UNIVERSITÀ DEGLI STUDI DELLA BASILICATA

DEPARTMENT OF AGRICULTURAL, FOREST AND ENVIRONMENTAL SCIENCES
DOCTORAL PROGRAMME IN AGRICULTURAL, FOREST AND FOOD SCIENCES
CURRICULUM IN FOOD SCIENCE AND ENGINEERING - FSE

SMART TECHNOLOGIES FOR AGRI-FOOD AUTOMATION: DEVELOPMENT OF A SMART CART FOR PRECISION AGROCHEMICAL APPLICATION AND HARVEST ASSISTANCE

Scientific Disciplinary Sector

"AGRI-04/B – Agricultural Mechanics"

(ex AGR/09 – Agricultural Mechanics)

Chair of the Doctoral Programme:

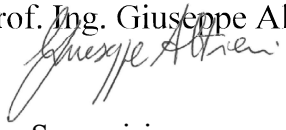
Prof.ssa Teresa Zotta

PhD Student:

Sabina Laveglia

Supervisor

Prof. Ing. Giuseppe Altieri



Co- Supervisior

Dr. Attilio Matera

2025/2026 – Cycle XXXVIII

A mio padre, apicoltore, e ai miei nonni contadini, per i valori che mi hanno trasmesso:da loro ho imparato il valore del tempo, della cura e della perseveranza.

Abstract

The rapid advancement of precision agriculture has fostered the adoption of optical sensing, data-driven modelling, and robotic platforms to improve crop management, quality assessment, and resource efficiency. However, the practical deployment of these technologies in real agricultural environments remains constrained by sensor reliability, limited robustness of predictive models, computational restrictions of embedded systems, and insufficient integration between perception, decision-making, and actuation. This doctoral thesis addresses these challenges through the development of an integrated framework combining multispectral and Vis–NIR sensing, advanced statistical and machine learning models, and a modular robotic platform designed to support agrochemical distribution and harvest-related operations. The research focuses on short-range crop monitoring, early stress and disease detection, non-destructive fruit quality assessment, and system-level robotic integration within a Robot Operating System 2 (ROS 2) architecture. A sensor-less approach for pixel-level alignment of multispectral images is proposed and validated, enabling reliable vegetation index computation under proximal sensing conditions. Physiological monitoring and disease detection are investigated through the integration of biochemical measurements, spectral indices, and chemometric modelling, highlighting the strengths and limitations of Vis–NIR spectroscopy for quantitative and classification tasks. Harvest assistance is addressed via portable spectroscopy, demonstrating accurate non-destructive prediction of fruit quality attributes and effective classification of ripening stages and postharvest defects using reduced spectral information. These sensing and modelling strategies are embedded within a distributed robotic architecture, resulting in the development of a modular smart agricultural cart. The platform integrates heterogeneous sensors, embedded computing, and navigation capabilities, and is validated through proof-of-concept autonomous navigation experiments. The work further identifies hardware limitations of low-power computing platforms and outlines a scalable pathway toward next-generation embedded solutions for real-time, closed-loop agricultural applications. The proposed approach combines sensing, modelling, and robotics to support more efficient, sustainable, and technology-enabled agricultural practices within precision farming contexts.

Keywords: Precision agriculture; Multispectral and Vis–NIR sensing; Chemometrics and machine learning; Non-destructive quality assessment; Agricultural robotics; ROS 2

Acknowledgements

At the conclusion of this professional and academic journey, I would like to express my deepest gratitude to my family for their constant and unconditional support, and to my life partner and his family for their closeness, patience, and encouragement throughout this path. I would also like to thank my colleagues and friends with whom I shared precious moments, making this period not only an experience of professional growth.

I am also grateful to my research team for their support.

A sincere and heartfelt thank you goes in particular to my supervisor, Prof. Giuseppe Altieri, for the trust placed in me, his availability, and his constant scientific and human support.

Finally, I would like to thank myself: for the strength of will, for the perseverance in believing in this journey, and above all for the curiosity that has driven me to continuously seek, study, and explore new stimuli, making this doctoral experience a continuous path of growth.

Sabina

Preface

The activities presented in this thesis were carried out within the framework of the Agritech National Research Center – National Research Center for Agricultural Technologies, with reference to Spoke 2 – Low Impact: Reducing wastage and environmental impact, and in particular to Task 2.3.3 – Precision Agriculture and Smart Technologies for Application of Agrochemicals, which is dedicated to the development and application of precision agriculture technologies and smart solutions for a more efficient, sustainable, and targeted use of agrochemicals.

This work was conducted within the Agritech National Research Center and received funding from the European Union – NextGenerationEU, under the National Recovery and Resilience Plan (PNRR), Mission 4 – Component 2 – Investment 1.4 (D.D. No. 1032 of 17/06/2022, project CN00000022).

This manuscript reflects the views and opinions of the author solely; neither the European Union nor the European Commission can be held responsible for the information and interpretations contained herein.

All materials, results, and code developed within the framework of the thesis activities are made available for scientific dissemination on the "Smart Agri" website, available at: https://sites.google.com/unibas.it/smartagri/home-page.

Table of contents

CHAPTER 1

1. Precision agriculture

1.1. Precision agriculture and the challenge of reducing agrochemicals

In recent decades, the urgent need for environmental sustainability has been at the center of global policy discussions, driven by a growing awareness of the challenges posed by a projected global population of nearly 10 billion by 2050 [1]. Agriculture, as a vital sector for food production, has attracted renewed attention in this context. Meeting the demand for increased agricultural production, necessitated by population growth, has led to a reliance on intensive farming methods. However, a fundamental concern arises: is it possible to increase production without a corresponding increase in environmental emissions, or even achieve a reduction? To address this crucial issue, it is mandatory to determine the factors influencing environmental conservation and food production [2]. The agriculture has emerged as a significant contributor to global Greenhouse Gases (GHG) emissions, accounting for 19-29% of annual emissions globally [3], considering that agricultural land use was responsible for a substantial 37.8% of GHG emissions in the European Union in 2018 [4].

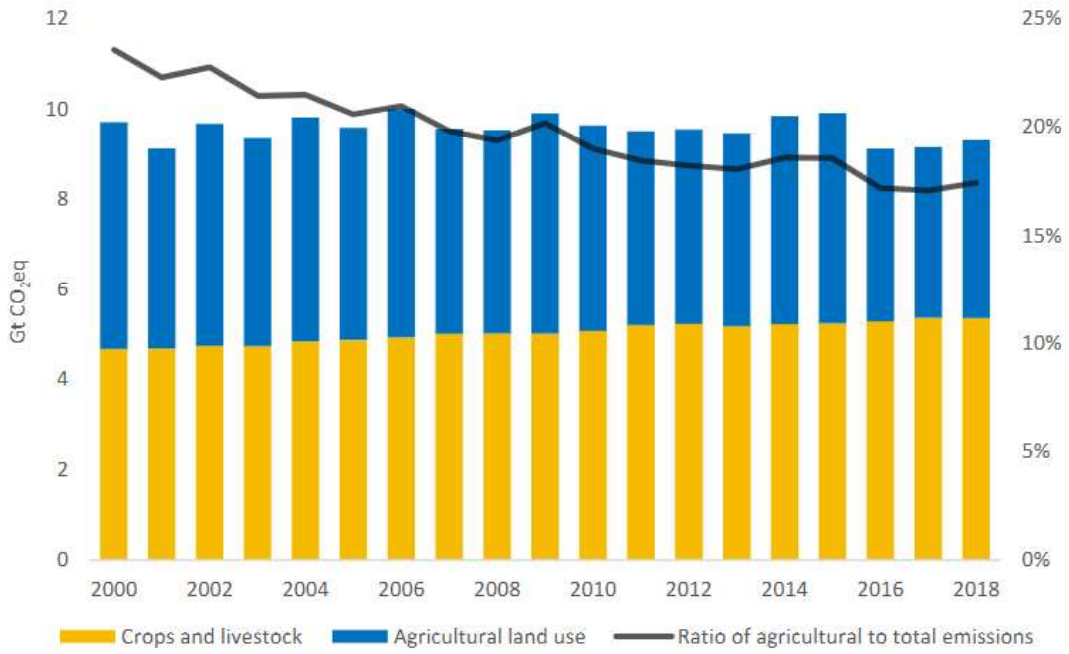


Figure 1.1.Yearly emissions from crops and livestock land use in global GHG emissions from all sectors, 2000–2018. Source: FAOSTAT 2018

The relationship between soil carbon emissions and fertilizer use has been identified. [5], highlighting the challenge of balancing production needs with environmental management. Over-utilization of agricultural chemicals, including pesticides and fertilizers, has negative effects on the environment, soil quality, water resources, and human health. It contributes to water and air pollution, biodiversity loss, and the development of pesticide-resistant pests. [6]. Additionally, the economic burden of agrochemicals can be significant for farmers, impacting their long-term profitability and sustainability. [7].

Although agricultural agrochemicals such as fertilizers and pesticides play a crucial role in helping to ensure stable production yields, their overuse raises concerns about the pollution of vital resources such as water, air, and soil, with far-reaching consequences for future generations. To address these challenges, the United Nations has defined reducing chemical use in agriculture as a key component of the Sustainable Development Goals, calling for innovative approaches to complement traditional agricultural methods. The strategy, which is at the heart of the European Green Deal to make food systems fair, healthy and environmentally friendly, aims to reduce pesticides by 50 percent by 2030 [8]. The successful implementation of these technologies are based on the acquisition of big data, often collected through sensors, drones or satellites, resulting in the concept of "smart agriculture" [9]. Precision agriculture, supported by the collection and analysis of big data, provides farmers with decision support systems (DSS) to optimize production, minimize resource use and improve product quality [10].

1.2. Optical sensing technologies in agriculture: an overview from remote to close-range applications in crop health assessment

The modernization of technological tools available to farmers and agronomists, the combined utilization of these tools and agronomic knowledge has led to the concept known as Precision Agriculture (PA) [7]. Many authors have offered their interpretations of PA, considering environmental, farming, and food security implications. For example, [11] discuss its significance in the resource management, while [12] emphasize its role in enhancing productivity. However, a more comprehensive definition was provided by the International Society of Precision Agriculture (ISPRA), defining it as "*a management strategy that gathers, processes, and analyzes temporal, spatial, and individual data, combining it with other information to support management decisions based on estimated variability, thereby enhancing resource use efficiency, productivity, quality, profitability, and sustainability of agricultural production*".

The use of sensors and their interconnection with agricultural input metering systems plays a crucial role in the sustainability of agricultural ecosystems. In a previously published review, the state of the art of monitoring technologies applied in agriculture was comprehensively analysed, highlighting their impact on field-scale resource management:
Laveglia, S.; Altieri, G.; Genovese, F.; Matera, A.; Di Renzo, G.C. *Advances in Sustainable Crop Management: Integrating Precision Agriculture and Proximal Sensing*. **Agriengineering** 2024, **6**(3), 177. https://doi.org/10.3390/agriengineering6030177.

The following section provides a concise and selective overview of the most relevant concepts required to introduce the present study. In particular, attention is given to optoelectronic proximal sensing systems and their application in crop health monitoring, with emphasis on map-based and sensor-based approaches.

1.2.1. Fundamentals of plant stress sensing

Optical sensing is a technology that uses light to measure and analyze the properties of various objects and materials. It has emerged as a powerful tool in agriculture and farming for monitoring plant growth and health, as well as optimizing the use of resources such as water, fertilizer, and pesticides [13].

The reflectance data collected by these devices can be represented using images acquired by imaging techniques or spectral graphs produced using spectroscopic methods. The integration of advanced imaging and sensing technologies has revolutionized the

assessment and management of both abiotic and biotic stresses in crops. These tools provide quick, accurate, and non-invasive methods to monitor plant health, optimize nutrient and pesticide management, and mitigate stress impacts. Optoelectronic sensing records reflected and emitted plant radiation in visible and near infrared (VSWIR, 400–2500 nm) and thermal infrared (TIR, 8–14 μm) wavelengths. The information associated with each region in the electromagnetic spectrum can be helpful for a particular purpose related to field crops. Leaf optical responses to a wide range of biotic and abiotic stresses have been widely researched [14]. These include heat stress [8-9], heavy metal toxicity [10], exposure to ultraviolet radiation [11], water status [15–17], insect pest attack [18], herbicide treatment [19], salinity effects [20,21] and extremes in nutrient availability [22,23]. In many studies, the spectral wavebands investigated as predictors of plant health status across species range from 400 – 2500 nm [24]. The logic behind these correlations is that unfavorable growing conditions result in morphological, physiological and/or biochemical changes that impact on the manner with which plants interact with light. Reflectance characteristics in the 400 – 700 nm range are primarily influenced by the cellular level of colored pigments, like chlorophyll, anthocyanins and carotenoids [25,26], in the 700 – 1400 nm range by cell structure and in the 1400 – 2000 nm range by the water content in the tissues [14]]. Leaf reflectance patterns have been employed to measure leaf pigment content.

Leaf spectral reflectance provides a vast data resource for assessing plant health based on the impact of biotic and abiotic stresses on leaf biochemistry and anatomy which in turn produces distinct changes in leaf optical properties. Key regions of a reflectance spectrum are:

1. Blue region (400 – 499 nm) which is strongly influenced by absorption of chlorophylls and carotenoids.
2. Blue-green edge (500 – 549 nm) leading to the green peak at 550 nm.
3. Red edge (650 – 699 nm) associated with strong chlorophyll absorption.

Reflectance patterns are influenced by leaf surface features, internal architecture, and biochemical composition. Figure 1.2 shows a schematic representation of the key anatomical structures in relation to their mode of interaction with light. The absorption spectrum of intact chlorophyll molecules shows two dominant bands in the red (Q band) and blue (Soret band) regions and an absorption minimum around 550 nm, giving rise to the perception of a green color. The presence of a methyl group in chlorophyll a instead of an aldehyde group in chlorophyll b at the C-7 position accounts for differences in the absorption wavelengths

of the Q (669 nm vs 644 nm) and Soret (432 nm vs 455 nm) bands. Because of the central function of these pigments in photosynthesis, chlorophyll content is generally regarded as a good indicator of plant physiological health. Many nutrient deficiencies result in a decrease in chlorophyll content, a concomitant increase in reflectance in the visible (400 – 700 nm) and infrared (700-1100 nm) ranges, and a blue shift in the red edge inflection point. Visually, chlorotic changes are perceived as yellowing of leaves [27].

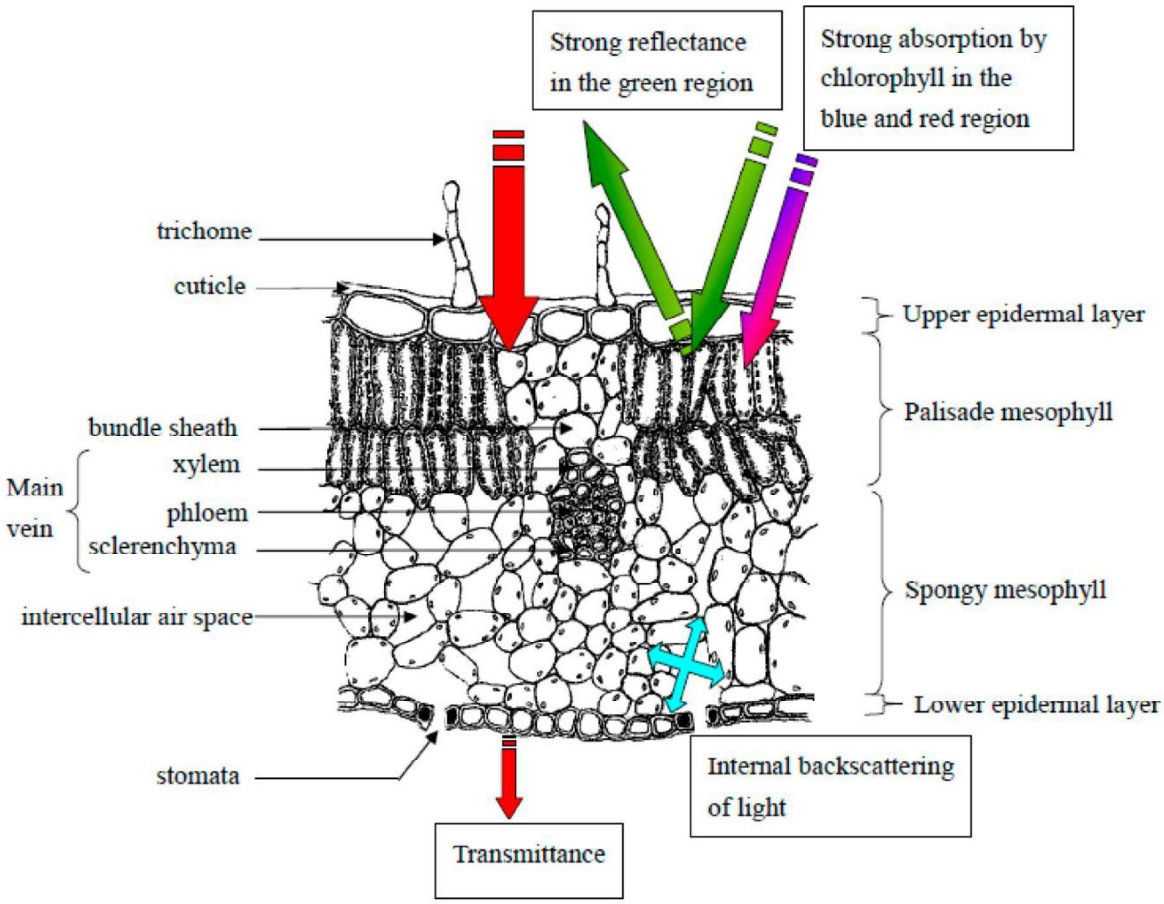


Figure 1.2. Basic light interactions with leaf layers are annotated. Reprinted from Liew et al 2008[27].

1.2.2. Optoelectronic sensor technologies in precision agriculture

Advancements in RGB, Multispectral, and thermal imaging technologies, along with the development of snapshot sensors, have improved real-time disease management capabilities. These innovations offer faster image capturing, though sometimes at the cost of spatial resolution.

Two key concepts underpin image-based analyses:

1.2.3. Spatial Resolution

Spatial resolution refers to the level of detail captured in an image, commonly described by the size of its pixels or grid cells. Higher spatial resolution means that each pixel represents a smaller area on the ground, allowing finer details to be detected.

Conversely, lower spatial resolution results in larger pixel sizes, reducing the amount of visible details.

1.2.4. Spectral Resolution

Spectral resolution describes the number and width of the wavelength bands that a sensor can capture. Imaging systems with high spectral resolution can detect subtle differences in reflected or emitted energy across many narrow bands, enabling detailed discrimination of plant traits, stress responses, and disease symptoms. RGB sensors have low spectral resolution (three broad bands), while multispectral and hyperspectral sensors offer progressively higher spectral detail.

Multispectral sensor technologies can combine spectroscopy and imaging to capture a detailed spectrum of light reflected from an object or scene, providing information on the composition and properties of the materials in the image [28]. It involves collecting and processing data from a large number of narrow, contiguous spectral bands across the electromagnetic spectrum, which can reveal unique characteristics of different materials and help distinguish between them [28]. Researchers use three main spectral scanning technologies to obtain spatial and spectral information: point scanning, line scanning, and band sequential scanning [29]. Spectral scanning combines the dispersive spectrometer with raster scanning. As depicted in Figure 1. 3, point scanning captures one spectral data point at a time, while line scanning captures a slit of spatial information [30]. However, these two scanning approaches require a longer period of time to collect the data cubes [29]. Similar to the point and line scan approaches, band sequential scanning can acquire a high-resolution 2D image one wavelength at a time. While scanning systems allow the acquisition of continuous spectra, dispersive scanning spectrometers suffer from high optical losses at the entrance slit, resulting in long acquisition times. Alternatively, filter-based spectral imaging systems employ a tunable spectral filter placed in front of a monochrome imaging camera, sequentially acquiring images at different wavelengths.

The approaches are straightforward and can obtain both the spatial and spectral information of a scene with one single exposure, which makes them superior to the former methods, as no scanning is required. However, there are three major disadvantages to snapshot technology: (1) the induced image blurring due to motion artifacts; (2) low spectral resolution; and (3) a limited number of wavelengths. For this reason, scanning devices are the most widely used instruments.

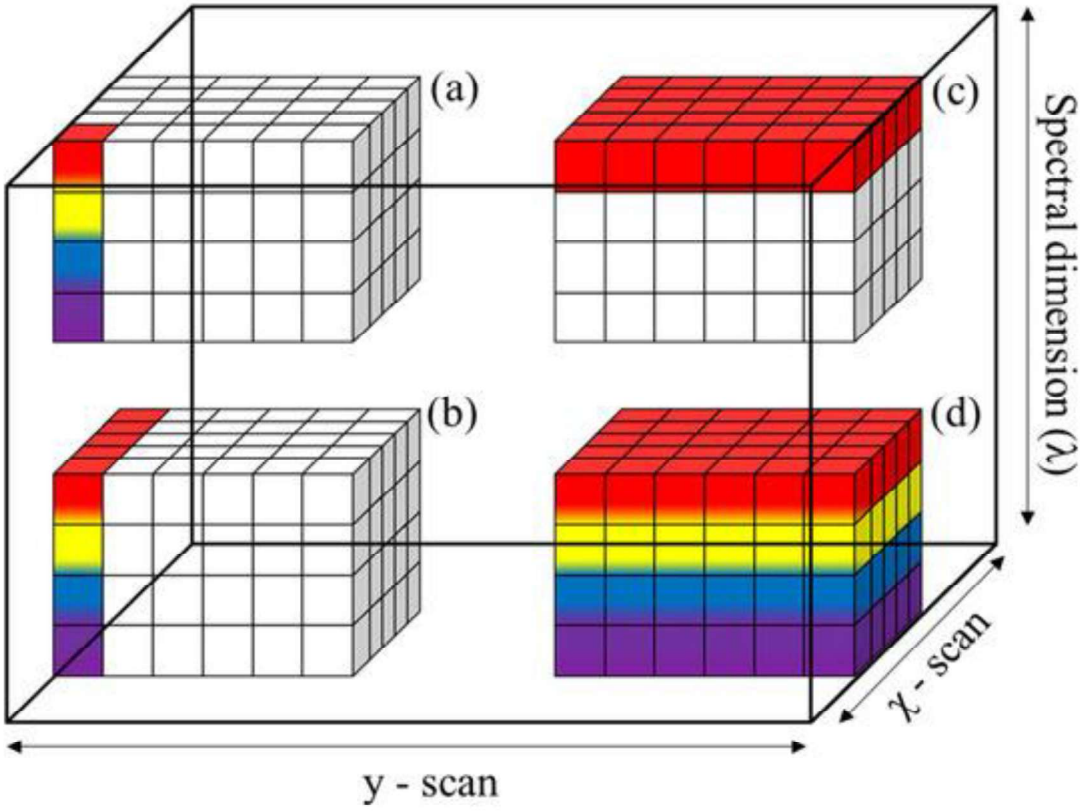


Figure 1.3. Spectral imaging technologies used to obtain images of spatial and spectral information; (a) point scanning; (b) line scanning; (c) band sequential scanning; (d) snapshot. Reprinted by Omia E., et al. 2023[28]

Close-range hyperspectral measurements have effectively identified leaf diseases, while spectral indices and feature selection methods improve classification accuracy. Hyperspectral and multispectral imaging combined with automated disease detection sensors facilitate real-time monitoring and precise management responses.[31].

Other areas of the spectrum are still being explored in terms of their capability for plant stress detection. Due to the sensors' ability to detect a wide range of wavelengths in the electromagnetic spectrum, many possibilities remain for evaluating new combinations of wavelengths for plant stress detection. In contrast to conventional spectral signatures, combinations of reflectance data from two or more spectral bands are used to form the so-called vegetation indices (VI) [32], and have been used extensively for plant stress monitoring [15,33,34]]. In addition to VIs, hyperspectral data can be used to develop spectral disease indices (SDIs) to discriminate between specific plant diseases. [35,36]. Notable vegetation indices include the normalized difference vegetation index (NDVI) [37], water index (WI) [38], and photochemical reflectance index (PRI) [39]. The vast amount of spectral data that is collected using hyperspectral imaging provides great potential in developing new VIs and SDIs for the detection of highly specific plant stresses.

Optoelectronic sensors, mounted on satellite, airborne, or ground-based platforms, are the main source of data in agriculture [24]. Both remote and proximal sensor approaches have been validated in agricultural applications and used as a solution for sustainable crop management. However, they differ in terms of the timeliness with which the collected data by these platforms can be used in field applications, i.e., to rationalize agricultural inputs [40]. This has led to two approaches: the map-based approach and the sensor-based approach.

1.2.5. Map-based approach

One of the most commonly employed approaches in agriculture is to consider the spatial variability of nitrogen (N) within the field in the management strategies, which is critical to optimize the amount of N fertilizer applied [41]. The effectiveness of fertilizer application is enabled by the delimitation of the field into subregions, thereby resulting in homogeneous areas, also known as management zones (MZs), which have low variability within them. [42,43]. MZs are commonly delineated using information on soil [42], yield [44], electrical conductivity [45], farmer knowledge [46], proximal/remote sensing [47], or a combination of them [48]

This map-based approach utilizes essential tools for analyzing spatio-temporal variations within large-scale fields more directly and efficiently, including georeferencing techniques [48–50].

Geographic information systems (GIS), remote sensing (RS), and geographic positioning systems (GPS), have played a key role in the 21st century [51], the increase in their use in agriculture has been due to new platforms and sensors with improved spatial, temporal, and spectral capabilities [52].

Among the data sources, satellite remote sensing stands out for providing high-resolution and cost-effective spatio-temporal information that can accurately delineate MZs [51]

The use of high-resolution satellite images and digital spatial data has made it possible to yield monitoring [49,53], pest management [54], as well as, a map-based approach to variable rate application of different agricultural inputs, such as water [45], nutrients [55,56] and pesticides [57]. These successes are due to the current satellite platforms having significantly improved their spatial and spectral resolution, making it possible to acquire spatial data daily and open source. [58]. According to [51]. estimation of spatial patterns in crop biomass or yield requires better spatial and spectral resolution (1–3 m) than variable rate application of fertiliser (5–10 m). Meanwhile, variable rate spraying of herbicides for spot weed control requires better spatial and spectral resolution (0.5–1 m) than variable rate

irrigation (5–10 m). Several authors have investigated the aspect related to the comparison between different platforms to discriminate localized conditions of inhomogeneity in the field, determined by abiotic or biotic stresses (Table.1.1).

Table 1.1. Satellite and UAVs (unmanned aerial vehicles) sensing applications in precision agriculture: the impact of spatial resolution and sensors mounted abroad

Crop	**Aim**	**Platform (spatial resolution or distnace from the target) Satellite**	**Refere nce**
Cotton (*Gossypium* spp .)	Nitrogen VRT fertilization	• RapidEye (5m)	[56]
Soil	Variable Rate Irrigation based on soil properties	• Sentinel-2 (10 m)	[45]
Winter wheat	Compare RS and PS for site-specific crop management	• Sentinel-2 satellite (10 m); • PS: Fritzmeier ISARIA (n.p)	[47]
Wheat	Disease detection pathogens, powdery mildew (Blumeria graminis) and leaf rust (Puccinia recondita).	• QuickBird Satellite (2.4 m); • Airborne (4 m); • PS: ASD FieldSpec Pro (Analytical Spectral Devices, Boulder, CO, USA)	[59]
Pigeonpea (Cajanus cajan) plants	Disease detection (Fusarium wilt)	• ASI-PRISMA, DESIS (DLR, Germany) and EnMAP (DLR, Germany) HyS satellite; • Sentinel-2 MSI satellite	[60]
Onion	Comparing data acquired by fixed-wing UAV satellite to crop monitoring.	• Parrot Disco-Pro AG fixed-wing UAV (5 m); • Satellite Images: Sentinel-2 (10 m) and PlanetScope (3.7 m)	[61]
Pea and strawberry	Weed detection	DJI Spark (Multirotor) (2 m) | 0.3 cm/px **RGB**: CMOS sensor (3968 × 2976 pixels)	[62]
Lettuce	Weed detection	Multi-rotor DJI Mavic Pro (2 m) | 0.22 cm/px **MSI:** SEQUOIA (Parrot Drone SAS, Paris, France): Green: 550 nm ± 40 nm Red: 660 nm± 40 nm Red Edge: 735 ± 10 nm Near-Infrared (NIR): 790 nm ±: 20 nm.	[63]
Olive tree	Pest detection (Xylella fastidiosa subsp. pauca (Xfp))	Multi-rotor DJI Mavic Pro (70 m) | 6,6 cm/px; **MSI:** SEQUOIA (Parrot Drone SAS, Paris, France): Green: 550 nm ± 40 nm Red: 660 nm± 40 nm Red Edge: 735 ± 10 nm Near-Infrared (NIR): 790 nm ± 20 nm.	[64]
Wheat (cv. 'Mingxian 169'	Disease detection (Yellow rust)	Six-rotor electric UAV system (DJI Innovations, Shenzhen, China) (30 m) | 1.2 cm/px **HYS:** UHD 185 (Cubert GmbH, Ulm, Baden-Württemberg, Germany): 450–950 nm ±4 nm.	[65]
Tomato	Diseases detection (Tomato Yellow Leaf Cur) - TYLC; Target Spot ((Corynespora cassiicola) – TS; Bacterial Spot (Xanthomonas perforans) - BS)	(Matrice 600 Pro Hexacopter, DJI, Shenzhen, China) (30 m) | 1.03 cm/px **HYS:** Pika L 2.4 hyperspectral camera (Resonon, Bozeman, MT, USA): 380 to 1020 nm.	[66]
Vineyard	Spraying	Hexacopter (model: DroneHEXA, Dronetools SL, Sevilla, Spain) (95 m) **MSI:** MicaSense RedEdge: Red:668 nm± 5 nm,Green: 560 nm ± 10 nm, Blue: 475 nm ± 10 nm, RedEdge: 717 nm ± 5 nm, Near Infrared (NIR): 840 nm ± 20 nm;	[67,68]

Despite the images provided by the very high satellite resolution [60] and Unmanned Aerial Vehicles (UAVs) have proved to be useful in guiding localized agronomic operations, such as fertilization and agrochemical application.[60,61]. Satellite imagery may not always provide the necessary information due to limitations in spatial resolution and monitoring frequency; weather conditions are still one of the main obstacles to image acquisition. [45,47,55]. Unlike satellites, UAVs allow for a more precise selection of the required degree

of spatial resolution. The very high spatial resolution provided by UAVs can be an efficient alternative for weed control. [69] and [70] Conducted weed mapping at different altitudes (30, 60, and 100 meters) to determine the optimal resolution for monitoring (30 meters). At this height, [69] proposed a classification algorithm for characterizing pixels on three levels (such as crop, soil, and weeds), achieving an accuracy of 75-87% [69]. Similarly,[71] Proposed a nitrogen management strategy for wheat (*Triticum aestivum*) based on 4 years of investigation, aiming to optimize fertilization rates in relation to the N nutritional index (NNIOA). The NNIOA, calculated with data acquired via UAV, efficiently regulated the state of nitrogen deficiency, optimal levels, and excess through increases, fine-tuning, and decreases in fertilizer application by 54.17%, 0.67%, and 18.18%, respectively. The integration of map-based technologies in PA offers substantial benefits in yield prediction, disease detection, and resource management. [67,72]. These methods enhance the precision and efficiency of agricultural practices, promoting sustainable farming. [73,74]. [75] reviewed the actual progress in crop disease detection, with an emphasis on machine learning and deep learning techniques using UAV-based remote sensing. Although some of the research lines are increasingly moving in this direction, another step on image processing is needed for multi-sensor image registration, also saving processing time for real-time applications [76].

1.2.6. Sensor-based approach

Despite UAVs being able to capture high-resolution data quickly and being flexible in terms of height and flight times, allowing for an intra-daily monitoring application [77], map-based systems are not suitable for sudden changes in ground conditions attributed to meteorological circumstances that may occur between field variability mapping and subsequent intervention [40][78].

On the contrary, scouting systems based on proximal sensors are situated in proximity to the target and can range from being in physical contact with the object to being a few meters away, potentially resolving the issues of the aforementioned methods [36,79].

Its advantages include high-resolution imagery, independence from external factors like cloud cover, suitability for small fields, and ease of application, such as mounting the sensor on a tractor. Proximal sensing allows farmers to obtain detailed information about specific areas, like the health of individual plants or the presence of weeds, and provides real-time or high-frequency data, enabling quick monitoring and action [80].

Current sensing techniques used in remote sensing have been inspired by proximal sensors; specifically, researchers have made an effort to spatialize data on a large scale from

the knowledge obtained between the interaction of single plants using point measurements [90]. As a result of the growing demand for real-time, large-scale detection, of plant diseases, which is becoming increasingly important in PA [81], several reviews have considered spectroscopic techniques in the visible and infrared spectrum, such as ViS-NIR spectroscopy [82], fluorescence spectroscopy, and imaging techniques for large-scale real-time disease monitoring of plants [35,83,84].

Among the various technologies discussed, the evaluation of electromagnetic radiation interactions with plant tissue currently represents the most promising technique [36].

1.2.7. Abiotic leaf assessment

Stress detection of nutritional deficiencies is similar to disease recognition as the goal is to identify visual signs characterizing the disorder of interest [85].

Nitrogen is the primary nutrient supplied to most crops. Unbalanced fertilizer application can impact near-surface and deep aquifers. Both chlorophyll meters and canopy reflectance sensors are used in close-range of the target, following the principles of proximal sensing. They can guide field-specific mid-season fertilizer N application in different field crops [51]. Active Canopy Sensors (ACS) are much more accurate in measurements. [86] evaluated the effects of time of day and nitrogen (N) addition on measurements made with two chlorophyll meters (SPAD-502 and MC-100) and two active canopy reflectance sensors (GreenSeeker handheld and Crop Circle ACS-470) in a sweet pepper crop grown in a greenhouse. Measurements from the GreenSeeker and Crop Circle ACS-470, specifically the Normalized Difference Vegetation Index (NDVI) and Green Normalized Difference Vegetation Index (GNDVI), were unaffected by the time of day across all N treatments. Compared to the chlorophyll meter, these sensors provide reliable measurements at any time of the day, demonstrating their robustness in variable lighting conditions.

The Crop Circle ACS-470 [87] and the GreenSeeker (Trimble Navigation Limited, Sunnyvale, California, USA), developed in 2004 and 2001, respectively [51], have been widely used in the literature for estimating foliar nitrogen content. Several journals discuss the use of the canopy spectroradiometer in precision nitrogen applications [13,85,88]. However, these two are the more common commercial canopy reflectance sensors used in field applications.

The GreenSeeker (GS) sensor (Trimble Navigation Limited, Sunnyvale, California, USA) has two fixed wavebands (red and near infrared (NIR)) and provides two vegetation indices (VI): normalized difference vegetation index (NDVI) and ratio vegetation index (RVI). The Crop Circle ACS-470 is a multispectral sensor, which operates in six bands (blue: 0.43-0.47

μm; green: 0.53-0.57 μm; red: 0.63-0.67 μm, 0.66-0.68 μm; red-edge: 0.72-0.74 μm; NIR: > 0.76 μm) able to perform most vegetation indices for estimating nitrogen status in plants [89].

Consequently, proximal sensing characteristics are believed to be more suitable for predicting yields and managing nitrogen at the field level [90–92]. Many applications have been investigated, the use of crop reflectance indices to map water stress in greenhouse-grown bell pepper[15], salt stress [93], and water deficit [22]. Other applications have evaluated on effects of water and nitrogen stress in tomatoes (*Solanum lycopersicum L.*) [22], lettuce yields and quality [94], to characterize plant responses to multiple stresses [94], and to discriminate them [95].

1.2.8. Biotic leaf assessment

Changes in VIS/IR reflectance spectra due to plant diseases and pathogens can also be influenced by alterations and changes in chemical composition within the affected tissue. These changes can be seen through the appearance of typical fungal structures or as a result of toxins production [96]. For example, aphid infestation has a significant impact on cotton leaves, causing reductions in chlorophyll content and water levels, resulting in a significant decrease in both the visible (350-700 nm) and near-infrared (700-1300 nm) ranges [97]. These spectral reflectance bands have proven useful in detecting aphid infestation, particularly around 470 and 670 nm, where the chlorophyll a/b ratio showed a substantial decrease, and in the NIR, causing an increase in leaf transmittance due to the disruption of cellular structures, leading to a decrease in internal scattering as a result of damage to leaf tissues. The red edge and NIR (755 and 1100 nm) offer insights into plant condition, potentially related to factors such as chlorophyll content, water levels, leaf area index, seasonal variation, and canopy biomass [98]. Similarly, the root infection of *Fusarium spp. (Fv)* in soybeans shows visual symptom effects on both root and surface biomass development, affecting yield. When analyzing the average spectrum of asymptomatic plants in infected plots, higher relative reflectance values were observed in the chlorophyll absorption band around 550 nm and the slope between 550 and 670 nm. This is attributed in part to the reduced chlorophyll content due to root infection and the potential presence of phytotoxins in the canopy [98].

Other applications have shown the proficiency of the reflectance-sensor approach to assess leaf minor damage in tomatoes [99]. aphid pest detection in cotton [97], cancer-infected leaves and *Huanglongbing (HLB in citrus* [100], *Mosaic virus* in cucumber [101], multiple

diseases (fungal pathogens *Cercospora beticola, Erysiphe betae,* and *Uromyces betae,* causing Cercospora leaf spot, powdery mildew, and rust) in sugar beet [102].

The analysis of leaf spectral reflectance data has been extensively explored by researchers, showing its strong predictive capabilities for disease detection. Comprehensive reviews of this field is provided by [35,36,84]. The potential for early prediction of disease, even in latent stages, has been a focus of research and has been confirmed. Although the spectral reflectance of leaves shows promising potential for disease detection, evaluating its robustness in the presence of more than one disease remains an area of interest for future studies [102]. Few studies focus on scenarios where more than one disease coexists, although some work [101,103–106] has suggested clear and well-delineated courses of action, highlighting the need to improve monitoring techniques to accurately discriminate between diseases.

These reviews demonstrate the potential for early prediction of disease, even in latent stages. Although the spectral reflectance of leaves shows promising potential for disease detection, evaluating its robustness in the presence of more than one disease remains an area of interest for future studies [102]. Few studies focus on scenarios where more than one disease coexists, although some work [101,104–106]; [103] has suggested clear and well-delineated courses of action, highlighting the need to improve monitoring techniques to accurately discriminate between diseases.

Recently, several authors have investigated the potential of spectroscopic techniques for disease analysis and discrimination; portable spectral sensor [107] and leaf clip device [96] are the most suitable choices in this domain. Overall, the literature consistently shows that VIS–NIR spectral alterations caused by pathogens reflect underlying physiological and biochemical disturbances, enabling early identification of symptoms, even during latent stages. However, the robustness of spectral approaches in more complex scenarios, such as the simultaneous presence of multiple diseases, remains limited and represents an important area for future research.

1.2.9. Comparison and key findings on the choice of sensor technologies

The use of RGB sensors mounted on UAVs has proven to be a flexible and low-cost solution for health and physiological assessments [16,23,108].

Digital photos have been applied to obtain crop canopy cover, and a new method for nitrogen nutrition index (NNI) estimation was developed using canopy cover instead of crop dry matter. These optical instruments allow rapid acquisition of crop growth information through

proximal sensing; however, they require extensive and professional on-site measurements, which limits large-scale applicability [109].

Changes in the concentration of carotenoids can also be related to color image parameters: carotenoids mainly absorb blue and green light, whereas chlorophylls primarily absorb blue and red light. Stress or senescence causes the degradation of both pigments, although carotenoids degrade more slowly [25,110]. Chlorophyll content can be associated with RGB, HSB, and CIELab (Lab*) color models[111]. Digital image processing is cost-effective, easy to operate, and portable, making it highly suitable for practical field applications. However, water content, nutrient levels, and other physiological variables can influence growth and must be considered in dynamic analysis to improve diagnostic accuracy. RGB sensors have been used on robotic platforms for monitoring and estimating fertilizer needs for more than a decade [112,113]. As example, [114] used a smartphone (LG G5) to capture images of soil samples at different moisture levels, predicting soil organic matter (SOM) with R^2 values of 0.91–0.72 and soil moisture content with R^2 values of 0.84–0.86. Similarly, another smartphone-based application showed high accuracy for soil texture prediction [115]. Digital RGB images captured at a proximal scale can be combined with advanced computer-vision methods to gain insights into plant stress. A machine vision-based robotic system utilizing a CCD color camera was developed for monitoring plant health and growth in greenhouses. By surveying the trends of changing image textural features of crops, a power equation was presented as the growth model of greenhouse crops under conventional conditions to ensure optimal nitrogen fertilization [116]. Several studies have used color, morphological, and structural features to detect stress [112,113,117,118]. Similar to the visual inspections, the high resolution offered by RGB images allows for the classification of healthy and unhealthy plants based on color characteristics. [119] developed a color image segmentation task.

Among recent developments, the variable application of agrochemicals can be based on leaf area. For example, [120] developed a real-time variable-rate fungicide spraying system using correlations between a CROP-Meter sensor and canopy leaf area index. Similarly, [19] developed an image processing algorithm to estimate disease density, allowing the corresponding required amount of chemical to be applied to the target area.

However, the quality of RGB images is very sensitive to light and can present information only in the visible spectrum [18]. Conversely, multispectral and hyperspectral imaging have become key tools for disease detection and plant phenotyping [35].

Close-range measurements have been successfully applied for the identification of leaf diseases based on hyperspectral data, with [105,121] and without imaging [103].

Within the optical domain, the spectral response of photosynthetic pigments is not unique, as several pigments absorb in overlapping spectral regions [122]. Within the optical domain, the spectral response of photosynthetic pigments is not unique, as several pigments absorb in overlapping spectral regions[123–125] or multiple stresses simultaneously [22,95]

Despite the potential of digital and multispectral imaging to assess disease severity [126], several challenges remain for real-time field deployment due to the high variability in field conditions [127].

Lighting conditions and sensor positioning significantly affect performance. For example, the viewing angle influences detection sensitivity[128]; and [129] showed that fixed-base systems miss symptoms developing on leaf undersides.

Illumination heterogeneity is another limiting factor. To mitigate this,[128] combined spectral indices with a gradient-based approach, while [119] shielded imaging scenes from direct sunlight.

Although the concept of using imaging systems has been extended to crop disease management [130],automated and selective spraying for diseases still has some limitations. Therefore, despite this being a currently thriving research field, there remains enormous potential for improvement.

Hyperspectral/multispectral snapshot sensor, using RGB-like mosaic principles, has recently emerged, offering higher speed but lower spatial resolution compared to conventional hyperspectral scanners [131].

Measurement scale is also critical: better model performance is generally obtained at the sub-leaf level compared to canopy-scale measurements [132,133], partly due to reduced external disturbances in non-imaging sensors. In hyperspectral imaging, spatial resolution is directly tied to disease detection accuracy.

Small-scale investigations by [121] for the detection of *Cercospora leaf spot and* powdery mildew on sugar beet, showed that differences between healthy and diseased tissues diminish at lower spatial resolutions, due to mixed pixels [134].

Although the integration of chromatic, morphological, and structural features—including NIR information—can significantly enhance classification accuracy, current sensing strategies still face major limitations related to cost, operational complexity, and environmental variability. Advanced sensors such as hyperspectral, thermal, and 3D imaging

systems offer clear advantages for pathogen detection, yet their high price, calibration requirements, and energy demand limit their widespread adoption in everyday farming[135]. For this reason, future research should prioritize cost-effective and practical multi-sensor solutions, combining affordable RGB or multispectral cameras with low-cost thermal modules and lightweight depth sensors. Recent advances in edge computing, embedded vision, and open-source robotics offer promising opportunities to develop compact, farm-ready systems capable of real-time disease detection without relying on expensive laboratory-grade instruments.

1.3. Robotics and automation for crop monitoring and decision support for variable rate application (VRA)

Researches focus on refining data fusion techniques [136], expanding the application of precision technologies across different crops and regions, and ensuring their accessibility and ease of use for farmers. The most direct use of monitoring data is their application in variable rate application (VRA). Both map-based and sensor-based approaches are employed for this operation. In some applications, herbicide savings, calculated in terms of untreated area, can reach up to 39.2% with UAV-based applications, with a cost reduction ranging from 16 to 45 € ha^{-1} [137]. Also, [67] used UAV to monitor vineyard canopies, generating a vigor map that was subsequently converted into a prescription map using the DOSAVIÑA® software. This allowed real-time adjustments of pesticide spraying parameters, reducing application by 45% compared to conventional methods. Continuing along this line of research, the authors developed a variable-rate sprayer that demonstrated accurate pesticide distribution with coverage values of 20-40%, achieving biological effectiveness against powdery mildew. Similarly, [73] evaluated two VRA strategies for the application of copper in vineyards, reducing pesticide use by 33-44% / ha.

On the other hand, one of the main advantages of implementing sensor-based variable rate application (VRA) systems lies in their ability to adjust, in real time, the actuators of agricultural machinery that apply inputs, without the need for mapping or prior field data collection [40].

VRA based on remote sensing has several limitations, as it relies on a limited number of samples per hectare [118] and may be affected by errors related to the location of sampling points, the position of the applicator, and map interpolation [40]. The site-specific sensor-based system can overcome these issues by focusing on real-time data acquisition and decision-making [118]. This continuous stream of sensor data must then be transferred and converted into operational information and recommendations, which are subsequently implemented by a real-time controller [40].

The use of integrated sensors on agricultural machines is well-established and known as on-the-go sensor applications. Several site-specific sensor-based applications have already been developed and evaluated for variable-rate fertilizer application [118] and spot applications for agrochemicals and fungicides [118,138]. [118] developed an on-the-go VRA phosphorus (P) fertilizer system, used a vis-NIRS soil sensor tractor-mounted mounted saving fertilizer rate (average phosphate applied on VR plots was 28.75, 1.25 kg/ha^{-1} less than the UR (30

kg/ha^{-1}) recommended according to the standard soil test). [113] proposed an on-the-go system of variable rate herbicide and fungicide using a μ-eye color camera tractor-mounted, saving (9.90–51.22 %) of chemical applications in wild blueberry fields.

The new era of agriculture is moving toward vehicles, also known as "*agrobots*", that can operate autonomously, reducing the need for labor and increasing efficiency through variable rate application (VRA) of inputs. Compared with the automated driving approach, based on the interoperability between agricultural tractors, actuators, and on-board computers that led to the development of the international standard ISO 11783 (ISOBUS), *agrobots* do not require direct human intervention but are capable of autonomously performing various and repetitive agricultural tasks [139].

A classification, based on the comparison of robots with humans, and thus on their level of autonomy, is suggested by [140]. These authors demonstrated the feasibility of human-robot collaboration spraying system, saving 50% of agronomic input. Similarly, [116] developed a farmer-assistant robot for nitrogen fertilizing management of greenhouse crops. A machine vision robotic platform was able to decrease the nitrogen fertilizer consumption by about 18%.

To reduce workload and labor costs, many studies have investigated the development of fully autonomous machines. [128] tested an agricultural robot equipped with a precision-spraying end-effector and a machine vision-based disease-sensing system for spot spraying over the diseased grapevine canopy area. The testing results showed a reduction in pesticide use from 65% to 85% as compared to conventional homogeneous spraying of the canopy. [19] developed and tested an image processing-based variable-rate chemical sprayer assisted with a remote monitoring system for disease and pest-infested coconut plantations. The prototype developed has been shown to be able to reduce labor requirements, prevent chemical hazards to workers, and increase coconut pest control efficiency.

More recently, [119] developed a variable-rate spraying system for precise application of agrochemicals based on plant disease severity that showed a significant reduction in the amount of chemical used compared to the conventional uniform spraying. The smart sprayer reduced spray volume by 47% and 51% for weed and diseased plant detection experiments, respectively, compared to a Constant-rate Application (CA).

Some researchers tried to provide a cost-effective solution using digital image analysis and a computer-based decision system [141]. [142] developed a smart sprayer using machine vision and artificial intelligence to distinguish target weeds from non-target objects (e.g., vegetable crops) to precisely spray on the desired target/location, achieving an overall

precision of 71% and recall of 78% (for plant detection and target spraying accuracy). Similarly, classification CNNs (Convolutional Neural Networks) models have been used to classify weeds by [143] with an overall accuracy on the sprayed weeds target of 93%. More recently, [144] developed a low-cost agricultural robot for spraying fertilizers and pesticides in agriculture fields. In order to keep the costs to a minimum, the fertilizer and pesticide spraying robot prototype was assembled using simple, cost-effective, and off-the-shelf components. The low-cost robot was developed, consisting of a mobile base, a wireless controller for controlling the movement of the robot, and a camera providing a live video feed for general crop health.

A simple configuration of these vehicles frequently uses a digital camera as an effective means of recognizing the position, size, shape, color, and texture of vegetation [129]. The more widespread use of robots in indoor environments has meant that imaging sensors derived from remote sensing (multispectral and hyperspectral) are also needed for leaf-based applications [18,145] As an example,[18] developed a RobHortic remote-controlled field robot for inspecting the presence of pests and diseases in horticultural crops using proximal sensing. The robot was equipped with color, multispectral, and hyperspectral (400–1000 nm) cameras, located looking at the ground (towards the plants). Many robotic platforms developed by researchers are depicted in Figure 1.4.

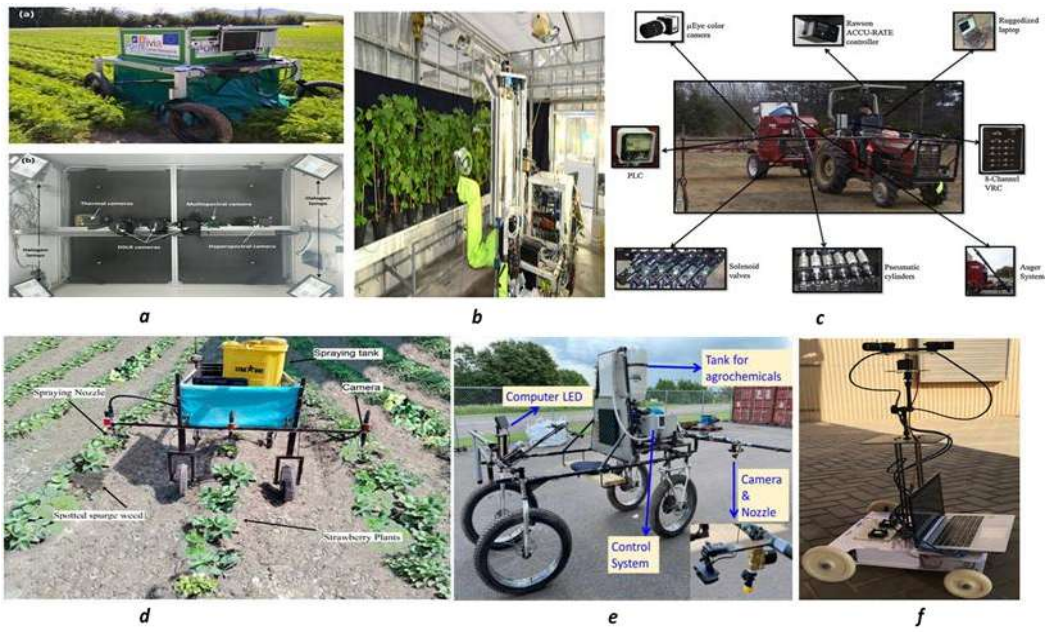


Figure.1.4. Robotic platforms for crop health monitoring and variable rate application (VRA) developed in recent years: (a) field robot to detect pests and disease[18]; (b) selective spraying for disease control using a modular agricultural robot [128]; (c) variable rate spreader for real-time spot-application of granular

fertilizer [113]; (d) variable rate ag-rochemical spraying system for targeted weeds control [143]; (e) smart variable-rate sprayer for targeted application of agrochemicals [146]; (f) robotized early plant health monitoring system [145].

However, hyperspectral imaging is a very costly technique, and there is still a limitation in the capability of designing systems for the detection of diseases in real time under field conditions. Most techniques are completed under controlled environments and automation techniques are still not fully implemented [145]. This holds back real-time applications due to both the high computational analysis time and the drop in performance at higher speeds, which is a challenge for real-time processing.

Moreover, although fertilizer and pesticide spraying robots are capable of carrying large storage tanks [143], the problem remains that they are too complicated, slow, and expensive to be made available to the public [18,142,145].

As a result, the agricultural sector is still lagging behind in integrating modern technologies [144] even to operate safely and even autonomously, there are other steps to be taken to ensure satisfactory safety.

To enable robots to perform harvesting, pesticide spraying, and monitoring tasks, a mobility mechanism for navigating within the farm, sensors for recognizing crops and the farming environment, and manipulators or other tools for carrying out specific tasks are essential.

Despite its promise for labor safety and to reduce the exposure to chemicals [19], navigating robots over rough and uneven terrain or in different weather conditions makes the agricultural robots navigation even more challenging [146], which means that navigation algorithms must adapt to identify specific features of the target crop [128,147]. Several challenges remain in deploying agricultural robots effectively. Indoor environments, such as greenhouses, present specific issues, including navigation difficulties, network disruptions, and robots that are designed to address only particular tasks. One notable challenge is the narrowness of lanes and corners in greenhouses, which are designed to maximize crop production space. Additionally, the structural materials of smart farms, such as steel, vinyl, and glass, can interfere with communication signals, leading to potential malfunctions if the connection is lost. In addition, other limitations are due to specific and operational conditions, such as lighting [146], and speed [142].

This translates into the need for state-of-the-art sensors and equipment, such as LIDAR or stereoscopic imaging systems, needed for guidance systems [148].

More frequently, researchers suggest a ROS-based platform to mitigate these operational challenges [149].

Hence, PA can be considered a form of advanced agriculture, aimed at using techniques and technologies focused on the variable application of crop inputs (e.g., water, fertilizers, and pesticides) within the fields based on the real needs of the crop and the chemical-physical and biological properties of the field. This is done mainly in order to pursue benefits due to both the increase or stable maintenance of yield (agronomic benefits) and rationalizing inputs and saving on crop costs (environmental and economic benefits). This perspective highlights the findings of [6], which explores PA technologies that can decrease greenhouse gas emissions while enhancing farm productivity and economic benefits.

1.4. Research gaps and objectives of the thesis

Despite the rapid evolution of precision agriculture, several challenges still limit the widespread and effective deployment of smart technologies for the optimized application of agrochemicals. Current sensing approaches often face difficulties in early stress detection, sensitivity to environmental variability, and constraints related to sensor alignment, lighting conditions, and canopy geometry. Additionally, many advanced sensing systems rely on costly hardware that is difficult to scale and integrate into routine farm operations. Limited interoperability between optical sensing, robotics, and decision-support tools further restricts practical adoption.

To address these challenges, the general objective of this thesis is the development of a smart cart suitable for agrochemical distribution and for supporting harvest operations.

