

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Vision-based Robotic Manipulator for Arbutus Berry Harvesting

Joana Catarina de Sousa Pereira



Mestrado em Engenharia Eletrotécnica e de Computadores

Orientador: Filipe Baptista Neves dos Santos

July 28, 2025

Resumo

A crescente procura global de uma produção alimentar sustentável, num contexto de escassez de mão de obra, desafios climáticos e recursos limitados, tem estimulado avanços significativos na robótica agrícola. Este trabalho explora o desenvolvimento de um manipulador robótico baseado na visão para a colheita automática de medronhos. O trabalho é motivado pela importância ecológica e económica do medronheiro e pelos desafios associados à sua colheita manual, incluindo copas densas e maturação variável dos frutos. Este estudo descreve a capacidade dos sistemas robóticos para enfrentar estes desafios através de uma maior eficiência, precisão e sustentabilidade ambiental.

A metodologia proposta envolve uma abordagem em várias fases: recolha de dados ambientais utilizando um sensor *Light Detection and Ranging* (LiDAR) e câmaras RGB-D; desenvolvimento e controlo de um manipulador robótico adaptável à estrutura da árvore; e implementação de um sistema de perceção com planeamento da trajetória de colheita. Este sistema combina técnicas de visão por computador e os modelos de aprendizagem profunda que permite a deteção e localização precisa dos frutos, a avaliação do estado de maturação e a colheita selectiva. Adicionalmente, é proposta uma estrutura do manipulador e técnicas de apanha capazes de se adaptarem à morfologia única do medronheiro, com o objetivo de minimizar os danos nas plantas e otimizar a eficiência da colheita.

Esta solução robótica permite colmatar a falta de mão de obra, melhorar a eficiência da colheita e promover práticas ambientalmente sustentáveis através de métodos de colheita precisos e selectivos. Os resultados desta investigação fazem avançar o campo da robótica agrícola, particularmente para culturas não convencionais como o medronheiro, ao mesmo tempo que oferecem uma abordagem escalável para desafios semelhantes na automação agrícola.

Abstract

The increasing global demand for sustainable food production amidst labour shortages, climate challenges, and limited resources has spurred significant advancements in agricultural robotics. This work explores the development of a vision-based robotic manipulator for the automated harvesting of arbutus berries. The work is motivated by the ecological and economic significance of the Arbutus tree and the challenges associated with its manual harvesting, including dense canopies and variable fruit ripeness. This study outlines the potential of robotic systems to address these challenges through enhanced efficiency, precision, and environmental sustainability.

The proposed methodology involves a multi-stage approach: collection of environmental data using a Light Detection and Ranging (LiDAR) sensor and RGB-D cameras; development and control of a robotic manipulator adaptable to the structure of the tree; and implementation of a perception system with harvest trajectory planning. This system combines computer vision techniques and deep learning models to enable precise fruit detection and localization, ripeness assessment, and selective harvesting. In addition, a manipulator structure and picking techniques are proposed that can adapt to the unique morphology of the strawberry tree, with the aim of minimizing damage to the plants and optimizing harvesting efficiency.

This robotic solution addresses labor shortages, enhances harvesting efficiency, and promotes environmentally sustainable practices through precise and selective harvesting methods. The outcomes of this research advance the field of agricultural robotics, particularly for unconventional crops like arbutus berries, while offering a scalable approach to similar challenges in agricultural automation.

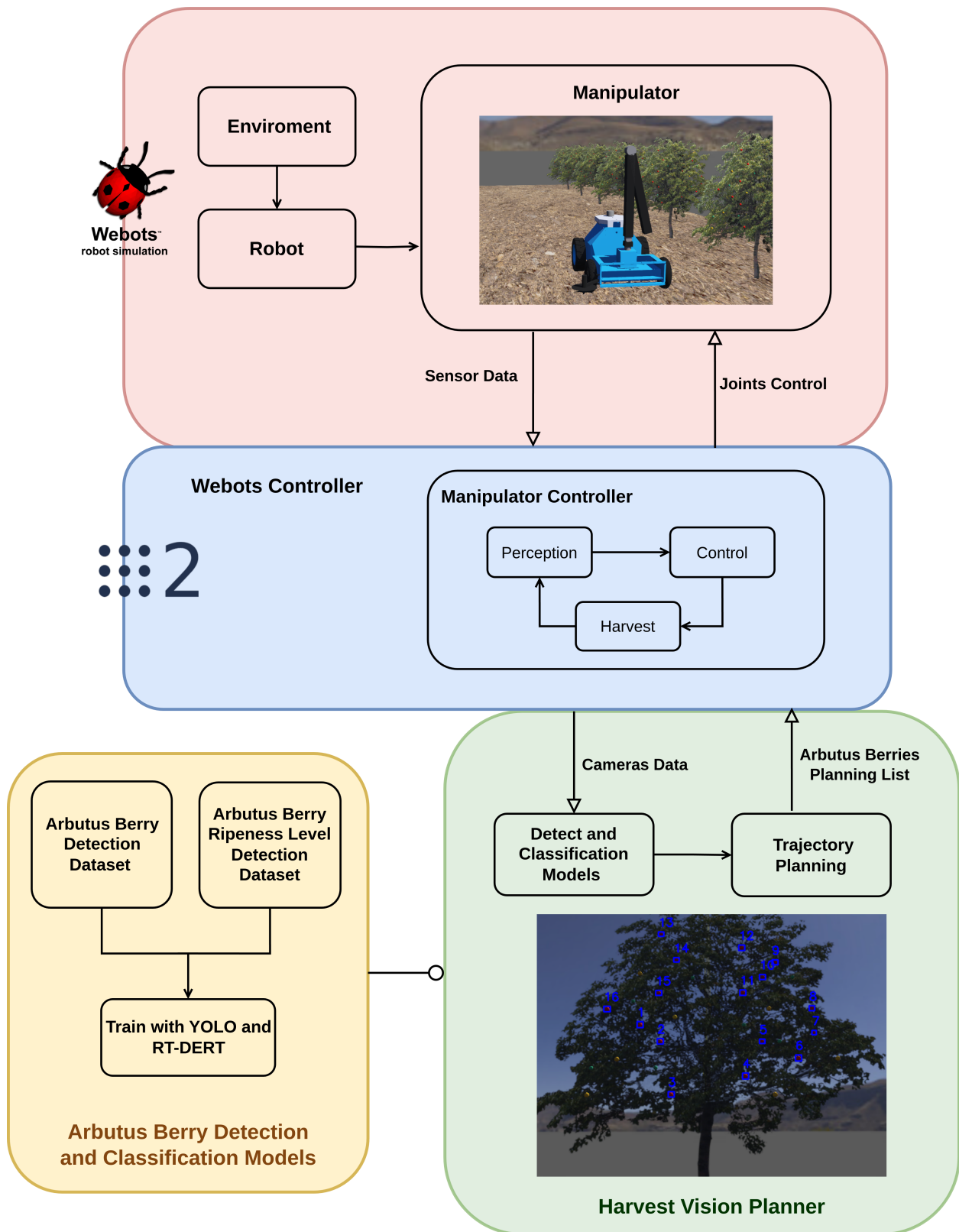


Figure 1: Graphical Abstract

UN Sustainable Development Goals

The United Nations Sustainable Development Goals (SDGs)¹ define 17 global objectives aimed at promoting prosperity, protecting the planet, and ensuring peace and justice for all. In the context of this work, there is a direct contribution to several of these goals, particularly in the areas of sustainable agriculture, technological innovation, and adaptation to climate change.

The research developed integrates innovative solutions based on automation, robotics, and artificial intelligence, promoting more efficient, resilient, and sustainable agricultural practices. This multidisciplinary approach not only boosts the productivity and efficiency of agricultural systems but also contributes to waste reduction, sustainable management of natural resources, and increased resilience to climate challenges.

In context, this project has the potential to contribute of the following SGD:

SDG 2 Zero Hunger: The robotic manipulator increases productivity and efficiency in food production by automating the selective harvesting of arbutus berries, helping address labor shortages and supporting sustainable agricultural practices

SDG 8 Decent Work and Economic Growth: By introducing advanced automation and robotics in agriculture, the project promotes technological innovation, increases operational efficiency, and reduces the need for physically demanding manual labor, thus supporting economic growth

SDG 9 Industry, Innovation and Infrastructure: The development and integration of novel robotic and computer vision systems contribute to scientific research and technological advancement in precision agriculture, fostering innovation and building research infrastructure

SDG 12 Responsible Consumption and Production: Selective and precise harvesting enabled by robotics reduces food waste and ensures efficient use of natural resources, aligning agricultural production with sustainability goals.

SDG 13 Climate Action: Automation and precision technologies make agricultural systems more resilient to climate-related challenges and resource limitations, supporting adaptation and sustainable practices in response to climate change

¹<https://sdgs.un.org/goals>

SGD	Target	Contribution	Performance Indicators and Metrics
2	2.4	Automation of harvesting and the use of precision technologies (LiDAR sensors, RGB-D cameras, computer vision) promote resilient food production systems, increase resource-use efficiency, improve the sustainability of agricultural systems, and increase productivity.	Increase in sustainable management of agricultural area; reduction in resource use (water, labor); improvement in productivity and yield; performance of detection and harvesting metrics.
8	8.2	Technological innovation in agriculture (robotics, AI, automation) increases economic productivity through diversification, technological upgrading, and innovation.	Number of technological innovations implemented; increase in operational efficiency; reduction in production costs.
9	9.5	Research and development in agricultural robotics contribute to scientific advancement and foster innovation in sustainable agriculture.	Number of scientific publications, open datasets, and research collaborations; adoption of new technologies in agriculture.
12	12.2	Selective and precise harvesting reduces waste and promotes the sustainable management and efficient use of natural resources.	Reduction in food waste; percentage of fruits harvested at optimal ripeness; minimization of plant damage.
13	13.1	Automation increases the resilience of agricultural systems to climate-related hazards and supports adaptation strategies.	Increase in resilience metrics; ability to maintain productivity under adverse conditions; reduction in labor shortages.

Agradecimentos

Primeiro, gostaria de agradecer ao meu orientador Filipe Neves dos Santos, por confiar nas minhas capacidades e me motivar nos momentos mais desafiantes. A sua orientação e apoio ao longo do desenvolvimento desta dissertação, bem como a sua experiência e visão crítica e realista do projeto, foram cruciais para que todos os objetivos definidos fossem alcançados.

Agradeço ao meu coorientador André Baltazar, por ter acreditado em mim, pela colaboração e acompanhamento, pela motivação constante e pela disponibilidade constante para esclarecer dúvidas. Sem a sua ajuda, este projeto não teria alcançado o mesmo sucesso.

Agradeço à Isabel Pinheiro e ao Daniel Queirós da Silva, pela colaboração, dedicação e apoio no acompanhamento deste projeto.

Agradeço a todos os elementos da equipa TRIBE-Laboratório de Robótica e IoT para Agricultura e Floresta de Precisão Inteligente, pela forma como me acolheram e integraram, bem como pela disponibilidade e apoio prestados sempre que precisei.

Agradeço ao Tiago e aos meus amigos por estarem sempre presentes nos bons e maus momentos e me ajudarem a manter o equilíbrio entre o trabalho e a descontração.

Por fim, agradeço à minha família por sempre acreditarem nas minhas capacidades e pelo apoio incondicional que me deram ao longo destes últimos cinco anos.

Joana Pereira

*“Valeu a pena? Tudo vale a pena
Se a alma não é pequena.”*

Fernando Pessoa

Contents

1	Introduction	1
1.1	Contributions and Document Organization	2
2	Background	5
2.1	Overview of Robotic Applications in Agriculture	5
2.2	Advantages of Robotics in Agriculture	6
2.3	Overview of the Arbutus Berry Harvest and functional requirements	6
2.4	Support Technologies	8
2.4.1	Sensors and Operating Principles	8
2.4.2	Computer Vision	9
2.4.3	Image Annotation Tools	10
2.4.4	Integration and Control	10
3	State of the art	11
3.1	Harvesting Manipulators	11
3.1.1	Robots for apple harvesting	11
3.1.2	Robots for tomato harvesting	13
3.1.3	Robots for strawberry harvesting	14
3.1.4	Robots for kiwi harvesting	15
3.2	End Effectors for Agricultural Robotic Harvesting	17
3.3	Visual perception	20
3.3.1	Visual Sensing Technology	20
3.3.2	Object Recognition Methods	22
3.4	Review of the Research	24
4	Proposed System Architecture	27
4.1	System Architecture	27
4.2	Challenges from the Robotics Point of View	29
4.3	Requirements for the Perception and Harvesting System	30
4.4	Conclusion	31
5	Perception System	33
5.1	Materials and Methods	33
5.1.1	Data Collection	33
5.1.2	Data Labelling and Augmentation	34
5.1.3	Data Generation	37
5.1.4	Deep Learning Models	38
5.1.5	Training Configuration	39

5.2	Results	42
5.2.1	Methodology	42
5.2.2	Detection for Arbutus Berry Ripeness Stage	43
5.2.3	Detection and classification of Different Classes of Arbutus Berry Ripeness	47
5.2.4	Image Splitting	51
5.3	Discussion	57
6	ArBeH-Harvesting robotic tool	59
6.1	Simulation Environment	59
6.2	Manipulator Design	60
6.2.1	Angle Calculation	62
6.2.2	Camera	65
6.3	Manipulator Controller	67
6.3.1	Perception	69
6.3.2	Control and Harvesting	71
6.4	Results and Discussions	72
7	Conclusion and Future Work	75
	References	79

List of Figures

1	Graphical Abstract	iv
1.1	Arbutus tree (Arbutus unedo).	2
2.1	Arbutus berry.	7
2.2	LiDAR sensor [10].	8
2.3	RGB-D camera [12].	9
3.1	The proposed robot for apple harvesting [18].	12
3.2	SCARA Manipulator for Tomato Harvesting [19].	13
3.3	Strawberry harvesting manipulator [20].	14
3.4	Kiwi harvesting robot [21].	16
3.5	Fruit-holding grippers [23].	18
3.6	Stem-holding grippers [24].	18
3.7	Air Suction [25].	18
3.8	Suction Device [26].	19
3.9	Rotation Mechanisms [26].	19
3.10	Scissors/Saw-like Tools [26].	20
4.1	ArBeH system architecture composed of three main blocks: the simulated environment in Webots, the Webots-ROS2 communication and control bridge, and the Harvest Vision Planner, which integrates perception, detection, and trajectory planning for automated Arbutus berry harvesting.	28
4.2	Simulation of the ArBeH manipulator in the Webots virtual environment and the displays of the views from the two onboard cameras.	28
5.1	Setups of sensors used in data collection, including the OAK-D S2 RGB-D sensor and the Livox Mid-360 LiDAR sensor.	34
5.2	Example of manual annotations using CVAT for the Arbutus Berry Ripeness Level Detection Dataset [4]	35
5.3	Workflow for detection and classification of Arbutus berries.	36
5.4	Training metric evolution for YOLO models with different optimization strategies.	40
5.5	Training metric evolution for RT-DETR models with different optimization strategies.	41
5.6	Example of Arbutus berries detection for each class.	44
5.7	Examples of false negatives in the detection of Arbutus berries.	47
5.8	Example of Arbutus berries detection for each class using image splitting.	52
6.1	Simulation environment of Arbutus tree plantation.	59
6.2	Manipulator structure.	61

6.3	Coordinate frames attached to the manipulator.	62
6.4	Geometric representation of the manipulator for inverse kinematics calculations. .	63
6.5	Depth estimation geometry using a tilted camera and image projection.	66
6.6	Manipulator controller state machine. The blue path illustrates the sequence of states followed during the suction harvesting method, while the red path shows the sequence for the cutting and gripping method.	68
6.7	Optimal harvesting sequence of detected berries.	70

List of Tables

5.1	Number of annotations for each class in the original dataset.	35
5.2	Augmentation operations applied to the original images.	37
5.3	Distribution of the number of images per set after image splitting.	38
5.4	Number of epochs each model was trained for the different tasks.	42
5.5	Number of epochs each model was trained for the different tasks with the image splitting dataset.	42
5.6	Results on the performance metrics for Arbutus berry ripeness stage detection. . .	45
5.7	Detection results of Arbutus berry ripeness stage.	46
5.8	Results on the performance metrics for Arbutus berry detection.	48
5.9	Detection results of Arbutus berry.	48
5.10	Results on the performance metrics for Arbutus berry classification.	49
5.11	Classification results of Arbutus berry.	49
5.12	Results on the performance metrics for Arbutus berry classification and detection.	50
5.13	Detection and classification results of Arbutus berry.	51
5.14	Results on the performance metrics for Arbutus berry ripeness stage detection with image splitting.	53
5.15	Detection results of Arbutus berry ripeness stage with image splitting.	54
5.16	Results on the performance metrics for Arbutus berry detection with image splitting.	55
5.17	Detection results of Arbutus berry with image splitting.	55
5.18	Results on the performance metrics for Arbutus berry classification and detection with image splitting.	56
5.19	Detection and classification results of Arbutus berry with image splitting.	57
6.1	Forward Kinematic.	62
6.2	Performance of harvesting methods with and without trajectory planning, in seconds.	72

List of Acronyms

Adam	Adaptive Moment Estimation
AdamW	Adam with Weight Decay
AP	Average Precision
ArBeH	Arbutus Berries Harvest
CNN	Convolutional Neural Network
CVAT	Computer Vision Annotation Tool
DDS	Data Distribution Service
DoF	Degrees of Freedom
ETFA	Emerging Technologies and Factory Automation
FoV	Field of View
FP	False Positive
FN	False Negative
GELAN	Generalized Efficient Layer Aggregation Network
HVP	Harvest Vision Planing
HSV	Hue, Saturation, and Value
IoU	Interception over Union
LiDAR	Light Detection and Ranging
mAP	mean Average Precision
OpenCV	Open Source Computer Vision Library
PGI	Programmable Gradient Information
PID	Proportional-Integral-Derivative
PRR	Prismatic-Revolute-Revolute
PSA	Partial Self-Attention
R-CNN	Region-Based Convolutional Neural Networks
RRR	Revolute-Revolute-Revolute
RPN	Region Proposal Network
ROS2	Robot Operating System 2
RT-DETR	Real-Time Detection Transformer
s	Seconds
SCARA	Selective Compliance Assembly Robot Arm
SCDown	Spatial-Channel Decoupled Downsampling
SGD	Stochastic Gradient Descent
SPPF	Spatial Pyramid Pooling - Fast
TP	True Positive
ToF	Time of Flight
TPU	Tensor Processing Unit
YOLO	You Only Look Once

Chapter 1

Introduction

The quest to ensure sustainable food production worldwide, with dwindling resources, a worsening climate, and fewer people willing to work in agriculture, has become one of the major driving forces for innovation and technological development in the last decade. The use of robotics in agriculture and forestry has become an essential tool to respond to these kinds of challenges. Robotics marks an evolution in cultivation and harvesting methods, reducing the dependence on intensive manual labor and contributing to more sustainable production that is less stressful on the environment [1].

Robotic applications in agriculture span a wide range of tasks, including automated harvesting, planting, seeding, and land preparation. Among these, automated harvesting presents particularly significant challenges. Robotic systems must accurately identify and locate fruit, distinguish between different stages of ripeness, and select only those fruits ready for harvest. These systems must deal with adverse conditions common in agricultural environments, such as occlusions caused by leaves and branches and variations in natural lighting. To address these challenges, efficient visual perception algorithms are needed for real-time execution on robotic systems. In addition to direct application in harvesting, these algorithms can provide processed visual data that can then be analyzed. Automatic fruit detection, even in the early stages of growth, is a valuable resource for yield estimation and monitoring crop development over time, supporting agronomic decision-making based on detailed information [2].

The dissertation focuses on the automated harvesting of Arbutus berry (*Arbutus Unedo*). This plant is a kind of bush that is valued both for its ecological importance and for its fruit. The arbutus berry serves as raw material for a range of products such as arbutus brandy (“*medronheira*”), honey, and jams. It is also used for pastries and other culinary applications due to its antioxidant properties. This bush’s high resilience to fire, rapid regeneration capacity, and capability to adapt to adverse soil conditions also contribute to the plant’s environmental value, as arbutus can help create fields and forests that are more resilient to wildfires.

Regarding the morphological characteristics of the arbutus tree, its canopy is dense and rounded (Figure 1.1), making it more of a challenge to harvest when compared to other fruits. This prevents pre-existing harvesting robots from being easily adapted to the arbutus tree, driving the need for

the development of dedicated solutions for mechanized harvesting of the arbutus berry.

The main aim of this work is to study and propose a conceptual solution for a vision-based Robotic Manipulator for Arbutus Berry Harvesting, which can be subdivided into the following specific objectives:

- Collect a real data set of the Arbutus Berry tree, with a 3D LiDAR sensor and RGB-D camera;
- Study and implement a simulated robotic environment using TRIBE LAB robot designs;
- Study and propose a manipulator for harvesting the Arbutus Berry;
- Study, propose, and develop a perception system for Arbutus Berry detection;
- Test and validate the proposed system in a simulated environment.

Furthermore, the project also requires a robotic manipulator capable of adapting to the bush structure of the arbutus tree, allowing for fruit harvesting without damaging the plant. Additionally, implementing a perception system capable of accurately determining fruit ripeness is essential to selectively harvest only fully ripe berries. A robotic system with these characteristics not only helps to enable mechanized harvesting in less conventional crops (such as arbutus and other plants with similar morphology) but can also address labor shortages and help improve yields sustainably.



Figure 1.1: Arbutus tree (*Arbutus unedo*).

1.1 Contributions and Document Organization

The main contributions of this work are the release of two fully annotated and open-access datasets, Arbutus Berry Detection Dataset [3] and Arbutus Berry Ripeness Level Detection Dataset [4], developed through a structured manual labeling process provides a valuable resource for the

research community working on fruit detection and classification in challenging agricultural environments. A specialized algorithm was developed to control the robotic manipulator, carefully adapted to deal with the unique morphological challenges of the strawberry tree. In addition, a real-time perception and planning algorithm (Harvest Vision Perception) was implemented, capable of detecting and classifying *Arbutus* berries at different stages of ripening, as well as autonomously and efficiently planning selective harvesting trajectories. This integrated approach allows the robotic system to operate fully autonomously, optimizing harvesting efficiency and ensuring careful fruit selection, even in complex and dynamic scenarios. Finally, a scientific article focusing on the perception system (*Arbutus Berry Detection and Classification for Harvesting*) was peer-reviewed and accepted for presentation at the 30th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2025).

The document is composed of six distinct chapters:

- Chapter 2 presents the main background concepts relevant to the development of a robotic manipulator for *arbutus* berry harvesting. This includes an overview of robotics applications in agriculture, the benefits of robotic automation, the specific challenges associated with *arbutus* berry harvesting, and the functional requirements for the perception and harvesting systems. The section also introduces the main enabling technologies—such as sensors, computer vision, and simulation platforms—that support this project.
- Chapter 3 provides a comprehensive literature review on the state-of-the-art in agricultural robotics. The discussion focuses on harvesting manipulators and visual perception systems, with particular emphasis on solutions for apples, tomatoes, strawberries, and kiwis. The chapter also reviews the types of end effectors typically used in robotic harvesting, as well as visual perception technologies and object recognition methods.
- Chapter 4 details the architecture of the proposed system for automated *arbutus* berry harvesting and defines the main functional requirements.
- Chapter 5 describes the perception system, including the procedures for data collection, dataset annotation and augmentation, and the development and training of deep learning models for fruit detection and ripeness classification. The chapter also presents and discusses the results obtained from the evaluation of different models and approaches.
- Chapter 6 details the manipulator design, starting with the creation of a simulation environment. It describes the kinematic structure of the manipulator, the integration of visual and proximity sensors, and the two harvesting methods considered (suction-based and cutting/gripping). The chapter also presents the results obtained in the simulations and includes a discussion of the different approaches used.
- Chapter 7 presents the main conclusions of the dissertation and discusses future work, highlighting the contributions of the developed system to the field of agricultural robotics and identifying directions for further research and practical implementation.

