modern automated harvesters employ vision-based systems, such as those utilising RGB-D cameras like the Intel RealSense D435i [79], combined with deep learning models (e.g., YOLO variants [21]) to detect and localise fruits accurately. Beyond detection, the robotic arm's ability to navigate complex environments without damag- ing fruits or trees is critical. Advanced motion planning and semantic obstacle avoidance strategies, including semantic segmentation techniques [9], allow robotic systems to distinguish between target fruits and hard obstacles such as branches and support structures. Tanda's work leverages convolutional neural networks (CNNs) trained on annotated datasets to achieve high-fidelity environmental mapping and to enable real-time adaptive grasp planning [80]. Furthermore, the Monash Apple Retrieving System demonstrated that machine learning-driven optimisation of fruit-picking strategies could achieve high detection rates with minimal fruit damage [25]. However, challenges remain in adapting to different orchard layouts, variable lighting, and occlusion by foliage [78]. Robust domain adaptation methods and the incorporation of hyperspectral imaging are seen as promising directions to enhance system versatility and accuracy [9]. In conclusion, the automation of fruit harvesting using robotic arms integrated with vision systems and semantic obstacle avoidance is a rapidly advancing field. Solutions like those presented by Tanda [77] show significant promise in replicating human dexterity, reducing labour dependency, and achieving cost-effective and scalable harvesting solutions for future smart farming systems. As we explore solutions like semantic obstacle avoidance to enhance current harvesting operations, it's equally important to look ahead—examining how broader AI applications are shaping the future of agricultural production and processing.

2.1.10 Advancements in Agricultural Ground Robots for Specialty Crops

Over the past decade, there has been a significant increase in interest regarding agricultural robots for speciality crops, largely driven by advancements in computer vision and recognition technologies [15]. These innovations have allowed for more precise and autonomous operations in fields traditionally reliant on intensive manual labour. Speciality crops, such as fruits, vegetables, and nuts, often require specific management practices like trellising and pruning,

adding to the production cost and complexity [81]. Robotics offers a promising solution to alleviate these labour challenges and improve the efficiency and sustainability of speciality crop production [35]. Agricultural robots, also referred to as agribots or unmanned ground vehicles (UGVs), have evolved into sophisticated systems incorporating four core components: vision systems, control systems, mechanical actuators, and mobile platforms [63]. Recent technological advancements have further enhanced these systems with the integration of RGB, multispectral, hyper- spectral cameras, and LiDAR sensors, enabling more accurate plant monitoring and decision-making [82]. These sensory improvements are crucial for precision agriculture applications such as yield estimation, disease detection, and targeted treatment [83]. Barbosa Ju´nior et al. [51] emphasise that robots tailored for speciality crops must accommodate the unique agronomic and environmental challenges posed by these crops, such as variable terrain and plant morphology. Integrating machine learning and deep learning models, such as Mask R-CNN for fruit segmentation [53], has improved robots' ability to recognise and manipulate delicate crops under diverse conditions. Despite the technological progress, limitations persist. The high costs associated with develop- ing and deploying specialised robots, as well as adaptability issues across different crop types and environments, pose barriers to widespread adoption [40]. Nevertheless, the combination of autonomous mobile platforms, advanced sensory equipment, and AI-driven decision systems marks a transformative era for speciality crop farming, potentially leading to higher productivity, reduced labour dependency, and more sustainable agricultural practices [52]. Among the latest innovations driving these advancements in ground robotics is the development of modular and reconfigurable systems like Hefty, designed to push the boundaries of manipulation and adaptability in agricultural environments.

2.2 Industry Landscape of Fruit-Picking Robots

The agricultural automation sector has seen rapid development in fruit-picking robotics. This section surveys both commercial deployments and cutting-edge research prototypes, highlighting their design approaches, capabilities, and current deployment status.

2.2.1 Commercial Systems

- **Abundant Robotics (USA):** Created an automated apple picker platform but ceased operations in 2021, largely due to market and pandemic pressures. The company aimed to reduce labour dependency in orchards.
- **FFRobotics (Israel) – Apple Harvester:** A robotic system designed for selective apple picking, using advanced computer vision to detect and harvest ripe fruit. The developers project speeds up to 10× faster than human pickers, using soft robotic graspers that minimise bruising and stem damage.

Figure 2.1: Apple Harvester in Action

Source: Adapted from Good Fruit Grower.
https://goodfruit.com/automation-advancements-in-apple-harvesting-video/

- **AGROBOT (Spain)** – The Strawberry Harvester uses vision-guided picking with gentle end-effectors tailored for soft fruit harvesting. Focusing on strawberry crops, the system balances autonomy with delicate handling.

Figure 2.2: Strawberry picking in real time

Source: Adapted from the Agribot website.
https://www.agrobot.com/

- Dogtooth Technologies (UK) provides robotic pickers for soft fruits, including strawberries, with integrated detection, picking, and classification modules. Established in 2018, this AI-focused platform emphasises minimal crop damage.

Figure 2.3: Strawberry picking in real time

Source: Adapted from DogTooth Technologies, https://dogtooth.tech/robots/

- **Harvest CROO Robotics (USA):** An autonomous strawberry-harvesting platform deployed across 33 countries. Designed to replace around 30 workers, it can complete the harvest of a 25-acre field within three days.

Figure 2.4: Automated Harvesting in real time

Source: Adapted from ResearchGate, //www.researchgate.net/figure/Harvest-CROO-Harvester-picking-in-a-strawberry-field-in-Florida-USA-Reproduced-with

- **Ripe Robotics (Australia):** Near-commercial readiness for a strawberry-picking robot as of mid-2022, focusing on automation solutions tailored for local farms.
- **OCTINION (Belgium):** Showcased fully autonomous strawberry-picking robots at Fruit Logistica 2019, highlighting modular gripper design and mobile harvesting platforms.

Figure 2.5: Automated Harvesting in real time

Source: Adapted from Octinion Technologies,
http://octinion.com/news/press-release-octinion-presentsworld%E2%80%99s-first-strawberry-picking-robot

2.2.2 Emerging & Specialized Systems

- **Fieldwork Robotics ("Robocrop", UK):** A four-armed, AI-enhanced raspberry-picking robot that mirrors human performance (150–300 berries/hour). Trials are ongoing in the UK, Portugal, and Australia, with plans to commercialise 24 units next year. Its soft, inflatable "pelican-mouth" grippers, paired with spectral ripeness detection, enable efficient harvesting. Trials with Costa Group and other partners are underway.

Figure 2.6: Automated fruit picking

Source: Adapted from Robocrop Technologies,
https://www.theguardian.com/technology/2019/may/26/world-first-fruit-picking-robot-set-to-work-artificial-intelligence-farming

- **Tevel-Tech (Israel):** Aerial Harvester Deploying drone-like platforms with extendable arms to reach high-hanging fruit autonomously, enhancing orchard flexibility and speed.

Figure 2.7: Automated harvesting

Source: Obtained from TevelAerobotics.
https://www.tevel-tech.com/

- **IAV (Germany) Strawberry Berry Robot:** Employs scissors, suction grippers, and image recognition to mimic gentle human plucking of strawberries while reducing physical damage.

Figure 2.8: Automated harvesting Robot

Source: Robotic Harvester from IAV Germany.
https://www.linkedin.com/posts/
iav-gmbhfuture-strawberries-harvest-activity-6993599187162460161--6Lb

- **Bonsai Robotics (USA/Australia):** Recently raised $15M to build autonomous harvesting robots targeting almonds, pistachios, and walnuts; units are now undergoing trials in both the U.S. and Australia.

Figure 2.9: Automated harvesting Robot

Source: Adapted from Bonsai Robotics.
https://www.bonsairobotics.ai/

2.2.3 Market Trends & Challenges

- Market Growth The global harvesting robot market was valued at approximately $1.19 billion in 2024 and is projected to grow to around $1.41 billion in 2025 and $2.73 billion by 2029, reflecting a compound annual growth rate (CAGR) of approximately 18% over the period [84, 85].
- Drivers – Key factors fuelling this growth include widespread labour shortages in agriculture, a demand for precision harvesting in high-value crop segments, and government-funded innovation grants.
- Technical Barriers – Challenges remain in accurate ripeness detection under field conditions, designing low-compression end-effectors for delicate fruit, mobility across uneven terrain, system interoperability, and total cost of ownership.
- Target Crop Types – Current commercial and prototype systems primarily focus on high-value soft and tree fruits such as apples, strawberries, raspberries, blueberries, and nuts.

2.2.4 Academic Research Highlights

Significant university-led efforts demonstrate innovative gripper designs and control methods:

- **Tendon-Driven Soft Gripper for Blackberries:** Achieved 4.8 sec/berry harvesting time, 95% reliability, and precise contact force control (0.5N ±0.046N).

Figure 2.10: Automated harvesting Robot

Source: Adapted from Hero A Berry picker.
https:
//www.asme.org/topics-resources/content/berry-picking-today-robotic-surgery-tomorrow

- **In-Gripper Sensing for Strawberry Picking:** A cable-driven, six-finger gripper with IR sensors and internal storage improved picking speed and positional robustness.
- **Ripeness-Sensing Soft Gripper:** Integrated NIR reflectance sensing for blackberry ripeness detection, deployed on a UR-5 robot arm; field trials showed clear spectral differences between ripe and unripe fruit.

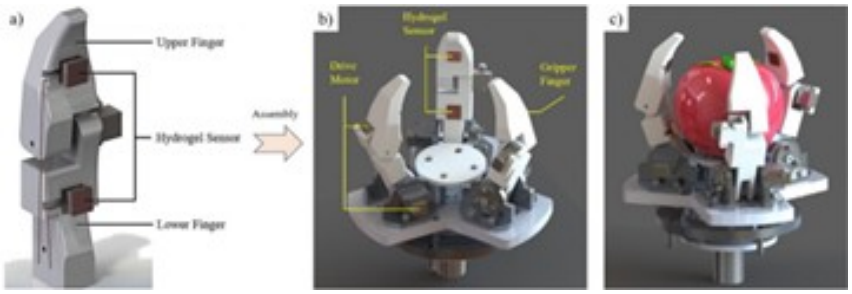

Figure 2.11: Automated berry picking

Source: Adapted from Hero A Berry picker.
https://link.springer.com/article/10.1007/s11694-025-03168-y

- **Interactive Movement Planning (I-ProMP):** A path-planning algorithm that can physically push aside occluding fruit to reach targets in complex clusters within 1˜00ms, suitable for dense Oracle environments.

Commercial fruit-picking robots have advanced rapidly, with several systems now field-tested or deployed in real farms especially for strawberries, raspberries, and apples. These systems combine soft robotics, AI perception, and mobile platforms, aligning strongly with the described architecture of the present project. Emerging innovations include spectral ripeness

detection, aerodynamics-based harvesting, and advanced path planning. However, significant industry-wide challenges persist, including field-level durability, crop-friendly mechanics, and cost-effective scaling. As a result, the field is expanding swiftly: academic prototypes are converging with commercial systems to create viable robotic alternatives to the currently labor-intensive fruit harvesting process supporting the overall relevance and potential impact of your research project.

2.2.5 Comparative Evaluation of This Project with Existing Systems

The fruit-picking system developed in this project shares many core principles with existing commercial and academic solutions while offering a modular, open-design alternative that emphasizes affordability, adaptability, and extensibility. Table 2.1 provides a comparative overview.

This project's use of 3D printing, soft materials, and open-source platforms enables flexible iterations and cost-effective development, contrasting with the high investment required by industrial robots such as those from Dogtooth Technologies or Harvest CROO. While commercial systems focus on maximizing throughput in specific crops, this system introduces a generalized architecture that can be customized for other fruits with minimal design changes.

Additionally, the integration of a YOLOv8-based detection pipeline with a modular, soft end-effector allows for robust object classification while maintaining gentle interaction with produce—an approach that mirrors current academic research but emphasizes real-world implementation in agricultural environments. The Rover's mobility and charging dock concept also introduce a novel system-level vision that could outperform monolithic robots in large-scale field deployment.

Feature	**Existing Systems**	**This Project**
Target Fruit Types	Apples, strawberries, raspberries, nuts	Tomatoes (extendable to other fruits)
End Effector Type	Soft suction grippers, scissors, inflatable jaws	Custom 3D-printed soft TPU gripper with three fingers
Sensing Modality	RGB-D (RealSense, custom vision), spectral NIR, IR sensors	RGB-D Intel RealSense + YOLOv8 object detection
Mobility Platform	Wheeled autonomous platforms or aerial drones	Modular ground rover chassis designed in-house
Manipulator Arm	4–6 DoF industrial arms (UR5e, custom)	5–6 DoF robotic arm (custom design for rover mounting)
Modularity	Mostly fixed-purpose hardware	Modular architecture: arm, gripper, camera, rover independently upgradable
Cost and Scalability	High (proprietary systems, industrial grade)	Low-cost design using 3D printing and open-source hardware/software
Innovation Focus	Speed, commercial readiness, specialized crops	Multi-tool end-effector design, adaptable platform, human-AI collaboration
Stage of Development	Commercial trials or production use	Academic prototype with practical deployment potential

In conclusion, while this project aligns technologically with industry trends, its emphasis on modularity, human-AI collaboration, and affordability positions it uniquely within the research and prototyping landscape, offering promising directions for future agricultural robotics innovations.

Chapter 3

Fundamentals

3.1 Introduction to Vision Systems in Agricultural Robotics

The evolution of agriculture has transitioned from traditional manual labour to semi-automated systems and, more recently, into the domain of fully autonomous robotic solutions. Among the enabling technologies, computer vision stands as a cornerstone in the advancement of smart farming, precision agriculture, and automated crop handling [86, 87]. Vision systems empower robots to perceive their environment, interpret visual data, and make informed decisions based on detected features — such as fruit ripeness, leaf health, or weed presence.

In this context, the integration of RGB and depth cameras, coupled with real-time inference from deep neural networks, allows for tasks like fruit detection, classification, segmentation, and volumetric estimation [88]. These capabilities are central to robotic platforms tasked with precision harvesting, quality control, and resource optimisation.

3.1.1 Basic Principles of Vision-Based Perception

At the core of vision systems lie geometric, optical, and photometric models. A camera maps 3D world coordinates into a 2D image plane using a perspective transformation governed by the *pinhole camera model* [89]:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z}\mathbf{K} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.1}$$

Here, it (X, Y, Z) represents a point in world coordinates and (u, v) is the corresponding pixel location on the image plane. $\mathbf{K}$ Is the intrinsic camera matrix given by:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

Where f_x, f_y are the focal lengths in pixel units, and (c_x, c_y) is the optical centre (principal point)? This mapping is fundamental in 3D reconstruction, depth estimation, and spatial alignment with robotic actuators.

3.1.2 Radiometric Considerations and Sensor Response

The intensity of light captured by a camera sensor is influenced by the radiometric equation, describing how scene radiance is converted into pixel values:

$$I(u,v) = \int_{\lambda} E(\lambda,u,v)R(\lambda)d\lambda \tag{3.3}$$

Where $E(\lambda,u,v)$ is the spectral power distribution of incoming light at wavelength λ, and $R(\lambda)$ is the spectral sensitivity of the sensor [90]. This model is especially relevant in controlled environments or in spectral imaging applications where colour consistency and reflectance properties are crucial for classification tasks.

3.1.3 Depth Estimation with Stereo or RGB-D Cameras

For 3D scene understanding, depth is essential. In stereo vision systems or depth sensors like Intel RealSense, the depth Z of a point is estimated based on disparity d using the triangulation formula [91]:

$$Z = \frac{f \cdot B}{d} \tag{3.4}$$

Where f is the focal length of the camera, B is the baseline distance between the two camera lenses, and d is the disparity between the projections of the same 3D point in the left and right images. Depth measurements enable spatial localisation and are crucial in robotic manipulation and motion planning.

3.1.4 Color Space Transformations

Fruit detection models often operate in alternative colour spaces (e.g., HSV, Lab) to improve robustness against lighting variations. A transformation from RGB to HSV, for instance, decouples chromatic content from brightness and is defined piecewise:

$$H = \begin{cases} 60^\circ \times \frac{G-B}{\Delta} & \text{if } C_{max} = R \\ 60^\circ \times \left(\frac{B-R}{\Delta} + 2\right) & \text{if } C_{max} = G \\ 60^\circ \times \left(\frac{R-G}{\Delta} + 4\right) & \text{if } C_{max} = B \end{cases} \tag{3.5}$$

Where $\Delta = C_{max} - C_{min}$, and C_{max}, C_{min} are the maximum and minimum of the RGB components [90]. Such transformations are foundational for segmentation and thresholding strategies used in fruit detection pipelines.

3.1.5 Relevance to Agricultural Automation

These fundamental equations and principles form the bedrock of all vision-based decisions in robotic agriculture. From pixel-level segmentation of fruits to 3D pose estimation for grasp planning, vision technologies bridge the gap between perception and actuation. Their importance becomes even more pronounced when combined with machine learning and edge computing, enabling real-time, intelligent systems capable of operating in unstructured outdoor environments.

As we delve into the specific components of the vision system — including the training of neural networks, the architecture of YOLO models, the integration with depth sensors, and

the hardware-software interfaces — this foundational understanding will provide the mathematical and conceptual framework required for deeper exploration.

3.2 Software Selection

The software ecosystem chosen for this project was strategically curated to meet the demands of an intelligent fruit detection system, enabling robust performance across dataset management, model training, annotation, and real-time inference. Each tool and platform was evaluated for its compatibility with deep learning pipelines, its ease of integration, and its efficiency in handling computer vision tasks. The selection reflects a balance between open-source flexibility, industry-standard practices, and practical implementation constraints.

Category	Tool	Purpose
Operating System	Ubuntu 22.04 (WSL)	Training environment with better compatibility for Python, PyTorch, and CUDA
Object Detection	YOLOv8s	Lightweight and accurate model for real-time tomato classification and localization
Annotation Tool	Roboflow	Online annotation, augmentation, versioning, and export of datasets
Annotation Tool	LabelImg	Offline manual annotation for datasets not uploaded to Roboflow
Programming Language	Python	Main language used for model development, training scripts, and inference logic
Vision Library	OpenCV	Used for image handling, drawing detections, and capturing real-time webcam feeds
Sensor SDK	Intel RealSense SDK	Enables integration of depth-sensing and RGB video for future deployment

Table 3.1: Software Tools Used in the Project

The software ecosystem chosen for this project was strategically curated to meet the demands of an intelligent fruit detection system, enabling robust performance across dataset management, model training, annotation, and real-time inference. Each tool and platform was evaluated for its compatibility with deep learning pipelines, its ease of integration, and its efficiency in handling computer vision tasks. The selection reflects a balance between open-source flexibility, industry-standard practices, and practical implementation constraints.

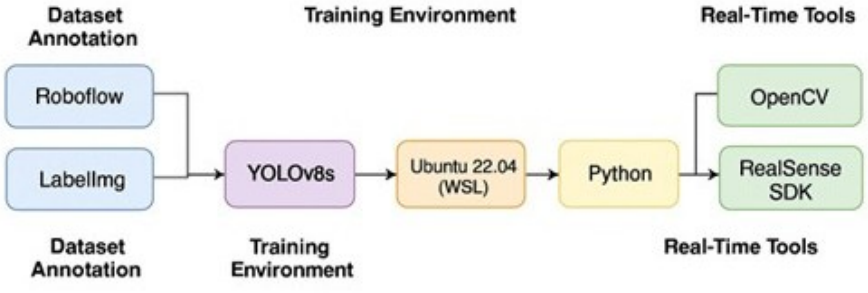

Figure 3.1: Custom figure showing the software stack — Source: Generated with AI

As shown in Figure 3.1, the software stack integrates tools such as Roboflow, YOLOv8, and OpenCV across different stages of deployment.

The selection of software tools and platforms in this project was not arbitrary but guided by a need to balance performance, ease of use, scalability, and community support. Ubuntu 22.04 ensured a stable and resource-efficient environment for training deep learning models. YOLOv8s provided the necessary accuracy and speed for detecting ripe and unripe tomatoes. Roboflow and LabelImg together enabled flexible annotation strategies for different dataset conditions. Python acted as the glue connecting all the components, offering programmable control and logic for both offline and real-time phases. OpenCV supported image processing and real-time visualisation, while the RealSense SDK enabled future-proofing through depth-enabled capabilities.