To achieve this objective, the present thesis has developed three complementary research lines, each articulated into a specific scientific objective:

- Crop health modelling using multispectral sensors, with a focus on understanding sensor limitations (e.g., misalignment, short-distance distortions) and improving the reliability of stress detection.
- Harvest assistance through VIS–NIR systems, aimed at developing non-destructive methods for estimating quality and maturity attributes, with particular emphasis on creating cost-effective and practical predictive tools.
- Operational development of the robotic prototype “A-bot X1”, designed at laboratory scale, integrating optical sensors, autonomous navigation, and real-time data acquisition for future site-specific agrochemical applications.

The main results of this thesis aim to define a concrete pathway for integrating accessible sensors, robotic systems, and predictive models, converging toward a common vision: making agriculture more sustainable, efficient, and digitally advanced through technologies that are truly applicable in the field.

CHAPTER 2

2. Case study 1: Multispectral sensing to assess the health status of baby leaf crops

2.1. Introduction

Advanced techniques like multi- and hyperspectral imaging provide spectral and spatial information, making them useful for food and agricultural applications. The ability of multispectral cameras to collect data on a limited number of spectral bands, as opposed to hyperspectral imaging, allows for a more rapid and cost-effective assessment of plant physiology, including size, shape, and color, but also presents the potential for disease detection [150] . Over the last few years, several researchers have proposed phenotyping platforms to estimate several morphological indices for plants grown in greenhouses, such as weight, height, and leaf area [151,152] supporting cultivation and weed management decisions [153]. In contrast to traditional point spectroscopy, imaging spectroscopy allows for easier to automatic acquisition of spectral data from specific crop regions. To identify and detect drought stress [154–156], weed competition [153], the impact of environmental changes [157], nutrient deficits [158], and to distinguish between various types of stress [159], multi and hyperspectral imaging has proven to be effective. Although the integration of sensors is considered one of the major advantages underlying short-range phenotyping platforms [160], the acquisition of images from one or more sensors or different points of view has emerged as an open challenge in the field of image analysis [150].

Misalignment is particularly problematic when using low-altitude imaging systems that capture images from non-orthogonal angles relative to the plant surface. This can lead to significant errors in spectral index calculations, compromising the quality of the analysis and, consequently, the accuracy of crop health diagnoses [161].Recent studies have highlighted the relevance of image registration methods in the case of non-georeferenced multispectral images acquired at close range; for example, to identify the presence of powdery mildew on cucumber plants [162], water stress on lettuce [163], identification of weeds in wheat [164], and the effects of potassium water deficit in cassava [165]. The Scale-Invariant Feature Transform (SIFT) algorithm [165] and Speeded-Up Robust Features

(SURF) [166] are commonly used for feature detection, based on the alignment of fixed and moving image spaces.

One of the main advantages of these methods is that they are invariant to scale changes. SIFT is also invariant to orientation and illumination [167]; however, it struggles with processing time constraints. In contrast, SURF was developed to overcome this issue [168].

With the increasing use of computer vision, other feature detection techniques have been proposed, such as KAZE, AKAZE, ORB, and BRISK, with an overview and comparison provided by [169].

Although registration techniques differ in preserving image parameters like scale, angles, etc., they are based on affine transformations. An affine transformation combines translation, rotation, scaling, and image shearing (parallel distortion). According to the comparative analysis proposed by [170], the affine transformation was found to be the most accurate in terms of multispectral image registration accuracy (RMSE < 1 pixel). Applying automatic image registration techniques has been the focus of several researchers, who have found success using open-source programming. Different sensor perspectives have been used to test registration algorithms in complex ecosystems [166] and on individual plants [165] .

There is a lack of a metric to adjust the registration offset of images related to changes in the sensor acquisition distances. Therefore, the primary objective of this chapter is to further investigate the implications of sensor acquisition distance in the calibration of multispectral imaging systems, specifically addressing the gaps in image alignment. The methodological framework and results presented in this chapter consolidate previous published research on multispectral image alignment and calibration by introducing a unified and quantitative approach to account for acquisition distance variability:

Laveglia, S.; Altieri, G. (2024). *A Method for Multispectral Images Alignment at Different Heights on the Crop*. In: Cavallo, E.; Auat Cheein, F.; Marinello, F.; Saçılık, K.; Muthukumarappan, K.; Abhilash, P.C. (Eds.), 15th International Congress on Agricultural Mechanization and Energy in Agriculture (ANKAgEng 2023). Lecture Notes in Civil Engineering, Vol. 458. Springer, Cham. https://doi.org/10.1007/978-3-031-51579-8_36.

Laveglia, S.; Altieri, G.; Genovese, F.; Matera, A.; Scarano, L.; Di Renzo, G.C. (2025). *Development of a Short-Range Multispectral Camera Calibration Method for Geometric Image Correction and Health Assessment of Baby Crops in Greenhouses*. Applied Sciences, 15, 2893. https://doi.org/10.3390/app15062893.

2.2. Image registration

Image registration (also known as image alignment) is the process of finding correspondences between two or more images, such that there is a high degree of affinity between the corresponding parts of the images, a common objective is the fusion of the distinct information of different images captured at different times or by different modalities. There is a rich taxonomy of image registration scenarios and methods; an important factor is whether the images come from the same modality (monomodal), sharing appearance and structure, or not (multimodal/cross-modal). Methods are typically divided into feature-based approaches, where sparse sets of feature points are extracted and then matched, and intensity-based (area-based) approaches where distances or similarities incorporating information from the entire images are used to find correspondences. Furthermore, it is common to categorize methods and scenarios w.r.t. the flexibility of the transformation model, e.g., deformable (non-rigid) or rigid/affine, since this has a large impact on the complexity of the process.

2.2.1. Transformations

Selection of the transformation model is one of the main decisions when performing image registration; it is important to select a transformation model, or a sequence of multiple models, that fits the scenario at hand. The more flexibility a transformation supports, the greater is the risk of finding a transformation that unrealistically distorts the image structures, and the more challenging is the optimization process with a greater number of local optima. If a transformation model is selected with too little flexibility, the registration process will fail to find a transformation that successfully matches all the corresponding parts of the images. Selecting an unsuitable flexible model typically yields suboptimal results. An illustration of the transformations presented here is shown in the Figure. 2.1.

2.2.2. Affine transformations

Affine transformations are operators $T : \mathbb{R}^n \rightarrow : \mathbb{R}^n$ which can be modelled as $n + 1 \times n + 1$ matrices using homogeneous coordinates

$$T(x) = \begin{bmatrix} \alpha_{11} & \cdots & \alpha_{n1} & t_1 \\ \vdots & \ddots & \vdots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nn} & t_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ 1 \end{bmatrix} \quad (2.1)$$

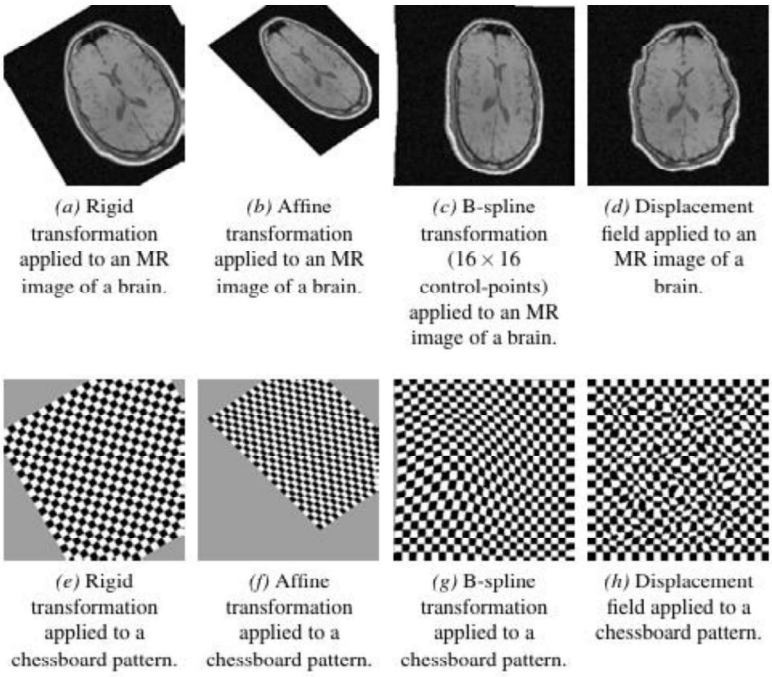


Figure 2.1. Four transformation models presented in the ascending order of flexibility, applied to a T1 MR image of a brain, and shown in the upper row (a-d). The lower row (e-h) shows the same transformations applied to a chessboard pattern. Original MR images are from [171].

The transformations that are expressible by an affine transformation are combinations of translation, rotation, scaling, shearing, and reflection.

2.2.3. Rigid transformations

Rigid transformations are combinations of rotations and translations. They are subsets of the affine transformations in 2D and 3D, with the extra imposed constraint of orthonormality on the non-translation part of (2.1),

$$\begin{bmatrix} \alpha_{11} & \cdots & \alpha_{1n} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nn} \end{bmatrix} \tag{2.2}$$

Rigid transformations may either be encoded as the matrix (6.1), two tra.nslation parameters and an angle for rotations in 2D and three translation parameters, or three Euler angles or unit quaternions for rotations in 3D.

2.2.4. Multi-scale and multi-resolution strategies for image registration

Multi-scale processing is an important part of image registration, focusing on coarse registration of larger structures first, followed by registration of finer structures; it is especially important for most of the common similarity measures based on overlapping points (e.g., NMSD, PCC, and, MI), since operating at a larger scale increases their so-called receptive field. Rather than a multi-scale scheme, a multi-resolution scheme can be used; the

down sampling of a multi-resolution scheme is very important in some contexts and less important for others. For simplicity of presentation, multi-scale is the term used for the remainder of the chapter to refer to both multi-scale and multiresolution schemes unless specifically noted. The most common approach to multi-scale image registration is to use Gaussian pyramids, where the large scale/lower resolution images are processed first, followed by the small scale images. At each stage, the transformation is initialized from the transformation of the previous stage. For some similarity measure and transformation combinations (e.g., NMSD, PCC, and MI combined with displacement fields), it may be highly beneficial for the run-time to operate on down-sampled images. For other combinations (e.g., NMSD, PCC, and MI combined with affne transformations), multi-scale processing without downsampling is sufficient since there may not be much extra cost to keep the original image sizes. For B-spline transformations, multi-resolution B-spline grids are useful in addition to the Gaussian pyramid, where a sequence of B-spline transformations with an increasing number of control points are considered at each stage, performing a spline fitting procedure to initialize the higher resolution transformation as close as possible to the lower resolution transformation.[172]

2.2.5. Monomodal and multimodal registration

Monomodal image registration refers to image registration scenarios where the images are acquired through a highly similar process and share appearance and structures, whereas multimodal image registration refers to image registration scenarios where the images are acquired through different processes and may have a distinct appearance with structures visible in only one of the images. Monomodal image registration can, in principle, be solved using NMSD as a similarity measure, or PCC if there are minor differences between the images, such as differences in mean and contrast between the images. A feature-based method can also be used. Multimodal image registration tends to be substantially more challenging, requiring more advanced similarity measures or sophisticated representation extraction, such as MI [173] or the similarity of NGF [174] .

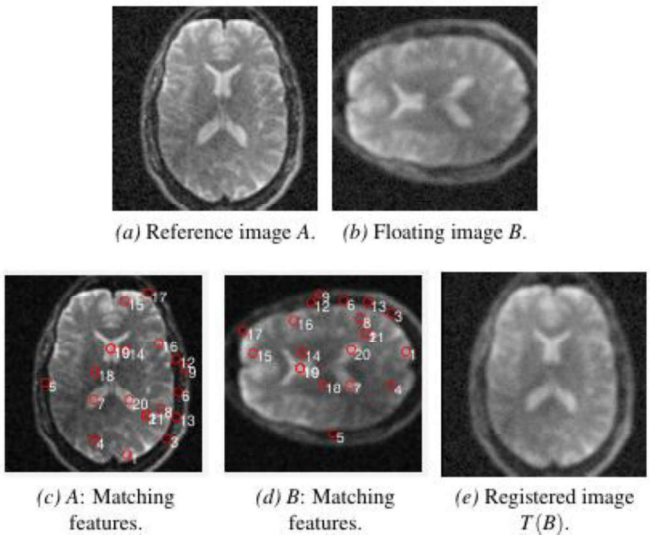


Figure 2.2. Feature-based monomodal registration applied to two MR T2 brain images (a-b) that are registered with SIFT feature descriptors and a rigid transformation model. A random subset of 20 matching feature points are shown in (c-d). The registration is successful and the resulting image T(B) (e) is well aligned with (a). Images are from[171]

2.2.6. Feature-based image registration

Feature-based image registration consists of finding a set of distinct points such that an accurate mapping between the two image spaces can be established. Feature-based methods for image registration are highly useful for parametric transformations (e.g., rigid and affne) and images with enough common sparse and salient features such that several matching features can be successfully detected and matched. Natural images acquired with cameras tend to ft this profle well, and these methods are commonly used in computer vision. The Scale-Invariant Feature Transform (SIFT) [175] is one of the most popular feature detectors and descriptors for 2D images, which is invariant to image scaling, rotation, and translation, as well as some insensitivity to affine or 3D projection and illumination variation. An approach to feature-based registration can be formulated as follows: Feature points are detected, descriptors are computed, and then the feature points are matched, which can be done by, e.g., a brute force nearest neighbor search. Finally, the transformation is estimated from the matched sets of points; one common approach to estimate the transformation from the matched point sets is random sample consensus (RANSAC) [176], which operates on randomly selected subsets and excludes outliers. If the feature extraction method fails to detect good feature points, the entire registration fails as well.

2.2.7. Intensity-based image registration

Intensity-based image registration is a general methodology that relies on the similarity of the entire image areas/volumes to find a transformation between images,

enabling the registration of images lacking distinct spatial markers. Images lacking distinct spatial markers are common in biomedical applications, particularly in multimodal scenarios.

Intensity-based image registration can be formulated as an optimization process. Given a distance measure d between images or a negated similarity measure and a set of valid transformations Ω, intensity-based registration of two images, *A* (floating) and *B* (reference), can be formulated as the optimization problem,

$$\hat{T} = \underset{T \in \Omega}{argmin}\, d(T(A), B), \tag{2.3}$$

where T(A) denotes a valid transformation of image A into the reference space of image B. However, (2.3) is in general ill-posed (in particular for deformable image registration). The following formulation may be used instead:

$$\hat{T} = \underset{T \in \Omega}{argmin}\, d(T(A), B) + R(T), \tag{2.4}$$

where R denotes a regularization functional on T, modeling prior knowledge (or preference), e.g., smoothness or invertibility of the solutions (if not already guaranteed through the choice of Ω).

2.3. Structure from Motion (SfM) and 3D Reconstruction

3D reconstruction is a complex task due to the high amount of information required to perform it. The main problem is to estimate the depth of the points that were lost with their projection in the 2D world. This chapter will give an overview on the main approaches to building a 3D model starting from at least two images. Firstly, camera calibration will be described to estimate the mapping parameters from the 3D to the 2D world. Then, triangulation, matching, and stereocalibration will be considered. Finally, Structure-from-Motion (SfM) and dense point cloud reconstruction will be described.

2.3.1. Calibration and pose estimation of a single camera

A picture is the projection of the 3D world into the 2D domain. To define this mapping, the parameters that model the camera that performs the projection have to be retrieved. This process is called calibration. To know the number of parameters that have to be estimated, the camera model has to be introduced.

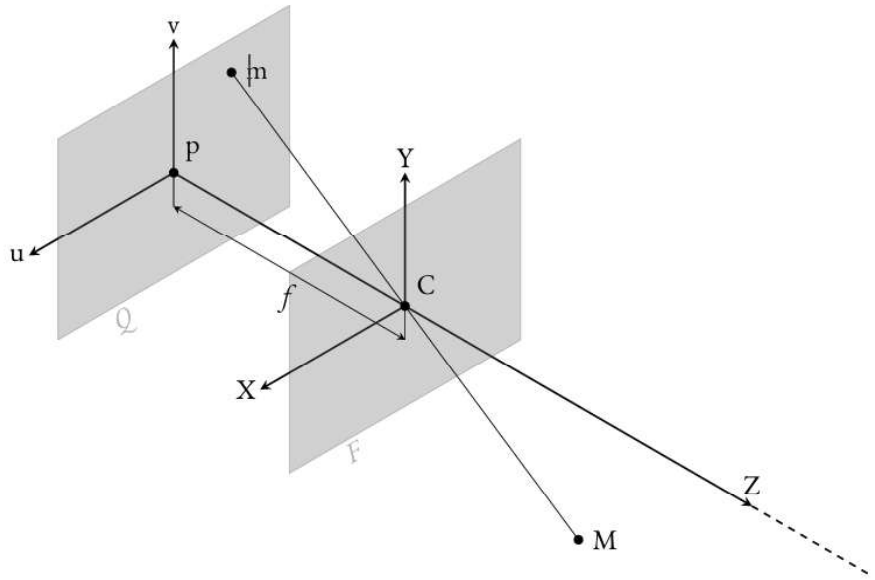


Figure.2.3. ProjectioFigura 2n of the3D point M into m,2D point

2.3.2. Pinhole model

The pinhole model is the basic model of a camera and holds under some assumptions [177].

In this model, the world reference system is represented by (X, Y, Z), and it is centered in *C*, called *Centre Of Projection* (COP). The Z axis is called the principal axis. In this model, it coincides with the optical ray, so the world reference system and the camera one coincides. The plane *F*, that is the perpendicular plane to the principal ray and it is centered in the COP, is called the *focal plane*. The parallel plane to *F*, called *Q* in Figure. 2.3, is called image plane. Its distance from *F* is *f*, where f is called the *focal length*. It is centered on the principal point *p,* and it contains the camera reference system (u, v).

Therefore, we can write a 3D point, in Cartesian coordinates, as M = $\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$, and its 2D projection through C, belonging to *Q*, as **m**=$\begin{bmatrix} u \\ v \end{bmatrix}$

From similar triangles, Fig. 2.4, we can see that

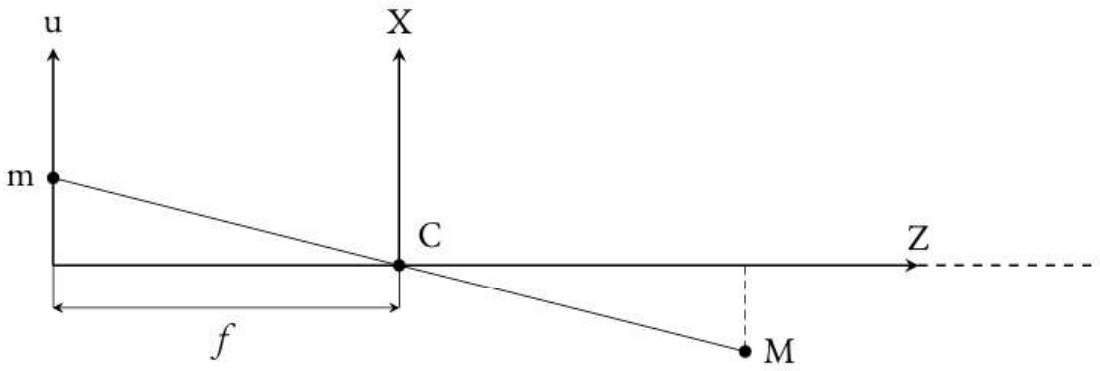


Figure 2.4. 2D view of the projection of the 3D point M into m

$$\frac{f}{Z} = -\frac{u}{X} = -\frac{v}{Y} \qquad \begin{cases} u = -\frac{f}{Z}X \\ v = -\frac{f}{Z}Y \end{cases} \tag{2.5}$$

This mapping is nonlinear if we consider Cartesian coordinates. To have a linear relation, homogeneous coordinates need to be used instead. The 2D and 3D points in homogeneous coordinates are represented as:

$$m = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \qquad M = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.6}$$

The mapping can then be vritten as:

$$Z = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = 2\begin{bmatrix} -fx/z \\ -fy/z \\ 1 \end{bmatrix} = 2\begin{bmatrix} -fX \\ -fY \\ Z \end{bmatrix} = \begin{bmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.7}$$

Therefore, the projection of a 3D point into a 2D one is:

$$m = -\frac{1}{2}PM \qquad \text{or} \qquad m \simeq PM \tag{2.8}$$

Where the last equation gives us equality up to a constant and with P, called *Camera Projection Matrix*, defined as:

$$P = \begin{bmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.9}$$

2.3.3. General Model

This model is an extension of the pinhole model: it takes in consideration differences between the camera and world coordinate frames and the conversion from meters to pixels. To have this last transformation, the matrix A is used:

$$A = \begin{bmatrix} -k_u & 0 & u_0 \\ 0 & -k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.10)$$

With (u_0,v_0) coordinates of the principal point and *u* and *v* the number of pixels per unit distance in image coordinates in *u* and *v* directions.

P can now be defined as:

$$P = \begin{bmatrix} -k_u & 0 & u_0 \\ 0 & -k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = K[I \mid 0] \quad (2.11)$$

Where

$$K = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.12)$$

With $\alpha_u = fk_u$ and $\alpha_v = fk_v$

K contains the internal characterization of the camera defined by four internal camera parameters: α_u, α_v, u_0, and v_0.

To completely describe the internal structure of the camera,we have to consider a parameter called *skew*: it is the angle between the *u*-axis and the *v*-axis. If it is considered, we can rewrite K as:

$$k = \begin{bmatrix} fk_u & -\frac{fk_u}{tan\theta} & u_0 \\ 0 & -\frac{fk_v}{sin\theta} & u_0 \\ 0 & 0 & 0 \end{bmatrix} \quad (2.13)$$

Where the skew is defined as $\gamma = \frac{f \cdot k_u}{\tan \theta}$

Usually, θ is considered equal to $\frac{\pi;}{2}$, therefore γ=0, i.e., the skew can be neglected.

When the camera reference system is different from the world coordinate frame, we need to introduce an isometry to make the two systems coincide. It is composed by a rotation, R, followed by a translation,t. We can write this transformation as:

$$M_c = VM \quad (2.14)$$

Where Mc is the point in the camera coordinate frame, M is the one in the world reference system and V is the view matrix:

$$V = \begin{bmatrix} R & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} \quad (2.15)$$

V provides information about the orientation and position of the camera with respect to the world reference system. The parameters contained in V are called *external parameters* of the camera. Therefore, we can describe the mapping from the 3D world to the 2D camera sensor as:

$$\mathrm{m} \approx \mathrm{k}[I|O]V\mathrm{M} = \mathrm{k}[R|\boldsymbol{t}]\mathrm{M} = P\mathrm{M} \qquad (2.16)$$

with

$$\mathrm{P} = \mathrm{k}[R|\boldsymbol{t}] \qquad (2.17)$$

2.3.4. Zhang calibration technique

To compute the internal parameters of our cameras, the Zhang calibration method was used [178]. It requires several pictures of the same planar pattern, usually a checkerboard-like pattern, taken with different orientations without changing the internal parameters of the camera. Without loss of generality, we can assume that the checkerboard is on the plane Z =0 of the world coordinate system. For each picture, a projection matrix P=k[R|t] is computed. The intrinsic parameters, , are the same for each image, while *R* and *t* change for each image. Using the identified corners of the board, the homography for each picture can be computed up to a scale factor. It has 8 degrees of freedom, so to estimate the parameters of the projection matrix, which has 11 degrees of freedom, at least 3 pictures are needed. 5 of these 11 degrees of freedom are given by the intrinsics, so they are the same for all the images, while 6 are for the extrinsic, and they differ from image to image.

Re-projecting the corner projection back to the image, we can minimize the error with an iterative process. In this way the calibration procedure gives more accurate results.

2.3.5. Radial distortion

The introduction of real lenses in the camera model generates distortion effects.

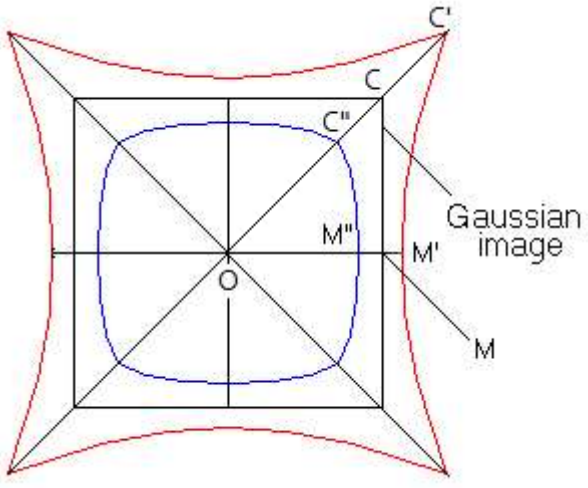


Figure 2.5. Distortion effects caused by the lens. The black rectangle is the undistorted ideal case, the red one represents the effect of pin-cushion distortion, while the blue one is the barrel distortion case.

We can write the transformation from undistorted, ideal coordinates (*u, v*) to distorted ones ($\hat{u}$,$\hat{v}$) as:

$$\begin{cases} \hat{u} = (u - u_0)(1 + k_1 r_d^2) + u_0 \\ \hat{v} = (v - v_0)(1 + k_1 r_d^2) + v_0 \end{cases} \quad (2.18)$$

where $r_d^2 = \left(\frac{u-u_0}{a_u}\right) + \left(\frac{v-v_0}{a_v}\right)$ and (u_0, v_0) are the coordinates of the principal point. The coefficient that represents radial distortion is k_1.

Writing $x = \left(\frac{u-u_0}{a_u}\right)$ and $y = \left(\frac{v-v_0}{a_v}\right)$ we have:

$$\begin{cases} \hat{x} = x(1 + k_1 (x^2 + y^2)) \\ \hat{y} = y(1 + k_1 (x^2 + y^2)) \end{cases} \quad (2.19)$$

To compute the distortion coefficient, the camera projection matrix P is needed, but to compute P, we need to know k_1.

This problem is therefore solved iteratively: firstly, P is estimated. Using this result, k_1 is then computed and P is refined after distortion removal. This process is repeated up to convergence.

2.3.6. Stereo rig

When a 3D point is projected into a 2D one, information regarding its position in the 3D world is lost. To recover it, we can use two different images taken from slightly different points of view. This operation is called *triangulation* and can be performed when the calibration parameters of the two cameras are known. Of course, the correspondence between the same point in the two images must also be known.

2.3.7. Triangulation

Triangulation is the procedure that allows us to estimate the position of a point in space starting from its projections onto two images (i.e., conjugate points). To accomplish this, the camera matrices of the two cameras that captured the images must be known.

2.3.8. Simple stereo case

Let us consider a stereo pair composed of two cameras with the same focal length f and parallel image planes. The corresponding projection centers, *C* and *C′,* the corresponding centres of projection, are on theX-axis at a distance *b*, called *baseline*. The reference system of the left camera coincides with the world coordinate system (Figure 2.6).

A 3D point M (in the figure, only the X and Z axes are shown) is projected onto two 2D image points: m = (u, v) and m′ = (u′, v′). The first belongs to the image plane of the left camera, while the second lies on the image plane of the right camera.

Using the similitude among triangles, we can write:

$$\begin{cases} \frac{f}{z} = -\frac{u}{x} \\ \frac{f}{z} = -\frac{u\prime}{b-x} \end{cases} \quad (2.20)$$

Where *b* and *f* depend on the camera: the former is related to the extrinsic parameters and the latter to the intrinsic ones. From 2.20 we can write:

$$x = \frac{-bu}{u'-u} \quad (2.21)$$

Using equations 2.20 and 2.21, we can finally write the equation for the z component:

$$z = \frac{bf}{u'-u} \quad (2.22)$$

We have now the relation between the depth z and the disparity d=u ′ - u equation 2.22: z is inversely proportional to the disparity.

Computing the disparity, we can therefore estimate the depth z and reconstruct the coordinates of M.

2.3.9. General case

The situation described in the previous section is very restrictive; therefore, we need a more general solution.

We can write the two camera matrices in the following way:

$$P = \begin{bmatrix} p_1^T \\ p_2^T \\ p_3^T \end{bmatrix} \qquad P' = \begin{bmatrix} p'^T_1 \\ p'^T_2 \\ p'^T_3 \end{bmatrix} \tag{2.23}$$

where each element of the vector is a row of the camera matrix.

We can then write:

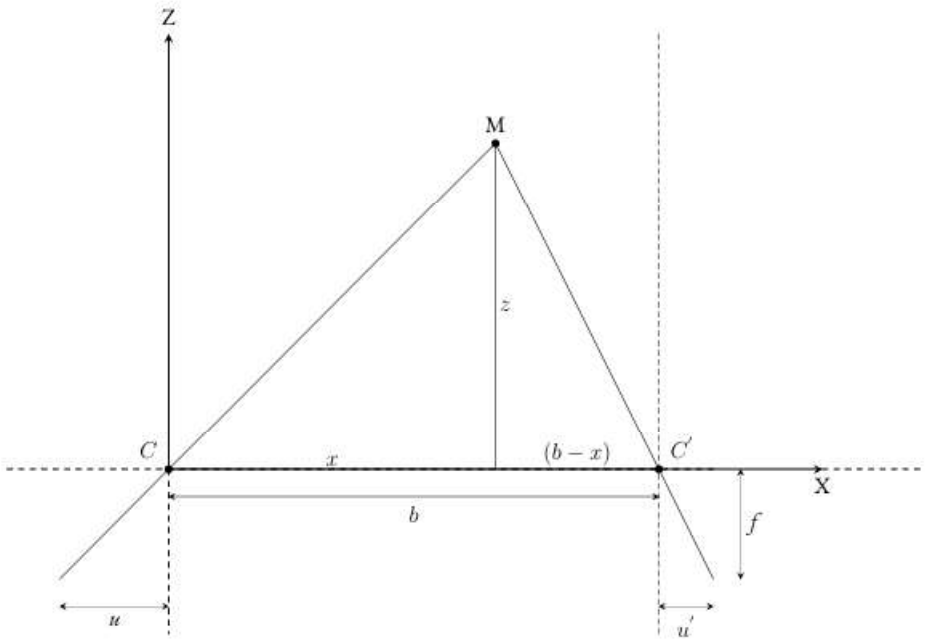


Figure 2.6. Naive triangulation case.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P\mathrm{M} \qquad \begin{cases} u = p_1^T M \\ v = p_2^T M \\ 1 = p_3^T M \end{cases} \tag{2.24}$$

from which we obtain:

$$\begin{cases} up_3^T M = p_1^T M \\ vp_3^T M = p_2^T M \end{cases} \qquad \begin{bmatrix} p_1^T - up_3^T \\ p_2^T - up_3^T \end{bmatrix} M = 0 \tag{2.25}$$

Repeating this process also for the second camera we have:

$$AM = \begin{bmatrix} p_1^T - up_3^T \\ p_2^T - up_3^T \\ p_1'^T - u'p_3^T \\ p'^T_2 - u'p_3^T \end{bmatrix} M = 0 \tag{2.26}$$

The solution of equation (2.19) is the kernel of the matrix A. We can decompose A using SVD and selecting the last eigenvector. The problem with this solution is that it is

algebraic and does not consider the geometric error. To minimize the geometric cost, we can re-project back the point and compute the error with respect to its original position and iteratively try to minimize it. The cost function is the following:

$$\epsilon(M) = \left\| \begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} \frac{p_1^T M}{p_3^T M} \\ \frac{p_2^T M}{p_3^T M} \end{bmatrix} \right\|^2 - \left\| \begin{bmatrix} u' \\ v' \end{bmatrix} - \begin{bmatrix} \frac{p_1'^T M}{p_3'^T M} \\ \frac{p_2'^T M}{p_3'^T M} \end{bmatrix} \right\|^2 \qquad (2.27)$$

2.3.10.Rectification

The rectification operation consists of bringing back the two cameras to the configuration presented in section "triangulation", without changing the coordinates of the two centres of projections C and C'. That setup simplifies the computation because the two conjugate points, which we are trying to match, lay on the same line: we reduce the number of possible conjugate points (our search becomes unidimensional from bi-dimensional), therefore the number of possible false matches.

We need the rectification operation because usually we have a different system with respect to the naive one.

To perform this operation, the two cameras are rotated around their center of projection (to make the two image planes parallel) and their reference systems are translated to obtain the situation in which $v = v'$. We can write the rotation matrix:

$$R = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix} \qquad \begin{cases} r_1 = \frac{\bar{C}\prime - \bar{C}}{\|\bar{C}\prime - \bar{C}\|_2} \\ r_2 = k \; x \; r_1 \\ r_3 = r_1 \; x \; r_2 \end{cases} \qquad (2.28)$$

where k is an arbitrary versor that is orthogonal to the new Y axis. The optical ray associated to m and m' do not change.

If, after these operations, we have still $v \neq v'$, we can obtain the equality translating the reference systems.

2.3.11.Stereo rig calibration

Stereo calibration is the process that allows to estimate the transformations between two camera orientations of a rig. These transformations will be constant during the acquisitions if the rig is not modified.

The needed information to perform stereo calibration are the intrinsic parameters of the two cameras and pairs of acquisitions of the same planar cheeseboard-like pattern. It will be therefore possible to compute Rstereo and tstereo that are the rotation matrix and the

translation vector that defines the cameras orientation difference. From them it is possible to estimate also the baseline as $b = ||tstereo||$.

2.3.12.Matching strategies

Having two views of the same object taken with a small, but not too small, baseline, it is possible to find the same point that is projected in the two different images: we can associate m with m'. Using triangulation, we can then recover the position of the point in the space. Repeating this procedure for all the pixels in the picture, we can build a dense 3D model. We call this process *disparity* estimation.

To perform the matching process, two different approaches can be used:

• local methods: they consider only a small window around the pixel that we want to match;

• global methods: they pose the constraint on the whole line or on the whole image.

2.4. Materials and methods

2.4.1. Stereo camera calibration

As mentioned before, camera calibration aims to establish the transformation from the three-dimensional (3D) real world to two-dimensional (2D) digital images. In general, the simplest case assumes that the camera operation can be described by an ideal pinhole model, known as the Pinhole model [177]When working with a multi-camera (stereo) system, it is possible to determine the geometric relationship (rotation and translation) between the two captured images [179]; this process is known as stereo calibration.

Stereo calibration is commonly used in autonomous vehicle navigation to estimate the geometric and optical parameters related to the position of the stereo system [180]. As is well known, the relationship between homologous points in the left and right images of a stereo pair (known as disparity (d)) allows for the determination of the depth (Z) of the captured scene.

$$Z = f * B/d \qquad (2.29)$$

where: f is the focal length of the sensor, B is the baseline distance between the camera pair, and d is the disparity.

Experimental images of a checkerboard pattern (35 mm squares, printed on an A3 sheet and fixed to a rigid plywood surface); The images have been taken from different distances from the target (11 selected distances, ranging from 500 mm to 1500 mm with a step of 100 mm). At each distance, the target has been positioned centered in axis with regard to the camera and moved (out of axis) left, right, inside and outside, taking an overall of five images for each distance from the target (Figure 2.7).

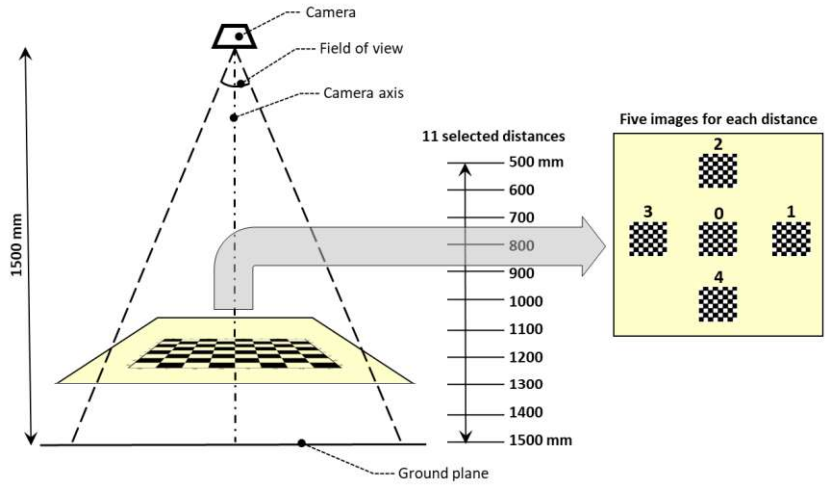


Figure 2.7. Experimental design.

The multispectral images were captured using a MicaSense RedEdge P™ multispectral (MS) camera (AgEagle Aerial Systems Inc., Wichita, Kansas, USA). This camera features a sensor resolution of 1456 × 1088 pixels (1.6 MP per band) and captures images in five spectral bands: Blue (B) (475 nm ± 32 nm), Green (G) (560 nm ± 27 nm), Red (R) (668 nm ± 14 nm), Red Edge (RE) (717 nm ± 12 nm) e NIR (NR) (842 nm ± 57 nm). The proposed binocular system (Figure 2.8) utilizes the Red (R) band sensor as a fixed reference lens, while the sensors of the other spectral bands (G, B, NIR, RE) function as relative lenses.

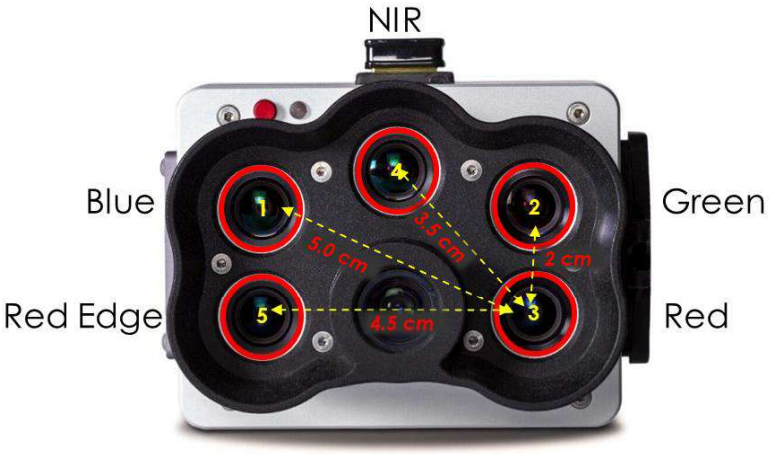


Figure 2.8. Location of the optical lens of the Micasesnse RedEdge P sensor and relative pose of the spectral bands (B, G, NR, RE) to the R band, with distance from the reference lens (cm).

The acquired images were processed using the flowchart proposed by [181] within the Camera Calibration Toolbox in MATLAB software [182].

Briefly, the stereo calibration process was implemented using the '*estimateCameraParameters*' function, applied to checkerboard images, which computed the internal and external parameters of each camera, including rotation matrices (*R*), translation vectors (*T*), and intrinsic calibration. The calibration accuracy was assessed using the reprojection error, calculated as the pixel distance between detected and reprojected corner points. Subsequently,'*rectifyStereoImages*' was used to geometrically align the stereo image pairs, ensuring that conjugate epipolar lines became parallel and horizontally aligned, simplifying disparity computation. The disparity map was then generated using *'disparitySGM'*, which applied a semi-global matching algorithm to determine the pixel-wise displacement between rectified stereo pairs. Finally, the *'reconstructScene'* function converted the disparity information into 3D world coordinates, reconstructing the scene geometry based on known camera calibration and baseline distance.

The obtained calibration parameters were applied to a new set of images to validate the system's accuracy in 3D reconstruction. In this phase, the distance to a known target, a potted plant**,** was estimated using images acquired at fixed distances (800 mm, 1000 mm, 1300 mm, and 1400 mm). The distance between the camera lens and the target was calculated by determining the Z-component (depth) in the 3D reconstruction space, corresponding to a Region of Interest (ROI). To reduce computing time, the depth (Z) was calculated using up to 150,000 randomly selected points within the ROI. This approach optimized computation time while maintaining accuracy. Finally, the relative error provided an indication directly related to the accuracy of the system in terms of depth measurement in real-world scenarios.

2.4.2. Image registration

Image alignment aims to overlay two or more images using spatial transformations such as translations, rotations, and cropping [183]. Typically, images are aligned by selecting one image as the reference image and aligning it with the others (referred to as the moving images).The multispectral images were acquired according to the experimental setup described before.(2.4.1)

The acquired band images are misaligned, in the sense that their pixels don't correspond to each other. The misalignment requires that each band images must be shifted along the X and Y direction in order to align the image bands to correctly analyze the spectral content of each pixel.

The operation is performed considering an image band as the fixed or reference band and then moving the other band image in order to guarantee a satisfying alignment for each pixel. Unfortunately, the X and Y offset to be considered change with respect to both the considered band and the distance from the target. To solve this issue,the checkerboard utilization arises from the alignment algorithm found in the Image Processing Toolbox of Matlab software. In particular, the function '*detectCheckerboardPoints'* allows for the measure of the points coordinates of a checkerboard pattern image based on the work of Geiger et al., 2012 [184].

The basic inspirating idea is extremely simple and pertains to the class of so called point-based methods. Indeed, if a set of corresponding points pairs can be identified for a given set of band images, then the alignment can be achieved by selecting some kind of transformation aligning these points. As these fiducial points must be reliable for the purposes of alignment, they must have clearly identifiable features and, therefore, the use of a checkerboard guarantees this characteristic. Indeed, the Matlab function

'*detectCheckerboardPoints*' allows for the automatic identification of the checkerboard pattern fiducial points coordinates. Therefore, the transformation to be applied must simply align the corresponding fiducial points for each band image.

Consequently, for each band image (at the same distance from the target), the detection algorithm is applied and, for each band, the algorithm outputs a series of 48 fiducial points (exactly where two black squares cross two white squares), an example is shown in Figure 2.9.

Then, calculating the differences with respect to the band image considered fixed (in our case the Red band), 48 X offsets and 48 Y offsets are generated, X offsets must be the same with low statistical noise, due to distortions induced by the lens, and the same must be valid for the Y offsets; thus, the average of these offsets, for each band, brings to the determination of the offsets that must be applied to align the bands along the X dimension and along the Y dimension.

The same happens when considering the images out of axis, even for these, the averaged X offsets and Y offsets are generated.

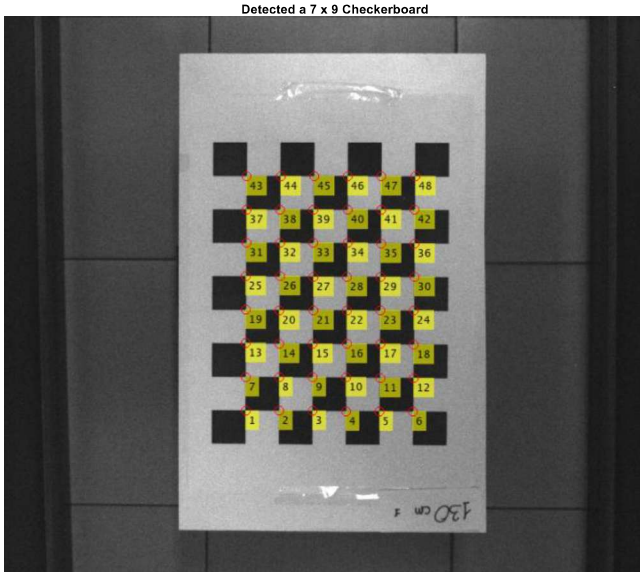


Figure 2.9. Identification of the 48 points of the 9x7 checkerboard; the band B at 1300 mm distance (centred with the camera axis) is shown.

In addition, the offsets calculated for the five band images at the same distance from the target must be the same with only some statistical noise. Therefore, the final offsets that must be considered they are the average of the offsets of the five band images on the same plane (i.e. taken at the same distance). Indeed, as final result, the averaged offset is considered. However, these values are valid only for the considered distance. Therefore, the previously exposed alignment method is repeated considering the images collected at

different distances. The described alignment method is a simple translation of the band images. The more general transformation to align two images (a moving image with regard to a fixed one) is an affine transformation. The affine transformation is a linear transformation preserving collinearity and ratios of distances. In general, an affine transformation is composed of four basic transformations, i.e. rotation, translation, dilation and shear. All these basic transformations are specified by 3x3 matrices in 2D space as the use of the augmented matrices technique allows for the simply matrix multiplication when composing the various listed basic transformations.

The Image Processing Toolbox of Matlab software contains a complete set of functions to handle this type of transformations. The preliminary calculation of the affine transformations on a series of test band images confirmed that they are subject to translation only, with negligible rotation, dilation and shear. The transformation matrices, related to each band image, depend on the distance from the target. E.g., focusing on the X offset related to the Blue band (the fixed band is the Red), the data show how they change with respect to the distance from the target. These data have been successfully interpolated using as inverse linear functions: $\boldsymbol{y=a+b/x}$.

In this equation, y represents the offset, x is the distance, and a and b, are model parameters estimated through the analysis of experimental data.

From a physical perspective, the value of the constant a can be interpreted as the minimum geometric offset for alignment, while b represents a weighting factor between the offset (y) and the distance (x). As the distance increases ($x \to \infty$), the term b/x approaches zero, and therefore, y approaches the value of a. Under these conditions, the geometric offset is minimally affected by the distance, and the model simplifies to: $y \approx a$. On the contrary, as the distance decreases ($x \to 0$), the term b/x increases, thereby significantly affecting the offset (y).

The relationship expressed by the model is particularly useful in contexts where the geometric offset is strongly dependent on the distance between the sensor and the measurement object, such as in close-range applications.

Because of this, the alignment procedure has been repeated at varying distances from the target in order to achieve a good modelling of the affine transformation matrices (Figure 2.10).

Furthermore, considering the repetitive nature of the used checkerboard, the use of a bi-dimensional Fourier Transform has been positively tested. This algorithm arises from the work of Sicairos et al., 2008 [185]. In this case, for each image band, a single hot spot appears

in the complex plane corresponding to checkerboard centre, and, therefore, a single offset value along the X and Y directions is generated, without the need to estimate the mean value on all the fiducial points as in the previous case of the checkerboard. Compared to a conventional method has also been tested; that consists of the RG (Rigid Transformation), which involves a rigid affine transformation based on the similarity of homologous points between images.

Therefore, two image alignment methods have been tested:

- **CB**: image band offsets measure based on the checkerboard method;
- **FT**: image band offsets measure based on the bi-dimensional Fourier transform.
- **RG**: image band offsets measure based on rigid affine transformation

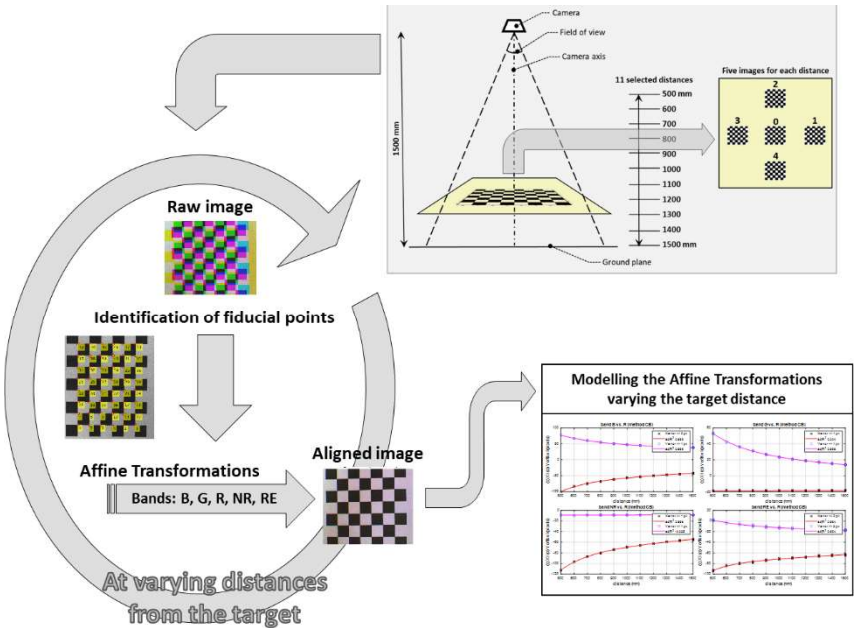


Figure 2.10. The alignment procedure is repeated at varying distances from the target to achieve a good modelling of the affine transformation matrices.

2.4.3. Crop health assessment

Baby leaf lettuce (*Lactuca sativa cv. Maverik*) plants grown in greenhouses were used to test the performance of the multispectral image alignment model. Images of individual baby lettuce leaves were acquired using the multispectral camera at a distance of 1 meter from the ground.

Images were collected from two groups of plants: lettuce grown under a conventional irrigation system (healthy) and lettuce not irrigated for 7 days (unhealthy). For each group, 100 images were acquired. To assess the physiological condition of the leaves, chlorophyll concentration (μmol/m²) was measured using an Apogee MC-100 Portable Chlorophyll Meter (Apogee Instruments, Inc., 721 West 1800 North, Logan, Utah, 84321, USA).

Several vegetation indices (VIs) (listed in Table 2.1) were calculated for each leaf following the application of the alignment models. The mean value of individual leaves was used to correlate spectral variations with chlorophyll content. Additionally, the NDVI with a threshold of 0.40 was applied as a binary mask to calculate the leaf area (cm^2).
Image analysis and vegetation index calculations were performed using MATLAB software (2023b) from MathWorks.

Table 2.1. Multispectral band ratios for the calculation of vegetation indices used.

Vegetation Indices	Formula	References
NDVI (Normalized Difference Vegetation Index)	(NIR-RED)/(NIR+RED)	[37]
MCARI (Modified Chlorophyll Absorption in Reflectance Index)	[(RE−RED)−0.2×(RE−G)]×(RED / RE)	[37]
SR (Spectral Ratio)	RED / NIR	[186]
CLr (Chlorophyll Red Index)	(RED / NIR) −1	[187]
NARI (Normalized Anthocyanin Reflectance Index)	(1/G +1/RED) / (1/G -1/RED)	[188]
ARI (Anthocyanin Reflectance Index)	(1/G) − (1/RED)	[26]
mARI (Modified Anthocyanin Reflectance Index)	(1/G − 1/RE) ×NIR	[189]
GNDVI (Green Normalized Difference Vegetation Index)	(NIR+G)/ (NIR-G)	[190]
SIPI (Structural Independent Pigment Index)	(NIR+RED)/(NIR−B)	[191]

2.5. Results and discussion

2.5.1. Stereo camera calibration results

The reprojection error, expressed as the mean value on the X- and Y-axis of the fiducial points found in the chosen pair of bands, clearly shows the bias on calibration accuracy (Figure 2.11).

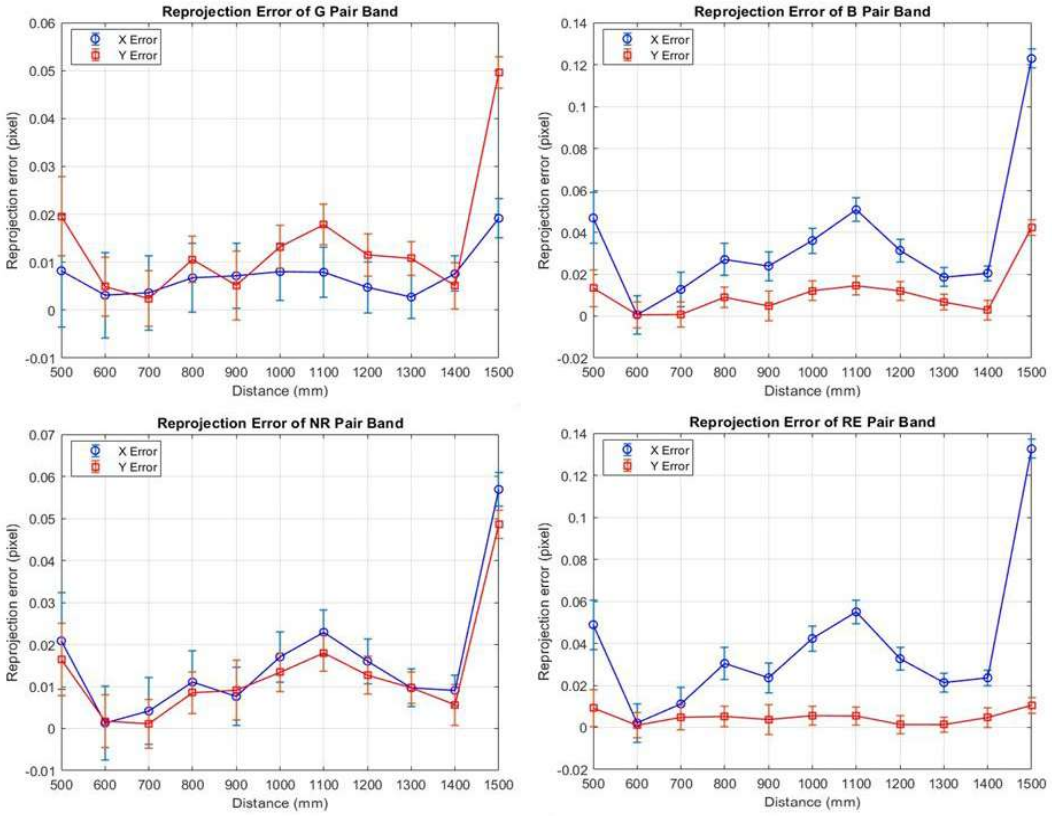


Figure 2.11. Averages of the x-axis and y-axis reprojection errors (pixels) for the stereo setups of the G, B, NR, and RE bands to the R band as a function of distance.

The overall trend of the reprojection error for several bands (G, B, NIR, RE) shows a uniform behavior for all pairs of bands relative to the distance.

Notably, higher errors are observed at extreme distances (500 mm and 1500 mm), while the lowest error values occur at intermediate distances, particularly between 600 mm and 700 mm, for both the X and Y coordinates. Reprojection error tends to increase with increasing distance, as shown in previous studies [192], while the error at lower distances could be due to the high resolution of the camera, which reduces the detected fiducial point size at these distances during the calibration process. However, the bands show differing behavior for errors along the axes. While the B and RE bands exhibit lower accuracy along the X axis, the error along the Y axis is generally more significant for the G band. The NR band, conversely, shows a homogeneous trend across both axes, with no notable impact from a single axis.

Despite these variations, the average values of the reprojection error, calculated for both the x- and y-axis, at the distances analyzed indicate a well-calibrated model. Figure 2.12 shows that the overall reprojection error of calibration patterns is much less than 1 pixel. This result demonstrates that, despite the observed variability, the system maintains good accuracy for practical applications within the calibration range.