Chapter 2

Background

This chapter presents the foundational elements necessary for developing a robotic manipulator for arbutus berry harvesting. It discusses the applications and advantages of robotics in agriculture, the specific challenges associated with harvesting arbutus berries, and the requirements for perception and harvesting systems. Finally, it describes the main supporting technologies, including sensors, computer vision, and integration and simulation platforms.

2.1 Overview of Robotic Applications in Agriculture

The continuous advancements in agricultural robotics are designed to meet the challenges posed by population growth, rapid urbanization, intense competition for high-quality products, environmental preservation, and a shortage of skilled labor. As the need for greater efficiency and sustainability in food production grows, the integration of robotics in agriculture becomes increasingly vital. In the context of labour shortages, robotic systems provide innovative solutions for various agricultural practices to try and satisfy the demand for greater yields [5].

Below are some of the primary applications of robotics in agriculture, referenced from [5]:

- **Land Preparation** — Robotic systems efficiently prepare the land for planting by performing tasks such as tilling and fertilization;
- **Planting and Seeding** — Automated planting robots optimize seed placement by adjusting depth and spacing based on specific crop requirements;
- **Plant Treatment** — Equipped with advanced cameras and sensors, robots can detect diseases and pests, applying pesticides and fertilizers selectively;
- **Automated Harvesting** — Specialized harvesting robots utilize robotic arms and vision algorithms to identify and selectively harvest delicate fruits and vegetables, such as strawberries and lettuce;

- **Monitoring and Phenotyping** – Robotics plays a crucial role in monitoring plant health and estimating yields. By employing computer vision, these systems collect data on crop conditions, facilitating informed decision-making to improve productivity.

2.2 Advantages of Robotics in Agriculture

Robotics automation in agriculture supports continuous operation, accelerating harvest times and improving overall efficiency, an advantage that is difficult to achieve with manual labor alone. Increased productivity through automation is essential to meet the world's rising food demand.

Precision and quality are also greatly enhanced by robotics, thanks to advanced sensors and computer vision algorithms. These technologies enable real-time analyses of crops, accurately detecting diseases, pests, and fruit maturity levels. During harvesting, for instance, robots can selectively pick only ripe fruits, minimizing waste and ensuring that only high-quality produce reaches the market. Land preparation robots can operate with high precision, minimizing soil disruption, maximizing land use, and enhancing yields [6].

Environmental sustainability is another critical benefit of robotics in agriculture. By enabling the precise application of water, fertilizers, and pesticides, robots significantly reduce resource overuse and environmental contamination, thus decreasing the ecological footprint of farming practices. This targeted approach helps to conserve biodiversity and encourages more sustainable farming methods [1].

The agricultural and forestry sectors, like many others, depend on human labor to carry out tasks essential to production. However, society's search for a higher quality of life has shifted the focus to more qualified jobs. Given that agricultural and forestry work has low wages compared to other professions, these sectors face a labor shortage. Robotics plays a crucial role in solving this problem, as they take on physically exhausting tasks, such as harvesting. In regions where work is seasonal, robots provide a stable workforce, ensuring harvests are carried out at the optimal time without complete dependence on human workers [1].

Moreover, robotic systems equipped with sensors and monitoring tools gather valuable data on soil health, crop status, and environmental conditions. This data-driven approach enables farmers to make more informed decisions regarding planting, irrigation, harvesting, and treatment applications. By improving data management, robots help agricultural practices become more adaptable and responsive to variables such as climate change, pest infestations, and other unpredictable factors.

2.3 Overview of the Arbutus Berry Harvest and functional requirements

The *Arbutus unedo* is a small tree that usually grows up to five meters in height, typical of the Mediterranean region and parts of Western Europe. It is highly valued for its ecological and economic contributions. The tree produces small, round fruits, known as arbutus berries or *medronho*,

which are used in the production of various products such as [7] [8]:

- *Aguardente de Medronho*: A traditional Portuguese spirit distilled from fermented berries, widely appreciated for its distinctive flavor.
- Honey: Bees that forage in the Arbutus tree blossoms produce arbutus honey, noted for its unique, slightly bitter taste.
- Jams and Preserves: The berries are often used in homemade compotes and jams, valued for their sweet and tangy flavor.
- Pharmaceutical and Nutritional Applications: The fruit contains antioxidants and other bioactive compounds, contributing to health benefits and potential pharmaceutical applications.

The *Arbutus unedo* produces round fruits with a rough surface, which ripen in the autumn. This ripening period contrasts with the cycle observed in most native species of Portugal. One of the reasons for this discrepancy lies in the Arbutus tree's extended flowering period: its white-green or pinkish flowers only emerge as the fruits from the previous year are reaching maturity, persisting until February. As a result, flowers and fruits coexist in the autumn, necessitating careful harvesting to avoid damaging the flowers (Figure 2.1) [8].



Figure 2.1: Arbutus berry.

Ecologically, the *Arbutus unedo* plays a vital role in its native habitats. It supports biodiversity by providing food for wildlife and serving as a natural firebreak due to its high moisture content and resilience to wildfires. Its rapid regeneration after fires makes it a key species in ecosystem restoration. Moreover, the tree is an essential food source for pollinators during autumn and winter, when floral resources are limited [8].

The harvesting of arbutus berries is predominantly manual, with pickers collecting berries directly from the tree. While this approach allows for the careful selection of ripe fruits, it presents several challenges. The dense canopy often hampers access to inner or higher branches, and the variability in fruit size, maturity, and position complicates efficient collection. Moreover, the

berries must be picked one by one, without stems or leaves, and with care to avoid damaging nearby flowers that will produce future fruits. This level of delicacy means that a person typically takes between 3 to 5 seconds (*s*) to harvest and store each berry. As a result, the process is highly labor-intensive, requiring several dozen helpers during peak season to maintain productivity. This reliance on skilled manual labor makes the operation costly and time-consuming and increasingly unsustainable due to labor shortages. These limitations significantly reduce scalability and efficiency, underscoring the need for improved, automated harvesting methods.

Addressing these challenges requires innovative solutions and functional requirements that can improve efficiency, reduce costs, and maintain the quality of harvested berries. These solutions should consider the biological and physical characteristics of the tree and its fruit, as well as the socio-economic factors surrounding arbutus berry production.

2.4 Support Technologies

2.4.1 Sensors and Operating Principles

LiDAR

LiDAR (Figure 2.2) sensors are an advanced technology for creating precise 3D maps of the environment and for navigation and obstacle detection. These sensors emit laser pulses that, when they hit objects, return to the sensor, enabling precise identification of three-dimensional space. This process generates a dense point cloud that captures the shape, structure, and details of the environment with high precision. In the context of autonomous navigation, such as in agricultural and robotic vehicles, these sensors enable real-time terrain mapping, identifying topographic variations and obstacles [9].



Figure 2.2: LiDAR sensor [10].

RGB and RGB-D cameras

RGB cameras provide color images that are essential for visual analysis and segmentation of fruits, enabling the identification of characteristics such as color, shape, and texture through computer vision algorithms. In addition, RGB-D cameras integrate depth sensors that add a crucial dimension to image processing, enabling the precise location of fruits in three-dimensional

space. These operate using different principles, such as structured light, time of flight (ToF), or stereoscopy, generating not only color values for each pixel but also spatial coordinates. This capability is essential for robotic harvesting operations, as it allows calculating exact distances between the end effector and the target object, planning approach trajectories, and determining optimized harvesting sequences based on the spatial arrangement of the fruits [11].



Figure 2.3: RGB-D camera [12].

2.4.2 Computer Vision

Computer vision represents a fundamental component of this system, allowing the identification, location, and analysis of fruits in complex agricultural environments.

Open Source Computer Vision Library (OpenCV)

OpenCV is an open-source library for computer vision and machine learning that contains over 2,500 algorithms optimized for image and video analysis and processing. This library is used for image pre-processing, providing filtering techniques to reduce noise and enhance relevant features. In addition, the library supports basic segmentation operations, allowing the separation of objects and regions of interest within an image.

OpenCV is also widely used for data augmentation, a technique that generates new training samples from transformations of existing images, such as rotations, mirroring, and scale changes. These operations increase the diversity of the dataset, contributing to improving the performance of machine learning models [13].

Ultralytics

Ultralytics is a software company specializing in computer vision and artificial intelligence, dedicated to making model development more accessible, efficient to train, and simple to implement [14].

This framework offers:

- Development and improvement of You Only Look Once (YOLO) models for real-time object detection.
- Simplified tools for training, validation, testing
- Open source library for training and inference of detection, segmentation, and classification models.

2.4.3 Image Annotation Tools

Image annotation with the Computer Vision Annotation Tool (CVAT) plays a key role in creating robust datasets for training computer vision algorithms. This process involves accurately demarcating objects of interest through bounding boxes and segmentation masks, allowing models to learn relevant visual features. The robustness of the dataset is ensured by the diversity of scenarios represented in the annotated images, including variations in lighting, perspective, partial occlusion, and diverse contexts. CVAT facilitates this process by offering intuitive annotation tools, support for multiple annotation formats, and quality control functionalities. CVAT facilitates this process by offering intuitive annotation tools, support for multiple annotation formats, and quality control functionalities.

2.4.4 Integration and Control

Robot Operating System 2 (ROS2)

ROS2 is an open-source framework that provides libraries, tools, and conventions for robotics software development. It is designed to support real-time and safety-critical systems and diverse platforms ranging from embedded devices to large-scale distributed systems. ROS2's communication architecture is based on the Data Distribution Service (DDS) standard, which enables flexible, reliable, and decentralized publish-subscribe communication between nodes, even in dynamic or unreliable network environments. In this model, nodes publish messages to topics, and other nodes subscribe to those topics to receive data, facilitating modular and scalable integration of sensors, algorithms, and actuators in complex robotic systems. Additionally, ROS2's programming model encourages using independently executable nodes, which improves parallel development, testing, and component reuse. The active ROS2 community and mature ecosystem provide numerous ready-to-use implementations for various sensors and algorithms, significantly accelerating the development of advanced robotic applications [15].

Webots

Webots is an open-source robotic simulator that provides a realistic environment for prototyping, programming, and simulating robots. Developed by Cyberbotics, this software provides an advanced physics engine for accurate simulations of interactions with the environment and high-fidelity 3D rendering. The choice of Webots for designing and implementing a simulated robotic environment is based on its ability to model diverse sensors (cameras, LiDAR) and its native integration with ROS2, enabling the testing of algorithms in a controlled environment before physical implementation. In addition, its extensive library of pre-built models, intuitive graphical editors for building custom scenarios, and support for multiple programming languages (Python, C++, Java) significantly reduce development time, facilitating iterative validation of algorithms in controlled and reproducible conditions [16].

Chapter 3

State of the art

This chapter presents a comprehensive literature review on harvesting manipulators and visual perception systems in agricultural robotics. It begins by examining the main types of manipulators and end-effectors, discussing their design principles, operational characteristics, advantages, and limitations. Afterwards, the chapter explores the main visual perception technologies and object recognition techniques that underpin autonomous fruit harvesting. The review concludes with a critical assessment of these approaches, evaluating their suitability for addressing the specific challenges associated with Arbutus berry harvesting.

3.1 Harvesting Manipulators

Harvest manipulators are essential in agricultural robotics and are designed to enable efficient and precise harvesting operations. These systems must have sufficient range and Degrees of Freedom (DoF) to access the target object with the correct orientation while navigating through dynamic environments with diverse characteristics. Furthermore, it is necessary to ensure that the harvested product is handled delicately, avoiding damage during the harvesting and storage process [17].

3.1.1 Robots for apple harvesting

The authors of [18] developed a robotic system for apple harvesting, integrating three main components: an RGB-D camera for fruit detection and localization, a three DoF manipulator for positioning, and a vacuum-based end effector for fruit detachment (Figure 3.1a). The manipulator's design is simple, balancing mechanical capability with practical efficiency for orchard environments.

As illustrated in Figure 3.1b, the manipulator consists of two revolute joints and one prismatic joint. The revolute joints enable flexible horizontal and vertical movements, allowing the manipulator to navigate a dense canopy. These joints are driven by servo motors, ensuring precise and smooth rotations. The prismatic joint extends the manipulator's reach, allowing it to access apples located at varying distances within its harvesting range. This joint uses a pneumatically actuated

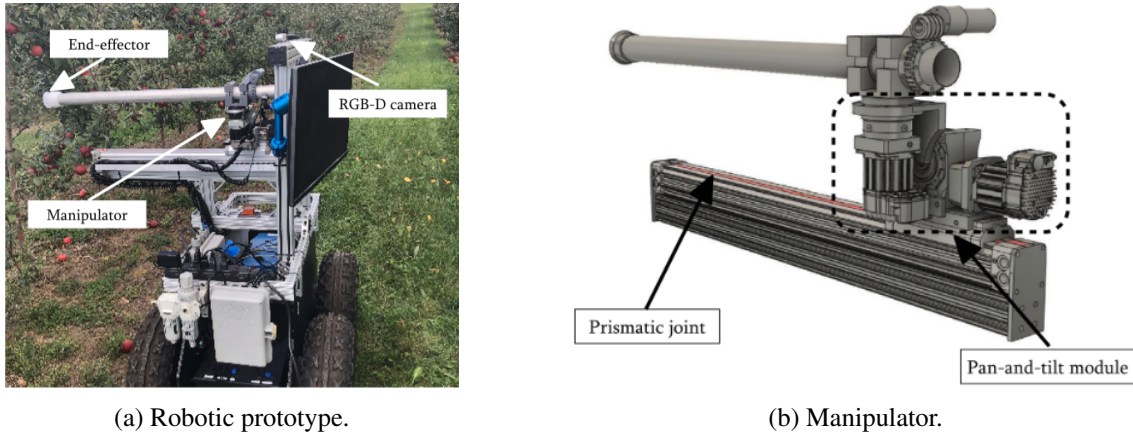


Figure 3.1: The proposed robot for apple harvesting [18].

Lintra rodless air cylinder, enabling rapid and reliable linear motion. The design supports high-speed movements while maintaining the precision required to reach apples at different heights and depths in the orchard canopy.

The simplicity of the three DoF configuration reduces the computational demands for motion control while maintaining sufficient dexterity for apple harvesting tasks. This efficient design ensures precise movements without the added complexity of industrial robots with higher DOF (i.e. 7 DoF).

An important system component is the end effector, responsible for gripping and detaching apples from trees while minimizing fruit damage. The end effector incorporates a soft and adaptive gripping mechanism that conforms to the apple's shape and size, providing a secure hold. Integrated sensors offer real-time feedback on contact forces and positioning, allowing the system to dynamically adjust to variations in apple size, stem resistance, and canopy structure. This adaptability ensures reliable performance across diverse orchard conditions.

Apple detection is performed using a method based on deep neural networks, using an architecture derived from Mask Region-Based Convolutional Neural Networks (R-CNN), adapted to improve segmentation and accuracy in orchard environments. The captured images are analyzed by a network that combines two modules: a feature extraction backbone based on the original Mask R-CNN architecture and an additional false positive suppression module (feature suppression). This combination allows the system to effectively distinguish apples from elements such as leaves, branches, and lighting variations. After identifying apples in bounding boxes, their three-dimensional locations are estimated using the depth map provided by the RGB-D camera. The Suppression Mask R-CNN architecture with the suppression module achieved an accuracy rate of 90.5% on the test set. Field tests were conducted in orchard settings, where the robot achieved an average harvesting efficiency of 82.47% in "human-in-the-loop picking experiments". During fully autonomous tests, the system demonstrated a harvesting efficiency of 64.06%, with an average cycle time of 8.8s per apple. The primary factors contributing to unsuccessful harvests were localization errors due to foliage occlusion and difficulties detaching apples from flexible stems.

For future work, the authors propose enhancing the perception system to address these challenges and introducing collaborative multi-arm systems to improve the efficiency and robustness of the harvesting process.

3.1.2 Robots for tomato harvesting

The authors of [19] presented a novel manipulator system designed specifically for the automated harvesting of tomatoes. The main goal was to address the unique challenges posed by the delicate nature of the fruit and its clustered growth pattern while also providing a cost-effective and efficient solution for greenhouse environments.

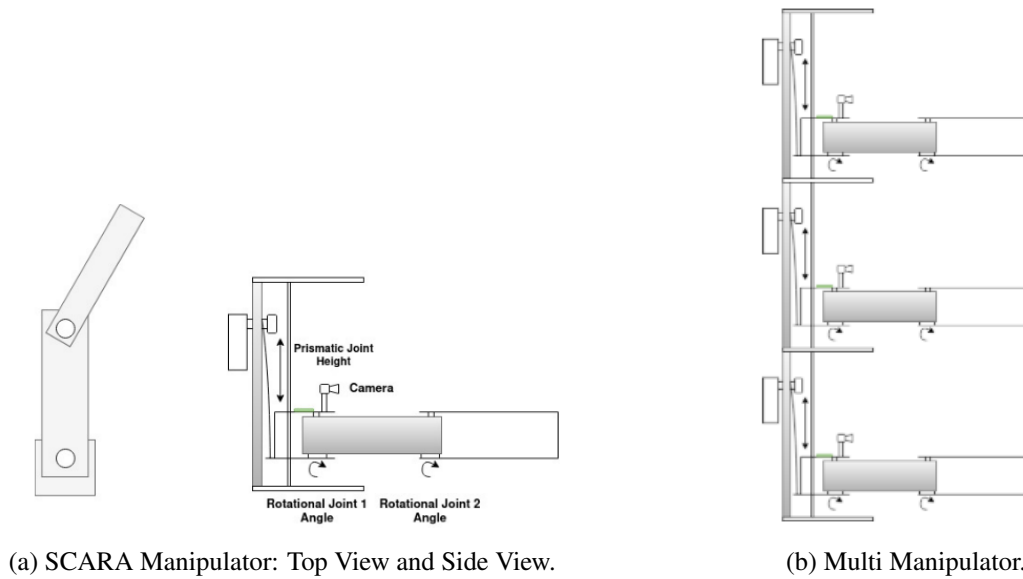


Figure 3.2: SCARA Manipulator for Tomato Harvesting [19].

The proposed system consists of a triple stackable three DoF SCARA manipulator PRR (Prismatic-Revolute-Revolute) (Figure 3.2). The prismatic joint allows for vertical adjustment, while two rotary joints enable horizontal and rotational movements. Each manipulator is independently controlled by a custom electronic circuit integrated with brushed DC motors equipped with helical gear mechanisms for precise control and stability. The motors' integrated magnetic encoders provide real-time feedback, ensuring accurate joint positioning. Control is achieved through cascade Proportional-Integral-Derivative (PID) controllers, with position and speed loops ensuring smooth and precise movements.

The system is powered by a Raspberry Pi 4B for high-level control and a Tensor Processing Unit (TPU) for efficient tomato detection. Detection is based on two main methods: color filtering and neural network classification. The images, captured by a monocular camera, are converted into the HSV color space, where thresholds are applied to the Hue component to isolate the red tones typical of ripe tomatoes. From these regions, centroids are identified, and areas of interest (ROIs) are extracted. These are analyzed by a classifier based on the MobileNetV2 architecture and trained to distinguish tomatoes from other objects.

This process enables fast and accurate detection, with the tomatoes being located using depth estimation based on stereo vision. The system demonstrated high detection accuracy in tests conducted in a laboratory environment with artificial tomatoes, registering only one false positive and no false negatives in 55 observations.

Researchers tested the system in a controlled laboratory environment using artificial tomatoes. Experimental results showed effective tomato detection, with the classifier achieving a high accuracy rate, detecting one false positive and zero false negatives in 55 observations. The manipulator demonstrated reliable positioning, with errors below 3.5 cm in most configurations. However, larger errors were observed in the first test configuration, with deviations exceeding 30 cm.

For future work, the authors plan to focus on optimizing control systems to reduce positioning errors and adapting the manipulator to handle the variability of real-world agricultural environments.

3.1.3 Robots for strawberry harvesting

The proposed work in [20] presented a multi-arm Cartesian manipulator system designed for strawberry harvesting (Figure 3.3). The author's contributions are threefold: a novel active obstacle-separation path-planning algorithm for cluster picking; improvements to the vision system, the gripper, the arm, and the control system; and a fully integrated harvesting system that enables the robot to pick strawberries continuously in polytunnels.

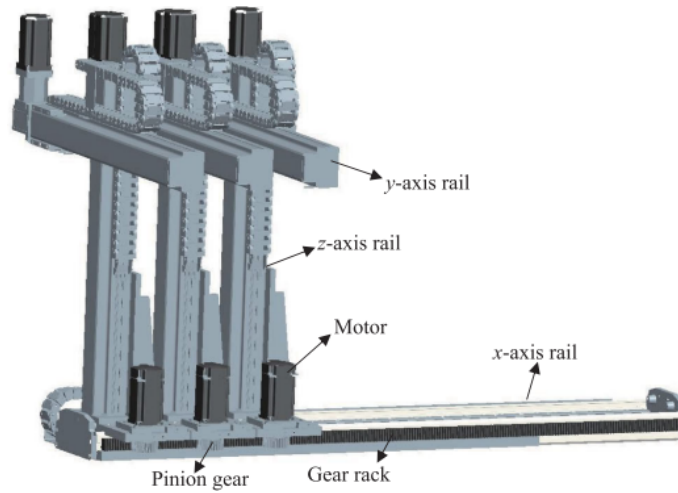


Figure 3.3: Strawberry harvesting manipulator [20].

The system consists of two or more independent Cartesian arms with three DOF mounted on a common rail. This single rail design allows the arms to operate simultaneously, sharing the X axis while maintaining independent movement along the Y and Z axis. This configuration offers several advantages, such as eliminating collisions within the workspace and enabling an efficient division of the picking area between the arms.

Each arm is equipped with a three-finger gripper as its end effector. The system employs RGB-D depth cameras to generate point clouds of its environment, which are used to identify

strawberries. For the detection of strawberries, color-based identification is applied, specifically using the HSV values of ripe strawberries, which are adjusted according to variations in ambient light intensity sensed by the gripper's internal Infrared sensors. The arms are powered by stepper servomotors, which provide precise control at a lower cost compared to industrial robotic arms. These motors are equipped with integrated encoders and controllers to ensure accurate positioning and smooth operation.

The use of multiple arms in the harvesting process is a key innovation that enhances the robot's operational efficiency. The robot divides the workspace into zones, with each arm assigned to a specific section. This zoning ensures systematic coverage of the picking area and minimizes redundant movements.

Field experiments were conducted in a tabletop strawberry greenhouse. For a single isolated ripe strawberry, the robot achieved a 97% success rate for picking the target in only one attempt and 100% success in two attempts. However, the success rate dropped as the number of obstacles increased, like when unripe ones surrounded the target ripe strawberries, or there were multiple targets tightly grouped. Unripe strawberries were the major cause of picking failures, with success rates of 50% to 75% (for 1 and 2 attempts respectively) for targets only partially surrounded, and as low as 5% to 20% for completely surrounded strawberries. The main reasons for unsuccessful harvests included failures or errors in locating the ripe strawberries, strawberries that exceeded the gripper's size limit or capacity, and target fruits that were out of reach. The harvesting speed in single-arm mode was 6.1 s, including harvesting and arm travel time to the next target, while for double-arm mode, the average harvesting time was recorded as 4.6s per strawberry.