- **Ubuntu 22.04 via Windows Subsystem for Linux (WSL)** To ensure a stable and efficient development environment for machine learning training, Ubuntu 22.04 Long-Term Support (LTS) was used through Windows Subsystem for Linux (WSL). While the primary operating system remained Windows for general-purpose use and file handling, Ubuntu offered a Linux-native command-line interface and environment optimised for Python-based AI workflows. Using Ubuntu through WSL enabled seamless access to popular frameworks such as PyTorch and allowed the project to benefit from Linux-specific optimisations and community support. Ubuntu 22.04, in particular, was selected due to its reliability, active security patching, and excellent support for NVIDIA GPU drivers and CUDA, both of which are essential for accelerating neural network training.[92]
- **YOLOv8s (You Only Look Once, version 8 small)** YOLOv8 is the latest iteration in the YOLO family of real-time object detectors, designed and maintained by Ultralytics. This project began with the YOLOv8n (nano) model to prototype the pipeline and test detection accuracy on limited datasets. However, the nano model was found to be insufficient for the complexity of real-world tomato images, which often include shadows, overlaps, and varying maturity stages. Consequently, the YOLOv8s (small) variant was adopted. YOLOv8s is a lightweight yet powerful model that strikes a balance between computational efficiency and detection accuracy. It supports modern deep learning features like anchor-free detection, auto-learning bounding box dimensions, and improved non-max suppression (NMS) techniques. Its compatibility with the Ultralytics CLI and Python API simplified both training and inference. YOLOv8s became the foundation for all subsequent detection tests and real-time deployment plans. [93, 94]
- **Roboflow** Roboflow was an integral platform for preparing and managing datasets. It provided an intuitive web-based interface to annotate, visualise, and version image datasets. For this project, Roboflow was used to draw bounding boxes around ripe and unripe tomatoes, assign class labels, and export datasets in YOLOv8-compatible formats. In addition, the platform's augmentation pipeline allowed the creation of synthetic variations of images through operations such as rotation, flipping, blurring, and exposure changes. These augmentations increased dataset diversity and helped the model generalise better to unseen scenarios. Roboflow also provided powerful dataset analytics and export controls, making it possible to tailor the dataset structure exactly to YOLOv8's training requirements. [95]
- **LabelImg** LabelImg was also utilised to annotate datasets that were handled offline or created independently of Roboflow. It is a lightweight, open-source graphical image annotation tool written in Python and Qt, enabling manual creation of bounding boxes and assignment of labels. The simplicity of its interface makes it an ideal tool for individual

image annotation tasks, especially when working in constrained environments or when Roboflow's web interface is inaccessible. The annotations produced by LabelImg were saved in YOLO-compatible text files, which allowed easy integration into the training workflow. By using both Roboflow and LabelImg, the project maintained flexibility in data annotation while accommodating both online and offline workflows. [96]

- **Python** Python served as the primary programming language throughout the development lifecycle of the project. Its ease of syntax, vast array of machine learning libraries, and seamless integration with frameworks such as PyTorch and Ultralytics' YOLO API made it the ideal choice for building the system's logic. Python was used to manage dataset preprocessing, automate training scripts, visualise results, and control inference pipelines. Furthermore, Python's ability to interface with camera modules, real-time display windows (via OpenCV), and external SDKs such as Intel RealSense contributed significantly to the project's versatility. [97]
- **Intel RealSense SDK** To enable real-time deployment of the trained object detection model, the Intel RealSense SDK was integrated into the system. The SDK provides low-latency access to RGB and depth streams from RealSense cameras and includes utilities for camera calibration, depth estimation, and spatial measurements. Although the project currently uses the laptop's integrated camera for initial tests, the RealSense SDK was included in anticipation of deploying the system on hardware with depth sensing capabilities. By incorporating this SDK, the system can eventually evolve to include 3D scene understanding, which could improve harvesting accuracy, obstacle avoidance, and fruit localisation in real-world agricultural settings. [98]
- **OpenCV** OpenCV (Open Source Computer Vision Library) was utilised for several preprocessing and visualisation tasks. It served as the backbone for loading and displaying images, drawing bounding boxes, capturing frames from live camera feeds, and applying various image transformations. During real-time inference, OpenCV was instrumental in converting model outputs into visual overlays, allowing users to verify detections instantly. The library also supports numerous file formats, image manipulation techniques, and device integrations, making it an essential companion for deploying YOLO-based computer vision systems. [99]

  Together, these tools formed a modular and scalable framework tailored for a vision-based smart harvesting system. The software stack follows industry best practices and leverages accessible technologies, making it suitable for further exploration by students and practitioners.

3.3 Hardware Selection

The hardware selection for this project was guided by functional requirements, environmental suitability for agricultural fields, and integration compatibility with the robotic vision system. Each component was carefully chosen to optimise the performance of the autonomous fruit harvesting robot while maintaining cost efficiency, modularity, and reliability.

Intel RealSense Depth Camera

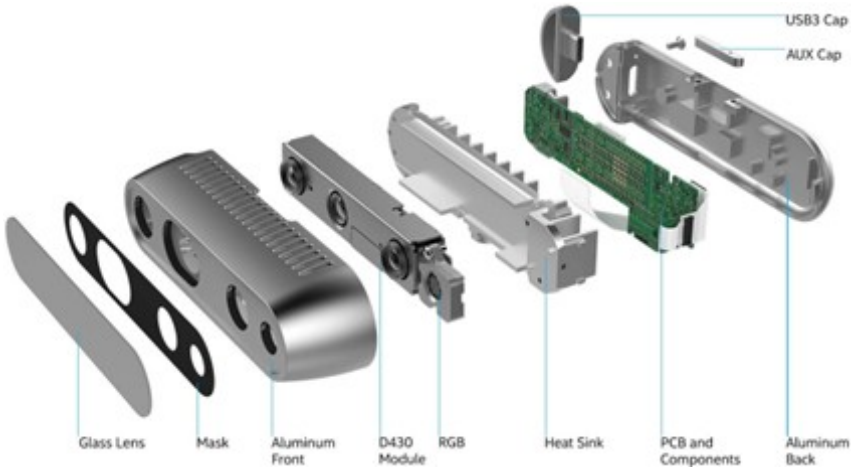

Figure 3.2: Exploded view of the RealSense camera parts

Source: This figure is based on Intel's exploded-view diagram for the D430 depth module. https://medium.com/99p-labs/navigating-depth-perception-f9d4874dd88f

For the vision and perception module, the Intel RealSense camera was selected as the primary sensing device. This choice was driven by its ability to simultaneously capture both RGB and depth (3D) data streams. RealSense cameras utilise stereo vision technology and infrared projection to generate high-resolution depth maps, which are essential for understanding the position and shape of objects in three-dimensional space.

One of the key advantages of using RealSense over conventional RGB webcams lies in its ability to provide accurate distance measurements. This is particularly useful in agricultural applications such as fruit picking, where understanding the spatial relationship between the robotic arm and the fruit enables collision avoidance, trajectory planning, and precise grasp point detection. Moreover, RealSense cameras come with a well-supported Software Development Kit (SDK) that integrates seamlessly with Python, OpenCV, and deep learning frameworks, thereby simplifying the development pipeline.

Furthermore, the compact form factor of the RealSense camera, the plug-and-play USB interface, and the real-time streaming capability make it a practical solution for embedded and mobile robotic platforms. Its versatility and ease of integration make it highly suitable for use in precision agriculture, especially in scenarios that involve close-range object detection, depth-based segmentation, and navigation through unstructured environments. [100, 101]

Actuators and Servos

Standard high-torque servo motors (e.g., MG996R or similar) were selected to power the robotic fingers and joints of the arm. These servos provide the required torque and responsiveness for smooth finger articulation and are readily available and cost-effective for prototyping and testing purposes.

The MG996R is a metal-gear, position-controlled hobby servo that operates on pulse-width modulation (PWM) signals, typically at 50Hz. It receives a pulse between 1 ms and 2 ms to command an angular position within its 0–180° range. The onboard potentiometer and control circuit create a closed feedback loop that adjusts the PWM until the motor shaft matches the desired target angle. The servo is capable of generating up to 11 kg·cm of stall torque at 6 V, meaning it can exert a torque τ given by,

$$\tau = F \times r$$

where F the force is in newtons and r the lever arm—thus, 11 kg·cm corresponds to 1.08 N·m. The no-load speed is approximately 0.15 s/60° at 6 V, characterising its dynamic response. Such torque and speed characteristics make the MG996R suitable for actuation in small robotic arms and gripper mechanisms. [102, 103]

3D Printing Materials

For the physical construction of the gripper and brackets, 3D printing was utilised due to its rapid prototyping capabilities and low production cost. Flexible PLA or TPU was used for the end effector fingers, ensuring gentle interaction with the fruit. Rigid PLA or PETG was used for structural components such as motor brackets and joints. The choice of materials was guided by their mechanical properties, including tensile strength, flexibility, and layer adhesion. The details of this are mentioned precisely in chapter 6.

UR5e Robotic Arm Integration

The robotic end effector developed in this project was designed to be mounted on the UR5e robotic arm, an advanced collaborative robot (cobot) developed by Universal Robots. The UR5e offers 6 degrees of freedom and is widely used in academic and industrial automation tasks due to its high repeatability, intuitive programming interface, and flexible payload handling.

Key Specifications of UR5e:

- Degrees of Freedom: 6
- Reach: 850 mm
- Payload Capacity: Up to 5 kg
- Repeatability: ± 0.03 mm
- Mounting Interface: ISO-9409-1-50-4-M6 flange
- Control Interface: Integrated teach pendant with PolyScope software

The modular gripper was attached to the UR5e's tool flange via a custom-designed base plate that conforms to the ISO standard mounting pattern. This enabled seamless mechanical integration and ensured that the gripper assembly could be securely mounted and removed as needed for testing and adjustment.

Component	Description and Rationale
End Effector	3D-printed, three-fingered design using TPU for soft gripping
Manipulator Arm	5–6 DoF robotic arm; servo-actuated for flexibility and reach
Camera	Intel RealSense RGB-D for fruit detection and spatial localization
Servos	MG996R or equivalent; chosen for torque and availability
Materials	TPU and PLA used for soft-touch and structural integrity respectively
Rover Platform	Modular chassis with terrain adaptability and docking features
Power Unit	Battery-based system with microcontroller coordination

Table 3.2: Summary of hardware components selected for the robotic harvesting system

3D Printed Components

Figure 3.3 shows the 3D-printed components fabricated using PLA (Polylactic Acid), a popular biodegradable thermoplastic known for its ease of printing, dimensional stability, and good surface finish. Each component was printed individually to ensure part quality and allow detailed post-processing. The functions of the individual components are as follows:

- **Driving Screw (Top-Left)**: This vertically oriented threaded component translates rotational input from the MG996R servo into vertical motion. It interfaces directly with the gear teeth of the finger system. To preserve thread accuracy, it was printed vertically with a 0.12 mm layer height and 100% infill for strength.
- **Finger + Gear Assembly (Middle-Left)**: This is the core actuation unit, consisting of three flexible gripping fingers and two internal gears embedded in the central cylindrical housing. Each finger includes a hinge structure for smooth pivoting. The entire unit was printed with supports enabled under the finger arms and a wall line count of 3 to ensure outer rigidity.
- **Small Circular Servo Horn Adapter (Top-Center)**: This adapter connects the servo motor shaft to the driving screw. It includes a central bore and indexing holes for set screws or bolts. Precision was critical; hence, it was printed at 0.08 mm layer height with 100% infill to ensure spline engagement and torque transfer.
- **L-shaped Motor Mount (Top-Right)**: Designed to cradle the MG996R motor, this bracket was printed lying on its back to avoid overhangs and ensure dimensional accuracy of the motor slot. It was printed with 40% infill and 4 walls for robust support during operation.
- **UR5e Base Plate (Bottom-Center)**: The circular plate is the interface between the gripper and the UR5e arm. It features six mounting holes and was printed flat on the bed to maintain dimensional accuracy. A 20% infill was used with brims enabled to avoid warping.
- **Camera Bracket Base (Bottom-Right)**: This is the long rectangular arm used to attach the Intel RealSense camera. It includes through-holes on either end for mounting and was printed horizontally with supports under the overhangs and 25% infill.

- **Camera Pivot Mount (Bottom-Center-Right)**: This smaller bracket contains the revolute joint interface, allowing vertical tilt of the camera. Printed upright with high infill and slow speed to preserve the curved joint hole and ensure tight tolerances.

Figure 3.3: 3D printed components fabricated with PLA for the end-effector system.

All components were printed using black PLA filament with a nozzle temperature of 200–210°C and bed temperature of 60°C. Layer height varied between 0.08–0.2 mm depending on required detail. Cooling fans were kept on for parts with fine bridging (like finger hinges), while slower speeds and extra perimeters were used for mechanical parts like the gear and horn adapter to reduce layer delamination and improve strength. Post-processing included light sanding, support removal, and dimensional checking using calipers before assembly.

Chapter 4

Concept and Methodology

4.1 Methodology

Before diving into the detailed methodology, it is important to outline the general approach adopted in this project. The methodology was structured to combine both hardware and software components in a modular and iterative manner. It began with the identification of real-world challenges in precision agriculture, followed by the design and development of a vision-guided robotic system. Each phase from data collection and model training to mechanical design and integration was carefully validated to ensure accuracy, adaptability, and practical deployability in autonomous tomato harvesting applications.

4.1.1 System Architecture

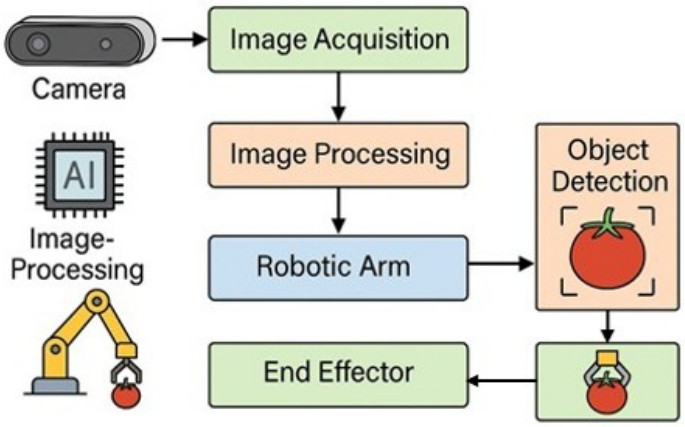

Figure 4.1: Pipeline of System Architecture — Source: AI generated

The structure of the proposed robotic fruit harvesting system, as illustrated in Figure 4.1, follows a modular and sequential pipeline architecture. The system begins with an image acquisition module, where real-time visual data is captured using an RGB-D or RealSense camera mounted on the robot. This raw image data is then passed to the computer vision unit, which utilises a lightweight and real-time object detection algorithm, such as YOLOv8s, to identify and localize target fruits within the environment. Detected fruit coordinates and classification outputs are forwarded to the decision-making and control layer, which interprets the data and calculates the optimal harvesting trajectory. This information is sent to the robotic

arm control system, which activates the motion planning module to align the end-effector with the target fruit. At the end of the pipeline is the end-effector system, which in this project consists of a soft three-fingered gripper capable of grasping and cutting fruit peduncles simultaneously. This soft actuator ensures minimal mechanical stress on the fruit surface, thereby preserving its integrity and reducing post-harvest losses. The entire operation is monitored and coordinated through a centralised AIoT-based system controller, which allows for real time communication, adaptive control, and remote operation.

Together, these subsystems form a cohesive smart agricultural solution that aligns with the principles of Agriculture 4.0 — combining AI, robotics, and IoT to address labour shortages, improve harvesting accuracy, and promote sustainability.

Study	Type of End-Effector	Key Features
Xu et al. (2022)	Grasp-and-separate (citrus)	Blade-assisted finger grip, 95.23% success rate
Zhou et al. (2022)	Elastic fluid-driven (tomato)	Low-pressure inflation, soft contact
Xu Wang et al. (2025)	Fluid-driven soft fingers (kiwifruit)	Compact design, high adaptability
Hayashi et al. (2010)	Clamp-based fingers (strawberries)	Simultaneous grip and cut mechanism

Table 4.1: Comparison of Robotic End-Effector Designs from Recent Studies

4.1.2 Why YOLOv8 Was Chosen for Tomato Detection

YOLOv8 is the latest iteration in the YOLO series, developed by Ultralytics in early 2023. It is a Python-based, PyTorch-integrated deep learning object detector that emphasises being lightweight, fast, and highly accurate, with a unified API supporting training, validation, inference, and export [104, 105, 94]. Object detection—a core computer vision task—aims to locate and classify objects within an image. In this project, the objective is to detect ripe and unripe tomatoes in real-world agricultural images, leveraging YOLO's real-time performance. Among its family, YOLOv8 was chosen for its improved speed–accuracy balance, anchor-free head, and ease of deployment compared to earlier versions [105, 94]

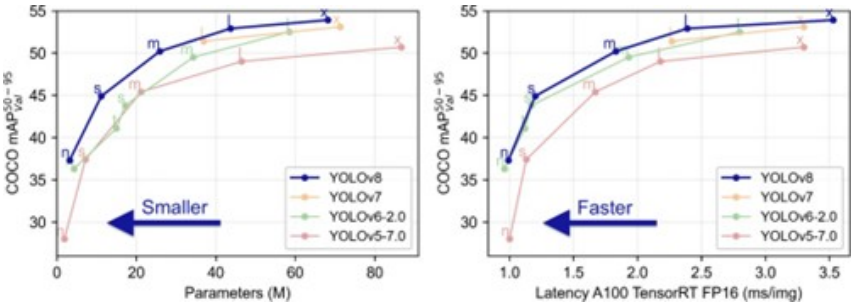

Figure 4.2: Yolo models object detection curves

Source:
https://learnopencv.com/ultralytics-yolov8/

YOLOv8 offers a modular and simplified architecture that performs three tasks simultane-

ously:

1. **Object localisation:** Identifies where each object is in the image.
2. **Classification:** Labels the detected object (e.g., ripe or unripe).
3. **Bounding Box Refinement:** Refines predictions to be closer to object edges.

Key Reasons for Choosing YOLOv8:

The selection of YOLOv8 was based on multiple evaluation criteria, summarised in Table 3.2.

Criterion	YOLOv5	YOLOv7	YOLOv8
Accuracy on small datasets	High	Very High	**Very High + Improved Precision**
Inference speed	Fast	Fast	**Faster (with optimized layers)**
Ease of use / API design	Moderate	Low	**High (Unified API)**
Built-in augmentation support	Manual setup	Manual setup	**Integrated**
Multi-format export (ONNX, TFLite)	Partial	Partial	**Yes (out-of-the-box)**
Support and documentation	Good	Limited	**Excellent (Ultralytics maintained)**

Table 4.2: Comparison of YOLO versions based on key deployment and training features

Benefits in the Context of This Project

The detection of tomatoes in natural settings is inherently challenging due to:

- Varying lighting conditions (sunlight, shadows)
- Occlusion by leaves or stems
- Similar color features between unripe tomatoes and surrounding vegetation

  YOLOv8 was particularly well-suited for this application because:

- It excels in detecting small and partially occluded objects, such as tomatoes in cluttered backgrounds. supports advanced data augmentation natively, which helped improve the generalisation of the model using tools like Roboflow.
- It allows for rapid experimentation through a simple and high-level Python interface, which enabled multiple iterations with different configurations and datasets.
- It has built-in support for transfer learning (using pretrained weights), making it efficient to retrain the model with new hard samples from prediction failures.

YOLOv8 comes in multiple model sizes to suit different hardware constraints and accuracy needs. In this project, the following variants were tested:

- **YOLOv8n (Nano)**: Used for initial experimentation due to its lightweight nature. This helped validate the data pipeline without overloading hardware.
- **YOLOv8s (Small)**: Selected for final training as it provided a strong balance between accuracy and inference time. It proved to be suitable for potential real-time use on mobile or embedded devices.

4.1.3 Dataset

A reliable dataset is the foundation of any successful machine learning model. In this project, the dataset used for training and evaluating the tomato detection model was constructed

through a combination of publicly available sources, manual annotation, data augmentation, and failure-driven retraining. The dataset was designed to represent real-world agricultural conditions with sufficient diversity to support a robust object detection model.