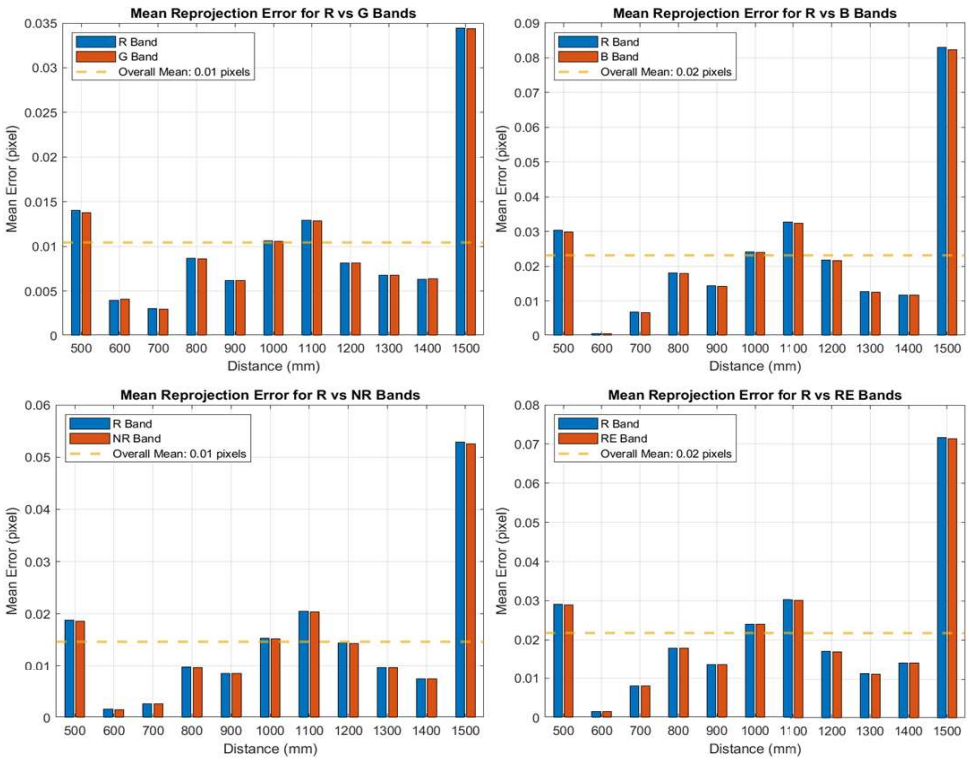


Figure 2.12. Reprojection errors expressed as the average values along the X and Y components for a single distance, relative to the stereo pairs used in the calibration process.

Table 2.2 summarizes the performance of the depth estimation (Z) in mm at several distances for the spectral band combinations (G vs. R, B vs. R, NR vs. R, RE vs. R). Overall, it is observed that the percentage error (Err%) associated with each stereo system increases proportionally to distance. As expected, this behavior reflects the performance limitations previously identified in the calibration dataset. The best results are observed at the shortest distance (800 mm), with percentage errors of 18.36%, 8.85%, 6.20%, and 4.72% for the G, NR, B, and RE bands, respectively. The worst results are estimated at 1000 mm, with percentage errors of 23.22%, 14.31%, 11.60%, and 10.41% for the G, NR, B, and RE bands, respectively. The B and RE spectral bands were found to be more accurate than the G and NR combinations, with lower percentage errors. Among the proposed stereo systems, the G-band showed the worst performance, suggesting that the reprojection error observed on the Y-axis may have significantly impacted the accuracy of this stereo configuration. Concerning the G-band, the error is relatively low at 800 mm (18.36%), but increases at longer distances, peaking at 23.22% at 1000 mm, and then decreasing to more moderate values (18.88% and 21.32% at 1300 mm and 1370 mm, respectively). Although this behavior was replicable for all the other band configurations, the RE band shows the best stability in terms of error, with a maximum of 1000 mm error of 10.41%. Also in this case, the effect of

the Y-axis reprojection error emerges, which for the RE band shows a more stable and linear behavior.

Table 2.2. Performance of the depth estimation (Z) in mm for stereo pairs (G vs. R, B vs. R, NR vs. R, RE vs. R) at fixed distances (800 mm, 1000 mm, 1300 mm, and 1370 mm). The number of points used (nPoints), excluding NaN values and outliers (a < 0.05), the mean value ± standard error, and relative error (%), are reported.

Depth estimation (Z) error, band G vs. R				**Depth estimation (Z) error, band B vs. R**			
dist_ mm	nPoints	mean ± std	Err%	dist_ mm	nPoints	mean ± std	Err%
800	1.49E+05	653.11 ± 2.86E-05	18.36	800	1.26E+05	750.43 ± 3.13E-05	6.20
1000	1.05E+05	767.77 ± 2.58E-05	23.22	1000	91071	884.05 ± 3.19E-05	11.60
1300	70961	1054.60 ± 5.35E-05	18.88	1300	55859	1207.90 ± 4.45E-05	7.08
1370	55048	1077.90 ± 1.02E-04	21.32	1370	45625	1238.10 ± 8.46E-05	9.63
Depth estimation (Z) error, band NR vs. R				**Depth estimation (Z) error, band RE vs. R**			
dist_ mm	nPoints	mean ± std	Err%	dist_ mm	nPoints	mean ± std	Err%
800	1.37E+05	729.23 ± 2.50E-05	8.85	800	99057	762.22 ± 3.23E-05	4.72
1000	99340	856.87 ± 2.85E-05	14.31	1000	82246	895.92 ± 3.38E-05	10.41
1300	52389	1173.40 ± 4.53E-05	9.74	1300	48286	1222.70 ± 4.19E-05	5.95
1370	49960	1195.80 ± 6.28E-05	12.72	1370	55965	1258.60 ± 8.57E-05	8.13

2.5.2. Image registration results

The image alignment results were evaluated by comparing the model performance parameters of the three selected methods (CB, FT, and RG). The coefficient of determination (R2) and Root Mean Square Error (RMSE) were chosen as the main indicators to assess the fitting of the model, as they provide an overall view of the goodness-of-fit and an average accuracy of the interpolation.

In Figure 2.13, is shown the result of the obtained offsets with respect to R band, along X and Y directions, related, for simplicity, only to centred images, identified as 0 in Fig. 1, using the CB method. These offsets must be applied to B, G, NR and RE bands to align their pixels with those of R band.

As it can see, the offset is changing with the target distance, however, the X offsets of G band and Y offsets of NR band are virtually constant with regard to the change in distance.

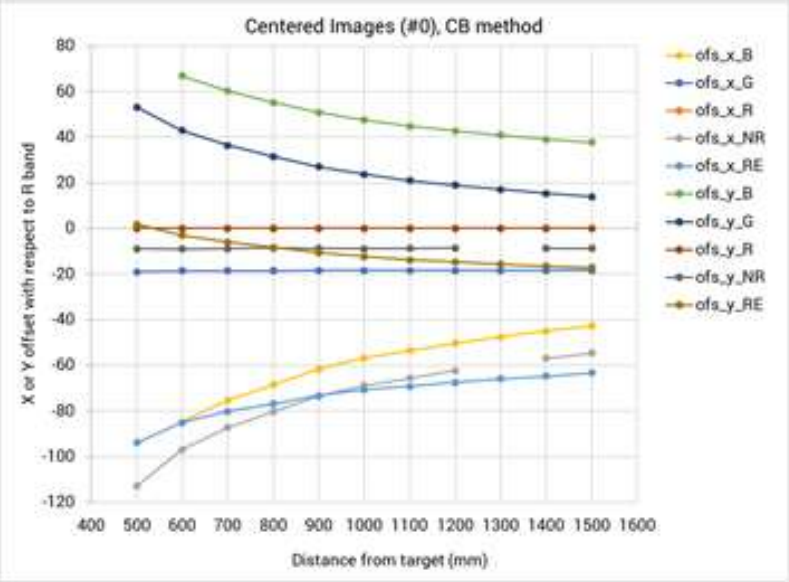


Figure 2.13. Result of the offsets with respect to R band, along X and Y directions, considering the only the centred images, using the CB method.

The accuracy of the spatial alignment between images was assessed by measuring the remaining residual displacement among the multispectral bands after fitting the model. This was expressed as the residual offset error along the x-axis (Xerr) and y-axis (Yerr) expressed in pixels.

Figure 2.14 shows the results of the interpolation of the model applied to the experimental data, considering the offsets of the B, G, NR, and RE bands to the R band. The offsets are analyzed both along the X- and Y-axes, and considering all five positions, using the CB, FT, and RG methods. The adjusted R2 indicates that all models achieved a very high degree of correlation with the experimental data. Similarly, the residual offset error along the x and y axes is, on average, between ± 2 and ± 3 pixels.

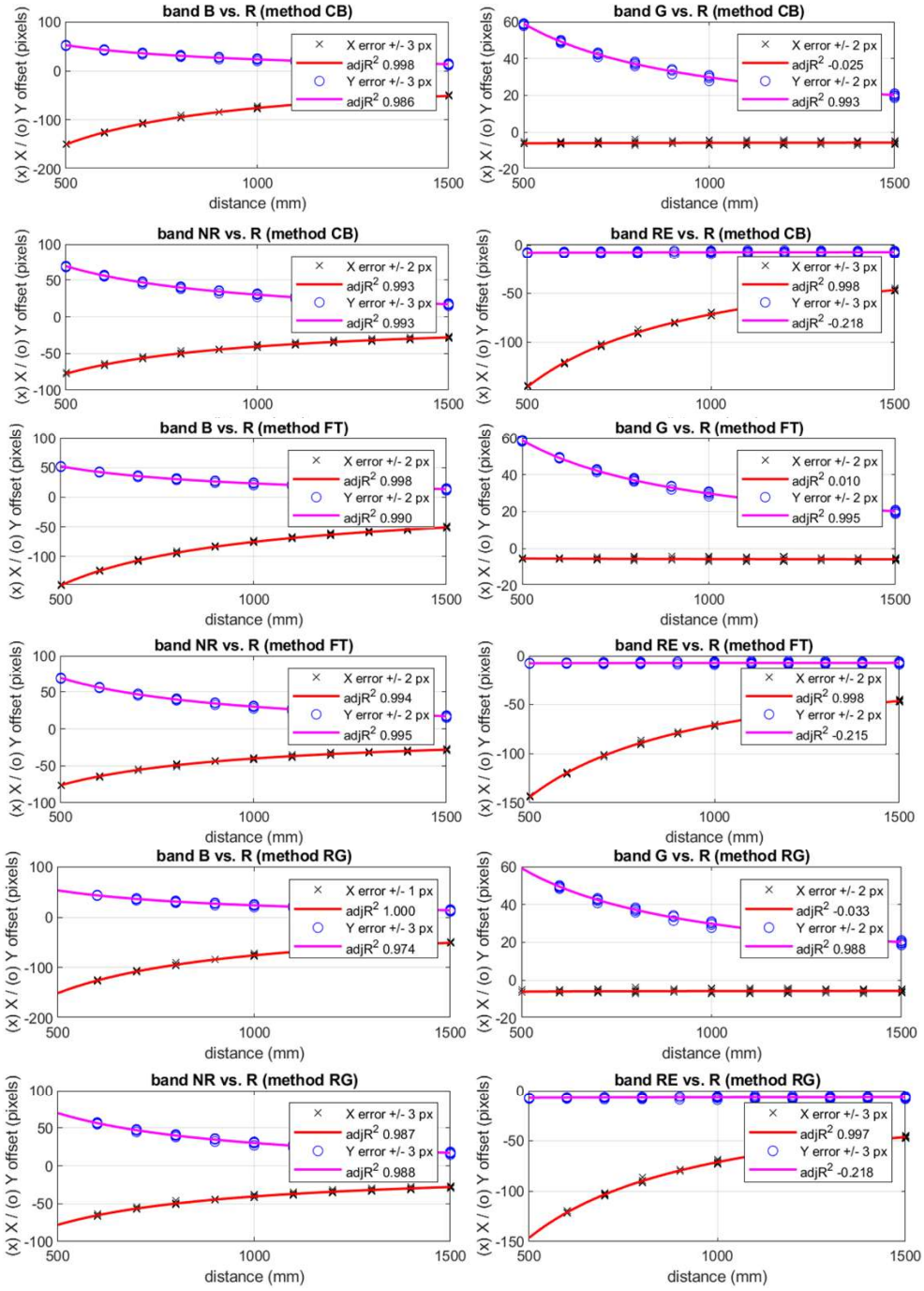


Figure 2.14. Results of the model interpolation on the experimental data, with the offsets of the B, G, NR, and RE bands relative to the R band, analyzed along the X-axis and Y-axis using the CB, FT, and RG methods.

The B and NR bands exhibited the best overall results for both axes, indicating a good fit for the model. Specifically, Tables 2.3, 2.4, and 2.5 show that the offsets of the B vs. R band demonstrated a high correlation (mean adjusted R^2: 0.9986 and 0.9835 on the x and y axes, respectively) and low errors (mean RMSE: 1.0529 and 1.3834 on the x and y axes, respectively). This same trend is confirmed by the results of the NR band offsets, showing mean adjusted R^2: 0.9916 and mean RMSE: 1.2636 along the x-axis, and mean

adjusted R^2: 0.9923 and mean RMSE: 1.2803 along the y-axis. In contrast, the G band, compared to the R band, exhibited the worst correlation results along the x-axis, as evidenced by an R^2 value of -0.0159. Similarly, the RE band showed fitting issues along the y-axis with mean adjusted R^2 values of -0.2168.

Table 2.3. Performance of the CB method correlation models of offsets on the X-axis and Y-axis. The 95% confidence limits of the model parameters are given in round brackets.

X offset MODEL, band B vs. R	**Y offset MODEL, band B vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -1.266 (-2.457, -0.07416)	a = -5.597 (-6.756, -4.437)
b = -7.432e+04 (-7.531e+04, -7.332e+04)	b =2.897e+04 (2.8e+04, 2.994e+04)
adjrsquare: 0.9977	adjrsquare: 0.9860
rmse: 1.4953	rmse: 1.4550
X offset MODEL, band G vs. R	**Y offset MODEL, band G vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -5.427 (-6.151, -4.703)	a = 0.7586 (-0.05633, 1.574)
b = -293.5 (-898, 311)	b = 2.909e+04 (2.841e+04, 2.977e+04)
adjrsquare: -0.0248	adjrsquare: 0.9930
rmse: 0.9082	rmse: 1.0229
X offset MODEL, band NR vs. R	**Y offset MODEL, band NR vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -3.452 (-4.464, -2.439)	a = -8.98 (-10.04, -7.919)
b = -3.716e+04 (-3.801e+04, -3.632e+04)	b = 3.897e+04 (3.808e+04, 3.985e+04)
adjrsquare: 0.9934	adjrsquare: 0.9934
rmse: 1.2709	rmse: 1.3311
X offset MODEL, band RE vs. R	**Y offset MODEL, band RE vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = 2.719 (1.645, 3.794)	a = -6.818 (-7.848, -5.788)
b = -7.421e+04 (-7.51e+04, -7.331e+04)	b = -409.5 (-1270, 451.2)
adjrsquare: 0.9981	adjrsquare: -0.2176
rmse: 1.3484	rmse: 1.2931

Table 2.4. Performance of the FT method correlation models of offsets on the X-axis and Y-axis. The 95% confidence limits of the model parameters are given in round brackets.

X offset MODEL, band B vs. R	**Y offset MODEL, band B vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -2.171 (-3.162, -1.18)	a = -5.471 (-6.427, -4.516)
b = -7.311e+04 (-7.394e+04, -7.229e+04)	b = 2.873e+04 (2.794e+04, 2.953e+04)
adjrsquare: 0.9984	adjrsquare: 0.9903
rmse: 1.2438	rmse: 1.1996
X offset MODEL, band G vs. R	**Y offset MODEL, band G vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -6.028 (-6.664, -5.392)	a = 0.83 (0.1613, 1.499)
b = 319.5 (-211.6, 850.6)	b = 2.898e+04 (2.842e+04, 2.954e+04)
adjrsquare: 0.0104	adjrsquare: 0.9952
rmse: 0.7980	rmse: 0.8393
X offset MODEL, band NR vs. R	**Y offset MODEL, band NR vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -4.199 (-5.128, -3.271)	a = -8.899 (-9.784, -8.014)
b = -3.599e+04 (-3.677e+04, -3.521e+04)	b = 3.877e+04 (3.803e+04, 3.951e+04)
adjrsquare: 0.9940	adjrsquare: 0.9954
rmse: 1.1650	rmse: 1.1107
X offset MODEL, band RE vs. R	**Y offset MODEL, band RE vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = 1.81 (0.8442, 2.775)	a = -6.91 (-7.78, -6.04)
b = -7.298e+04 (-7.379e+04, -7.218e+04)	b = -304.6 (-1031, 422.1)
adjrsquare: 0.9984	adjrsquare: -0.2151

rmse: 1.2115	rmse: 1.0918

Table 2.5. Performance of the RG method correlation models of offsets on the X-axis and Y-axis. The 95% confidence limits of the model parameters are given in round brackets.

X offset MODEL, band B vs. R	**Y offset MODEL, band B vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -0.8794 (-1.293, -0.4655)	a = -6.139 (-7.659, -4.618)
b = -7.479e+04 (-7.517e+04, -7.441e+04)	b = 2.956e+04 (2.813e+04, 3.099e+04)
adjrsquare: 0.9997	adjrsquare: 0.9742
rmse: 0.4196	rmse: 1.4956
X offset MODEL, band G vs. R	**Y offset MODEL, band G vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -5.421 (-6.224, -4.619)	a =0.4707 (-0.5691, 1.511)
b = -301.1 (-1004, 401.8)	b = 2.939e+04 (2.843e+04, 3.036e+04)
adjrsquare: -0.0333	adjrsquare: 0.9877
rmse: 0.9353	rmse: 1.0540
X offset MODEL, band NR vs. R	**Y offset MODEL, band NR vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = -3.174 (-4.51, -1.837)	a = -9.352 (-10.73, -7.972)
b = -3.739e+04 (-3.863e+04, -3.615e+04)	b = 3.936e+04 (3.809e+04, 4.064e+04)
adjrsquare: 0.9874	adjrsquare: 0.9880
rmse: 1.3549	rmse: 1.3990
X offset MODEL, band RE vs. R	**Y offset MODEL, band RE vs. R**
General model: y(x) = a+b/x	General model: y(x) = a+b/x
a = 3.196 (1.836, 4.556)	a = -6.818 (-7.848, -5.788)
b = -7.472e+04 (-7.598e+04, -7.346e+04)	b = -409.5 (-1270, 451.2)
adjrsquare: 0.9967	adjrsquare: -0.2176
rmse: 1.3787	rmse: 1.2931

The coefficients of the model, a and b, exhibited variability both across the bands and along the axes. In particular, the wide 95% confidence intervals for the b coefficient along the x and y axes for the G and RE bands confirm the poor performance of the model for these two bands. Nevertheless, the obtained model reasonably explained the effect of the x distance on the y geometric offset, highlighting an inverse proportionality between x and y, compensated by the value of b. Example calculations are summarized in Table 6, which presents the reference values of the Y offset model for the B vs. R band of CB method (a = -5.597, b = 28.970) found in Table 2.3.

From the example, it is evident that assuming a sensor distance of x = 30 meters, the ratio b/x tends to 0.966 pixels, which results in an alignment offset y of -4.631 pixels. Considering the relative error with respect to a, calculated as the percentage of the ratio between the term b/x and $|a|$, it is observed that the term b/x contributes a relative error of 17.26%, which is not negligible and may be problematic for high-precision applications. Repeating the calculation at greater distances, such as 50 meters and 100 meters, the y offset remains constant (≈ -5 pixels), while the relative error progressively decreases. These results highlight that the b parameter has a significant effect at short distances ($x \rightarrow 0$), but its influence diminishes at higher distances, where y(x) approaches the constant value defined

by *a*. However, as the distance decreases, the relative error increases, suggesting that the model must necessarily be robust.

This physical interpretation is particularly important for applications requiring high alignment accuracy, as confirmed by the error values reported in Table 2.6. The ability to describe the trend of offsets as a function of distance through an analytical relationship improves the applicability of the model, offering a more interpretable and generalizable solution

Table 2.6. Examples of calculating the model y(x)=a+b/x and relative error (Err%), using reference coefficients values of: a =-5,597, b=28,970, and distance (x) =1, 10, 20, 30, 50 and 100 meters.

Distance (m)	b/x (pixel)	y(x)=a+b/x (pixel)	Relative Error (err%)
1	28.970	23.373	517.91%
10	2.897	-2.700	51.77%
20	1.449	-4.148	25.88%
30	0.966	−4.631	17.26%
50	0.579	−5.018	10.34%
100	0.290	−5.307	5.18%

The alignment models were applied to a set of baby lettuce images acquired at a distance of 1000 mm from the ground using a multispectral (MS) sensor. The results of the alignment are presented in Figure 2.15, which shows the raw images of the R, G, and B channels of the MS sensor alongside the corresponding images aligned using the CB, FT, and RG methods. The aligned images demonstrated satisfactory results, with precise alignment of the bands, thus confirming the effectiveness of the proposed methods. The alignment of the multispectral bands plays a crucial role in accurately assessing the physiological condition of the crops. This further supports their application in short-range image analysis contexts (Section 2.5.3).

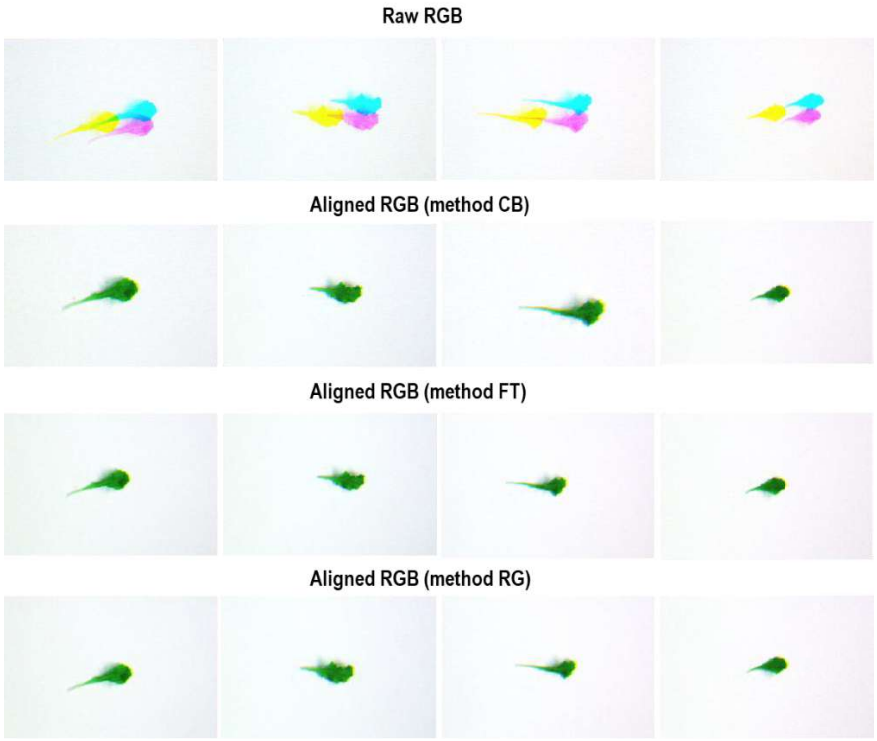


Figure 2.15. Raw images of R-, G-, and B-bands of the multispectral sensor (MS) and corresponding aligned images using CB, FT, and RG methods on baby lettuce leaves.

2.5.3. Crop health assessment results

The correlation of the vegetation indices with chlorophyll content was calculated for all samples, including both healthy and **unhealthy** crops, with a significance threshold of $p < 0.05$, using Pearson's correlation coefficient. Pearson's correlation matrix helped to identify the most significant correlations (Figure 2.16).

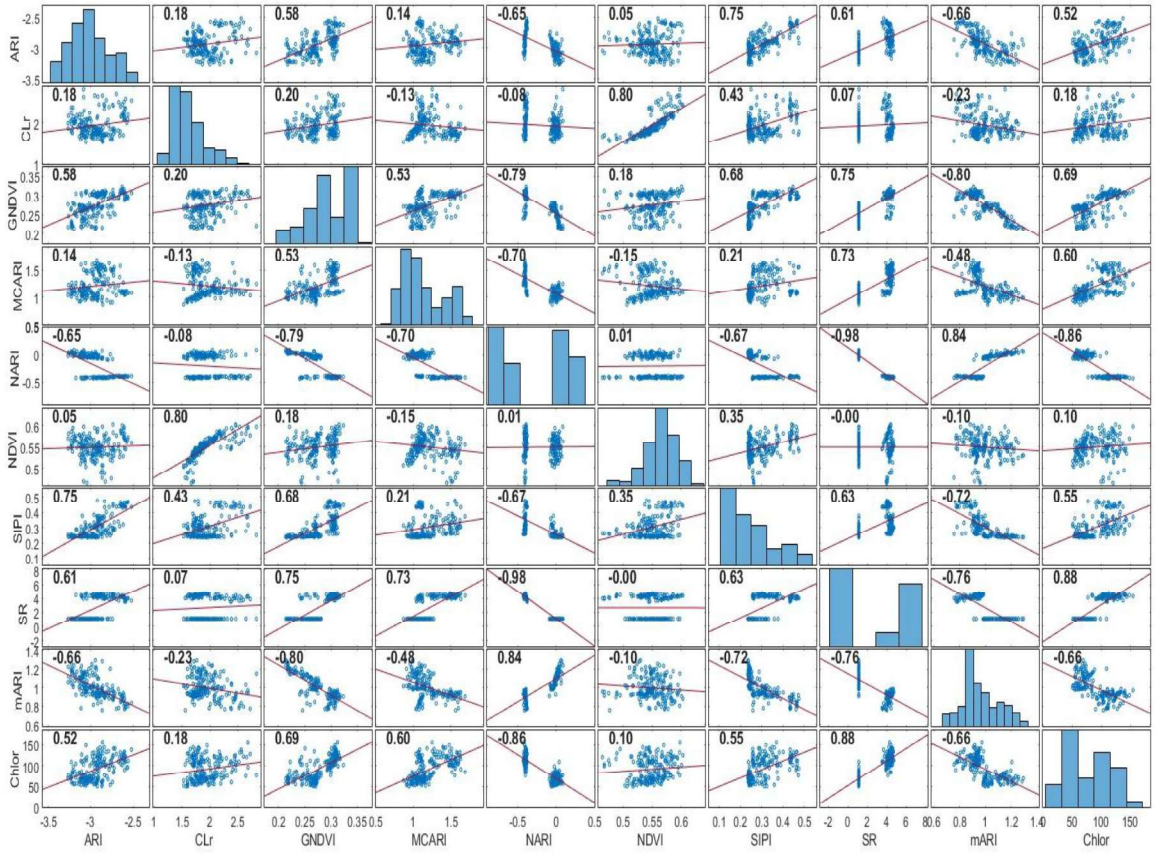


Figure 2.16.Correlation matrix plot of Pearson coefficient among chlorophyll content and different vegetation indices for the lettuce leaves.

The SR, GNDVI, and MCARI indices show a positive correlation with chlorophyll content (r = 0.88, 0.69, and 0.60, respectively). ARI and SIPI also exhibit a stronger correlation (r = 0.52 and 0.55, respectively). A low correlation was observed for NDVI (r = 0.10) and CLr (r = 0.18). Furthermore, among the vegetation indices analyzed, only NDVI did not reach the significance threshold (p-value > 0.05).

Vegetation indices have been widely used to quantify plant photosynthetic absorption, particularly chlorophyll content and anthocyanins. The SR index serves as an indicator of canopy structure, light absorption, and photosynthetic capacity [26], while GNDVI and MCARI are more sensitive to foliar chlorophyll and have been used to estimate photosynthetic activity under water stress conditions [165].

On the other hand, NARI and mARI show a strong negative correlation with chlorophyll content (r = -0.86 and -0.66, respectively). These VIs are highly correlated with anthocyanin content [193], which was not considered in this study. Although the correlation results are not optimal for a predictive model of plant photochemical content under water

deficit conditions, baby lettuce exhibited reduced leaf area and higher chlorophyll values. The significance of the differences between leaf area and chlorophyll values for healthy and non-irrigated plants was analyzed using a post-hoc Tukey test (α=0.05). As shown in the statistics in Figure 2.17, this significant difference may reflect a plant strategy to optimize photosynthetic efficiency under water stress, which helps explain the observed correlations. As a reaction to water stress, crops typically show foliar adaptation mechanisms to reduce light absorption, such as decreased chlorophyll concentration and down-regulation of photosynthesis [21], as well as increased concentrations of antioxidant components [14] In addition, wilting and reduced leaf area in water-stressed crops are well-documented [194], and have a significant influence on biomass and yield [195]. However, the effect of water stress is strongly influenced by the time scale and phenological stage of the plant cycle [14]. In this study, differences were observed only at harvest time, which limits the possibility of conducting a more in-depth analysis of the physiological state of the plant at different stress conditions.

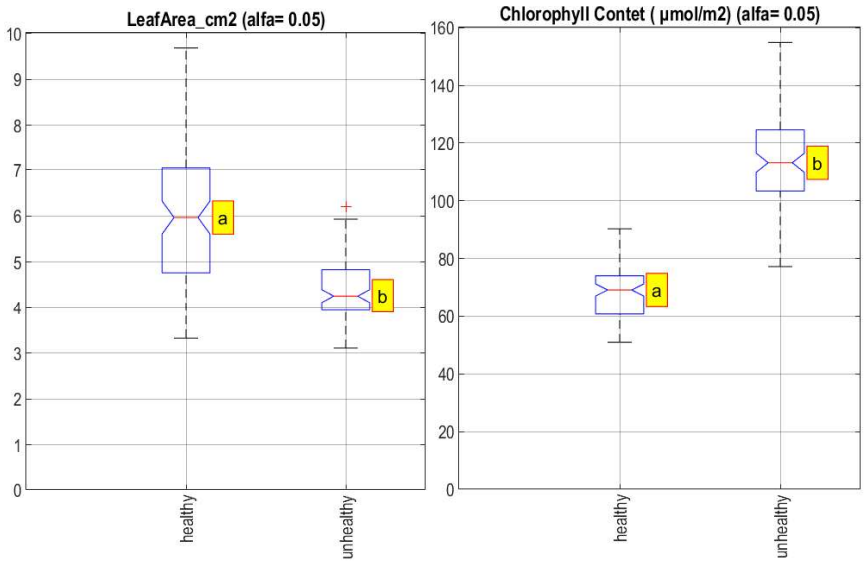


Figure 2.17.Box plot of chlorophyll content (µmol/cm²) and leaf area(cm2) of baby lettuce plants grown in good water condition (healthy) and water deficit condition(unhealthy) with significance (α=0.05).

Moreover, the main goal of testing the application of the MS image alignment model was achieved with satisfactory results. Figure 2.18 demonstrates how the accurate alignment of the MS bands using the CB method effectively distinguished specific leaf areas through the vegetation indices. This provides a promising foundation for future sub-pixel analyses of leaf areas. Although the correlations were not strong enough to propose a predictive model for crop physiological status, the distribution of experimental data from several VIs, such as SR, NARI, GNDVI, and MCARI, clearly differentiates between healthy and unhealthy

plants. This suggests that properly aligned band ratios can effectively distinguish between these two physiological conditions.

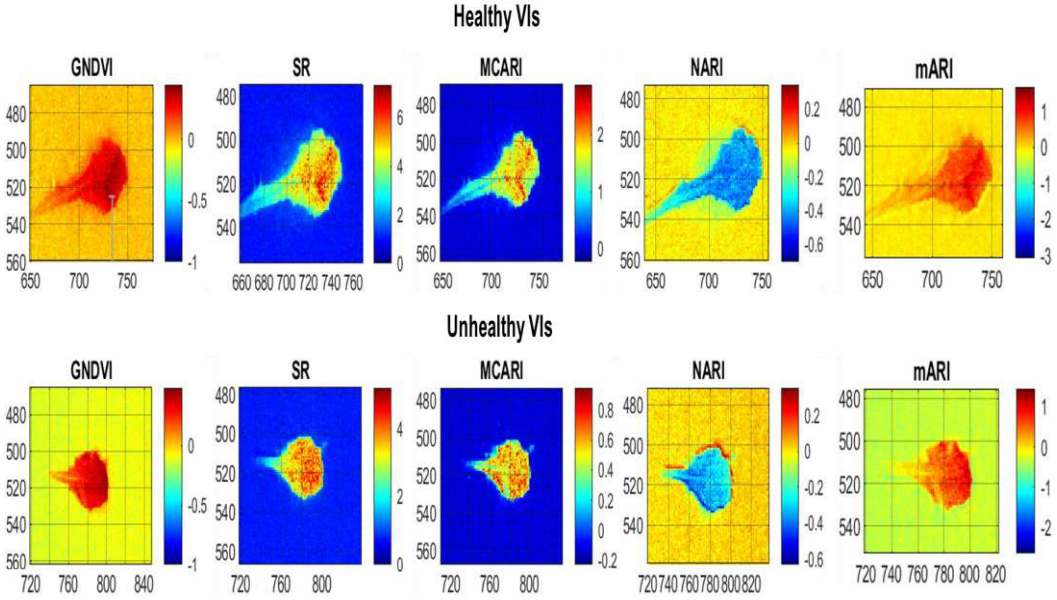


Figure 2.18. Selected vegetation indices (VIs) (GNDVI, SR, MCARI, NARI, and mARI) were applied to baby lettuce leaves as a result of image alignment on healthy and stressed plants.

2.6. Conclusions

Using multispectral (MS) bands (G, B, NIR, and RE) as binocular lens pairs (with the fixed red band) in the stereo vision systems, the image-alignment model showed significant differences in performance depending on the spectral band and the range of distances selected for calibration, ranging from 500 mm to 1500 mm. The stereo vision approach, evaluated by reprojection error (expressed in pixels), demonstrated promising accuracy in the calibration set, with total mean errors of approximately 0.013 pixels for all band pairs. The minimum deviation between recognized homologous points in the two scenes of the stereo system, for the calibration of two or more camera lenses, was less than 1 pixel, a value considered robust in these contexts.

The effect of distance on error was consistent across all pairwise systems used, with maximum values recorded at the extremes of the distance set. However, the behavior of the error along the X- and Y-axes of the image differed. In particular, the reprojection error along the Y-axis had a noticeable impact on our calibration set, especially for the G and RE band pairs. Compared to the B, RE, and NIR bands, the G band exhibited a higher average error along the Y-axis. In contrast, the RE band showed a more stable and less variable mean trend.

The accuracy of the depth (Z) estimation is directly influenced by the reprojection error, which is conceptually related to the calculated disparity in the 3D coordinates of the images in the set [196]. The higher errors observed at distances greater than 1000 mm are likely due to reduced accuracy in disparity estimation and the challenges of identifying similarities between stereo images at larger distances. Indeed, the larger the error, the lower the accuracy in aligning homologous points between the two scenes acquired from the same viewpoint.

As expected, the G and RE bands exhibited opposite behaviors in estimation accuracy, with minimum and maximum errors of 18% and 5% for the G band, and 23% and 10% for the RE band, respectively. This highlights the greater stability of the RE band compared to the G band in estimates. The best results were obtained at smaller distances (800 mm) in the validation set, while a peak in error was observed at the intermediate distance of 1000 mm. In the image alignment model, the tested methods (CB, FT, RG) showed good results in terms of correlation with the experimental data, as highlighted by adjusted R^2 values around 0.99, suggesting strong fitting capability.

Analysis of the residual offset, expressed as the error along the X (X_{err}) and Y (Y_{err}) axes, highlights the G and RE bands as a failed combination. Despite this, the residual offset errors along the X- and Y-axes were found to be in the range of ±2 to ±3 pixels, suggesting that the spatial alignment predicted by the model was still satisfactory for all methods tested.

The proposed image alignment method enhances the accuracy of sub-pixel vegetation index calculation in greenhouse environments where short-range imaging is required. Although the results are not yet sufficiently robust for developing a comprehensive predictive model of plant chlorophyll content, the analysis of vegetation indices revealed a clear distinction between healthy and unhealthy plants. This approach holds significant promise for precision agriculture applications, particularly for the continuous monitoring of baby crop health in controlled environments, contributing to more effective and data-driven greenhouse management strategies.

CHAPTER 3

3. Case Study 2: Hyperspectral imaging for biotic stressdetection in tomato plants

3.1. Introduction

Tomato (*Solanum lycopersicum* L.), family *Solanaceae*, has become in the past fifty years one of the most important and extensively grown horticultural crops in the Mediterranean region and throughout the world. In 2019, more than 180 million tonnes of tomato were produced worldwide, with approximately 42 million tonnes in Mediterranean countries. Due to the profound impact of climate change, the recent revision of approved pesticide lists, and the globalization of seed trade, a further increase in the spread of pathogens is expected. This is particularly concerning given the crop's intrinsically low genetic diversity, which makes it highly susceptible to pathogen attacks.

Late blight (*Phytophthora infestans*) is one of the most common and devastating fungal diseases affecting tomato plants. The disease initially appears as small, irregular, water-soaked lesions on the leaves, which quickly enlarge and turn brown or dark olive-green. Under favorable conditions of high humidity and moderate temperatures, the symptoms can rapidly expand, covering the entire leaf blade and causing leaf wilting, necrosis, and defoliation. White fungal growth may also appear on the underside of the leaves around lesion margins, particularly during moist weather. Conventionally, disease detection on crop leaves has relied on visual inspection and laboratory-based diagnostic methods.

Hyperspectral imaging (HSI) technology stands as a successful paradigm of the integration between visual imaging and spectral techniques. As a powerful tool, HSI has undergone rapid development in agriculture and other fields, offering superior advantages over conventional spectral techniques. HSI technology demonstrates significant potential for the early diagnosis of tomato diseases. Its core advantage lies in the ability to detect physiological and biochemical abnormalities and their corresponding changes in reflectance spectral features induced during the initial stages of pathogen infection, which are not yet discernible to the naked eye.

This chapter investigates the potential of HSI for the early assessment of plant disease status.

The aims are: I) To predict spectral changes in tomato plants with the content of key photosynthetic pigments; II) To evaluate the relationship between pigments (such as chlorophyll, and phenol content) and several vegetation indices (VIs); III) Classify healthy and diseased plants, as well as discriminate between several stages of disease progression, based on their spectral signatures.

3.2. Materials and methods

3.2.1. Hyperspectral imaging system

A visible and near-infrared hyperspectral imaging (HSI) system was used (Figure 3.1), based on the HyS ULTRIS S 5 camera (Cubert GmbH, Science Park II, Lise-Meitner Straße 8/1, D-89081 Ulm, Germany). The camera operates over a spectral range of 450–850 nm, capturing data across 51 discrete spectral bands. With a spectral resolution characterized by a full width at half maximum (FWHM) of 26 nm at 532 nm, the system enables accurate spectral discrimination and material analysis. The maximum spatial resolution of the camera is 290 × 275 pixels, ensuring consistent image quality across all bands. The system utilizes a snapshot sensor configuration, meaning that the sensor remains stationary while the sample is moved beneath the field of view. A complete hyperspectral image (hypercube) is acquired in a single capture during the scan of the sample. All data acquisition and preprocessing were carried out using MATLAB R2023b (The MathWorks, Inc., Natick, MA, USA).

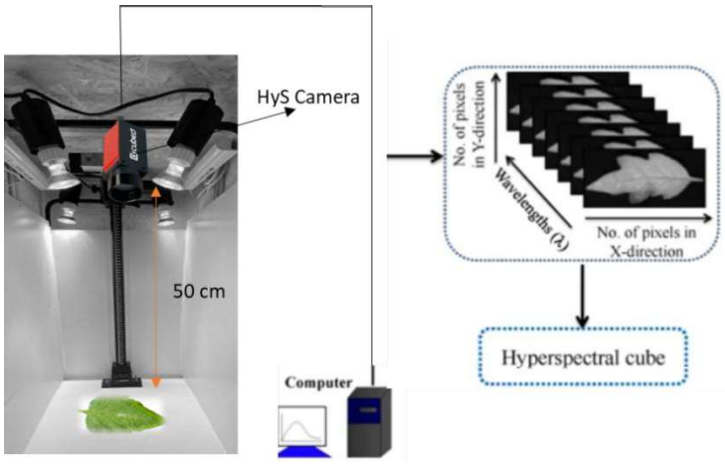


Figure3.1. Schematic diagram of the hyperspectral imaging system.

3.2.2. Experimental design: Plant materials

The study focused on characterizing the effects of late blight (*Phytophthora infestans (LB)*) on tomato (*Solanum lycopersicum L., cv. Trebian F2*). The experimental activities were carried out in a greenhouse located in Scalea (CS), Calabria, Italy (Figure 3.2).

Tomato plants were subjected to three different levels of fungicide treatment (hereafter referred to as "Dose"), applied in accordance with the guidelines of the Integrated Pest Management (IPM) protocol of the Calabria Region. All plants were cultivated under uniform agronomic conditions, following the standards established by the regional Integrated Production Protocol. Natural disease development was allowed, and the evolution of symptom expression was monitored. The applied treatment levels were as follows:

- Dose 0: No treatment (control).
- Dose 0.5: Medium dose (50% of the standard dose).
- Dose 1: Standard dose.

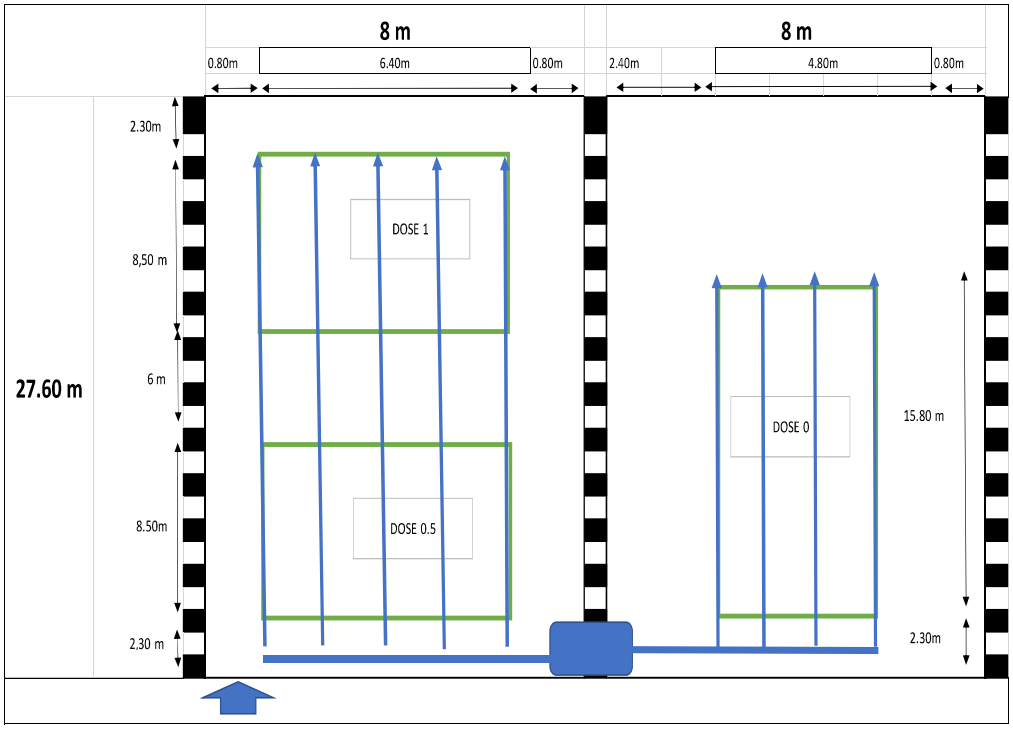


Figure 3.2. Experimental greenhouse layout.

Farm technicians conducted weekly randomized assessments of plant health status across the three treatment areas to track disease progression. Upon the first positive detection of pathogen development at the microscopic level and the appearance of early visual

symptoms (pre-symptomatic leaf indicators), a tri-weekly sampling protocol was initiated to collect representative samples of the three disease development stages: healthy (treatment-protected), pre-symptomatic, and symptomatic plants. The sampling was aligned with the natural progression of the disease under field conditions.

The experimental design included five replicates for each dose, with 5 plants × 3 doses × 5 replicates = 75 plants per treatment. Each treatment group was duplicated.

3.2.3. Statistical analysis

Statistical differences among treatments were evaluated using the Kruskal–Wallis test and a two-way ANOVA, considering time and plant health status (healthy vs. non-healthy) as factors. Statistical significance was set at $p \leq 0.05$.

3.2.4. Biochemical and physiological assessments of the impact of LB infection

The chlorophyll content of each leaf was measured at 3-day intervals, and the total polyphenol content every 6 days. Both parameters are widely reported in the literature as sensitive indicators for assessing physiological stress in plants[197].

3.2.5. Chlorophyll Content

Chlorophyll concentration (μmol/m²) was measured using an Apogee MC-100 Portable Chlorophyll Meter (Apogee Instruments, Inc., 721 West 1800 North, Logan, Utah, 84321, USA), with a cylindrical sampling area of Ø 20 mm.

3.2.6. Phenolic Compounds

The total phenolic content (TPC) was determined following the method described by [198], with some modifications. To optimize phenolic extraction, two solvents, ethanol and methanol (at 70% and 80% concentrations, respectively), and two extraction methods, maceration (MAC) and ultrasound (US) bath, were evaluated.

Leaf tissue was extracted using a double cylindrical punch (Ø 20 mm), ensuring a representative fresh weight between 0.1 and 0.3 g. In the maceration method, the tissue was placed in 5 ml of solvent in a 50 ml glass Falcon tube and magnetically stirred for 1 hour. In

the ultrasound method, the tissue was suspended in 5 ml of solvent inside a 15 ml plastic Falcon tube and placed in a 37 °C ultrasound bath for 1 hour.

The selected method for final analysis was ultrasound extraction using 80% methanol (Figure 3.3).

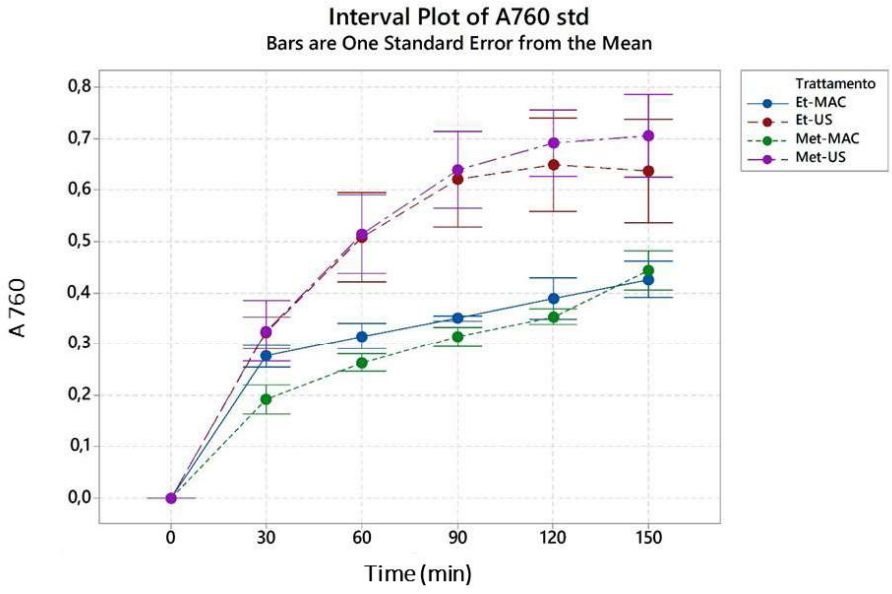


Figure 3.3.Variation of absorbance vaue at 760 nm (A760) over time for different extraction treatments of total phenolic compounds from tomato leaf tissue. Four methods were compared: maceration with ethanol (Et-MAC), ultrasound with ethanol (Et-US), maceration with methanol (Met-MAC), and ultrasound with methanol (Met-US).

Briefly, a fresh leaf sample (0.1–0.3 g) was suspended in 80% methanol and treated in an ultrasound bath at 37 °C for 90 minutes. The resulting methanolic extract was then mixed with Folin–Ciocalteu reagent (1:10) and 20% sodium carbonate (Na_2CO_3), left at room temperature for 2 hours. The absorbance was measured at 760 nm, and the total phenolic content was calculated using a gallic acid standard curve, expressed as mg/g of fresh weight (FW).

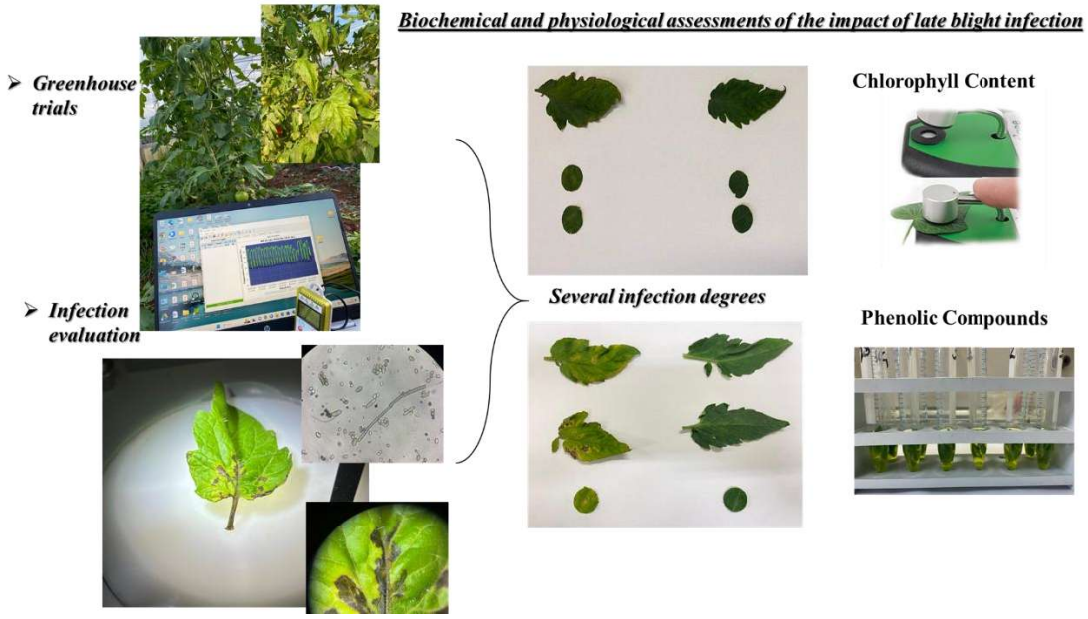


Figure 3.4.Biochemical and physiological assessments of the impact of Late blight (Phytophthora infestans) infection on tomato leaves.

3.2.7. Flow of the study

The hyperspectral images of healthy and diseased tomato leaves were obtained by the hyperspectral imaging system in the spectral wavelength range of 450–850 nm. Then, the raw hyperspectral images were analyzed in "Image Processing Toolbox" in MATLAB.

The reflectance values of all pixels in the region of interest (ROI) were extracted and treated as X variables. All samples were divided into training and testing sets as 80:20 percent.

3.2.8. Spectral preprocessing:

Standard Normal Variate (SNV): Standardization and normalization are essential preprocessing steps for mitigating non-target variations in hyperspectral data. These procedures aim to remove systematic spectral intensity shifts caused by factors such as illumination variability, sensor sensitivity differences, sample surface heterogeneity, and inconsistent acquisition conditions. The core objective is to ensure the intrinsic comparability of spectral data acquired from tomato samples across different batches, time points, or instruments—thereby establishing a reliable foundation for the development of robust and generalizable predictive models. SNV performs both scatter correction and standardization by centering and scaling each spectrum individually. This approach

effectively corrects for variations due to pathlength differences and light scattering, enhancing spectral contrast and consistency among tomato samples**.**

Noise suppression and signal enhancement: Noise suppression and signal enhancement represent critical aspects of hyperspectral preprocessing, addressing both instrumental noise and environmental interferences present in raw spectral data. The primary goal is to improve the signal-to-noise ratio (SNR) while preserving key spectral information that reflects the biochemical state of the samples. One commonly applied technique is *Savitzky–Golay (SG) filtering***,** which smooths high-frequency random noise and corrects digitization artifacts without distorting essential spectral features. This enhances the reliability and accuracy of subsequent multivariate analyses.

3.2.9. Feature selection

To reduce model complexity and improve predictive performance, two feature selection techniques were applied: the Coefficient of Variation Algorithm (CVA) and the Variable Importance in Projection (VIP) method.

CVA was implemented through an iterative process that progressively removed wavelengths exhibiting the highest relative variability across the spectral dataset. This unsupervised method, combined with 10-fold cross-validation, enhanced model performance and generalizability by filtering out noisy or redundant spectral features, leading to robust PLS regression models with RPD values up to ≥ 5.3 [198].

Simultaneously, VIP scores, derived from PLS model weights, were used to rank and retain wavelengths with the most significant contribution to the latent variables explaining the response.

3.2.10. Image analysis: Vegetation indices

From each image acquired, an area of interest (ROI) was selected using automatic segmentation based on the Otsu method, in order to isolate the leaf portion useful for analysis. The vegetation indices (VI) shown in Table 3.1 were calculated for each ROI, obtained as the average of the spectral values of the entire area selected for each image.

Table 3.1. The Vegetation Indices (VI evaluated in this study

Indices	Formula	Theoretical bands from MSI	Bands used from HyS	Indicators
Normalized Difference Vegetation Index (NDVI)	(R_NIR - R_Red) / (R_NIR + R_Red)	800, 670	802, 682	Chlorophyll content, plant vigor
Enhanced Vegetation Index (EVI)	2.5*(R_NIR - R_Red)/(R_NIR + 6*R_Red - 7.5*R_Blue + 1)	800, 670, 450	802, 682, 450	Vegetation vigor in high-background or bright-soil conditions
Photochemical Reflectance Index (PRI)	(R_531 - R_570)/(R_531 + R_570)	531, 570	530, 570	Photosynthetic efficiency (xanthophyll cycle activity)
Simple Ratio (SR)	R_NIR / R_Red	800, 670	802, 682	BiomassaBiomass and canopy density
Modified Chlorophyll Absorption Ratio Index (MCARI)	[(R_700 - R_670) - 0.2*(R_700 - R_550)] * (R_700 / R_670)	700, 670, 550	698, 674, 554	Chlorophyll content and leaf structure degradation
Green Normalized Difference Vegetation Index (GNDVI)	(R_NIR - R_Green)/(R_NIR + R_Green)	800, 550	802, 554	Chlorophyll content and photosynthetic activity
Red Normalized Difference Vegetation Index (RNDVI)	(R_800 - R_700)/(R_800 + R_700)	800, 700	NIR, 700	Vegetation stress and canopy physiological status
Normalized Difference Anthocyanin Index (NDAI)	(R_Red - R_Green)/(R_Red + R_Green)	670, 550	674, 550	Anthocyanin content (stress and protective pigments)
Normalized Anthocyanin Reflectance Index (NARI)	(1/R_550 - 1/R_700)/(1/R_550 + 1/R_700)	550, 700	550, 698	Anthocyanin accumulation
Anthocyanin Reflectance Index (ARI)	1/R_550 - 1/R_700	550, 700	550, 698	Anthocyanin content and stress signaling
Modified Anthocyanin Reflectance Index (mARI)	(1/R_530-570 - 1/R_690-710) * R_>760	530–570, 690–710, >760	530, 570, 690, 698, 706, 770–810	Enhanced detection of anthocyanins (modified anthocyanin index)

For the calculation of vegetation indices (VIs), only the spectral bands corresponding to those of the multispectral camera used in the previous chapter (Section XXX) were extracted from the hyperspectral signature, in order to ensure compatibility between datasets and allow for an integrated extension of the multispectral analysis.