In conclusion, the dual-arm manipulator system represents a significant advancement in agricultural robotics. However, addressing the challenges identified during field tests will be crucial to improving the system's performance in complex harvesting environments.

3.1.4 Robots for kiwi harvesting

The authors of [21] developed a robotic system for kiwifruit harvesting, centered on a custom-designed end-effector that integrates three key functions: fruit gripping, detachment, and unloading (Figure 3.4). The system was specifically engineered for orchard environments using scaffolding-based canopy structures, typical of Hayward kiwifruit cultivation.

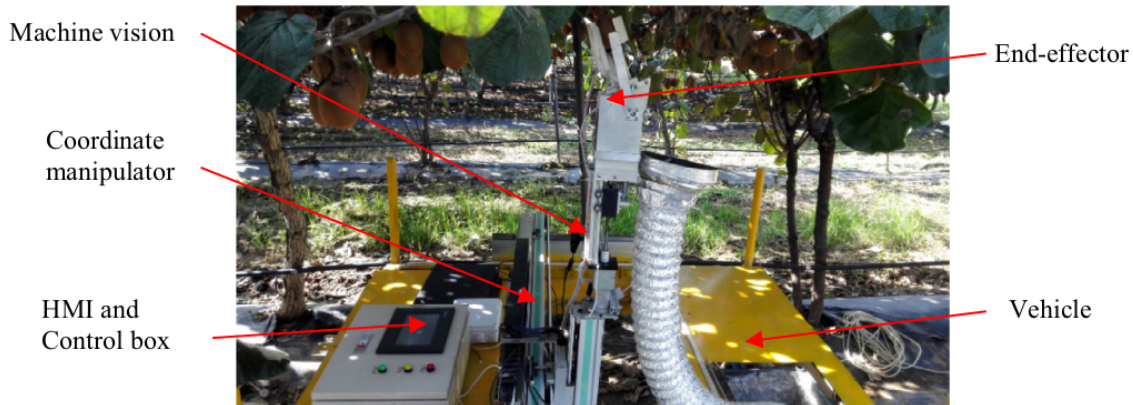


Figure 3.4: Kiwi harvesting robot [21].

The end-effector features two bionic fingers designed based on the physical properties of the kiwi, such as size, curvature, and separation force. These fingers operate through a guided motion trajectory driven by an inverse cam mechanism and a connecting rod linkage system. The mechanical design was simulated and optimized using ADAMS software to ensure smooth and coordinated finger movement for the grasp-pick-release sequence.

The harvesting process begins with the end-effector approaching the fruit from below. A fiber-optic sensor detects the presence of the fruit, after which the stepper motor actuates the rod that drives the fingers to close around the fruit. The fingers then apply a bending motion to detach the fruit by pulling it downward at an angle of approximately 60° , which minimizes the required separation force. Once detached, the fruit is automatically released into a collection basket with soft padding. This entire operation is executed through a compact and integrated mechanical system controlled by a microcontroller, pressure sensors, position sensors, and a stepper motor.

The robotic platform also includes a Cartesian coordinate manipulator and a machine vision system using a Kinect sensor. RGB and depth images are processed using a convolutional neural network (Faster R-CNN implemented in MATLAB) to identify and localize kiwifruit in 3D space. This data guides the manipulator to position the end-effector accurately. The system demonstrated acceptable localization tolerance, with a maximum observed error of 15.3 mm along the Z-axis.

Field tests conducted in a kiwifruit orchard showed a high picking success rate of 94.2%, with an average picking cycle time of 4–5s per fruit. Only 4.9% of the fruits suffered peel damage, which is lower than typical manual harvesting. The primary causes of failure were misalignment due to occlusion, slipping of small fruits, and tight clustering within the canopy.

For future work, the authors propose improving the cam-guided mechanism to reduce trajectory errors, enhancing adaptability to fruit variability, and integrating multi-arm collaboration to boost overall picking efficiency and commercial viability of the system.

3.2 End Effectors for Agricultural Robotic Harvesting

End effectors play a pivotal role in the performance and efficiency of agricultural robotic systems, particularly for harvesting applications. These components serve as the point of interaction between a robot and the crops, enabling tasks such as grasping, cutting, pulling, or rotating to detach the produce from the plant. The design and functionalities of end effectors must be aligned with the specific requirements of each type of harvest, considering factors such as size, shape, delicacy, and fruit detachment method [22].

Several types of end effectors can be categorized based on their release methods and design principles:

- **Contact-Grasping Grippers:** These are the most commonly used end effectors due to their versatility and adaptability. They establish direct contact with the fruit or its stem, ensuring a secure grasp. There are two primary subcategories:
 - **Fruit-holding grippers:** Designed to grasp the body of the fruit directly (Figure 3.5). These grippers typically feature two to five finger-like structures capable of applying controlled force to secure the fruit for detachment and handling. Their adaptability to various fruit shapes and sizes is a significant advantage, particularly when augmented with soft-touch materials or integrated pressure sensors to mitigate the risk of bruising. However, these grippers present challenges when handling delicate fruits, such as strawberries or tomatoes, as excessive force can easily cause bruising or other forms of damage. Furthermore, their operation may be impeded by interference from surrounding branches and foliage, which can obstruct the gripping mechanism or complicate access to the fruit [22].
 - **Stem-holding grippers:** This robotic gripper is designed to secure the stem of the fruit rather than grasping the fruit directly. This approach minimizes the risk of damaging delicate products, making these grippers particularly suitable for crops where surface integrity is crucial, such as tomatoes (Figure 3.6). The grippers are equipped with advanced mechanisms to detect, identify, and precisely hold the stem, ensuring a secure grip without compromising the quality of the product. However, detecting and accessing the stem in dense foliage can be a complex challenge and requires high precision, advanced sensory input, and robust algorithms to differentiate the stem from surrounding obstacles. These technological requirements make their implementation more demanding compared to conventional fruit-gripping methods [22].
- **Air-Suction:** These devices use vacuum pressure to grip and extract fruit or crops, making them particularly useful for lightweight and small-sized produce such as strawberries and tomatoes. The suction mechanism draws the fruit into the end-effector, where detachment occurs either through shearing or pulling, followed by transport into a stowage system (Figure 3.7). Air-suction systems are advantageous in scenarios where fruits are located deep

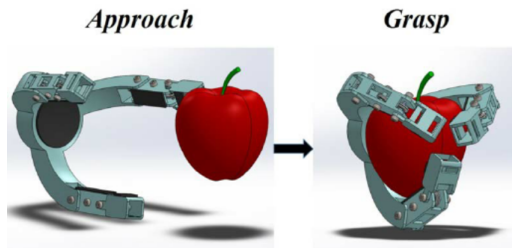


Figure 3.5: Fruit-holding grippers [23].



Figure 3.6: Stem-holding grippers [24].

within canopies or where mechanical gripping is difficult. For instance, vacuum-assisted harvesters have been successfully applied in tomato and apple picking, achieving high success rates while minimizing mechanical complexity. However, these systems require high power to generate sufficient suction, which can increase the robot's overall energy consumption. Additionally, their use must be carefully calibrated to avoid damaging soft or delicate produce, limiting their application primarily to firmer crops or those less prone to bruising. As a result, air-suction is often integrated with supplementary mechanisms, such as cutting blades or rotating joints, to improve harvesting reliability and minimize fruit damage [25].



Figure 3.7: Air Suction [25].

- **Suction-Holding Devices:** They are used as auxiliary tools for harvesting fruit, particularly when fruits are hidden behind dense foliage or branches. These devices create a vacuum seal on the fruit's surface, holding it securely for detachment without requiring full access to it (Figure 3.8). While they are effective in reaching fruits that are difficult to access, suction devices are generally less suitable for delicate or sensitive fruits due to the potential risk of skin damage. The vacuum pressure must be carefully controlled to prevent bruising or piercing the fruit, limiting their use primarily to firmer produce like apples or tomatoes. As a result, suction devices are often combined with other detachment methods, such as cutting or grasping, to ensure efficient and safe harvesting [22].

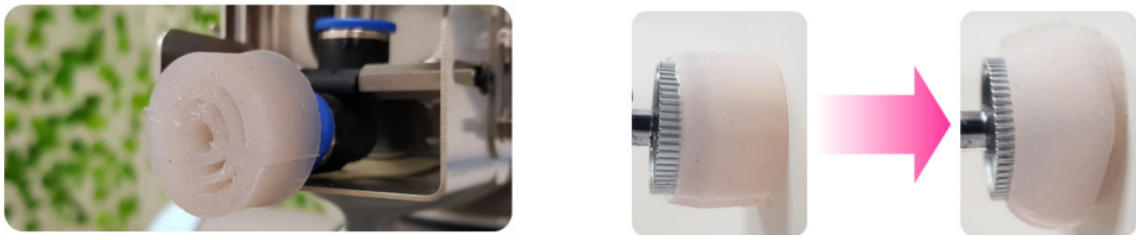


Figure 3.8: Suction Device [26].

- Rotation Mechanisms:** Inspired by traditional manual harvesting techniques, rotation mechanisms employ a twisting motion to detach fruits from their stems or peduncles, such as tomatoes. This method replicates the natural harvesting process by folding the stem at the fruit–stem interface, facilitating a clean detachment (Figure 3.9). The primary advantage of this approach is the reduced pulling force exerted on the fruit, which minimizes the risk of damage to both the produce and the plant. Rotation mechanisms are particularly effective for crops with tough peduncles, where simple pulling may not suffice for detachment. However, precise control of the rotational motion is essential to prevent excessive force, which could potentially harm the fruit or stem. Proper calibration is required to accommodate the variability in stem strength among different crops [22].

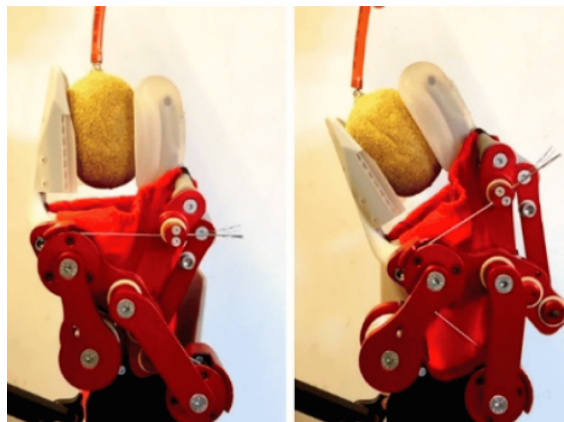


Figure 3.9: Rotation Mechanisms [26].

- Scissors/Saw-like Tools:** These devices, equipped with cutting mechanisms such as scissors or saw-like tools, are designed for crops with thick or woody stems where detachment by pulling or twisting is impractical, like sweet peepers. These tools typically work in conjunction with gripping components to stabilize the fruit during the cutting process (Figure 3.10). This approach ensures a precise severing of the stem, preserving the integrity of both the plant and the harvested produce. However, the effectiveness of these devices is highly dependent on accurate stem detection and precise blade alignment. Misalignment or incorrect positioning may result in incomplete cuts or unintended harm to the plant, highlighting the importance of precision engineering in their design and operation [22].

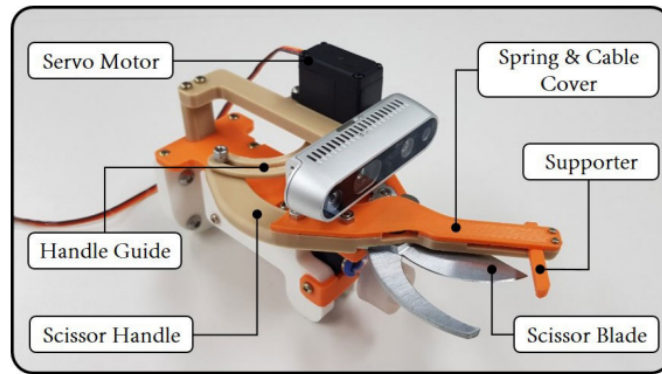


Figure 3.10: Scissors/Saw-like Tools [26].

3.3 Visual perception

Fruit perception refers to the process of using image processing and sensor-based techniques to determine the status and location of fruit trees. However, real-world environments pose significant challenges, including non-uniform lighting, unstructured fields, occlusions, and other unpredictable factors, which complicate the accurate identification and localization of fruit trees.[26].

3.3.1 Visual Sensing Technology

Harvesting robots utilize visual detection technology to identify fruits and vegetables. A robotic servo controller gathers 3D data about the fruit's surrounding environment, including its geometry and spatial coordinates. The visual camera and its control mechanism act as the hardware basis for the visual sensing technology, acting as a communication interface between the robot and its external environment. Images captured by the camera are typically classified into digital images, laser images, and multi-spectral images [27].

3.3.1.1 Monocular camera system

Monocular camera systems provide a cost-effective and reliable alternative for posture estimation in robotic manipulators, particularly in agricultural applications. These systems utilize a single camera to track visual markers, converting their pixel coordinates into real-world coordinates through planar homography. In agricultural contexts, such systems offer flexibility by requiring fewer sensors and maintaining accuracy suitable for tasks such as fruit detection and manipulator control [28] [29].

Planar homography plays a pivotal role in translating 2D image data into 3D world coordinates, as the markers are coplanar with the manipulator joints. This approach eliminates the dependency on incremental rotary encoders, mitigating issues such as backlash, joint friction, and elastic deformation. Furthermore, this setup simplifies calibration and facilitates integration into agricultural environments, where manipulator precision is less stringent than in industrial settings [28].

The advantages of monocular camera schemes include reduced hardware complexity, lower costs, and versatility. However, the system faces challenges, including environmental lighting variations affecting marker detection and calibration errors in the homography matrix. Future developments aim to address these limitations by incorporating LED-based markers synchronized with the camera shutter and implementing enhanced self-calibration techniques. These improvements will further optimize the system for precise operations such as pruning and harvesting. [28].

3.3.1.2 Stereo-Vision Systems

Stereo-vision systems are widely used in agricultural robotics to reconstruct three-dimensional (3D) coordinates of targets, particularly for fruit detection and localization. These systems operate by utilizing two or more cameras separated by a fixed distance to capture images of the same object from slightly different perspectives. The primary techniques include binocular vision systems, which rely on optical geometry and traditional optimization algorithms, and RGB-D cameras, which use ToF methods to gather depth information through infrared sensors [29].

Advantages of stereo-vision systems include their ability to operate effectively in both indoor and outdoor environments without relying on artificial light sources. They are highly suitable for 3D reconstruction tasks and can provide accurate spatial positioning data. However, these systems have challenges, including susceptibility to dynamic environmental factors such as inconsistent lighting, occlusions, and movement. Additionally, the computational demands for stereo matching and the inefficiencies of operating in unstructured environments present further limitations [27].

3.3.1.3 Laser Active Vision Technology

Laser active vision technology uses infrared laser sensors to provide precise 3D spatial data about the environment. This approach is particularly effective in agricultural applications, such as detecting spherical fruits like citrus. The technology combines laser-generated depth measurements with image analysis algorithms to identify objects based on size, shape, and surface reflectivity.

In controlled environments, this technology offers strengths such as high accuracy in-depth measurement and the ability to differentiate between closely spaced objects. For example, early implementations achieved detection rates of up to 80–90% for certain fruits. However, laser-based systems also face significant limitations. They are prone to occlusions caused by environmental obstacles, and their accuracy can diminish with increasing distance. Furthermore, integrating laser systems into robotic platforms increases structural complexity, making them less suitable for field conditions with variable terrain [27].

3.3.1.4 Multi-Spectral Imaging Technology

Multi-spectral imaging technology expands the capabilities of agricultural robots by capturing images across multiple spectral bands. This enables the detection of features that are invisible to

the naked eye or standard RGB cameras. Multi-spectral systems are particularly useful for tasks such as monitoring crop health, identifying pest infestations, and assessing fruit maturity.

In agricultural robotics, multi-spectral imaging has been applied to segment images of fruits, classify their features, and evaluate their growth stages. The technology enables precise yield estimation, creating detailed vigor maps, and assisting in path planning for fruit-picking robots. Its integration with drones further enhances its utility, allowing for efficient terrain analysis, damage assessment, and vegetation monitoring. Compared to satellite imaging, drone-based multi-spectral systems offer higher resolution and more accurate on-site evaluations. These features make multi-spectral imaging a powerful tool for precision agriculture, improving efficiency and decision-making processes. However, like other optical imaging methods, multi-spectral systems are affected by uneven lighting conditions, which can impact sampling quality. Despite this, multi-spectral imaging remains a valuable tool for enhancing the accuracy and functionality of vision-based harvesting robots [27].

3.3.2 Object Recognition Methods

3.3.2.1 Single Feature Vision Method

Single-feature vision methods rely on a single characteristic of an object, such as color or shape, to recognize and locate it, using specific attributes for operation. For example, color thresholding techniques can distinguish fruits such as red tomatoes or yellow citrus fruits from the environment. Similarly, shape detection algorithms identify geometric patterns, such as circularity, to locate targets. Since these methods follow a simpler approach, they are susceptible to environmental variations, such as changes in lighting or occlusion by leaves and branches. To overcome these limitations, improvements such as advanced segmentation algorithms and robust feature detection models, such as those described in the following sections, were implemented, which increases accuracy in more challenging conditions [27].

3.3.2.2 Multi-Feature Fusion Method

Multi-feature fusion methods combine several characteristics, such as color, texture, and shape, to recognize the object more robustly. This approach is particularly effective in complex scenarios, such as fruits overlapping or hidden by leaves or branches. By integrating multiple features, these methods significantly improve recognition rates and reduce sensitivity to environmental noise. For example, the combination of texture and color characteristics has proven effective in distinguishing between fruits and foliage, even under varying lighting conditions. However, the computational complexity of these techniques can pose challenges, especially when processing large-scale datasets in real-time. [27]

3.3.2.3 Deep Learning Methods

Deep learning methods have revolutionized object recognition in agricultural robotics, particularly for fruit detection in automatic harvesting systems. Using convolutional neural networks (CNNs) and advanced architectures such as Faster R-CNN, Mask R-CNN, and YOLO, these approaches enable robust feature extraction from raw images, effectively addressing challenges like varying lighting conditions, occlusions, object overlap, and morphological diversity among fruit species [30].

- **Faster R-CNN:** is a model for object detection that integrates region proposal generation and object classification into a single network for efficiency and accuracy. This architecture combines two key components: the Region Proposal Network (RPN) for detecting regions of interest (ROIs) and the classification with bounding box regression to classify and refine those regions. This unified deep neural network simplifies both region proposal and object classification tasks. Designed for high performance, Faster R-CNN achieves superior detection results by sharing convolutional layers between these components, enabling end-to-end training. In agricultural applications, such as fruit detection in orchards, it excels in handling high object densities, varying scales, and challenging outdoor conditions, achieving high F1 scores across multiple fruit types. Techniques like data augmentation and transfer learning further enhance its performance, reducing the need for extensive labeled datasets [31].
- **Mask R-CNN:** is a model that combines object detection with instance segmentation, making it very effective in scenarios that require precise differentiation of objects overlapping or hidden by foliage or branches. The architecture is based on Faster R-CNN by adding a segmentation branch, which allows the simultaneous generation of bounding boxes and pixel-level masks for detected objects. The model comprises two key components: a backbone network, typically a convolutional neural network (e.g., ResNet), for feature extraction and region proposal generation, and a mask head for pixel-level segmentation. The backbone uses a RPN to identify candidate object regions, which are refined and classified. In parallel, the mask head predicts binary masks for each region, enabling precise segmentation [18].
- **YOLO:** is an object detection model that balances accuracy and speed, making it ideal for real-time applications. Unlike multi-stage methods such as R-CNN, YOLO performs detection in a single pass through the network, significantly reducing inference time. Its architecture consists of three main components: the backbone, which extracts features from the image at multiple scales; the neck, which combines these features using structures like PANet or FPN to fuse spatial and contextual information from different resolutions; and the head, responsible for generating the final predictions, such as bounding boxes and class probabilities. Additionally, this architecture can be broken down into components such as the stem, which processes raw input data, downsampling layers to reduce spatial resolution

and improve computational efficiency, and stages with convolutional blocks that extract features at different levels. This structure makes YOLO particularly effective in scenarios that require fast and efficient responses [32].

These deep learning architectures have been widely applied in agricultural contexts, demonstrating their effectiveness in real-world scenarios. This is illustrated by the work of [33], who proposed an automated system for detecting strawberries at different stages of ripeness using the YOLOv7 architecture with the aim of increasing harvesting efficiency. The system was trained using images captured in a greenhouse, covering six growth stages: green, white, early, turning, late, and red. The images feature different angles and lighting conditions, which simulate real agricultural scenarios. The YOLO-v7 architecture was chosen due to its balance between accuracy and speed, enabling real-time detection. The main modification to the model consisted of increasing the resolution of the images to 960×960 to improve the detection of smaller strawberries. During testing, the system achieved a precision of 79.2%, a recall of 57.5%, and a mean average precision (mAP) of 55.8%, emphasizing the turning phase, which obtained a mAP of 100%, and the red phase, with 80.4%. The results indicate that the YOLO-v7 model demonstrated satisfactory performance even in challenging scenarios with imbalanced data, being effective for real-time applications and with potential for integration into robotic harvesting systems.

3.3.2.4 3D Reconstruction Method for Vision-Based Target

The 3D reconstruction method (i.e. point clouds) is another important technology for harvesting robots, providing spatial information and structural details necessary to locate and recognize objects more accurately. This process involves capturing images from multiple angles, camera calibration, and feature extraction to create a detailed 3D model of the harvesting environment. The reconstructed 3D data allows the robot to calculate the exact position and orientation of the fruit, guiding its arm for efficient harvesting. Furthermore, 3D mapping of the environment helps with navigation and obstacle avoidance. However, this method presents a high computational demand and susceptibility to environmental factors, including lighting and occlusions.

3.4 Review of the Research

Regarding manipulators and end-effectors, none of the systems currently developed are directly applicable to harvesting *Arbutus* berries due to their unique morphological characteristics. The dense, rounded, and often asymmetric canopy and the simultaneous presence of flowers and fruits in different ripeness stages require a highly selective and cautious approach. This scenario differs significantly from crops such as apples, tomatoes, strawberries, or kiwis, for which current robotic systems have been designed and tested.

For example, the apple harvesting system [18] includes a 3-DoF manipulator with a suction-based end-effector. Although effective in well-organized orchard environments, its rigid structure and limited reach would hinder access to *Arbutus* berries, particularly those inside the canopy or

on thinner, more flexible branches. Moreover, suction may be inadequate for *Arbutus* berries with a rough surface and small size, increasing the likelihood of grasping failures or fruit damage.

Similarly, the SCARA manipulator developed for tomatoes [17], while compact and equipped with robust control, is not easily adaptable to irregular terrain or the bushy and non-linear arrangement of *Arbutus* branches, typical of natural plantations. Additionally, its operation is optimized for controlled environments such as greenhouses.

Strawberry harvesting systems [20] using multiple Cartesian arms and dedicated grippers have achieved high success rates in polytunnels. However, their efficiency drops significantly in the presence of obstacles, such as immature fruits or densely clustered targets—conditions even more pronounced in *Arbutus* plants, where flowers and fruits coexist, requiring millimetric precision to avoid damage. Furthermore, the complexity of multi-arm systems raises concerns regarding scalability, cost, and maintenance, especially in forested or semi-wild environments.