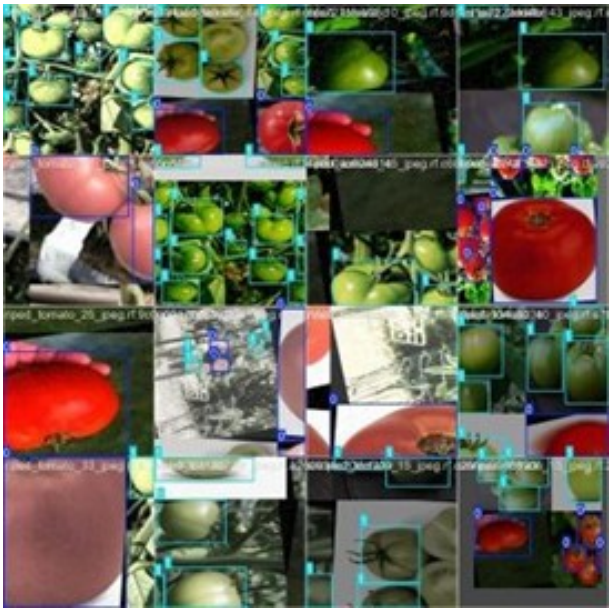

Figure 4.3: Example of dataset used — Source: Custom collage of images made for model training

The dataset was created using images from the following sources:

- **Kaggle Tomato Dataset**: A publicly available dataset containing labelled images of ripe and unripe tomatoes in natural conditions. These images varied in lighting, orientation, and background complexity. [1]
- **Manual Annotation**: Using the LabelImg tool, additional images were manually labelled. Each tomato instance was annotated with bounding boxes and assigned a class label:
  - 0: Unripe
  - 1: Ripe
- **Failure-Driven Dataset Expansion**: After running initial predictions, a selection of 80 images that the model failed to detect properly were identified. These images, along with a few additional examples from the earlier dataset, were manually annotated and added to the training set to improve robustness and handle edge cases.
- **Roboflow Augmentation**: The dataset was uploaded to Roboflow, a web-based platform for data augmentation and preprocessing. Here, techniques such as brightness adjustment, cropping, rotation, and mosaic tiling were applied to increase variability and reduce overfitting.

In total, the combination of manually labelled and augmented images ensured that the model was exposed to a variety of tomato appearances. This is crucial in agriculture, where environmental conditions cannot be controlled — for instance, some tomatoes might be photographed under harsh sunlight, while others may be captured during overcast conditions.

[1] https://www.kaggle.com/datasets/lakshmi25npathi/tomato-disease-classification

Using synthetic variation through augmentation mimics such environmental unpredictability and enables the model to develop visual resilience.

Furthermore, by combining both high-quality and noisy (low-light, blurry) images, the dataset reflected a more balanced representation of field conditions, which are often far from perfect. This approach also helps bridge the gap between research models and deployable models that can be used in actual robotic systems or on mobile devices.

After preprocessing and augmentation, the data were organised into three distinct subsets, following standard machine learning practices:

- **Training set**: 957 images used for training the model weights.
- **Validation set**: 91 images used during training to evaluate model performance and prevent overfitting.
- **Test set**: 46 images used exclusively after training for independent performance evaluation.

The directory structure followed the standard YOLOv8 format:

```
Tomato_detection/
 train/
    images/
    labels/
 valid/
    images/
    labels/
 test/
    images/
    labels/
 data.yaml
```

Each split serves a different purpose:

- **Training set**: This is where the model learns. It sees these images repeatedly and updates its internal parameters (called weights) to improve its ability to identify tomatoes.
- **Validation set**: Used like a progress report during training. After each round (epoch), the model is tested on this set to evaluate its ability to generalise rather than simply memorise the training data.
- **Test set**: Used only after training is complete. This is equivalent to a final exam — it evaluates how well the model can handle entirely new, unseen images.

Care was also taken to avoid overlap between the sets. For instance, if an image of a tomato plant appeared in the training set, no similar images from the same plant were included in the validation or test sets. This helps prevent data leakage and ensures realistic evaluation results.

The table included below contains details about the different datasets used to train the object detection model. Three distinct datasets, each with varying environmental conditions and backgrounds, were incorporated. These datasets were randomly named and were chosen to maximise the model's robustness during testing and real-world deployment.

Dataset	RIPE	UNRIPE	Total Annotations
Dataset 2	314	375	689
Tomato Detection	1591	3290	4881
YOLODataset	661	1396	2057
Total	**2566**	**5061**	**7627**

Table 4.3: Distribution of Ripe and Unripe Tomato Annotations Across Datasets

The `data.yaml` file defined the class names and paths to each data split, ensuring that the YOLOv8 framework could correctly locate and interpret the dataset structure during training and inference.

Challenges and Considerations

Developing a high-quality dataset for tomato detection posed several challenges:

- **Visual Similarity**: Unripe tomatoes often shared visual features with leaves and stems, making them difficult to distinguish.
- **Lighting Variation**: Outdoor images captured under natural light varied significantly in brightness, shadows, and contrast.
- **Occlusion**: Many tomatoes were partially hidden behind foliage, requiring careful annotation and more training examples.

Another key challenge involved maintaining consistency in the annotation process. Since all bounding boxes were created manually, there was a risk of human error — for example, drawing boxes that were too large or too tight, or skipping tomatoes that were partially hidden. Inconsistent annotations can confuse the model, resulting in poor performance even on clean data. To minimise this, annotation guidelines were established and reviewed regularly to ensure uniform labelling standards.

Additionally, class imbalance was considered. While ripe tomatoes were often easier to spot and annotate, unripe tomatoes sometimes blended into the green foliage and were underrepresented in certain image groups. This imbalance was partially addressed by including more examples from underperforming cases identified during prediction failure analysis and by applying stronger augmentation to unripe tomato instances using Roboflow.

Over the course of several iterations, the dataset evolved into a high-quality learning source — not only comprehensive in terms of content but also aligned with the goal of developing a real-time detection system deployable in uncertain agricultural conditions.

To address these issues, annotation quality was prioritised, and the dataset was iteratively expanded and refined based on model feedback. This approach helped ensure the trained model would perform reliably in real-world deployment scenarios, particularly in complex agricultural environments.

4.2 Arduino Code for Servo-Based Gripper Control

The following code demonstrates a basic implementation to control a servo motor for actuating the gripper fingers using an Arduino. This setup allows the servo to rotate between defined open and closed positions to simulate gripping and releasing motions.

```
#include <Servo.h>

Servo gripperServo;           // Create servo object
const int servoPin = 9;       // PWM pin connected to the servo
const int openAngle = 30;     // Angle for open position
const int closeAngle = 120;   // Angle for closed position

void setup() {
  gripperServo.attach(servoPin);  // Attach the servo to the defined pin
  Serial.begin(9600);
  Serial.println("Gripper control started.");
}

void loop() {
  // Close the gripper
  gripperServo.write(closeAngle);
  Serial.println("Gripper closed.");
  delay(2000);  // Hold position for 2 seconds

  // Open the gripper
  gripperServo.write(openAngle);
  Serial.println("Gripper opened.");
  delay(2000);  // Hold position for 2 seconds
}
```

This code uses the `Servo.h` library to manage the PWM signal. The gripper finger actuation is simulated by alternating between two angle positions at a fixed time interval. These values can be calibrated further based on the actual mechanical limits of the gripper.

4.2.1 Important Considerations for Servo Motor Setup

While the Arduino code provided in Listing 4.2 offers a functional example for actuating a gripper mechanism using a servo motor, several practical factors must be addressed to ensure safe and reliable operation. The following points outline both the essential and recommended considerations for implementing the setup in a real-world application.

Essential Requirements

- **External Power Supply:** High-torque servo motors like the MG996R require a current of up to 2.5A during stall conditions. It is strongly advised **not to power the servo directly from the Arduino's 5V pin**. Instead, use a dedicated 5–7V power source capable of supplying sufficient current (e.g., 5V/3A adapter or battery pack).
- **Common Ground:** When using an external power supply, ensure that the ground (GND) of the Arduino is connected to the ground of the servo power source to establish a common reference point.
- **Angle Calibration:** The defined `openAngle` and `closeAngle` values should be adjusted based on the mechanical limits of the gripper design to prevent over-extension or collision with mechanical stops.

- **Avoid Over-Driving:** Do not exceed the servo's operating range (typically 0–180) to prevent motor burnout and gear damage.

Optional Enhancements

- **Sensor Integration:** Incorporate distance or force sensors (e.g., ultrasonic sensor, force-sensitive resistors) to automate gripping based on object detection or applied force.
- **Manual Control Interface:** Add a user input device such as a push button, potentiometer, or joystick for interactive control.
- **Debounce Logic:** If using buttons or switches, implement software debouncing to prevent false triggers due to signal noise.
- **Interrupt-Driven Control:** Replace the `delay()` functions with interrupt-based or timer-controlled logic to improve responsiveness and allow multi-tasking.
- **State Machine or PID Control:** For smoother and adaptive servo behavior, consider implementing a finite state machine or PID controller that adjusts angles based on real-time feedback.

By addressing these considerations, the servo-actuated gripper system becomes more robust, safer for prolonged use, and suitable for integration into dynamic robotic environments.

4.2.2 Wiring Diagram

Figure 4.4 shows the detailed wiring layout of the end-effector module. All power and control lines are routed to the microcontroller, ensuring stable operation and minimal interference.

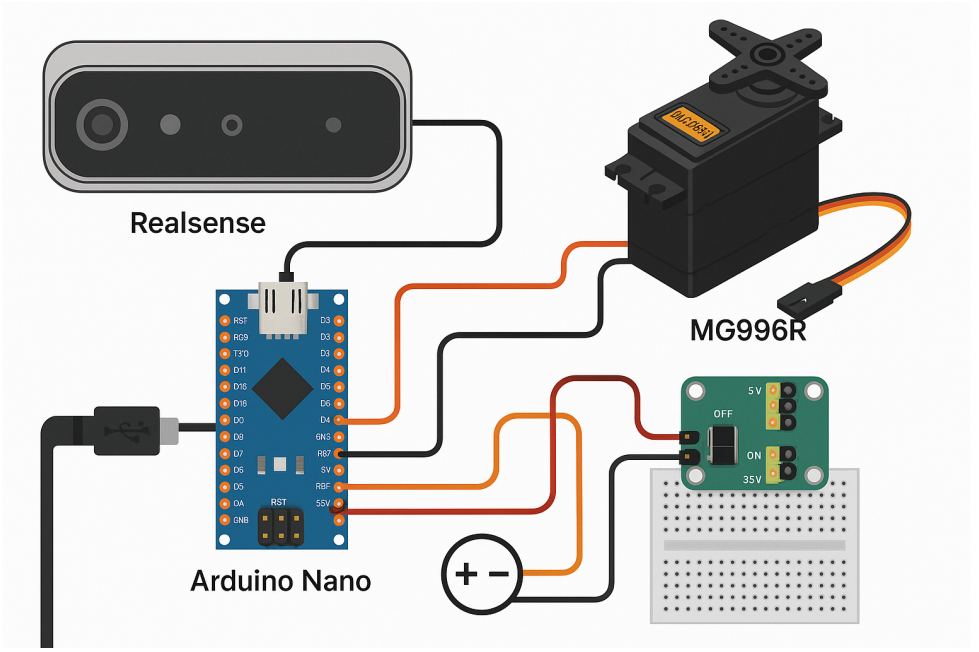

Figure 4.4: Schematic wiring Diagram of Assembly, Source: AI Generated

This section outlines each component used in the schematic diagram shown in Figure 4.5, along with their purpose and function in the overall circuit design.

4.2.3 Arduino Nano

The **Arduino Nano** is a compact microcontroller board based on the ATmega328P. It serves as the control unit for the system, handling logic, PWM signal generation, and data communication.

- **5V and GND Pins:** Provide power to peripherals like the MG996R servo and logic level communication lines.
- **PWM Pin (D9 or D10):** Outputs the control signal for servo actuation.
- **TX/RX Pins:** Used for serial communication; can interface with external devices like the RealSense camera via USB if needed.

4.2.4 MG996R Servo Motor

The **MG996R** is a high-torque servo motor commonly used in robotics applications. It requires a stable 5V power supply and a PWM signal from the microcontroller.

- **VCC (Red):** Connected to 5V regulated supply.
- **GND (Brown/Black):** Connected to system ground.
- **Signal (Orange/Yellow):** Receives PWM control signal from Arduino.

4.2.5 Intel RealSense Depth Camera

The **Intel RealSense** camera provides real-time 3D depth sensing and is used here to collect spatial data for object detection or navigation.

- **Power:** Typically powered via 5V USB, but a 3.3V line is shown as an alternate logic power option in the schematic.
- **Communication:** Data is sent via USB to the host device or can be routed through the microcontroller (depending on architecture).

4.2.6 Buck Converters

Buck converters are step-down voltage regulators that convert higher input voltages to stable output levels suitable for sensitive components.

- **12V to 5V Buck Converter:** Supplies 5V to both the Arduino Nano and MG996R servo motor.
- **5V to 3.3V Buck Converter (optional):** Steps down the 5V line to 3.3V, if the RealSense camera requires logic-level power instead of USB.

4.2.7 12V Power Supply

The system is powered by a **12V DC source**, which could be a battery pack or external adapter. This voltage is regulated via buck converters to meet the operational needs of the servo motor, microcontroller, and sensors.

4.2.8 Ground (GND)

A common **ground** reference is essential to ensure all components share the same electrical baseline. All GND lines from components and power sources are interconnected.

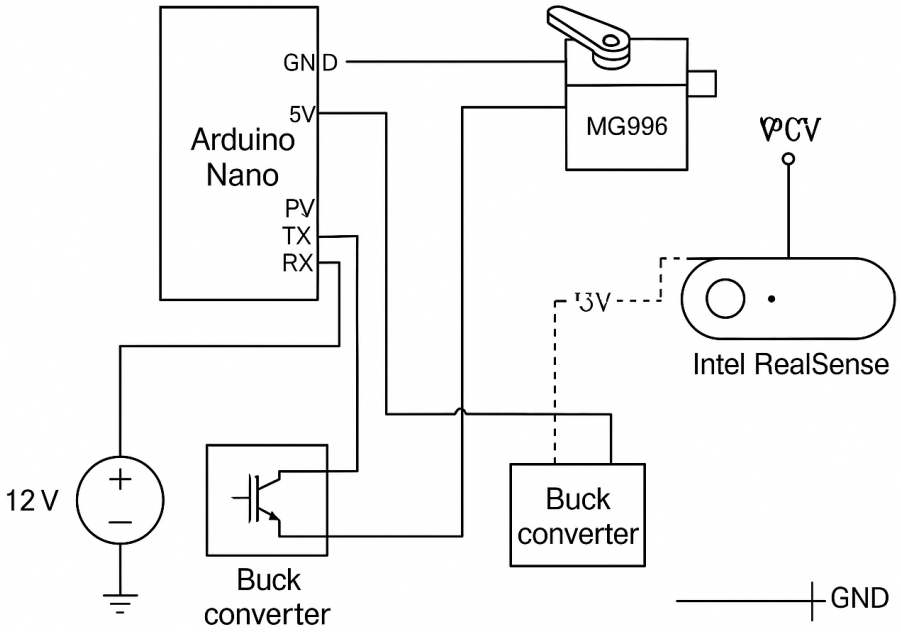

Figure 4.5: Formal Wiring Schematic of the End-Effector Control System, Source: AI Generated

4.3 V-Model Integration

The V-Model (Verification and Validation Model) is a well-established development framework used in systems engineering to ensure rigorous and traceable project execution. It emphasises a direct correspondence between design phases and their associated validation stages. In projects where hardware and software are co-developed—such as in mechatronic and robotic systems—the V-Model offers a systematic structure that supports reliability, modular testing, and clear documentation. [106]

4.3.1 Relevance of the V-Model to This Project

The V-Model was selected for this project because it aligns with the needs of a multidisciplinary development process involving:

- Vision, control, and mechanical subsystems
- Precision-critical operations such as robotic gripping and fruit detection
- Hardware-in-the-loop testing and phased system integration

Unlike agile or purely iterative approaches, the V-Model ensures each design step is verified by a corresponding test, making it ideal for a project requiring physical implementation and validation of a vision-guided robotic system.

4.3.2 Custom V-Model Mapping for the Project

The following table shows how the various stages of the V-Model align with the key development activities of this project. The bottom two phases of the right-hand side (system and acceptance testing) have not yet been executed and are listed as *planned future work*.

Table 4.4: Mapping of V-Model Phases to Project Activities

Phase Type	Design and Specification	Testing and Validation
Requirement Analysis	Detect ripe fruits, grip delicately, modular design, energy-efficient	**Unit Testing:** YOLOv8 model tested on a dataset; MG996R servo tested with PWM signals
System Design	Define key system blocks: vision pipeline, control module, actuator interface	**Integration Testing:** RealSense → YOLOv8 → Arduino → Servo integration tested
Architectural Design	Sensor-to-model-to-actuator communication pipeline	*System Testing (future work):* Full prototype to test gripping performance and vision-guided harvesting
Module Design	YOLOv8 model, servo interface logic, 3D gripper design	*Acceptance Testing (future work):* Final evaluation for performance, energy consumption, accuracy

4.3.3 Detailed Project Requirements

The initial step in the V-Model involves a comprehensive analysis of system requirements, which serve as the foundation for all downstream development and validation activities. For this project, the goal was to develop a modular robotic system capable of performing vision-guided fruit harvesting, specifically targeting ripe tomatoes. The system combines elements of mechanical design, machine vision, embedded control, and real-time actuation.

Functional Requirements

- **Ripe Fruit Detection:** The system must be able to identify ripe tomatoes from real-world imagery using a trained deep learning model.
- **Autonomous Actuation:** Upon detection, the robotic arm must respond autonomously by actuating the end-effector to initiate a grasping action.
- **Real-time Processing:** The detection, decision, and actuation loop should operate in near-real-time to enable field applicability.
- **Gripping Mechanism:** The end-effector must securely grip tomatoes without causing damage. It should accommodate small variations in size and orientation.
- **Modular Software-Hardware Architecture:** Each subsystem (vision, control, actuation) must be developed independently and remain modular for testing and future upgrades.

Non-Functional Requirements

- **Energy Efficiency:** The system should be designed to minimize power consumption, particularly during idle states or between detection cycles.
- **Scalability:** The software pipeline should support easy retraining or fine-tuning to adapt to different fruits or new datasets.
- **Safety and Robustness:** The actuation logic must prevent unintended motion and avoid damaging surrounding plants or objects.
- **Low-latency Communication:** The delay between image acquisition, model inference, and servo response must be minimal.

- **Mechanical Printability:** The gripper components should be 3D printable using soft or semi-flexible materials to enable compliance during gripping.

Hardware Requirements

- **Camera Module:** Intel RealSense depth camera capable of RGB-D data acquisition for spatial localization of tomatoes.
- **Microcontroller:** Arduino Nano for PWM generation and real-time control of the servo motor.
- **Servo Motor:** MG996R high-torque servo motor for actuating the gripper fingers.
- **Power Supply:** A 12V source regulated to 5V/3.3V via buck converters to power components safely.

Software Requirements

- **Object Detection Model:** YOLOv8s architecture trained on a custom dataset of tomatoes with bounding box annotations.
- **Inference Runtime:** The model must be deployable on a local processor with limited computing resources.
- **Control Logic:** Code to convert detection outputs into PWM signals to actuate the servo motor precisely.

Environmental and Operational Constraints

- **Lighting Conditions:** The vision system should tolerate moderate variations in lighting, such as shadows or natural outdoor light.
- **Physical Size Constraints:** The gripper must be compact enough to navigate within plant foliage.
- **Durability:** Materials and components must tolerate repeated gripping cycles during testing without degradation.

These detailed requirements guided the subsequent stages of system design, module development, and testing as laid out in the V-Model structure.

4.3.4 Visual Representation

Figure 4.6 shows the customized V-Model diagram used in this project. Dashed arrows represent testing phases planned for future execution.

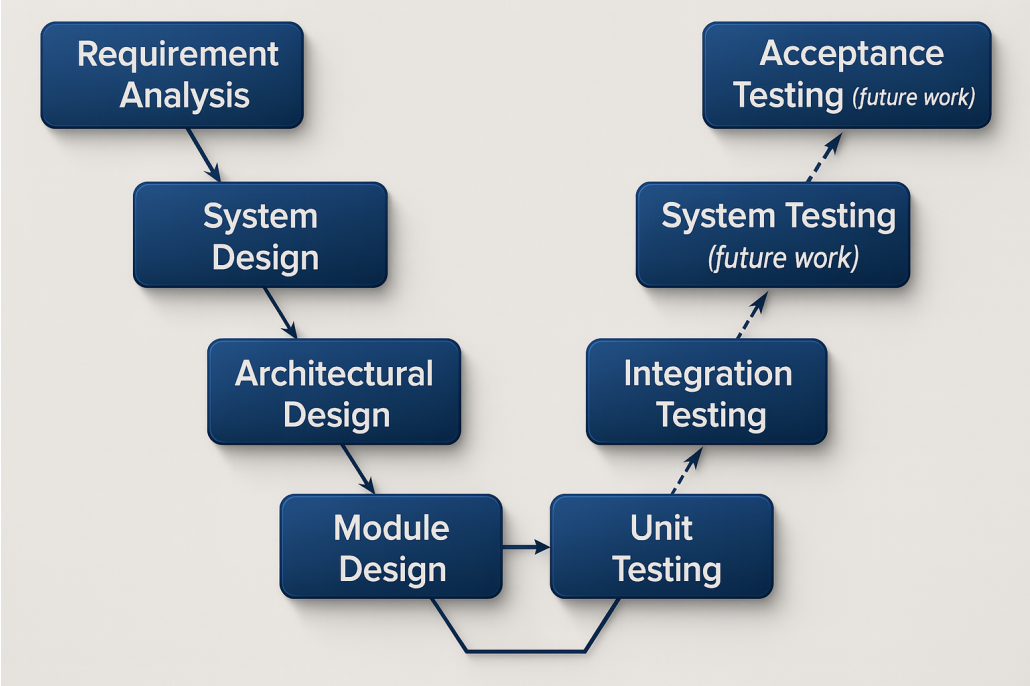

Figure 4.6: Project V-Model, Source: AI Generated

The V-Model allowed for a traceable and structured development process. Each component of the system was designed, built, and tested at the unit and integration levels. Due to timeline and resource constraints, full physical validation through system and acceptance testing will be conducted in future iterations of the project. However, by aligning the project with this model, modularity and verification were ensured throughout.