The selected VIs were evaluated based on their temporal evolution in both infected (UN) and healthy (H) plants, with the aim of identifying characteristic patterns of stress response. Furthermore, to assess their ability to describe leaf physiological parameters, the VIs were correlated using the Pearson coefficient with chlorophyll pigments and total phenolic content (TPC).

3.2.11. Chlorophyll and phenols regression modeling

To predict chlorophyll and total phenolic contents from spectral data, regression models were developed using Partial Least Squares Regression (PLSR).

PLSR is particularly well-suited for analyzing highly collinear and high-dimensional datasets, such as those generated by spectroscopy, as it projects the predictor variables (spectral features) into a new latent space that maximizes covariance with the response variable (chemical properties). To optimize predictive performance, the interaction between different spectral preprocessing techniques and variable selection strategies was also evaluated.

The performance of the spectral preprocessing and feature extraction combinations for the models was evaluated using 3-fold cross-validation and 100 Monte Carlo (MC) runs in both validation and external prediction. Model performance was expressed in terms of the coefficient of determination (R2), Root Mean Square Error (RMSE), and Ratio of Performance to Deviation (RPD), as shown in Eq. (3.1), (3.2), and (3.3):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (3.1)$$

$$RMSE = \sqrt{1\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \quad (3.2)$$

$$RPD = \frac{SD}{RMSE} \quad (3.3)$$

where y_i is the measured value, $\hat{y}_i$ is the predicted value, $\bar{y}$ is the mean of predicted values, n is the number of samples, and SD_y is the standard deviation of measured values.

3.2.12. Machine Learning for disease stages classification

Binary classification involved differentiating healthy (class 0) and diseased (class 1) plants. Four supervised algorithms were tested: Linear Discriminant Analysis (LDA), Partial Least Squares Discriminant Analysis (PLS-DA), Random Forest (RF), and Support Vector Machine (SVM). All models were implemented and optimized in MATLAB using Bayesian optimization to identify the best combination of hyperparameters for each classifier.

In LDA and PLS-DA, the number of latent variables was optimized. These represent the dimensionality of the latent space used to maximize class separation while reducing data complexity. For SVM, the radial basis function (RBF) was selected as the kernel function, with optimization of: the penalty parameter (C), which balances model complexity and

misclassification penalty; and the Kernel Scale, which controls the width of the RBF function and thus affects the flexibility of the decision boundary. To control ensemble stability and generalization, and to manage model complexity and the risk of overfitting, both the number of trees and the maximum tree depth were optimized for the RF model.

To further refine disease monitoring, a multiclass classification scheme was applied, where plants were categorized as healthy (0), pre-symptomatic (1), intervention threshold (2), and damage threshold (3), based on visual scoring and biochemical markers. Three classifiers, SVM, RF, and Naive Bayes (NB), were evaluated. For NB, the modeling assumes conditional independence among predictors. Bayesian optimization was used to tune the following hyperparameters: Distribution Type (e.g., normal or kernel), which determines the distributional model; Kernel Function (e.g., normal, box, triangle, epanechnikov), which shapes the density estimation; and Kernel Width, which controls the smoothing factor, thus affecting model generalization and the risk of overfitting.

Classification model performance was assessed through 5-fold cross-validation using the following parameters.

Accuracy is the proportion of correctly classified instances among the total number of samples and is defined as:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \qquad (3.4)$$

where:TP: True Positives,TN: True Negatives,FP: False Positives FN: False Negatives.

The F1-score is the harmonic mean of precision and recall and is especially useful for imbalanced datasets. It is calculated as:

$$F1 - score = 2\ x\ \frac{Precision\ x\ Recall}{Precision + Recall} \qquad (3.5)$$

The F1-score balances the trade-off between the model's ability to correctly identify positive cases (recall) and its tendency to avoid false positives (precision), where:

$$Precision = \frac{TP}{TP + FP'} \qquad Recall = \frac{TP}{TP + FN} \qquad (3.6)$$

3.3. Results and discussion

3.3.1. Temporal evolution of leaf pigments, phenolic compounds, and spectral indices during the progression of Phytophthora infestans infection

Due to suboptimal pedoclimatic conditions, the infection by Phytophthora infestans did not develop uniformly across the different fungicide doses; therefore, it was not possible to evaluate the dose effect. The analysis focused instead on the differences between healthy and infected plants over time, with particular attention to leaf pigments and phenolic compounds.

Figure 3.5 shows the mean trends for each sampling time point for total chlorophyll content (Chl) and total phenolic content (TPC).

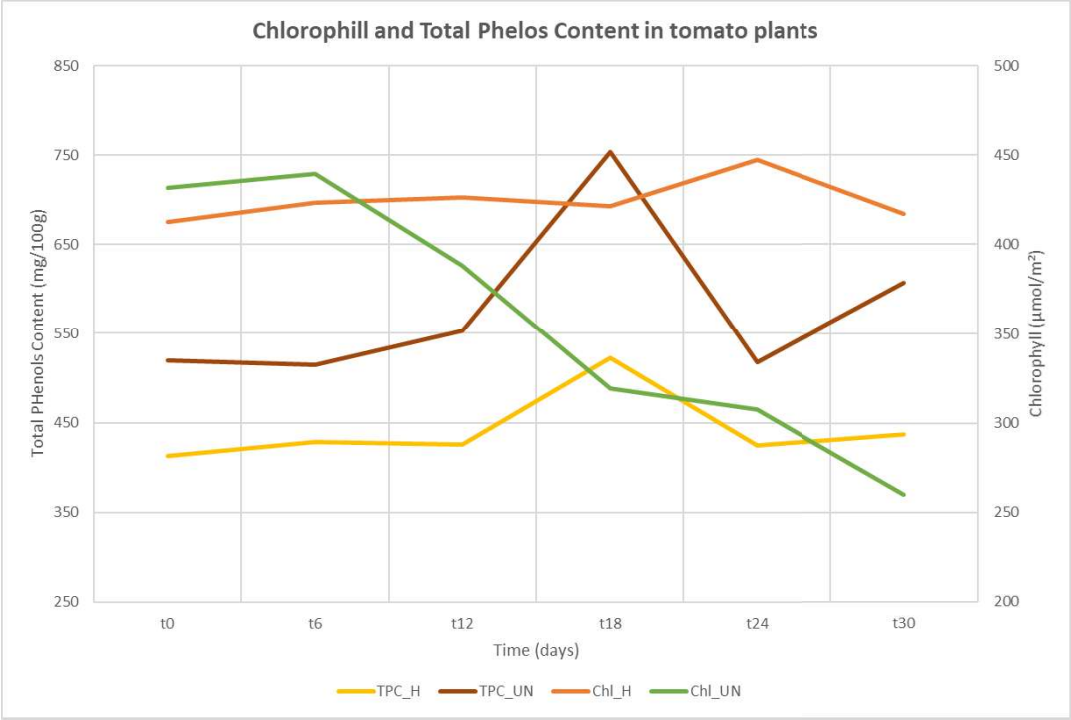


Figure 3.5.Trends in total chlorophyll (Chl) content and total phenolic compound (TPC) content in healthy and unhealthy tomato plants infected with Phytophthora infestans during the experimental period (t0–t30).).

The results indicate that TPC increases significantly over time under both experimental conditions, but with a markedly stronger intensity in infected plants. As highlighted in Table 3.2, significant differences in infected plants are already evident starting from t12, with progressively higher values compared to the initial phases and a highly significant peak at t18 (754 mg/100g). This trend is also reflected in the graph, which shows a marked rise in the TPC_UN curve during the central phases of the experiment. The increase in TPC in infected plants aligns with the well-known defensive role of phenolic compounds, which are typically activated during plant–pathogen interactions as part of the oxidative response and reinforcement of cell wall barriers. The Standard Error of the Mean (SEM)

intervals in the table further confirm a greater physiological heterogeneity induced by the disease.

Chlorophyll analysis reveals a trend opposite to that of phenols. Healthy plants maintain relatively stable values throughout the period, as shown by both the table and the graph, with a slight but significant increase at t24 (447 µmol/m²). In contrast, infected plants show a progressive and significant reduction starting from t12, with a marked and statistically significant decline at t18 and t24, reaching a minimum at t30 (260 µmol/m²). The graph visually confirms this trend, showing a sharp drop in the Chl_UN curve during the later stages. The significant chlorophyll reduction in infected plants can be attributed to progressive damage to the leaf mesophyll and the consequent impairment of photosynthetic efficiency, typical of Phytophthora infestans pathogenesis.

However, the table results shows that differences between healthy and infected plants are consistently significant for TPC (A vs. B at all time points), whereas for chlorophyll they only become significant after t18, when tissue damage becomes more pronounced. This indicates that phenolic content is an early marker of biotic stress response, while chlorophyll reduction represents a later physiological effect, also confirmed in the graph by the inversion of the chlorophyll curves starting from the central phases of the experiment.

Table 3.2. Mean values (+/- Standard Error of the Mean (SEM)) of tomato quality parameters in relation to time (from t0 to t 30) The parameters analyzed include Total Phenol Content (TPC) and Chlorophyll Content. Lowercase letters indicate statistically significant differences between times ($p < 0.05$), while uppercase letters indicate significant differences between infected (unhealthy) and uninfected (healthy) theses ($p < 0.05$).

MEAN(+/- SDOM)	Parameter	T0	T6	T12	T18	T24	T30
TPC	healthy	413.36 (+/- 22.6862) aA	428.11 (+/- 19.6834) aA	425.549 (+/- 19.7275) abA	522.958 (+/- 22.7605) bA	424.765 (+/- 15.4042) abA	437.548 (+/- 29.5747) aA
	unhealthy	519.598 (+/- 167.835) aB	514.636 (+/- 26.1995) abB	553.194 (+/- 18.324) bcB	754.047 (+/- 46.5177) cB	517.407 (+/- 25.4534) abB	606.733 (+/- 27.4374) bcB
Chlorophyll	healthy	412.456 (+/- 5.8811) aA	423.406 (+/- 5.88674) abA	426.489 (+/- 7.27117) abA	421.617 (+/- 9.6208) abA	447.317 (+/- 8.41711) bA	417.164 (+/- 9.78379) abA
	unhealthy	431.744 (+/- 8.1284) aA	439.794 (+/- 6.75201) aA	388.039 (+/- 14.3873) aA	319.117 (+/- 13.0269) bB	307.506 (+/- 12.8262) bB	260.094 (+/- 12.6887) bB

The analysis of spectral indices (Table 3.3) provides a progressive representation of stress evolution in infected plants. The first signs of physiological alteration emerge between t12 and t18, a phase in which a reduction in NDVI and GNDVI values is observed, accompanied by a gradual decline in PRI. The latter indicates a decrease in photosynthetic efficiency and an increase in non-photochemical stress dynamics. Between t18 and t30, stress reaches its highest intensity. During this period, the decrease in indices related to leaf structure and chlorophyll content (NDVI, GNDVI, SR) becomes particularly pronounced.

At the same time, there is a marked increase in indices associated with non-photosynthetic pigments and defense responses, such as ARI, NARI, and MCARI. This trend reflects the transition from an initial stress condition to an advanced stage of damage, characterized by chlorophyll degradation, mesophyll impairment, and activation of secondary metabolites with protective functions.

Table 3.3. Mean values (± SD) of the main spectral vegetation indices (VIs) measured in leaves of infected plants (UN) at different experimental time points.

Time-Class	NDVI	EVI	PRI	SR	MCARI	GNDVI	RNDVI	NDAI	NARI	ARI	mARI
t0-UN	0.472 (0.004)	2.145(0.214)	0.004 (0.001)	3.015 (0.026)	1.133 (0.032)	0.447 (0.004)	0.383 (0.004)	-0.097 (0.002)	0.043 (0.002)	2.542 (0.106)	0.521 (0.012)
t3-UN	0.487 (0.003)	2.147 (0.248)	-0.003 (0.001)	3.129 (0.021)	1.115 (0.028)	0.468 (0.004)	0.395 (0.003)	-0.089 (0.002)	0.055 (0.001)	3.413 (0.11)	0.619 (0.012)
t6-UN	0.505 (0.003)	2.498 (0.218)	0.006 (0.001)	3.169 (0.022)	1.255 (0.03)	0.481 (0.003)	0.408 (0.003)	-0.103 (0.002)	0.045 (0.002)	2.827 (0.117)	0.609 (0.013)
t9-UN	0.503 (0.005)	2.451 (0.272)	0.003 (0.001)	3.172 (0.039)	1.28 (0.037)	0.476 (0.006)	0.406 (0.006)	-0.108 (0.002)	0.044 (0.002)	2.757 (0.108)	0.593 (0.014)
t12-UN	0.47 (0.005)	1.749 (0.184)	-0.002 (0.001)	2.984 (0.034)	1.368 (0.054)	0.448 (0.006)	0.368 (0.006)	-0.1 (0.003)	0.058 (0.002)	3.325 (0.139)	0.585 (0.016)
t15-UN	0.483 (0.004)	2.442 (0.234)	0.003 (0.001)	3.007 (0.025)	1.37 (0.047)	0.464 (0.004)	0.382 (0.004)	-0.096 (0.002)	0.06 (0.002)	3.494 (0.1)	0.606 (0.012)
t18-UN	0.445 (0.005)	2.402 (0.271)	0 (0.001)	2.814 (0.033)	1.376 (0.066)	0.42 (0.005)	0.344 (0.006)	-0.102 (0.003)	0.051 (0.003)	2.73 (0.158)	0.501 (0.014)
t24-UN	0.476 (0.004)	2.362 (0.188)	-0.007 (0.001)	2.953 (0.028)	1.622 (0.045)	0.45 (0.005)	0.364 (0.005)	-0.116 (0.003)	0.056 (0.003)	3.129 (0.141)	0.579 (0.013)
t27-UN	0.445 (0.005)	1.877 (0.249)	-0.006 (0.001)	2.816 (0.029)	1.453 (0.065)	0.426 (0.005)	0.34 (0.005)	-0.098 (0.004)	0.065 (0.002)	3.537 (0.149)	0.566 (0.016)
t30-UN	0.48 (0.003)	1.984 (0.321)	-0.008 (0.001)	2.96 (0.024)	1.676 (0.069)	0.454 (0.003)	0.364 (0.004)	-0.118 (0.005)	0.061 (0.002)	3.422 (0.114)	0.588 (0.01)

Unlike what was observed in infected plants, healthy plants exhibited a remarkably stable spectral behavior throughout the experiment, indicative of a good physiological state and the absence of biotic stress signals (Table 3.4). Indices reflecting vegetative vigor and chlorophyll content, such as NDVI, GNDVI, and RNDVI, showed a regular trend, with a gradual increase during the initial stages (t0–t6) and consistently high values in the later phases.

Similarly, indices describing leaf structure and the balance of photosynthetic pigments, such as SR and MCARI, did not exhibit critical fluctuations or sudden increases. The slight variations observed fall within the physiological phenological development of the

plant and do not indicate chlorophyll degradation or alterations in cellular structure, behavior clearly detected in infected plants during the advanced stages of infection.

Indices related to secondary pigments, such as ARI, NARI, and mARI, also showed a moderate increase in the final stages, consistent with normal foliar metabolism and lacking the anomalous peaks seen in infected plants between t18 and t30, which reflected a defensive response to disease. Likewise, as confirmed by tabular values, early stress indices such as PRI and NDAI remained overall stable, with no marked deviations,

Infected plants (UN) showed clear physiological deterioration, with a pronounced decrease in indices related to chlorophyll and leaf structure, alongside a parallel increase in indices associated with secondary pigments and defense-related processes. In contrast, healthy plants (H) maintained a stable spectral profile throughout the observation period, consistent with pigment and phenol trends. As also reported for pigments and phenolic compounds, the most pronounced divergence between healthy and infected plants becomes evident as early as t12–t18, confirming that spectral indices are sensitive and reliable tools for the early detection of plant health status and for monitoring disease progression.

Table 3.4. Mean values (± SD) of the main spectral vegetation indices (VI) measured on leaves of healthy plants (H) at different experimental time points.

Time-Class	NDVI	EVI	PRI	SR	MCARI	GNDVI	RNDVI	NDAI	NARI	ARI	mARI
t0-H	0.458 (0.004)	1.909 (0.161)	0.003 (0.001)	2.901 (0.022)	1.146 (0.016)	0.429 (0.004)	0.368 (0.003)	-0.103 (0.002)	0.036 (0.001)	2.041 (0.067)	0.461 (0.009)
t3-H	0.475 (0.004)	2.343 (0.195)	-0.003 (0.001)	3.013 (0.029)	1.174 (0.028)	0.449 (0.005)	0.38 (0.004)	-0.102 (0.003)	0.044 (0.002)	2.682 (0.134)	0.535 (0.016)
t6-H	0.492 (0.003)	2.458 (0.222)	0.003 (0.001)	3.053 (0.021)	1.259 (0.032)	0.466 (0.003)	0.393 (0.003)	-0.108 (0.003)	0.045 (0.002)	2.793 (0.104)	0.554 (0.011)
t9-H	0.487 (0.004)	1.977 (0.231)	0.003 (0.001)	3.03 (0.03)	1.275 (0.029)	0.457 (0.005)	0.39 (0.004)	-0.111 (0.002)	0.038 (0.002)	2.376 (0.116)	0.524 (0.015)
t12-H	0.465 (0.004)	1.986 (0.21)	-0.001 (0.001)	2.927 (0.026)	1.185 (0.029)	0.443 (0.004)	0.37 (0.004)	-0.096 (0.002)	0.05 (0.001)	2.979 (0.112)	0.538 (0.015)
t15-H	0.485 (0.003)	2.312 (0.206)	0.005 (0.001)	3.002 (0.025)	1.226 (0.027)	0.466 (0.004)	0.388 (0.003)	-0.096 (0.003)	0.054 (0.001)	3.35 (0.107)	0.582 (0.014)
t18-H	0.456 (0.004)	2.015 (0.275)	0 (0.001)	2.882 (0.029)	1.022 (0.02)	0.438 (0.005)	0.37 (0.004)	-0.082 (0.002)	0.05 (0.002)	3.014 (0.126)	0.526 (0.016)
t24-H	0.505 (0.003)	2.015 (0.165)	-0.002 (0.001)	3.169 (0.022)	1.159 (0.029)	0.486 (0.003)	0.41 (0.003)	-0.094 (0.003)	0.054 (0.001)	3.518 (0.105)	0.65 (0.013)
t27-H	0.476 (0.003)	1.86 (0.182)	0 (0)	3.031 (0.024)	1.009 (0.022)	0.463 (0.004)	0.39 (0.003)	-0.075 (0.002)	0.058 (0.001)	3.581 (0.09)	0.623 (0.012)
t30-H	0.511 (0.003)	1.928 (0.219)	-0.003 (0.001)	3.225 (0.023)	1.028 (0.028)	0.497 (0.003)	0.421 (0.002)	-0.081 (0.003)	0.059 (0.001)	4.015 (0.108)	0.689 (0.013)

Based on the analysis of temporal trends of vegetation indices (VIs) in healthy and infected plants, certain indices exhibit particularly marked and physiologically interpretable dynamics. Specifically, GNDVI proves to be a sensitive indicator of vegetative vigor and baseline chlorophyll content; MCARI is strongly responsive to chlorophyll degradation and structural changes in the mesophyll; finally, mARI and ARI consistently reflect the activation of secondary pigments associated with plant defense responses. These indices were selected for the visual representation of stress dynamics, shown in Figure 3.6, which displays the leaf-level distribution of the selected VIs across the experimental time points, with their characteristics detailed in the following chapter. In infected plants (Figure 3.6 a), a clear spatial and temporal heterogeneity of spectral values is observed, with irregular patterns that reflect the progression of physiological damage. In contrast, healthy plants (Figure 3.6 b) show stable and homogeneous VI distributions over time. This divergence highlights the sensitivity of the selected indices in detecting early physiological alterations, starting as early as t12, when no visible symptoms are yet detectable at the macroscopic level. These findings underscore the usefulness of these indices for the presymptomatic detection of biotic stress.

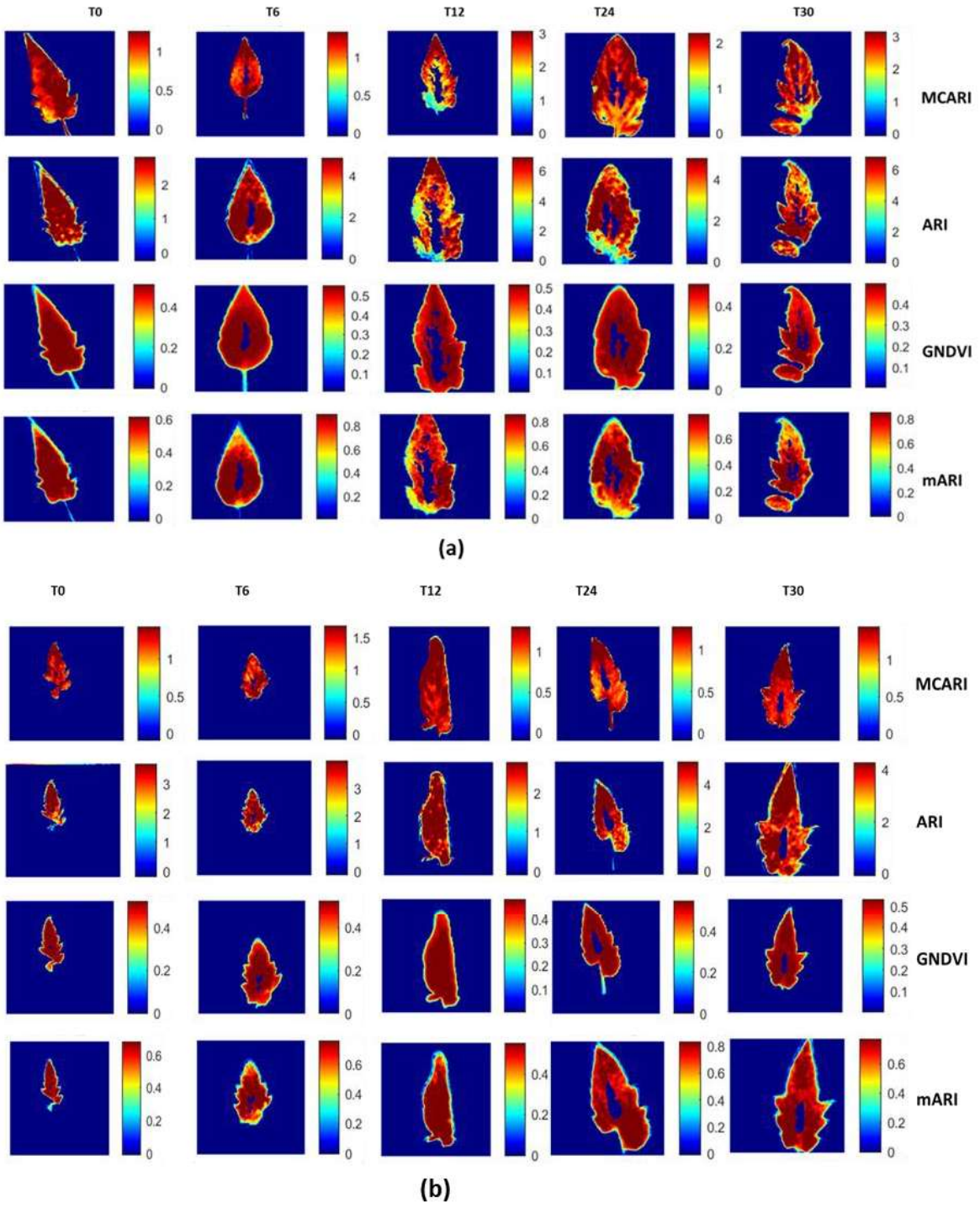


Figure 3.6. Spatial distribution of selected spectral vegetation indices (GNDVI, MCARI, ARI, and mARI) in infected (a) and healthy (b) leaves across the experimental time points.

However, to further explore the potential of spectral vegetation indices (VI), the correlation between chlorophyll and VIs (Figure 3.7), and between TPC and Vis (Figure 3.8), was analyzed without distinguishing between healthy and infected plants. The goal was to identify physiologically meaningful relationships that could be useful for estimating variations in leaf pigments and phenolic compounds related to oxidative stress based solely

on spectral responses. The correlogram reveals that several VIs exhibit strong positive correlations with each other (e.g., NDVI, GNDVI, RNDVI, SR), confirming their similar response to changes in leaf structure and chlorophyll content. In particular, chlorophyll shows notable correlations with indices such as NDAI (r = 0.53), RNDVI (r = 0.54), and MCARI (r = –0.66), suggesting that these indicators are especially sensitive to variations in photosynthetic pigments. However, none of these values exceed high correlation thresholds ($|r| \geq 0.80$), indicating a limitation in the ability of individual VIs to accurately predict chlorophyll content**.**

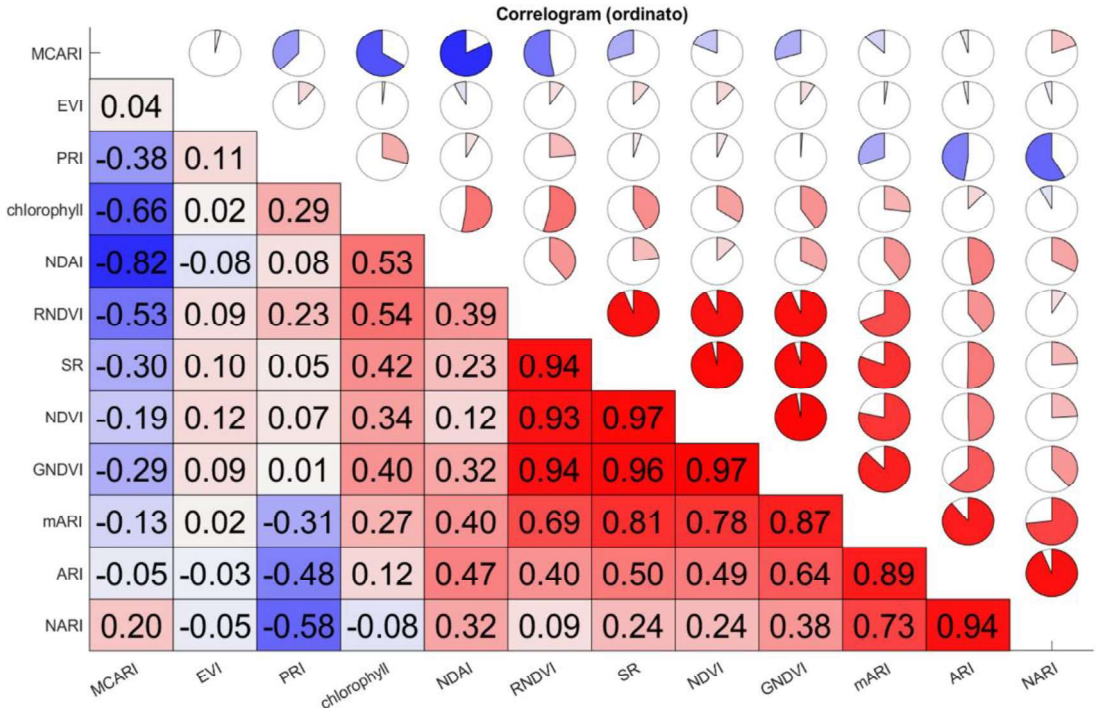


Figure 3.7. Correlogram showing the Pearson correlation coefficients (r) between spectral vegetation indices (VIs) and chlorophyll content values. The matrix is sorted in descending order of correlation coefficients, facilitating the identification of the strongest associations. The color map visually represents both the strength and direction of the correlations.

Regarding the total phenolic content, the correlations are generally weaker, but some interesting relationships emerge, particularly with MCARI (r = 0.32) and NARI (r = 0.14). These indices, associated respectively with variations in photosynthetic and secondary pigments, may serve as potential predictors of phenolic response, especially during the later stages of stress. However, the correlation values remain low ($|r| < 0.40$), highlighting a limitation in the ability of VIs to directly reflect changes in TPC**.**

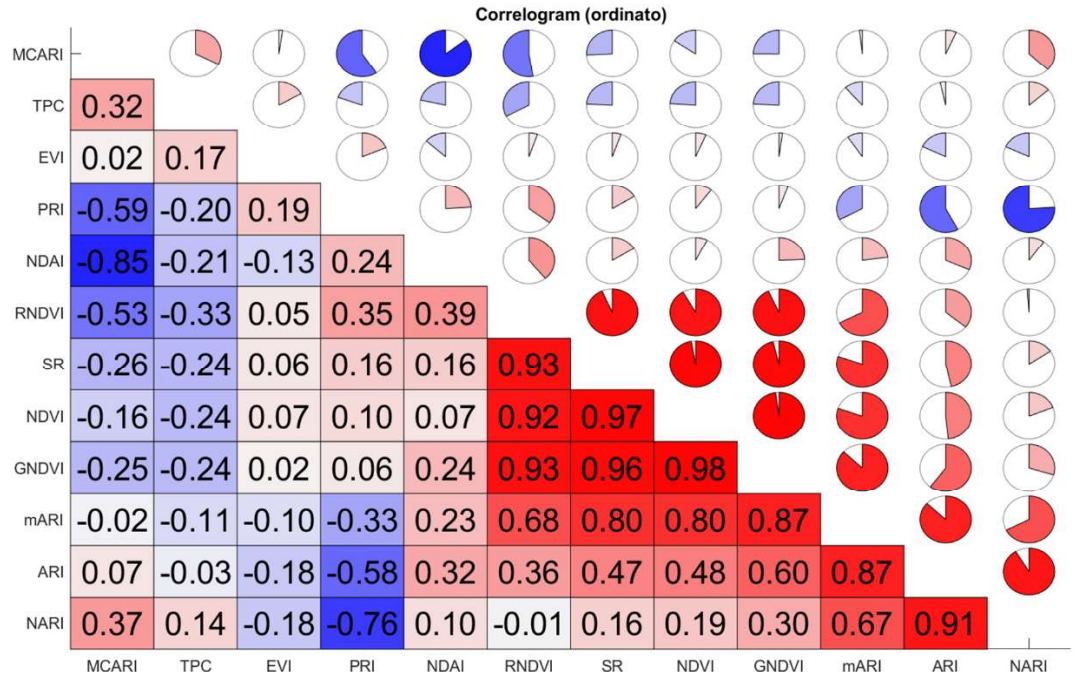


Figure 3.8.. Correlogram showing the Pearson correlation coefficients (r) between spectral vegetation indices (VIs) and Total Phenols Content (TPC). The matrix is sorted in descending order of correlation coefficients, facilitating the identification of the strongest associations. The color map visually represents both the strength and direction of the correlations.

3.3.2. Prediction of the chlorophyll and phenol content

Figure 3.9, shows the reflectance spectra of healthy and infected leaves following the application of different preprocessing techniques (RAW, SNV, SG1, SG2, and their combinations). In all representations, the spectra of infected leaves (in red) exhibit greater variability compared to those of healthy leaves (in blue), reflecting the physiological and structural heterogeneity induced by the pathogen.

The main differences are observed in three key spectral regions: 520–580 nm, which is sensitive to secondary pigments (e.g., anthocyanins) and oxidative stress; 650–680 nm, associated with chlorophyll a absorption and its degradation; and 700–740 nm, where deformation of the red-edge reflects mesophyll damage and reduced photosynthetic efficiency. Derivative preprocessing (SG2) enhances these differences by reducing spectral noise and amplifying variations in the most physiologically relevant regions, improving spectral separability between healthy and infected samples.

Several studies have confirmed that pathogen-infected leaves exhibit significantly different spectral profiles compared to healthy ones, in line with our observations [[199,200]. The most characteristic alterations are observed in three spectral bands:

- Green (~520–580 nm): this region is affected by the presence of secondary pigments, such as anthocyanins, which increase in response to biotic stress. Infected leaves

often show anomalous reflectance in the green range. For example, in citrus plants affected by *Huanglongbing (HLB*), significant differences were observed at early stages around 520–580 nm [201].

- Red (~650–680 nm): the reduction in chlorophyll content due to infection leads to increased reflectance in this region. Studies on aphid-infested wheat have reported significantly higher reflectance in infected leaves between ~630 and 680 nm, consistent with chlorophyll degradation under stress [199,202] . As disease severity increases, additional reflectance increases have been reported in the blue-yellow bands (450–500 nm and 560 nm), indicating progressive chlorosis [199].
- Red-edge (~700–740 nm): this region reflects structural changes in the mesophyll. Numerous studies have shown that the position and slope of the red-edge shift in infected leaves [162]. In citrus trees affected by HLB, spectral changes initially appear in the red-edge region (660–750 nm), even before visible symptoms develop [199]. Generally, diseased leaves show a "flattened" red-edge or a shift toward shorter wavelengths, indicative of mesophyll damage and reduced photosynthetic efficiency[203].

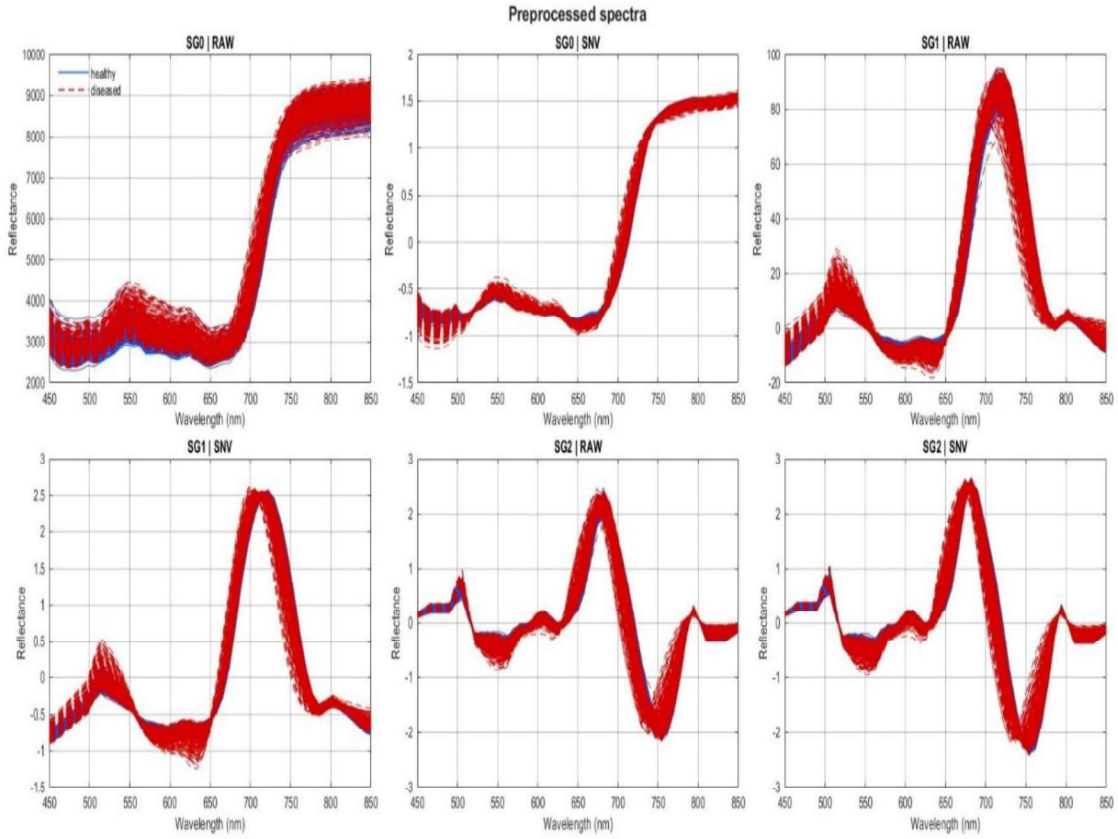


Figure 3.9. Mean reflectance spectra of healthy (blue lines) and infected leaves(red lines) after applying different preprocessing techniques (RAW, SNV, SG1, SG2, SG1+SNV, SG2+SNV).

The PLS models developed using spectral data and CVA as a feature selection method show overall good performance, with R^2CV values ranging from 0.72 to 0.81 and RPD values in cross-validation between 1.8 and 2.3, indicating an adequate level of accuracy for screening and physiological monitoring purposes on chlorophyll response. (Table 3.5) External prediction shows slightly lower performance (R^2P = 0.68–0.72), as expected, yet remains consistent with the results obtained in cross-validation. The application of SNV did not substantially alter the performance compared to the RAW spectra, with R^2CV remaining around 0.80. Although the use of first and second derivatives (SG1 and SG2) did not significantly improve cross-validation performance, the SG1 model achieved the best external prediction accuracy (R^2P = 0.718, RMSEP = 44.0), which was the lowest RMSEP among all PLS models.

The use of VIP-based variable selection reduced the number of wavelengths from 51 to between 11 and 22, resulting in more compact and interpretable models. While cross-validation performance was slightly lower than PLS-CVA models (R^2CV = 0.70–0.73; RPD = 1.83–1.93), the external prediction still yielded RPD values greater than 2, outperforming the CVA-based PLS models in this regard. The best predictive accuracy was obtained using the SG1 + SNV combination (R^2P = 0.771, RMSEP = 40.68), corresponding to the lowest prediction error among all models. Nevertheless, a more balanced trade-off across performance metrics was observed in the SG2-based model (R^2P = 0.759, RMSEP = 41.804, RPD = 2.04), which suggests robust generalization with slightly improved stability.

Table 3.5.Summary of the performance of PLS regression models for predicting chlorophyll content based on different spectral preprocessing and wavelength selection methods. Key performance parameters include: R^2 (coefficient of determination), RMSE (root mean square error), and RPD (residual predictive deviation), reported for both cross-validation (CV) and external validation (P). Additional information includes the number of outliers excluded during calibration (OUTLIERS), number of calibration samples (SAMPLE), number of selected wavelengths (WL), and the observed value range in the calibration set (MIN–MAX).

Parameter	Model		R2CV	RMSECV	RPDCV	OUTLIERS	SAMPLE	WL	MIN- MX	R2P	RMSEP	RPDP
CHL	PLS CVA	Raw	0.804 (0.005)	35.170 (0.409)	2.26 (0.03)	28	524	51	141.9 - 533.8	0.701 (0.001)	45.374 (0.050)	1.83 (0.00)
CHL	PLS CVA	SNV	0.808 (0.005)	34.878 (0.423)	2.29 (0.03)	28	524	51	141.9 - 533.8	0.691 (0.001)	46.121 (0.059)	1.81 (0.00)
CHL	PLS CVA	SG1	0.779 (0.010)	36.991 (0.805)	2.13 (0.05)	28	524	51	141.9 - 533.8	0.718 (0.005)	44.047 (0.354)	1.89 (0.02)
CHL	PLS CVA	SG1 + SNV	0.780 (0.010)	36.896 (0.846)	2.14 (0.05)	28	524	51	141.9 - 533.8	0.707 (0.006)	44.882 (0.451)	1.86 (0.02)
CHL	PLS CVA	SG2	0.798 (0.141)	35.199 (6.635)	2.29 (0.15)	28	524	51	141.9 - 533.8	0.686 (0.155)	45.992 (6.747)	1.83 (0.11)
CHL	PLS CVA	SG2 + SNV	0.727 (0.654)	37.867 (17.085)	2.18 (0.20)	28	524	51	141.9 - 533.8	0.681 (0.094)	46.626 (4.732)	1.80 (0.10)

CHL	PLS _VIP	Raw	0.714 (0.007)	40.930 (0.469)	1.87 (0.02)	21	531	20	155.1 – 533.8	0.769 (0.001)	40.930 (0.056)	2.09 (0.00)
CHL	PLS _VIP	SNV	0.710 (0.007)	40.492 (0.518)	1.86 (0.02)	26	526	22	155.1 – 533.8	0.759 (0.001)	41.788 (0.045)	2.04 (0.00)
CHL	PLS _VIP	SG1	0.722 (0.004)	40.400 (0.296)	1.90 (0.01)	21	531	11	141.9 - 533.8	0.751 (0.001)	42.433 (0.047)	2.01 (0.00)
CHL	PLS _VIP	SG1 + SNV	0.702 (0.006)	41.009 (0.428)	1.83 (0.02)	23	529	18	155.1 – 533.8	0.771 (0.001)	40.678 (0.071)	2.10 (0.00)
CHL	**PLS _VIP**	**SG2**	**0.730 (0.005)**	**39.135 (0.367)**	**1.93 (0.02)**	**24**	**528**	**17**	**155.1 – 533.8**	**0.759 (0.001)**	**41.804 (0.048)**	**2.04 (0.00)**
CHL	PLS _VIP	SG2 + SNV	0.723 (0.006)	39.316 (0.409)	1.90 (0.02)	28	524	17	155.1 – 533.8	0.767 (0.001)	41.044 (0.059)	2.08 (0.00)

The results indicate that the prediction of total phenolic content (TPC) from spectral data is overall less accurate than the estimation of chlorophyll content (Table 3.6). This behavior is expected and consistent with the literature, as phenolic compounds are characterized by high physiological variability and their accumulation is often related to induced defense processes that are not linearly correlated with leaf reflectance. In PLS models, cross-validation R^2 (R^2CV) values range between 0.38 and 0.47, with RPD between 1.28 and 1.38. Among the various preprocessing methods tested, SG1 proved to be the most effective for TPC, achieving the best performance in CV (R^2CV = 0.472; RPD = 1.38), suggesting that the first derivative enhances spectral signals linked to phenolics, likely in the 520–580 nm region. However, in external validation, a marked drop in performance is observed: R^2P ranges from 0.067 to 0.330, and RPD remains modest (around 1.05–1.24), indicating limited and potentially unstable predictive capacity.

Unlike what was observed for chlorophyll, the use of VIP-based variable selection does not substantially improve the prediction of TPC. On average, PLS-VIP models show R^2CV values between 0.24 and 0.28 and RPD values around 1.16–1.18, lower than the corresponding PLS-CVA models. Similarly, in external validation, performance remains weak: the best result is obtained with the PLS-VIP + SG1 + SNV combination (R^2P = 0.376; RPD = 1.28), yet still inferior to CVA-based models. Overall, the most reliable model for TPC estimation is the PLS-CVA with SG1 preprocessing, which achieves the best external performance (R^2P = 0.330; RMSEP = 60.48; RPD = 1.24), representing an acceptable compromise for qualitative screening applications, though with the accuracy limitations highlighted above.

Table 3.6.Summary of the performance of PLS regression models for predicting Total Phenols Content (TPC) based on different spectral preprocessing and wavelength selection methods. Key performance parameters include: R^2 (coefficient of determination), RMSE (root mean square error), and RPD (residual predictive deviation), reported for both cross-validation (CV) and external validation (P). Additional information includes the number of outliers excluded during calibration (OUTLIERS), number of calibration samples (SAMPLE), number of selected wavelengths (WL), and the observed value range in the calibration set (MIN–MAX).

Parameter	Model		R2CV	RMSECV	RPD CV	OUTLIERS	SAMPLE	WL	MIN - MAX	R2P	RMSEP	RPDP
TPC	PLS CVA	Raw	0.392 (0.027)	46.037 (1.033)	1.29 (0.03)	9	164	16	88.7 - 428.1	0.067 (0.003)	71.379 (0.119)	1.05 (0.00)
TPC	PLS CVA	SNV	0.397 (0.028)	45.729 (1.066)	1.29 (0.03)	9	164	19	88.7 - 428.1	0.072 (0.005)	71.163 (0.182)	1.05 (0.00)
TPC	PLS CVA	**SG1**	**0.472 (0.024)**	**45.053 (1.041)**	**1.38 (0.03)**	**9**	**164**	**23**	**88.7- 534.1**	**0.330 (0.005)**	**60.478 (0.214)**	**1.24 (0.00)**
TPC	PLS CVA	SG1 + SNV	0.390 (0.037)	48.924 (1.469)	1.29 (0.04)	9	164	28	88.7- 534.1	0.107 (0.006)	69.836 (0.241)	1.07 (0.00)
TPC	PLS CVA	SG2	0.443 (0.032)	47.379 (1.374)	1.35 (0.04)	9	164	29	88.7- 534.1	0.174 (0.004)	67.147 (0.148)	1.11 (0.00)
TPC	PLS CVA	SG2 + SNV	0.385 (0.035)	46.442 (1.300)	1.28 (0.04)	9	164	27	88.7 - 428.1	0.193 (0.005)	66.358 (0.185)	1.13 (0.00)
TPC	PLS_VIP	Raw	0.276 (0.051)	59.843 (2.085)	1.18 (0.04)	4	169	17	88.7- 556.7	0.205 (0.006)	79.594 (0.303)	1.13 (0.00)
TPC	PLS_VIP	SNV	0.246 (0.059)	61.055 (2.363)	1.16 (0.04)	4	169	18	88.7- 556.7	0.279 (0.010)	75.794 (0.535)	1.19 (0.01)
TPC	PLS_VIP	SG1	0.264 (0.042)	59.869 (1.696)	1.17 (0.03)	5	168	11	88.7- 556.7	0.205 (0.005)	79.605 (0.274)	1.13 (0.00)
TPC	PLS_VIP	SG1 + SNV	0.258 (0.058)	60.566 (2.338)	1.17 (0.04)	4	169	25	88.7- 556.7	0.376 (0.013)	70.543 (0.713)	1.28 (0.01)
TPC	PLS_VIP	SG2	0.258 (0.058)	59.661 (2.306)	1.17 (0.04)	6	167	19	88.7- 556.7	0.283 (0.011)	75.571 (0.576)	1.20 (0.01)
TPC	PLS_VIP	SG2 + SNV	0.254 (0.062)	59.687 (2.454)	1.16 (0.05)	5	168	22	88.7- 556.7	0.294 (0.015)	74.989 (0.773)	1.20 (0.01)

Similar to the present results, the PLS models developed by [204]demonstrated that near-infrared spectroscopy (NIRS) applied to grapevine leaves yielded strong predictive accuracy for chlorophyll content, with an external coefficient of determination (R^2P) of approximately 0.92 and a residual predictive deviation (RPD) of ~3.41. In contrast, the prediction of total phenolic content (TPC) was notably less accurate, with an R^2P of ~0.76 and a higher error (RPD ≈ 1.90). These results align with a broader body of literature showing that Partial Least Squares (PLS) regression typically performs well in estimating chlorophyll from Vis/NIR spectra, whereas the prediction of phenolic compounds proves more challenging due to their physiological variability and diffuse spectral features. [205], using hyperspectral data ranging from 350 to 2500 nm, reported high predictive performance for chlorophyll pigments ($r \approx 0.86$), but poor results for TPC. Specifically, their PLS model

for phenolic compounds yielded negligible predictive power, with a correlation coefficient of $R^2 \approx 0$ and an RPD close to 1, indicating near-random predictions.

Nonetheless, several recent studies have demonstrated substantial improvements in TPC prediction by leveraging advanced modeling strategies. For instance, [206] applied hyperspectral imaging extended into the short-wave infrared (SWIR) region (up to 2500 nm) on Arabidopsis thaliana, employing feature selection techniques such as VIP, interval PLS (iPLS), and genetic algorithms (GA). Their PLSR model, applied exclusively to SWIR data, achieved a remarkably high predictive accuracy for TPC with an R^2 of approximately 0.96 in validation. Taken together, these findings underscore that while chlorophyll content can be reliably predicted using standard spectral approaches, the estimation of phenolic compounds often requires extended spectral coverage and more sophisticated multivariate techniques to capture their complex physiological dynamics.

The Figure 3.10 shows the mean leaf reflectance spectrum, with superimposed wavelengths selected by the PLS-VIP models for chlorophyll (green dashed lines) and total phenolic content (solid red lines). The visual analysis clearly highlights differences in the spectral distribution of the selected variables for each target compound.

For chlorophyll (CHL), the selected wavelengths are predominantly concentrated in two physiologically relevant regions: the 650–680 nm range, which corresponds to the peak absorption of chlorophyll a, and the red-edge region (700–740 nm), which is strongly influenced by mesophyll structure and chlorophyll density. This distribution confirms that the model is capturing robust biochemical signals that reflect actual changes in photosynthetic pigments. The recurrence and physiological coherence of these bands explain the high predictive performance observed for chlorophyll ($R^2P \approx 0.75$; RPD > 2), supporting the model's suitability for reliable quantitative estimation. Conversely, the wavelengths selected for TPC are more widely scattered across the spectrum, with substantial overlap with the chlorophyll-related bands and minimal representation in the 520–580 nm range, typically associated with phenolic pigments such as anthocyanins and flavonoids. This overlap suggests that the TPC model is likely capturing indirect signals related to general physiological stress, rather than specific phenolic absorption features.

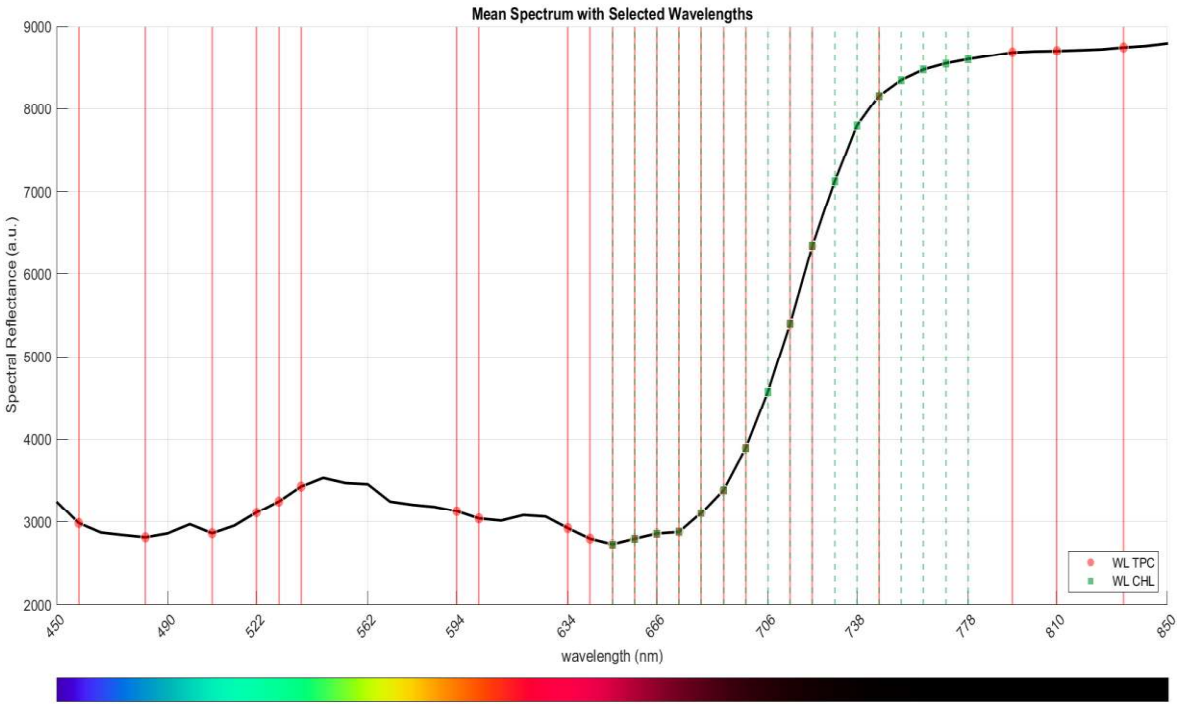


Figure 3.10. Mean reflectance spectrum of tomato leaves, calculated as the average of healthy and infected plants. Highlighted on the spectrum are the wavelengths selected by the best PLS model for chlorophyll prediction (17 wavelengths, dashed green lines) and for total phenolic content (TPC) prediction (23 wavelengths, solid red lines).

Compared to the temporal trends of pigments and total phenolic content (TPC) shown in Figure 3.4, the spatial distribution of predictions on leaf data across different time points reveals low model stability and spatial consistency, especially for TPC.

While the average trend depicted in Figure 3.11 clearly reflects the progressive decline in chlorophyll content in infected plants, the predicted TPC distribution (Figure 3.12) appears highly fragmented and temporally inconsistent. This pronounced mismatch highlights the limited reliability of the PLS models for estimating phenolic compounds, confirming the weak predictive performance already suggested by the model metrics.

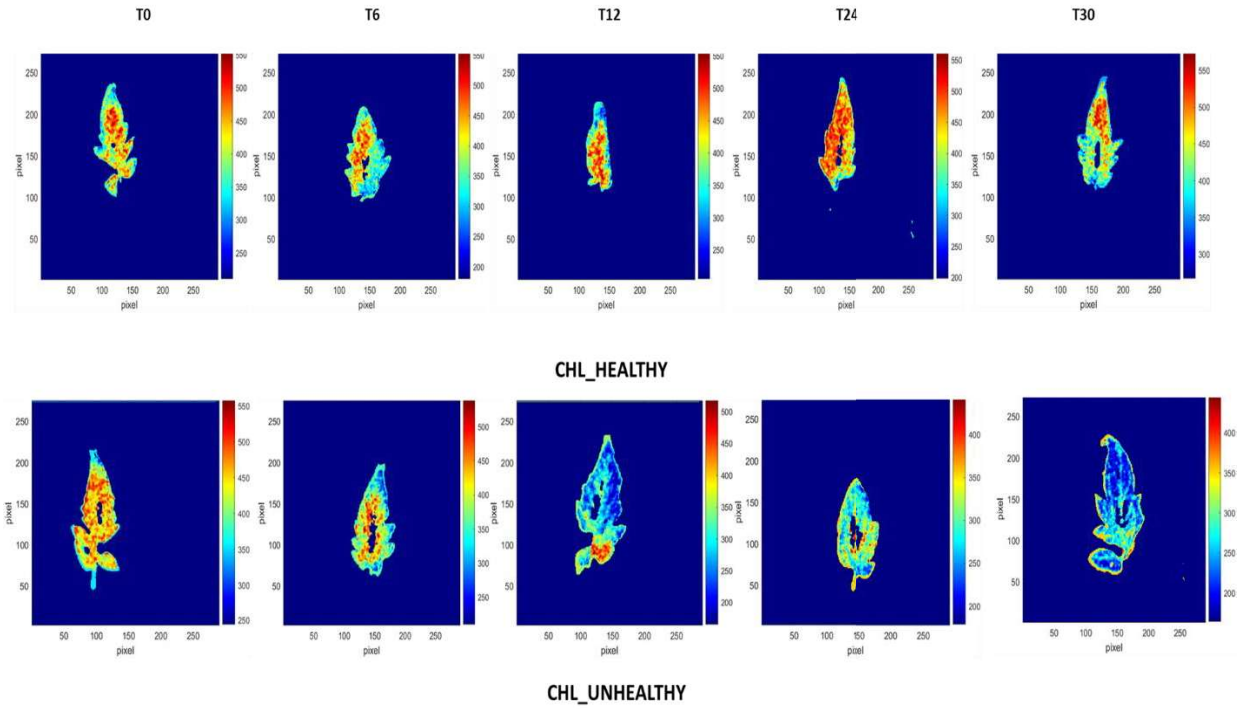


Figure 3.11.Spatial distribution maps of chlorophyll content in tomato leaves predicted by the best PLS model (VIP-PLS with SG2 preprocessing) at different experimental time points.