The most promising solution from an operational efficiency perspective is the kiwi harvesting robot [21], which uses a transport tube to direct the harvested fruit straight to the basket, eliminating the need for the manipulator to carry the fruit. This method increases harvesting speed and reduces the risk of collisions with branches or flowers. Such an approach is particularly relevant for *Arbutus* harvesting, where each extra movement by the manipulator increases the risk of contact with delicate structures and potentially harms future production. However, this system also has disadvantages, including mechanical complexity and reliance on precise fruit detection for alignment with the tube, which may be impractical in very dense canopies or when fruits are partially hidden.

The studies reviewed present varied approaches to fruit detection in automated harvesting, ranging from methods based on simple features (such as color and shape) to advanced deep learning architectures like Mask R-CNN, Faster R-CNN, and YOLO. These solutions reflect different trade-offs between performance, robustness, and processing time.

Among the analyzed models, those based on the YOLO architecture stand out due to their high speed and accuracy, key factors for real-time applications like automated harvesting. YOLO's lightweight and optimized structure allows fast inference even on devices with limited computational resources, which is particularly advantageous in dynamic and outdoor agricultural environments. Models such as YOLOv7 have shown strong performance in real-world scenarios, with sufficient precision to distinguish ripeness stages, as demonstrated in recent work on strawberry detection [33].

In summary, despite significant advancements in agricultural robotics, existing manipulators still lack the flexibility, morphological adaptability, and precision required to address the specific challenges posed by *Arbutus* berry harvesting. These limitations emphasize the need for dedicated solutions that combine lightweight structural design, mobility in irregular environments, and gentle, selective manipulation strategies to minimize the risk of damaging immature fruits or flowers. The inspiration from the kiwi system's transport tube is promising, provided it is appropriately adapted to the *Arbutus* bush morphology.

In perception, there is a clear need for robust and real-time detection, which is essential in automated outdoor harvesting where variables such as uneven lighting, occlusion, and fruit overlap are frequent. In this context, the YOLO architecture is particularly suitable. It offers an excellent balance between accuracy and speed, capable of running in real time even on low-power embedded systems, making it ideal for integration into mobile robotic platforms. Therefore, YOLO will be adopted in this work as the core of the fruit detection system for Arbutus berries.

To ensure effective automated harvesting in outdoor environments, this project requires a robust and real-time detection system capable of handling common challenges such as uneven lighting, occlusions, and overlapping fruits. For this purpose, we adopt the YOLO architecture, which provides an optimal balance between accuracy and speed. Its ability to perform real-time inference on low-power embedded systems makes it ideal for integration into mobile robotic platforms. With these advantages, YOLO will serve as the core detection system for identifying Arbutus berries in our proposed solution.

Chapter 4

Proposed System Architecture

This chapter presents the proposed system solution, detailing the modular structure that integrates perception, planning, and manipulator actuation. The main functional requirements necessary to ensure the system's efficiency in the automatic and selective harvesting of *Arbutus* berries are defined, considering the plant's morphological characteristics and the demands of the agricultural environment. This chapter delves into the interconnections between the different system modules, with each block being further developed in detail in the following chapters.

4.1 System Architecture

The architecture of the proposed system is a simulation framework that includes the *Arbutus Berry Harvesting* (ArBeH) module. This framework enables efficient interaction between perception, planning, and actuation, addressing the specific challenges of *Arbutus* cultivation, such as dense canopies, variable fruit ripeness, and the need for selective harvesting. This architecture, illustrated in Figure 4.1, is composed of three main blocks: the simulated environment (Webots), the manipulator control and communication interface between the simulator and ROS2, and the perception and planning system (Harvest Vision Planner, HVP). Figure 4.2 presents the simulation of the ArBeH manipulator in the Webots virtual environment, highlighting the integration between manipulator control and the perception system, demonstrating that without perception, the manipulator would not be able to reach the berry. The figure also displays the images provided by the manipulator's two onboard cameras: the top-left image shows the perspective from the camera mounted at the base of the manipulator, while the bottom-right image presents the view from the camera located at the tip of the manipulator, that offering visual perspectives crucial for fruit detection and localization.

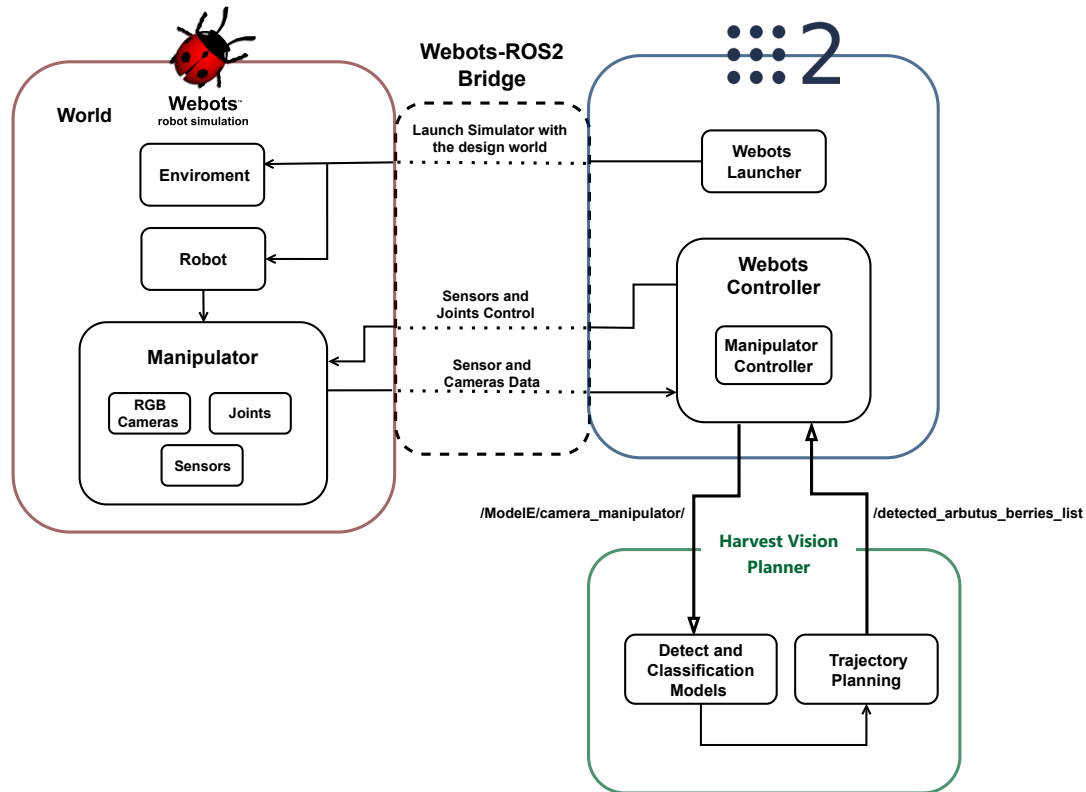


Figure 4.1: ArBeH system architecture composed of three main blocks: the simulated environment in Webots, the Webots-ROS2 communication and control bridge, and the Harvest Vision Planner, which integrates perception, detection, and trajectory planning for automated Arbutus berry harvesting.



Figure 4.2: Simulation of the ArBeH manipulator in the Webots virtual environment and the displays of the views from the two onboard cameras.

The first block corresponds to the virtual environment created in the Webots simulator, where an Arbutus plantation is modeled. In this environment, the robotic manipulator is equipped with RGB cameras, distance sensors, and actuators (joints), enabling the collection of essential sensory and visual data for perception and control. The simulator allows for testing different manipulator configurations, assessing their reach, and validating navigation and harvesting strategies in complex scenarios.

The second block is responsible for the control logic and integration of the manipulator and the development and execution of the Arbutus berry harvesting algorithms (see Chapter 6). This block's core is the Webots Controller Node, which manages the bidirectional communication interface between the simulator and ROS2. This node hosts the actuator control plugins, ensuring real-time feedback and closed-loop control. The manipulator controller operates as a dedicated plugin within this node, sending joint positions required for selective berry harvesting and translating them into precise movements of the manipulator's joints. The Webots Controller Node receives sensor and camera data from the simulated environment, publishes it to ROS2 topics, and processes control commands sent from ROS2 back to the simulator. This architecture enables seamless, modular integration between the virtual environment and the robotics middleware, ensuring that all relevant information from the virtual world is available in real time to the perception and control algorithms.

The third block encompasses the entire perception and planning component, serving as the intelligent core of the system. The HVP includes advanced detection and classification models (see Chapter 5), based on deep learning, capable of identifying and classifying Arbutus berries at different ripeness stages. It also features a trajectory planning algorithm that receives the list of detected fruits and calculates the optimal sequence of movements for the manipulator, ensuring efficient reach to all harvestable fruits while minimizing redundant movements. The detection results are published to a ROS2 topic (/detected_arbutus_berries_list) and are consumed by the manipulator controller to execute harvesting actions.

4.2 Challenges from the Robotics Point of View

Automating fruit harvesting presents significant challenges due to the demands of navigating dense foliage, identifying ripe fruit, avoiding damage, and adapting to external conditions. This section discusses the main aspects that contribute to these complexities:

- **Navigating Dense Canopies:** Orchard environments are often characterized by tightly packed trees with overlapping branches and dense foliage. This configuration not only restricts the maneuverability of robotic systems but also makes it challenging to locate fruits, complicating path planning and precise movements. To address these obstacles, fruit-harvesting robotic arms must overcome the limited visibility caused by the foliage. Advanced vision systems, such as LiDAR and depth cameras, are essential for generating detailed 3D models of the environment. These technologies enable accurate navigation through complex canopies while minimizing physical disruptions. [34]

- **Detecting and Identifying Ripe Fruits:** A significant challenge in automated harvesting is the accurate detection and differentiation between ripe and unripe fruits. This demands sophisticated machine vision systems capable of analyzing diverse fruit shapes, sizes, and colors, often in the presence of inconsistent lighting and occlusions from leaves and branches. For fruits like arbutus berries, which ripen unevenly even on the same branch, this complexity is heightened. Careful selection is essential to ensure unripe or overripe fruits are not harvested, requiring the integration of advanced multispectral imaging and adaptive artificial intelligence systems. [34]
- **Avoiding Damage to Fruits and Branches:** The design of end-effectors, the mechanical tools used for fruit detachment, is crucial to prevent damage to the fruits and the plants. Various mechanisms, including suction-based, cutting, and gripper-based designs, have been developed, each with its challenges. Suction systems, for instance, may bruise fragile fruits, while cutting tools can leave behind residual branches. Achieving the right balance of firmness and precision is essential for robotic harvesters to handle fruits gently without bruising and to protect the plant from harm. This requires end-effectors to incorporate flexible materials and sensors capable of measuring and adjusting force during harvesting. [34]
- **Operating in Outdoor Environments:** Agricultural fields pose dynamic environmental challenges, such as varying light, wind, rain, and humidity, which can greatly affect the performance of robotic systems. Unlike controlled settings, these ever-changing conditions demand robust, weather-resistant hardware and intelligent software that can adapt in real-time to ensure consistent performance.

Addressing these challenges requires a multidisciplinary approach, combining advancements in robotics, computer vision, sensor technologies, and artificial intelligence to achieve reliable and efficient automated harvesting systems.

4.3 Requirements for the Perception and Harvesting System

The design and implementation of a perception and harvesting system for agricultural environments require addressing several critical operational requirements. These requirements are informed by the unique challenges of agricultural operations, including complex natural environments, diverse crop characteristics, and varying environmental conditions. By adhering to these requirements, the system can ensure functionality, reliability, and efficiency in real-world scenarios.

Accurate fruit detection and localization are essential for harvesting in natural environments, where foliage, branches, or other structural elements of the plant frequently occlude fruits. The perception system integrates RGB cameras to accurately detect and localize fruits, even in dense and cluttered scenes. It provides precise spatial coordinates (2D/3D) required for planning and executing the manipulator's trajectory, ensuring efficient and collision-free operation.

Identifying different ripeness stages is another key requirement, as only fruits at the optimal ripeness—typically red—should be harvested. To this end, the system employs deep learning models trained on extensive annotated datasets, covering various ripeness stages and viewing angles. These models allow for robust classification of green, yellow/orange, and red fruits, ensuring selective harvesting based on maturity. In terms of performance, the system achieves a precision rate above 85%, meeting the accuracy demands of practical deployment in agricultural environments. This high detection reliability minimizes missed or incorrect picks, improving harvesting efficiency and reducing crop waste.

To ensure full canopy coverage, the manipulator was designed with a maximum reach of approximately 2 meters, enabling access to fruits across the entire tree structure. This capability ensures that no ripe fruit is left unharvested due to positional constraints.

The system can also process at least 10 frames per second, supporting real-time perception and decision-making necessary for responsive autonomous operation. This frame rate is critical for maintaining a continuous workflow during harvesting tasks.

Finally, the system is designed to be as efficient as a human operator in terms of speed and care in handling fruits. This includes minimizing physical damage during collection through the use of gentle end-effectors, such as soft grippers or suction mechanisms, thereby preserving fruit quality and tree health.

By fulfilling these operational criteria—ranging from accurate perception and classification to mechanical reach and real-time processing—the proposed system offers a viable solution for autonomous arbutus harvesting, reducing labor demands while promoting productivity and sustainability.

4.4 Conclusion

The proposed architecture, as illustrated in the diagram (Figure 4.1), represents an innovative and robust solution for automated Arbutus berry harvesting, integrating computer vision, robotics, and simulation technologies. Meeting the defined requirements ensures the effectiveness of selective harvesting and the sustainability and adaptability of the system to different agricultural challenges, positioning it as a reference for future applications in unconventional crops.

Chapter 5

Perception System

This chapter presents the perception system, including the methodologies used and the development of deep learning models for detecting and classifying Arbutus berries. It discusses the challenges associated with detecting small, densely clustered fruits at various ripeness stages and analyzes the results obtained with different approaches and model architectures. It is important to note that the content of this chapter is based on a paper submitted and accepted for presentation at the ETFA conference, thereby validating its scientific relevance and contribution to advancing the state of the art in agricultural robotics.

5.1 Materials and Methods

5.1.1 Data Collection

The first step in creating an Arbutus Berry dataset was the acquisition of data to meet the research objectives. The data was collected in an Arbutus tree plantation with two sensors: the OAK-D S2 sensor [12] and the Livox Mid-360 LiDAR sensor [10]. Both sensors were integrated into a structure that could easily be carried by a single person (Figure 5.1). The data collection process involved two main setups. Some data was collected as the person moved through an inter-row of the plantation, passing by several Arbutus trees. This allowed for a broader range of data, capturing the information from various tree canopies along the row. In addition, more data were gathered around a single Arbutus tree canopy. In this setup, the sensors were moved around the tree inside and outside the canopy, ensuring that the sensors captured detailed information from different angles and distances, allowing for a more focused analysis of the tree's canopy and Arbutus berries.



Figure 5.1: Setups of sensors used in data collection, including the OAK-D S2 RGB-D sensor and the Livox Mid-360 LiDAR sensor.

5.1.2 Data Labelling and Augmentation

After acquiring images, the collected images were manually annotated using the CVAT [35]. Each annotation included a bounding box to delimit the fruit, providing information on its area, position, and ripeness classification. The annotation process considered the ripeness stage of each *Arbutus* berry, which is essential for accurate classification. There are three stages of ripeness in *Arbutus* berry: green, representing unripe fruit that is not yet ready to be harvested; yellow and orange, corresponding to fruit that is suitable for fresh consumption; and red, indicating that the fruit is fully ripe and ready for harvest, although it presents low firmness and requires careful handling and storage [36]. The images were exported from CVAT in the Ultralytics YOLO Detection format, where each image file has an associated annotations text file. Figure 5.2 illustrates an image from the dataset with manual annotations represented by red, yellow, or green bounding boxes depending on the fruit class.



Figure 5.2: Example of manual annotations using CVAT for the Arbutus Berry Ripeness Level Detection Dataset [4]

Two datasets were annotated: the Arbutus Berry Detection Dataset [3], containing only the detection of Arbutus berries, and the Arbutus Berry Ripeness Level Detection Dataset [4], which includes both detection and ripeness classification. Additionally, a third dataset, the Arbutus Berry Ripeness Level Classification Dataset [4], was derived from the latter, in which the berry bounding boxes were cropped and grouped by ripeness class for training the classifier. These datasets are openly available in the digital repository Zenodo. Both annotated datasets contain the same images and were processed identically, undergoing the same splits and data augmentation techniques.

The distribution of annotations between classes and the number of Arbutus berries annotated in the dataset are presented in Table 5.1. The images include Arbutus berries at different distances, angles, and ripening stages. The disparity in fruit counts between classes is because the dataset was collected when the Arbutus berries were already at a more advanced maturation stage. On average, each image contains 43 Arbutus berries.

Table 5.1: Number of annotations for each class in the original dataset.

	Arbutus Berry Green	Arbutus Berry Red	Arbutus Berry Yellow	Total of Arbutus Berry
Annotations	524	2644	859	4249

Dividing the dataset into training, validation, and testing is essential to ensure the robustness

and generalizability of DL models [37]. The training set is used to train the model, while the validation set is used to compare different models and hyperparameters, helping to prevent overfitting and to validate whether the model is effectively converging during training. Finally, the test set is entirely independent and used only to prove the effectiveness of the model's training, being composed of data never seen during training or validation, thus ensuring a fair and realistic evaluation of the model's performance. For this division, the following proportion was used: 70% for the training set, 20% for validation, and 10% for testing. Figure 5.3 shows the flow chart representing the methodological approach adopted in this work.

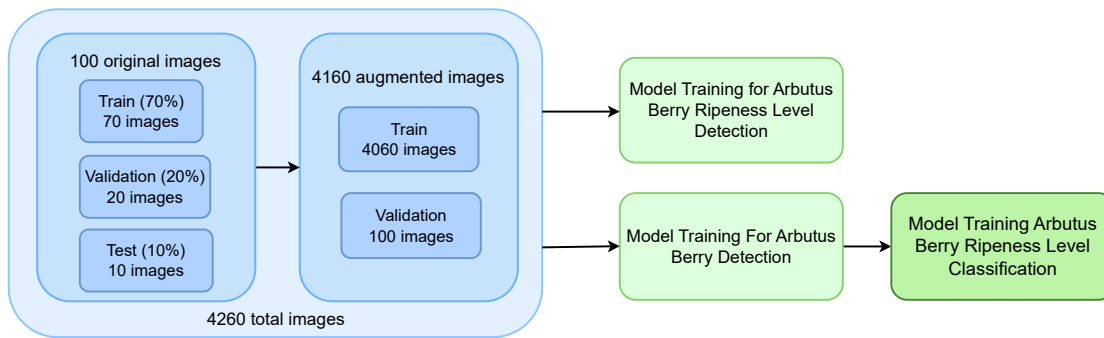


Figure 5.3: Workflow for detection and classification of Arbutus berries.

Data augmentation operations were performed using the Albumentations [38], a Python library designed for image enhancement. Only transformations that generated visually realistic results in the images to which they were applied were selected. Table 5.2 details the operations used in the training and validation sets. These transformations were divided into two main groups: the first three operations (Flip, Scale, and Rotation), which involve geometric modifications by moving or removing pixels without changing their values; and the remaining 8 operations (Optical Distortion, Hue and Saturation, Contrast, Emboss, Sharpen, Blur, ISONoise, and Illumination) that directly modify pixel values, affecting properties such as color, sharpness, and lighting. For the training set, it was decided to apply both the individual transformations and the combinations between the two groups. This approach resulted in 58 distinct variations, generating a total of 4060 augmented images. For the validation set, only the first three operations were used, which resulted in 100 augmented images. In its augmented version, the final datasets were composed by grouping these 4160 generated images with the 100 original images, totaling a comprehensive set of 4260 images, as illustrated in Figure 5.3. This stage is fundamental when training DL models, as it improves data quality, reduces overfitting, increases performance, and contributes to the models' ability to generalize.

Table 5.2: Augmentation operations applied to the original images.

Operation	Value	Description
Flip	-	Flip the image horizontally
Scale	0.7x	Scale the image by 0.7
	1.3x	Scale the image by 1.3
Rotation	+15°	Rotate image +15°
	-15°	Rotate image -15°
Optical Distortion	Random	Apply optical distortion
Hue and Saturation	Random	Change image's hue and saturation levels
Contrast	Random	Change image's contrast level
Emboss	Random	Apply embossing effect
Sharpen	Random	Sharpen the image
Blur	Random	Apply Gaussian blur
Noise	Random	Apply ISO noise
Illumination	Random	Simulate lighting effects

5.1.3 Data Generation

As the Arbutus berries are small and often occur in high density with different ripening stages within the same image, a specific approach was adopted to maximize the ability of deep learning models to detect small fruits. Inspired by the methodology proposed by the authors in [39], the original images were divided into smaller images of 300×300 pixels, with a 20% overlap between consecutive crops. This strategy allows smaller fruits to become more prominent within the model's field of view, reducing the likelihood of them being overlooked during detection, decreasing the number of FN.

Each original image was divided into 30 sub-images, which significantly increased the number of images in each of the training, validation, and test sets, contributing to the robustness of the model. The separation of data into training, validation, and test sets was maintained as previously defined in section 5.1.2, with the process of dividing into sub-images carried out independently for each subset. In this way, the integrity and separation of data between the different stages of model development were ensured.

After splitting the images, data augmentation techniques were applied exclusively to the training set. Only five operations from those listed in Table 5.2 were selected, since the number of images had already increased substantially with the image split, making it unnecessary to use a wider range of transformations. The chosen operations were: illumination, scale07 with emboss, scale07 with Gaussian blur, scale13 with contrast, and scale13 with iso noise. These transformations aim to increase the variability of the training set by simulating different environmental conditions, which are crucial for improving the model's robustness and generalization ability in real-world scenarios. By artificially diversifying the data, the risk of overfitting is reduced and the

model's performance is enhanced. Table 5.3 presents the distribution of the number of images per set after this process.

	Train	Train Augmented	Val	Test
Images	2100	10500	600	300

Table 5.3: Distribution of the number of images per set after image splitting.

This method was applied to the Arbutus Berry Detection Dataset [3] and the Arbutus Berry Ripeness Level Detection Dataset [4]. By segmenting the original images into smaller crops, this approach created a more robust and diverse dataset, essential for training models to detect and classify Arbutus berries at different ripening stages. One of the main advantages of this process is the generation of sub-images that sometimes do not contain any Arbutus berries, which provides valuable negative examples and increases the model's ability to distinguish between regions with and without fruit. The methodology adopted in this approach is the same as that applied to the complete images and is illustrated in Figure 5.3.

5.1.4 Deep Learning Models

The evolution of the YOLO family, from YOLOv8 to YOLOv11, alongside advanced models such as Real-Time Detection Transformer (RT-DETR), has been marked by significant innovations in efficiency, precision, and adaptability for computer vision tasks. These latest architectures emphasize efficient training, robust generalization to complex environments, and a softer design, making them highly suitable for real-time applications.

The YOLOv8 began a new generation by abandoning anchors and adopting anchor-free detection, simplifying the architecture and improving performance, especially on small objects. Its structural basis is the Cross-Stage Feature Fusion (C2f) block, which stood out for its efficient extraction of features at a lower computational cost. In addition, the model provided a unified architecture for various tasks: detection, segmentation, tracking, pose estimation, and classification. The Spatial Pyramid Pooling - Fast (SPPF) module was used to represent multiple scales, improving accuracy for objects of different sizes [32].