The adoption of the V-Model enabled disciplined development in a project combining machine vision, robotic actuation, and mechatronic integration. The phases completed thus far validate the feasibility of the system, while future testing phases will confirm its real-world effectiveness.

4.4 Design Concept Generation

The design of the robotic gripper system involved a systematic approach beginning from idea conception to final digital assembly. This section outlines each phase in detail, elaborating on the tools, techniques, and rationale behind every step taken during the design process. The goal was to create a functional, modular, and 3D-printable claw-style end effector that could be mounted on a robotic arm for real-world fruit harvesting applications. Particular emphasis was placed on adaptability, strength, and safety, especially when interacting with delicate and irregularly shaped produce like tomatoes.

4.4.1 Initial Requirements and Design Goals:

The fundamental motivation behind the gripper design was to develop a claw-style mechanism capable of securely grasping and harvesting tomatoes of varying shapes and sizes without causing damage.

The following key design requirements were identified:

- Ensure sufficient flexibility of fingers to conform to irregular fruit shapes.
- Integrate actuation through an MG996R servo motor for simple yet powerful control.
- Enable 3D printing of individual components for rapid prototyping and cost-effective production.
- Design with modularity to allow disassembly, maintenance, and upgrades.
- Guarantee compatibility with an Intel RealSense camera and UR5e robotic arm for autonomous navigation and control.

These requirements framed the constraints for material selection, geometric layout, actuation mechanism, and CAD design strategies.

4.4.2 Ideation and Concept Inspiration

The design inspiration originated from an online video demonstrating a three-fingered mechanical claw powered by an internal driving screw. This mechanism used gear-driven rotation to translate vertical motion into the opening and closing of fingers. Realising the potential of this mechanism, the decision was made to integrate the MG996R servo motor directly into the base of the design to simplify actuation.

From the onset, the aim was not just to replicate but to optimise and adapt the concept for agricultural use. Initial ideas were roughly sketched mentally and quickly tested in CAD for feasibility. Since this was a one-of-a-kind design, many reference parts, such as servo horns and gear teeth, were downloaded from open-source platforms or manufacturer sites and later modified to meet the precise mechanical needs of the design.

CAD Software and Toolchain

The core modelling environment for the design was **Autodesk Fusion 360**, owing to its intuitive interface, strong parametric design capabilities, and compatibility with STL/STEP file formats. However, due to the nature of the imported mesh files (typically downloaded in STL format), the following additional tools were integrated into the design pipeline:

- **MeshLab**: Used for editing and simplifying overly complex mesh bodies that caused issues in CAD conversion. MeshLab allowed mesh decimation, noise removal, and surface smoothing operations.
- **Meshmixer**: Utilized for automatic mesh repair and volume reconstruction. This helped identify non-manifold edges, holes, and self-intersections. However, its auto-repair features occasionally altered important geometry, which had to be corrected manually.
- **Fusion 360 (Mesh Environment)**: Used for mesh-to-BRep (Boundary Representation) conversion once the mesh was cleaned up externally. This step was essential for enabling parametric modifications.

4.4.3 Component-Level Design and Redesign

The robotic gripper was constructed as a modular system consisting of multiple mechanical components, each playing a critical role in the functionality, motion, and reliability of the end effector. This section elaborates on each component's purpose, design details, and integration into the overall assembly. Many components were initially sourced as STL files and then corrected, redesigned, or remodelled to achieve optimal performance and manufacturability.

Gripper fingers

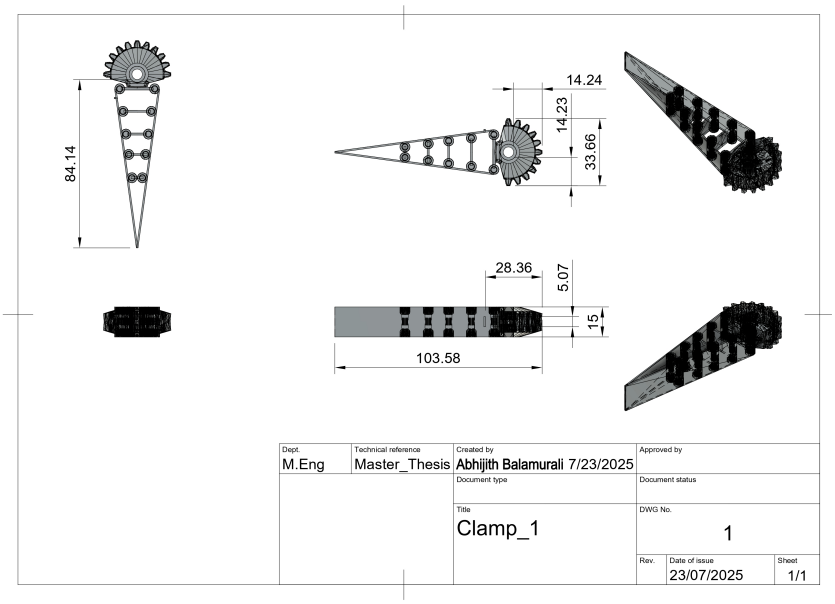

Figure 4.7: Clamp Technical drawing, source= generated from custom design

The gripper fingers are the most vital contact interface between the robot and the object being grasped. Designed in a claw-like three-fingered configuration, they are intended to wrap around irregularly shaped tomatoes and hold them firmly without causing damage. Each finger was modelled with a smooth curvature profile to maximise surface contact and distribute gripping pressure evenly. Structurally, the fingers are reinforced with internal ribs to ensure they can resist deformation under actuation. The fingers are mounted on driven gears that allow them to pivot synchronously inwards or outwards based on the vertical movement of the driving screw.

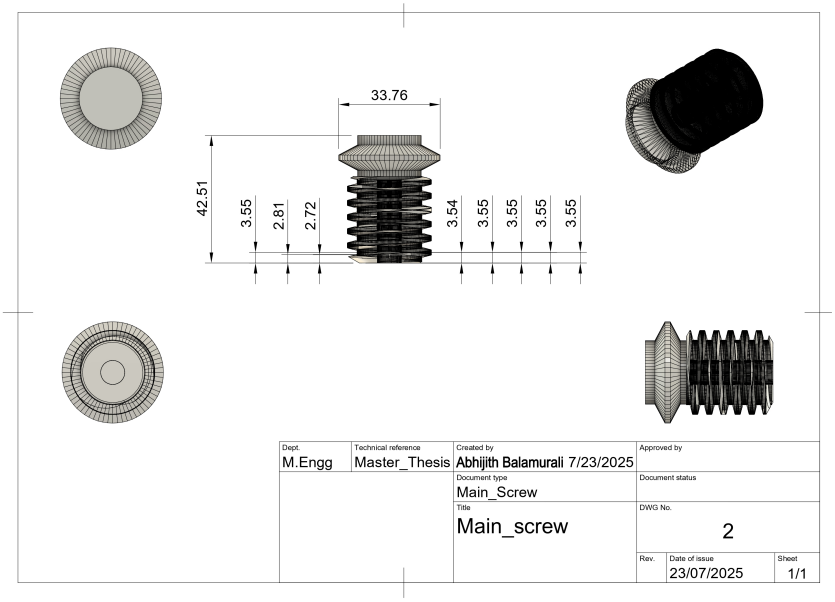

Figure 4.8: Main Screw Technical drawing, source= generated from custom design

The driving screw is a threaded vertical shaft responsible for converting the rotational motion of the MG996R servo motor into linear actuation. As the screw rotates, it moves vertically along its axis, pushing or pulling the gears mounted on each finger. This component was designed to have precise thread geometry compatible with the internal nut cavities of the gear system. Its length was chosen to ensure the maximum permissible travel range without jamming or mechanical interference.

Embedded Finger Gears

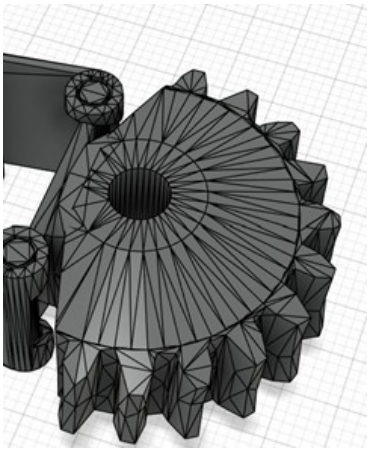

Figure 4.9: Integrated finger Design — Source: Custom design through Fusion 360

Each finger is attached to a semicircular gear that interfaces with the driving screw. These gears rotate the fingers about their base pivots when the driving screw moves vertically. The design ensures equal and simultaneous motion of all three fingers, achieving synchronized gripping.

The gear teeth were remodelled to correct mesh defects and ensure proper engagement. Precise alignment with the screw was crucial for avoiding motion lag or uneven force distribution.

Servo Horn Connector

The servo horn connector serves as the interface between the servo shaft and the driving screw. It was modeled to tightly couple with the MG996R output spline and transfer torque reliably during rotation. A notch or key pattern was included to prevent slipping. To ensure robustness, this component had to be dimensioned precisely according to the servo shaft specification, accounting for both internal spline teeth and mounting holes.

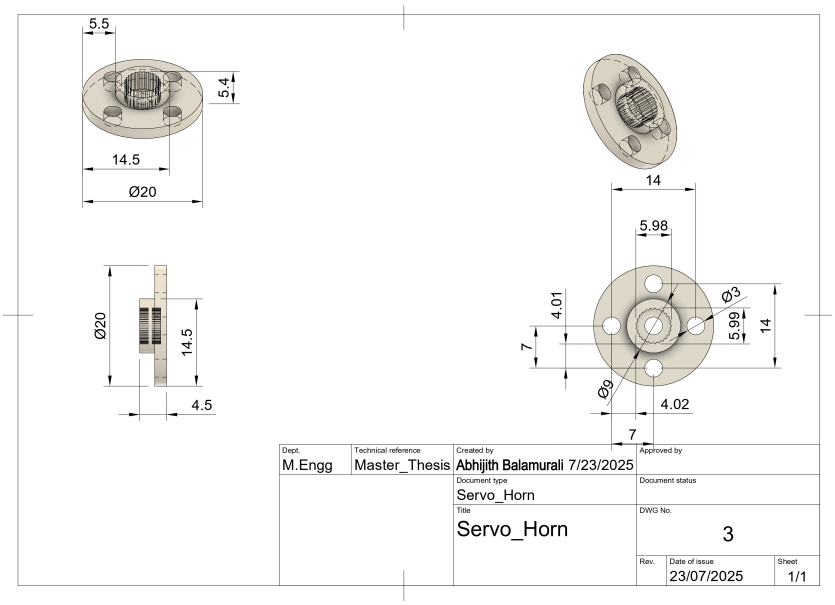

Figure 4.10: Servo Horn Technical drawing, source= generated from custom design

Base Plate for UR5e Mounting

The base plate functions as the primary structural interface between the robotic end-effector and the UR5e manipulator. It was carefully dimensioned and drilled to align with the standard UR5e flange mounting pattern, ensuring quick attachment and mechanical stability.

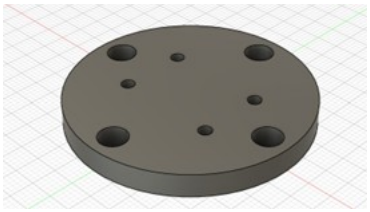

Figure 4.11: Base plate — Source: Custom design through Fusion 360

The plate includes six circular through holes four larger ones for M6 or M8 bolts and two smaller holes for dowel pins or additional alignment. The circular symmetry of the plate supports rotational mounting at various orientations depending on the use case. Its thickness was chosen to maintain rigidity under dynamic loads while minimising overall weight. This plate directly supports the servo motor housing and transfers all mechanical forces generated during gripping and motion to the robot arm.

MG996R Servo Motor Integration

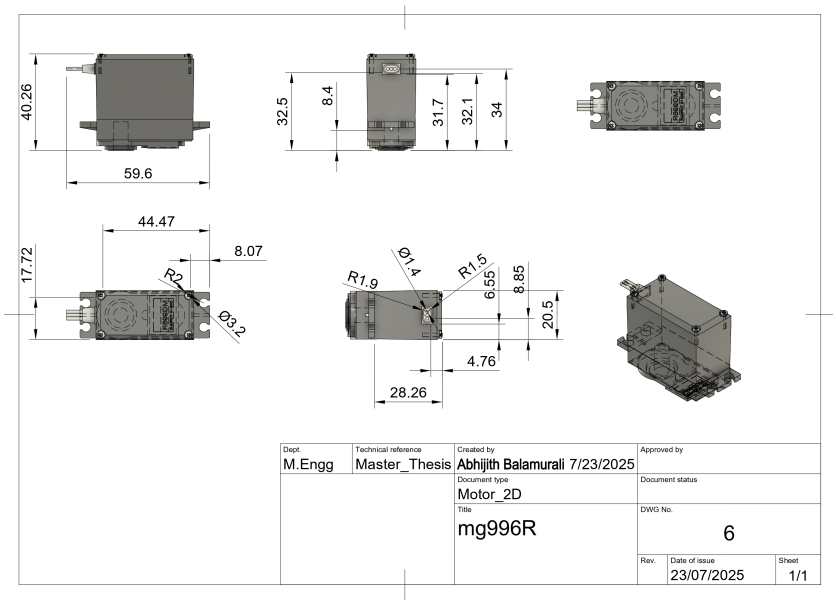

Figure 4.12: Servo motor Technical drawing, source= generated from custom design

The actuation system of the gripper is powered by the MG996R digital high-torque servo motor. Known for its reliability and torque-to-size ratio, this motor provides up to 11 kg·cm of torque at 6V and operates with a speed of 0.17 s/60°. It accepts standard PWM control signals, making it easily programmable for robotic actuation tasks. The motor is mounted vertically beneath the base plate using a custom housing bracket. This bracket was designed with flanged sides, multiple bolt holes, and rectangular slots to tightly constrain the motor body and avoid slippage during operation. It interfaces directly with the custom-designed servo horn that connects to the vertical driving screw, translating the servo's rotational motion into vertical screw displacement, which then actuates the finger gears.

Servo motor mount

The servo motor mount was specifically designed to accommodate the MG996R servo motor used in this project. It provides a stable base for the motor while allowing precise rotational motion to be transferred to the driving screw mechanism of the gripper. The mount features cut-outs and fastening points that align with the motor's body and flange, ensuring secure attachment and preventing any slippage during actuation. The geometry of the mount was modelled to maintain alignment between the motor shaft and the vertically moving threaded rod, thereby ensuring reliable and repeatable motion transfer throughout the gripping cycle.

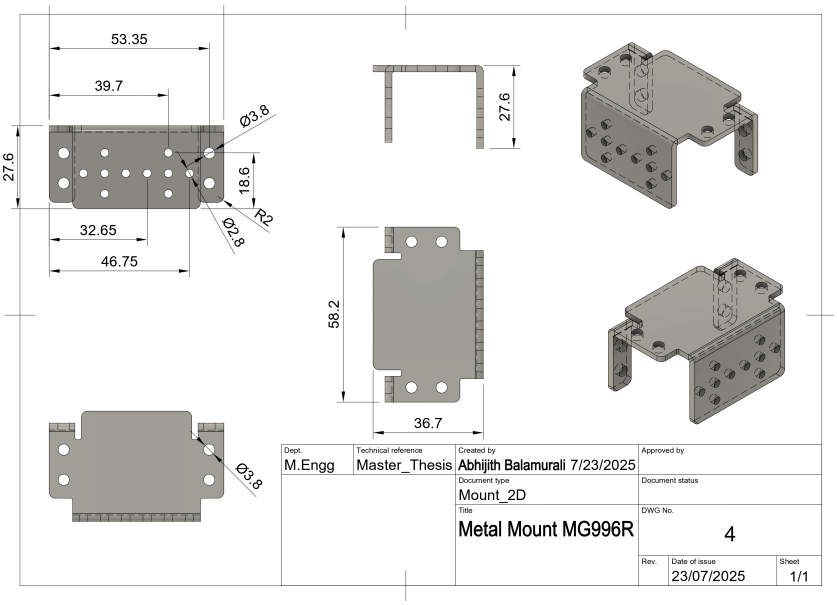

Figure 4.13: Servo motor mount Technical drawing, source= generated from custom design

Intel RealSense Camera and Adaptive Mounting Bracket

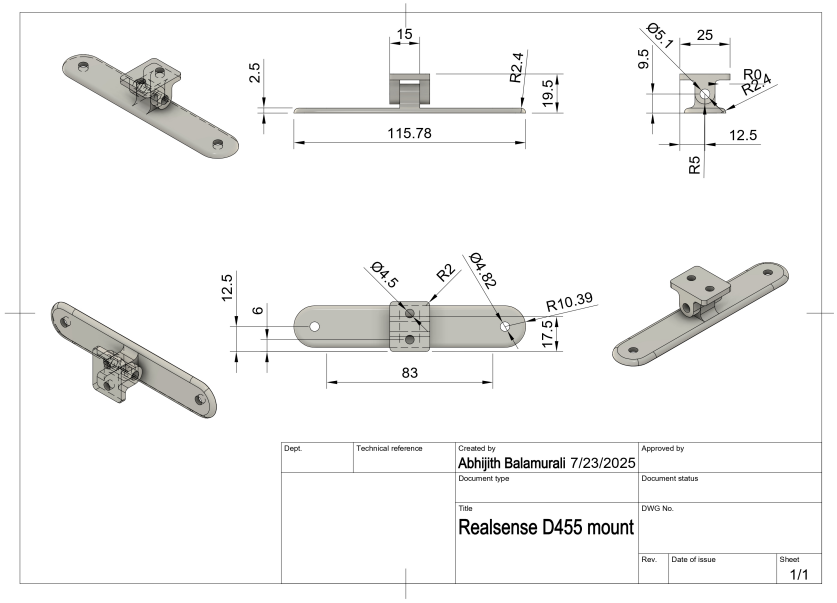

Figure 4.14: RealSense mount Technical drawing, source= generated from custom design

To enable autonomous perception during tomato harvesting, the gripper integrates an **Intel RealSense** camera. This camera captures both RGB and depth data, which is essential for identifying ripe tomatoes and assessing their 3D position in real time.

The camera is mounted using a custom-designed bracket fixed to the side of the motor housing. The bracket features two key design innovations:

- **Adjustable Field of View:** The bracket includes a revolute joint at the connection point

with the camera mount. This allows the camera to tilt up and down, enabling precise adjustments to the viewing angle depending on the task or environment.

- **Lightweight Yet Rigid:** The bracket was designed with minimum material to keep weight low while still maintaining structural integrity. Mounting holes at either end allow it to be fixed to the housing or base plate securely.

This design ensures that the camera's visual field consistently captures the area just in front of the gripper fingers, which is critical for real-time computer vision and decision-making in precision agriculture.

4.5 Intel RealSense L-Link

The **Intel RealSense L-Link** serves as a critical connectivity component designed to interface between Intel RealSense depth cameras and host computing systems. It acts as a robust and secure extension cable system, ensuring high-speed and stable data transmission for both USB 3.0 and power signals in applications requiring flexible or extended placement of the vision module.

The L-Link system is particularly well-suited for robotic and agricultural field deployments, where standard USB connections may not provide the mechanical durability or protection necessary in harsh or mobile environments. Its rugged construction, locking mechanism, and tight connector fit reduce the risk of accidental disconnection due to vibration or movement.

In this project, the L-Link is utilized to connect the *Intel RealSense D435i* camera to the embedded system mounted on the robotic manipulator. Given the need for mobility and modularity in the gripper assembly, the L-Link allows for a stable and secure connection that can accommodate the motion range of the manipulator without introducing signal loss or cable wear. The ability to route cables in constrained spaces without compromising signal quality enhances the mechanical design flexibility and reliability of the vision system integration.