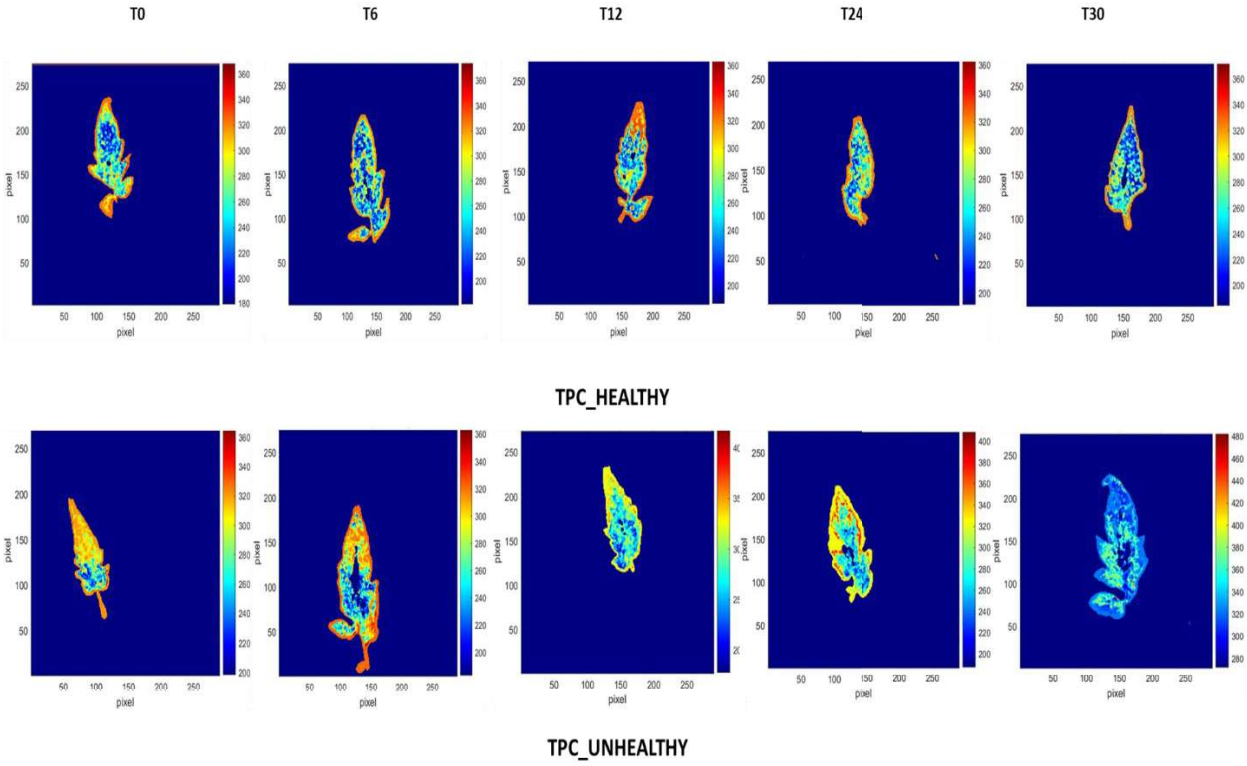


Figure 3.12. Spatial distribution maps of total phenol content (TPC) in tomato leaves predicted by the best PLS model (CVA-PLS with SG1 preprocessing) at different experimental time points.

3.3.3. Classification of diseases and healthy tomato plants

3.3.3.1. Binary classification

The analysis of confusion matrices on the external validation set (Figure 3.13) provides additional insight into the models' ability to correctly distinguish between healthy and infected plants. Linear models (PLS-DA and LDA) show a balanced distribution between true positives and true negatives, whereas SVM reveals a higher incidence of false negatives. Conversely, the RF model tends to generate a higher number of false positives, reflecting less reliable classification in the prediction set.

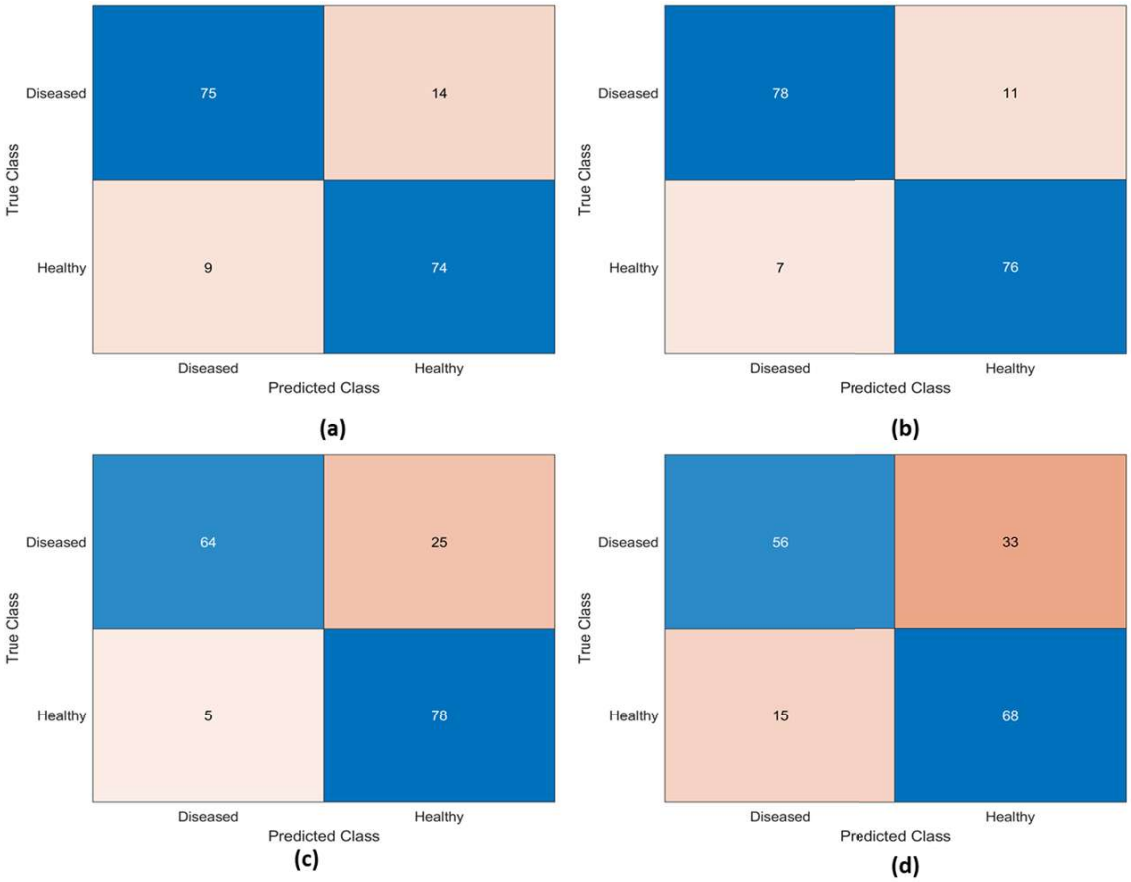


Figure 3.13. Confusion matrices for binary classification models: a=LDA; b= PLS-DA; c= SVM;d= RF) on the external validation set.

Specifically, Table 3.7 summarizes the binary classification performance between healthy and diseased plants.

Table 3.7.Performance metrics for binary classification (healthy vs. infected plants) using different models: Linear Discriminant Analysis (LDA), Partial Least Squares Discriminant Analysis (PLS-DA), Random Forest (RF), and Support Vector Machine (SVM). The table reports accuracy, precision, recall, and F1-score for both cross-validation (CV) and external validation (EXT) sets.

	CLASSIFIER	ACCURACY_CV	PRECISION_CV	RECALL_CV	F1_CV	ACCURACY_EXT	PRECISION_EXT	RECALL_EXT	F1_EXT
BINARY CLASS	**LDA**	0.89	0.91	0.87	0.89	0.87	0.89	0.84	0.87
	PLS-DA	0.89	0.90	0.89	0.89	0.90	0.92	0.88	0.90
	RF	0.75	0.77	0.71	0.74	0.72	0.79	0.63	0.70
	SVM	0.91	0.94	0.87	0.90	0.83	0.93	0.72	0.81

Accuracy and F1-score results on the external test set highlight how linear models (LDA, PLS-DA) achieved better and more consistent performance compared to more complex models (RF, SVM) on VIS/NIR spectral data. In general, robustness in external prediction—i.e., the ability to generalize beyond the calibration dataset—depends on how well the model avoids fitting to noise or overfitting the training data.

Focusing on linear models, PLS-DA and LDA showed high accuracy (87–90%) on the external test set, indicating good generalization ability. These models represent two commonly used linear approaches for spectral data analysis in binary classification, especially in agricultural and phytopathological contexts. LDA is based on linear combinations of predictive variables to maximize class separation, assuming Gaussian distributions with a shared covariance matrix. This makes it easy to implement, as it requires estimating a small number of parameters, reducing the risk of overfitting in high-dimensional datasets with relatively few samples. However, LDA is highly sensitive to collinearity among predictor variables, a common characteristic in spectral data. PLS-DA, on the other hand, is explicitly designed to handle high-dimensional and multicollinear data. It enables dimensionality reduction in an informed way, optimizing class discrimination. One of the key advantages of PLS-DA is its interpretability, allowing the extraction of the most relevant wavelengths for classification. Nevertheless, its performance is highly influenced by the number of latent components selected, which may lead to either overfitting or underfitting .

From the perspective of non-linear models, SVM is often reported as a promising method for high-dimensional data with limited sample sizes, due to its margin maximization and use of non-linear kernels (such as the RBF kernel used in this study). Similarly, Random Forest (RF), an ensemble of decision trees, is designed to improve generalization through

bagging. The variability introduced by randomly selecting data subsets and features for each tree helps reduce variance and mitigate overfitting [207]. Although these models often show promising results in the literature [208–210], findings from this study indicate a more substantial performance drop in external validation (accuracy 0.83 for SVM, 0.72 for RF), confirming the known tendency of more complex models to overfit training data and lose performance on unknow samples [211].

Although thorough tuning is always recommended for such models, several authors have confirmed this trend, noting that simplification strategies (e.g., reducing tree depth in RF) often yield better predictive performance[210,212]. Finally, model selection should not rely solely on aggregate metrics such as accuracy or F1-score, but rather involve careful evaluation of the balance between precision and recall, as these reflect two critical aspects of agronomic diagnosis. Specifically, a model's ability to minimize false positives (FP), i.e. healthy plants wrongly classified as infected, and false negatives (FN), infected plants not detected, is crucial to ensure effective and sustainable interventions.

In crop protection, false positives can lead to unnecessary treatments, increasing management costs, chemical usage, and environmental impact. On the other hand, false negatives hinder early diagnosis, allowing pathogen spread and resulting in significant yield losses. In this context, the results indicate PLS-DA as the best operational trade-off: with an external validation precision of 0.92 and recall of 0.88, the model shows high capacity to detect infected plants while maintaining a low false alarm rate. This makes it especially suitable for agronomic applications where timeliness and selectivity are both essential. LDA, while simpler and statistically stable, showed comparable performance (precision: 0.89, recall: 0.84), with a slightly higher risk of missing some infected cases. In field scenarios, this could reduce the effectiveness of phytosanitary containment unless complemented by frequent monitoring. Conversely, the SVM model, despite showing high precision (0.93), had lower recall (0.72), indicating a greater tendency to miss infected plants, particularly critical in early stages of infection. Lastly, the RF model demonstrated the weakest recall performance (0.63), making it less suitable for early disease detection scenarios where prevention is the priority.

3.3.3.2. Multiclass classification

The multiclass classification of plant physio pathological conditions , structured into four stages (0 = healthy, 1 = presymptomatic, 2 = intervention phase, 3 = advanced damage), revealed clear performance differences among the tested models, highlighting substantial variation in predictive capabilities, as summarized in Table 3.8.

Table 3.8.Performance of multiclass classification models (Random Forest – RF, Support Vector Machine – SVM, and Naive Bayes – CNB) across four plant physiological condition classes (0 = healthy, 1 = presymptomatic, 2 = intervention phase, 3 = severe damage), evaluated on the external validation set. For each classifier, class-wise accuracy (Accuracy_EXT) and F1-score (F1_EXT) are reported, along with the overall class average.

	CLASSIFIER	CLASSES	0	1	2	3	MEAN
MULTICLASS	**RF**	Accuracy_EXT	0.61	0.54	0.35	0.70	0.55
		F1_EXT	0.51	0.57	0.36	0.75	0.54
	SVM	Accuracy_EXT	0.78	0.71	0.52	0.81	0.71
		F1_EXT	0.70	0.69	0.60	0.81	0.70
	CNB	Accuracy_EXT	0.44	0.54	0.13	0.59	0.44
		F1_EXT	0.31	0.53	0.17	0.65	0.42

The Support Vector Machine (SVM) model demonstrated the best overall performance, with an average accuracy of 71% and an average F1-score of 0.70. It performed particularly well in classifying healthy cases (class 0) and advanced stress conditions (class 3), with F1-scores of 0.70 and 0.81, respectively. These results confirm the ability of SVM to effectively model even complex spectral patterns, as also reported by [212], where SVM outperformed LDA and RF in detecting asymptomatic infected plants using hyperspectral data. However, its moderate performance in the intermediate stage (class 2, F1 = 0.60) indicates that its ability to capture physiological transitions could be improved. The Random Forest (RF) model, with an average accuracy of 55% and a mean F1-score of 0.54, showed significant variability across classes. While it was effective in detecting severe damage (class 3, F1 = 0.75), its low sensitivity in classes 1 and 2 (F1 = 0.57 and 0.36, respectively) limits its applicability for early stress detection. This pattern aligns with previous findings [150,213], which noted that RF tends to overweight dominant features while neglecting weaker or transient signals, such as those associated with early stages of infection or stress.

The Naive Bayes (CNB) model delivered the weakest performance, with an average accuracy of 44% and a mean F1-score of 0.42. It struggled particularly with the intermediate classes, especially class 2 (F1 = 0.17). This is due to the model's simplifying assumption of

feature independence—an assumption clearly violated in spectral data, where high collinearity among bands is common. CNB's lack of flexibility makes it poorly suited for complex VIS-NIR data, as also demonstrated by [214], who reported significantly lower performance for CNB compared to kernel-based or ensemble models.

From an agronomic perspective, classes 1 (presymptomatic) and 2 (intervention phase) are the most critical for plant protection strategies, as they represent key time windows for early diagnosis and effective treatment. In this context, the analysis of ROC (Receiver Operating Characteristic) curves provides a detailed representation of classification performance, highlighting the trade-off between sensitivity (TPR) and specificity (1 – FPR) across all decision thresholds. Figure 3.14 shows the ROC curves for the SVM, RF, and CNB models across the four physiological classes (0–3). The SVM model stands out with high AUC values in all classes, achieving optimal performance in class 0 (AUC = 0.93) and class 3 (AUC = 0.97), indicating high precision in detecting both healthy plants and cases of advanced infection.

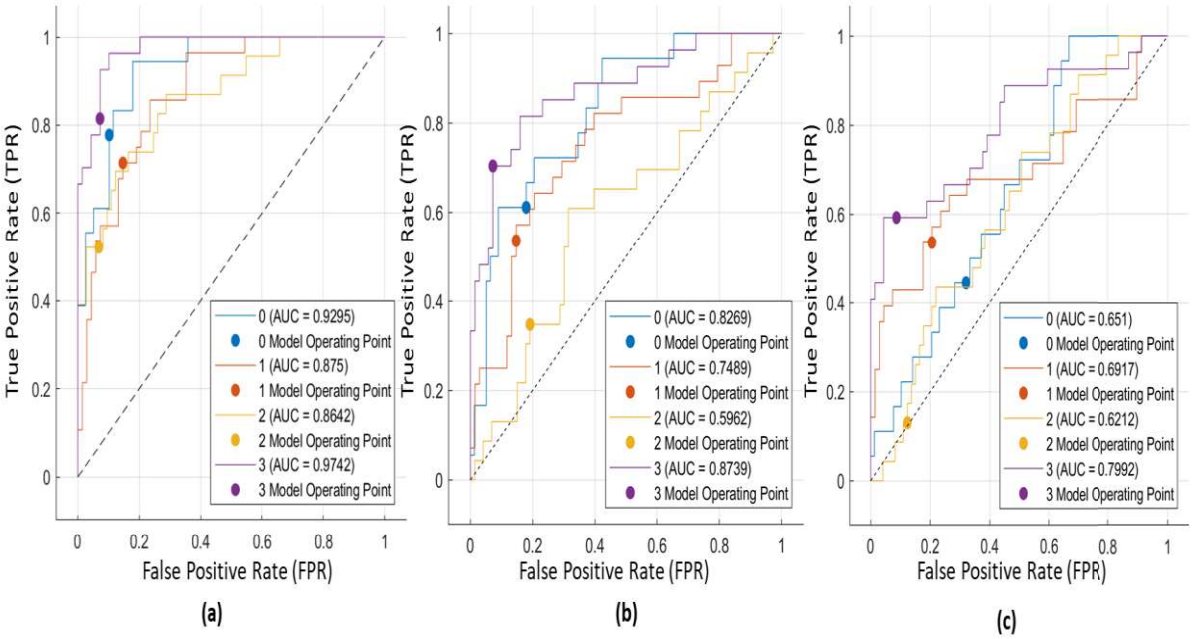


Figure 3.14. Receiver Operating Characteristic (ROC) curves for each physiological class (0 = healthy, 1 = presymptomatic, 2 = intervention stage, 3 = severe damage) generated by the Support Vector Machine (SVM) (a), Random Forest (RF) (b), and Naive Bayes (CNB) (c) models, based on the external validation set. Area Under the Curve (AUC) values are reported in the legend for each class, indicating each model's discriminative ability across different stages of leaf condition

In contrast, the Random Forest model displays less stable ROC curves: while it achieves a good AUC in class 3 (0.87), its performance drops significantly in class 2 (AUC = 0.59), reflecting low sensitivity in detecting intermediate stages, which are critical for timely containment. The Naive Bayes (CNB) model shows the weakest ROC curves, with AUC values consistently below 0.70. In particular, the discrimination of the presymptomatic class (class 2, AUC = 0.62) is especially poor, with serious implications for the effectiveness of early monitoring. These results highlight that high overall accuracy does not necessarily guarantee effective presymptomatic diagnosis, especially when the model lacks sensitivity to the early alterations in spectral profiles, which are often masked by phenological noise or individual variability.

3.4. Conclusions

The combined analysis of leaf pigments, phenolic compounds, and spectral indices revealed a clear physiological divergence between healthy and infected plants, with Phytophthora infestans inducing measurable responses from the earliest stages of pathogenesis. Total phenolics were confirmed as an early marker of biotic stress, showing significant increases as early as t12 and peak at t18 (maximum value of 754 mg/100 g), approaching the damage threshold. In contrast, chlorophyll reflected a later structural impairment associated with mesophyll disruption: healthy plants showed a slight increase at t24 (447 µmol/m²), whereas infected plants exhibited a significant reduction beginning at t12, with a pronounced decline in the advanced stages of infection and a minimum value of 260 µmol/m² at t30.

Following these physiological changes, the response of spectral vegetation indices (VIs) demonstrated high sensitivity in capturing the transition from early stress to advanced physiological deterioration. Between t12 and t18, the first signs of alteration appeared (reductions in NDVI, GNDVI, and PRI), while between t18 and t30 a marked decline in leaf structure and chlorophyll content was observed, as indicated by decreases in SR and MCARI. In particular, indices such as GNDVI, MCARI, and ARI proved able to detect early physiological alterations not yet visible macroscopically, supporting their use in presymptomatic diagnosis and in monitoring metabolic responses to infection. These findings underscore the potential of spectroscopic techniques for early disease detection, even under experimental conditions characterized by non-uniform responses such as varying fungicide doses.

The temporal evolution of pigments and phenolic compounds clearly delineates the transition across the four defined physio pathological classes: class 0 (t0, healthy plants), showing stable profiles and high photosynthetic integrity; class 1 (t6–t9), characterized by initial metabolic shifts such as early phenolic increases; class 2 (t12–t18), marked by sharp chlorophyll declines and distinct spectral variations; and class 3 (t24–t30), corresponding to advanced tissue damage.

Quantitative chlorophyll and phenols prediction closely follows this behavior. The PLSR models showed a pronounced disparity between the ability to predict chlorophyll and phenolic content. Chlorophyll exhibited robust performance in both cross-validation and

external prediction (R^2P up to ~0.77; RPD > 2), due to well-defined spectral signatures in chlorophyll absorption and red-edge regions. This confirms, in agreement with the literature, the effectiveness of SG1 preprocessing and VIP-based variable selection, evidenced by the physiological coherence of the selected wavelengths (17 key wavelengths). These results support the reliability of Vis–NIR spectroscopy for non-destructive and quantitative estimation of photosynthetic pigments. On the contrary, predicting total phenolic content proved far more challenging. The modest performance ($R^2P \leq 0.33$; RPD ~1.1–1.3) and the low spatial stability of predictive maps indicate that the models primarily capture indirect signals related to general stress rather than specific spectral signatures. Although phenolics are known to be early stress markers, their spectroscopic detection is hindered by the absence of distinct absorption features and by their overlap with other pigments. In other words, predictive accuracy increases when physiological parameters follow a clear and progressive trend throughout disease development. The difficulty reflects both the high physiological variability of phenolic compounds and the weak presence of diagnostic bands within the Vis–NIR region. Moreover, reliably modelling spectral signatures specific to plant–pathogen interactions remains an open challenge, as phenolic responses are highly variable and poorly specific.

The analysis of binary and multiclass classification models further showed that predictive performance strongly depends on the diagnostic task. In binary classification (healthy vs. infected), linear models, PLS-DA and LDA, achieved the best generalization, maintaining high accuracy (≈87–90%) and an optimal balance between precision and recall. Notably, PLS-DA represented the best operational compromise, achieving high sensitivity while maintaining a low false-positive rate (precision 0.92; recall 0.88), a key requirement for timely and selective agronomic decision-making. This result can be attributed to the strong physiological separation between class 0 and the more advanced damage stages (classes 2 and 3). Conversely, more complex models such as SVM and RF displayed substantial performance decline in external validation, confirming their susceptibility to overfitting when applied to Vis–NIR datasets with limited sample size. In multiclass classification, used to distinguish among different physio pathological stages, SVM emerged as the most effective model, with strong discriminative ability particularly in extreme classes (0 and 3). However, the difficulty observed in identifying intermediate stages (classes 1 and 2) highlights a challenge common to all models, stemming from the subtle and transient physiological signals characterizing early infection. From an applied perspective, these

findings indicate that the most effective models (PLS-DA and SVM) can support early monitoring strategies, while future developments should focus on enhancing sensitivity to subtle physiological changes—potentially by integrating SWIR regions, lightweight deep learning architectures, and multimodal approaches.

In conclusion, the results demonstrate that integrating physiological measurements, quantitative prediction, and classification models provides a robust framework for early disease diagnosis. However, the limited strength of correlations between individual spectral indices and pigment or phenolic content highlights the need for more sophisticated multivariate approaches and more articulated experimental designs to achieve reliable quantitative estimates. Furthermore, Vis–NIR spectroscopy appears sufficiently mature for operational applications related to chlorophyll estimation and physiological monitoring, while phenolic prediction will require more advanced modelling strategies, extended spectral coverage into the SWIR region, and refined variable selection techniques. Combining the robustness of physiological markers (such as chlorophyll), the informativeness of spectral patterns, and the discriminative power of classification models represents a key step toward precision agriculture decision-support systems capable of identifying optimal intervention timing and preventing the progression to severe disease stages. The primary remaining challenge concerns the discrimination of presymptomatic (class 1) and intervention-threshold (class 2) stages, where spectral variations are subtle and easily masked by phenological noise. Developing models capable of capturing these weak signals will be essential for building truly early, accurate, and field-transferable diagnostic tools.

CHAPTER 4

4 Development of the robotic platform (A-bot-X1)

4.1 Introduction: Objectives and system requirements

Autonomous indoor navigation is a specialized area within autonomous robotics and represents one of the most significant challenges in the field for the coming decades. As mobile robots are increasingly deployed in real-world scenarios, it becomes essential for them to navigate independently across diverse environments without relying on GPS, such as densely populated urban spaces, residential buildings, tunnels, and office complexes. To achieve this, autonomous systems must continuously perceive, respond to, and adapt to dynamic surroundings. They should also be resilient to sensor or control failures and capable of adjusting their behavior as environmental conditions evolve. LiDAR sensors are commonly used to detect general obstacles; however, they are limited when it comes to distinguishing how a robot should behave in the presence of specific obstacles, particularly humans. A more intuitive approach to address this issue is the use of video data captured by cameras mounted on the robot. Computer vision enables the processing of image sequences or visual sensor data to identify objects, estimate their three-dimensional structure and motion, and track their positions over time. This technology offers a powerful means of enabling indoor autonomous navigation. Nevertheless, challenges such as occlusions, shadows, high computational complexity for optimal control and obstacle avoidance, and the demanding sampling rates of vision sensors remain open problems.

4.2 ROS (Robot Operating System) and ROS 2

The Robot Operating System (ROS) is an open-source framework designed to support the development of complex and modular robotic software. It provides a structured set of tools, libraries, and conventions that facilitate the creation of robust and scalable robotic applications. ROS was originally developed at the Stanford Artificial Intelligence Laboratory in collaboration with Willow Garage and was publicly released in 2007. ROS 2 [215] represents the direct evolution of ROS 1[216] and is currently one of the main reference platforms for modern robotics. Compared to its predecessor, ROS 2 introduces substantial improvements in terms of flexibility, reliability, and performance. In particular, it supports multiple operating systems and enables real-time capabilities, making it suitable

for a wide range of robotic platforms and application domains. The framework offers a comprehensive ecosystem of middleware, libraries, and tools that simplify both the development and deployment of complex robotic systems. Another major strength of ROS 2 lies in the strong support of an active and collaborative open-source community, which promotes the sharing of solutions and accelerates research and innovation in the robotics field. Thanks to its focus on robustness, security, and scalability, ROS 2 is widely adopted in domains such as manufacturing, healthcare, and logistics.

4.2.1 Nodes

A node is the fundamental computational unit in ROS 2 and is designed to perform a single modular function within a robotic system [215] For example, tasks such as wheel motor control, LiDAR sensor management, or visual data processing are typically handled by separate nodes. Nodes communicate with each other through specific interaction mechanisms provided by ROS 2, including topics, services, actions, and parameters. This node-based modular architecture allows complex systems to be decomposed into independent yet cooperative components; consequently, a complete robotic system is composed of multiple nodes operating concurrently. In ROS 2, a single executable may host one or more nodes, offering additional flexibility in software design.

4.2.2 Topics, Services, and Actions

The communication infrastructure of ROS 2 is built upon the Data Distribution Service (DDS), a standardized protocol for data exchange in distributed systems. The adoption of DDS marks one of the most significant differences with respect to ROS 1, where communication was centralized around the ROS core node. In ROS 2, DDS directly manages communication between the APIs of different supported programming languages. This transition from a centralized to a distributed model significantly enhances system reliability, robustness, and real-time performance.

Topics, implemented through the DDS protocol, act as unidirectional communication channels through which nodes exchange data. A node may publish to or subscribe to multiple topics simultaneously, enabling one-to-one, one-to-many, or many-to-one communication patterns. The collection of nodes and topics forms the so-called ROS graph, which represents a network of interconnected computational modules processing data in parallel. Within this architecture, communication can occur via messages, services, or actions.

Messages constitute the most basic communication mechanism in ROS 2 and are used to transmit typed information between nodes. They are defined using a textual format known

as the Message Definition Language (.msg), which specifies the structure and data types contained in the message. Each message consists of one or more fields, each characterized by a type and a name. Messages may be provided by standard ROS libraries or defined by users for application-specific purposes.

Services allow a node to request the execution of a specific operation from another node using a request–response communication model. Similar to messages, services are defined through a Service Definition Language (.srv), which describes both the request and response message formats. In ROS 2, services are typically synchronous and blocking, meaning that the client node waits for the server's response before continuing its execution.

Actions provide a more advanced communication mechanism compared to messages and services and are designed for long-running tasks that require intermediate feedback and the ability to cancel or preempt execution. Actions are defined using an Action Definition Language (.action), which specifies the structure of the goal, result, and feedback messages. This mechanism enables asynchronous interactions between nodes and is particularly well suited for complex tasks such as navigation or manipulation [217].

In ROS 2, a node can function as both an action server and an action client. The server declares the supported actions and defines the callbacks executed upon receiving a goal, while the client sends goal requests and receives feedback and final results. During action execution, the client is not required to block while waiting for completion, thus enabling asynchronous behavior. Furthermore, goals can be cancelled or preempted at any time, providing a high degree of control over task execution.

4.2.3 ROS2 in MATLAB Environment

Interoperability of MATLAB in Industrial and Academic Environments – MATLAB, equipped with dedicated toolboxes such as the ROS Toolbox and Robotics System Toolbox, is widely adopted in both academia and industry for real-time control and integration with frameworks like ROS 2. In modern robotic workflows, engineers typically design control algorithms (e.g., PID, LQR/LQG) in MATLAB or Simulink, simulate them in complex environments, and then generate C++ code compatible with ROS 2 for deployment on embedded systems. This integration is facilitated by tools such as *ros2node, ros2publisher, and ros2subscriber*, which allow direct communication between MATLAB and ROS 2 networks. As a result, MATLAB becomes an effective bridge between the algorithm development environment and the real-time robotic execution platform [218].

Academic literature highlights that this interoperability is particularly advantageous due to MATLAB's comprehensive suite of tools for data analysis, visualization, and control

system design [217]. The ability to perform automatic code generation from Simulink models to ROS 2 nodes ensures that algorithms developed in simulation can be transferred efficiently to real robotic systems, meeting both performance and industrial standards [219]. Overall, this workflow combines the strengths of both ecosystems: MATLAB for prototyping, analysis, and tuning, and ROS 2 for real-time, distributed execution. Numerous research studies have shown that robust control strategies for autonomous robots often originate from MATLAB designs and are executed through ROS/ROS 2, underlining the strategic value of this integration [218].

4.2.4 MATLAB for Sensor Data Analysis (LiDAR & Camera):

MATLAB is also a popular tool for analyzing and processing data from sensors such as LiDAR and cameras. Over the last few years, the integration of MATLAB with ROS and ROS 2 has become much more robust, driven by demand in both academic and industrial robotics. In fact, it's commonly used to import, visualize, and edit recorded sensor data (such as ROS bag log files) offline. As one research center noted, *"Another popular option is MATLAB. The data can be imported, viewed and edited in MATLAB"* [220], referring to its use in working with ROS bag datasets for robotics. This indicates that in academic and industrial practice, MATLAB's environment and toolboxes (like the Computer Science or LIDAR Toolbox) are routinely leveraged for sensor data analysis and calibration. For example, MATLAB provides dedicated functions for camera and LiDAR calibration and point cloud processing, allowing engineers to accurately calibrate camera–LiDAR setups and analyze sensor data with high precision. The advantage of using MATLAB in this context lies in its extensive library of algorithms for filtering, signal processing, and computer vision.

Moreover, the integration between the MATLAB and ROS ecosystems (including tools like *Foxglove Studio for visualization*) has enabled a hybrid approach, where MATLAB is used for in-depth analysis and debugging of sensor data, as well as for designing algorithms, which can then be deployed into a ROS 2 environment [221]. This kind of interoperability and data analysis capability in MATLAB has been highlighted in recent literature, emphasizing how it simplifies the development of robotic systems by providing a high level interface for tasks like LiDAR point cloud processing, image processing, and even machine learning integration, all while maintaining compatibility with real-time ROS 2 operations [219].

4.3 Hardware architecture: Mechanical structure of A-bot-X1 prototype

4.3.1 Prototype design

The prototype design criteria have been identified as follows.

The wheels are constituted of two 20'' bike with fat tires (diameter of 508 mm). The prototype is operated by two motorized wheels (24V, 250W brushed motors, 75 RPM, 13.4 A, with an efficiency of 0.78) and two caster wheels. The system is able to adapt its working width from **1300 mm up to 1800 mm** (wheel to wheel).

Using the camera field of view (FOV) parameters (50.1 degree of vertical aperture view and 63.9 degrees of horizontal aperture view), the calculation of camera height on the ridge can be easily calculated in order to take the image of the full horizontal ridge base (1713 mm) below the prototype, the height of the picture taken covers 1283 mm. As the camera has 1280x960 pixels resolution, each camera pixel covers an area of 1.79 mm2, i.e. **a square of about 1.34 mm**.

This last value must be the precision that must be guaranteed by the movement controller.

The calculated camera height is **1373 mm** to which must be added the 100 mm ridge height over the campaign plan; therefore, the camera height must be 1473 mm over the campaign plane (Figure 4.1).

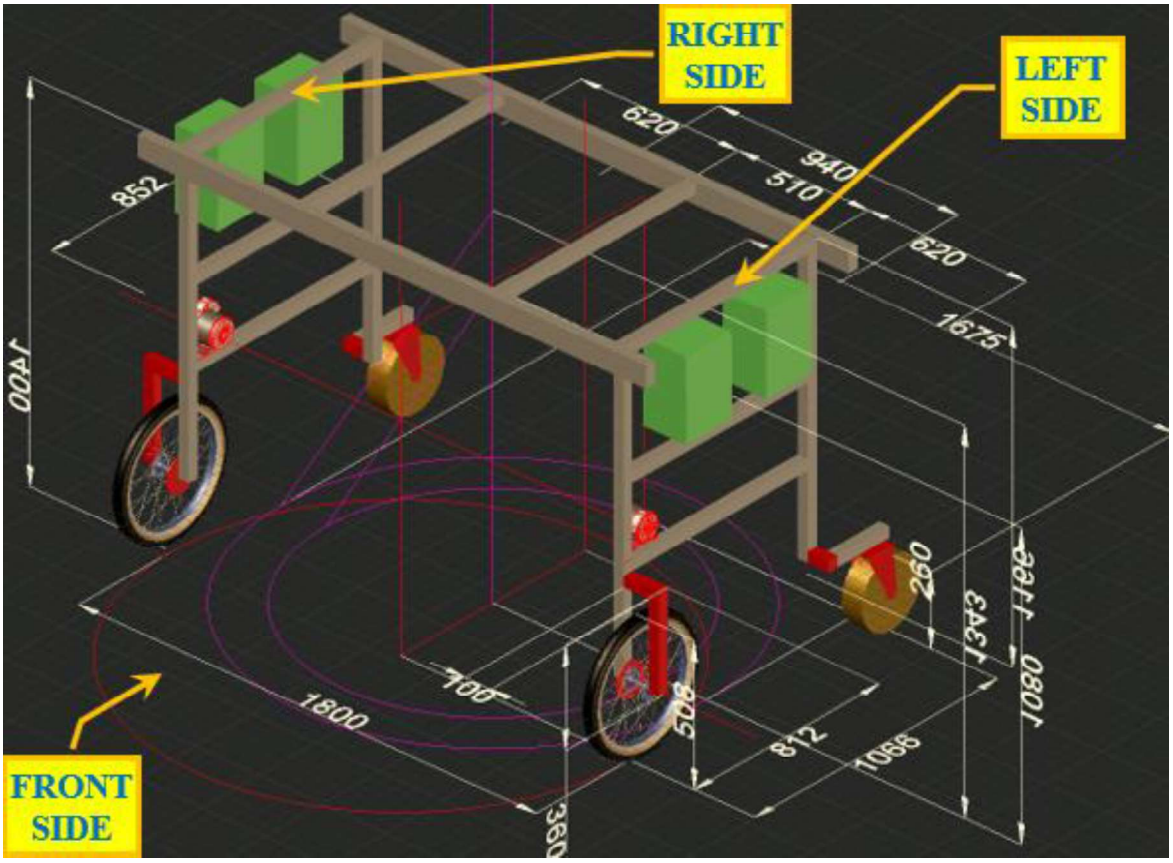
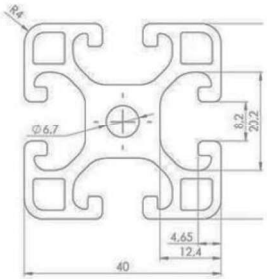


Figure 4.1. A-bot-X1 prototype design (V11).

4.3.1.1 Strut profile sizing and verify

The selected general assembling profile is an aluminium structural strut profile, 40x40L. Thanks to their computer-optimized design, the strut profiles L with 8 mm slot (modular dimensions 40 mm) combine high strengths with minimum use of materials.

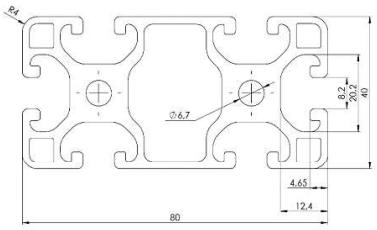
https://uk.rs-online.com/web/p/tubing-and-profile-struts/7613319



Material	Anodised Aluminium
Number of Grooves	4
Mechanical Specifications	
Strut Profile	40mm x 40mm
Groove Size	8mm
Linear Mass (G)	1.5 kg/m
Moment of Inertia (I2)	9.18 cm4
Section Modulus (WX)	4.59 cm3

In addition, four 40x80L strut profiles are used in the upper frame.

https://uk.rs-online.com/web/p/tubing-and-profile-struts/7613329



Material	Anodised Aluminium
Number of Grooves	6

Mechanical Specifications	
Strut Profile	40mm x 80mm
Groove Size	8mm
Linear Mass (G)	3.06 kg/m
Moment of Inertia (I2)	70.16 cm4
Section Modulus (WX)	17.54 cm3

4.3.1.2 Verifying the profile buckling load

In engineering structures, the buckling is the abrupt change in curvature of a structural element under stress, like the bending of a pillar during compression. A element or the structure itself is said to have buckled if a steadily rising load is applied to it to the point at which a part abruptly changes shape. The buckling stress of a beam may be found using Johnson's parabolic formula and Euler's critical load.

According to AISC (American Institute of Steel Construction) "Manual of Steel Construction" "Simplified Design of Structural Steel", Harry Parker; the calculations follow.

Assuming:

- Fa = Permitted buckling strain N/cm2
- K = Factor for effective buckling length;
- L = Length
- r = Radius of gyration
- Fy = Yielding point
- E = Elastic modulus
- Cc = Coefficient of fineness

The selected 40x40L strut profile characteristics are as follows:

- Ix = 9 cm4
- Iy = 9 cm4
- A = 5 cm2
- Fy = 25000 N/cm2
- E = 7E6 N/cm2
- SF = 4 (safety factor)

Our case is:

- K = 2 (fixed below, free above, pivoted)

- L = 140 cm
- r = sqrt(Iy/A)

The radius of gyration is the root mean square distance of the object's parts from either its centre of mass or a given axis, depending on the relevant application; in our case it is r= sqrt(Iy/A).

The calculation involves some steps.

Step 1:

$$Cc = \sqrt{\frac{2\,\pi^2\,E}{Fy}} \tag{4.1}$$

Step 2:

$$Z = (K\,L\,/\,r)\,/\,Cc \tag{4.2}$$

Step 3:

If Z<1 then

$$Fa = \frac{1-\frac{1}{2}Z^2}{\frac{5}{3}+\frac{3}{8}Z-\frac{1}{8}Z^3}\,Fy \tag{4.3}$$

Else, if Z>=1 then

$$Fa = \frac{12\,\pi^2\,E}{23\left(\frac{K\,L}{r}\right)^2} \tag{4.4}$$

In current case, the calculated buckling load is F = Fa*A = 4138 N, however, considering the SF, it is 1035 N (or **105.5 kgf**). This value is above the overall sum of hypothesized prototype mass (30 kg) and payload of 50 kg.

The following Matlab script performs the calculation.

```
K= 2 % no dim
L_cm= 140 % cm
Iy_cm4= 9 % cm4
A_cm2= 5 % cm2
Fy_N_cm2= 25000 % N/cm2
E_N_cm2= 7E6 % N/cm2
SF= 4 % no dim
r_cm= sqrt(Iy_cm4/A_cm2) % cm
Cc= sqrt(2*pi^2*E_N_cm2/Fy_N_cm2) % no dim
Z= (K*L_cm/r_cm)/Cc % no dim
if Z<1
    Fa_N_cm2= (1-0.5*Z^2)/(5/3+3/8*Z-1/8*Z^3)*Fy_N_cm2
else
    Fa_N_cm2= 12*pi^2*E_N_cm2/( 23*(K*L_cm/r_cm)^2 )
end
F_N= Fa_N_cm2*A_cm2 % N
F_N_SAFE= F_N/SF %N
F_kgf_SAFE= F_N_SAFE/9.81 % kgf
```

```
F_kgf_SAFE =   105.4505
```

4.3.1.3 Verifying the profile bending load

The verification starts considering a simple horizontal strut fixed on the left (one degree freedom) and loose on the right (two degrees of freedom). The solution of the problem of finding the maximum bending deflection is very straightforward. The considered profile is the 40x80L oriented along the vertical edge, in this case Iy= 63.4 cm4, Ix= 17.3 cm4 and m= 2.7 kg/m.

The maximum bending deflection (fmax) is obtained from the equation:

$$fmax = \frac{F\,L^3}{48\,E\,Iy} + \frac{5\,(m{*}9.81)\,L^4}{384\,E\,Iy} \tag{4.5}$$

In current case **L=180 cm and F = 500 N at L/2**.

The result, considering also the mass per metre of the strut profile, is **fmax = 1.45 mm**. The maximum bending stress is **14.83 N/mm2** being inferior to the permissible bending stress of 97.50 N/mm2.

The following Matlab script performs the calculation.

```
Iy_cm4= 63.4 % cm4
m_kg_cm= 2.7/100 % kg/cm
E_N_cm2= 7E6 % N/cm2
L_cm= 180 % cm
F_N= 500 % N
a_cm= L_cm/2 % from left cm
b_cm= L_cm-a_cm % from right cm
fmax_cm= F_N*b_cm*(3*L_cm^2-4*b_cm^2)/...
    (48*E_N_cm2*Iy_cm4)+...
    5*(m_kg_cm*9.81)*L_cm^4/(384*E_N_cm2*Iy_cm4) % cm
fmax_mm= fmax_cm*10
```

```
fmax_mm =    1.4504
```

4.3.1.4 Prototype mass estimation

With reference to the prototype design (Table 4.1), the following mass prototype estimation can be done (Table 4.1).

Table 4.1. Prototype mass estimation.

			ESTIMATED TOTAL MASS (kg)	**56.83**
number	**item**	**length (mm)**	**linear mass (kg/m)**	**mass (kg)**
4	40x80L	620	3.06	7.59
4	40x40L	1400	1.5	8.40
4	40x40L	940	1.5	5.64
8	40x40L	850	1.5	10.20
			struts overall mass (kg)	**31.83**
number	**item**		**unit mass (kg)**	**mass (kg)**
4	20" wheel (fat tire)		2	8.00
2	motor 250W, 24V, 75 RPM		3	6.00
2	battery LiFePO4, 50 Ah		5.5	11.00
			other overall mass (kg)	**25.00**

4.3.2 Mechanical assembly

4.3.2.1 Left and Right parts

The dimensions and the assembly of the **left part** of **A-bot-X1** prototype is depicted in Figure 4.2.

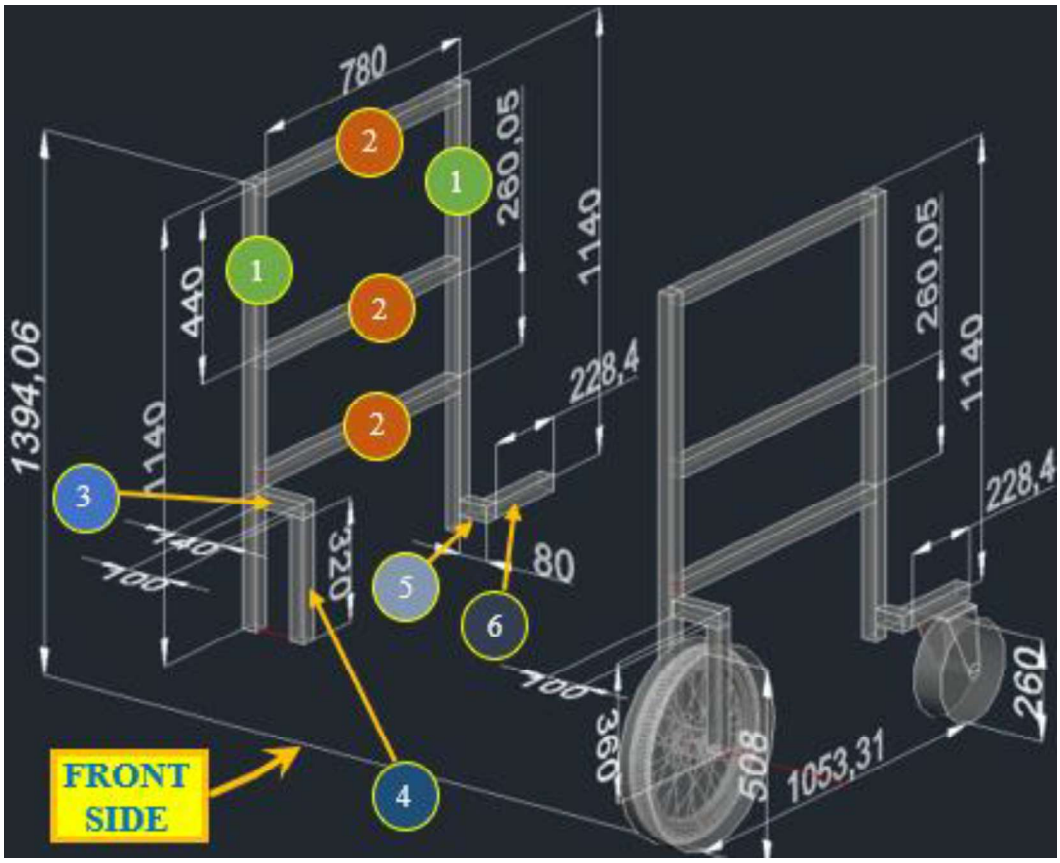


Figure 4.2. The dimensions and the assembly of the LEFT PART of the A-bot-X1 prototype (V11).

Required 40x40L struts for **EACH PART**:

(1) N.2 struts 1140 mm long
(2) N.3 struts 780 mm long
(3) N.1 strut 140 mm long
(4) N.1 strut 320 mm long
(5) N.1 strut 80 mm long
(6) N.1 strut 230 mm long

4.3.2.2 Upper frame

The dimensions and the assembly of the **upper frame part** of **A-bot-X1** prototype is depicted in Figure 4.3.

The wheel-to-wheel maximum distance is 1800 mm that can be shortened to 1800-2x240= 1320 mm or even less, up to 1304 mm if not considering the four upper connecting wedges.

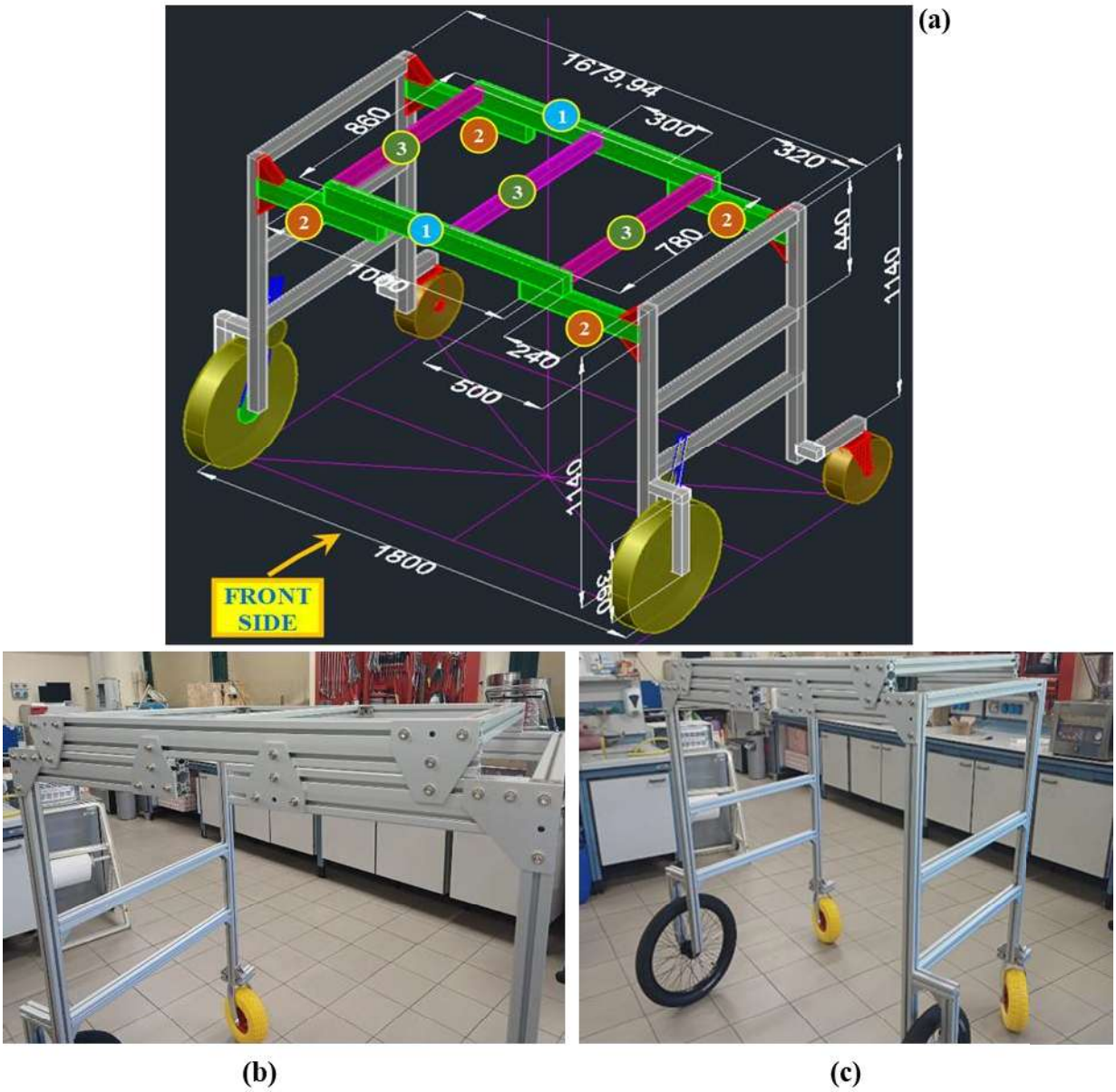


Figure 4.3. The dimensions (a) and the assembly (b,c) of the UPPER FRAME part of the A-bot-X1 prototype (V11).

Required 40x80L struts:

(1) N.2 struts 1000 mm long (40x80L)
(2) N.4 struts 500 mm long (40x80L)
(3) N.3 struts 780 mm long (40x40L)

4.3.2.3 Motor, encoder and wheel connection

The layout of the mechanical connection (chain) among electrical motor, encoder and wheel is shown in Figure 4.4.

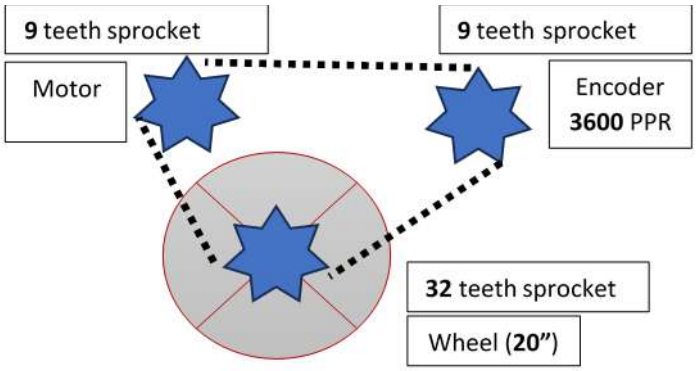


Figure 4.4. The layout of the mechanical connection (chain) among electrical motor, encoder and wheel.

The mechanical assembly of the left part of A-bot-X1 prototype is depicted in Figure 4.5.

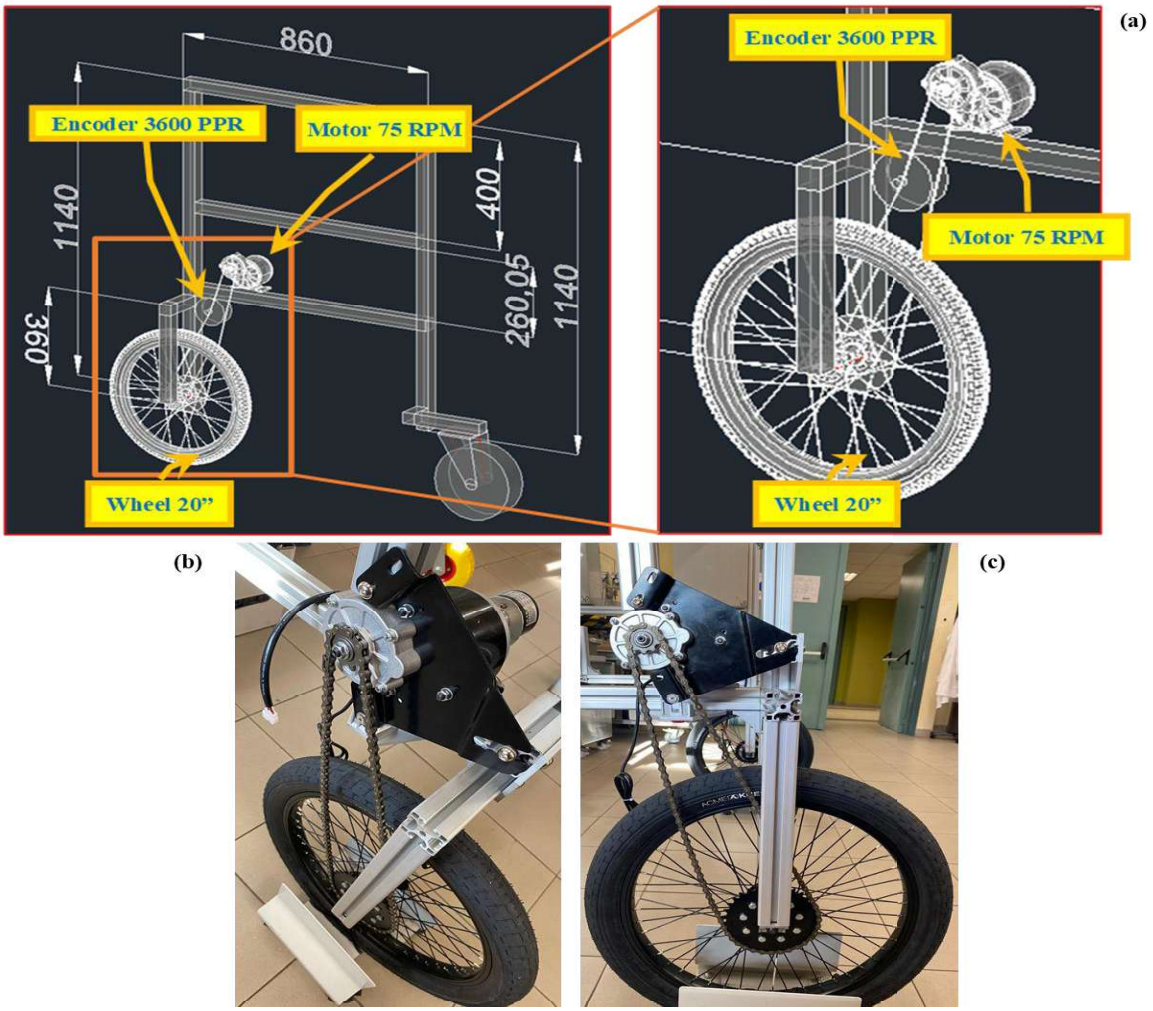


Figure 4.5. Mechanical assembly of the LEFT PART of the A-bot-X1 prototype (V11) (a). Focus on the layout of the mechanical connection (chain) between the electric motor, encoder and wheel (b,c).