In 2024, with YOLOv9, there was a significant evolution in terms of feature aggregation and gradient flow. The model introduced Programmable Gradient Information (PGI) to improve back-propagation during training and avoid information loss in deep networks. The Generalized Efficient Layer Aggregation Network (GELAN) architecture was incorporated to make feature fusion more efficient. In addition, an auxiliary module was added exclusively for training, optimizing learning without impacting inference. Detection benefited from multiple additional heads, which improved recognition of objects of different scales [32].

Also, in 2024, the YOLOv10 brought about a paradigm shift by eliminating the Non-Maximum Suppression (NMS) step, using a new dual assignment strategy, which speeds up training and simplifies inference. This version also introduced C2fCIB, which replaces blocks in layers with high

redundancy using an "intrinsic rank" approach, increasing efficiency without sacrificing accuracy. Spatial-Channel Decoupled Downsampling (SCDown) helped to reduce the loss of information in spatial downsampling, and the Partial Self-Attention (PSA) module introduced more efficient attention mechanisms at deeper stages of the network [32].

In YOLO11, architectural refinement has reached a new level. The new C3k2 block, the successor to the C2f, features a lighter structure with better gradient flow. It replaces the traditional bottlenecks with C3k modules with and without shortcuts, adapting to the complexity required at each stage. The C2PSA module, an evolution of PSA, improves spatial attention by increasing sensitivity to partially occluded objects or complex visual contexts. YOLOv11 uses SPPF, concat, and upsample but improves accuracy by integrating these components more efficiently. Like YOLOv8, this model also incorporates a classifier for image classification tasks [32][40].

RT-DETR represents a major breakthrough in transformer-based object detectors, being the first end-to-end model capable of real-time operation while overcoming the high computational costs typical of classical DETRs. Its architecture integrates a convolutional backbone (such as ResNet), whose multi-stage feature maps feed an efficient hybrid encoder. This encoder decouples intra-scale feature interaction—handled by attention mechanisms at higher semantic levels—from cross-scale fusion, which is managed by CNN-based modules, optimizing both speed and accuracy. RT-DETR further introduces an uncertainty-minimal query selection mechanism, ensuring only the most relevant features are used by the decoder, thereby improving detection precision and stability. The model's flexible design allows inference speed to be adjusted by varying the number of decoder layers without retraining, making it highly adaptable to diverse real-time scenarios. With these innovations, RT-DETR surpasses advanced YOLO models in both speed and accuracy, setting a new standard for real-time object detection tasks [41].

The DL models selected for this task were YOLOv8 Nano, YOLOv9 Tiny, YOLOv10 Nano, YOLO11 Nano and RT-DETR Large. These models were chosen because of their efficiency, good performance in accurate, real-time object detection, and classification capabilities. The "Nano" and "Tiny" versions are particularly suitable for applications requiring low consumption of computing resources while maintaining a high hit rate in identifying and differentiating the objects of interest.

These models were trained using the Ultralytics platform [14], an open-source tool developed to facilitate the training and implementation of computer vision models. This platform supports developing and improving YOLO models and simplifies the training, validation, and testing processes in object detection, segmentation, and classification tasks.

5.1.5 Training Configuration

In this work, two distinct approaches were explored for model training. In the first approach, a detection model was trained solely to locate the Arbutus berry in the images, followed by a separate classifier responsible for categorizing the fruits into three ripeness level classes: green, yellow-orange, and red (left side of Figure 5.3). This two-stage approach separated the tasks of localization and classification, aiming to improve precision, particularly in complex scenarios such

as those involving Arbutus berry, where the fruits are often very small, densely distributed within the images, and very often occluded by leaves. The second approach trained a multiclass detection model to detect and classify Arbutus berries into the three classes (right side of Figure 5.3). This strategy aims to simplify the inference pipeline by enabling the model to perform both tasks in an integrated manner. The same approaches are applied when using image splitting, ensuring consistency and comparability across both the original and split-image workflows.

The training process for the YOLO models began with the use of the Adaptive Moment Estimation (Adam) optimizer, a learning rate (lr) of 0.01, and a batch size of 16. However, the initial results were unsatisfactory, as shown in Figure 5.4a, where the precision and recall metrics were quite unstable throughout the epochs. This instability is primarily because Arbutus berries are tiny fruits and appear in the dataset at varying distances, making it challenging for the model to learn effectively. Several adjustments were made to the parameters, including changes to the learning rate and image size, but these did not significantly improve the results. The most impactful change occurred when the optimizer was switched to Stochastic Gradient Descent (SGD) and the batch size was increased to 32. Additionally, the nano versions of the YOLO models were used, which resulted in considerably more stable metric graphs, as shown in Figure 5.4b.

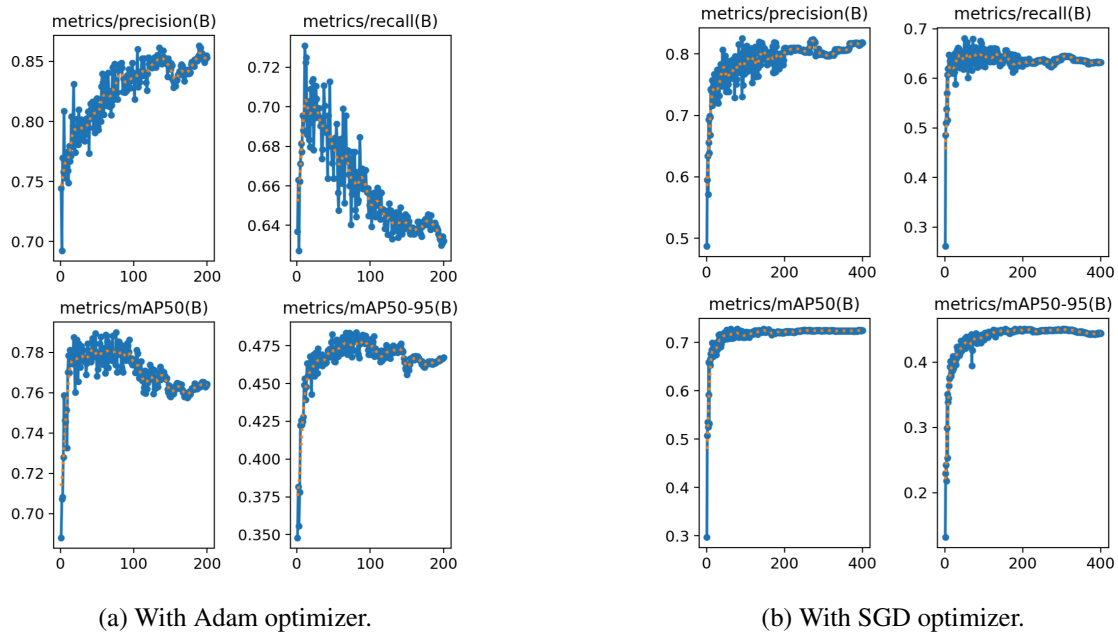


Figure 5.4: Training metric evolution for YOLO models with different optimization strategies.

The Adam optimizer automatically adapts the learning rate for each parameter during training, combining the advantages of momentum and RMSprop. This generally results in faster and more efficient convergence, especially in deep networks and with complex data [42]. In contrast, SGD uses a fixed learning rate and updates parameters based on small batches of data. SGD is known for its simplicity, computational efficiency, and good generalization, although it may require more epochs to converge and is more sensitive to hyperparameter selection[43].

All the YOLO models were trained with the following parameters: 400 epochs with an early stopping set at 200 epochs, SGD optimizer, batch size of 32, learning rate of 0.01, and input resolution of 640×640 . This configuration yielded the best results, demonstrating a good balance between performance, training time, and generalization capacity.

The RT-DETR model is available in only two variants: Large and Extra Large, with no lighter versions. For this reason, the Large version, the most compact among those available, was chosen. Initially, the training followed the same hyperparameter configuration used for the YOLO models. However, the results showed significant instability, as illustrated in Figure 5.5a, where the precision and recall metrics are markedly unstable throughout the epochs. Stability in the metrics was only achieved after replacing the optimizer with AdamW and reducing the learning rate to $1e-4$. These changes resulted in a much more consistent evolution of the performance indicators, as shown in Figure 5.5b. In this new configuration, the loss curves and metrics display a more stable behavior, indicating a significant improvement in the model's generalization capability. The train configuration for RT-DETR-L is 400 epochs with an early stopping set at 200 epochs, Adam with Weight Decay (AdamW) optimizer, batch size of 32, learning rate of 0.0001, and input resolution of 640×640 .

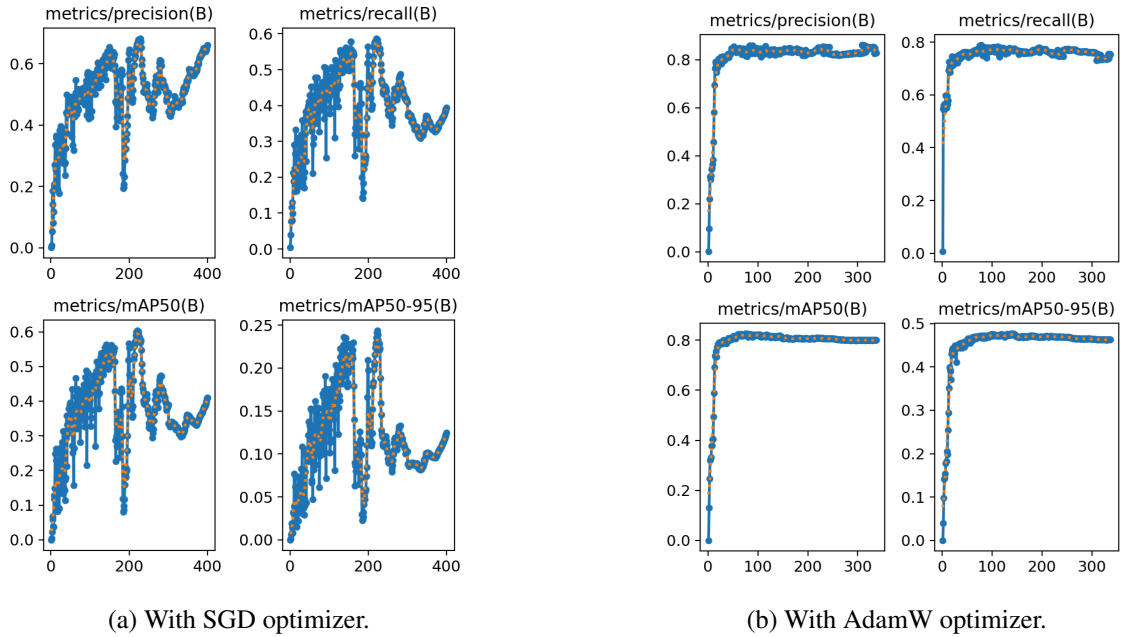


Figure 5.5: Training metric evolution for RT-DETR models with different optimization strategies.

The choice of optimizers and the fine-tuning of hyperparameters were crucial for achieving stable and robust results, especially considering the added difficulty of detecting small and densely clustered fruits like *Arbutus* berries.

Table 5.4 shows the number of epochs each model was trained for in the different training tasks using the original images.

Table 5.4: Number of epochs each model was trained for the different tasks.

Model	Detection 3 class	Detection	Classification
YOLOv8n	400	359	200
YOLOv9t	400	400	-
YOLOv10n	400	400	-
YOLO11n	400	400	200
RT-DETR-L	338	376	-

Table 5.5 presents the corresponding number of training epochs for each model when using the image splitting dataset.

Table 5.5: Number of epochs each model was trained for the different tasks with the image splitting dataset.

Model	Detection 3 class	Detection
YOLOv8n	236	256
YOLOv9t	238	381
YOLOv10n	255	325
YOLO11n	228	247

5.2 Results

The FiftyOne platform [44] was used to evaluate the results of each training, which allowed a detailed analysis of the performance of the four networks. This tool provided a comprehensive visualization of each model’s detection and classification capabilities, making it easy to identify strengths and limitations in different scenarios.

5.2.1 Methodology

The selection of evaluation methods for DL models is based on different perspectives of model performance. Therefore, using a combination of metrics is a common practice to gain a more comprehensive understanding of the model’s behavior. This work used these specific analyses: precision, recall, F1 score, average precision (AP), and mAP. To plan these analyses, let us consider the following set of concepts:

- Intersection over Union (IoU): a measure based on the Jaccard index that is used to measure the overlap between two bounding boxes using the ground truth and the predicted bounding boxes;
- True Positive (TP): correct detection of a ground truth bounding box, i.e., $\text{IoU} \geq \text{threshold}$;

- False Positive (FP): incorrect detection of a ground truth bounding box, i.e., $\text{IoU} < \text{threshold}$;
- False Negative (FN): a ground truth bounding box not detected.

Based on these definitions, the following evaluation metrics were used:

- Precision: measures the accuracy of positive predictions, the percentage of TP out of all detections, and is given by:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (5.1)$$

- Recall: measures the model's ability to find all ground-truth bounding boxes, i.e., the percentage of TP detections divided by all the ground truths, and it's calculated:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (5.2)$$

- F1 score: represents the harmonic mean of precision and recall, and is given by:

$$\text{F1score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (5.3)$$

- AP: compares the performance of object detectors to calculate the area under the precision $\times$ recall curves. A high area represents both high precision and recall;
- mAP: is a metric used to measure the accuracy of object detectors across all classes, calculated as the mean AP for all the object classes.

To evaluate the performance of each model, the test set was used for inference, applying an IoU threshold of 70% and a confidence threshold of 0.25. Additionally, performance was also assessed using other confidence levels, including 0.5 and the threshold that maximized the F1 score on the validation set. However, the most balanced and consistent results were obtained with the confidence threshold set at 0.25.

5.2.2 Detection for Arbutus Berry Ripeness Stage

The results presented in this section correspond to the detection of Arbutus berries categorized into three ripeness stages, using the Arbutus Berry Ripeness Level Detection Dataset [4]. The detection model was trained to predict the ripeness class of each fruit directly. Figure 5.6 illustrates an example of the detection process, where the bounding box colors reflect the ripeness stage assigned by the model.



Figure 5.6: Example of Arbutus berries detection for each class.

Analyzing class metrics provides a deeper understanding of each model's performance. Table 5.6 presents the average performance metrics for each model across the three Arbutus berry classes: green, red, and yellow-orange. Overall, all models achieved the highest precision for the red class, with values above 97%, indicating strong confidence in identifying red Arbutus berries. However, recall and F1 scores varied more significantly across classes and models.

The RT-DETR-L model demonstrated the strongest and most balanced overall performance among all models, as reflected in its total metrics: precision of 95.3%, recall of 86.4%, F1 score of 90.6%, and AP of 50.5%. This confirms RT-DETR-L's superiority for multi-class detection of Arbutus berry ripeness stages. The model particularly excelled in the yellow class, achieving the highest recall (85.2%), F1 score (89.1%), and AP (54.7%), while also maintaining leading results for green (F1: 85.9%, AP: 51.9%) and red (F1: 92.1%, AP: 44.7%) berries. These results highlight RT-DETR-L's robustness and reliability across all ripeness stages, outperforming the YOLO variants in both class-specific and overall detection metrics. Among the YOLO models, YOLOv10n stood out for its balanced results, achieving the highest recall (79.1%) and F1 score (85.5%) for the green class, as well as strong F1 scores for red (86.4%) and yellow (80.3%) berries. YOLOv11n also showed consistent performance, with high F1 scores for green (84.3%) and red (87.9%), and solid results for yellow (79.6%). YOLOv8n achieved the highest F1 score for the red class among the YOLO variants (86.6%) and maintained high precision. Still, its recall for yellow (69.6%) was the lowest, indicating some difficulty detecting yellow berries. YOLOv9t excelled in precision, particularly for the red class (99.2%), and showed competitive F1 scores across all classes, although its recall was slightly lower than the best-performing models.

Table 5.6: Results on the performance metrics for Arbutus berry ripeness stage detection.

Model	Class	Precision(%)	Recall(%)	F1(%)	AP(%)
YOLOv8n	Green	94.2	73.1	82.4	43.5
	Red	98.5	77.3	86.6	35.9
	Yellow	93.0	69.6	79.6	49.4
	Total	96.5	74.7	84.2	42.5
YOLOv9t	Green	96.1	73.1	83.1	44.5
	Red	99.2	77.3	86.9	39.4
	Yellow	91.5	74.8	82.3	49.7
	Total	97.0	76.2	85.4	44.5
YOLOv10n	Green	93.0	79.1	85.5	50.1
	Red	97.3	77.6	86.4	36.9
	Yellow	92.1	71.3	80.4	50.3
	Total	95.6	76.4	84.9	45.8
YOLO11n	Green	94.4	76.1	84.3	47.6
	Red	98.9	79.2	87.9	37.8
	Yellow	92.6	75.7	83.3	51.0
	Total	96.6	77.6	86.1	45.0
RT-DETR-L	Green	90.2	82.1	85.9	51.9
	Red	97.0	87.6	92.1	44.7
	Yellow	93.3	85.2	89.1	54.8
	Total	95.3	86.4	90.6	50.5

Table 5.7 presents the number of detections per class and the total detections. The test set includes 68 green fruits, 331 red fruits, and 115 yellow fruits, totaling 514 Arbutus berries.

Among all the evaluated models, RT-DETR-L achieved the best overall detection performance, surpassing all YOLO variants. RT-DETR-L recorded the highest total number of TP across all classes (443) and the lowest number of FN (70), confirming its superior effectiveness and robustness for multi-class detection of Arbutus berry ripeness stages. Within the YOLO models, YOLOv11n stands out for its strong detection capability, achieving the highest TP for both the red (262) and yellow (86) classes, as well as the lowest FN for these classes (69 and 29, respectively), indicating particular effectiveness in identifying berries at these ripeness stages. YOLOv10n distinguishes itself in the green class, with the highest TP (53) and the lowest FN (14), suggesting greater sensitivity for detecting green berries than the other YOLO models. Meanwhile, YOLOv9t achieves the lowest overall number of FP (12), reflecting high precision, although its FN values are slightly higher, which may impact recall.

Table 5.7: Detection results of Arbutus berry ripeness stage.

Model	Class	TP	FP	FN
YOLOv8n	Green	49	3	19
	Red	256	4	75
	Yellow	79	7	36
	Total	384	14	130
YOLOv9t	Green	49	2	18
	Red	256	2	75
	Yellow	86	8	29
	Total	391	12	122
YOLOv10n	Green	53	4	14
	Red	257	7	74
	Yellow	82	7	33
	Total	392	18	121
YOLO11n	Green	51	3	17
	Red	262	3	69
	Yellow	86	8	29
	Total	399	14	115
RT-DETR-L	Green	55	6	12
	Red	290	9	41
	Yellow	98	7	17
	Total	443	22	70

The analysis of FP and FN is crucial to understand the limitations of the models and identify scenarios where the networks fail. A significant number of FN is noted across all models, mainly due to small Arbutus berries in the images (Figure 5.7), making their detection challenging and directly impacting overall performance, particularly the recall and AP values. Although F1 scores for most classes are high, AP values remained relatively low due to the difficulty in detecting these small berries, one of the main challenges of the dataset. Most FP occur in regions where the Arbutus berries overlap — the model tends to detect an Arbutus berry at the intersection between two fruits. Additionally, in some situations, incorrect detections are also made on the tree trunk, usually due to lighting conditions that confuse the model. Figure X shows an example of the FP that can occur in these scenarios.



Figure 5.7: Examples of false negatives in the detection of Arbutus berries.

5.2.3 Detection and classification of Different Classes of Arbutus Berry Ripeness

5.2.3.1 Detection of Arbutus berry

The following results present the detection of Arbutus berries, based on the Arbutus Berry Detection Dataset [3]. The analysis of Table 5.8 provides an overview of the performance of models trained exclusively for Arbutus berry detection without class distinction. All the models exhibited solid precision, recall, and F1 score performance. YOLOv8n and YOLOv9t achieved the highest precision values, with 98.1% and 98.8%, respectively, reflecting high confidence in their detections. In terms of recall, RT-DETR-L stood out with the highest value (88.8%), followed by YOLOv8n (78.8%) and YOLOv10n (77.8%). RT-DETR-L also achieved the highest F1 score (91.9%), indicating a balanced and robust detection performance. Regarding mAP, RT-DETR-L again led with 53.2%, surpassing all YOLO variants, whose mAP values ranged from 46.3% to 47.8%. These results confirm RT-DETR-L as the most effective model for Arbutus berry detection in this evaluation, while the YOLO models also maintained consistently high precision and competitive F1 scores.

Table 5.8: Results on the performance metrics for Arbutus berry detection.

Model	Precision(%)	Recall(%)	F1(%)	mAP(%)
YOLOv8n	98.1	78.8	87.4	47.4
YOLOv9t	98.8	78.2	87.3	46.3
YOLOv10n	95.7	77.8	85.8	47.8
YOLO11n	98	77.6	86.6	47.2
RT-DETR-L	95.2	88.8	91.9	53.2

The test set used in this evaluation contains a total of 518 annotated Arbutus berries, including fruits of different sizes, orientations, and visibility conditions. Table 5.9 presents the absolute detection results for each model trained exclusively for detection. The RT-DETR-L model obtained the highest number of TP with 460 and the lowest number of FN at 58, confirming its superior detection capability. Among the YOLO models, YOLOv8n recorded the highest TP (408) and a low number of FP (8), while YOLOv9t demonstrated the most balanced performance, with 405 TP, the lowest FP (5), and 113 FN. YOLO11n and YOLOv10n showed similar results, with 402 and 403 TP, and 116 and 115 FN, respectively, but YOLO11n maintained a lower FP (8) compared to YOLOv10n (18).

Table 5.9: Detection results of Arbutus berry.

Model	TP	FP	FN
YOLOv8n	408	8	110
YOLOv9t	405	5	113
YOLOv10n	403	18	115
YOLO11n	402	8	116
RT-DETR-L	460	23	58

5.2.3.2 Classification of Arbutus Berry Ripeness Level

The following results correspond to the classification of the ripening stages of Arbutus berries, using the Arbutus Berry Ripeness Level Classification Dataset [4]. Table 5.10 presents the performance results of the classifier models based on the metrics. Both models performed excellently in classifying the red class, with precision and recall values exceeding 97% and 99%, respectively. The YOLO11n-cls model achieved the highest performance for the yellow class, with a precision of 99.1%, recall of 94.8%, and an F1 score of 96.9%. In the green class, YOLO11n-cls also outperformed YOLOv8n-cls, reaching a precision of 95.7%, recall of 98.5%, and an F1 score of 97.1%. Despite the green class presenting the lowest scores overall, both models maintained F1 values above 92%, confirming consistently high classification performance across all ripening stages.

Table 5.10: Results on the performance metrics for Arbutus berry classification.

Model	Class	Precision(%)	Recall(%)	F1(%)
YOLOv8n-cls	Green	92.6	92.6	92.6
	Red	97.6	99.4	98.5
	Yellow	95.4	90.4	92.9
YOLO11n-cls	Green	95.7	98.5	97.1
	Red	98.8	99.7	99.2
	Yellow	99.1	94.8	96.9

Table 5.11 reinforces the previously observed patterns. The test set is distributed among 68 green fruits, 331 red fruits, and 115 yellow fruits, and includes 514 Arbutus berries, as detailed in subsection 5.2.2. Both versions of the model could correctly identify almost all red Arbutus berries (329 and 330 TP) with few errors. However, there was some confusion between the green and yellow classes and the yellow and red classes, resulting in FP and FN. This confusion may be related to the similarity of hues between these classes at certain stages of ripening, as well as to variations in lighting conditions, which make it difficult to distinguish precisely between them. Overall, the YOLO11n-cls model demonstrated superior performance across all classes, achieving higher precision, recall, and F1 scores, and fewer classification errors than the YOLOv8n-cls model.