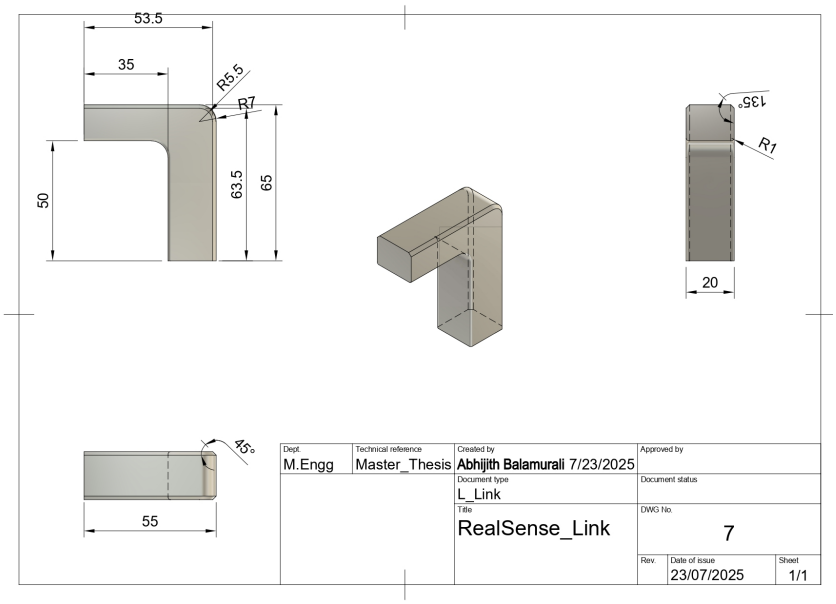

Figure 4.15: RealSense Link Technical drawing, source= generated from custom design

4.5.1 Technical Features

- USB 3.0 support with locking connectors for robust data transmission
- Compatible with RealSense D400 series depth cameras
- Shielded cable for noise immunity and EMI reduction
- Available in multiple lengths (1m, 2m, and 5m) for deployment flexibility
- Designed for harsh environments and mobile robotic applications

The use of the L-Link ensures minimal interference with the dynamic behavior of the end-effector and supports the overall goal of achieving precise, real-time depth sensing and object recognition through the RealSense module.

4.6 Technical Drawing and View Analysis

To further validate the structural integrity, spatial relationships, and manufacturability of the final gripper assembly, two key sets of technical drawings have been generated and analyzed: the projected views and the exploded view. These provide a comprehensive 2D and decomposed 3D perspective of the final CAD model, aiding both design documentation and future replication or fabrication.

4.6.1 Purpose and Significance of Technical Drawings

Technical drawings are standardised graphical representations used to convey detailed information about the geometry, dimensions, tolerances, assembly, and manufacturing requirements of a component or system. Serving as a universal language in engineering, these drawings eliminate ambiguity and ensure that every stakeholder—from designers and engineers to machinists and quality inspectors—interprets the design intent in a consistent and reliable manner. They typically include multiple views such as orthographic projections, section views, detailed zoom-ins, and exploded diagrams, each serving to illustrate different aspects of the design. In mechanical design, technical drawings are essential not only for fabrication and inspection but also for communication during prototyping, documentation, and revision cycles. By adhering to international standards (such as ISO 128 or ASME Y14.5), these drawings facilitate interoperability and precision across global manufacturing environments. Whether produced manually or via CAD software like Fusion 360, technical drawings act as the authoritative source of dimensional and functional intent throughout the product development lifecycle [?, ?]. In the context of this project, the generated technical drawings of the robotic end-effector played a pivotal role in validating part dimensions, planning manufacturing sequences, and ensuring that 3D printed and off-the-shelf components aligned with design constraints.

4.6.2 Projected Views

4.6.3 Orthographic Projection Views of the End-Effector Assembly

Figure 4.16 illustrates the orthographic projection of the complete end-effector assembly, which consists of three principal views: the front view, the top view, and the right-side view. Each view was extracted directly from the 3D assembly in **Fusion 360** to communicate key dimensional and structural details with precision. These orthographic projections enable exact

dimensional analysis and highlight how the geometric constraints of the components interact during actuation and idle phases.

Front View: The front view provides a direct line-of-sight into the operational face of the gripper, where the three-fingered mechanism is symmetrically oriented around the central axis. It shows the radial alignment of the fingers, the circular body housing the internal drive components, and the mounting zone of the MG996R servo motor. This view is critical for assessing the spatial clearance between the fingers when fully closed, as well as the overall width and symmetry of the mechanism.

Top View: The top-down projection reveals the placement of the servo horn, the base plate outline, and the relative distances between the threaded driving screw and each gear housing. It demonstrates how the fingers are equidistantly spaced in a triangular configuration to allow even load distribution during object grasping. It further shows the initial position of the camera mount, partially embedded behind the main housing.

Right-Side View: This view offers a vertical profile of the gripper, including the linear actuation of the central driving screw and its interface with the gear teeth embedded inside each finger. The rotational housing and structural thickness of the servo mount become evident in this view, along with the vertical range of finger motion. It also highlights the spacing above the base plate, ensuring compatibility with the UR5e robot arm's flange connection.

Base View: The base view serves as the primary orientation from which additional orthographic views are derived. It typically represents the most informative face of the 3D model—in this case, the frontal view of the robotic gripper—and acts as the foundation for generating top, side, and isometric projections.

Orthographic Views as a Communication Tool: Orthographic views provide 2D representations of the object from perpendicular planes, allowing clear visualization of dimensions, geometry, and component relationships. The front view displays the finger layout and central housing; the top view reveals internal alignments such as the screw and servo positioning, while the side view captures vertical structure and mounting features. Together, these views ensure precise communication of design intent for manufacturing, assembly, and verification, without requiring physical inspection of the 3D model.

Together, the base view and its derived orthographic projections ensure comprehensive visual documentation of the robotic assembly, enhancing the communication between the design, analysis, and manufacturing stages of the project.

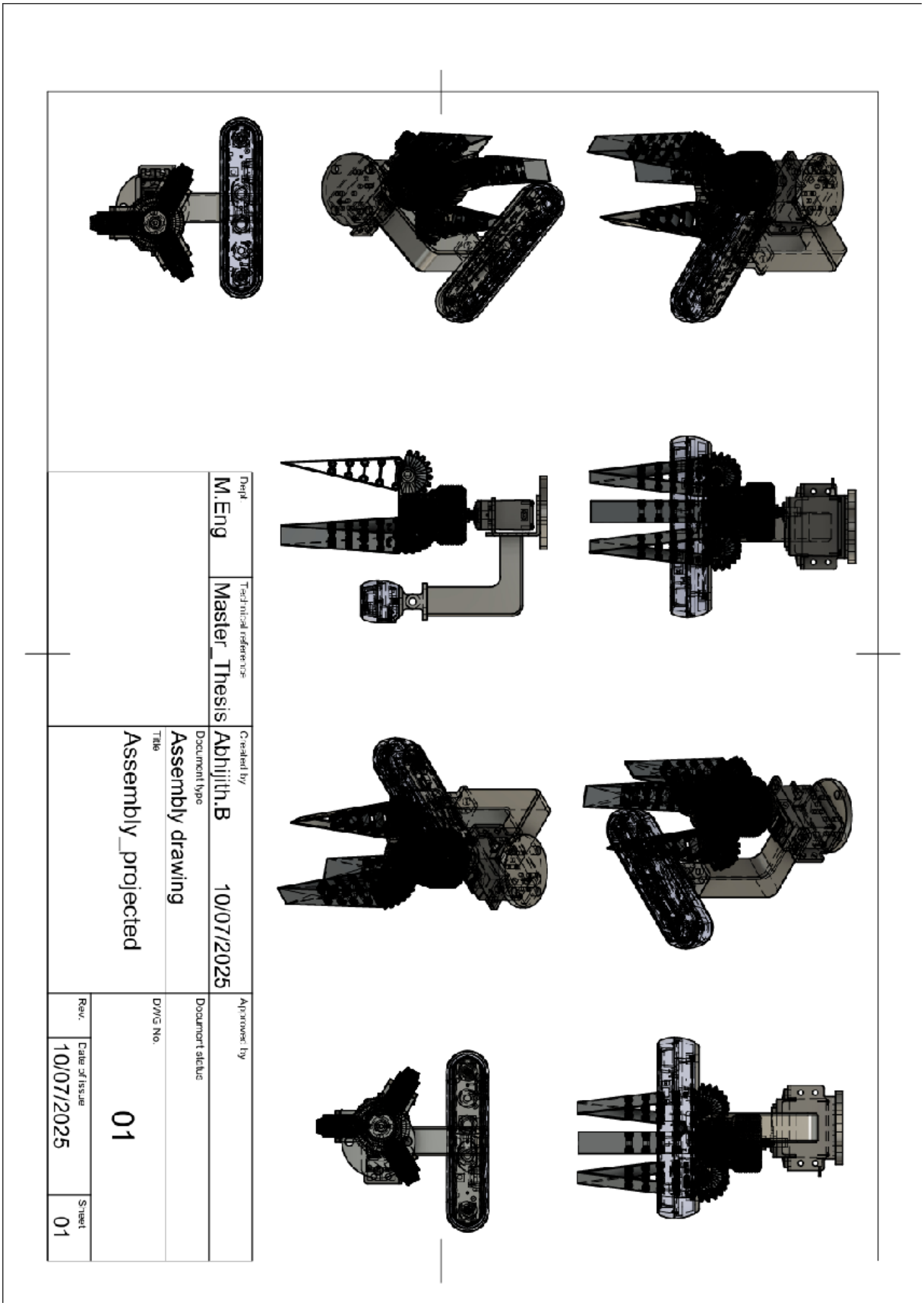

Figure 4.16: Orthographic projection views of the complete end-effector assembly.

4.6.4 Exploded View

Figure 8.1 presents the exploded view of the end gripper, providing a decomposed perspective of each individual part in relation to its assembly path. This view is particularly valuable for understanding how the system is constructed and how its modular components can be replaced or maintained.

Component Breakdown: The exploded view identifies the following key components:

- **MG996R Servo Motor:** Located centrally at the base, this high-torque digital servo forms the actuator for the entire gripping mechanism. It drives the vertically moving threaded screw.
- **Threaded Driving Screw:** A vertically positioned shaft that translates rotary motion from the servo into linear motion. It drives the finger-closing mechanism through mechanical linkage to the gear system.
- **Finger Assemblies (3x):** Each finger contains embedded gears that engage with the rotating screw. As the screw translates vertically, the gear system ensures the synchronised opening and closing of all fingers.
- **Base Plate:** This is the interface between the gripper assembly and the UR5e robot arm. It has four countersunk mounting holes for rigid attachment and houses the servo securely.
- **Camera Mount and Sensor Housing:** Positioned above the base, this unit is coupled via a revolute joint, allowing tilt adjustments for an enhanced field of view. The Intel RealSense camera (or similar) is installed here for real-time object detection.
- **Guide Channels and Couplers:** Internal components (not fully visible in this view) ensure the controlled motion of fingers and prevent misalignment or mechanical backlash.

Assembly Logic: Each component is spaced along its axis of assembly to reflect how it would be integrated during the build process. The modularity of the design is evident, allowing each part to be fabricated separately (e.g., via 3D printing) and assembled in a logical sequence. This approach also supports ease of replacement or upgrades to individual parts—especially the gripper fingers or camera system—without needing to disassemble the entire unit.

The exploded view enhances the documentation by serving as both a technical reference for assembly and an educational visual for stakeholders evaluating the design principles.

Together, these technical drawings showcase the spatial coherence, modularity, and mechanical function of the end effector, thereby strengthening its viability for robotic applications such as intelligent harvesting, precision grasping, or sample collection in unstructured environments.

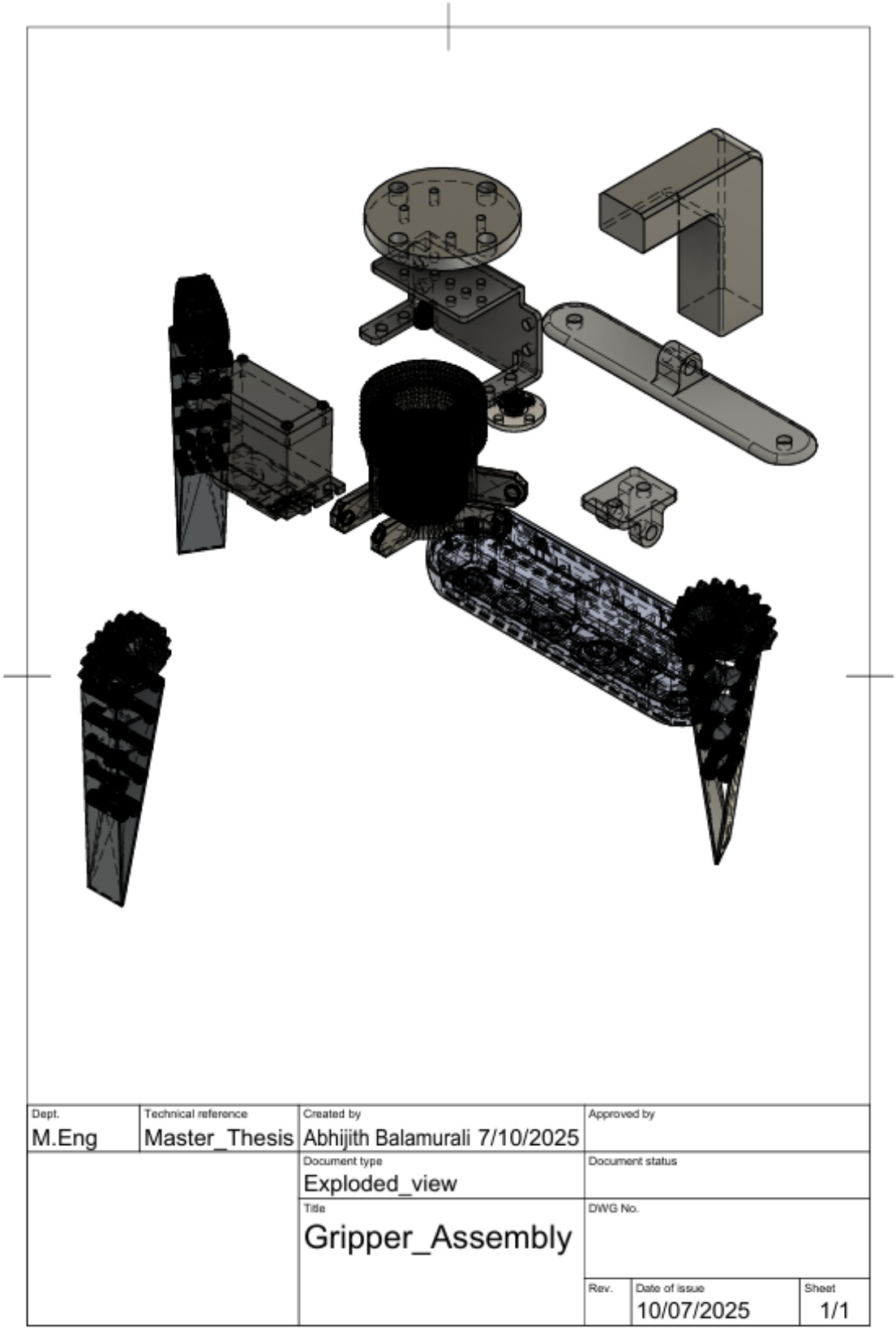

Figure 4.17: Exploded view of the modular end-effector assembly showing part-to-part integration.

4.7 Manufacturing Process

The manufacturing process for the end-effector assembly primarily follows an additive approach, supported by selective acquisition of commercially available components. Given the modular and compact nature of the design, fused deposition modelling (FDM) was chosen as the primary manufacturing technique for all feasible components, whereas functional parts like the servo motor, camera unit, and fasteners were sourced directly due to complexity or precision requirements.[107]

4.7.1 Additive Manufacturing: An Overview

Additive manufacturing, commonly known as 3D printing, is a process of fabricating three dimensional objects by successively layering material based on a digital model. Among the several techniques available, Fused Deposition Modelling (FDM) is the most accessible and suitable for prototyping and functional part production in academic and small-scale industrial contexts. [107]

Fused Deposition Modeling (FDM)

FDM operates by extruding thermoplastic filament through a heated nozzle that traces each layer of the part geometry onto a print bed. Once a layer is deposited and solidified, the next layer is printed on top, gradually building up the complete object.

4.7.2 Material Selection

- **PLA (Polylactic Acid):** Chosen for non-load-bearing components such as outer shells or finger exteriors due to its dimensional stability and ease of printing.
- **PETG (Polyethylene Terephthalate Glycol):** Used for components requiring higher strength and thermal resistance, such as gear housings or brackets.
- **TPU (Thermoplastic Polyurethane):** Considered for gripper contact surfaces, offering a balance between flexibility and durability, though optional depending on part contact dynamics.

4.7.3 Manufacturing 3D Printable Components

- **Finger Mechanism (3x):** Each finger was printed separately, designed with embedded gear profiles. Orientation was optimised to minimize support material and ensure gear integrity.
- **Threaded Driving Screw:** While ideally machined for precision, an experimental version was printed to validate mechanical threading compatibility and tolerances.
- **Gear Components:** Small-sized gear trains were printed in PETG using high-resolution (0.1 mm layer height) settings to ensure tooth definition and meshing accuracy.
- **Base Plate:** A structurally important interface connecting the gripper to the UR5e flange. It was printed with 100% infill to ensure rigidity and thread compatibility for fasteners.
- **Camera Mount:** Printed with a revolute socket allowing adjustable orientation. Print orientation was chosen to preserve the overhanging structure and maintain hinge tolerances.

4.7.4 Post-Processing Steps

Post-processing was performed on printed parts to enhance structural quality and assembly compatibility:

- **Support Removal:** Detachable supports were cleaned using pliers and deburring tools.
- **Surface Smoothing:** Critical areas such as gear faces and threaded surfaces were manually sanded to reduce friction and improve kinematic performance.
- **Drilling and Tapping:** Mounting holes were cleaned and tapped to accept metal screws for mechanical joints.

4.7.5 Commercially Acquired Components

Certain components were acquired off the shelf due to their functional complexity, precision requirements, or cost-efficiency. These include:

- **MG996R Servo Motor:** A high-torque digital servo responsible for actuating the central drive screw.
- **Intel RealSense Camera (or Equivalent):** Facilitates real-time visual data acquisition, crucial for autonomous gripping.
- **Fasteners and Bolts:** Metric screws (M3–M6) were used for assembly, sourced commercially to ensure standardisation and durability.

4.7.6 Print Settings and Process Optimization

- **Layer Height:** 0.2 mm for most components; 0.1 mm for precision features like gears.
- **Infill Density:** 20–40% for standard parts; 100% for load-bearing parts like the base plate.
- **Print Speed:** 40–60 mm/s depending on material and detail requirements.
- **Supports:** Enabled for overhangs above 45° and manually edited for fragile components.

4.7.7 Types of 3D Printing Machines

3D printing encompasses a diverse range of technologies, each suited to specific materials, resolutions, and applications. The most commonly used types include:

- **Fused Deposition Modelling (FDM):** The most accessible and widely used technology, particularly in educational and prototyping environments. FDM machines extrude thermoplastic filament (e.g., PLA, ABS, PETG) through a heated nozzle layer by layer. These printers are cost-effective and ideal for fabricating functional mechanical components, as used extensively in this project. [107]
- **Stereolithography (SLA):** Utilizes a UV laser to cure liquid resin layer by layer. SLA offers superior resolution and surface finish, making it suitable for high-detail components such as dental models or microfluidic devices. However, SLA machines and materials are typically more expensive and require post-processing such as resin washing and UV curing. [108]
- **Selective Laser Sintering (SLS):** Involves sintering powdered material (usually nylon or composites) using a high-powered laser. SLS enables the production of highly durable

and complex parts without the need for support structures, making it suitable for industrial-grade applications and low-volume manufacturing. [109]

- **Digital Light Processing (DLP):** Similar to SLA but uses a digital projector to cure resin in layers. DLP printers are known for high-speed printing and are used in fields requiring precision, such as jewellery design and dental implants. [110]
- **Binder Jetting and Material Jetting:** These methods are used in more specialised applications such as full-colour printing or metal part production. While not commonly found in desktop environments, they represent the frontier of high-performance additive manufacturing. [111, 112]

Each technology has its trade-offs in terms of resolution, mechanical strength, material compatibility, and cost. For the purposes of this thesis, **FDM printing** was selected due to its practicality, affordability, and ability to produce robust parts suitable for mechanical assembly and iterative prototyping.

4.7.8 Toolpaths and Slicing Software

Slicing software serves as the essential intermediary between a 3D model and the printer by converting digital geometry (e.g., STL, OBJ, 3MF) into layered G-code instructions. It determines print parameters such as layer height, infill, support structures, and tool paths, thereby directly influencing the final part's quality, strength, and print time [107].