4.3.3 Sensor and interface integration

4.3.3.1 Environmental and 4-20 mA Efento sensors

4.3.3.1.1 Scope and role within the system

The system integrates Efento wireless sensors for environmental monitoring and electrical signal acquisition, chosen for their robustness, low power consumption, and native Bluetooth Low Energy (BLE) communication. These sensors enable the real-time acquisition of key environmental and operational parameters relevant to precision agriculture, while supporting flexible deployment and straightforward integration within mobile and robotic platforms. In particular, Environmental Efento sensors are employed to monitor local microclimatic conditions

4.3.3.1.2 Sensor types and measured variables

Efento (https://getefento.com/) Bluetooth Low Energy (BLE) sensors measure various values and wirelessly send the data to smartphones or, through Efento Gateways, to Efento Cloud. Wireless loggers are equipped with large built in memory, and battery which ensures long maintenance free operation.

Large capacity battery ensures five years of maintenance free operation. Efento BLE sensors use standard batteries, which can be replaced by the user.

Sensor's range is 100 meters in open space and 30 meters in buildings.

Efento BLE sensor stores 40 000 measurements in its memory. This allows you to keep the measurements from a few months in device's memory and read it in a convenient moment. Moreover, if your sensor lost the connection with Efento Gateway for a while, after re-establishing it, it will send the missing data. User manual: https://drive.google.com/file/d/1fVfnbqGzr_sEAJExw0OkioXLGXC_xsze/view

Environmental sensor:

Efento wireless carbon dioxide (CO2) logger measures gas concentration and transmit its values over BLE. On top of the gas concentration, the logger also measures temperature, humidity and barometric pressure. Logger automatically calibrates the CO2 sensor based on the barometric pressure sensor and the concentration of CO2 in the atmosphere.

- CO2: 400 ppm – 5000 ppm, accuracy ±(40 ppm + 5 % of reading)
- Temperature: -15 to +70°C, accuracy: up to 0.4°C
- Humidity: 0 to 99% RH, accuracy up to 4% in the range of 0 to 80%
- Pressure: 300 to 1100 hPa, absolute accuracy +/- 3 hPa at 0 to + 65°C, relative +/- 0.12 hPa at 0 to + 65°C, p = 800 - 900 hPa

- Battery ensures up to 5 years of maintenance-free operation
- Measurement period: 1 minute – 10 days (configurable by the user)

4-20mA sensor:

Efento 4-20 mA sensor interface allows users to connect analog sensors to the cloud. Device can be equipped with analog interfaces (2 x 4-20 mA). Sensors can be powered either from the device's battery or by an external power source. Data from the analog sensors can be presented as current (4-20 mA) or converted to other measurement types once it reaches the Efento Cloud platform.

Analog input allows users to connect external 4-20 mA sensors. External sensors can be powered by the device or by external power supply.

- Battery ensures up to 5 years of maintenance-free operation
- Measurement period: 1 minute – 10 days (configurable by the user)
- Simultaneous operation with up to 2 external sensors: 2 x 4-20 mA
- Accuracy: 0.1% with factory default calibration, 0.01% with user 4 points calibration

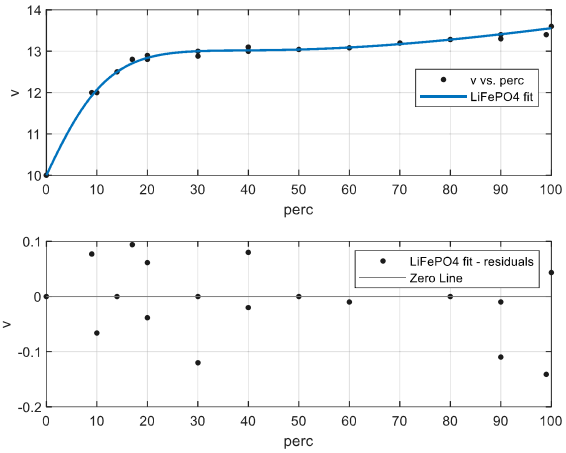
This sensor will be used to monitor the battery voltage that is nominally 12.8 V. The inner reading circuit has a 3 Ohm resistor in series. Therefore, when considering a series resistor of exactly 1500 Ohm. The following table (Table 4.2) summarizes the relationship between current readings (in mA), the external precision resistor (1.503 kΩ), and the resulting estimated battery voltage and charge percentage. The 4–20 mA analog sensor output, in combination with the resistor, provides an indirect measure of battery voltage, which is then mapped to a battery charge estimate (Table 4.2).

Table 4.2. Battery voltage and charge estimation from 4–20 mA sensor data

Current (mA)	Resistor (Ohm)	Voltage (V)		Charge Percent
4	1503	6.012	Range Min	
20	1503	30.060	Range Max	
8.516	1503	12.800	Nominal	
9.049	1503	13.60	Full Battery	100%
8.916	1503	13.40		90%
8.836	1503	13.28		80%
8.782	1503	13.20		70%
8.703	1503	13.08		60%
8.676	1503	13.04		50%
8.649	1503	13.00		40%
8.570	1503	12.88		30%
8.516	1503	12.80		20%
7.984	1503	12.00		10%
6.653	1503	10.000	Empty Battery	0%

The model of battery voltage vs. charge percent brings to:

```
General model Rat32:
fitresult(x) = (p1*x^3 + p2*x^2 + p3*x + p4) / (x^2 + q1*x + q2)
Coefficients (with 95% confidence bounds):
  p1 =     0.02355  (0.006455, 0.04065)
  p2 =       10.36  (7.841, 12.88)
  p3 =       154.1  (-297.6, 605.9)
  p4 =        4414  (-1179, 1.001e+04)
  q1 =       3.995  (-22.68, 30.67)
  q2 =       441.4  (-117.9, 1001)
sse: 0.0831
rsquare: 0.9959
dfe: 16
adjrsquare: 0.9946
rmse: 0.0721
```



To invert this function the interval is spl

inverse function is defined over these two inter

The Matlab function calculating the battery percent charge from the battery voltage is the following. The function result is coerced between 0% and 100%.

```
function res= EstimatePercentCharge(batvolt)

fL = @(vbat) ( 3.48011469e+00*vbat.^3 - 1.28543881e+02*vbat.^2 ...
               + 1.56843911e+03*vbat - 6.31127417e+03 ) ./ ...
             ( vbat.^2 - 2.63875363e+01*vbat + 1.74049010e+02 );

fU = @(vbat) ( 5.49521949e+01*vbat.^3 - 2.05529015e+03*vbat.^2 ...
               + 2.55852533e+04*vbat - 1.05994316e+05 ) ./ ...
             ( vbat.^2 - 2.58125299e+01*vbat + 1.66561740e+02 );

fPercCharge = @(vbat) fL(vbat).*(vbat<13.02) + fU(vbat).*(vbat>=13.02);

    res=fPercCharge(batvolt);
    res(res<0)=0;
    res(res>100)=100;

end
```

The same function in JavaScript is listed below.

```
function EstimatePercentCharge(batvolt) {
        let res;
        if (batvolt < 13.02) {
            res = (3.48011469e+00 * Math.pow(batvolt, 3)
                   - 1.28543881e+02 * Math.pow(batvolt, 2)
                   + 1.56843911e+03 * batvolt
                   - 6.31127417e+03)
                   / (Math.pow(batvolt, 2)
                   - 2.63875363e+01 * batvolt
                   + 1.74049010e+02);
        } else {
            res = (5.49521949e+01 * Math.pow(batvolt, 3)
                   - 2.05529015e+03 * Math.pow(batvolt, 2)
                   + 2.55852533e+04 * batvolt
                   - 1.05994316e+05)
                   / (Math.pow(batvolt, 2)
                   - 2.58125299e+01 * batvolt
                   + 1.66561740e+02);
        }
```

```
        // Clamp result between 0 and 100
        if (res < 0) res = 0;
        if (res > 100) res = 100;
        return res;
}
```

4.3.3.1.3 Sensor interface: Efento Bluetooth Low Energy sensors – decoding advertising data

All Efento sensors (both Bluetooth Low Energy and NB-IoT) are equipped with a Bluetooth interface. This allows mobile phones, BLE gateways and custom hardware to communicate with the sensors, read the measurements taken by them and/or change their configuration.

Bluetooth defines two transmissions types: data and advertising transmissions. Bluetooth advertising transmission is used by BLE equipped devices to broadcast useful data to others around. Bluetooth Low Energy advertising packets are used to initiate the communication between the devices (e.g., a smartwatch that wants to connect to a mobile phone) or simply send a small message to nearby devices (e.g. a wireless sensor broadcasting temperature to anyone who wants to listen).

When using the advertising transmission, no connection is established between the devices – any Bluetooth Low Energy can receive the advertising packets sent by the others. As the data is sent in a broadcast, the messages do not need to be acknowledged by the receiving side. Advertising packet structure is defined by BLE standard, and you can find its full specification here.

Efento sensors use Bluetooth advertising transmissions to share the most recent measurement and some important technical information with the nearby devices. Each advertisement data packet contains: the current measurement along with its type, information about battery level, software version, calibration date (optional) and measurement period. The advertising packets sent by Efento sensors can be encrypted, so only devices with a proper encryption key will be able to decrypt and parse the payload.

Efento sensors broadcast the advertisement packets by default and it is impossible to turn off these transmissions (it is possible to completely disable Bluetooth, but as long as it's enabled the device will broadcast the advertisement packets). The data is received by all the nearby BLE equipped devices (phones, tablets, laptops, etc.).

The in-depth description continues here: https://drive.google.com/file/d/1nw6-ifto49XFHjwEn7-8UevybnAW0CYj/view

4.3.3.2 The Distance Measurement System Development

4.3.3.2.1 Role within the system

This section describes the development of the distance measurement system integrated into the platform, aimed at providing reliable and real-time range information to support navigation, obstacle detection, and operational safety. The system is designed to be compact, energy-efficient, and easily integrable within mobile and robotic architectures, ensuring robustness under variable environmental and operational conditions.

4.3.3.2.2 TFmini-S LiDAR Sensor

The identified hardware is the sensor: **TFmini-S Lidar Sensor 0.1-12m** allowing for "Distance measurement Single point module compatible with Pixhawk and Raspberry Pi for Drone/Motion Detection/Robot" (https://www.amazon.it/dp/B08FFFFQ65)

The supplied software (see **BW_TFDS_x86_V1.4.0_20200911_Beta** into development companion compressed folder **tfmini-s-lidar.zip**) allows for its rapid setup (Figure 4.6).

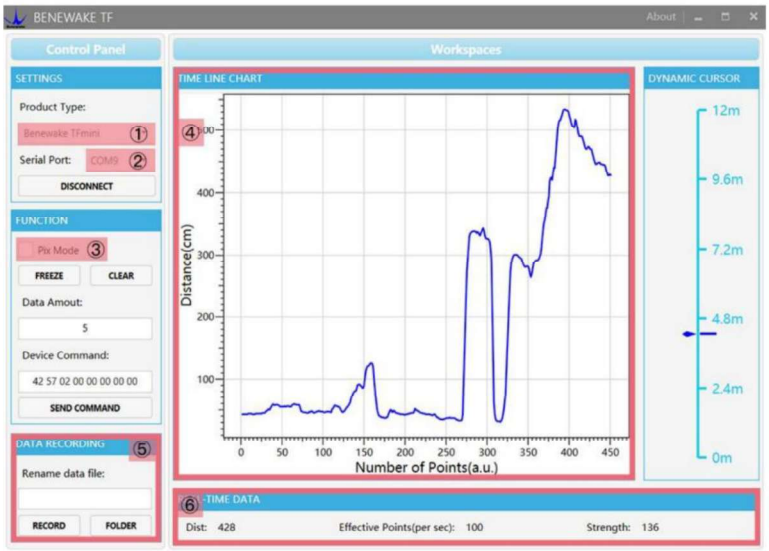


Figure 4.6. TFmini-S Lidar Sensor software for the rapid device setup

4.3.3.2.3 TFmini-S LiDAR Sensor Interface

Arduino Nano 33 IoT

Preliminarily, an Arduino Nano 33 IoT [ABX00027] device (https://www.amazon.it/dp/B07VW9TSKD) was used to develop a standalone distance measuring device. The connection was performed through serial port using an inhouse developed Arduino's script (see **arduino_nano_dist_test.ino** into supporting scripts companion folder, here the link https://tinyurl.com/27r6j3od) (Figure 4.7).

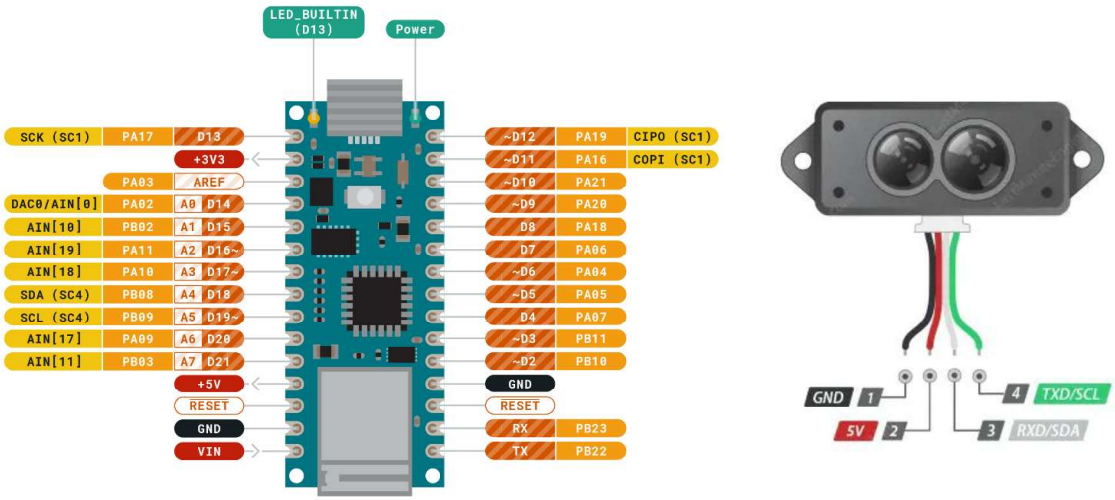


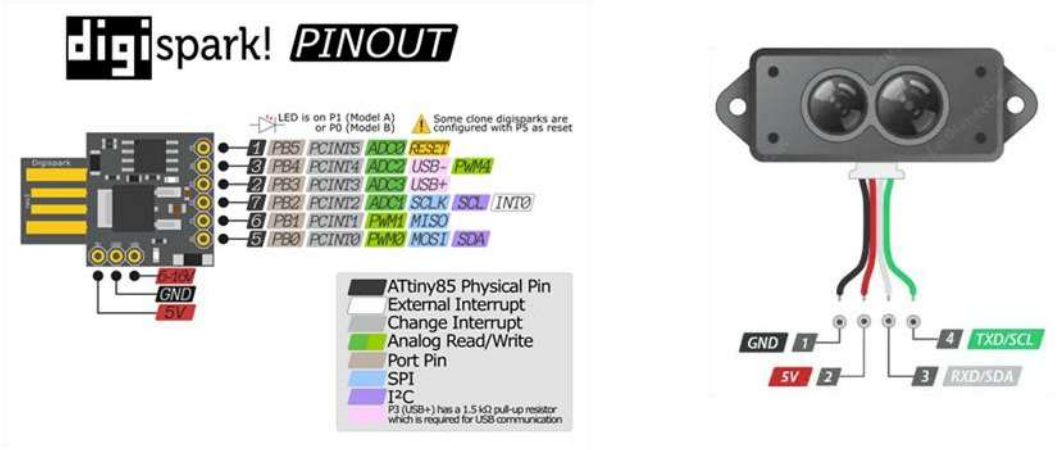
Figure 4.7. Pinout diagram of the Arduino Nano 33 IoT

```
// pin5 as RX (to TXD cable GREEN of LiDAR)
// pin6 as TX (to RXD cable WHITE of LiDAR)
#define rxpin 5
#define txpin 6
```

However, the software developed was not fully satisfactory as it required the subsequent tailoring of piece of software to link all the system together with the master system.

Digispark ATtiny85 USB device (direct connection to /dev/ttyACM0 port)

The subsequent testing (**Errore. L'origine riferimento non è stata trovata.**8) doesn't s howed useful enhancement of speed and overall management. For this reason, this development was abandoned because, even this solution, required subsequent tailoring of piece of software to link all the system together with the master system.



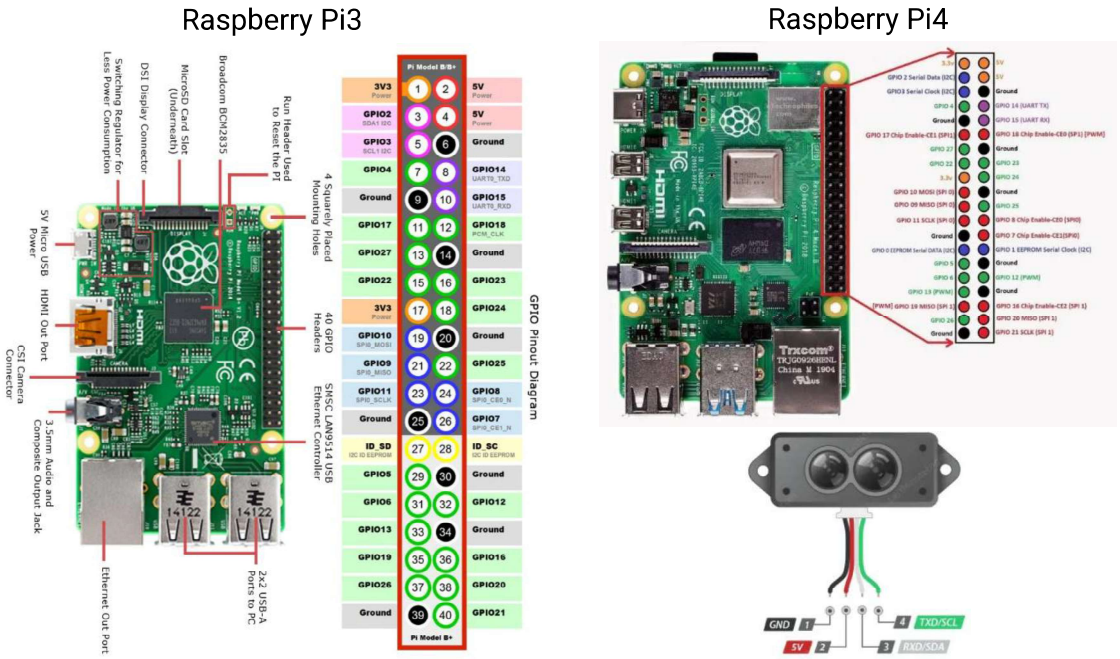

```
// pin2 as RX (GREEN TXD cable of LiDAR)
// pin3 as TX (WHITE RXD cable of LiDAR)
```

Figure 4.8. Pinout diagram of the Digispark ATtiny85 USB device.

Direct connection to /dev/ttyAMA0 port of Raspberry Pi

At this point, considering that Raspberry Pi4 has up to six usable serial ports, the direct connection of the TFmini-S Lidar Sensor to the Raspberry Pi hardware has been explored.

To this aim, the pinout diagram of the Pi4 has been preliminarily collected. However, the Pi3 has been preliminary considered, being less expensive and immediately available because already owned by the task investigator. Some tricks have been implemented into the startup code in order to free and make available the **/dev/ttyAMA0** system resource, so disabling the Bluetooth device. In this case, the GREEN TXD cable of LiDAR sensor was connected to GPIO_15 (UART0_RXD or UART RX) and the WHITE RXD cable of LiDAR sensor was connected to GPIO_14 (UART0_TXD or UART TX). The LiDAR sensor serial pins are 3.3 Volt level compatible (Figure 4.9) and were directly connected to the correspondent pins of the Raspberry Pi.



The GREEN TXD cable of LiDAR sensor is connected to GPIO_15 (UART0_RXD or UART RX) and the WHITE RXD cable of LiDAR sensor is connected to GPIO_14 (UART0_TXD or UART TX)

Figure 4.9. Pinout diagram of the Raspberry Pi.

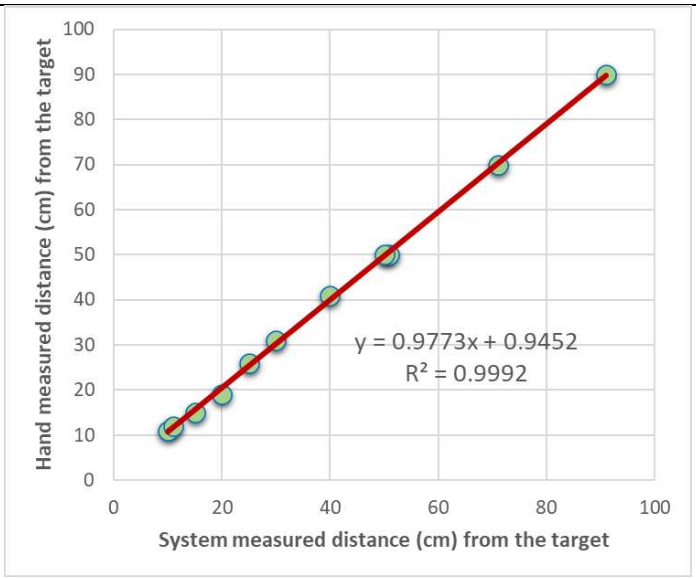
The calibration of the distance system used the LiDAR sensor as reference basepoint (**Errore. L'origine riferimento non è stata trovata.**), the calibration results are listed in T able 4.3.

Figure 4.10.Reference basepoint of the measuring distance LiDAR system

To assess the accuracy and linearity of the TFmini-S LiDAR sensor, a set of controlled distance measurements was performed. The sensor was placed at fixed distances from a flat, non-reflective target in a static indoor environment. For each position, the actual distance (measured manually) was compared with the value reported by the sensor. The results, shown in Table 4.3, include the real and measured distances, the absolute difference, and the percentage error. These data confirm the stability and reliability of the sensor across its operating range, with particularly low deviations observed between 30 cm and 500 cm, where the relative error consistently remained under 5%.

Table 4.3. LiDAR sensor calibration results.

Hand measured distance (cm) from the target	System measured distance (cm) from the target
71	70
91	90
51	50
10	11
11	12
15	15
20	19
25	26
30	31
40	41
50	50



The calibration data show an offset of 1 cm and an impressive measuring performance of the LiDAR sensor.

4.3.3.3 Distance-Based spectral alignment: A proof-of-concept approach

As discussed in Section 2.4.2, one of the main issues encountered with multispectral sensors is the presence of spatial misalignments among spectral bands, which become more pronounced under short sensor-to-target distance conditions. Consequently, it is essential to implement a system that not only acquires data correctly but is also capable of adapting the alignment correction according to the instantaneous distance from the target.

4.3.3.3.1 LiDAR-Based Compensation and Raspberry Pi Processing

In the initial phase of the project, a proof-of-concept was developed using a MicaSense RedEdge-M multispectral camera and a TFmini-S LiDAR distance sensor. The objective was to compute, through a manual calibration procedure, the spatial offsets among spectral bands as a function of the measured distance. The image registration method is implemented by creating a transformation matrix to completely minimize the error in the alignment of the captured images close to the target object.

To preliminarily demonstrate the proof of concept, a MicaSense multi-spectral camera has been used (RedEdge-M, 6 mm focal length) at the shots have been taken from an height of 1500 mm.

The shots have been manually segmented (**go_calibrate_micasense.m** available here https://drive.google.com/drive/folders/1wgOIxi_sCWhURPJU31n5xvp4fkcgzaTT) using

MATLAB's Image Segmenter App (Figure 4.11) in order to identify the alignment pattern consisting of a symmetric circles pattern equally spaced of 50 mm.

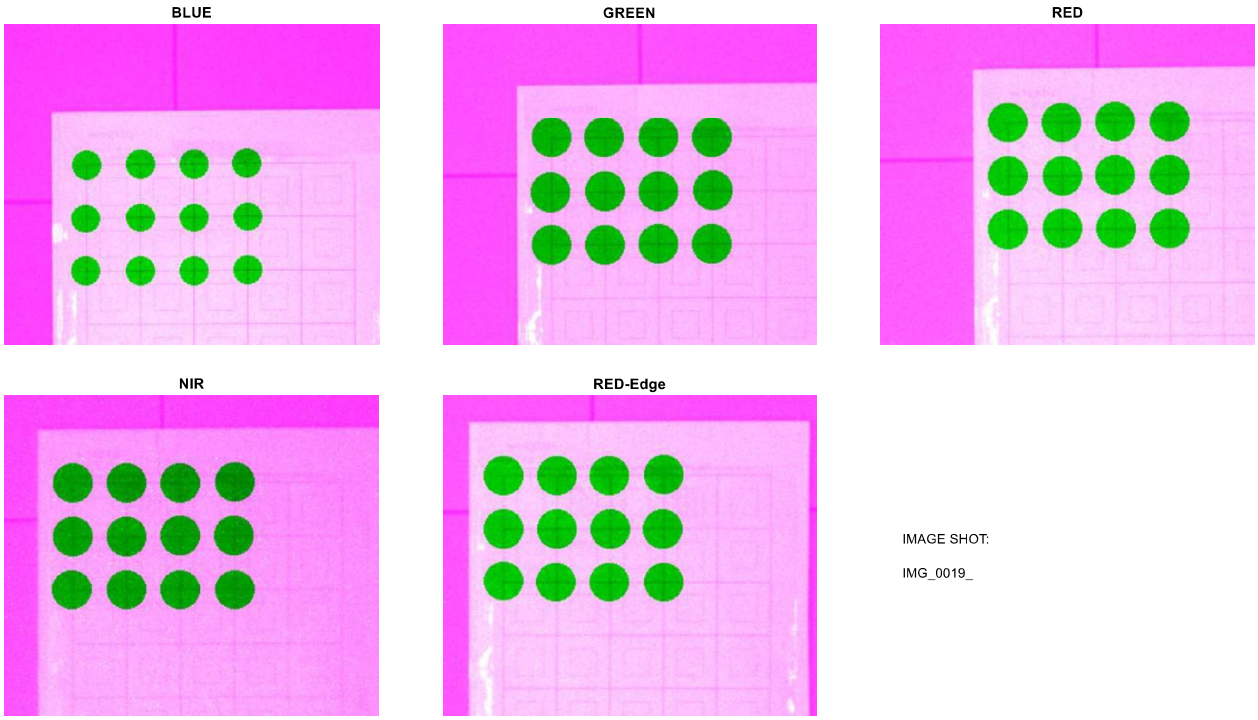


Figure 4.11. Image segmentation and circles pattern creation.

The alignment of the channels of the image are performed considering one channel as the fixed image (e.g. the Red channel in our case) and then the row and column offset are calculated of all the other channel with respect to the one considered as fixed. The Matlab algorithm of pattern detection in images is used and, in particular, the **detectPatternPoints** Matlab function. The script developed (**go_calibration_apply_micasense.m** available into website supporting scripts companion folder https://tinyurl.com/24cdtx27) shows that the image has been correctly aligned (Figure 4.12).

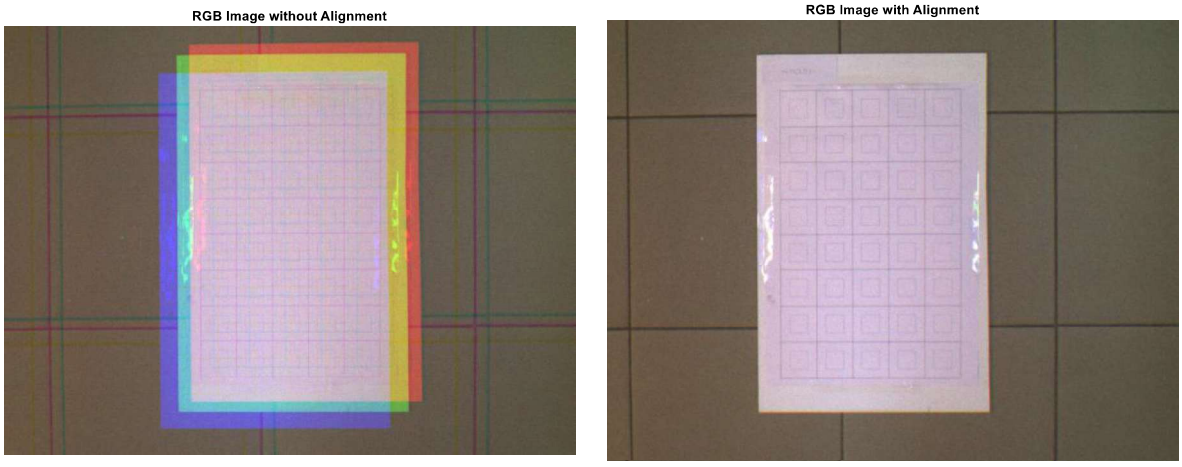


Figure 4.12. Image aligned considering the Red channel as the fixed image.

In addition, the developed script demonstrates the feasibility of the idea behind this type of alignment.

Once the camera alignment is determined at different distance from the target, thus obtaining the different offsets to be used, that change with the distance of the target, then, the measured target distance by the LiDAR sensor can be used to align on the fly the multi-spectral camera image making it ready for further analysis. To make the system portable and suitable for real-time execution, the same scripts were subsequently adapted to run on an embedded architecture based on a Raspberry Pi , equipped with a Linux operating system and MATLAB Runtime tools, as shown in Figure 4.13.

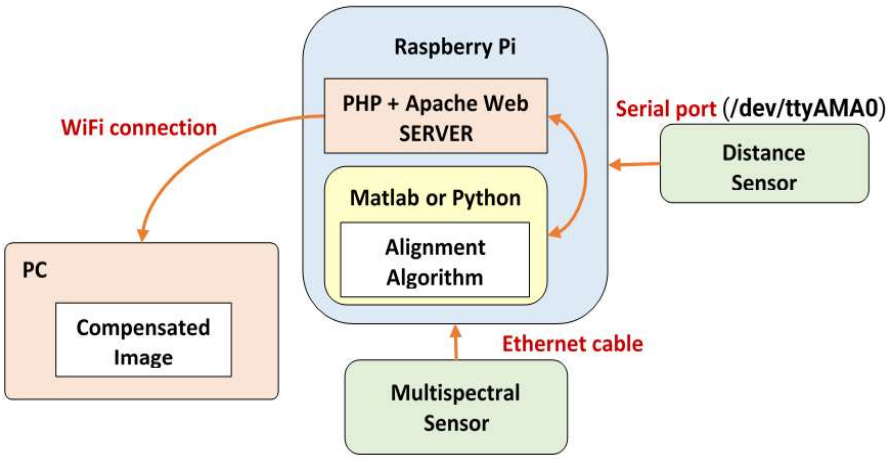


Figure 4.13. Planning of the working path considering the major calculation effort to be performed on a Raspberry Pi hardware deploying the Matlab scripts.

4.3.3.3.2 Network Interfaces and FTP/Web Access

The **MicaSense RedEdge-MX camera** supports remote network access via Ethernet through the following interfaces:

- **Web interface**: accessible via a standard web browser by entering the camera IP address (e.g., http://192.168.1.x). This interface allows preview visualization, configuration of acquisition sessions, and manual download of acquired data.
- **FTP access**: enables automatic or batch transfer of acquired images through an authenticated FTP client, with directories structured by spectral band and associated metadata.
- **HTTP GET requests**: the device supports remote commands for triggering acquisitions and querying the operational status.

These features make the MicaSense camera fully integrable into automated acquisition pipelines, such as deployments on **Raspberry Pi** platforms, and facilitate data

synchronization with other sensors (e.g., **TFmini-S LiDAR**) through timestamp-based alignment or file-based matching.

4.4 Software architecture

4.4.1 Overview of the Software Architecture

This section presents the overall software architecture developed for the system. For each hardware component integrated into the platform, the corresponding software interfaces are described, including the operating system configuration, the ROS 2 framework, sensor integration modules, data acquisition pipelines, control logic (e.g., navigation, image processing), and communication interfaces.

The architecture was designed following a modular and scalable approach, allowing individual hardware and software components to be developed, tested, and extended independently. Particular attention was devoted to ensuring real-time execution capabilities on embedded hardware, enabling reliable operation in field conditions and facilitating future integration of additional sensors, control modules, and decision-support functionalities

4.4.2 Environmental and 4-20 mA Efento sensors

4.4.2.1 Bluetooth Communication and Python Interface

The communication with the Efento sensors was implemented through a **Bluetooth Low Energy (BLE) interface**, using a custom Python-based software layer specifically developed for this work. The objective was to design a **flexible, modular, and robust acquisition interface** capable of handling heterogeneous sensor types and operating reliably on embedded hardware.

The BLE communication was managed through the *bluepy* library, which provides low-level access to BLE devices. To enable BLE communication with Efento sensors on a Linux-based embedded system, a preliminary software configuration was required. The following steps summarize the installation and permission setup adopted in this work:

a. Preliminary installation:

```
pip install blupy
cd ~/.local/lib/python3.10/site-packages/bluepy
sudo setcap 'cap_net_raw,cap_net_admin+eip' bluepy-helper
```

This allows the **bluepy-helper** executable to be run with encapsulated super user privileges from any caller running at user level.

A dedicated Python-based software layer was developed to manage BLE communication, sensor discovery, and data acquisition. The scripts were designed to support multiple sensor types while maintaining a unified acquisition interface.

b. Python scripts and Bluetooth:

The sensors' communication is handled by some dedicated scripts in Python stored into **~/efento_support** (available here https://sites.google.com/unibas.it/smartagri/code-repository) folder:

- get_sensor_readings.py (the main script)
- efento_sensors_final.py (the class script)

To validate the flexibility and configurability of the acquisition interface, the main script provides a command-line interface (CLI) supporting multiple runtime options, as shown below:

```
ubuntu@ubuntu:~$ python ~/efento_support/get_sensor_readings.py -h
usage: get_sensor_readings.py [-h] [-a ADAPTER] [-s SELECT] [-l] [-w
WAITINGTIME] [-i] [-d]

Read Efento Sensors

options:
  -h, --help            show this help message and exit
  -a ADAPTER, --adapter ADAPTER
                        Select the BT ADAPTER (default: hci0)
  -s SELECT, --select SELECT
                        Select a sensor from the available list (default:
0)
  -l, --list            Show the available sensors LIST and exit
  -w WAITINGTIME, --waitingtime WAITINGTIME
                        Set the waiting time for reading (default: 20)
  -i, --info            Show INFO log
  -d, --debug           Show DEBUG log

! Available sensors:
! 0, 28:2C:02:42:4A:0A, Environmental (Efento)
! 1, 28:2C:02:42:4A:0B, Environmental (Efento)
! 2, 28:2C:02:42:4A:0C, mA Current (Efento)
! 3, 28:2C:02:42:4A:0D, mA Current (Efento)
! 4, 28:2C:02:41:D4:56, Soil moisture (Efento)
! Example: [--adapter hci0] -select 0 --waitingtime 20 [--info] [--debug]
[--list]
! Example: [-a hci0] -s 0 -w 20 [-i] [-d] [-l]
```

The correct functioning of the BLE communication and data parsing pipeline was verified through real sensor acquisitions. The following examples report the output obtained from both environmental and 4–20 mA Efento sensors.

The simple call: **`python ~/efento_support/get_sensor_readings.py -s 1 -w 5`** allows to collect the data advertised by the sensor identified as '1' corresponding to the following dictionary entry:

```
  '1': {
      'name': 'Environmental (Efento)',
      'serial': "28:2C:02:42:4A:0B",
      'type': bytearray([1,2,3,0x1A])
  }
```

listed into the **get_sensor_readings.py** script, using a waiting time of 5 seconds.

The obtained answers are something like this:

```
python ~/efento_support/get_sensor_readings.py -s 1 -w 5
*SERIAL, 28:2C:02:42:4A:0B
*HEXCODETYPE, 0102031A
*DATETIME, 2025-04-26 12:01:43
*TEMPERATURE, 22.600, °C
*HUMIDITY, 36.000, %
*ATMOSPHERIC_PRESSURE, 934.700, hPa
*CO2_GAS, 317.000, ppm

python ~/efento_support/get_sensor_readings.py -s 2 -w 5
*SERIAL, 28:2C:02:42:4A:0C
*HEXCODETYPE, 2929
*DATETIME, 2025-04-26 12:03:31
*CURRENT, 0.000, mA
*CURRENT, 0.000, mA

~/call_installed_efento_sensors.sh
*SERIAL, 28:2C:02:42:4A:0B
*HEXCODETYPE, 0102031A
*DATETIME, 2025-04-26 12:05:27
*TEMPERATURE, 22.600, °C
*HUMIDITY, 36.000, %
*ATMOSPHERIC_PRESSURE, 934.700, hPa
*CO2_GAS, 317.000, ppm
*SERIAL, 28:2C:02:42:4A:0C
*HEXCODETYPE, 2929
*DATETIME, 2025-04-26 12:05:35
*CURRENT, 0.000, mA
*CURRENT, 0.000, mA
```

c. Work space creation for Efento Sensors (Environmental and 4-20 mA sensor)

```
# package creation
cd ~
mkdir ~/efento_ws
mkdir ~/efento_ws/src
cd ~/efento_ws/src
ros2 pkg create --build-type ament_python --node-name efento_node efento_pkg
# custom message building
ros2 pkg create --build-type ament_cmake my_custom_interfaces
# package building
cd ~/efento_ws
colcon build
# package sourcing
source install/local_setup.bash
# to be inserted into ~/.bashrc
source ~/efento_ws/install/setup.bash
# node launch
ros2 run efento_pkg efento_node
```

To integrate the Bluetooth-based Efento sensor acquisition into the overall robotic software architecture, a dedicated ROS 2 workspace was created. This step allows the sensor data to be handled as standardized ROS messages, enabling seamless interaction with other system components such as navigation, control, and monitoring modules.

d. Packages implementation in Python (efento_ws/src)

To support the transmission of heterogeneous sensor data within a single communication channel, a custom ROS 2 message was defined. The message encapsulates environmental measurements, battery information, and diagnostic timeout flags.

Contents of **./my_custom_interfaces/msg/Efentos.msg**

```
string now ""
bool env_timeout True
bool bat_timeout True
float64 temperature
float64 humidity
float64 atmospheric_pressure
float64 co2_gas
float64 battery_1
float64 battery_2
```

A dedicated ROS 2 publisher node was implemented to periodically execute the BLE acquisition scripts, parse the returned sensor data, manage error and timeout conditions, and publish the results on a dedicated ROS topic.

Contents of **./efento_pkg/efento_pkg/efento_node.py**

```
from numpy import NaN
import rclpy
from rclpy.node import Node
from my_custom_interfaces.msg import Efentos
import subprocess
import sys
import os
from datetime import datetime

INVALID_VAL= NaN

class MyEfentoSensors_Pub(Node):

    def __init__(self):
        super().__init__('my_efento_sensors_pub')
        self.publisher_ = self.create_publisher(Efentos, 'efento_sensors', 5)
        timer_period = 5.0  # seconds
        self.timer = self.create_timer(timer_period, self.timer_callback)
        date_time= datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        print(f"* {date_time} - MyEfentoSensors_Pub node started ...")

        self.now = ""
        self.env_timeout = True
        self.bat_timeout = True
        self.temperature = INVALID_VAL
        self.humidity = INVALID_VAL
        self.atmospheric_pressure = INVALID_VAL
```

```
        self.co2_gas = INVALID_VAL
        self.battery_1 = INVALID_VAL
        self.battery_2 = INVALID_VAL

    def timer_callback(self):
        msg = Efentos()
        msg.now = self.now
        msg.temperature = self.temperature
        msg.humidity = self.humidity
        msg.atmospheric_pressure = self.atmospheric_pressure
        msg.co2_gas = self.co2_gas
        msg.battery_1 = self.battery_1
        msg.battery_2 = self.battery_2

        execfile= os.path.expanduser('~/call_installed_efento_sensors.sh')
        any_error = False
        try:
            res = subprocess.run(args=[execfile],capture_output=True, text=True)
            output = res.stderr+res.stdout

        except subprocess.CalledProcessError as e:
            any_error = True
            output = e.stderr

        print(output)

        RESISTOR_KOHM = 1.503

        if not any_error:
            # Parse the output
            lines = output.split('\n')
            bat1 = True
            gotaval = False
            env_timeout = True
            bat_timeout = True
            for line in lines:
                if line.startswith('*TEMPERATURE'):
                    temperature = float(line.split(',')[1].strip())
                    msg.temperature = temperature
                    self.temperature = temperature
                    gotaval=True
                    env_timeout = False
                elif line.startswith('*HUMIDITY'):
                    humidity = float(line.split(',')[1].strip())
                    msg.humidity = humidity
                    self.humidity = humidity
                    gotaval=True
                    env_timeout = False
                elif line.startswith('*ATMOSPHERIC_PRESSURE'):
                    atmospheric_pressure = float(line.split(',')[1].strip())
                    msg.atmospheric_pressure = atmospheric_pressure
                    self.atmospheric_pressure = atmospheric_pressure
                    gotaval=True
                    env_timeout = False
                elif line.startswith('*CO2_GAS'):
                    co2_gas = float(line.split(',')[1].strip())
                    msg.co2_gas = co2_gas
                    self.co2_gas = co2_gas
                    gotaval=True
                    env_timeout = False
                elif line.startswith('*CURRENT'):
                    batval = float(line.split(',')[1].strip())
                    # convert CURRENT to VOLTAGE
                    batval = batval * RESISTOR_KOHM
                    if bat1:
                        msg.battery_1 = batval
                        self.battery_1 = batval
                        bat1=False
```

```
                        gotaval=True
                        bat_timeout = False
                    else:
                        msg.battery_2 = batval
                        self.battery_2 = batval
                        gotaval=True
                        bat_timeout = False

        date_time= datetime.now().strftime("%Y-%m-%d %H:%M:%S")

        if env_timeout:
            print(f"- {date_time} >>> Environment sensors timeout")
        if bat_timeout:
            print(f"- {date_time} >>> Battery sensors timeout")
        else:
            print(f"- Conversion Resistor: {RESISTOR_KOHM} kOhm")

        gotaval= True
        if not gotaval:
            print(f"- {date_time} >>> No valid data to publish")
        else:
            msg.env_timeout = env_timeout
            self.env_timeout = env_timeout
            msg.bat_timeout = bat_timeout
            self.bat_timeout = bat_timeout
            msg.now = date_time
            self.now = date_time
            # msg.header.stamp = self.get_clock().now().to_msg()
            self.publisher_.publish(msg)
            print(f"- Published: {msg}")

def main(args=None):
        rclpy.init(args=args)
        my_efento_sensors_pub = MyEfentoSensors_Pub()
        rclpy.spin(my_efento_sensors_pub)
        # my_efento_sensors_pub.destroy_node()
        rclpy.shutdown()

if __name__ == '__main__':
    main()
    print('*** Ended MyEfentoSensors_Pub')
```

e. Package launch

~/go_efento.sh (content: ros2 run efento_pkg efento_node)

```
* 2025-04-26 12:13:27 - MyEfentoSensors_Pub node STARTED ...
*SERIAL, 28:2C:02:42:4A:0B
*HEXCODETYPE, 0102031A
*DATETIME, 2025-04-26 12:13:41
*TEMPERATURE, 23.000, °C
*HUMIDITY, 35.000, %
*ATMOSPHERIC_PRESSURE, 934.600, hPa
*CO2_GAS, 274.000, ppm
*SERIAL, 28:2C:02:42:4A:0C
*HEXCODETYPE, 2929
*DATETIME, 2025-04-26 12:13:51
*CURRENT, 0.000, mA
*CURRENT, 0.000, mA

- Conversion Resistor: 1.503 kOhm
- Published: my_custom_interfaces.msg.Efentos(now='2025-04-26 12:13:51',
env_timeout=False, bat_timeout=False, temperature=23.0, humidity=35.0,
atmospheric_pressure=934.6, co2_gas=274.0, battery_1=0.0, battery_2=0.0)
```

This output confirms the correct end-to-end operation of the Efento sensor integration pipeline, from BLE communication and data parsing to ROS 2 message publication on the embedded platform.

4.4.2.2 ROS2 Bridge Suite proof of concept

To demonstrate the interoperability of the ROS 2 ecosystem with external applications, a proof-of-concept integration was developed using the rosbridge_suite. The goal was to enable real-time visualization of the Efento sensor data through a standard WebSocket interface, making the ROS 2 topics accessible to web-based clients without requiring ROS-specific tooling on the user side.

After installing, using **`sudo apt install ros-humble-rosbridge-server`**

Then, just execute:

1) ros2 launch rosbridge_server rosbridge_websocket_launch.xml (**aka ~/go_ros2bridge.sh**)
2) **~/go_efento.sh**

Subsequently, connect to the IP of the server: **192.168.1.13** (in our case) using the port **9090**. Once the **rosbridge_server is running**, a client can subscribe to ROS 2 topics via WebSocket and decode the streamed messages in real time.

The subscription is on the **/efento_sensors** topic. The code sample is very straightforward. JavaScript language is used in this case.

The link to the sample code is here: [rosbridge_proof_of_concept/index.html.]

The Figure 4.14 shows a web-based proof-of-concept dashboard built using the rosbridge_server and JavaScript. It visualizes real-time data streamed over the /efento_sensors topic through a WebSocket connection to the ROS 2 network (IP: 192.168.1.13, Port: 9090). The gauges represent sensor readings from Efento BLE devices, including current (mA), voltage (V), estimated battery charge (%), ambient temperature (°C), humidity (%), CO2 concentration (ppm), and atmospheric pressure (hPa). This visualization demonstrates the interoperability between ROS 2 and modern web technologies.

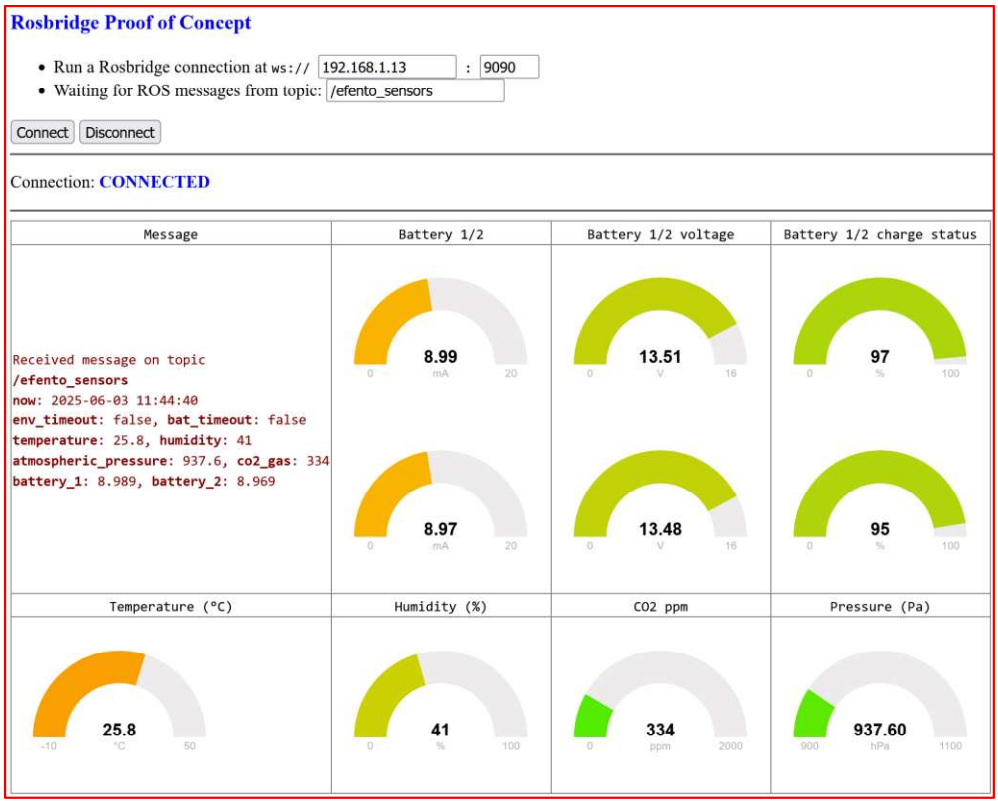


Figure 4.14. Real-time sensor monitoring via ROS2 WebSocket Dashboard

This proof-of-concept confirms that the sensor acquisition pipeline can be extended beyond the ROS environment, enabling lightweight monitoring tools and facilitating remote access during field operations.

4.4.2.3 Raspberry Pi 4 Device Telemetry over the ROS 2 Network

In addition to sensor streaming, a lightweight telemetry module was developed to monitor the embedded device status over the ROS 2 network. This functionality was introduced to support debugging and multi-device deployments by providing a simple mechanism to identify each Raspberry Pi unit and publish basic health metrics during operation.

The device ID (as an integer) is stored, as temporary file, into the file **/dev/shm/THIS_RP_DEVICE_NAME**, residing into the temporary folder (**/dev/shm**) using the ram file system.

To ensure unique topic names across multiple devices, the numeric hardware ID is mapped to an alphabetical suffix, generating topics such as rp_telemetry_a, rp_telemetry_b, etc.

GPIO26, GPIO22, GPIO27 are used as inputs (with pullup active) to hardcode the number identifying the RP device (1 to 8). 0, 0, 0 identifies RP1, 0, 0, 1 identifies RP2, and so on.

A service, started at startup, read the pins and store the identification number into **/dev/shm/THIS_RP_DEVICE_NAME**.

Because the topic name cannot be a number, consequently, the number is converted to a letter.

Therefore, the emitted topic will be “rp_telemetry_a” for the ID=1 and so on.

To integrate this functionality within ROS 2, a dedicated workspace was created including a custom message interface and a Python-based publisher node.

a. Work space creation for device telemetry

```
# package creation
cd ~
mkdir ~/this_rp_device_telemetry_ws
mkdir ~/this_rp_device_telemetry_ws/src
cd ~/this_rp_device_telemetry_ws/src
ros2 pkg create --build-type ament_python --node-name rp_telemetry_node
rp_telemetry_pkg
# custom message building
ros2 pkg create --build-type ament_cmake rp_telemetry_interfaces
# package building
cd ~/this_rp_device_telemetry_ws
colcon build
# package sourcing
source install/local_setup.bash
# to be inserted into ~/.bashrc
source ~/this_rp_device_telemetry_ws/install/setup.bash
# node launch
ros2 run rp_telemetry_pkg rp_telemetry_node
```

After building and sourcing the workspace, the telemetry node can be executed to publish periodic status messages on the device-specific topic.

b. Package implementation in Python (this_rp_device_telemetry_ws/src)

A minimal custom message (RpTelemetry.msg) was defined to transmit a timestamp and a representative telemetry variable (CPU temperature), providing a compact monitoring stream with low computational overhead.

Contents of **./rp_telemetry_interfaces/msg/RpTelemetry.msg**

```
string now ""
float64 temperature
```

The publisher node reads the device ID from /dev/shm/THIS_RP_DEVICE_NAME, selects the appropriate topic name, and publishes telemetry messages at a fixed rate (2 s). CPU temperature is retrieved from the Linux thermal interface to provide a simple health indicator.

Contents of **./rp_telemetry_pkg/rp_telemetry_pkg/rp_telemetry_node.py**

```
import rclpy
from rclpy.node import Node
from rp_telemetry_interfaces.msg import RpTelemetry
from datetime import datetime
```

```
class RpTelemetrySensors_Pub(Node):
    def __init__(self):
        super().__init__('rp_telemetry_sensors_pub')
        id_num=self.id_number()

        if id_num == 0:
            self.topic_name = 'rp_telemetry_noid'
        else:
            self.topic_name = 'rp_telemetry_' + chr(ord('a') + id_num - 1)

        self.publisher_ = self.create_publisher(RpTelemetry, self.topic_name, 5)
        timer_period = 2.0  # seconds
        self.timer = self.create_timer(timer_period, self.timer_callback)
        date_time= datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        print(f"* {date_time} - Publisher created for topic: {self.topic_name}")
        print(f"* {date_time} - RpTelemetrySensors_Pub node started ...")
        print(f"* {date_time} - Publishing RpTelemetry messages every
{timer_period} seconds ...")

    def timer_callback(self):
        msg = RpTelemetry()
        msg.now= datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        msg.temperature = self.get_cpu_temperature()
        self.publisher_.publish(msg)
        print(f"- ({self.topic_name}) {msg}")

    def id_number(self):
        try:
            with open("/dev/shm/THIS_RP_DEVICE_NAME", "r") as f:
                id_str = f.readline()
                return int(id_str)
        except Exception as e:
            print("* Error: Unable to read device ID from
/dev/shm/THIS_RP_DEVICE_NAME")
            return 0

    def get_cpu_temperature(self):
        try:
            with open("/sys/class/thermal/thermal_zone0/temp", "r") as f:
                temp_str = f.readline()
                return float(temp_str) / 1000.0  # Convert millidegree to degree
Celsius
        except Exception as e:
            print(f"* Error reading CPU temperature: {e}")
            return 0.0

def main(args=None):
        rclpy.init(args=args)
        rp_telemetry_sensors_pub = RpTelemetrySensors_Pub()
        rclpy.spin(rp_telemetry_sensors_pub)
        rclpy.shutdown()

if __name__ == '__main__':
    main()
    print('*** Ended RpTelemetrySensors_Pub')
```

c. Package launch

```
~/go_core_telemetry.sh (content: ros2 run rp_telemetry_pkg rp_telemetry_node)

* 2025-06-12 15:02:20 - Publisher created for topic: rp_telemetry_a
* 2025-06-12 15:02:20 - RpTelemetrySensors_Pub node started ...
* 2025-06-12 15:02:20 - Publishing RpTelemetry messages every 2.0 seconds ...
- (rp_telemetry_a) rp_telemetry_interfaces.msg.RpTelemetry(now='2025-06-12
15:02:22', temperature=45.277)
```

The output confirms that the node correctly identifies the device, selects the corresponding topic, and publishes telemetry messages at the intended frequency.