Table 5.11: Classification results of Arbutus berry.

Model	Class	TP	FP	FN
YOLOv8n-cls	Green	63	5	5
	Red	329	8	2
	Yellow	104	5	11
YOLO11n-cls	Green	67	3	1
	Red	330	4	1
	Yellow	109	1	6

5.2.3.3 Detection and Classification of Arbutus Berry Ripeness Level

The following results present the performance of the detectors, followed by the classification stage. The YOLO11n-cls model was used for classification, as it achieved the best results and was applied uniformly across all detectors. The analysis of Table 5.12 provides an overview of the performance of the models in detecting and classifying the ripening stages of Arbutus berries. All models demonstrated solid results regarding precision, recall, and F1 score across the three classes. The YOLOv8n model achieved the highest F1 scores for both the green (89.6%) and red (87.9%) classes, as well as the highest AP for the green class (51.3%), indicating robust and balanced performance. RT-DETR-L stood out in the yellow class, obtaining the highest recall (87.0%), F1 score (87.0%), and AP (54.8%), surpassing all YOLO variants in this category. YOLOv9t maintained

the highest precision for both green (98.1%) and red (99.3%) classes, reflecting high confidence in its detections, but showed slightly lower recall, particularly for the green class (75.0%).

Overall, while AP values ranged from 36.4% to 54.8%, reflecting the inherent challenges in distinguishing between ripeness stages due to color similarities and environmental factors, the results confirm that both YOLO and RT-DETR models deliver consistently strong detection and classification performance for Arbutus berry ripeness assessment.

Table 5.12: Results on the performance metrics for Arbutus berry classification and detection.

Model	Class	Precision(%)	Recall(%)	F1(%)	AP(%)
YOLOv8n	Green	98.2	82.4	89.6	51.3
	Red	97.4	80.1	87.9	36.4
	Yellow	95.4	72.2	82.2	49.8
YOLOv9t	Green	98.1	75.0	85.0	45.6
	Red	99.3	80.1	88.6	37.2
	Yellow	92.3	73.0	81.6	49.5
YOLOv10n	Green	96.5	80.9	88.0	51.5
	Red	95.2	78.5	86.1	37.9
	Yellow	92.3	73.0	81.6	50.2
YOLO11n	Green	96.4	77.9	86.2	48.7
	Red	97.7	77.9	86.7	37.4
	Yellow	92.3	73.0	81.6	49.6
RT-DETR-L	Green	93.1	79.4	85.7	50.2
	Red	93.6	87.9	90.7	44.0
	Yellow	87.7	87.0	87.3	54.8

Table 5.13 presents each class's detection and classification results using the test set. The test set is the same as the one used for the classifier. RT-DETR-L achieved the highest number of TP in the yellow class 100 and maintained strong results for green 54 TP and red 291 TP, though it also had the highest FP in yellow 14 and red 20. The highest FP values, especially for RT-DETR-L, are related to correctly detected fruits, but assigning a class is impossible due to ambiguous lighting conditions. Among the YOLO models, YOLOv8n minimized FP overall (total of 17 across all classes) and achieved the highest TP for the red class (265), though it showed a higher number of false negatives (FN), particularly in the red (66) and yellow (32) classes. YOLOv9t demonstrated the lowest FP in the red class (2), reflecting high precision, but had the highest FN for green (17), indicating reduced recall for this stage. YOLO11n and YOLOv10n maintained a good balance between TP and FP, with consistent performance across classes.

Table 5.13: Detection and classification results of Arbutus berry.

Model	Class	TP	FP	FN
YOLOv8n	Green	55	2	12
	Red	265	7	66
	Yellow	83	4	32
YOLOv9t	Green	50	2	17
	Red	265	2	66
	Yellow	84	7	31
YOLOv10n	Green	54	3	13
	Red	260	13	71
	Yellow	84	7	31
YOLO11n	Green	52	3	15
	Red	258	6	73
	Yellow	84	7	31
RT-DETR-L	Green	54	4	13
	Red	291	20	40
	Yellow	100	14	15

5.2.4 Image Splitting

The results in this section correspond to the detection of Arbutus berries classified into three ripeness stages, utilizing the Arbutus Berry Ripeness Level Detection Dataset [4] with image splitting. The detection model was trained to predict the ripeness stage of each fruit directly. Figure 5.8 shows an example of Arbutus berry detection and classification using image splitting, where each sub-image corresponds to one of the thirty segments produced by the image split process and demonstrates the identification of berries at various ripeness stages.

The annotations from the different image splits were consolidated into a single file corresponding to the original image to enable a more precise and consistent evaluation of the models' performance. At this stage, only models from the YOLO family were evaluated, as they had previously shown less satisfactory results, with the aim of verifying whether their performance would improve when using image splitting.

The merging of annotations eliminated duplications of Arbutus berries that were fully present in the overlapping areas between image crops. However, berries that were only partially detected in these overlapping zones could not always be fully filtered, resulting in some residual duplications. To address this issue, several approaches were tested, including the temporary expansion of bounding boxes for IoU calculation and the combination of IoU with the distance between centers and size difference. These approaches had limitations, such as removing correctly detected berries close to others and failing to remove all partial duplicates. Therefore, the adopted approach was based on IoU calculation, prioritizing the preservation of the detection with the highest confidence.



Figure 5.8: Example of Arbutus berries detection for each class using image splitting.

This solution provided the best removal of partial duplicates, although it could not eliminate them entirely.

It is therefore essential to note that some false positives (FP) present in the results are a consequence of these partial detections in the overlapping zones that could not be completely filtered. These methodological considerations are essential for correctly interpreting the results presented in the analysis of image crops and for understanding the sources of errors.

5.2.4.1 Detection for Arbutus Berry Ripeness Stage with image splitting

Table 5.14 provides a performance of each YOLO model for Arbutus berry ripeness stage detection using image splitting. Overall, all models achieved the highest accuracies in the red class, with values close to or greater than 90%, indicating high confidence in identifying ripe fruits. Despite this, recall and F1 values also remained high and close to precision, which reflects a globally balanced and consistent performance across specifications.

The YOLOv8n model demonstrated the most balanced overall performance, achieving the highest precision (90.3%), recall (90.7%), and F1-score (90.5%) among the models, as well as recording the highest mean AP (53.6%). This model also delivered the best results for the red class (F1: 91.9%) and the highest F1-score for the yellow class (88.0%), highlighting its effectiveness in detecting multiple ripeness stages.

Among the others, YOLOv10n stood out with the highest AP for the green class (56.0%) and yellow class (54.0%). The YOLOv9t model achieved the highest precision for the green class

(87.5%) and competitive results across all classes, with F1-scores of 84.8% (green), 90.6% (red), and 84.8% (yellow). Meanwhile, YOLOv11n showed the lowest precision for the green class (77.3%) but maintained strong performance for the red and yellow classes.

Table 5.14: Results on the performance metrics for Arbutus berry ripeness stage detection with image splitting.

Model	Class	Precision(%)	Recall(%)	F1(%)	AP(%)
YOLOv8n	Green	87.0	88.2	87.6	53.8
	Red	91.1	92.7	91.9	50.3
	Yellow	90.0	86.1	88.0	56.7
	Total	90.3	90.7	90.5	53.6
YOLOv9t	Green	87.5	82.4	84.8	53.2
	Red	89.9	91.2	90.6	43.9
	Yellow	89.4	80.9	84.8	55.2
	Total	89.5	87.7	88.6	50.8
YOLOv10n	Green	86.6	85.3	85.9	56.0
	Red	90.8	89.4	90.1	46.2
	Yellow	91.1	80.0	85.2	54.1
	Total	90.3	86.8	88.5	52.1
YOLOv11n	Green	77.3	85.3	81.1	52.9
	Red	90.9	90.9	90.9	44.7
	Yellow	89.9	77.4	83.2	55.2
	Total	88.7	87.2	87.9	50.9

Table 5.15 presents the number of detections per class and the total number of detections for the ripeness stages of Arbutus berries using image splitting. The test set includes 68 green fruits, 331 red fruits, and 115 yellow fruits, totaling 514 fruits—the same as used in the analysis Detection for Arbutus Berry Ripeness Stage (section 5.2.2).

The YOLOv8n model achieved the best overall performance, with the highest total number of true positives (466) and the lowest number of false negatives (48), demonstrating its superior effectiveness and robustness for multiclass detection. It also stood out in the detection of the red class, with the highest TP (307) and the lowest FN (24), indicating high sensitivity in identifying ripe fruits. On the other hand, YOLOv10n showed superior performance in the green class, with a TP of 58 and only 10 FN, suggesting greater accuracy in detecting unripe fruits. For the yellow class, the best TP performance was achieved by YOLOv8n (99), although YOLOv10n recorded the lowest number of FPs (9) in this category. The YOLOv10n model distinguished itself by achieving the lowest total number of FPs (48), reflecting higher overall precision in its detections, although its FN values were slightly higher than those of YOLOv8n.

Table 5.15: Detection results of Arbutus berry ripeness stage with image splitting.

Model	Class	TP	FP	FN
YOLOv8n	Green	60	9	8
	Red	307	30	24
	Yellow	99	11	16
	Total	466	50	48
YOLOv9t	Green	56	8	12
	Red	302	34	29
	Yellow	93	11	22
	Total	451	53	63
YOLOv10n	Green	58	9	10
	Red	296	30	35
	Yellow	92	9	23
	Total	446	48	68
YOLO11n	Green	58	17	10
	Red	301	30	30
	Yellow	89	10	26
	Total	448	57	66

With the image splitting approach, there is a significant reduction in the number of FN across all YOLO models, indicating a clear improvement in sensitivity and the ability to detect fruits, especially those of smaller size. However, this improvement in detection has also led to an increase in the number of false positives (FP), a phenomenon partly due to the previously mentioned issue of partial duplications resulting from overlapping image splits. Additionally, because the images are more zoomed in compared to the original, the models tend to detect more tree trunks, dried flowers, and yellowed leaves as if they were Arbutus berries, contributing to the rise in FP. This behavior is especially noticeable in environments with high vegetation density or under challenging lighting conditions, where distinguishing between fruit and other elements becomes more difficult.

5.2.4.2 Detection and classification of Different Classes of Arbutus Berry Ripeness with image splitting

Detection of Arbutus berry with image splitting

Table 5.16 presents the performance metrics for Arbutus berry detection using image splitting. All evaluated models demonstrated solid results across precision, recall, and F1-score, indicating their reliability for this detection task.

Among the models, YOLOv9t and YOLOv10n achieved the highest precision values, with 89.2% and 89.6%, respectively, reflecting strong confidence in their positive detections. In terms of recall, YOLOv8n stood out with the highest value (91.5%), followed closely by YOLOv9t

(90.7%) and YOLOv10n (90.0%), highlighting their effectiveness in identifying the majority of berries present in the dataset. Regarding the F1-score, which balances both precision and recall, YOLOv10n achieved the highest value (89.8%), with YOLOv8n and YOLOv9t also maintaining competitive scores (88.7% and 90.0%, respectively). When analyzing mAP, YOLOv9t led with 54.6%, followed by YOLOv8n (54.0%). The mAP values for all models were close, ranging from 53.3% to 54.6%, indicating consistent performance in overall detection accuracy.

Table 5.16: Results on the performance metrics for Arbutus berry detection with image splitting.

Model	Precision(%)	Recall(%)	F1(%)	mAP(%)
YOLOv8n	86.0	91.5	88.7	54.0
YOLOv9t	89.2	90.7	90.0	54.6
YOLOv10n	89.6	90.0	89.8	53.4
YOLO11n	88.0	89.2	88.6	53.3

Table 5.17 presents the absolute detection results for Arbutus berry identification using the image splitting technique. The test set used in this evaluation consists of 518 annotated berries, the same as employed in Section 5.2.3.1.

Among the models analyzed, YOLOv8n stood out by achieving the highest number of TP, with 474 correct detections, and the lowest number of FN, totaling 44. However, this model also recorded the highest number of FP, with 77 occurrences. YOLO11n demonstrated balanced performance, reaching 470 TP, 57 FP, and 48 FN, indicating a good trade-off between precision and sensitivity. YOLOv10n achieved the lowest number of FP (54), with 466 TP and 52 FN, indicating high precision, although with a slight increase in FN compared to YOLOv8n. Finally, YOLOv9t obtained 462 TP, 63 FP, and 56 FN, maintaining consistent performance, but with FP and FN values slightly higher than the other models.

Table 5.17: Detection results of Arbutus berry with image splitting.

Model	TP	FP	FN
YOLOv8n	474	77	44
YOLOv9t	470	57	48
YOLOv10n	466	54	52
YOLO11n	462	63	56

Detection and Classification of Arbutus Berry Ripeness Level with image splitting

Table 5.18 summarizes the performance metrics for Arbutus berry detection and classification using image splitting. All YOLO models demonstrated consistent and solid results regarding precision, recall, F1-score, and AP for all classes.

YOLOv10n achieved the highest precision (88.5%) and F1-score (89.4%) for the red class, indicating strong performance in detecting ripe berries. YOLOv8n showed the best F1-score for the green class (82.6%) and the highest AP for green (52.6%), suggesting robust and balanced

detection capabilities. For the yellow class, YOLOv9t led with the highest precision (90.7%) and a strong F1-score (87.4%), while YOLOv8n and YOLOv10n also maintained competitive results.

Overall, the AP values ranged from 45.0% to 54.5% across the models, which are relatively high compared to the values obtained with complete images, but still reflect the challenges posed by the presence of tiny *Arbutus* berries. Despite these difficulties, the results demonstrate that all YOLO variants provide reliable and effective performance for detecting and classifying *Arbutus* berry ripeness stages when using image splitting.

Table 5.18: Results on the performance metrics for *Arbutus* berry classification and detection with image splitting.

Model	Class	Precision(%)	Recall(%)	F1(%)	AP(%)
YOLOv8n	Green	81.4	83.8	82.6	52.6
	Red	82.6	91.5	86.8	45.0
	Yellow	83.3	82.6	83.0	54.5
YOLOv9t	Green	79.7	80.9	80.3	51.2
	Red	85.8	90.9	88.3	46.6
	Yellow	90.7	84.3	87.4	54.3
YOLOv10n	Green	83.1	79.4	81.2	50.3
	Red	88.5	90.3	89.4	45.6
	Yellow	82.9	84.3	83.6	53.6
YOLO11n	Green	80.6	79.4	80.0	50.3
	Red	86.7	90.9	88.8	45.6
	Yellow	86.5	83.5	85.0	53.4

Table 5.19 presents the detection and classification results for each ripeness class of *Arbutus* berries using image splitting. The test set is the same as used for the Detection for *Arbutus* Berry Ripeness Stage (section 5.2.2). The YOLOv8n model achieved the highest number of TP in the red class (303) and strong results for green (57 TP). However, it also recorded the highest number of FP in the red (64) and yellow (19) classes. YOLOv9t demonstrated a balanced performance, with 301 TP for red, 55 TP for green, and 97 TP for yellow berries, while achieving the lowest FP in the yellow class (10). YOLOv10n stood out for having the lowest FP in the green class (11) and maintained competitive TP values across all classes, particularly in yellow (97 TP) and red (299 TP). However, it showed slightly higher false negatives (FN) in the green (14) and red (32) classes compared to YOLOv8n.

Table 5.19: Detection and classification results of Arbutus berry with image splitting.

Model	Class	TP	FP	FN
YOLOv8n	Green	57	13	11
	Red	303	64	28
	Yellow	95	19	20
YOLOv9t	Green	55	14	13
	Red	301	50	30
	Yellow	97	10	18
YOLOv10n	Green	54	11	14
	Red	299	39	32
	Yellow	97	20	18
YOLO11n	Green	54	13	14
	Red	301	46	30
	Yellow	96	15	19

5.3 Discussion

When comparing the detection and classification approach (Tables 5.12 and 5.13) with the three class detection approach (Tables 5.6 and 5.7), it becomes evident that the two-stage method—combining general detection followed by separate classification—yields better overall performance, this difference is primarily due to the distinct annotation strategies employed in the datasets used for training each model.

In the detection followed by classification approach, the detection model was trained with annotations corresponding to all visible Arbutus berries, including those whose color was not distinguishable due to adverse conditions such as poor illumination, occlusion, or distance. Conversely, the three class detection models were trained exclusively on berries with clearly identifiable ripeness stages, thus limiting the representativeness of the training data and resulting in lower TP values.

This distinction directly influences the models' detection capabilities: general detection models exhibit more TP as they can detect more Arbutus berries under challenging environmental conditions, even when color classification is not feasible. Subsequently, by applying a dedicated classifier to the detected fruits, it is possible to determine the ripeness stage accurately.

Overall, the two-stage approach, general detection followed by classification, not only achieves broader detection coverage but also enhances the precision of ripeness classification compared to the three class detection models. These findings highlight the greater robustness and effectiveness of the detection and classification strategy, particularly in complex real-world scenarios where fruit visibility and distinguishability are variable.

When these same approaches are compared using image splitting (Tables 5.18 and 5.19 versus Tables 5.14 and 5.15), the differences between the methods become less pronounced than when

using full images. This is because, by dividing the images into smaller segments, the models can detect more fruits. Nevertheless, the detection followed by classification approach still identifies more fruits than the direct three-class detection. However, it is essential to note that many of these additional detections, when passed to the classifier, are classified as false positives (FP) since the ripeness class cannot be reliably determined.

Comparing the image split strategy with the standard full-image approach, it is clear that image splitting significantly reduces the number of FN for the YOLO models—on average, cutting this number in half. However, this improvement comes at the cost of an increase in FP, as the models are more likely to confuse trunks and dried flowers with berries.

In summary, while the image split approach helps YOLO models by reducing missed detections, it also increases the risk of FP. Across all experiments, RT-DETR-L remains the most balanced model, nearly matching the TP and FN rates of all YOLO models with image splitting, but with fewer FP, confirming its robustness and reliability in multi-class *Arbutus* berry detection.

Chapter 6

ArBeH-Harvesting robotic tool

This chapter focuses on the design of the manipulator, beginning with the creation of a simulated Arbutus plantation environment, followed by the control of the manipulator and the development of algorithms for the considered harvesting methods, namely suction, cutting and gripping. The results of the simulations are presented, along with a discussion of different control strategies and movement optimization techniques, always aiming to maximize harvesting efficiency and minimize damage to the plant. This framework enables the evaluation of the manipulator's performance in various scenarios, helps identify limitations, and supports the proposal of improvements toward practical application in real-world environments.

6.1 Simulation Environment

A virtual simulation environment was developed using Webots to test and validate the proposed solution in this project. This virtual environment, shown in Figure 6.1, was designed to replicate the main features of an Arbutus tree plantation with terraces, including gentle ramps that allow transitions between different levels. This configuration is essential for analyzing the manipulator's performance in scenarios resembling real field conditions.



Figure 6.1: Simulation environment of Arbutus tree plantation.

In the developed scenario, an existing Modular-E robot from the TRIBE laboratory was used, with control previously implemented via a joystick. The robot is manually operated, moving parallel to the trees, stopping beside each tree to perform the harvesting task.

Trees with dimensions similar to those of the Arbutus tree were used to ensure a realistic approximation of the agricultural environment. To simulate the Arbutus berries, spheres of different diameters were distributed on the trees, representing the natural variability in fruit size. Additionally, the spheres were colored differently to simulate the various ripening stages of the fruit, allowing for the testing and validation of algorithms for identifying and selecting ripe fruits.

The use of Webots enabled the creation of an environment not only visually close to reality but also with precise configuration of the physical properties of objects, such as mass, friction, and texture, providing a faithful simulation of the manipulator's behavior and the elements of the environment. In this way, it is possible to conduct various tests and adjustments to the robotic system before its application in the field, optimizing future development processes and reducing costs and risks associated with real-world validation. Furthermore, this approach made it possible to evaluate which system configurations are most adaptable and to identify the most efficient harvesting method.

Therefore, constructing this virtual environment was a fundamental step to ensure the robustness and effectiveness of the proposed solution, allowing the identification and correction of possible system limitations in a controlled and repeatable context.

6.2 Manipulator Design

Due to the kinematic and dynamic complexity of manipulators with multiple DoF, developing more sophisticated control algorithms becomes necessary. However, considering the specific challenges of arbutus berry harvesting, such as the irregular positioning of the fruits and the dense surrounding vegetation, a manipulator with greater flexibility and angular reach is required. The design of this manipulator was based on an existing model previously developed in the laboratory, adapted to meet the specific demands of this agricultural application.

The developed manipulator features three degrees of freedom, all of the Revolute-Revolute-Revolute (RRR) type. This configuration was chosen due to its high flexibility and wide angular reach, which are essential attributes for effective operation in complex agricultural environments. The RRR manipulator allows the robotic arm to perform movements in multiple planes and angles, making it easier to access arbutus berries even when they are located in hard-to-reach places or partially hidden by leaves and branches. Each of the two main links of the manipulator has a length of one meter, providing the system with significant reach. Figure 6.2 shows the structure of the manipulator in its resting position, with all three links aligned vertically.

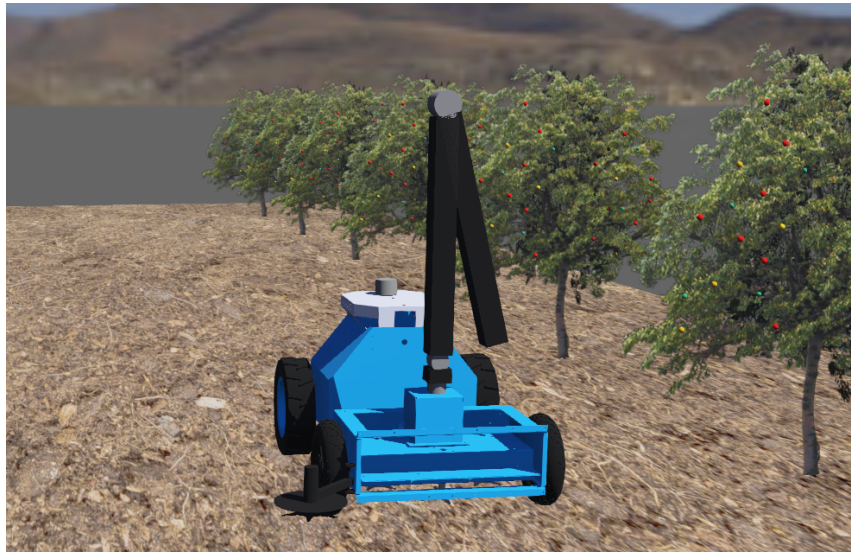


Figure 6.2: Manipulator structure.

To increase the autonomy and efficiency of the system, two cameras were integrated into the manipulator. The first camera is positioned at the manipulator's base and is used to detect arbutus berries on the tree. This camera provides a global view of the workspace, enabling the identification and localization of the fruits to be harvested through computer vision algorithms. The second camera is installed at the end of the manipulator and is used for fine adjustments during the approach to the fruit. This approach allows for the correction of deviations and ensures the precise positioning of the harvesting mechanism relative to the arbutus berry, thereby increasing the success rate of the operation. Additionally, two sensors are mounted at the top of the manipulator to prevent collisions with the surrounding vegetation or other obstacles. These sensors provide real-time feedback about the proximity of objects near the end-effector, enabling the system to make immediate adjustments and avoid unintended contact. This combination of visual and proximity sensing significantly enhances the safety and reliability of the harvesting process, especially in dense and unstructured environments.