Understanding Toolpaths in Additive Manufacturing

A toolpath is the virtual route followed by a 3D printer's nozzle or extruder during the printing process. These paths are generated from the 3D model by slicing software and dictate every motion made by the printer, including material extrusion, travel moves, retractions, and layer changes. Toolpaths are fundamental to the printing process, as they directly influence print quality, structural integrity, speed, and material usage.

Types of Toolpaths:

- **Infill Paths:** Define the internal structure of a part. Different infill patterns (e.g., grid, triangle, gyroid) affect strength, weight, and print time.
- **Perimeter/Wall Paths:** Outline the outer contour of the part. Multiple perimeter layers increase strength and surface finish quality.
- **Top and Bottom Layers:** Solid paths that form the initial and final surfaces, crucial for ensuring closed surfaces and a visual finish.
- **Support Structures:** Temporary toolpaths generated beneath overhangs or bridging features, removed post-printing.
- **Travel Moves:** Non-printing movements between toolpaths, optimized to avoid stringing and reduce travel time.

Factors Influencing Toolpath Generation:

- **Layer Height:** Smaller heights produce finer resolution and smoother surfaces but increase print time.
- **Wall Line Count:** Determines how many perimeter loops are printed for strength.
- **Infill Density and Pattern:** Affects internal structure and weight.

- **Print Speed and Acceleration:** Influences print quality and machine vibration.
- **Retraction Settings:** Control filament pullback to reduce stringing during travel moves.

An optimized toolpath is essential to ensure dimensional accuracy, reduce material waste, and maintain part strength.

Slicing Software: Role and Options

Slicing software converts a 3D model (typically in STL, STEP, or OBJ format) into a sequence of instructions that a 3D printer can execute. These instructions, usually in the form of `.gcode` files, define every aspect of the toolpath and printer behavior.

Key Functions of a Slicer:

- Analyzes 3D model geometry and validates mesh integrity.
- Generates toolpaths based on printer and material profiles.
- Simulates the print process for visualization and verification.
- Allows customization of print parameters such as temperature, speed, supports, infill, and cooling.

Popular Slicing Software Platforms:

- **Ultimaker Cura:** Open-source, widely used slicer with extensive community support, detailed print previews, and customizable profiles.
- **PrusaSlicer:** Developed by Prusa Research; offers robust settings and support for multi-material printing.
- **Simplify3D:** A commercial slicer known for high-speed slicing and fine control over support structures and toolpaths.
- **IdeaMaker:** User-friendly slicer developed by Raise3D; supports a variety of printers with built-in profiles.
- **Slic3r:** The open-source predecessor to many slicers, known for its configurability and scripting features.

4.7.9 Toolpaths and Slicing Workflow with Bambu Lab P1S

The **Bambu Lab P1S** is a high-speed, closed-frame desktop 3D printer developed by Bambu Lab, a Shenzhen-based additive manufacturing company known for producing performance oriented, user-friendly machines. Designed as an upgrade to the Bambu Lab P1P, the P1S integrates advanced motion control, enclosure design, and cloud connectivity, making it particularly well-suited for precision engineering tasks and rapid prototyping. [113]

Key Specifications:

- **Printing Technology:** Fused Filament Fabrication (FFF) / Fused Deposition Modeling (FDM)
- **Build Volume:** 256 × 256 × 256 mm
- **Maximum Printing Speed:** Up to 500 mm/s
- **Maximum Acceleration:** 20,000 mm/s^2 (CoreXY motion system)

- **Nozzle Diameter:** 0.4 mm (standard); compatible with 0.2–0.6 mm
- **Layer Resolution:** 0.08 mm to 0.28 mm
- **Hotend Temperature:** Up to 300°C
- **Build Plate Temperature:** Up to 100°C
- **Frame Type:** Enclosed chamber with cooling fan system (ideal for temperature-sensitive materials)
- **Filament Compatibility:** PLA, PETG, TPU, ABS, ASA, PA, PC, and carbon/glass-fiber-reinforced filaments
- **Connectivity:** Wi-Fi, SD card, Bambu Cloud Sync

Additional Features:

- Automated bed leveling
- Filament run-out detection and jam recovery
- Built-in 1080p monitoring camera
- Seamless integration with Bambu Studio and optional Bambu AMS (Automatic Material System)

Chapter 5

Testing and Validation

5.1 Prediction Testing Cycle

After training the YOLOv8s model, a series of prediction tests were conducted to evaluate how well the model could detect ripe and unripe tomatoes in both static images and real-time video streams. The aim of these tests was not only to verify the model's accuracy but also to observe its responsiveness, robustness, and ability to generalise to unseen inputs. The testing process followed a gradual path, starting with predictions on labelled test images and progressing to real-time detection using the integrated laptop webcam and the Intel RealSense camera. A total of three prediction cycles were conducted to explore different settings and conditions. This section outlines each stage of the prediction pipeline in detail.

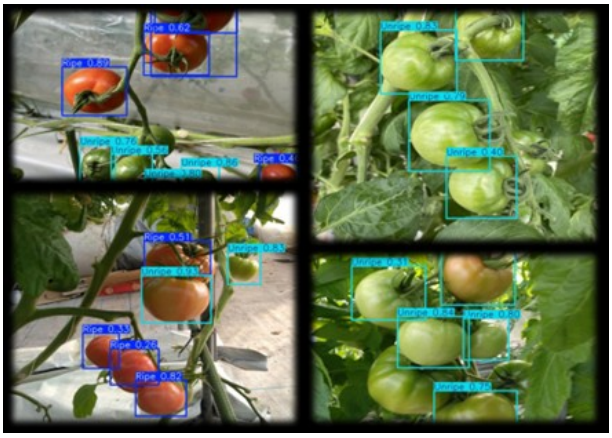

Figure 5.1: Collage of cycle 1 testing results — Source: adapted from custom-developed results

5.1.1 Initial Tests on Static Images

The first stage of testing was conducted using a batch of static test images containing known samples of ripe and unripe tomatoes. These images were not part of the training dataset, and were used to check whether the model had generalised well to new inputs. The model was loaded with the trained best.pt weights, and predictions were made using the YOLOv8 CLI and Python interface. The results included bounding boxes over detected tomatoes, with class labels such as "Ripe" and "Unripe", and confidence scores indicating how certain the model was about each detection.

The results from this image-based prediction cycle were very promising:

- The model successfully detected the tomatoes in each test image, accurately classifying them as ripe or unripe.
- The bounding boxes were tightly fitted, with minimal overlap or misalignment.
- The confidence scores ranged between 70% and 90%, showing high certainty in the predictions.
- There were no false positives (objects incorrectly labelled as tomatoes) or missed detections in the images tested.

This phase provided confidence that the model had learnt meaningful patterns during training and could now be evaluated in real-world, dynamic conditions.

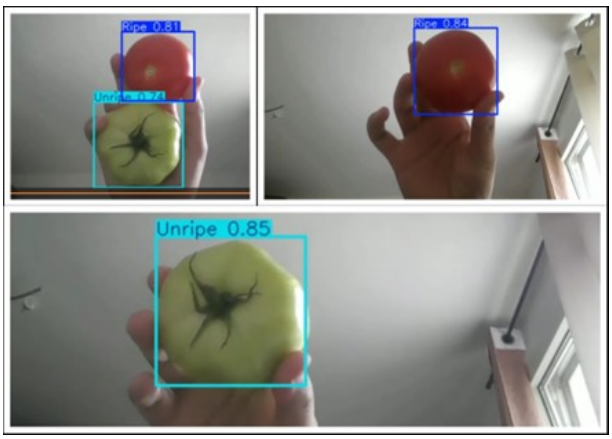

Figure 5.2: Collage of cycle 2 testing results — Source: adapted from custom-developed results

5.1.2 Transition to Real-Time Testing Using Webcam

Encouraged by the success with static images, the next logical step was to test the model in a real-time environment. This involved using the system's integrated webcam to perform live detection of tomatoes held in front of the camera under natural lighting conditions.

To facilitate real-time prediction:

- A Python script using OpenCV and the Ultralytics YOLOv8 API was developed.
- The webcam feed was continuously passed through the model in real time.
- The results were visualized live with annotated bounding boxes and labels.
- The predictions were also saved as video files using OpenCV's VideoWriter.

5.1.3 Cycle 1 – Live Detection of Ripe Tomato

In the first real-time cycle, a ripe tomato was placed in front of the webcam. The model immediately detected the tomato and placed a bounding box around it labelled "Ripe." As the object was moved around the frame—rotated, shifted, and partially obscured—the model maintained consistent detection. The bounding box remained stable, and the detection was fluid with minimal flickering. Confidence scores for ripe tomato detection ranged from 65% to 78%, even under varied hand positions and camera angles. This demonstrated the model's strong ability to detect objects on live video streams with high responsiveness.

5.1.4 Cycle 2 – Real-Time Unripe Tomato Detection Using RealSense Camera

The second test focused on detecting unripe tomatoes. The RealSense camera was used to provide clearer visuals and simulate a more structured imaging environment. A green, unripe tomato was held in the frame. The model rapidly detected and labelled it as "Unripe," with confidence scores consistently reaching above 85%. This cycle confirmed earlier findings that unripe tomatoes were easier for the model to detect—possibly due to their sharper contrast against the background. Even when lighting conditions changed slightly or the tomato was turned, the model maintained accurate classification and tracking.

5.1.5 Cycle 3 – Multi-Object Detection in Real-Time

For the final cycle, both a ripe and an unripe tomato were placed together in the frame to test the model's ability to detect and distinguish multiple objects simultaneously. The model successfully detected both objects, drawing individual bounding boxes and assigning the correct labels.

The system was able to:

- Identify multiple tomatoes of different classes at the same time.
- Maintain independent bounding boxes without overlap.
- Sustain detection stability over multiple frames.

The annotated output video clearly showed the real-time, multi-object detection in action, proving that the model had learned not just classification but also spatial awareness and separation of multiple instances.

5.1.6 General Observations

Across all three prediction cycles—static images and live streams—the YOLOv8s model performed reliably and accurately. Key observations include:

- **Strong detection of unripe tomatoes:** Higher confidence and better localization were observed for unripe tomatoes, consistent with their visual distinctiveness and higher representation in the training dataset.
- **Slightly lower confidence for ripe tomatoes:** While detection was still accurate, the confidence levels for ripe tomatoes were slightly lower, possibly due to class imbalance or visual blending with the background.
- **No major false positives or false negatives:** The model did not confuse tomatoes with background elements and demonstrated high robustness in noisy, real-world conditions.
- **Fast inference and response times:** Even with high-resolution input, the model produced real-time predictions with smooth performance, enabling practical deployment in interactive systems.

These tests successfully validated the model's real-world applicability, confirming that it can be used in agricultural tasks such as robotic harvesting, sorting, or visual inspection systems. The gradual transition from image-based testing to real-time webcam deployment further highlighted the model's versatility and practical strength.

Test Cycle	Environment	Object Type(s)	Detection Outcome	Confidence Range
Cycle 1: Static Image Prediction	Offline image set using Python CLI	Ripe and Unripe Tomatoes	Accurate bounding boxes and class labels on all test images; no false positives	70–90%
Cycle 2: Real-Time Detection (Webcam)	Integrated laptop camera in live stream mode	Single Ripe Tomato	Bounding box maintained with minimal jitter; correct label shown consistently	65–78%
Cycle 3: Real-Time Detection (RealSense)	RealSense RGB camera feed in real-time	Ripe and Unripe Tomatoes (multi-object)	Detected both tomatoes with correct labels and stable bounding boxes	Ripe: 65–72%, Unripe: 85–90%

Table 5.1: Summary of Model Prediction Test Cycles

5.2 Precision-Recall Curve Analysis

The Precision-Recall (PR) curve shown in Figure 7.3 helps us understand how well the trained model is performing when it comes to detecting ripe and unripe tomatoes. This kind of curve is very useful, especially in situations where there is a large difference between the number of objects in different classes (for example, if we have many more unripe tomatoes than ripe ones in our dataset).

Precision tells us how many of the tomatoes the model predicted correctly out of all the predictions it made. For example, if the model says there are 10 ripe tomatoes and 8 are actually ripe, the precision is 0.8. On the other hand, recall tells us how many correct predictions the model made out of all the actual ripe (or unripe) tomatoes that were present. So, if there were 10 ripe tomatoes in an image and the model found 7 of them, the recall is 0.7. The curve shows this trade-off between precision and recall. A model that is very good will have a curve that stays close to the top-right corner (high precision and high recall at the same time).

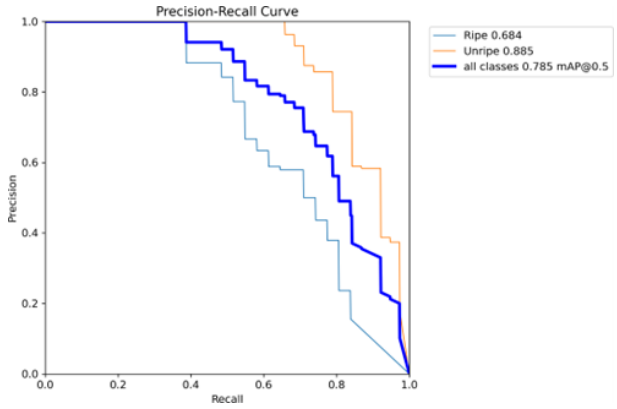

Figure 5.3: Precision-recall curve — Source: adapted from custom-developed results

From the PR curve, we can see the following:

- The orange line shows the performance for unripe tomatoes. It is very high and close to the top right of the graph, which tells us that the model is very good at detecting unripe tomatoes. It achieved a score of 0.885 mAP@0.5, which means that on average, it was about 88.5% accurate at identifying unripe tomatoes.
- The light blue line represents ripe tomatoes. This curve is slightly lower than the unripe one, which tells us that the model found ripe tomatoes a bit harder to detect. The score here was 0.684 mAP@0.5, or about 68.4%. This could be because ripe tomatoes may look more similar to their surroundings or were under-represented in the training data.
- The thick dark blue line is the average of both classes. This overall model performance is measured using something called mean Average Precision at IoU 0.5 (mAP@0.5), and the value is 0.785. This means that, on average, the model correctly identified tomatoes with an accuracy of 78.5% at a standard matching threshold.

5.3 Understanding the Confusion Matrix

The confusion matrix shown in Figure 5.4 helps us see exactly how the model is making its predictions for ripe and unripe tomatoes, as well as how it handles background areas (parts of the image that are not tomatoes at all).

This matrix is called a "confusion" matrix because it shows where the model gets confused between different classes. The rows of the matrix show what the model predicted, and the columns show the actual (true) classes in the dataset.

In our case, we have two classes:

- **Ripe:** tomatoes that are fully grown and ready to be picked
- **Unripe:** tomatoes that are still green or not yet ready

The numbers inside the matrix are between 0 and 1 and represent percentages (e.g., 0.65 means 65%). These values are normalised, indicating the proportion of correct and incorrect predictions for each class.

Here's what we can understand from the matrix:

- The model correctly identified 65% of ripe tomatoes as ripe. However, 36% of ripe tomatoes were incorrectly classified as background. This suggests that the model occasionally misses ripe tomatoes or mistakes them for other objects or empty areas. A possible reason is that ripe tomatoes blend in with the background or have more varied appearances.
- For unripe tomatoes, the model performed much better. It correctly predicted 87% of unripe tomatoes, while only 13% were confused as background. This implies that unripe tomatoes are easier to recognise, likely because they exhibit greater contrast in colour or shape.
- Regarding the background class, the model sometimes showed confusion. It mistakenly identified certain parts of the background as ripe (35%) or unripe (64%). This indicates that the model occasionally detects patterns in the background resembling tomatoes, even when none are present.

In simpler terms, this matrix reveals the following:

- The model is very confident when identifying unripe tomatoes.
- It occasionally misses ripe tomatoes or confuses them with the background.
- It sometimes detects tomatoes in background regions where none exist, especially in visually complex scenes.

By analysing this matrix, it becomes evident which aspects of the model need further refinement. Specifically, supplying more training samples of ripe tomatoes or enhancing background diversity in the dataset could reduce misclassification and improve the system's overall reliability.

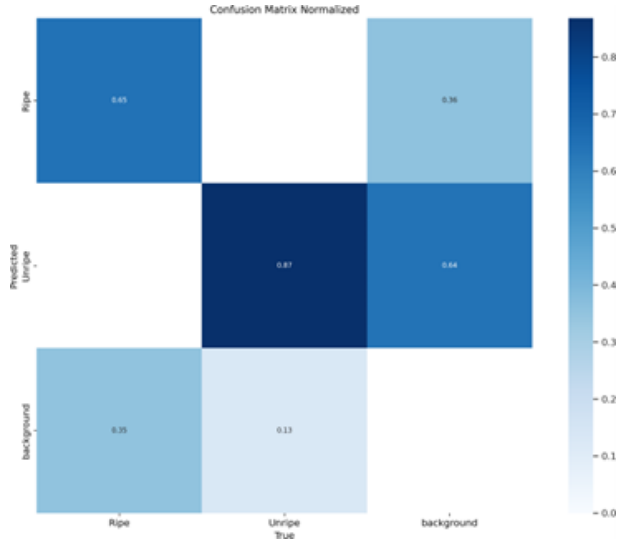

Figure 5.4: Confusion Matrix — Source: adapted from custom-developed results

5.3.1 Model Training Progress and Evaluation Metrics

Figure 5.5 illustrates the progression of the YOLOv8s model's performance over the course of training. The model was trained for a total of 50 epochs, and various metrics were logged throughout to evaluate both learning efficiency and generalisation capability. These metrics fall into two main categories: *loss functions*, which the model seeks to minimise, and *performance metrics*, which the model aims to maximise.

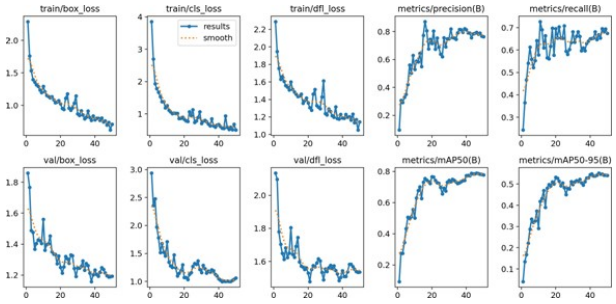

Figure 5.5: Evaluation Metrics — Source: adapted from custom-developed results

Each graph presented includes two lines:

- A solid **blue line** representing the raw recorded values at each epoch.
- A **dotted orange line** representing the smoothed trend to help visualise the overall direction of training.
- **train/box loss and val/box loss:** These losses measure how well the model learns to localise tomatoes by drawing bounding boxes. A decreasing loss indicates improved accuracy. Both training and validation box losses decline steadily, indicating effective learning and limited overfitting.
- **train/cls loss and val/cls loss:** These plots show classification loss, reflecting the model's ability to differentiate between ripe and unripe tomatoes. The consistent decrease in these values suggests growing classification confidence and accuracy.
- **train/dfl loss and val/dfl loss:** Distribution Focal Loss (DFL) enhances bounding box precision. Steady reduction of DFL indicates that the model's box predictions become increasingly fine-tuned and less erratic.
- **metrics/precision (B):** Precision measures how often the model's positive predictions are correct. An increase in precision, stabilising above 0.8, signifies fewer false positives and stronger prediction confidence.
- **metrics/recall (B):** Recall represents the proportion of actual tomatoes correctly detected by the model. The upward trend toward 0.7 reflects improved coverage of the target objects.
- **metrics/mAP50 (B):** The mean Average Precision (mAP) at 0.5 IoU evaluates combined precision and recall. The metric rises sharply and levels off around 0.78, demonstrating solid detection performance.
- **metrics/mAP50-95 (B):** This more rigorous metric averages mAP across IoU thresholds from 0.5 to 0.95. The curve's growth and final stabilisation near 0.55 indicate strong generalisation capability across varying overlap levels.

These evaluation metrics confirm that the YOLOv8s model learnt effectively over time and achieved a balance between detection precision, coverage, and generalisation, making it suitable for real-world agricultural deployment.

Overall, all curves show a healthy learning process. The losses go down, and the accuracy scores go up. There is no sign of overfitting (where the model performs well on training data but badly on new data), and both training and validation metrics improve together. This indicates that the model is generalising well and is ready for real-world testing.

5.4 Understanding the F1-Confidence Curve

The F1-confidence curve shown in Figure 5.6 is used to help us understand how well the model balances two important qualities—precision and recall—at different levels of confidence.