4.4.3 Propulsion system

4.4.3.1 System overview and mechanical layout

The propulsion system of the smart cart was designed to provide stable low-speed motion, precise positioning, and compatibility with autonomous operation. The platform adopts a differential drive configuration with two powered front wheels and two rear caster wheels, enabling manoeuvrability in narrow agricultural environments.

The propulsion system consists of two drive wheels (20 inches diameter) and two pivoting rear wheels (caster wheels). This configuration was selected to balance mechanical simplicity, load-bearing capability, and precise low-speed manoeuvrability, which are critical requirements for field operations and sensor-guided tasks.

4.4.3.2 Motor drive and control electronics

Each drive wheel is actuated by an independent DC motor and controlled through a dedicated motor driver and motion controller, enabling closed-loop speed control and feedback-based motion regulation.

The propulsion system of each wheel consists of a 250W brushed motor (24 VDC) DC powered and managed by an electronic board **Sabertooth Dual 2x32A** 6-30 VDC Regenerative Motor Driver (https://tinyurl.com/y2x4jyno, the manual is here https://tinyurl.com/2bronpls) and by a second electronic board performing the feedback wheel speed controller (**Kangaroo X2** Motion Controller, https://tinyurl.com/23ptwdpx, the manual is here https://tinyurl.com/2xv3xyfd).

The rotational speed of each motor is measured by a quadrature rotary incremental encoder (CALT **GHS52-8G3600BMP526**, https://tinyurl.com/28dzp4rx, https://tinyurl.com/2cx8r394), with power supply 5-26 VDC, PNP Push Pull with pulse output signal of 3600 pulses per rotation (PPR). The ABZ outputs of the encoder are connected directly to the **Kangaroo** board (https://tinyurl.com/2ytjuzl9), as can be seen from the following Figure 4.15.

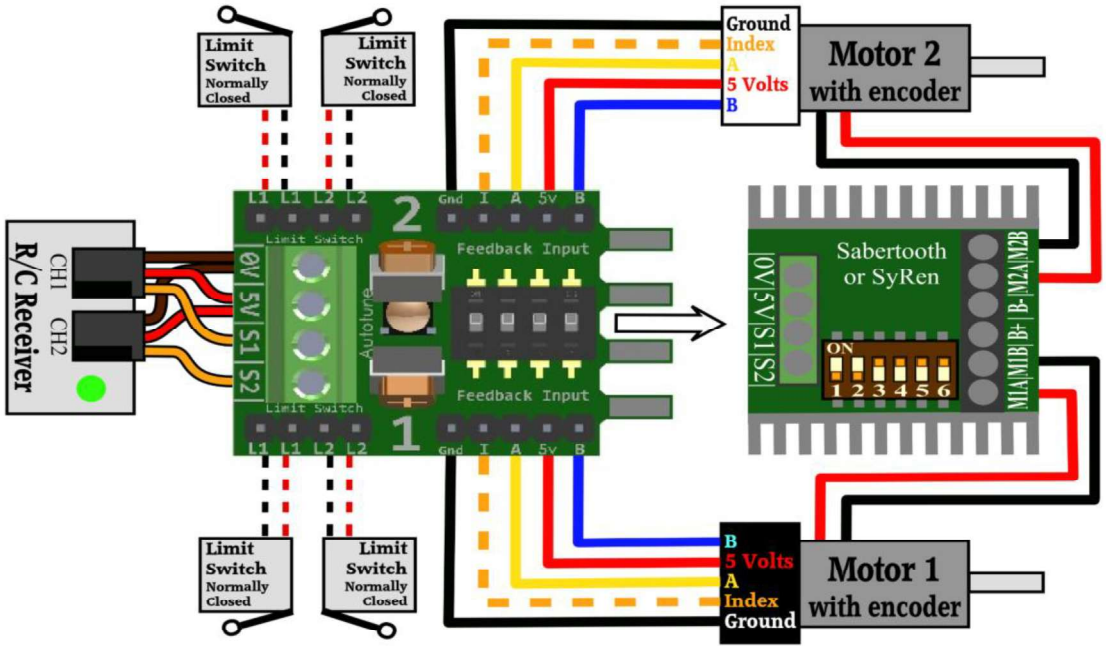


Figure 4.15. Connection of the encoders to the Kangaroo board (Kangaroo x2 Sample Wiring Diagram: https://tinyurl.com/2ytjuzl9).

This combination of motor driver, motion controller, and high-resolution encoder enables closed-loop control of wheel speed and position, forming the foundation for accurate and repeatable motion control.

4.4.3.3 Encoder-based feedback and wheel speed measurement

The layout of the mechanical connection (chain) among electrical motor, encoder and wheel is shown in the previous figure (Figure 4.15).

The electrical motor driving the wheel is a brushed 24 VDC, 250 W, 75 RPM, 13.4 A, with an efficiency of 0.78. With a reduction ratio of 32:9, the wheels reach 21.09 RPM maximum.

Therefore, the prototype maximum forward speed is equal to 2*pi*(2.54*20/2)*21.09/60 = 56.11 cm/s (2.02 km/h).

One rotation of the wheel (linear advancement of 1.5959 meters) corresponds to 32/9 rotations of the encoder; this corresponds to 3600*32/9=12800 pulses. Therefore, the error on the linear advancement is 1.5959/12800= 0.1247 mm.

As the camera has 1280x960 pixels resolution, each camera pixel covers an area of 1.79 mm2, i.e. a square of about 1.34 mm on each side. Therefore, this value must be the precision that must be guaranteed by the movement controller.

The error results one order of magnitude below the required movement precision by the camera. This result confirms that the propulsion system is capable of supporting sensor-

driven operations, such as imaging and alignment tasks, without introducing motion-induced spatial inaccuracies.

The following Matlab script performs the calculation.

calc_movement_precision.m

```
motor_max_rpm = 75
motor_sprocket = 9
encoder_sprocket = 9
encoder_ppr = 3600
wheel_sprocket = 32
wheel_diameter_cm = 20 *(2.54) % centimeters
wheel_max_rpm = motor_max_rpm * motor_sprocket/wheel_sprocket
prototype_max_forward_speed_cm_s = 2*pi*(wheel_diameter_cm/2)*wheel_max_rpm/60
prototype_max_forward_speed_km_h = prototype_max_forward_speed_cm_s * 3600/1E5
one_wheel_rotation_m = 2*pi*wheel_diameter_cm/2/100
encoder_pulses_one_wheel_rotation = encoder_ppr*wheel_sprocket/encoder_sprocket
linear_advancement_error_mm=one_wheel_rotation_m/encoder_pulses_one_wheel_rotatio
n*1000
```

```
motor_max_rpm = 75
motor_sprocket = 9
encoder_sprocket = 10
encoder_ppr = 3600
wheel_sprocket = 32
wheel_diameter_cm = 50.8000
wheel_max_rpm = 21.0938
prototype_max_forward_speed_cm_s = 56.1069
prototype_max_forward_speed_km_h = 2.0198
one_wheel_rotation_m = 1.5959
encoder_pulses_one_wheel_rotation = 12800
linear_advancement_error_mm = 0.1247
```

4.4.3.4 Control architecture and embedded integration: programming Arduino from command line on Raspberry PI 4

The low-level motor control logic is executed on a dedicated microcontroller, while high-level coordination is managed by the embedded computing unit. This separation allows real-time motor control to be handled locally by the microcontroller, while higher-level commands, monitoring, and system integration are managed by the Raspberry Pi within the ROS 2 environment.

An **Arduino Mega 2560** REV3 allows the management of the **Kangaroo** board via its **serial port #1**. The **Arduino Mega 2560** is managed by **Raspberry PI 4** through its USB port (serial port #0). (see Figure 4.16).

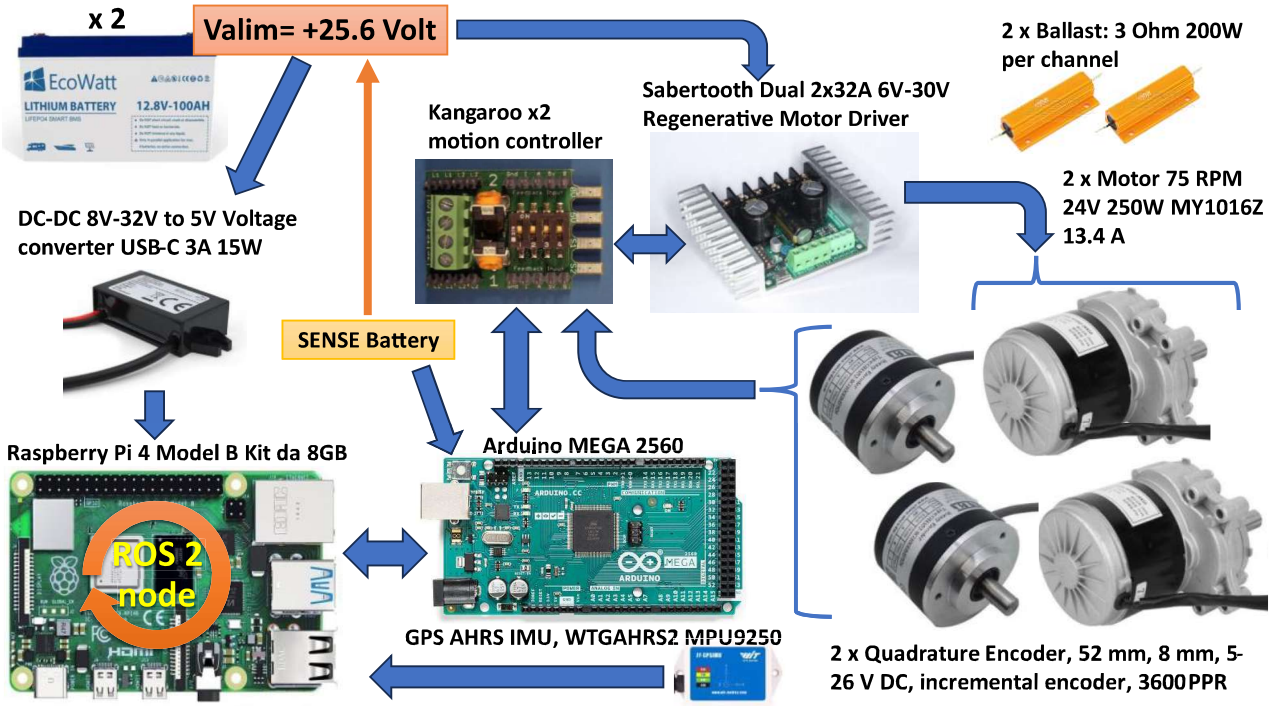


Figure 4.16. Connection of the encoders to the Kangaroo board (Kangaroo x2 Sample Wiring Diagram: https://tinyurl.com/2ytjuzl9).

This architecture provides a reliable and extensible interface between the propulsion hardware and the higher-level software stack, enabling future integration with autonomous navigation and control modules.

1. **Arduino programming environment and command-line workflow**

To enable rapid development, testing, and deployment of the motor control firmware, the Arduino Mega 2560 was programmed directly from the Raspberry Pi 4 using a command-line–based workflow. This approach allows tight integration between the embedded controller and the onboard computing unit, facilitating iterative development and debugging.

Execute. `pip install ino`.

2. **Arduino CLI setup on Raspberry Pi**

Initially, a command-line toolchain was configured on the Raspberry Pi to manage Arduino projects, compile firmware, and upload code to the microcontroller.

"ino" is a command line toolkit for working with Arduino hardware from the command line (https://pypi.org/project/ino/). It allows to: quickly create new projects; build a firmware from multiple source files and libraries; upload the firmware to a device; perform serial communication with a device (aka serial monitor). However, it requires Python 2.

Due to the dependency of the ino toolkit on Python 2, which is deprecated, the workflow was migrated to the official Arduino CLI, providing long-term maintainability and compatibility with modern systems.

Therefore, install Arduino CLI (https://arduino.github.io/arduino-cli/1.2/installation/) in **~/bin**

```
curl -fsSL https://raw.githubusercontent.com/arduino/arduino-cli/master/install.sh | sh
```

Then, define in ~/.bashrc the alias aino = '~/bin/arduino-cli' and then source.

```
aino board list will list the available connected boards
```

```
Port         Protocol Type              Board Name                FQBN            Core
/dev/ttyACM0 serial     Serial Port (USB) Arduino Mega or Mega 2560 arduino:avr:mega
arduino:avr
/dev/ttyAMA0 serial   Serial Port       Unknown
```

Arduino CLI system setup.

```
aino update will update the arduino-cli system
aino config init
aino core update-index
aino core install arduino:avr
aino core list
```

Install Improved Nano Syntax Highlighting Files executing:

```
curl https://raw.githubusercontent.com/scopatz/nanorc/master/install.sh | sh
```

3. Verification of the programming toolchain

To verify the correct configuration of the programming environment, a minimal test sketch was created, compiled, and uploaded into the Arduino Mega 2560.

Create a sample sketch folder to test the overall functionality, build and upload the code.

```
mkdir ~/myarduino
cd ~/myarduino
mkdir ablink
cd ablink
touch ablink.ino
nano ablink.ino
```

```
void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
    digitalWrite(LED_BUILTIN, HIGH);
    delay(1000);
    digitalWrite(LED_BUILTIN, LOW);
    delay(1000);
}
```

```
aino compile --fqbn arduino:avr:mega
aino upload -p /dev/ttyACM0 --fqbn arduino:avr:mega
```

The following aliases have been added to ~/**.bashrc**, and subsequently sourced.

```
alias ainoco = '~/bin/arduino-cli compile --fqbn arduino:avr:mega'
alias ainoup = '~/bin/arduino-cli upload -p /dev/ttyACM0 --fqbn arduino:avr:mega'
```

These aliases must be executed into the sketch folder.

Libraries are searched using:

```
aino lib search XXXXXXX
```

A specific library is installed using:

```
aino lib install YYYYYY
```

After validating the programming workflow, a bidirectional serial communication test was implemented to evaluate data exchange between the Raspberry Pi and the Arduino-based motor controller.

A more sophisticated test program showing communications between Raspberry PI 4 and Arduino Mega 2560:

a-send-receive.ino

```
#include <Arduino.h>
void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
    Serial.begin(115200); // Initialize serial communication baud
}
int onInterval= 1000; // Interval for LED blinking in milliseconds
int offInterval = 1000; // Interval for LED off in milliseconds
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);
    delay(onInterval);
    digitalWrite(LED_BUILTIN, LOW);
    delay(offInterval);
    // Check if data is available to read
    if (Serial.available() > 0) {
        int incomingByte = Serial.read(); // Read one byte
        switch (incomingByte)  // Use switch-case to handle different byte values
        {
        case 'q':
            onInterval = 100; // Change on interval to 500 ms
            offInterval = 100; // Change off interval to 500 ms
            break;
        case 'a':
            onInterval = 500; // Change on interval to 1000 ms
            offInterval = 500; // Change off interval to 1000 ms
            break;
        case 'z':
            onInterval = 1000; // Change on interval to 2000 ms
            offInterval = 1000; // Change off interval to 2000 ms
            break;
```

```
        default:
            Serial.println("");
                Serial.println("--- Unknown. Use 'q', 'a', or 'z' to change
intervals.");
            onInterval = 50;
            offInterval = 50;
            break;
        }
        Serial.println("");
        Serial.print("--- Received: ");
       Serial.println(incomingByte, DEC); // Print the received byte as a number
    }
}
```

This test demonstrates reliable serial communication over the USB interface, which is essential for transmitting motion commands and receiving feedback during system operation.

Use `picocom -c /dev/ttyACM0` to send/receive command to Arduino Mega 2560.

`picocom -b 115200 -c /dev/ttyACM0, this was aliased in ~/.bashrc as ainocom`

to exit from **picocom**: Ctrl+a and Ctrl+x.

The example demonstrates that Raspberry PI 4 and Arduino Mega 2560 can communicate over the serial (USB) link. However, most Arduinos reset when opening/closing the serial connection. It's a design feature and it allows them to be easily programmed over (USB) serial.

During testing, a known Arduino behavior related to serial port initialization was observed.

The Arduino is reset when the connection is next established (i.e. when the serial port is opened). This behaviour is by design. To overcome this a 10 uF capacitor must be connected between RESET and +5V or GND pins (the connection to GND is the preferred one).

This hardware mitigation ensures stable long-term serial communication during continuous operation.

Having validated the programming workflow and the serial communication channel, the next step focused on integrating the Arduino-based motor controller into the ROS 2 ecosystem.

4.4.3.5 The ROS2 node managing the motors: Arduino Mega 2560 publisher node proof of concept

To integrate the Arduino Mega 2560–based motor controller into the ROS 2 ecosystem. The objective is to enable bidirectional communication between the high-level control layer running on the Raspberry Pi 4 and the low-level motor control logic executed on the microcontroller, using a lightweight and reliable serial interface.

Serial communication between the Raspberry Pi 4 and the Arduino Mega 2560 was implemented using the PySerial library, which provides a simple and portable interface for accessing serial devices from Python-based ROS 2 nodes.

Pyserial is used when communicating with Arduino Mega 2560 over the serial USB port.

```
pip install pyserial
ubuntu@ubuntu:~$ python -m serial.tools.list_ports
/dev/ttyACM0
/dev/ttyAMA0
2 ports found
```

The detection of the Arduino Mega 2560 on /dev/ttyACM0 confirms the correct configuration of the USB serial interface.

4.4.3.6 XICRO: Firmware and middleware code generation for ROS 2–microcontroller integration (quoted from https://xicro-ros2.readthedocs.io/en/latest/)

4.4.3.6.1 "Why" XICRO?

To explore scalable and architecture-independent approaches for interfacing microcontrollers with ROS 2, the XICRO framework was evaluated as an alternative middleware solution.

The Robot Operating System (ROS) is a popular open-source framework for developing robotic systems. ROS2, the latest version of the framework, is designed to support real-time, distributed systems, and has become a popular choice for robotics researchers and developers. However, one limitation of ROS2 is that it is designed to run on high-performance computers, which can be expensive and impractical for many robotics applications.

To address this limitation, several middleware solutions have been developed to allow ROS2 to run on microcontrollers with limited resources. One such solution is MicroROS, a middleware that provides a communication layer between microcontrollers and ROS2. However, MicroROS is limited to microcontrollers with 32-bit architecture, and cannot be used with lower-bit architecture such as AVR. This limitation poses a challenge for developers who want to use ROS2 with low-cost, low-power microcontrollers that are widely available in the market.

To overcome this limitation, a software library is needed that allows any-bit architecture microcontroller to connect with ROS2 using UART or UDP. This library should provide a lightweight and efficient communication protocol that is optimized for microcontrollers with limited resources. By enabling low-cost microcontrollers to connect

with ROS2, this library can help democratize access to ROS2 and make it more accessible to developers working on low-cost robotics projects.
The general features of "xicro" are

- xicro auto-generates a library for a microcontroller (.h file)
- xicro auto-generates a node that communicates with UART
- xicro supports all Arduino, ESP32, ESP8266 and STM32 families.

The following steps summarize the installation and configuration procedure adopted to deploy XICRO within a ROS 2 workspace on the Raspberry Pi 4.

4.4.3.6.2 Installation and ROS 2 workspace setup

To deploy XICRO within the ROS 2 environment running on the Raspberry Pi 4, a dedicated workspace was created. XICRO is structured as a meta-package that enables automatic code generation for both the microcontroller firmware and the corresponding ROS 2 nodes.

The following steps describe the installation procedure adopted in this work.

Workspace creation and package installation:

arduino_ws is the workspace.

```
mkdir
cd ~/arduino_ws/src      #cd to your workspace
mkdir Xicro              #create metapackage
cd Xicro                 #cd to metapackage
git clone https://github.com/imchin/Xicro .
cd ~/arduino_ws
colcon build
# insert source ~/arduino_ws/install/setup.bash in ~/-bashrc
```

Install python library:

```
pip3 uninstall serial
pip3 install pyserial
pip3 install numpy
```

Configuration of system parameters:

System-specific parameters were defined in a YAML configuration file, describing both the microcontroller communication interface and the ROS 2 topics to be generated.

Setup parameters in yaml

```
cd ~/arduino_ws/src/Xicro/xicro_pkg/config
nano setup_xicro.yaml
```

The configuration file is structured into two main sections: `microcontroller` and `ros`.

These are contained in setup_xicro.yaml :

- **microcontroller:**
  - **idmcu**: It sets the ID of the MCU for the system Xicro. The mcu Id may range from 0 to 15.
  - **namespace**: This is the name of the file that will be created.
  - **generate_library_Path**: Is the path for creating .h and .cpp files. Xicro will create a folder with files at the path. Path reference from ~/
  - **connection**:
    - **type**: Support 2 mode 1.Serial UART 2.Wifi UDP (Only Arduino and esp32)
    - **serial_port**: Name open port of MCU
    - **baudrate**: affects the data transmission rate. Recommended 115200.
    - **ip_address_mcu**: Ip address of mcu in WLAN
    - **udp_port_mcu**: port of connection UDP mode

- **ros:** # setup of the ros2
  - **publisher**: Configuration for publishing topic from MCU to ROS2.
    Format: [[ID_topic, Name_topic, Interface], …]
    - **ID_topic**: It sets the ID of the topic_publish for Xicro, 0..255.
    - **Name_topic**: Specify the topic name you want MCU to publish to ROS2.
    - **Interface**: interface file configuration for use in the topic. It contains 2 part: "(The name of the package containing the interface file)/(Name_of_fileinterface).msg"
  - **subscriber**: Configuration for subscribe topic from ROS2 to MCU.
    Format [[ID_topic, Name_topic, Interface],…]
    - **ID_topic**: It sets the ID of the topic_subscribe for Xicro, 0..255.
    - **Name_topic**: Specify the topic name you want MCU to subscribe from ROS2.
    - **Interface**: interface file configuration for use in the topic. It contains 2 part: "(The name of the package containing the interface file)/(Name_of_fileinterface).msg"
  - **srv_client**: Configuration for service client.
    Format [[ID_service, Name_service, Interface, time_out],…]
    - **ID_service**: It sets the ID of the service_client for Xicro, 0..255.
    - **Name_service**: Specify the service name you want MCU service call to ROS2.
    - **Interface**: interface file configuration for use in the service. It contains 2 part: "(The name of the package containing the interface file)/(Name_of_fileinterface).srv"
    - **time_out**: Limit the maximum service usage time (as float).
  - **srv_server**: Configuration for service server.
    Format [[ID_service, Name_service, Interface, time_out],…]
    - **ID_service**: It sets the ID of the service_server for Xicro, 0..255.
    - **Name_service**: Specify the service name you want ROS2 service call to MCU.

- Interface: interface file configuration for use in the service. It contains 2 part: "(The name of the package containing the interface file)/(Name_of_fileinterface).srv"
- time_out: Limit the maximum service usage time (as float).

- action_client: Configuration for action client.
  Format [[ID_action, Name_action, Interface, time_out],…]
  - ID_action: It sets the ID of the action_client for Xicro, 0..255.
  - Name_action: Specify the action name you want MCU action send_goal to ROS2.
  - Interface: interface file configuration for use in the action. It contain 2 part: "(The name of the package that contains the interface file)/(Name_of_fileinterface).action"
  - time_out: Limit the maximum action usage time (as float).
- action_server: Configuration for action server.
  Format [[ID_action, Name_action, Interface, time_out],…]
  - ID_action: It sets the ID of the action_server for Xicro, 0..255.
  - Name_action: Specify the action name you want ROS2 action send_goal to MCU.
  - Interface: interface file configuration for use in the action. It contain 2 part: "(The name of the package that contains the interface file)/(Name_of_fileinterface).action"
  - time_out: Limit the maximum action usage time (as float).

List of all interfaces available:
ros2 interface list

List of packages with messages interface available:
ros2 interface packages

To see available interfaces in **std_msgs**:
ros2 interface package std_msgs

To show the content of a specific interface:
ros2 interface show std_msgs/msg/String

setup_xicro.yaml

```yaml
microcontroller:
  idmcu: 1 # id of the mcu 0..15
  namespace: "arduinomega" # namespace, name of the file that will be created
  generate_library_Path: "arduinolib/libraries" # path of generated library files from ~/
  connection:
    type: "UART"   # ["UART","UDP"] support 2 mode: Serial UART or Wifi UDP
    serial_port: "/dev/ttyACM0"  # for Serial UART e.g. "/dev/ttyACM0"
    baudrate: 115200  # for UART
    ip_address_mcu: ""  # for UDP example "192.168.1.xx"
    udp_port_mcu: 0  # for UDP

ros: # setup of the ros2
  publisher:  [ [1,"alive_mega","std_msgs/String.msg"] ]
  subscriber: [ ]
  srv_client: [ ]
  srv_server: [ ]
  action_client: [ ]
  action_server: [ ]
```

Based on the defined configuration, XICRO automatically generates both the microcontroller library and the corresponding ROS 2 node.

A. Creating xicro library for Arduino

The Arduino library is automatically generated based on the configuration file `setup_xicro.yaml`. This step produces the Arduino-compatible source files `(.h / .cpp)` in the destination path specified by `generate_library_Path`.

The library will be generated based on setup_xicro.yaml

```
cd ~/arduino_ws/src
colcon build
ros2 run xicro_pkg generate_library.py -mcu_type Arduino
```

B. Create xicro node for Arduino

The corresponding ROS 2 node is generated using the same `setup_xicro.yaml` configuration. The node entry point is created automatically and integrated into the ROS 2 workspace.

```
ros2 run xicro_pkg generate_xicro_node.py -mcu_type arduino
```

This step creates the Arduino-compatible library files (.h and .cpp) in the specified destination path.

The node entry point is generated automatically and integrated into the ROS 2 workspace.

The entry point is automatically added.

C. Deployment workflow summary:

After generating both the Arduino library and the ROS 2 node, the remaining steps required to complete the integration are:

- Create xicro library
- Create xicro node
- Upload code containing xicro_library to MCU
- Connect MCU to computer

D. Arduino firmware example (alive publisher)

The following Arduino sketch shows a minimal example used to validate the XICRO workflow. The microcontroller periodically publishes an “alive” message over UART.

The Arduino code:

alive_mega.ino

```
#include
"/home/ubuntu/arduinolib/libraries/Xicro_arduinomega_ID_1/Xicro_arduinomega_ID_1.cpp"
Xicro xicro;
uint32_t timestamp=0;
```

```
char buffer[32];
#define speed 100
void setup() {
   Serial.begin(115200);
   xicro.begin(&Serial);
}
void loop() {
   xicro.Spin_node();
   if(millis()-timestamp>=speed){
      timestamp=millis();
      sprintf(buffer, "%lu", timestamp);
      xicro.Publisher_alive_mega.message.data = buffer;
      xicro.publish_alive_mega();
   }
}
```

E. Running the XICRO node and validating the ROS 2 topic

Once the firmware is uploaded and the microcontroller is connected, the generated ROS 2 node can be executed on the Raspberry Pi 4:

```
ros2 run xicro_pkg xicro_node_arduinomega_ID_1_arduino.py
ros2 topic echo /alive_mega
ubuntu@ubuntu:~$ ros2 topic hz /alive_mega
average rate: 10.004
        min: 0.091s max: 0.108s std dev: 0.00512s window: 12
average rate: 9.952
        min: 0.086s max: 0.113s std dev: 0.00650s window: 23
```

The response shows how the /alive_mega topic is published at 10 Hz rate as required.

4.4.4 The Overall Master Control (OMC) using an Arduino Mega 2560

The Overall Master Control (OMC) is responsible of supervising the lifecycle management of the embedded computing nodes composing the system. Implemented on an Arduino Mega 2560, the OMC coordinates power-up sequences, safe shutdown procedures, and system readiness monitoring for multiple Raspberry Pi 4 devices, ensuring robustness, fault tolerance, and scalability.

4.4.4.1 Functional role of the OMC

The main responsibilities of **OMC** are the timed power-on sequence of Raspberry Pi 4 devices, coordinated and safe shutdown sequence, system readiness monitoring, power-domain isolation via relays, and distributed device identification.

The **OMC** performs a timed boot sequence of the devices (Raspberry Pi 4) composing the system and their timed safe shutdown sequence when required by the operator.

The safe shutdown of Raspberry Pi 4 devices is performed using the GPIO **dtoverlay** facility.

- **GPIO23** is used as an **input** for the safe shutdown request (when momentarily brought to **LOW**).
- **GPIO24** is an **output** signalling system start completed and ready (when **HIGH**).
- **GPIO25** is an **output** for the power down request granted signal (when toggling from **LOW** to **HIGH**) (Figure 4.17).

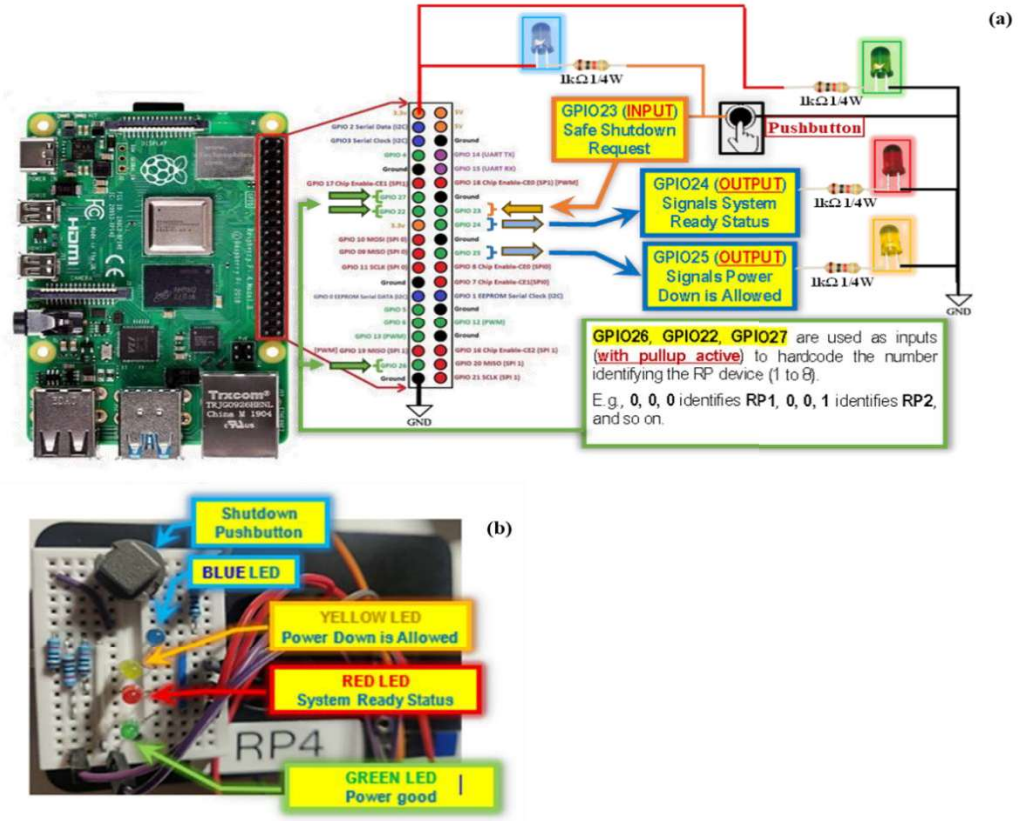


Figure 4.17. Raspberry PI 4 pins connection (a) and implementation (b).

The GPIO behavior is configured at boot time using device tree overlays defined in the Raspberry Pi firmware configuration.

This behaviour was coded in the file `/boot/firmware/config.txt`.

```
/boot/firmware/config.txt
gpio=22,26,27=ip,pu
gpio=24=op,dl
dtoverlay=gpio-shutdown,gpio_pin=23
dtoverlay=gpio-poweroff,gpiopin=25,active_delay_ms=1000,inactive_delay_ms=1000
```

"**`dtoverlay -l`**" shows a list of the applied overlays.

```
Behaviour of GPIO24: (as OUTPUT 3.3V)
Pin is low on powerup.
Subsequently, pin goes high to signal a System Ready Status when its associated Systemd
service is started. At shutdown pin goes low again and stays low.

This behaviour is managed by a Systemd service.
```

In addition to the **build_ros2_range_list.service** (hosted in **/etc/systemd/system**) that manages the building of the **DEFAULT_FASTRTPS_PROFILES.xml** listing the IP of the ROS2 nodes.

/etc/systemd/system/build_ros2_range_list.service

```
[Unit]
Description=ROS2 create range list
After=network-online.target
Wants=network-online.target
[Service]
User=ubuntu
Group=ubuntu
ExecStart= /home/ubuntu/MAKE_RANGE_LIST.sh
RemainAfterExit=no
Restart=on-failure
RestartSec=2s
[Install]
WantedBy=multi-user.target
```

The behaviour of **GPIO24** is managed using the service file below.

/usr/lib/systemd/system/my-signal-system-ready-status.service

```
[Unit]
Description= my-signal-system-ready-status

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/home/ubuntu/signal_system_ready.sh
ExecStop=/home/ubuntu/signal_system_not_ready.sh

[Install]
WantedBy=multi-user.target
```

Then: **systemctl enable my-signal-system-ready-status --now** command will install and enable the service.

signal_system_ready.sh and **signal_system_not_ready.sh** shell commands call simply two Python scripts handling the GPIO24 pin level accordingly to the system status:

gpio_system_ready.py

```
# gpio24 to ON when system is ready
# set THIS_RP_DEVICE_NAME in accord to hardware code
   import RPi.GPIO as GPIO
   GPIO.setwarnings(False)
   GPIO.setmode(GPIO.BCM)

   GPIO.setup(26, GPIO.IN)
   GPIO.setup(22, GPIO.IN)
   GPIO.setup(27, GPIO.IN)
   device_id = GPIO.input(27)
   device_id += GPIO.input(22)*2
   device_id += GPIO.input(26)*4
   device_id += 1
   open("/dev/shm/THIS_RP_DEVICE_NAME", "w").write(str(device_id)+"\n")

   GPIO.setup(24,GPIO.OUT)
   GPIO.output(24,True)
```

gpio_system_not_ready.py

```
# gpio24 to OFF when system NOT ready
   import RPi.GPIO as GPIO
   GPIO.setwarnings(False)
```

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(24,GPIO.OUT)
GPIO.output(24,False)
```

Behaviour of **GPIO23**: (as INPUT 3.3V)

Pin must be floating or high on powerup.

When the pin is momentarily grounded (on falling edge) a shutdown sequence is started.

dtoverlay -h gpio-shutdown will display the documentation.

```
Name:   gpio-shutdown
Info:   Initiates a shutdown when GPIO pin changes. The given GPIO pin
        is configured as an input key that generates KEY_POWER events.
        This event is handled by systemd-logind by initiating a
        shutdown. Systemd versions older than 225 need an udev rule
        enable listening to the input device:
                ACTION!="REMOVE", SUBSYSTEM=="input", KERNEL=="event*", \
                        SUBSYSTEMS=="platform", DRIVERS=="gpio-keys", \
                        ATTRS{keys}=="116", TAG+="power-switch"
        Alternatively this event can be handled also on systems without
        systemd, just by traditional SysV init daemon. KEY_POWER event
        (keycode 116) needs to be mapped to KeyboardSignal on console
        and then kb::kbrequest inittab action which is triggered by
        KeyboardSignal from console can be configured to issue system
        shutdown. Steps for this configuration are:
            Add following lines to the /etc/console-setup/remap.inc file:
                # Key Power as special keypress
                keycode 116 = KeyboardSignal
            Then add following lines to /etc/inittab file:
                # Action on special keypress (Key Power)
                kb::kbrequest:/sbin/shutdown -t1 -a -h -P now
            And finally reload configuration by calling following commands:
                # dpkg-reconfigure console-setup
                # service console-setup reload
                # init q
        This overlay only handles shutdown. After shutdown, the system
        can be powered up again by driving GPIO3 low. The default
        configuration uses GPIO3 with a pullup, so if you connect a
        button between GPIO3 and GND (pin 5 and 6 on the 40-pin header),
        you get a shutdown and power-up button. Please note that
        Raspberry Pi 1 Model B rev 1 uses GPIO1 instead of GPIO3.
Usage:  dtoverlay=gpio-shutdown,<param>=<val>
Params: gpio_pin                GPIO pin to trigger on (default 3)
                                For Raspberry Pi 1 Model B rev 1 set this
                                explicitly to value 1, e.g.:
                                    dtoverlay=gpio-shutdown,gpio_pin=1
        active_low              When this is 1 (active low), a falling
                                edge generates a key down event and a
                                rising edge generates a key up event.
                                When this is 0 (active high), this is
                                reversed. The default is 1 (active low).
        gpio_pull               Desired pull-up/down state (off, down, up)
                                Default is "up".
                                Note that the default pin (GPIO3) has an
                                external pullup. Same applies for GPIO1
                                on Raspberry Pi 1 Model B rev 1.
        debounce                Specify the debounce interval in milliseconds
                                (default 100)
```

Behaviour of **GPIO25**: (as OUTPUT 3.3V)

Pin is low on powerup.

Testing. Log in (via ssh, and then shutdown: $ **sudo shutdown -h now**).

Pin goes high for 1 second, then, pin goes low for 1 second, finally, pin goes high and stays high.

GPIO24 goes low (as previously depicted).

The temporal behavior of GPIO25 is shown in Figure 4.18.

Figure 4.18.Temporal Behavior of GPIO25 as Digital Output (3.3V)

4.4.4.2 Electrical layout and scalability of the OMC

The **OMC** is planned to manage **up to 8 Raspberry Pi 4 devices** (**RP1**..**RP8**) using a single Arduino Mega 2560 device. (Figure 4.19)

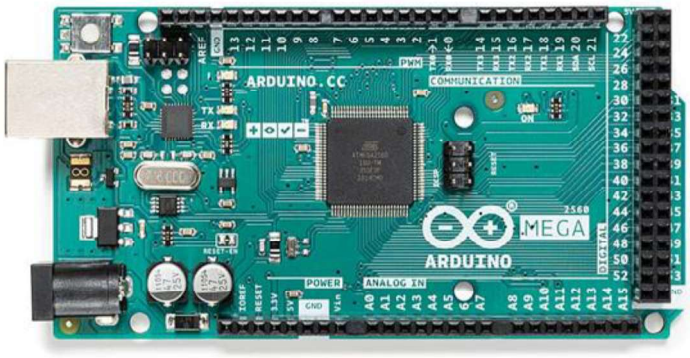


Figure 4.19. Arduino Mega 2560 device

The Arduino Mega 2560 device pins assignment is detailed below.

- **8 pins** (output pin: **22, 24, 26, 28, 30, 32, 34, 36**) managing the power supply to each **RP1**..**RP8** through 8 relays normally open.
- **8 pins** (output pin: **38, 40, 42, 44, 46, 48, 50, 52**) managing the Query Shutdown Signal (**QSS**) to each **RP1**..**RP8** through 8 relays normally open (this in order to simplify

interfacing between Arduino Mega 2560 and Raspberry PI 4). The connection could be made using a diode from Arduino digital output pin (+5V, cathode of the diode) and GPIO digital input pin of Raspberry PI 4 (anode of the diode), however, the Arduino pin must be operated with an inverted logic level. Indeed, while using relays, the ON level from Arduino activates the relays bringing the connection of GPIO to ground, but, when using direct connection by diode, it is the OFF level from Arduino that brings the connection of GPIO to ground. This aspect must be carefully considered.

- **8 pins** (input pin: **23, 25, 27, 29, 31, 33, 35, 37** referring to **RP1**..**RP8**) reading the Power Down Allowed (**PDA**) signal from each **R1**..**R8** (the +3.3V logic being compatible, with an **input with pullup**, with the +5V logic, **using a diode**).
- **8 pins** (input pin: **39, 41, 43, 45, 47, 49, 51, 53** referring to **RP1**..**RP8**) reading the System Ready Status (**SRS**) signal from each **RP1**..**RP8** (the +3.3V logic being compatible, with an **input with pullup**, with the +5V logic, **using a diode**).
- **1 pin** (input pin: **2**) reading the Overall Shutdown Command (**OSC**), lever button.
- **1 pin** (input pin: **3**) reading the Enable Start Sequence (**ESS**), lever button.
- **1 pin** (input pin: **4**) reading the Disable Serial Communication (**DSERCOM**), lever button.
- **3 pins** (input pin: **12, 11, 10**) number of **RP** devices managed in addition to **RP1** (always being required). E.g. **000** means **RP** number is 1 (only **RP1** device will be managed), **001** means **RP** number is 2 (**RP1** and **RP2** devices will be managed) and so on; the maximum allowed is **111**, meaning 8 devices (**RP1**..**RP8**) will be managed. This codification is read only once at the system start up.
- **1 pin** (analog input pin: **AN0**) reading the battery voltage using a two resistors divider (100 kOhm and 20 kOhm) allowing to read up to +30V (30V -> 5V).
- **8 pins** (analog input pin: **AN1..AN8**) reading the power voltage of each **RP1**..**RP8** (sensed after the relay powering **RP**).
- **2 pins**, **SCL** (pin **21**) and **SDA** (pin **20**) are used to connect 4 LCD displays (4 rows and 20 columns) using the **I2C** interface facility. The addresses used are **0x27** (**LCD1**, A0 open, A1 open, A2 open), **0x26** (**LCD2**, A0 **closed**, A1 open, A2 open), **0x25** (**LCD3**, A0 open, A1 **closed**, A2 open), **0x24** (**LCD4**, A0 **closed**, A1 **closed**, A2 open).

Special attention was devoted to logic-level compatibility between 3.3 V Raspberry Pi GPIOs and 5 V Arduino inputs.

The **START SEQUENCE** requires **ESS=0 and OSC=1**, while the **SHUTDOWN SEQUENCE** requires **ESS=1 and OSC=0**.

The digital input internal pullup resistor of the Arduino Mega 2560 is roughly 20-50 kOhm, for the Raspberry Pi 4 the value is roughly 50-65 kOhm.

To manage the problem of the direct connection between an output using +3.3V level and an input using a +5V level, some simulations have been carried out using the 1N4148 diode or a Red LED, testing if the logic high level (being the problem on the +5V input logic side) is meet (Figure 4.20)

To be noted that on Arduino side the PULLUP resistor on the digital input <u>MUST BE ACTIVE</u> or the LOW signal of the +3.3V logic cannot propagate to digital input of the Arduino Mega 2560.

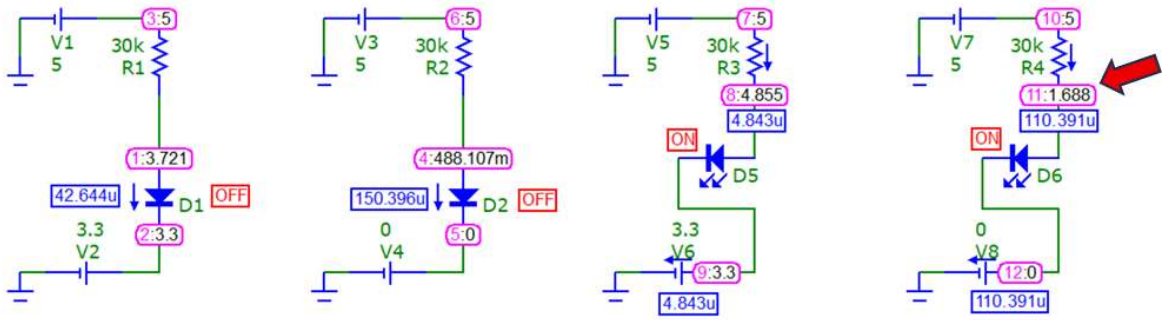


Figure 4.20. Simulations using protection components such as a 1N4148 diode and a red LED to verify whether the 5V logic HIGH threshold is reliably achieved.

For VCC=+5V we have, hence, **VIL**= 1.5V and **VIH**= 3.0V

Therefore, the solution is to use a **simple** diode to increase the reliability of the logic level 1, increasing the safety margin from +0.3V up to +0.7V. Note that a direct connection is also possible, but with a lower safety margin. The electrical connections schematic is reported below (the analog and digital pins identification refers to **RP1** as an example) (Figure 4.21).

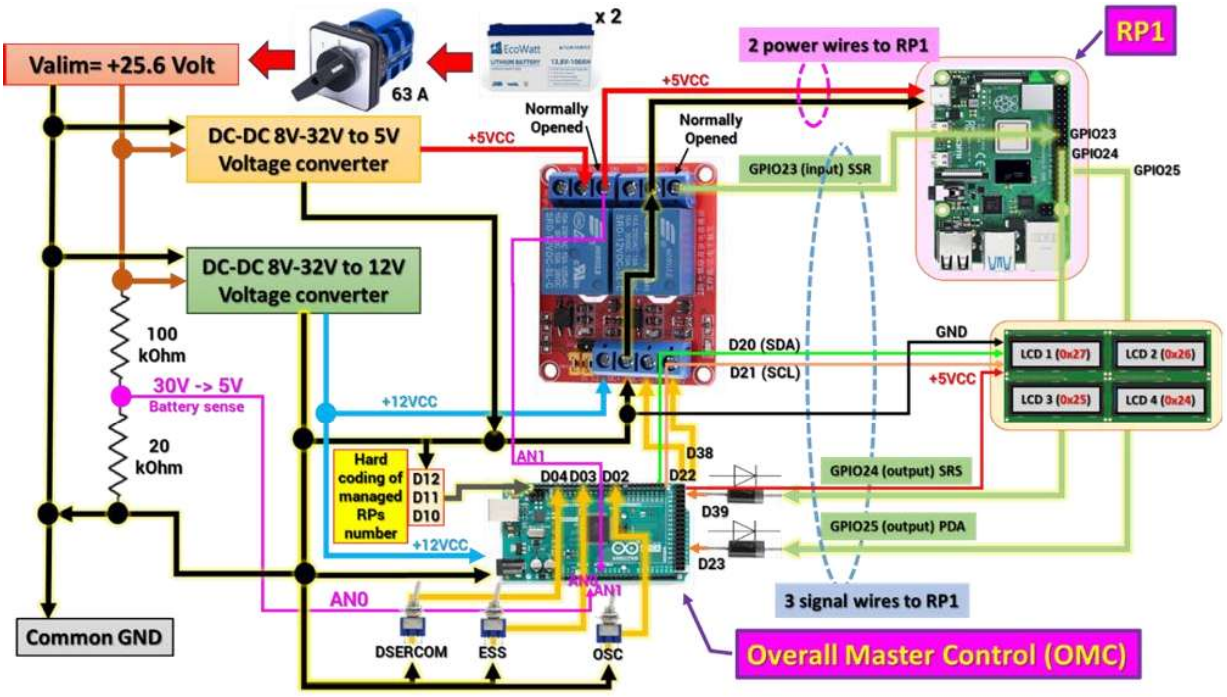


Figure 4.21. Electrical connections schematic; the analog and digital pins identification refers to RP1.

4.4.4.3 Raspberry Pi 4 devices identification map

A timed 5 seconds cron job is used (**crontab -e**):

"*** * * * * (/home/ubuntu/at_startup_each_5_s.sh >/dev/null 2>&1)**"

Where **at_startup_each_5_s.sh** is a script performing the 5 second job **at_startup_script.sh start** (here is the source code: https://tinyurl.com/2aapjdv3)

at_startup_each_5_s.sh