Regarding the harvesting mechanism, although a definitive end-effector has not yet been developed, the manipulator was designed to support two distinct harvesting methods, both of which were simulated in the development environment to evaluate the viability and efficiency of different approaches:

- **Suction Method:** This method uses a vacuum device attached to the end-effector, which sucks in the arbutus berry once it has been identified and properly positioned. A cutting mechanism is activated when the fruit is inside the suction tube, allowing for controlled detachment of the berry without causing damage. This method offers advantages in terms of speed and reduces direct physical contact with the fruit; however, it may not always guarantee the integrity of the berry due to the suction process.

- **Cutting and Gripping Method:** In this approach, the end-effector consists of a gripper that holds the arbutus berry and a pair of scissors that cut the stem. After harvesting, the manipulator transports the fruit to a basket, simulating a process closely resembling manual harvesting. Although this solution is slower, it may be more suitable when it is important to ensure the integrity of the Arbutus Berry.

6.2.1 Angle Calculation

The kinematic analysis of the manipulator is essential for determining the position and orientation of the end-effector based on the joint angles (direct kinematics) and, conversely, for calculating the angles required to reach a desired position (inverse kinematics). For this purpose, a model based on the Denavit–Hartenberg (DH) convention was adopted, which facilitates the definition of local reference frames on each link of the manipulator. The kinematic configuration used was inspired by the model presented in [45], and is illustrated in Figure 6.3

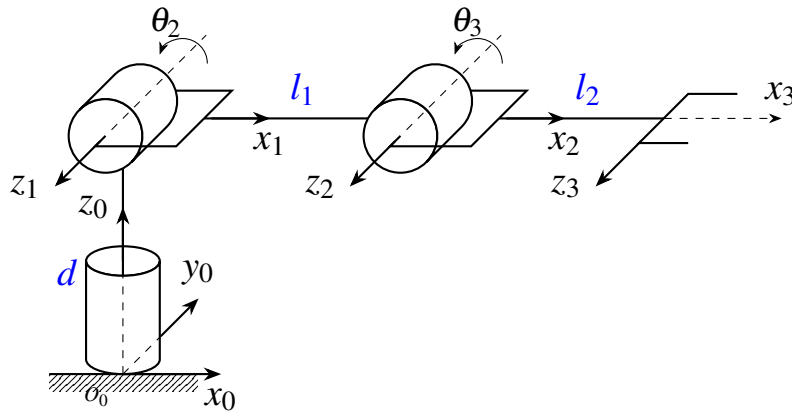


Figure 6.3: Coordinate frames attached to the manipulator.

Direct kinematics consists of calculating the position and orientation of the end-effector from the joint angles. For the proposed manipulator, the DH parameters are organized in Table 6.1.

Table 6.1: Forward Kinematic.

i	a_i	α_i	d_i	θ_i
1	0	90°	d	θ_1
2	l_1	0	0	θ_2
3	l_2	0	0	θ_3

where i is the number of the link and:

- a_i is the distance between the axes Z_i and Z_{i-1} ,
- α_i is the angle between the axes Z_i and Z_{i-1} ,

- d_i is the distance between the axes X_i and X_{i-1} ,
- θ_i is the angle between the axes X_i and X_{i-1} .

Inverse kinematics is the process of determining the joint angles (θ_1 , θ_2 , θ_3) of a desired end effector position (x, y, z). This calculation is more complex due to the non-linear nature of the equations involved. To facilitate the calculation of the joint angles, lateral and top views of the manipulator were created, geometrically illustrating the relationships between the system parameters and assisting in determining the angles from the desired end-effector position. The lateral and top views of the manipulator, which support the calculation of the joint angles, are shown in Figure 6.4a and 6.4b, respectively.

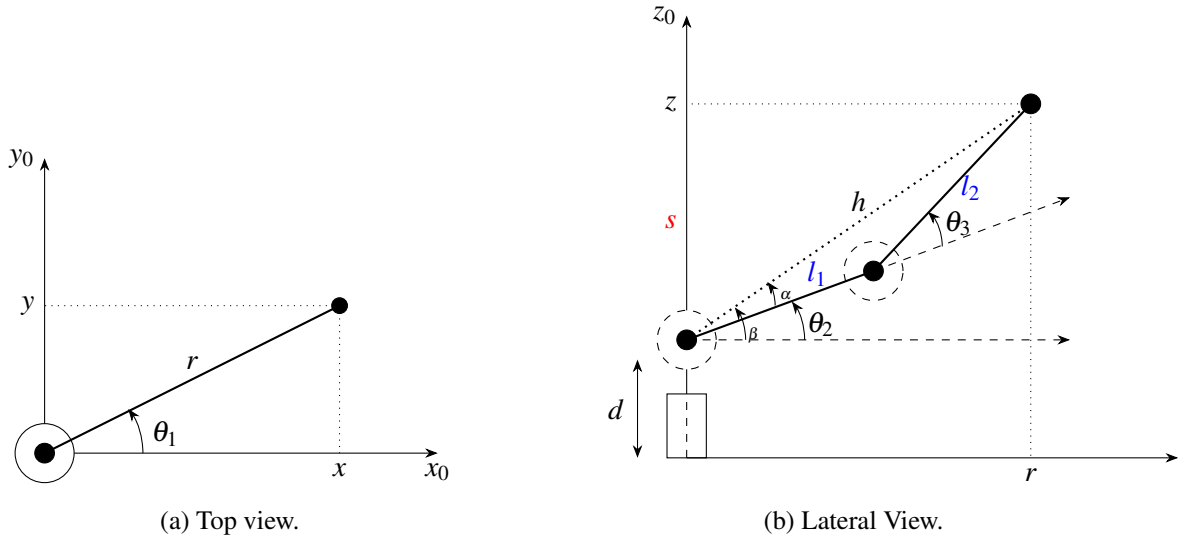


Figure 6.4: Geometric representation of the manipulator for inverse kinematics calculations.

The first angle, θ_1 , determines the orientation of the manipulator in the horizontal (XY) plane, directing the arm toward the position of the fruit. This angle is calculated based on the projection of the end-effector's final position onto this plane. Figure 6.4a presents this top view of the system, which is fundamental for understanding the influence of θ_1 on the manipulator's overall orientation. on the global orientation of the manipulator.

$$\theta_1 = \text{atan2}(y, x) \quad (6.1)$$

After the base rotation has been set, the subsequent calculations are carried out in the plane established by this orientation—a two-dimensional plane that contains the links l_1 and l_2 . Figure 6.4b shows the side view of the manipulator, making it easier to visualize the vertical positioning of the links and the influence of the angles θ_2 and θ_3 .

$$r = \sqrt{x^2 + y^2} \quad (6.2)$$

$$z = s + d \Leftrightarrow s = z - d \quad (6.3)$$

$$h^2 = r^2 + s^2 \quad (6.4)$$

These parameters represent the projection of the horizontal distance from the base to the end point (r), the vertical offset (s), and the total distance between the base and the target point (h).

By applying the law of cosines to the triangle formed by the arms l_1 , l_2 e h , we obtain:

$$h^2 = l_1^2 + l_2^2 - 2l_1l_2\cos(\pi - \theta_3) \quad (6.5)$$

$$\Leftrightarrow -\cos(\theta_3) = \frac{-h^2 + l_1^2 + l_2^2}{2l_1l_2} \quad (6.6)$$

$$\Leftrightarrow \cos(\theta_3) = \frac{x^2 + y^2 + (z - d)^2 + l_1^2 + l_2^2}{2l_1l_2} \quad (6.7)$$

$$\sin(\theta_3)^2 + \cos(\theta_3)^2 = 1 \quad (6.8)$$

$$\Leftrightarrow \sin(\theta_3) = \pm \sqrt{1 - \cos(\theta_3)^2} \quad (6.9)$$

$$\theta_3 = \text{atan2}(\pm \sin(\theta_3), \cos(\theta_3)) \quad (6.10)$$

This angle corresponds to the elbow flexion and defines the relative configuration between the two links of the manipulator. The calculation determines the precise angle required for the arm to reach the distance h , considering the lengths of the links. There are two possible solutions for θ_3 , which correspond to the elbow-up and elbow-down configurations, respectively. In this case, the negative angle was chosen to achieve the desired arm movement because, due to the specific design and construction of the manipulator, it is not capable of operating in the alternative configuration. The manipulator can only function effectively within the negative region of the configuration space.

The difference between the two auxiliary angles determines the angle θ_2 :

$$\theta_2 = \beta - \alpha \quad (6.11)$$

$$\alpha = \text{atan2}(l_2\sin(\theta_3), l_1 + l_2\cos(\theta_3)) \quad (6.12)$$

$$\beta = \text{atan2}(s, r) = (z - d, \sqrt{x^2 + y^2}) \quad (6.13)$$

Here, α represents the internal angle of the triangle formed by the links, while β is the angle between the baseline and the line connecting the base to the target point. The difference between these two angles determines the value of the second joint angle, θ_2 . Since α is directly affected by the value of θ_3 , any change in the elbow angle influences the calculation of θ_2 . This ensures that

the orientation of the first link is appropriately adjusted to position the elbow, taking into account the curvature introduced by the second link.

Applying these equations enables the precise calculation of the joint angles required to position the manipulator's end-effector at any point in three-dimensional space. This inverse kinematics approach is fundamental for real-time control, ensuring the precision and flexibility necessary for selective fruit harvesting in complex agricultural environments. The resting position of the manipulator in the simulator is given by $\theta_1 = 0^\circ$, $\theta_2 = 90^\circ$, and $\theta_3 = -166.2^\circ$, as shown in Figure 6.2.

6.2.2 Camera

For the robotic manipulator to interact with objects detected by the vision system, it is essential to transform the coordinates of detected points in the image plane (u, v) into coordinates in the manipulator base reference frame (x_m, y_m, z_m). This transformation enables precise localization and manipulation of objects within the robot's workspace. The camera used in this system has a resolution of 640×480 pixels and a field of view (FoV) of 1 radian, which is equivalent to approximately 57.3° . The camera is positioned with a translation of 6 cm along the Z-axis and is rotated by 15° around the X-axis.

The relationship between the pixel coordinates (u, v) and the 3D coordinates in the camera reference frame ($x_{cam}, y_{cam}, z_{cam}$) and the pixel coordinates (u, v) is described by the pinhole model, represented by the following matrix equation:

$$\alpha \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} \quad (6.14)$$

where:

- $\alpha = z_{cam} = Z$ is the point's depth relative to the camera,
- c_x and c_y are the coordinates of the optical center in pixels,
- f_x and f_y are the focal lengths in pixels (calculated from FoV and resolution).

$$c_x = \frac{w}{2}, \quad c_y = \frac{h}{2} \quad (6.15)$$

$$f_x = \frac{w}{2 \tan(\frac{FoV}{2})}, \quad f_y = \frac{h}{2 \tan(\frac{FoV}{2})} \quad (6.16)$$

To estimate Z , the method in Figure 6.5 was used, where D is the distance from the manipulator frame of reference to the tree, and the angle θ is the camera tilt relative to the ground plane.

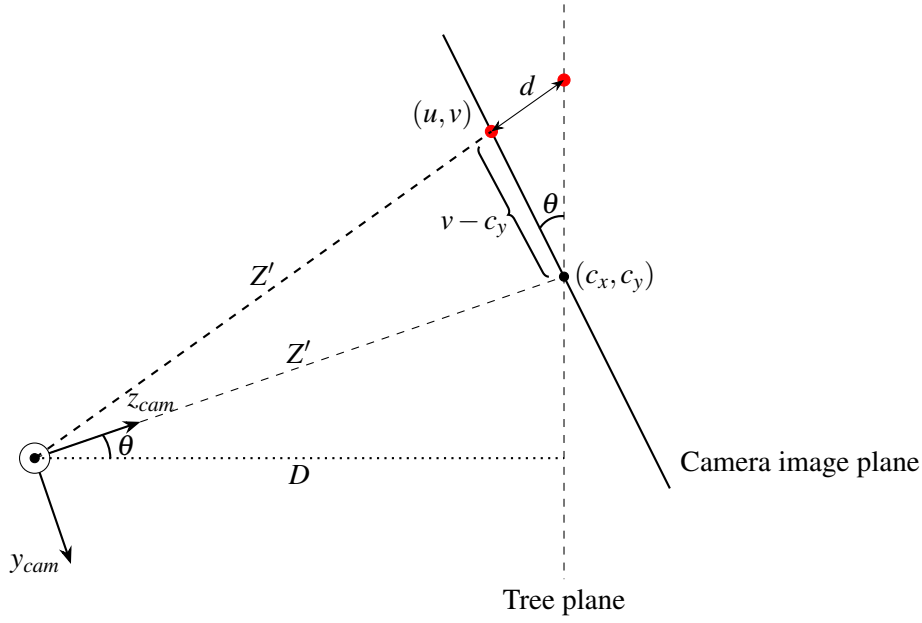


Figure 6.5: Depth estimation geometry using a tilted camera and image projection.

$$Z' = \frac{D}{\cos(\theta)} \quad (6.17)$$

$$d = \tan(\theta) * \left((v - c_y) * \frac{Z'}{f_y} \right) \quad (6.18)$$

$$Z = Z' + d \quad (6.19)$$

By inserting the equation 6.14, it is possible to obtain the camera coordinates from the pixel values:

$$\begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} = Z \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} (u - c_x) \frac{Z}{f_x} \\ (v - c_y) \frac{Z}{f_y} \\ Z \end{bmatrix} \quad (6.20)$$

The transformation of the camera coordinates to the manipulator base reference frame is done using the rotation matrix R and the translation vector t , which define the position and orientation of the camera relative to the manipulator.

$$\begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} = R \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} + t \Leftrightarrow \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = R^T \begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} - t \quad (6.21)$$

$$\begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = R^T \begin{bmatrix} (u - c_x) \frac{Z}{f_x} \\ (v - c_y) \frac{Z}{f_y} \\ Z \end{bmatrix} - t \quad (6.22)$$

In this way, it is possible to obtain the actual three-dimensional coordinates of a point observed in the image, expressed in the reference frame of the manipulator base. This enables the robot to plan and execute actions with precision in the physical space.

6.3 Manipulator Controller

The state machine of the manipulator controller was developed to manage all stages of the autonomous harvesting process, controlling both the movements of the manipulator and the actions of the end effector (Figure 6.6). Although the end effector is not yet implemented in the simulator, the structure of the state machine is already prepared to integrate this component in the future.

This state machine covers the entire process from the initial detection of the fruits to the final storage, ensuring the sequential and coordinated execution of each phase of the harvesting process. As previously described, two distinct approaches were considered for the harvesting process: one based on suction and the other on cutting and gripping. In the state machine, this distinction is visible through the two paths represented: the blue path corresponds to the suction simulation, while the red path refers to the cutting and gripping simulation.

Since the process is being simulated, both approaches were integrated into a single state machine, allowing the user to select the desired mode of operation. If this system were to be applied in a real scenario, it would be necessary to separate the state machine into two distinct controllers, each dedicated to a specific harvesting method. This flexibility ensures that the controller can be easily adapted to different operational requirements and types of tools.

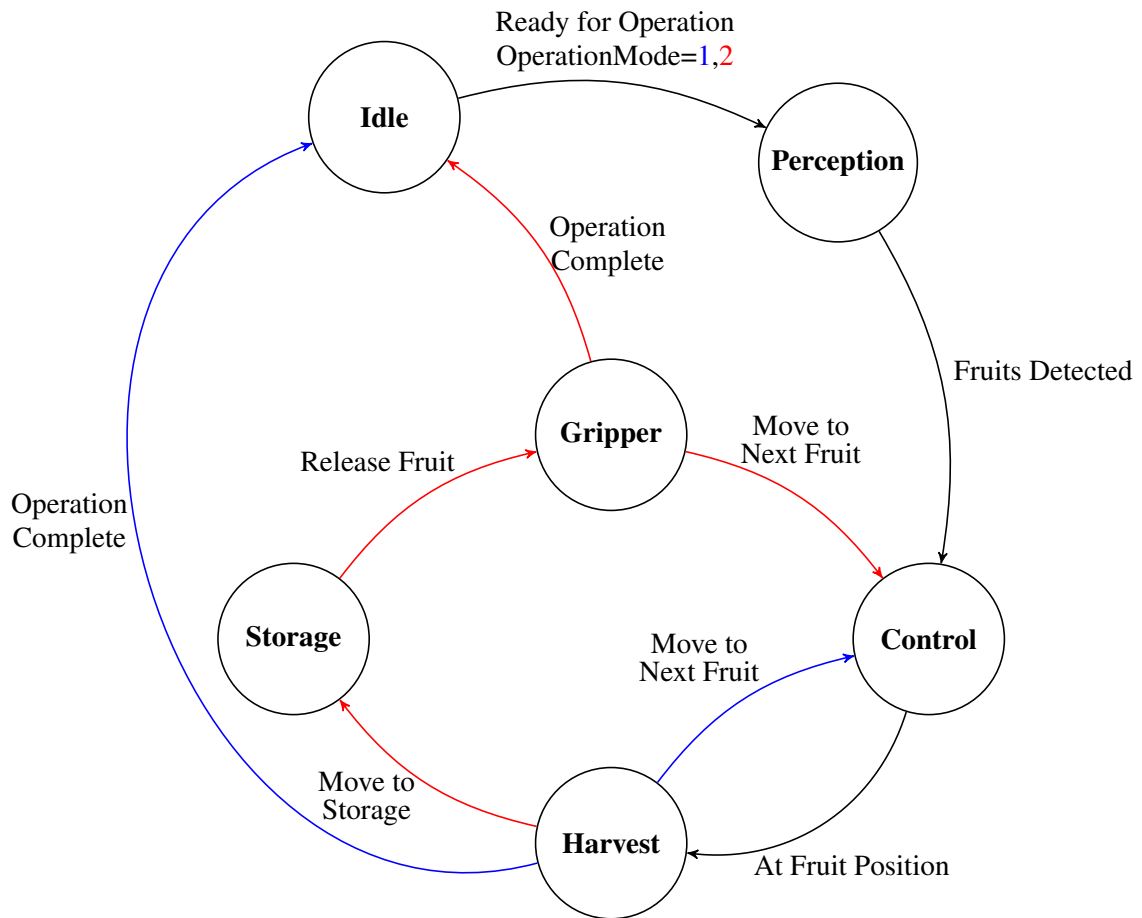


Figure 6.6: Manipulator controller state machine. The blue path illustrates the sequence of states followed during the suction harvesting method, while the red path shows the sequence for the cutting and gripping method.

The Manipulator control state machine is composed of the following states and their respective actions:

- **Idle:** The robot moves parallel to the trees. When it arrives at a tree, it stops. The operation mode for harvesting is then selected.
- **Perception:** The system detects the Arbutus berries and creates a list of the detected fruits.
- **Control:** The manipulator moves to the position of the detected fruit.
- **Harvest:** The end effector harvests the fruit.
- **Store:** The manipulator moves to the storage basket
- **Gripper:** The gripper releases the fruit into the basket.

And by the following transitions:

- **Ready for Operation:** If the robot is correctly positioned at the tree, it sends a command to the manipulator controller indicating readiness for operation and specifies the operation mode.
- **Fruits Detected:** If fruits are detected, the manipulator controller receives a list of the detected points.
- **At Fruit Position:** If the manipulator is in the correct position to harvest the fruit.
- **Move to Storage:** If the gripper has already picked the fruit.
- **Release Fruit:** If the manipulator is at the basket position
- **Move to Next Fruit:** If the manipulator has already harvested the fruit and there are more fruits to pick, it proceeds to the next one.
- **Operation Complete:** If there are no more fruits to harvest.

6.3.1 Perception

In the context of the developed robotic harvesting system, one of the fundamental states in the state machine is the perception state. This state is responsible for the precise localization of arbutus berries, a crucial step to ensure the efficiency and autonomy of the harvesting process. Accurate fruit detection and localization are essential for guiding the manipulator to interact with the target objects, enabling reliable and collision-free picking actions. In this state, a ROS topic is subscribed to, which contains an ordered list of the pixel coordinates of the detected arbutus berries. This list results from a detection and planning process implemented in a Python script (HVP) that operates in real time.

The process for creating this list begins by subscribing to a ROS2 topic associated with the camera image mounted at the manipulator's base. The captured image is processed by a pre-trained YOLO model, which can detect and classify Arbutus berries according to their ripeness stage. This model is one of those discussed in Chapter 5, enabling the identification of multiple fruits within the same image, distinguishing between green, yellow, and red Arbutus berries. After detection and classification, green berries are automatically ignored, as they are not intended to be harvested at this stage of ripeness. Depending on the user-defined configuration, yellow berries can also be excluded from the harvesting list. Only fruits deemed suitable for harvesting are selected, with the centroid of each one calculated and stored in a list for subsequent processing.

To optimize the trajectory of the robotic manipulator, the order in which the fruits are visited must minimize the total path length. For this purpose, a planning algorithm based on the solution of the Traveling Salesman Problem (TSP) is applied. The TSP is a classic problem in graph theory, where the goal is to determine the lowest-cost path that visits all points (in this case, the centroids of the fruits) and returns to the starting point. The algorithm implemented to solve the TSP uses dynamic programming with bitmasking and follows these steps:

- **Calculation of Euclidean Distances:** A distance matrix is generated, where each element represents the direct (Euclidean) distance between two centroids. This is computed using the Pythagorean theorem, allowing efficient lookup of distances during the optimization process.
- **Dynamic Programming with Bitmasking:** The core TSP solution employs dynamic programming combined with bitmasking to represent all possible subsets of visited centroids efficiently. The main function maintains a memoization table that stores the minimum cost for each combination of the current centroid and the set of visited centroids (represented as bitmasks). State transitions are performed by considering all possible next centroids that have not been visited, updating the minimum cost accordingly. An auxiliary matrix records the optimal choices at each step, enabling path reconstruction later.
- **Reconstruction of the Optimal Path** After computing the minimum total cost, the optimal sequence for visiting the centroids is reconstructed using the auxiliary matrix. This results in an ordered list of centroids, defining the robotic manipulator's most efficient harvesting route.

Figure 6.7 illustrates the final result of ordering the centroids of the detected red Arbutus berries, showing the optimal sequence calculated for the path of the robotic manipulator.

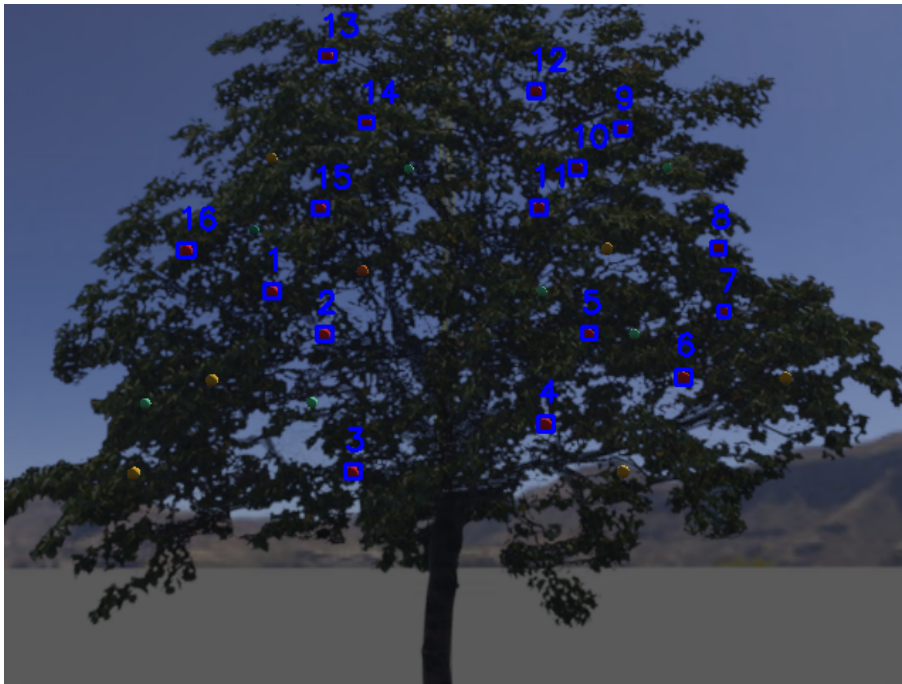


Figure 6.7: Optimal harvesting sequence of detected berries.