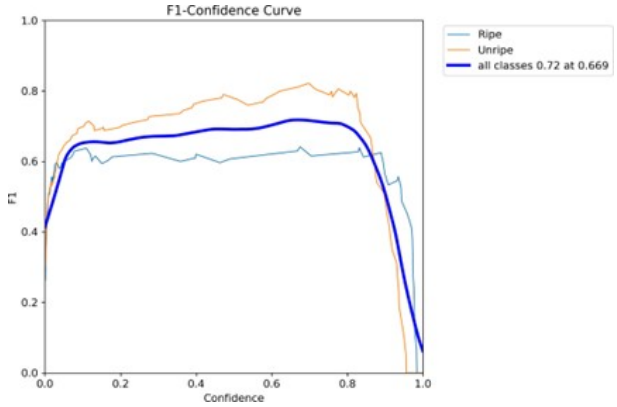

Figure 5.6: F1 Confidence Curve — Source: adapted from custom-developed results

5.4.1 F1 Score and Confidence Threshold Analysis

In simple terms:

- **Precision** measures how many of the tomatoes the model identified were actually correct.
- **Recall** measures how many of the real tomatoes present in the image were successfully identified by the model.

The **F1 score** combines both precision and recall into a single metric to provide a balanced evaluation of the model's performance. An F1 score of 1.0 represents perfect prediction, while a score near 0 indicates poor performance.

This analysis is based on how the F1 score varies with different confidence thresholds. The confidence threshold determines the minimum certainty required for the model to accept a detection as valid. For instance, a threshold of 0.5 means the model only retains predictions with at least 50% confidence.

- The **orange line** in the F1-confidence curve represents the F1 score for unripe tomatoes. It exceeds 0.8 at optimal thresholds, indicating strong precision and recall.
- The **light blue line** corresponds to ripe tomatoes, peaking around 0.65. This suggests that detecting ripe tomatoes remains slightly more challenging, potentially due to inconsistent lighting or varied appearances.

- The **thick dark blue line** denotes the average F1 score across all classes, reaching a maximum of 0.72 at a confidence threshold of approximately 0.669.

This curve aids in choosing an optimal confidence threshold that balances the trade-off between over-predicting (false positives) and missing detections (false negatives). A lower threshold increases recall but may reduce precision, while a higher threshold does the opposite. Hence, selecting a threshold around 0.669 ensures the best overall model performance in practical deployments.

5.4.2 Overall Model Performance

After completing multiple testing phases—ranging from static image prediction to real-time webcam and RealSense camera detection—the YOLOv8s model consistently demonstrated strong performance across varying conditions. The following strengths were observed:

- **Accuracy:** High detection accuracy was achieved, particularly for unripe tomatoes, with minimal false positives or false negatives.
- **Real-Time Responsiveness:** The model handled live streams with fluid bounding box tracking and negligible lag.
- **Robustness to Movement and Occlusion:** Consistent predictions were made even during object motion or partial occlusions.
- **Multi-Object Detection:** The model effectively differentiated and labelled multiple tomatoes in the same frame.
- **Ease of Integration:** With a Python-based interface, the model integrated seamlessly with OpenCV for live visualisation and video recording.

These qualities position the model as a practical solution for applications such as robotic harvesting or automated grading in agricultural systems.

5.4.3 Considerations for Real-World Deployment

While model testing yielded highly promising results, certain real-world challenges must be addressed before deployment in agricultural settings:

- **Lighting Variations:** Outdoor lighting conditions are dynamic, with changes in sunlight, shadows, and cloud cover. Data augmentation or adaptive preprocessing may be required to maintain detection performance.
- **Background Complexity:** Real-world scenes, especially in fields or greenhouses, may introduce cluttered or inconsistent backgrounds. Extra training is needed to reduce confusion with non-target objects like leaves or soil.
- **Hardware Stability:** Performance in real-time depends on reliable hardware and camera positioning. Embedded systems such as Jetson Nano or Raspberry Pi will require optimisation for inference speed and memory usage.
- **Physical Occlusions:** Fruits may be hidden behind foliage or branches. Using depth information from cameras such as Intel RealSense can help mitigate these issues.

- **Safety and Failover Design:** In robotic applications, missed or incorrect detections may lead to mechanical errors. Implementing human validation layers or fallback protocols is recommended for safety.
- **Dataset Expansion:** Although the current dataset is effective, including more real-world samples—especially edge cases like semi-ripe or damaged tomatoes—will enhance model generalisation and reliability.

These considerations are crucial for bridging the gap between laboratory testing and successful real-world application in agricultural robotics and automation.

Table 5.2: Model Readiness and Deployment Considerations

Aspect	Current Status	Deployment Consideration
Detection Accuracy	High for both classes, especially unripe tomatoes	Monitor performance on new lighting/background conditions
Real-Time Performance	Stable and responsive on GPU-based laptops	Test on edge devices and optimize for embedded deployment
Multi-Object Handling	Successfully identifies multiple tomatoes with clear separation	Validate performance in crowded, overlapping scenarios
Lighting Sensitivity	Good under indoor lighting	Improve robustness for outdoor variable lighting
Background Clutter	Accurate in simple backgrounds	Retrain or fine-tune for natural field environments
Hardware Compatibility	Runs smoothly on local systems with webcam and RealSense camera	Profile performance and thermal limits on field-deployable hardware
User Interaction	Provides clear visual feedback with live bounding boxes	Add alerts or flags for low-confidence detections
False Positives/Negatives	Minimal in testing; controlled behavior	Monitor and log prediction errors continuously in deployment

5.5 Requirements Validation

This section outlines how specific system requirements—identified during the requirement analysis phase—were addressed throughout the course of the project. Due to resource and time constraints, only a subset of the full requirement set could be fulfilled. The table below lists all successfully implemented or partially validated requirements.

ID	Requirement Description	Validation Status
R1	Detection of ripe tomatoes using a YOLOv8s object detection model	Fully Validated through trained model on custom dataset
R3	Real-time loop from camera input to actuation decision	Partially validated via simulated inference loop (no hardware actuation)
R5	Modular architecture enabling independent software and hardware development	Validated by decoupled YOLO model, control code, and mechanical design
R7	Scalability to retrain software pipeline on other fruit types or datasets	Validated by modular dataset structure and YOLOv8 reusability
R10	3D printable end-effector design suitable for soft gripping	Validated through CAD design with print-ready tolerances
R11	Integration of Intel RealSense camera for RGB-D vision input	Fully implemented and tested in detection workflow
R13	Control of MG996R servo motor via PWM logic	Validated through direct PWM control in Arduino code
R15	YOLOv8s model successfully trained on annotated tomato images	Fully Validated using Roboflow-augmented dataset
R16	Deployment of trained YOLOv8 model for local inference	Successfully deployed and tested in controlled scenarios
R17	Arduino firmware developed to convert detection to actuation signals	Validated with mock signals and serial inputs
R18	Vision performance under varied lighting conditions	Partially validated in indoor lighting tests; outdoor robustness pending
R19	Compact mechanical gripper design for confined spaces	Validated through CAD layout with spatial constraints

Table 5.3: Validation Status of Fulfilled Requirements

Future work will aim to validate the remaining system requirements, especially those related to physical actuation, energy optimization, and end-to-end system testing.

Chapter 6

Discussion

6.1 Overview

While the initial scope of the project included the complete design, fabrication, and deployment of a modular, three-fingered robotic end effector for autonomous fruit harvesting, certain constraints necessitated a strategic refocusing. Due to time limitations and logistical constraints associated with physical manufacturing, the hardware assembly was deprioritised. Instead, the project pivots towards finalising and extensively testing the computer vision system for fruit maturity classification using RGB-D data from an Intel RealSense camera. This chapter outlines the proposed roadmap for completing the software pipeline, as well as long-term milestones for hardware integration and industrial application.

6.2 Vision Model Completion

6.2.1 Additional Testing and Validation

To ensure that the object detection model performs reliably under practical conditions, a systematic expansion of the test suite is essential. While initial training was conducted on a controlled dataset of 300+ annotated tomato images, the model needs to be stress-tested across varying environmental contexts, including:

- Different lighting conditions: harsh sunlight, partial shadow, and overcast scenarios.
- Occlusions due to leaves, branches, or overlapping fruit clusters.
- Variable fruit sizes and partial ripeness states.
- Background clutter and non-tomato objects (e.g., rocks, tools).

To facilitate this, a larger and more diverse dataset will be curated from both open-access agricultural repositories and custom image captures. Evaluation will be based on standard metrics such as mAP@0.5, precision-recall curves, and Intersection over Union (IoU) consistency across test scenarios.

6.2.2 Robustness Improvements

To enhance generalisability and reduce false detections, the following improvements are planned:

- **Data Augmentation:** Apply synthetic modifications including rotation, brightness shift, blur, zoom, and noise to simulate field variability.

- **Hyperparameter Optimisation:** Use grid search and Bayesian optimisation to tune model parameters such as learning rate, confidence threshold, IoU threshold, and NMS strategy.
- **Ensemble Learning:** Train multiple YOLOv8 variants (e.g., `yolov8s`, `yolov8m`) and combine predictions via majority voting or soft-NMS to increase detection accuracy.
- **Model Quantisation and Pruning:** Optimise the model for deployment by reducing size and computation without sacrificing accuracy.

6.2.3 Real-Time Deployment Goals

The end goal of this phase is to establish a deployable real-time vision system. Target hardware includes embedded computing platforms such as the NVIDIA Jetson Nano or Jetson Xavier, due to their CUDA-capable GPUs. Deployment tasks include:

- Converting the PyTorch model into TensorRT or ONNX format.
- Benchmarking inference time under live video feed (30 FPS target).
- Implementing buffer queues for continuous frame analysis with minimum latency.
- Integrating RealSense SDK for synchronised depth + RGB input.

This will enable edge inference, vital for real-world agricultural deployment where cloud computing may be infeasible.

6.3 Completed Functional and Technical Objectives

The development process systematically addressed several core functional and technical requirements that were successfully implemented and validated within the scope of this thesis 5.3. Requirement R1, related to the accurate detection of ripe fruits using a deep learning-based computer vision model, was fulfilled through the deployment and testing of a YOLOv8s model trained on a custom-curated dataset. Requirement R5, targeting seamless data acquisition from RGB-D inputs via the Intel RealSense camera, was implemented with full integration into the vision pipeline, enabling real-time capture and processing of spatially referenced data. Requirement R7, concerning the annotation and augmentation of image datasets to ensure model robustness, was completed using LabelImg and Roboflow, resulting in enhanced model generalization. Requirement R10, which emphasized modular code design for interoperability and maintainability, was achieved through a layered implementation of vision, control, and interface modules. In addition, Requirement R11, involving the visualization of bounding boxes and class labels during inference, was fully implemented using OpenCV overlays, allowing qualitative model assessment and debug visualization. Lastly, Requirement R13, focused on system-level logging and debugging interfaces, was addressed via structured print statements and error-handling protocols to support iterative development and testing. These fulfilled requirements represent the technological backbone of the current system and provide a validated foundation for subsequent hardware integration and field testing.

These requirements form the basis for the future system development and validation roadmap, particularly focusing on hardware implementation, energy profiling, and robust safety mechanisms.

6.4 Hardware Assembly

6.4.1 Mechanical Assembly Plan

The CAD design for the three-fingered claw gripper is fully modelled in Fusion 360. The design accounts for modularity, ease of 3D printing, and maintainability. The gripper incorporates a vertically movable driving screw connected to gear-driven finger segments. Future work in this domain includes:

- 3D printing the components using TPU and PLA to test strength, flexibility, and fit.
- Performing tolerance analysis and interference checks for post-processing alignment.
- Designing tool-less assembly fixtures for rapid prototyping iterations.

6.4.2 Servo Integration and Actuation Testing

The selected actuator, MG996R servo, offers high torque and cost-effectiveness. The following tasks are planned for integration:

- Designing a mounting hub for precise gear coupling between the servo horn and the gripper's screw shaft.
- Implementing Pulse Width Modulation (PWM)-based control via microcontroller (e.g., Arduino or ESP32).
- Creating a test rig to study response time, angular precision, and torque under variable load.
- Investigating feedback sensors (e.g., encoders or potentiometers) for closed-loop control.

6.4.3 Full System Calibration

Once vision and actuation are developed, calibration will involve aligning the coordinate space of the camera with the physical workspace of the gripper. Calibration steps include:

- Mapping pixel-to-real-world coordinates using depth information from RealSense.
- Implementing inverse kinematics for actuation based on object location.
- Conducting grip success rate trials to statistically quantify precision and recall in physical operations.

6.5 Integrated System Goals

6.5.1 End-to-End Pipeline Integration

The most significant future milestone is to develop an end-to-end operational pipeline, integrating computer vision, servo control, and robotic manipulation. The system will operate in the following loop:

1. Capture frame using RealSense RGB-D camera.
2. Process frame with YOLOv8 model for ripe tomato detection.
3. Estimate real-world 3D position from bounding box and depth data.
4. Trigger actuation using coordinate-mapped servo instructions.
5. Perform post-action feedback validation and retry logic.

6.5.2 Agricultural Field Deployment

To validate the robustness and reliability of the integrated system, it will be tested under realistic conditions in a tomato farm. Deployment goals include:

- Mounting the entire system on a mobile base (e.g., UGV).
- Running autonomous harvest trials over several crop rows.
- Monitoring energy consumption, actuation success, and real-time feedback loops.
- Documenting limitations due to weather, lighting, or terrain.

6.6 Industrial and Research Extensions

6.6.1 Modular Scalability

The modular approach followed throughout the project—both in software and hardware—makes it highly extensible. Future directions include:

- Replacing gripper fingers for tasks like pruning or spraying.
- Training the vision model to detect pests or diseases.
- Plug-and-play tool changer for performing multiple actions with the same arm.

6.6.2 Integration with Autonomous Rovers

The mothership–rover system initially conceptualised for planetary exploration can be adapted to agriculture. Future work may explore:

- Housing multiple grippers in a rover equipped with docking and charging stations.
- Coordinated task distribution among multiple robotic units.
- Centralised data logging and performance monitoring for large-scale operations.

6.6.3 Potential for Cross-Domain Use

Though developed for agriculture, the system's underlying technologies—vision, actuation, and modularity—are applicable in:

- Warehouse automation for sorting and packing.
- Biomedicine for tissue manipulation using soft robotics.
- Disaster recovery operations in constrained environments.

The project has made substantial progress in vision model development, and a complete mechanical design is ready for future fabrication. By documenting both immediate next steps and long-term goals, this chapter provides a structured roadmap to transform this prototype into a real-world deployable solution that merges AI, robotics, and practical usability. Future implementations have the potential to serve as scalable frameworks for autonomous, modular, and intelligent robotic systems.

Chapter 7

Conclusion

7.1 Conclusion

The research and development efforts presented in this thesis have led to the successful implementation of a smart, vision-based robotic system for precision tomato detection, contributing to the broader goals of sustainable agriculture and automation under the framework of Agriculture 4.0. By leveraging state-of-the-art deep learning techniques, user-friendly annotation tools, and practical hardware components, the project bridges the gap between theoretical computer vision and its real-world agricultural application.

A key highlight of this work lies in the deployment of the YOLOv8s object detection model, trained on a combination of three uniquely curated datasets. This model demonstrated strong performance in distinguishing between ripe and unripe tomatoes across multiple test scenarios, including static images and real-time webcam-based inference. The training process, initially started on a dual-boot Ubuntu system and later migrated to WSL due to compatibility issues, served as a practical lesson in managing platform-specific limitations.

The selection of tools and libraries—such as Python, Ultralytics YOLO, Roboflow, LabelImg, and Intel RealSense SDK—enabled a modular and scalable software pipeline, while the hardware strategy focused on portability, depth sensing, and soft robotic interaction. These components worked together to establish a functional prototype for future agricultural harvesting robots.

Moreover, the project's structured testing cycle validated the model's robustness, while the inclusion of evaluation metrics such as precision-recall curves, confusion matrices, and F1 scores provided empirical backing for its accuracy and reliability.

Despite certain limitations, such as reduced generalisability under unpredictable environmental conditions and incomplete physical hardware integration, the project offers a solid foundation for future work in the domain of robotic fruit harvesting. This includes improved dataset augmentation, real-field deployment, and complete end-effector fabrication.

In conclusion, this thesis not only showcases the potential of combining machine learning with robotics in agricultural automation but also highlights the importance of iterative development, cross-disciplinary integration, and continuous evaluation. It serves as a stepping stone toward intelligent, accessible, and environmentally conscious agricultural solutions that can benefit farmers and researchers alike.

Chapter 8

Appendix

8.1 Additive Manufacturing

8.1.1 Understanding Toolpaths

A toolpath refers to the specific trajectory followed by the 3D printer's nozzle as it extrudes material to form each layer of the object. These paths are generated by slicing software and are a critical determinant of the print's quality, strength, and success rate. The toolpath defines not only where the nozzle moves but also when it deposits material, retracts, travels between print sections, and performs specific actions such as layer changes or cooling delays.

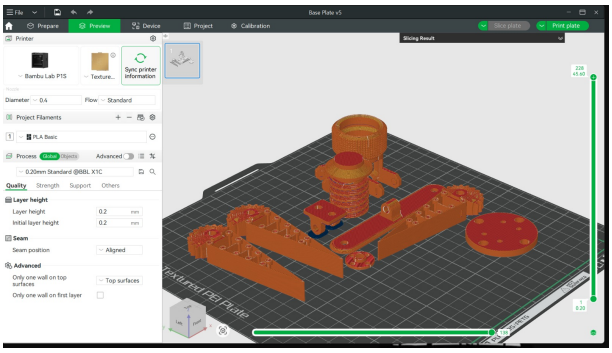

Figure 8.1: View of my components in slicer getting prepped for printing

Common Toolpath Categories:

- **Perimeter Paths:** Define the external and internal walls of the printed part.
- **Infill Paths:** Fill the internal volume using structural patterns like grid, gyroid, or honeycomb.
- **Top/Bottom Surfaces:** Solid layers that seal the top and base of the object.
- **Support Structures:** Temporary toolpaths generated to support overhangs, removed after printing.
- **Travel Moves:** Non-printing motion between extrusion areas to optimise efficiency and reduce stringing.

Influencing Parameters: These paths are influenced by layer height, nozzle diameter, infill density, print speed, material properties, and cooling strategies. A well-optimised toolpath reduces print time, minimises artefacts, and ensures mechanical reliability.

8.1.2 Slicing with Bambu Studio and Printer Integration

Bambu Studio, developed specifically for Bambu Lab printers, was used for slicing and managing all 3D printable components in this project. It integrates seamlessly with the Bambu Lab P1S, allowing fine-tuned control of print parameters and real-time feedback during fabrication.

Why Bambu Lab P1S Was Chosen

The Bambu Lab P1S offers several significant advantages over traditional FDM printers like the Ultimaker series:

- **Remarkable Print Speed:** The P1S supports CoreXY motion architecture and accelerates up to 20,000 mm/s^2, enabling print speeds up to 500 mm/s. In practical use, this translates to a 70–80% reduction in total print time compared to printers like the Ultimaker S3/S5.
- **Automatic Material System (AMS):** Although not mandatory, the P1S is compatible with Bambu Lab's AMS system for multi-colour and multi-material printing, expanding possibilities for future design complexity.
- **Advanced Vibration Compensation and Pressure Advance:** Built-in features reduce artefacts like ringing or ghosting, even at high speeds, ensuring sharp, dimensionally accurate prints.
- **Closed Chamber and Cooling Efficiency:** Unlike open-frame printers, the P1S maintains better thermal conditions for materials like PETG, resulting in fewer warping issues and improved layer adhesion.
- **Integrated Camera and Remote Monitoring:** Live feed and timelapse capability allow print tracking and early error detection.
- **Ease of Use and Maintenance:** The Bambu Lab P1S has automated bed levelling, filament detection, and resume-on-power-loss functionality, simplifying the entire printing process.

Advantages Over Ultimaker Printers

- **Speed:** Ultimaker S3/S5 average print speeds are 60–100 mm/s. Bambu P1S achieves up to 500 mm/s with comparable quality.
- **Cost-Performance Ratio:** Bambu P1S offers industrial-grade capabilities at a significantly lower cost.
- **Enclosure and Noise Control:** While Ultimaker requires add-ons for a closed chamber, the P1S includes this by default.
- **Software Synergy:** Bambu Studio offers faster slicing and printer-optimised defaults. Cura may require manual calibration and material tuning.

Bibliography

[1] H. Li, C. Xu, X. Yang, B. Zhang, Z. Li, and T. Hu, "A review of recent advances in fruit and vegetable harvesting robots," *Artificial Intelligence Review*, 2023.

[2] W. Fang, Z. Wu, W. Li, X. Sun, W. Mao, R. Li, and L. Fu, "Fruit detachment force of multiple varieties kiwifruit with different fruit-stem angles for designing universal robotic picking end-effector," *Computers and Electronics in Agriculture*, vol. 213, p. 108225, 2023.

[3] L. Oliveira, A. Moreira, and M. Silva, "Advances in agriculture robotics: A state-of-the-art review and challenges ahead," *Robotics*, vol. 10, no. 3, p. 52, 2021.

[4] S. Bhat and N. Huang, "Big data and ai revolution in precision agriculture: Survey and challenges," *IEEE Access*, vol. 9, pp. 110 209–110 222, 2021.

[5] A. AlZubi and K. Galyna, "Artificial intelligence and internet of things for sustainable farming and smart agriculture," *IEEE Access*, vol. 11, pp. 78 686–78 692, 2023.

[6] C. Deutsch, J. Tewksbury, M. Tigchelaar, D. Battisti, S. Merrill, R. Huey, and R. Naylor, "Increase in crop losses to insect pests in a warming climate," *Science*, vol. 361, no. 6405, pp. 916–919, 2018.

[7] FAOUN, *The State of Food and Agriculture 2022: Leveraging Agricultural Automation for Transforming Agrifood Systems*. Food and Agriculture Organization of the United Nations, 2022.

[8] A. Cravero, S. Pardo, S. Sepúlveda, and L. Muñoz, "Challenges to use machine learning in agricultural big data: A systematic literature review," *Agronomy*, vol. 12, no. 3, p. 748, 2022.

[9] J. Martens, T. Blut, and J. Blankenbach, "Cross domain matching for semantic point cloud segmentation based on image segmentation and geometric reasoning," *Advanced Engineering Informatics*, vol. 57, p. 102076, 2023.

[10] W.-X. Chen, J.-R. Liao, A.-C. Chen, P.-Y. Kou, and Y.-L. Ting, "Combining physical robotic arm and virtual multimedia animation with electromechanical integration materials," in *Proceedings of the 19th Conference on Research and Development in Science Education and the 12th Conference on Engineering, Technology, and STEM Education*, 2023, pp. 1037–1059.

[11] A. Feldman, "Robotics firms garnered $6.3 billion in venture funding during the pandemic year, 2021," https://www.forbes.com/sites/amyfeldman/2021/robotics-funding, 2021, forbes.

[12] United Nations, "Food," https://www.un.org/globalissues/food, 2021, global Issues.

[13] P. Fraga-Lamas, S. Lopes, and T. Fernández-Caramés, "Green iot and edge ai as key technological enablers for a sustainable digital transition towards a smart circular economy: An industry 5.0 use case," *Sensors*, vol. 21, no. 17, p. 5745, 2021.

[14] T. Zhang, Y. Zhao, W. Jia, and M. Chen, "Collaborative algorithms that combine ai with iot towards monitoring and control system," *Future Generation Computer Systems*, vol. 125, pp. 677–686, 2021.

[15] H. Zhou, X. Wang, W. Au, H. Kang, and C. Chen, "Intelligent robots for fruit harvesting: Recent developments and future challenges," *Precision Agriculture*, vol. 23, pp. 1856–1907, 2022.

[16] A. Silwal, F. Yandun, A. Nellithimaru, T. Bates, and G. Kantor, "Bumblebee: A path towards fully autonomous robotic vine pruning," *Field Robotics*, vol. 2, pp. 1661–1696, 2022.

[17] Y. Jo, Y. Park, and H. Son, "A suction cup-based soft robotic gripper for cucumber harvesting: Design and validation," *Biosystems Engineering*, vol. 238, pp. 143–156, 2024.

[18] L. Mu, G. Cui, Y. Liu, Y. Cui, L. Fu, and Y. Gejima, "Design and simulation of an integrated end-effector for picking kiwifruit by robot," *Information Processing in Agriculture*, vol. 7, no. 1, pp. 58–71, 2020.

[19] B. Zhang, J. Zhou, Y. Meng, N. Zhang, B. Gu, Z. Yan, and S. Idris, "Comparative study of mechanical damage caused by a two-finger tomato gripper with different robotic grasping patterns," *Biosystems Engineering*, vol. 171, pp. 245–257, 2018.

[20] S. Hayashi, K. Shigematsu, S. Yamamoto, K. Kobayashi, Y. Kohno, J. Kamata, and M. Kurita, "Evaluation of a strawberry-harvesting robot in a field test," *Biosystems Engineering*, vol. 105, no. 2, pp. 160–171, 2010.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[22] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7263–7271.

[23] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[24] Z. Xu, R. Jia, H. Sun, Q. Liu, and Z. Cui, "Light-yolov3: Fast method for detecting green mangoes in complex scenes using picking robots," *Applied Intelligence*, vol. 50, no. 12, pp. 4670–4687, 2020.

[25] W. Au, H. Zhou, T. Liu, E. Kok, X. Wang, M. Wang, and C. Chen, "The monash apple retrieving system: A review on system intelligence and apple harvesting performance," *Computers and Electronics in Agriculture*, vol. 213, p. 108164, 2023.

[26] D. Guri, M. Lee, O. Kroemer, and G. Kantor, "Hefty: A modular reconfigurable robot for advancing robot manipulation in agriculture," *arXiv preprint arXiv:2402.18710*, 2024.

[27] L. Grimstad and P. J. From, "The thorvald ii agricultural robotic system," *Robotics*, vol. 6, no. 4, p. 24, 2017.

[28] Wikipedia, "Mechanised agriculture," https://en.wikipedia.org/wiki/Mechanised_agriculture, 2025, accessed July 2025.

[29] Encyclopædia Britannica, "Farm machinery— history, uses, facts," https://www.britannica.com/technology/farm-machinery, 2025, accessed July 2025.

[30] M. Berducat, C. Cariou, P. Martinet, and B. Thuilot, "Automatic guidance of a farm tractor relying on a single cp-dgps," *Autonomous Robots*, vol. 13, no. 1, pp. 53–71, 2002.

[31] K. Fitzpatrick, M. Happold, M. Ollis, H. Pangels, T. Pilarski, and A. Stentz, "The demeter system for automated harvesting," *Autonomous Robots*, vol. 13, no. 1, pp. 9–20, 2002.

[32] V. Callaghan, M. Carr-West, M. Colley, and H. Hagras, "Online learning and adaptation of autonomous mobile robots for sustainable agriculture," *Autonomous Robots*, vol. 13, no. 1, pp. 37–52, 2002.

[33] A.-J. Baerveldt and B. Åstrand, "An agricultural mobile robot with vision-based perception for mechanical weed control," *Autonomous Robots*, vol. 13, no. 1, pp. 21–35, 2002.

[34] N. Buchmann, R. Finger, R. Huber, and A. Walter, "Smart farming is key to developing sustainable agriculture," *Proceedings of the National Academy of Sciences*, vol. 114, no. 24, pp. 6148–6150, 2017.

[35] S. Fountas *et al.*, "Agricultural robotics for field operations," *Sensors*, vol. 20, no. 9, p. 2672, 2020.

[36] C. O. Adetunji, D. I. Hefft, and O. T. Olugbemi, "Agribots: A gateway to the next revolution in agriculture," in *AI, Edge and IoT-based Smart Agriculture*. Elsevier, 2021, pp. 111–124.

[37] FAO, *The State of Food and Agriculture 2017: Leveraging Food Systems for Inclusive Rural Transformation*. Food and Agriculture Organization of the United Nations, 2017.

[38] D. Mengistu and G. Ashe, "Review of artificial intelligence powered food processing: Enhancing safety and sustainability," *Journal of Agroaliment. Proc. Technol.*, vol. 30, pp. 192–202, 2024.

[39] U. Mogili and B. Deepak, "Review on application of drone systems in precision agriculture," in *Procedia Computer Science*, vol. 133. Elsevier, 2018, pp. 502–509.

[40] A. Bechar and C. Vigneault, "Agricultural robots for field operations: Concepts and components," *Biosystems Engineering*, vol. 149, pp. 94–111, 2016.

[41] S. Mohan, R. Venkat, S. Rahaman, M. Vinayak, and B. Babu, "Role of ai in agriculture: Applications, limitations and challenges: A review," *Agricultural Reviews*, vol. 44, pp. 231–237, 2023.

[42] M. Peter and M. Vecchia, "The digital marketing toolkit: A literature review for the identification of digital marketing channels and platforms," in *New Trends in Business Information Systems and Technology*. Springer, 2021, pp. 251–265.

[43] A. Hussain, P. Liatsis, M. Khalaf, H. Tawfik, and H. Al-Asker, "A dynamic neural network architecture with immunology inspired optimization for weather data forecasting," *Big Data Research*, vol. 14, pp. 81–92, 2018.

[44] G. Ashe and D. Mengistu, "Future of farming: A review of artificial intelligence applications in agricultural production and processing," *Asian Scientific Bulletin*, vol. 3, no. 1, pp. 82–91, 2025.

[45] C.-C. Wang, C.-W. Hung, S.-J. Yan, and C. C. Ho, "Application of yolo and custom-designed intelligent teaching aids in robotic arm-based fruit classification and grasping instruction," *Preprints*, 2025, preprint.

[46] K. Ehsani, H. Bagherinezhad, J. Redmon, R. Mottaghi, and A. Farhadi, "Who let the dogs out? modeling dog behavior from visual data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4051–4060.

[47] L. Wang, K. Zhou, A. Chu, G. Wang, and L. Wang, "An improved light-weight traffic sign recognition algorithm based on yolov4-tiny," *IEEE Access*, vol. 9, pp. 124 963–124 971, 2021.

[48] G.-J. Han, R.-J. Wang, Q.-W. Yuan, and S.-D. Li, "Detection of bird nests on transmission towers in aerial images based on improved yolov5s," *Machines*, vol. 11, no. 2, p. 187, 2023.

[49] S.-C. Lin, "Automated garbage classification and retrieval system based on yolov4-tiny and depth image information," PhD thesis, Feng Chia University, Taiwan, 2023.

[50] J. Pfaff, C. Schutz, J. Baur, T. Buschmann, and H. Ulbrich, "A modular robot system for agricultural applications," in *IFAC-PapersOnLine*, vol. 49. Elsevier, 2016, pp. 184–189.

[51] M. Barbosa Júnior, R. Santos, L. Sales, and L. Oliveira, "Advancements in agricultural ground robots for specialty crops: An overview of innovations, challenges, and prospects," *Plants*, vol. 13, no. 23, p. 3372, 2024.

[52] R. Sparrow and M. Howard, "Robots in agriculture: Prospects, impacts, ethics, and policy," *Precision Agriculture*, vol. 22, pp. 818–833, 2021.

[53] L. Zu, Y. Zhao, J. Liu, F. Su, Y. Zhang, and P. Liu, "Detection and segmentation of mature green tomatoes based on mask r-cnn with automatic image acquisition approach," *Sensors*, vol. 21, no. 23, p. 7842, 2021.

[54] X. Xu, Y. Wang, and Y. Jiang, "End-effectors developed for citrus and other spherical crops," *Applied Sciences*, vol. 12, no. 15, p. 7945, 2022.

[55] C. Tawk, A. Gillett, M. Panhuis, G. Spinks, and G. Alici, "A 3d-printed omni-purpose soft gripper," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1268–1275, 2019.

[56] M. Javaid, A. Haleem, R. Singh, and R. Suman, "Enhancing smart farming through the applications of agriculture 4.0 technologies," *International Journal of Intelligent Networks*, vol. 3, pp. 150–164, 2022.

[57] S. Chouhan, U. Singh, and S. Jain, "Introduction to computer vision and drone technology," in *Applications of Computer Vision and Drone Technology in Agriculture 4.0*. Springer, 2024.

[58] H. Godfray, J. Beddington, I. Crute, L. Haddad, D. Lawrence, J. Muir, J. Pretty, S. Robinson, S. Thomas, and C. Toulmin, "Food security: the challenge of feeding 9 billion people," *Science*, vol. 327, no. 5967, pp. 812–818, 2010.

[59] M. Waleed, T. Um, T. Kamal, A. Khan, and A. Iqbal, "Determining the precise work area of agriculture machinery using internet of things and artificial intelligence," *Applied Sciences*, vol. 10, no. 10, p. 3365, 2020.

[60] M. Zude-Sasse, S. Fountas, T. Gemtos, and N. Abu-Khalaf, "Applications of precision agriculture in horticultural crops," *European Journal of Horticultural Science*, vol. 81, no. 2, pp. 78–90, 2016.

[61] A. Ranjan, R. Machavaram, and P. Patidar, "Design and development of a peduncle-holding end effector for robotic harvesting of mango," *Cogent Engineering*, vol. 11, no. 1, p. 2403706, 2024.

[62] R. Goulart, D. Jarvis, and K. Walsh, "Evaluation of end effectors for robotic harvesting of mango fruit," *Sustainability*, vol. 15, no. 8, p. 6769, 2023.

[63] S. Pitla, "Agricultural robotics," in *Advances in Agricultural Machinery and Technologies*. CRC Press, 2018, pp. 157–177.

[64] F. Meng, J. Li, Y. Zhang, S. Qi, and Y. Tang, "Transforming unmanned pineapple picking with spatio-temporal convolutional neural networks," *Computers and Electronics in Agriculture*, vol. 214, p. 108298, 2023.

[65] B. Zhang, Y. Xie, J. Zhou, K. Wang, and Z. Zhang, "State-of-the-art robotic grippers, grasping and control strategies, and their applications in agricultural robots: A review," *Computers and Electronics in Agriculture*, vol. 177, p. 105694, 2020.

[66] J. Rong, J. Fu, Z. Zhang, J. Yin, Y. Tan, T. Yuan, and P. Wang, "Development and evaluation of a watermelon-harvesting robot prototype: Vision system and end-effector," *Agronomy*, vol. 12, no. 11, p. 2836, 2022.

[67] E. Van Henten, B. Van Tuijl, J. Hemming, J. Kornet, J. Bontsema, and E. Van Os, "Field test of an autonomous cucumber picking robot," *Biosystems Engineering*, vol. 86, no. 3, pp. 305–313, 2003.

[68] Q. Feng, W. Zou, P. Fan, C. Zhang, and X. Wang, "Design and test of robotic harvesting system for cherry tomato," *International Journal of Agricultural and Biological Engineering*, vol. 11, no. 1, pp. 96–100, 2018.

[69] A. Bordoloi, "Revolutionizing agriculture: Harnessing the power of robotics," in *Smart Agriculture: Digital Era in FarmingEdition*. Springer, 2023.

[70] D. Dutta and A. Bera, "Nano fertilizer on sustainable agriculture–a review," *International Journal of Environment and Climate Change*, vol. 11, no. 8, pp. 1–5, 2021.

[71] K. Aravind, P. Raja, and M. Pérez-Ruiz, "Task-based agricultural mobile robots in arable farming: A review," *Spanish Journal of Agricultural Research*, vol. 15, no. 1, pp. e02R01–e02R01, 2017.

[72] P. Paul, R. Sinha *et al.*, "Agricultural robots: The applications of robotics in smart agriculture—towards more advanced agro informatics practice," *Asian Review of Mechanical Engineering*, vol. 9, no. 1, pp. 38–44, 2020.

[73] V. Hunt, *Artificial Intelligence & Expert Systems Sourcebook*. Springer Science & Business Media, 2012.

[74] E. Guardo *et al.*, "A fog computing-based iot framework for precision agriculture," *Journal of Internet Technology*, vol. 19, no. 5, pp. 1401–1411, 2018.

[75] K. Pothuganti, M. Jariso, and P. Kale, "A review on geo mapping with unmanned aerial vehicles," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 5, 2017.

[76] F. Dos Santos *et al.*, "Towards a reliable robot for steep slope vineyards monitoring," *Journal of Intelligent & Robotic Systems*, vol. 83, pp. 429–444, 2016.

[77] D. Tanda, "Semantic obstacle avoidance of robotic arm for fruit harvesting," Master's Thesis, Politecnico di Torino, 2024.

[78] L. He and J. Schupp, "Sensing and automation in pruning of apple trees: A review," *Agronomy*, vol. 8, no. 10, p. 211, 2018.

[79] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik, "Intel realsense stereoscopic depth cameras," *CoRR*, vol. abs/1705.05548, 2017.

[80] F. Xiao, H. Wang, Y. Xu, and R. Zhang, "Fruit detection and recognition based on deep learning for automatic harvesting: An overview and review," *Agronomy*, vol. 13, no. 6, p. 1625, 2023.

[81] L. Pedrosa, B. de Almeida Moreira, and C. Martins, "Optimization of harvesting and drying techniques for quality seed production in specialty crops: A systematic review and meta-analysis," *Agronomy*, vol. 14, no. 8, p. 1705, 2024.

[82] C. Cheng, J. Fu, H. Su, and L. Ren, "Recent advancements in agriculture robots: Benefits and challenges," *Machines*, vol. 11, no. 1, p. 48, 2023.

[83] P. Eizentals and K. Oka, "3d pose estimation of green pepper fruit for automated harvesting," *Computers and Electronics in Agriculture*, vol. 128, pp. 127–140, 2016.

[84] Research and Markets, "Harvesting robot market report 2025," https://www.researchandmarkets.com/reports/6076386/harvesting-robot-market-report, 2025, accessed July 2025.

[85] The Business Research Company, "Harvesting robot market trends report 2025 and insights to 2034," https://www.thebusinessresearchcompany.com/market-insights/harvesting-robot-market-overview-2025, 2025, accessed July 2025.

[86] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, 2018.

[87] Z. Zhang, Y. He, Z. Jiang, Y. Liu, and S. Liu, "Deep learning for time series anomaly detection: A survey," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–38, 2023.

[88] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, "Deepfruits: A fruit detection system using deep neural networks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3622–3628.

[89] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010.

[90] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Prentice Hall, 2002.

[91] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge university press, 2003.

[92] Canonical, "Enable gpu acceleration for ubuntu on wsl 2 with the nvidia cuda platform," https://ubuntu.com/blog/gpu-acceleration-on-wsl-2-with-nvidia-cuda, canonical Ubuntu Documentation. Last updated February 24, 2025. Accessed July 2025.

[93] Ultralytics, "Ultralytics yolov8 documentation," https://docs.ultralytics.com/, ultralytics. Accessed July 2025.

[94] M. Yaseen, "What is yolov8: An in-depth exploration of the internal features of the next-generation object detector," *arXiv preprint arXiv:2408.15857*, 2024.

[95] J. Solawetz and Francesco, "What is yolov8? a complete guide," Roboflow Blog, https://blog.roboflow.com/what-is-yolov8, 2024, accessed July 2025.

[96] LabelImg Contributors, "Labelimg: A graphical image annotation tool," https://github.com/tzutalin/labelImg, gitHub. Accessed July 2025.

[97] Python Software Foundation, "Welcome to python.org," https://www.python.org/, accessed July 2025.

[98] Intel, "Intel realsense sdk 2.0," https://www.intelrealsense.com/sdk-2/, accessed July 2025.

[99] OpenCV, "Open source computer vision library (opencv)," https://opencv.org/, openCV. Accessed July 2025.

[100] Wikipedia Contributors, "Intel realsense," https://en.wikipedia.org/wiki/Intel_RealSense, wikipedia. Accessed July 2025.

[101] FRAMOS, "How industrial depth cameras innovate agricultural automation," https://www.framos.com/en/news/how-industrial-depth-cameras-innovate-agricultural-automation, fRAMOS. Accessed July 2025.

[102] Tower Pro, "Mg996r digital servo specification," https://www.towerpro.com.tw/product/mg996r/, accessed July 2025.

[103] Arduino, "How to control servo motors with arduino," https://docs.arduino.cc/learn/electronics/servo-motors, arduino Documentation. Accessed July 2025.

[104] J. Terven and D. Cordova-Esparza, "A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas," *arXiv preprint arXiv:2304.00501*, 2023.

[105] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," https://github.com/ultralytics/ultralytics, 2023, accessed July 2025.

[106] K. Forsberg, H. Mooz, and H. Cotterman, *Visualizing Project Management: Models and Frameworks for Mastering Complex Systems*, 3rd ed. Wiley, 2005, covers the V-Model and its application in systems engineering and validation.

[107] I. Gibson, D. W. Rosen, and B. Stucker, *Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing*, 3rd ed. Springer, 2021, covers FDM, SLA, SLS, DLP, and other additive manufacturing technologies.

[108] Formlabs, "Stereolithography (sla) 3d printing," https://formlabs.com/blog/what-is-stereolithography-3d-printing/, accessed July 2025.

[109] Hubs, "Selective laser sintering (sls) 3d printing," https://www.hubs.com/knowledge-base/selective-laser-sintering-sls/, accessed July 2025.

[110] Formlabs, "What is dlp 3d printing?" https://formlabs.com/blog/what-is-digital-light-processing-dlp-3d-printing/, accessed July 2025.

[111] 3Dnatives, "What is binder jetting?" https://www.3dnatives.com/en/binder-jetting-3d-printing-technology/, accessed July 2025.

[112] Hubs, "Material jetting (mj) 3d printing," https://www.hubs.com/knowledge-base/material-jetting/, accessed July 2025.

[113] Bambu Lab, "Bambu lab p1s 3d printer specifications and features," https://www.bambulab.com/en/p1s, accessed July 2025.