The ordered list of centroids is then published on a dedicated ROS2 topic, as a Polygon message. Other system modules can consume this message, notably the robotic manipulator controller, which uses the sequence to harvest the fruits in the most efficient order. This approach is especially

relevant in robotic harvesting systems, as it significantly reduces the time and energy consumption of operations, optimizing the system's overall performance. Integrating computer vision, trajectory planning, and data publishing in ROS2 enables robust, scalable, and adaptable automation to different agricultural scenarios.

6.3.2 Control and Harvesting

In the Control state, the manipulator begins processing the list of detected fruits received from the previous state. The first point in the list is selected, and the transformation of the detected fruit's centroid coordinates from pixels to the manipulator base reference frame (section 6.2.2) is calculated. With these 3D coordinates, the joint angles are calculated through inverse kinematics (section 6.2.1). After obtaining the necessary angles, they are applied to the manipulator's joints, which move to the defined position. Arrival at the position is confirmed by position sensors present in each joint. Once the manipulator reaches the desired configuration, the state machine advances to the next state, Harvest, where the end effector acts.

The manipulator's behavior in the Harvest state is determined by the harvesting method the user selects before the process begins determines the manipulator's behavior in the Harvest state. This selection is made through the operation mode variable, allowing the user to switch between the available harvesting methods flexibly.

- **Suction Method:** After the manipulator reaches the fruit's position, the suction and peduncle cutting simulation is performed, with a 1s wait time to represent the execution of these actions. If more fruits are in the list, the system automatically advances to the next point, repeating the harvesting cycle; otherwise, it returns to the initial Idle state. This method corresponds to the transitions marked in blue in Figure 6.6. The manipulator uses sensors at the end of the arm to ensure operational safety and avoid collisions with the tree. Whenever an obstacle is detected nearby, the system retracts the manipulator arm until the obstacle is no longer identified, thus ensuring efficient operation and protecting both the equipment and the fruits during harvesting.

- **Cutting and Gripping Method**

In this harvesting mode, the manipulator simulates grasping and cutting the fruit at the peduncle, with a 1s wait to represent the execution of these actions. Once the fruit is harvested, the system transitions to the Store state, where the manipulator moves to the storage basket, whose position is fixed. Upon reaching the correct position, it advances to the Gripper state, simulating the release of the fruit into the basket for 0.5s. If there are more fruits to harvest, the system returns to the Control state to repeat the process; otherwise, it transitions back to Idle. This method follows the red transitions in Figure 6.6. To minimize the risk of collisions, the manipulator first adjusts its base joint to align with the direction of the detected fruit before extending towards it. Additionally, proximity sensors are used as needed—if an obstacle is detected, the manipulator will retract until the path is clear, ensuring safe and efficient operation throughout the harvesting cycle.

All the timers mentioned in this description are used to simulate the actuation of the end effector, which is not yet incorporated into this system. The timers serve as placeholders to represent the duration of actions such as grasping, cutting, suction, and releasing the fruit, ensuring that the state machine accurately reflects the sequence and timing of operations that will occur once the real end effector is implemented.

6.4 Results and Discussions

Timed tests were conducted to compare the efficiency of the two Arbutus berry harvesting approaches—suction and cutting with gripping—both with optimal trajectory planning (based on the solution to the TSP) and without any planning (random order). The objective was to determine not only which method is more time-efficient, but also whether planning the harvesting sequence significantly contributes to the overall efficiency of the process.

Twenty-five experiments were conducted for each combination of harvesting methods and execution strategies (with and without trajectory planning). The results are summarized in Table 6.2, which presents the maximum, minimum, and average times (in seconds) required to harvest 16 Arbutus berries distributed throughout the tree.

Table 6.2: Performance of harvesting methods with and without trajectory planning, in seconds.

		With planning (s)	Without planning (s)
Suction	Max	22.98	28.93
	Min	22.02	26.56
	Average	22.58	27.71
Cutting and Gripping	Max	104.48	104.32
	Min	103.46	103.62
	Average	103.94	103.96

The results show that the suction method is considerably faster than the cutting and gripping method, regardless of whether trajectory planning is used. With planning, the average time to harvest the berries using suction was 22.58s, while without planning, this time increased to 27.71s, a difference of approximately 23%. These data demonstrate that planning the harvesting order, by optimizing the manipulator’s trajectory, significantly impacts process efficiency by reducing unnecessary movements between fruits. Without planning, the manipulator moves less efficiently, traveling from point to point in a random sequence, which results in a longer total operation time.

However, for the cutting and gripping method, the difference between the times with and without planning is practically insignificant (103.94s and 103.96s). This is because, in this method, the manipulator must move to the basket position after each harvest to deposit the collected berries, making the order of visiting the fruits irrelevant in terms of total operation time. Therefore, planning the order of fruit visits does not significantly impact the overall efficiency of this method.

Each of the tested methods presents distinct advantages and limitations. The suction method was the most time-efficient, especially with trajectory planning. However, using suction may compromise the fruit's integrity, as the aspiration process can cause damage. This limitation may restrict its application in situations where the quality of the final product is a critical factor. Therefore, further studies in real-world environments are needed to assess the actual impact of suction on fruit preservation.

On the other hand, the cutting and gripping method, although slower, ensures the fruit's integrity by simulating the manual harvesting and careful storage process. The total time for this method could be optimized by improving the location of the storage basket since, in the experiments conducted, the basket was fixed at the rear of the robot. The operation time could be reduced if the basket were positioned closer to the tree. Thus, new configurations for the basket position can be explored to further increase the efficiency of this method.

The experimental results confirm that trajectory planning is essential for harvesting methods in which the manipulator can pick multiple fruits sequentially, such as the suction method. In contrast, for processes that require constant returns to the storage point, like cutting and gripping, the planning of the harvesting order has a negligible impact on the total time. The choice of harvesting method should consider not only time efficiency but also the integrity and quality of the fruit, making it necessary to tailor the solution to the specific context and the producer's requirements.

Videos demonstrating the harvesting using the two implemented methods have been made available. In these videos, it is possible to observe the movement of the manipulator during the harvesting of the detected *Arbutus* berries. The fruit is considered harvested as soon as the manipulator approaches it, which can be seen with the camera at the tip of the manipulator. These videos can be accessed at the following link: https://drive.google.com/drive/Arbutus_Berry_Harvest.

Chapter 7

Conclusion and Future Work

The work developed in this dissertation represents a significant advancement in the automation of Arbutus berry harvesting, as nothing had previously been explicitly designed for this crop, a crop of high ecological and economic value, but which presents unique challenges due to its morphology, dense canopy, and variability in fruit ripeness. The integration of computer vision technologies, deep learning models, and robotic simulation enabled the proposal and validation of a robust architecture for the detection, classification, and selective harvesting of Arbutus berries, demonstrating that it is possible to achieve high levels of precision and efficiency even in complex and dynamic scenarios.

The results demonstrate that the proposed architecture effectively addresses the unique challenges of Arbutus berry harvesting by integrating advanced sensing technologies, robust perception, and adaptive manipulation. Environmental data collection using LiDAR sensors and RGB-D cameras enabled the creation of detailed and annotated datasets that reflect the variability of real-world conditions, including dense canopies, occlusions, and different fruit ripeness stages. The use of state-of-the-art deep learning models—such as RT-DETR-L and the Nano/Tiny versions of the YOLO family—proved highly effective, achieving high precision and recall rates even with small fruits, high density, and significant leaf occlusion. Two approaches were considered: (i) direct detection of berries according to their ripeness stage and (ii) a two-step strategy consisting of berry detection followed by separate classification. The results showed that the two-step approach performed better, particularly in complex field scenarios. This superiority is explained by the fact that the general detection followed by classification approach not only achieves broader detection coverage but also significantly improves the precision of ripeness classification compared to direct three-class detection models. For example, with the two-step approach, the YOLO11n classifier achieved F1-scores above 97% for green and red classes and 96.9% for the yellow class, with precision and recall values consistently above 95%, while the direct three-class detection models showed lower recall and F1-scores—especially for the yellow class—and were more prone to confusion between similar ripeness stages due to color similarities and environmental factors. This separation allows each model to specialize: detection models optimize fruit localization, while classifiers, working with cropped high-quality fruit images, better distinguish subtle differences in

ripeness. Additionally, results show that even when image splitting is applied to YOLO models, the RT-DETR-L model combined with the two-step approach remains the most effective overall, confirming its superiority for both detection and classification of Arbutus berries in challenging conditions

In the manipulation field, developing and simulating a robotic manipulator allowed for comprehensive testing of different harvesting strategies and configurations. The manipulator could reach nearly all the fruits in the canopy, minimizing the risk of plant damage and validating the feasibility of selective harvesting in complex environments. Although no physical end effector was developed at this stage, two picking methods—suction and cut/grip—were designed and simulated to evaluate, in a virtual environment, which approach offers greater efficiency, considering the advantages and limitations of each for Arbutus berry harvesting. Simulation results indicate that, despite its potential limitations, the suction method combined with trajectory planning is the fastest option for fruit collection. However, if maintaining the integrity of the Arbutus berry is a primary objective, the cutting and gripping method proves more suitable, as it minimizes the risk of fruit damage during harvesting. Thus, the choice between methods should balance operational speed with the need to preserve fruit quality, depending on the specific goals of the harvesting operation.

Despite the advances achieved throughout this work, challenges remain that point to several lines of future research and development, which are essential for improving the proposed system. Firstly, it is crucial to increase the size and diversity of the dataset by collecting more data under different lighting conditions, seasons of the year, and varieties of Arbutus, to enhance the generalization and robustness of the detection and classification models. The annotation process for ripeness stages will be improved by employing a histogram-based method to distinguish the maturation state of the berries more accurately. Additionally, other types of classification models will be explored to further improve the system's accuracy and robustness. Regarding image splitting, it is necessary to refine the image merging method used after image splitting to eliminate duplications and false positives resulting from partial detections in the overlapping areas. It is also important to validate the perception system in a real-world environment, ensuring its effectiveness and robustness under real operational conditions. This step is crucial to verify whether the accuracy and robustness of the models are maintained or if adjustments are needed to achieve better results.

In the manipulation field, there is a clear need to develop a dedicated end effector specifically designed to cut and grasp Arbutus berries, considering their morphology and texture to ensure efficient harvesting without damaging the fruit, especially the ripest ones. Likewise, designing an optimized suction device is essential to minimize the risk of damage while enabling more efficient harvesting. In simulation, different configurations and degrees of freedom of the manipulator can still be tested. This will allow the identification of more efficient solutions, optimizing reach, precision, and cycle time during harvesting Arbutus berries. These lines of future research and development are essential for the system's evolution, bringing it closer to practical application and positively impacting sustainable agricultural production.

In summary, this work contributes to advancing agricultural robotics in unconventional crops, demonstrating the potential of selective and sustainable automation in Arbutus berry harvesting. The challenges identified and the proposed directions for future work outline a clear path for the system's evolution, bringing it ever closer to practical application and a positive impact on sustainable agricultural production.

References

- [1] F. Santos, “Automation and robots for a new age of agriculture and forestry,” *INESC TEC Science & Society*, vol. 1, pp. 40–43, 4 Jul. 2022.
- [2] A. S. Aguiar, S. A. Magalhães, F. N. D. Santos, *et al.*, “Grape bunch detection at different growth stages using deep learning quantized models,” *Agronomy*, vol. 11, 9 Sep. 2021, ISSN: 20734395. DOI: [10.3390/agronomy11091890](https://doi.org/10.3390/agronomy11091890).
- [3] J. Pereira, A. Baltazar, I. Pinheiro, D. Silva, and F. N. dos Santos, *Arbutus berry detection dataset*. [Online]. Available: <https://zenodo.org/records/15270858>.
- [4] J. Pereira, A. Baltazar, I. Pinheiro, D. Silva, and F. N. dos Santos, *Multi-stage detection and classification dataset for ripeness stage analysis of arbutus berries*. [Online]. Available: <https://zenodo.org/records/15272462>.
- [5] L. F. Oliveira, A. P. Moreira, and M. F. Silva, *Advances in agriculture robotics: A state-of-the-art review and challenges ahead*, Jun. 2021. DOI: [10.3390/robotics10020052](https://doi.org/10.3390/robotics10020052).
- [6] C. Cheng, J. Fu, H. Su, and L. Ren, *Recent advancements in agriculture robots: Benefits and challenges*, Jan. 2023. DOI: [10.3390/machines11010048](https://doi.org/10.3390/machines11010048).
- [7] C. Fonseca, *Medronheiro: Uma espécie multifuncional*, <https://florestas.pt/conhecer/medronheiro-uma-especie-multifuncional/>, Accessed:12/01/2025, Sep. 2022.
- [8] B. by The Navigator Company, *Medronheiro*, <https://biodiversidade.com.pt/biogaleria/medronheiro/>, Accessed:12/01/2025.
- [9] S. M. Farhan, J. Yin, Z. Chen, and M. S. Memon, *A comprehensive review of lidar applications in crop management for precision agriculture*, Aug. 2024. DOI: [10.3390/s24165409](https://doi.org/10.3390/s24165409).
- [10] Livox, *Mid-360 lidar sensor*, 2024. [Online]. Available: <https://www.livoxtech.com/mid-360>.
- [11] P. Kumar. “What are rgb-d cameras? why rgb-d cameras are preferred in some embedded vision applications?” (May 2025), [Online]. Available: <https://www.e-consystems.com/blog/camera/technology/what-are-rgb-d-cameras-why-rgb-d-cameras-are-preferred-in-some-embedded-vision-applications/>.
- [12] Luxonis, *Oak-d s2*, 2024. [Online]. Available: <https://shop.luxonis.com/products/oak-d-s2?variant=42455432233183>.
- [13] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.
- [14] G. Jocher and J. Qiu, *Ultralytics*, version 11.0.0, 2024. [Online]. Available: <https://github.com/ultralytics/ultralytics>.

- [15] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, “Robot operating system 2: Design, architecture, and uses in the wild,” *Science Robotics*, vol. 7, no. 66, eabm6074, 2022. DOI: [10.1126/scirobotics.abm6074](https://doi.org/10.1126/scirobotics.abm6074). [Online]. Available: <https://www.science.org/doi/abs/10.1126/scirobotics.abm6074>.
- [16] Webots, *Http://www.cyberbotics.com*, C. Ltd., Ed., Open-source Mobile Robot Simulation Software. [Online]. Available: <http://www.cyberbotics.com>.
- [17] V. Tinoco, M. F. Silva, F. N. Santos, *et al.*, “An overview of pruning and harvesting manipulators,” *Industrial Robot*, vol. 49, pp. 688–695, 4 Jun. 2022, ISSN: 0143991X. DOI: [10.1108/IR-07-2021-0139](https://doi.org/10.1108/IR-07-2021-0139).
- [18] K. Zhang, K. Lammers, P. Chu, Z. Li, and R. Lu, “System design and control of an apple harvesting robot,” *Mechatronics*, vol. 79, Nov. 2021, ISSN: 09574158. DOI: [10.1016/j.mechatronics.2021.102644](https://doi.org/10.1016/j.mechatronics.2021.102644).
- [19] V. Tinoco, M. F. Silva, F. N. Santos, S. Magalhaes, and R. Morais, “Design and control architecture of a triple 3 dof scara manipulator for tomato harvesting,” in *2023 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 87–92, ISBN: 9798350301212. DOI: [10.1109/ICARSC58346.2023.10129554](https://doi.org/10.1109/ICARSC58346.2023.10129554).
- [20] Y. Xiong, Y. Ge, L. Grimstad, and P. J. From, “An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation,” *Journal of Field Robotics*, vol. 37, pp. 202–224, 2 Mar. 2020, ISSN: 15564967. DOI: [10.1002/rob.21889](https://doi.org/10.1002/rob.21889).
- [21] 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, pp. 58–71, 1 Mar. 2020, ISSN: 22143173. DOI: [10.1016/j.inpa.2019.05.004](https://doi.org/10.1016/j.inpa.2019.05.004).
- [22] E. Vrochidou, V. N. Tsakalidou, I. Kalathas, T. Gkrimpizis, T. Pachidis, and V. G. Kaburlasos, *An overview of end effectors in agricultural robotic harvesting systems*, Aug. 2022. DOI: [10.3390/agriculture12081240](https://doi.org/10.3390/agriculture12081240).
- [23] J. Davidson and C. Mo, *Conceptual design of an end-effector for an apple harvesting robot*, 2014. [Online]. Available: <https://www.researchgate.net/publication/288840721>.
- [24] Q. Feng, “End-effector technologies,” in *Fundamentals of Agricultural and Field Robotics*, M. Karkee and Q. Zhang, Eds. Cham: Springer International Publishing, 2021, pp. 191–212, ISBN: 978-3-030-70400-1. DOI: [10.1007/978-3-030-70400-1_8](https://doi.org/10.1007/978-3-030-70400-1_8). [Online]. Available: https://doi.org/10.1007/978-3-030-70400-1_8.
- [25] C. Han, J. Lv, C. Dong, *et al.*, *Classification, advanced technologies, and typical applications of end-effector for fruit and vegetable picking robots*, Aug. 2024. DOI: [10.3390/agriculture14081310](https://doi.org/10.3390/agriculture14081310).
- [26] J. Jun, J. Kim, J. Seol, J. Kim, and H. I. Son, “Towards an efficient tomato harvesting robot: 3d perception, manipulation, and end-effector,” *IEEE Access*, vol. 9, pp. 17 631–17 640, 2021, ISSN: 21693536. DOI: [10.1109/ACCESS.2021.3052240](https://doi.org/10.1109/ACCESS.2021.3052240).
- [27] Y. Tang, M. Chen, C. Wang, *et al.*, *Recognition and localization methods for vision-based fruit picking robots: A review*, May 2020. DOI: [10.3389/fpls.2020.00510](https://doi.org/10.3389/fpls.2020.00510).
- [28] V. Tinoco, M. F. Silva, F. N. Santos, R. Morais, and V. Filipe, “Scara self posture recognition using a monocular camera,” *IEEE Access*, vol. 10, pp. 25 883–25 891, 2022, ISSN: 21693536. DOI: [10.1109/ACCESS.2022.3155199](https://doi.org/10.1109/ACCESS.2022.3155199).

- [29] Y. Zhao, L. Gong, Y. Huang, and C. Liu, *A review of key techniques of vision-based control for harvesting robot*, Sep. 2016. DOI: [10.1016/j.compag.2016.06.022](https://doi.org/10.1016/j.compag.2016.06.022).
- [30] F. Xiao, H. Wang, Y. Xu, and R. Zhang, *Fruit detection and recognition based on deep learning for automatic harvesting: An overview and review*, Jun. 2023. DOI: [10.3390/agronomy13061625](https://doi.org/10.3390/agronomy13061625).
- [31] J. U. S. Bargoti, *ICRA2017 : IEEE International Conference on Robotics and Automation : program : May 29-June 3, 2017, Singapore*. IEEE, 2017, ISBN: 9781509046331.
- [32] P. Hidayatullah, N. Syakrani, M. R. Sholahuddin, T. Gelar, and R. Tubagus, “Yolov8 to yolo11: A comprehensive architecture in-depth comparative review a preprint,” Tech. Rep.
- [33] M. NERGİZ, “Enhancing strawberry harvesting efficiency through yolo-v7 object detection assessment,” *Turkish Journal of Science and Technology*, vol. 18, pp. 519–533, 2 Sep. 2023, ISSN: 1308-9080. DOI: [10.55525/tjst.1342555](https://doi.org/10.55525/tjst.1342555).
- [34] A. Kaleem, S. Hussain, M. Aqib, M. J. M. Cheema, S. R. Saleem, and U. Farooq, *Development challenges of fruit-harvesting robotic arms: A critical review*, Dec. 2023. DOI: [10.3390/agriengineering5040136](https://doi.org/10.3390/agriengineering5040136).
- [35] OpenCV, *Cvat: Computer vision annotation tool*. [Online]. Available: <https://github.com/open-cv/cvat>.
- [36] C. F. Gomes, G. Botelho, R. Guilherme, *et al.*, *MEDRONHEIRO-MANUAL DE BOAS PRÁTICAS PARA A CULTURA TÍTULO Medronheiro-Manual de boas práticas para a cultura COOPERATIVA PORTUGUESA DE MEDRONHO CRL*, ISBN: 9789729920561.
- [37] I. Pinheiro, G. Moreira, S. Magalhães, A. Valente, M. Cunha, and F. N. D. Santos, “Deep learning based approach for actinidia flower detection and gender assessment,” *Scientific reports*, vol. 14, p. 24452, 1 Dec. 2024, ISSN: 20452322. DOI: [10.1038/s41598-024-73035-1](https://doi.org/10.1038/s41598-024-73035-1).
- [38] A. Buslaev, A. Parinov, E. Khvedchenya, V. I. Iglovikov, and A. A. Kalinin, “Albumentations: Fast and flexible image augmentations,” Sep. 2018. DOI: [10.3390/info11020125](https://doi.org/10.3390/info11020125). [Online]. Available: <http://arxiv.org/abs/1809.06839> <http://dx.doi.org/10.3390/info11020125>.
- [39] S. A. Magalhães, L. Castro, G. Moreira, *et al.*, “Evaluating the single-shot multibox detector and yolo deep learning models for the detection of tomatoes in a greenhouse,” *Sensors*, vol. 21, 10 May 2021, ISSN: 14248220. DOI: [10.3390/s21103569](https://doi.org/10.3390/s21103569).
- [40] R. Sapkota, Z. Meng, M. Churuvija, X. Du, Z. Ma, and M. Karkee, “Comprehensive performance evaluation of yolov12, yolo11, yolov10, yolov9 and yolov8 on detecting and counting fruitlet in complex orchard environments,” Jul. 2024. [Online]. Available: <http://arxiv.org/abs/2407.12040>.
- [41] Y. Zhao, W. Lv, S. Xu, *et al.*, *Detrs beat yolos on real-time object detection*. [Online]. Available: <https://zhao-yian.github.io/RTDETR..>
- [42] Ultralytics. “Adam Optimizer: Deep Learning | Ultralytics.” (2025), [Online]. Available: <https://www.ultralytics.com/glossary/adam-optimizer>.
- [43] Ultralytics. “Stochastic Gradient Descent (SGD) Explained | Ultralytics.” (2025), [Online]. Available: <https://www.ultralytics.com/glossary/stochastic-gradient-descent-sgd>.
- [44] B. E. Moore and J. J. Corso, “Fiftyone,” *GitHub*. <https://github.com/voxel51/fiftyone>, 2020.

- [45] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control First Edition*. JOHN WILEY & SONS, INC., 1989.