```
#!/bin/bash
# will be executed every 1 minute by cron
LOCKFILE="/dev/shm/at_startup_each_5_s.lock"
exec 200>"$LOCKFILE"
flock -n 200 || {
    echo "*** Another instance of $0 is ALREADY running."
    echo "*** Exiting."
    exit 1
}
# Ensure lock is released on exit or interruption
cleanup() {
    flock -u 200
}
trap cleanup EXIT INT TERM
HOWMANYSECONDS=5
HOWSLICES=$(( 60 / HOWMANYSECONDS )) # 12 intervals in 60 seconds
# The loop runs HOWSLICES-1 times (11 times),
# so the script runs for 55 seconds, not the full minute.
for ((i=0; i<HOWSLICES-1; i++)); do
  # run the script in the background
  /home/ubuntu/at_startup_script.sh start &
  sleep $HOWMANYSECONDS
done
```

Where **at_startup_each_5_s.sh** is a script performing the 5 second job **at_startup_script.sh start** (here is the source code:https://tinyurl.com/2aapjdv3).

4.4.4.4 Device-specific role assignment

The **at_startup_script.sh** 5-second timed script starts (if not already running) the selected required components as listed into the **SCRIPTS_LIST_RP_1** to **SCRIPTS_LIST_RP_8** files.

Each Raspberry Pi device is assigned a specific functional role within the system. The timed cron job allows to start the listed scripts of each RP (e.g go_core_telemetry.sh) if not previously running. The symbol # means a comment, therefore the job is not started.

- **RP1**: **OMC** companion device to collect RPs telemetry data over the ROS2 network Raspberry PI 4 connected to **OMC** through USB serial port) and, eventually, bringing up the foxglove bridge.

  Content of SCRIPTS_LIST_RP_1file:

  - **go_core_telemetry.sh** (this is common to all RP devices)
  - **go_subscribe_telemetry.sh**
  - **go_temp_serial_server.sh**
  - **# go_foxglove.sh**
  - **# go_ros2bridge.sh**

- **RP2**: Micasense camera management ROS2 node (Raspberry PI 4 & Lidar device measuring the target distance)

  Content of SCRIPTS_LIST_RP_2 file:

  - **go_core_telemetry.sh** (this is common to all RP devices)
  - **go_micasense.sh**
  - **# go_foxglove.sh**

- **RP3**: motion management ROS2 node (Raspberry PI 4 & another Arduino Mega 2560), IMU and 2D Lidar sensors

  Content of SCRIPTS_LIST_RP_3 file:

  - **go_core_telemetry.sh** (this is common to all RP devices)
  - **go_myimu.sh** (IMU sensor WTGAHRS2 connected using a TTL-USB converter)
  - **go_mylidar.sh** (2D Lidar SLD PRPLD-C1M1 (RPLIDAR-C1M1) connected using USB)

- **# go_foxglove.sh**
- **TO DO: reserved for inverse kinematics and motion management scripts.**

- **RP4**: main 2D Lidar (Raspberry PI 4)

  Content of SCRIPTS_LIST_RP_4 file:

  - **go_core_telemetry.sh** (this is common to all RP devices)
  - **TO DO: reserved for 2D Lidar mapping management scripts.**

- **RP5**: Joystick control (Raspberry PI 4)

  Content of SCRIPTS_LIST_RP_5 file:

  - **go_core_telemetry.sh** (this is common to all RP devices)
  - **go_joy.sh** (Joystick management scripts)

4.4.5 RP2: LiDAR Sensor Distance – Software Implementation

4.4.5.1 Python Script for serial connection

Preliminarily, Python software was used to test and monitor the proper functioning of the distance sensor through the system serial port (**/dev/ttyAMA0**). Therefore, a small Python script was prepared to this aim, the script is run on-demand on Raspberry Pi using Thonny Python scripting facility (see **READ_SER.PY** in the supporting web site scripts companion file). The run result of the Python script is shown in Figure 4.22.

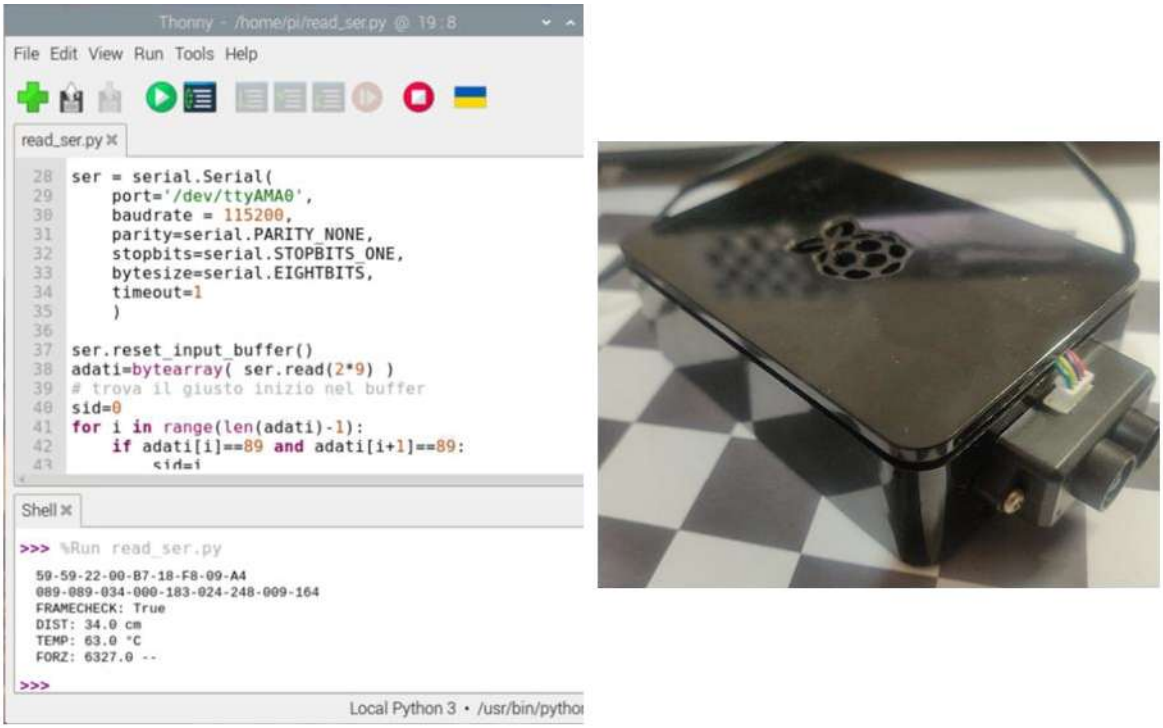


Figure 4.22. Run of the Python script to test for the serial connection of the LiDAR sensor and the LiDAR device connected to the Raspberry Pi3 hardware.

The implementation showed the proper functioning of the **/dev/ttyAMA0** system serial port and a very high responsiveness with very short latent time. This solution is definitely very convincing and will be the adopted final solution.

4.4.5.2 Integration with MATLAB/Simulink environment

Matlab software (Matlab/Simulink Support Package for Raspberry Pi Hardware) was installed onto Raspberry Pi hardware in order to take its full control and to allow for the query of the distance to the Lidar sensor connected to the Raspberry Pi serial port (**/dev/ttyAMA0**). Matlab allows to monitor and manage Raspberry Pi resources such as MATLAB/Simulink deployed processes, CPU, RAM, SD card, external peripheral devices, and interfaces. In addition, Matlab/Simulink processes can be deployed onto Raspberry Pi hardware. In order to use this facility, it is needed to install the Raspberry Pi Resource Monitor App in Matlab and the related libraries onto Raspberry Pi hardware.

The script **go_raspi_test.m** (available into supporting scripts companion folder https://drive.google.com/drive/folders/1wgOIxi_sCWhURPJU31n5xvp4fkcgzaTT) allowed to test the communication in order to setup the distance measuring sensor. The measured acquisition speed ranged from 1 up to 5 Hz (Figure 4.23).

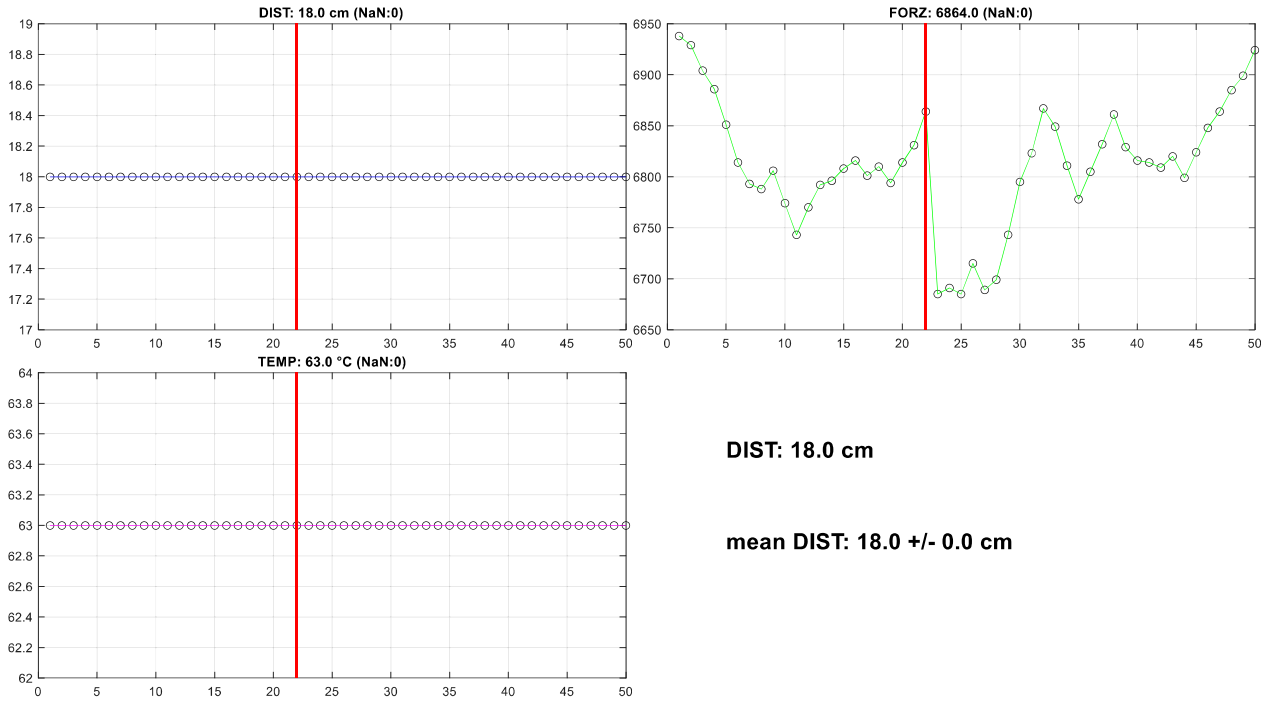


Figure 4.23. go_raspi_test.m script test running at fixed target distance.

In addition, the script **go_raspi_test_w_camera.m** (also available into supporting scripts companion folder https://tinyurl.com/27r6j3od) allowed to test both the distance measuring sensor and an USB camera connected to the Raspberry Pi. The measured acquisition speed is about 1 Hz (Figure 4.24).

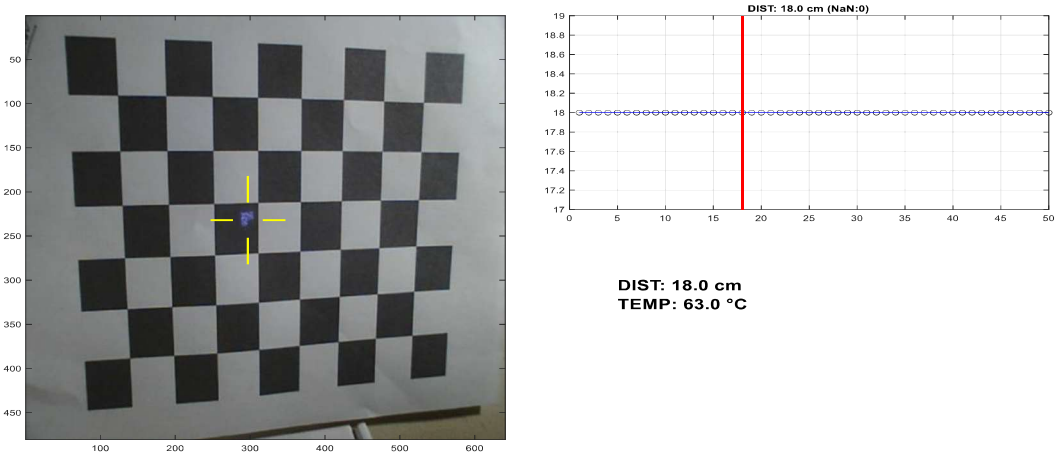


Figure 4.24. go_raspi_test_w_camera.m script test running at fixed target distance. The yellow cross identifies the hot spot emitted by the LiDAR sensor that the normal webcam can see.

A web server (both http:// and https://) was setup on the Raspberry Pi hardware using Apache software, in addition, the PHP support was enabled. This in order to automatize the capability of getting data by the LiDAR sensor. The web server responds to a **GET** from the address http://192.168.1.23/get_dist.php by emitting the distance measured by the LiDAR sensor in cm (Figure 4.25).

The web server has been tested using a LabView virtual instrument (**go_read_raspi_server.vi**) allowing for the assessment of the throughput rate contemporary to the use of the **go_raspi_test_w_camera.m** Matlab script (previously presented). The response rate to the **go_read_raspi_server.vi** was of 1 Hz, whereas the transfer rate of the Matlab script slowed down to 0.5 Hz.

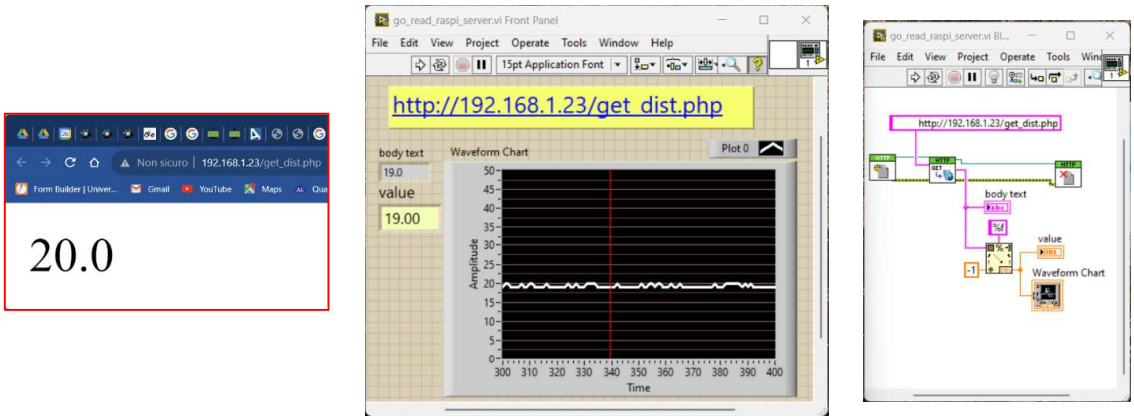
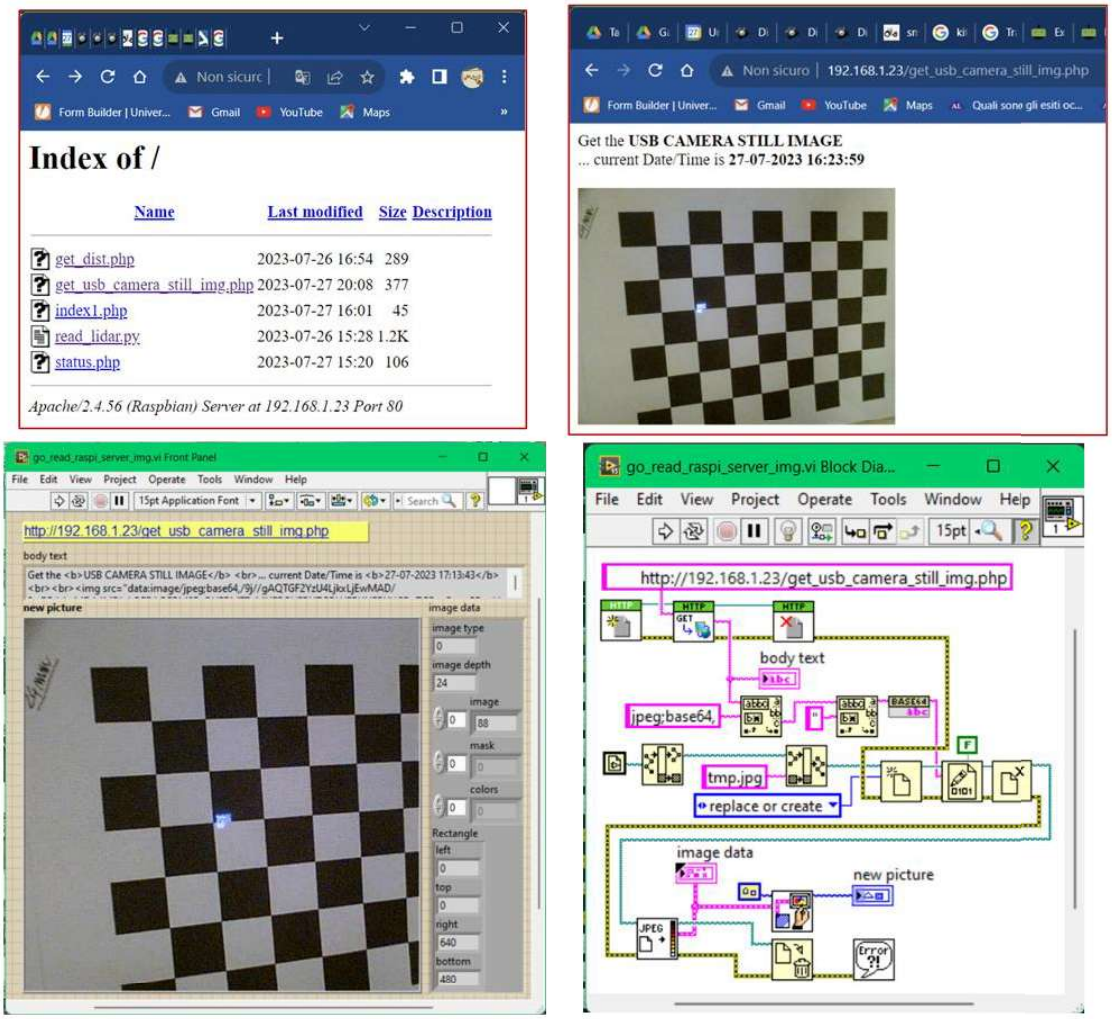


Figure 4.25.Response of the web server to a GET from the address http://192.168.1.23/get_dist.php: the distance measured by the LiDAR sensor in cm is outputted.

The **get_dist.php** file contains simply one line of PHP code: **<?php echo(shell_exec('python read_lidar.py')); ?>** Indeed, the hard work is done using a Python script (**read_lidar.py** available into supporting scripts companion folder https://tinyurl.com/2ytbcgp5) inspired to the previously developed **read_ser.py** Python script. This last implementation in considered definitive and closes the research related to the build of the measuring distance system by LiDAR.In addition, a few lines of PHP code allowed the handling of the request for a still image from an USB web camera connected to the Raspberry Pi hardware (**get_usb_camera_still_img.php** available into supporting scripts companion folder https://tinyurl.com/2ytbcgp5). The image is encoded as JPEG BASE64 using an **<img >** html tag, thus completely embedding the image into the response file (Figure 4



.26). The response rate resulted 0.5 Hz.

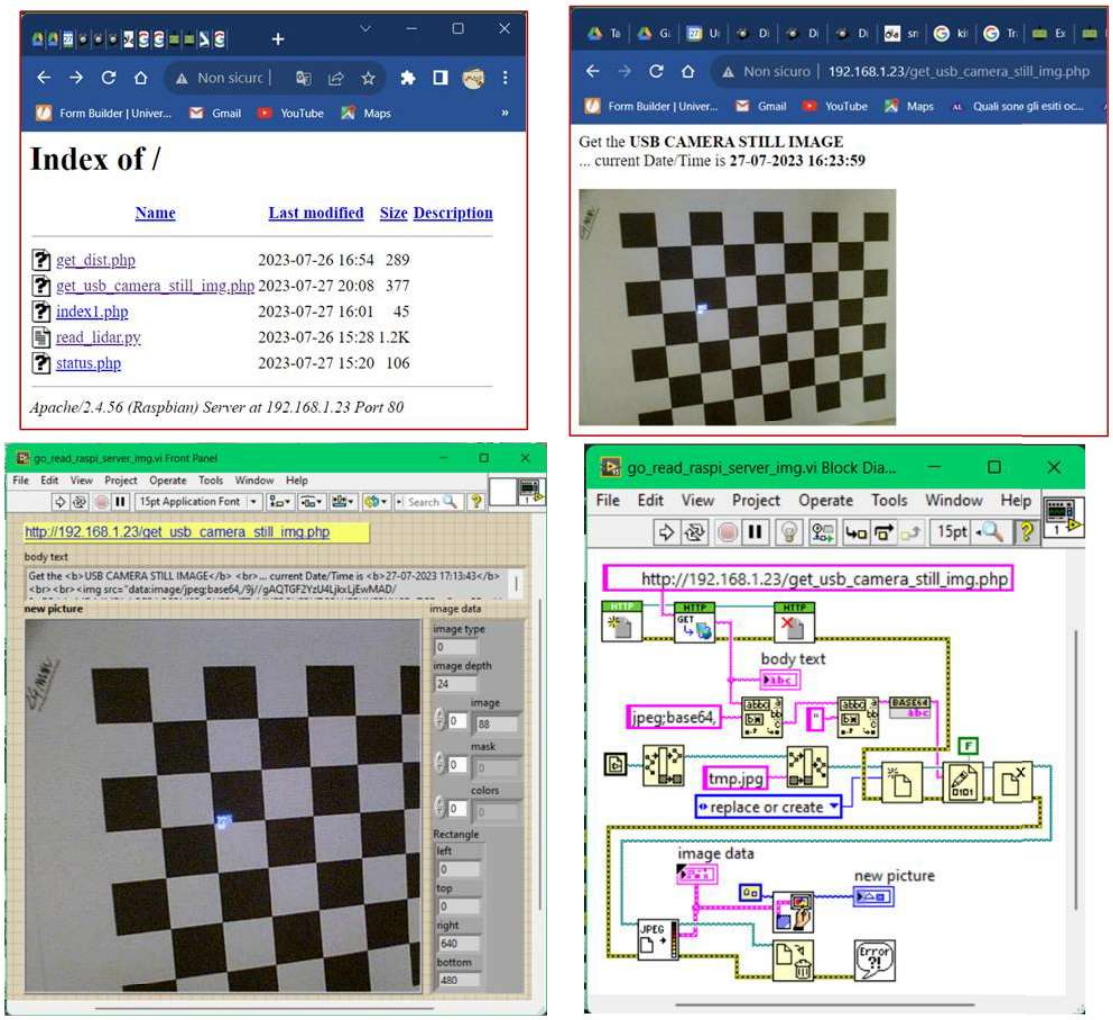


Figure 4.26. Response of the web server to a GET requests from http://192.168.1.23 (showing the contained files) and from http://192.168.1.23/get_usb_camera_still_img.php (showing the downloaded BASE64 encoded camera still image). In addition, a LabView virtual instrument (go_read_raspi_server_img.vi) has been built to demonstrate how to download and eventually store the JPEG image.

The proper functioning of the implemented system relies on the fact that the user ”www-data”, representing the Apache web server process, can access the “dialout” and the “video” resources. This is accomplished granting to the user “www-data” the access to the previously mentioned resources typing “sudo usermod -a -G dialout www-data” and “sudo usermod -a -G video www-data” into command console terminal.

4.4.6 RP2: Software implementation for Multispectral Camera Management and image alignment

This section describes the software architecture implemented on RP2 for multispectral camera management, distance-aware image alignment, and vegetation index computation. The system combines web-based services, compiled numerical routines, and ROS 2 integration to enable real-time and autonomous operation.

4.4.6.1 On-board installed software

The software system was installed and optimized for execution on a **Raspberry Pi 4 Model B** equipped with **8 GB of RAM** and a **128 GB microSD card**. This platform was chosen for its compactness, versatility, and compatibility with embedded Linux-based environments.

The operating system used is **Ubuntu 22.04.4 LTS (Jammy Jellyfish)** 64-bit (GNU/Linux 5.15.0-1049-raspi aarch64), which ensures system stability and full support for image processing libraries and ROS 2 dependencies.

The system was configured with **ROS 2 Humble Hawksbill**, installed according to the official ros.org documentation, to provide a modular, real-time infrastructure for managing robotic nodes, topics, and services.

The key installed packages include:

- **Backend languages and services**
  - python3.10 – version 3.10.12, used as the primary interpreter
  - apache2 – web server version 2.4.52 for local HTTP service management
  - php8.1 – version 8.1.2 for server-side processing and internal API support
- **Computer vision and image processing**
  - opencv-data and python3-opencv: arm64 – OpenCV 4.5.4 with ARM64 support, for video analysis and multispectral image processing

Installed MATLAB Add-ons:

To support the development and deployment of the multispectral image correction pipeline (see Section 4.3.5.3), the system includes several MATLAB packages:

- **MATLAB Compiler** – version 24.2: enables execution of compiled scripts independently of the MATLAB environment
- **MATLAB Coder** – version 24.2: supports automatic generation of C/C++ code from MATLAB algorithms
- **MATLAB Support Package for Raspberry Pi Hardware** – version 24.2.4: allows direct interfacing with GPIO, I2C, UART, and Raspberry Pi peripherals
- **Computer Vision Toolbox** – version 24.2: used for image pre-processing and automatic detection of calibration patterns
- **Image Processing Toolbox** – version 24.2: supports alignment and segmentation of multispectral imagery
- **Hyperspectral Imaging Library** – version 24.2.0: an extension for handling multiple spectral bands and processing high-resolution spectral images

This software configuration allows the RP2 system to operate autonomously, acquiring distance data from the LiDAR sensor, performing real-time multispectral image correction, and delivering the results via web or FTP interfaces for remote analysis.

4.4.6.2 Multispectral Camera Management: WEB Server

The Web Server–based management framework developed for the multispectral camera (here the webserver home https://tinyurl.com/235ocm5s), deployed on the Raspberry Pi, provides a lightweight and modular interface to trigger image acquisition, perform distance estimation, execute band alignment, and compute vegetation indices, enabling both standalone operation and integration within the ROS 2 robotic infrastructure.

Preliminarily, it must be noted that a command is available to activate the system's demo mode by creating a placeholder file in the temporary file system directory:

ab_activate_demo_mode.sh

```
#!/bin/bash
# chmod +x ap_activate_demo_mode.sh
```

```
# Create an empty file
touch /dev/shm/DEMOMODE.DAT
```

To revert the system to normal mode:

ab_deactivate_demo_mode.sh

```
#!/bin/bash
# chmod +x ap_deactivate_demo_mode.sh
# Delete the file
rm /dev/shm/DEMOMODE.DAT
```

4.4.6.2.1 Web Server–based distance estimation using stereo vision

The first web server command, attempts to acquire images from the multispectral camera and then applies the binocular algorithm on the Red vs. Red Edge band (see Figure 4.12) to estimate the target distance (stereo camera calibration), followed by the image registration algorithm to align the image bands.

The distance estimation script (**raspi_dist_estimate.m**) was initially developed in MATLAB and then converted into C++ using MATLAB Coder for deployment on Raspberry Pi hardware.

Example Web Server Response: `http://192.168.4.23/_raspi-dist-estimate/`

This endpoint returns the acquired multispectral images and the processed outputs, as shown by the directory listing below.

Directory Listing Output: (https://tinyurl.com/28qscsys)

```
Parent Directory                    -
B.tif                     2025-02-27 21:24    3.0M
G.tif                     2025-02-27 21:24    3.0M
NR.tif                    2025-02-27 21:24    3.0M
R.tif                     2025-02-27 21:24    3.0M
RE.tif                    2025-02-27 21:24    3.0M
_go_final_test_dist.php   2025-02-27 18:27   66
_go_final_test_dist_noimg.php 2025-02-27 18:27   66
_test_dist_common.php     2025-02-27 21:30   8.1K
go_raspi_dist_estimate.php 2025-02-17 09:15   1.5K
go_test_dist.php          2025-02-20 15:59   5.8K
raspi_calc_indices.py     2025-02-24 18:56   2.9K
raspi_calc_indices_v2.py  2025-02-26 15:02   2.3K
raspi_dist_calc.elf       2025-02-24 14:59   21K
raspi_dist_estimate.elf   2025-02-19 14:00   128K
user_dist_fun.php         2025-02-24 14:32   4.1K
```

The following Web Server command triggers the full acquisition and stereo-based distance estimation pipeline:

Web Server Camera Invocation:

`http://192.168.4.23/_raspi-dist-estimate/go_test_dist.php`

This operation requires approximately 9 seconds to complete, including image acquisition, stereo distance computation, and band alignment. As shown in Figure 4.27 the spatial displacement between the Red and Red Edge bands is exploited to perform stereo calibration. This approach enables distance-aware band registration but requires significant computational resources.

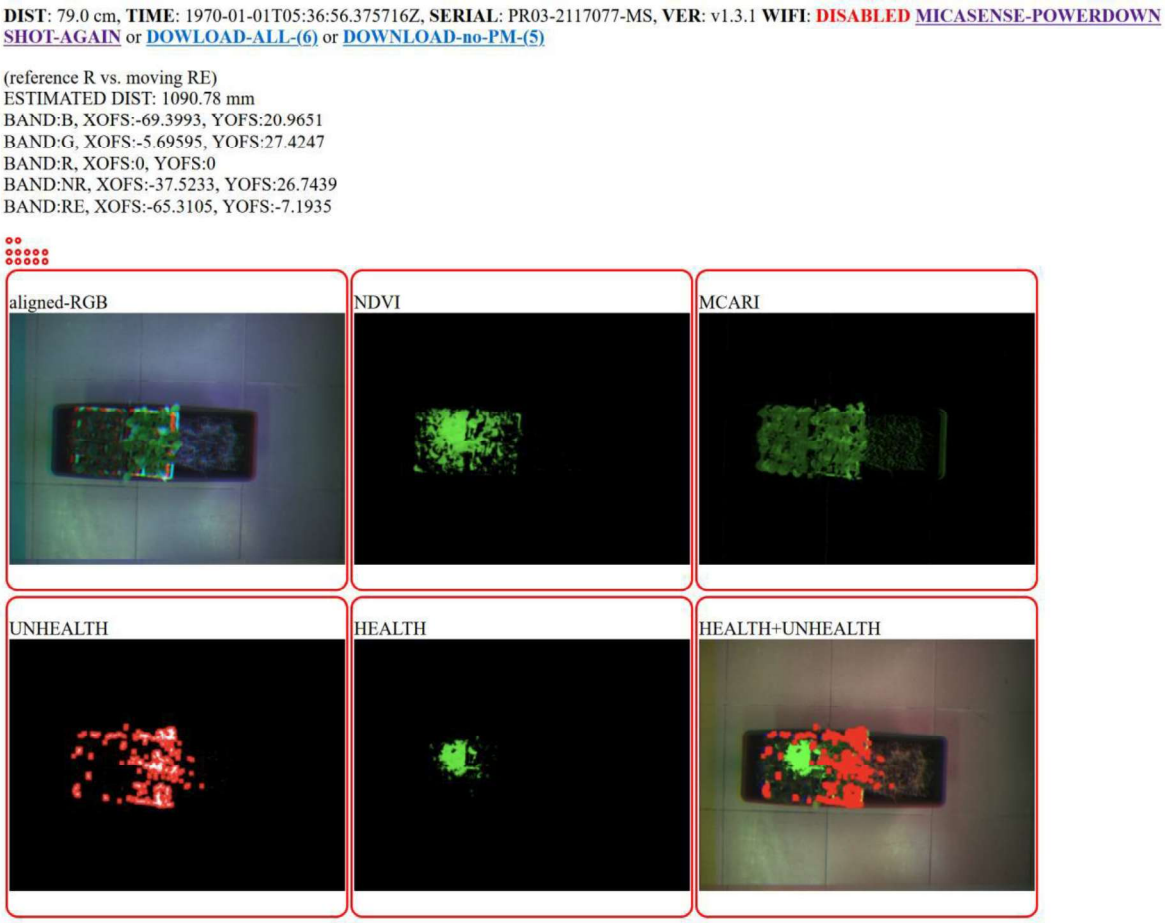


Figure 4.27. Stereo-based distance estimation exploiting spatial offsets between Red and Red Edge bands of the multispectral camera.

4.4.6.2.2 LiDAR-based distance estimation via Web Server

In addition, it has been tested the measure of the target distance using the Lidar sensor.

The response from the Web Server: `http://192.168.4.23/_raspi-dist-estimate/_go_final_test_dist.php` triggers image acquisition and subsequently performs distance measurement using the LiDAR sensor. The entire process takes approximately **5 seconds** (Figure 4.28).

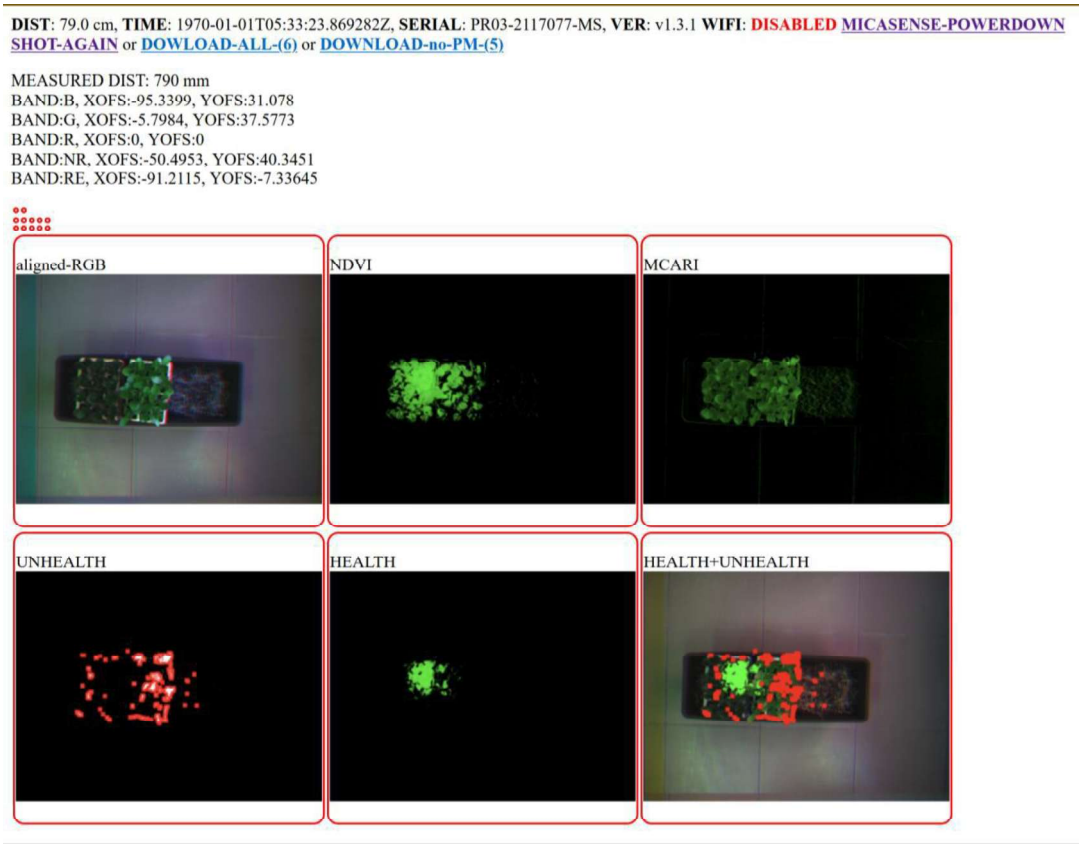


Figure 4.28. LiDAR-based target distance measurement integrated into the multispectral acquisition pipeline.

4.4.6.2.3 Performance comparison and design choice

The reduced execution time achieved using the LiDAR-based distance measurement makes this solution more suitable for integration within the **ROS 2 robotic network infrastructure**, where communication bandwidth and processing speed are critical constraints.

While stereo-based algorithms provide accurate results for image alignment, they impose a substantial computational burden on Raspberry Pi hardware. In contrast, LiDAR-based distance estimation offers:

- lower computational complexity,
- higher achievable frame rate,
- improved responsiveness for real-time operation.

Therefore, based on these evaluations, LiDAR-based distance measurement was selected as the preferred solution for real-time multispectral camera management.

4.4.6.2.4 Web Server API for ROS 2 integration

An additional Web Server endpoint was developed to support headless operation without returning image data in the HTTP response. The Web Server response: http://192.168.4.23/**_raspi-dist-estimate/_go_final_test_dist_noimg.php** captures the image and computes the distance using the LiDAR sensor, but without outputting any image in the response. The data are stored in the temporary file system of the Raspberry Pi. This API call is primarily used by the ROS 2 Micasense Node Management system.

4.4.6.2.5 Vegetation index computation and plant health mapping

Following alignment, NDVI and MCARI vegetation indices are computed, and HEALTHY/UNHEALTHY plant masks are generated using the Python scripts **raspi_calc_indices.py and raspi_calc_indices_v2.py**. These processes represent the core of the automated multispectral camera management framework onboard the Raspberry Pi.

4.4.6.3 Multispectral Camera Management: Real-time application by ROS2 Publisher Node

To integrate the Web Server–based camera management pipeline into the ROS 2 ecosystem, a dedicated publisher node was developed in Python.

The building of the Micasense Node Publisher in ROS2, using Python, is very

```
# package creation
cd ~
mkdir ~/micasense_ws mkdir ~/micasense_ws/src cd ~/micasense_ws/src
ros2 pkg create --build-type ament_python --node-name mymica_node
mymica_pkg
# package building
cd ~/micasense_ws colcon build
# package sourcing
source install/local_setup.bash

# to be inserted into ~/.bashrc
source ~/micasense_ws/install/setup.bash

# node launch
ros2 run mymica_pkg mymica_node
```

straightforward.

The data flow implemented by the ROS 2 node can be summarized as follows:

1. The node periodically triggers the Web Server API for image acquisition and distance measurement.
2. The Web Server executes the multispectral processing pipeline and stores the results in the temporary file system.
3. The ROS 2 node retrieves the processed image and associated metadata.
4. The image is published as a ROS 2 topic for visualization and downstream processing.

The Python source file that does the hard work is listed below:

Python code:

mymica_node.py

```
import    rclpy
import numpy as
np import cv2
from rclpy.node import Node
from   sensor_msgs.msg   import
Image   from   cv_bridge   import
CvBridge import urllib.request

eor='/dev/shm/'

class MyMica_Pub(Node):
def __init__(self):
    super().__init__('mymica_pub')
    self.publisher_   =   self.create_publisher(Image,   'mymicaIMG',   3)
    timer_period = 2  # seconds
    self.timer = self.create_timer(timer_period, self.timer_callback)
self.bridge= CvBridge()
print('*** Started ...')
    def timer_callback(self):
        self.get_logger().info('\n---  READING
        URL\n') try:
```

```
            with                    urllib.request.urlopen('http://127.0.0.1//_raspi-dist-
estimate//_go_final_test_dist_noimg.php', timeout=10) as furl:
                furl.read()
            self.get_logger().info('\n---     Image     START\n')
            img=cv2.imread(eor+'FINAL.tif')
            if img is None:
                self.get_logger().info('\n--- Image *** IS NULL
***\n') else:
                with  open(eor+'measured_dist.txt','r')  as
                    fid: dist_data = fid.readline()
                with  open(eor+'timed.txt','r')  as
                    fid: time_data = fid.readline()
                text=    '(DIST:    '+dist_data.strip()+'    mm)
                '+time_data.strip()            font            =
                cv2.FONT_HERSHEY_SIMPLEX
                org = (10, 40)
                fontScale = 1
                color = (0, 255, 255)
                thickness = 2
                img = cv2.putText(img, text, org, font, fontScale, color, thickness,
                cv2.LINE_AA)    self.publisher_.publish(self.bridge.cv2_to_imgmsg(img))
                self.get_logger().info('\n--- Image END\n')

        except:
            self.get_logger().info('\n--- Image *** TIMED OUT ***\n')
def main(args=None):
    rclpy.init(args=arg
    s)   mymica_pub   =
    MyMica_Pub()
    rclpy.spin(mymica_p
    ub)
    mymica_pub.destroy_
    node()
    rclpy.shutdown()
if __name
```

It should be noted that the call

```
urlopen('http://127.0.0.1//_raspi-dist-estimate//_go_final_test_dist_noimg.php',…)
```

invokes the local Web Server running on the Raspberry Pi, which exposes the API interface for multispectral camera management. Through this API, image acquisition, distance measurement, and preprocessing are triggered without direct interaction with the camera at the ROS 2 node level.

The PHP script **`_go_final_test_dist_noimg.php`** acts as a lightweight dispatcher to the common processing routines implemented on the embedded platform. In this configuration, image data are processed and stored in the temporary file system, while no image content is returned in the HTTP response, enabling efficient headless operation.

```
<?php
    $yesimg=0;
    require_once('_test_dist_common.
    php');
?>
```

After building the ROS 2 workspace:

```
# package building
cd ~/micasense_ws
colcon build
```

the ROS 2 publisher node can be launched using:

```
# node launch
ros2 run mymica_pkg mymica_node
```

Real-time visualization of the published multispectral image stream was performed activating the ROS Foxglove bridge (https://docs.foxglove.dev/docs/connecting-to-data/ros-foxglove-bridge) into both the Raspberry PI and Windows (Figure 4.29). The topic **`mymicaIMG`** was visualized at an effective frame rate of approximately **0.25 Hz** (one frame every 4 seconds), confirming correct end-to-end data flow from Web Server–based camera management to ROS 2 publication.

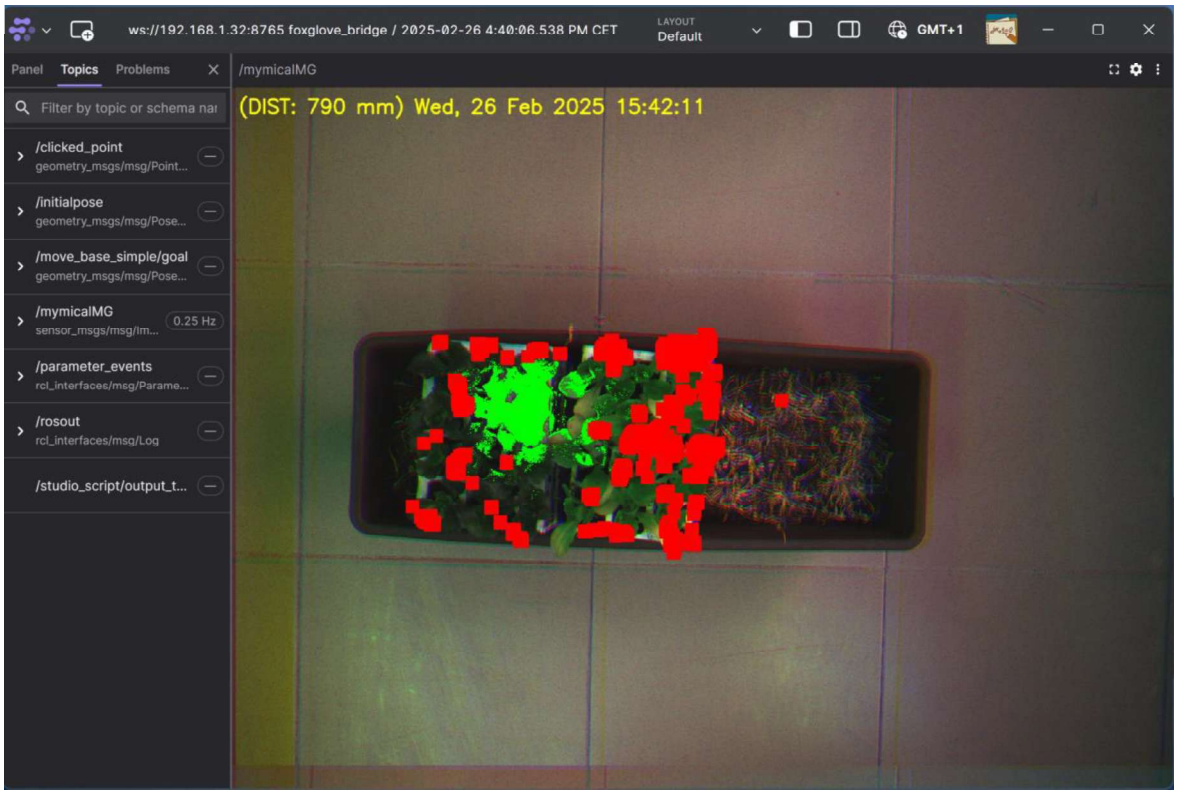


Figure 4.29. Real-time visualization of the published topic was performed using the ROS Foxglove bridge.

The subsequent activities will be related to the formalization of a decision system based on HEATHY and UNHEALTHY maps.

4.4.7 RP3: the motion management ROS2 node (Raspberry PI 4 & Arduino Mega 2560)

This section describes the motion management architecture implemented on RP3 and its integration within the ROS 2 framework.

While the propulsion system described in Section 4.3.3 focuses on the mechanical design, actuation hardware, and low-level embedded control enablement, the present section addresses the runtime coordination of motion commands, sensor feedback, and safety mechanisms during system operation.

The RP3 node acts as the interface between high-level ROS 2 motion commands and the Motor Control Interface (MCI), enabling continuous and reliable actuation of the propulsion system.

4.4.7.1 Motor Control Interface (MCI)

The Arduino Mega 2560 in this section is used as the Motor Control Interface (MCI) allowing the motors management and the linking into the ROS2 messages infrastructure using RP3 and the USB serial port of the MCI (serial port #0); the Kangaroo controller management is performed using the MCI serial port #1. MCI is powered by RP3. To RP3 is connected the GPS AHRS IMU, WTGAHRS2 MPU9250 2D Lidar in order to allow to perform some inverse kinematics algorithms. A detailed schematic of the electrical connections between the MCI, motor driver, and power supply is shown in Figure 4.30.

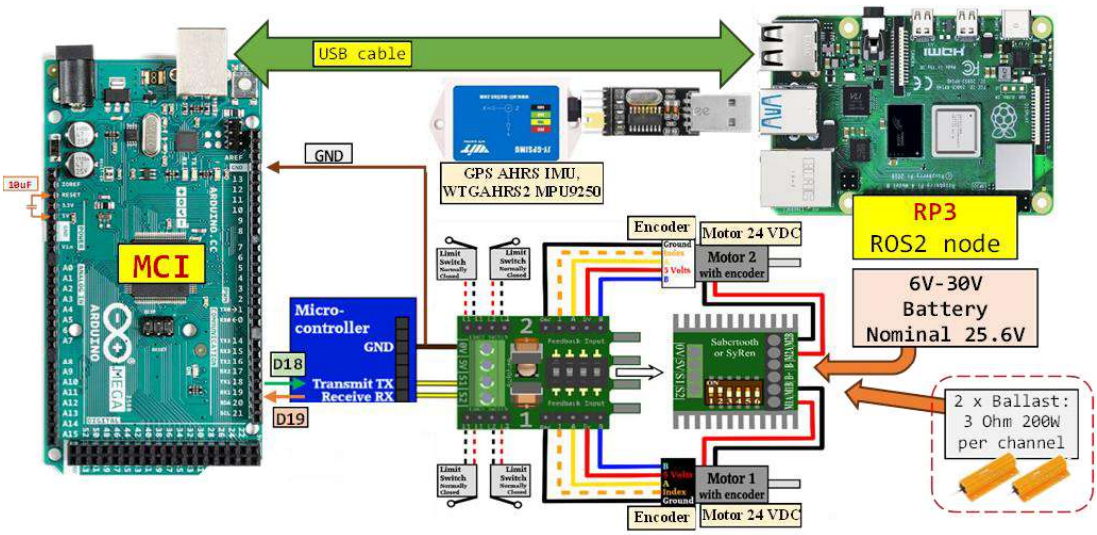


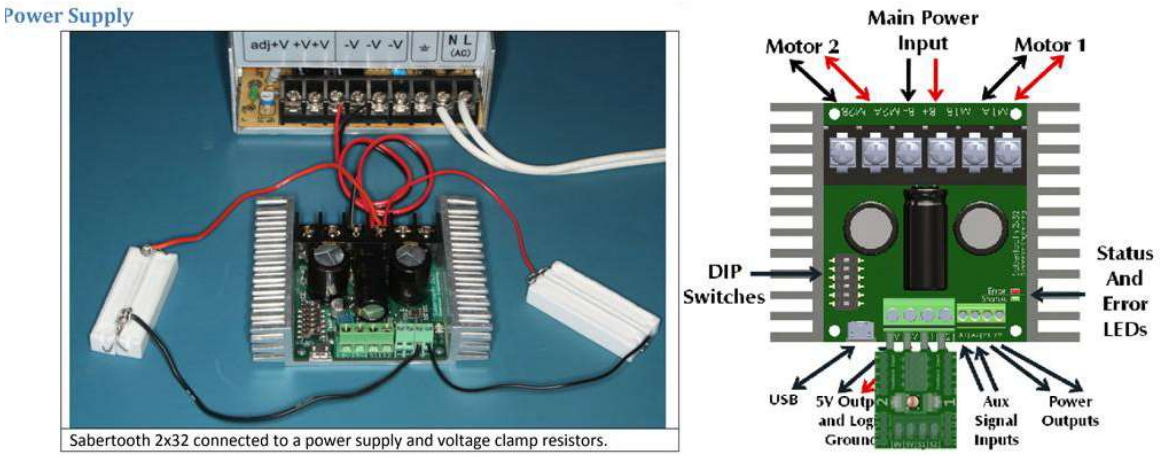
Figure 4.30. Electrical connections schematic of the motion management ROS2 node (Raspberry PI 4 & Arduino Mega 2560) (RP3).

When opening or closing the USB serial connection, the Arduino Mega 2560 is automatically reset. This behavior is a design feature that simplifies firmware uploading but may interfere with continuous motor control operation. The Arduino is reset when the

connection is next established (i.e. when the serial port is opened). This behavior is by design. To overcome this a 10 uF capacitor must be connected between RESET and +5V or GND pins (the connection to GND is the preferred one). This hardware modification ensures stable serial communication during continuous operation of the motion management node.

4.4.7.2 System integration and sensor interfaces

In addition to motor control, RP3 interfaces with positioning and orientation sensors, including GPS, AHRS, IMU (WTGAHRS2 MPU9250), and a 2D LiDAR. These sensors provide the necessary inputs for future implementation of inverse kinematics and navigation algorithms. The propulsion system is powered through a **Sabertooth 2×32 motor driver**, which supports supply voltages in the range **7–30 V**. The selected power supply must be capable of delivering sufficient current to meet the motor load requirements. In this configuration, **Power Supply mode** is enabled by setting **DIP switch 3 to the ON position**. A key feature of the third-generation Sabertooth motor drivers is the availability of regenerative power outputs (P1 and P2). These outputs allow dissipating the energy generated during deceleration or stopping phases by connecting appropriate resistor packs. This solution enables the use of power supplies without batteries or large capacitors to absorb regenerated energy. At least one of the power outputs must be configured as a voltage clamp when operating in power supply mode. The detailed electrical connections of the Motor Control Interface (MCI) are shown in Figure 4.31.



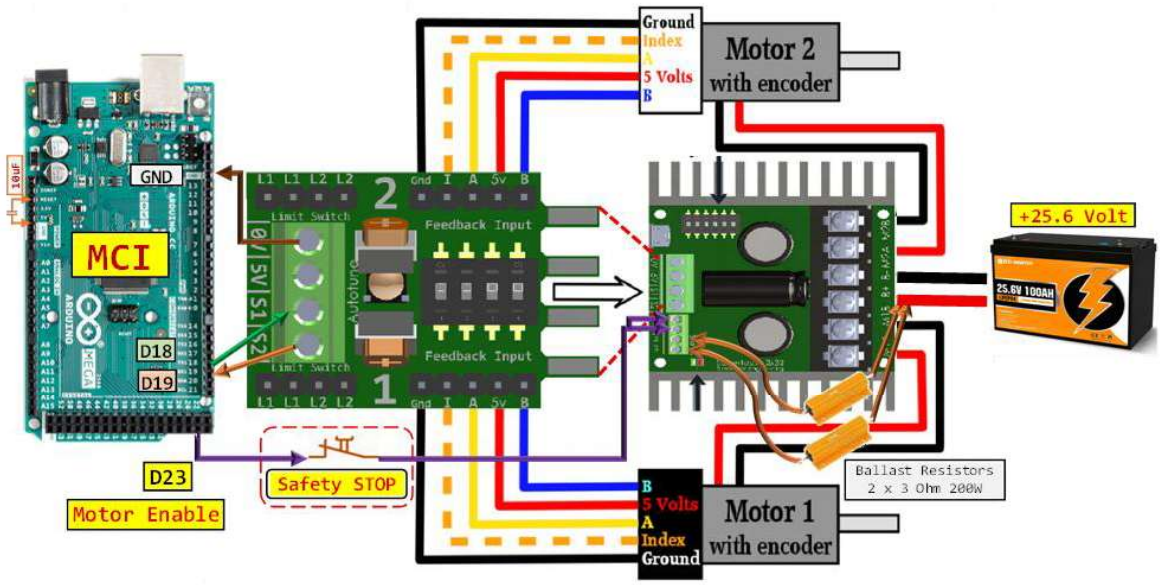


Figure 4.31. Detailed electrical connections schematic of the Motor Control Interface (MCI).

4.4.7.3 Motor driver control modes and configuration:

The Sabertooth 2×32 motor driver supports multiple control modes, selectable through DIP switches. In this work, Serial Mode was selected to enable direct control from the Arduino-based MCI. **Sabertooth 2x32** has four main control modes: 1) Analog Mode, 2) Radio Control Mode,3) Serial Mode,4) USB Mode, 5) User-defined Mode (Figure 4. 32). This last one can be used to create custom control modes. Control Modes are selected via the six DIP switches. DIP switches 1 and 2 select the control mode, and DIP switches 4, 5, and 6 select the options within each control mode. The key feature of the Sabertooth 2×32 driver is the availability of regenerative power outputs (P1 and P2), which allow dissipation of regenerated energy during deceleration or stopping. Emergency stop functionality is not available when operating in serial converter mode.

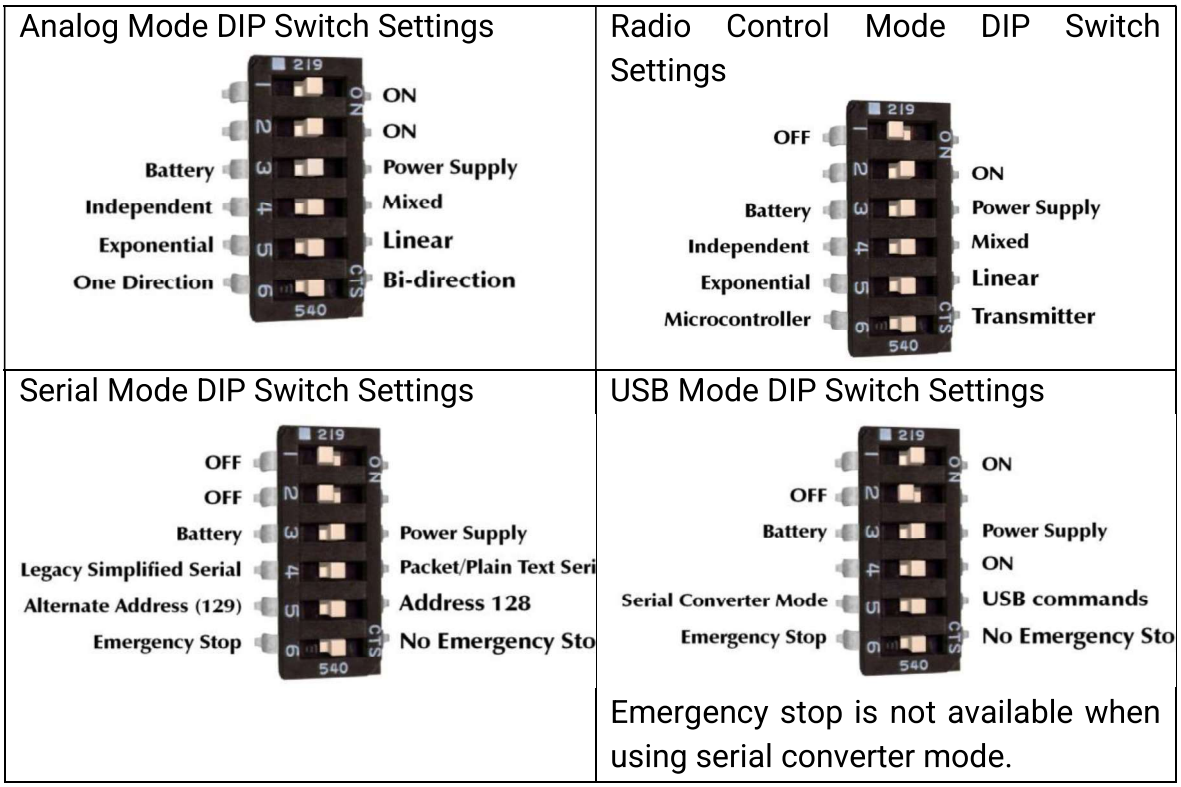


Figure 4.32. Pin configuration and control mode available for the Sabertooth 2×32 motor driver.

The selected configuration, corresponding to Serial Mode DIP Switch Settings, is shown in Figure 4.33.

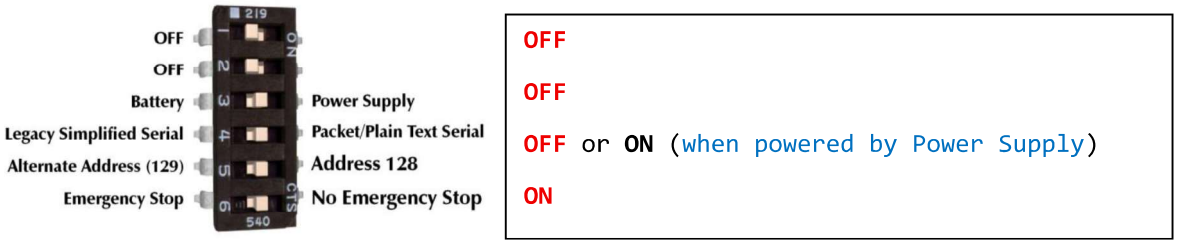


Figure 4.33. Pin configuration and control mode selection of the Sabertooth 2×32 motor driver.

Kangaroo X2 motion controller configuration:

The **Kangaroo X2** motion controller operates as a closed-loop speed and position controller, providing encoder-based feedback for precise motor regulation**.** Its operating mode and options are configured through a four-position DIP switch, allowing seamless integration with the Sabertooth motor driver and the Arduino-based MCI. Figure 4.34 shows the main ON and OFF configuration parameters of the Kangaroo X2 motion controller.

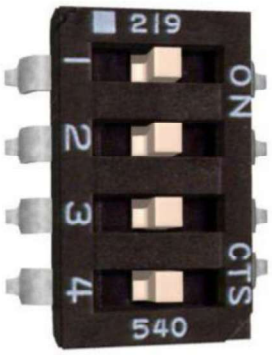
OFF setting		ON setting
1 off: Analog input. Connect 0-5V analog signals to the S1 and S2 inputs.		**1 on: Digital input** Connect TTL serial, TX to S1 and RX to S2, or R/C servo signals to S1 and S2
2 off: Analog feedback Connect a 0-5V signal to Feedback Input A		**2 on: Quadrature feedback** Connect an encoder to Feedback Inputs A and B
3 off: Velocity control Motor speed and direction are controlled by the input signal		**3 on: Position control** Motor position is controlled by the input signal
4 off: Mixed mode The outputs are mixed together for differential drive mobile robots		**4 on: Independent mode** The outputs are independent. S1 controls motor 1 and S2 controls motor 2.

Figure 4.34. ON/OFF configuration settings of the Kangaroo X2 motion controller.

The **Kangaroo x2** selected mode is **ON-ON-OFF-ON**.

This configuration enables reliable feedback-based motor control suitable for integration within the ROS 2 motion management framework. Overall, the RP3 motion management node provides a robust interface between high-level ROS 2 motion commands and low-level motor actuation hardware. The combination of dedicated motor controllers, embedded microcontroller mediation, and careful electrical design ensures reliability, scalability, and readiness for future autonomous navigation functionalities.

4.4.8 RP1: device collecting RPs telemetry data over the ROS2 network (Raspberry PI 4 connected to OMC through USB)

RP1 is dedicated to the collection and aggregation of telemetry data generated by the distributed Raspberry Pi devices operating within the ROS 2 network. Acting as a centralized telemetry node, RP1 subscribes to device-specific ROS 2 topics and stores the received data in a temporary shared memory workspace, making them available to the Overall Master Control (OMC) upon request through a USB serial connection.

Each Raspberry Pi device is assigned a unique numerical identifier (as an integer), stored as a temporary file in the RAM disk workspace: **/dev/shm/THIS_RP_DEVICE_NAME.**

Since ROS 2 topic names cannot be purely numeric, the device identifier is converted into an alphabetical label. Accordingly, telemetry topics are defined as:

Therefore, the topic will be “rp_telemetry_a” for the ID=1, “rp_telemetry_b” for the ID=2 and so on.

This package will subscribe to the appropriate topic (e.g. **/rp_telemetry_a** for RP1 and so on) and will store the collected data into the temporary ram disk workspace (e.g., file: **/dev/shm/RPA**).

Then, **OMC** will request the information to **RP1** using its serial port 0.

When the USB serial connection is established, the Arduino Mega 2560 connected to the OMC undergoes an automatic reset, which is a default design behavior. To prevent unintended resets during continuous telemetry exchange, a 10 µF capacitor must be connected between the RESET pin and GND or +5 V. The connection to GND is the preferred one

This hardware mitigation ensures stable and uninterrupted serial communication between RP1 and the OMC.

Package creation for the telemetry subscriber:

A dedicated ROS 2 Python package was created to implement the telemetry subscriber node.

1. Package creation for the telemetry subscriber

```
# package creation
cd ~/this_rp_device_telemetry_ws/src
ros2 pkg create --build-type ament_python --node-name
rp_subscriber_telemetry_node rp_subscriber_telemetry_pkg
# package building
cd ~/this_rp_device_telemetry_ws
colcon build
# package sourcing
source install/local_setup.bash
# to be inserted into ~/.bashrc
source ~/this_rp_device_telemetry_ws/install/setup.bash
# node launch
ros2 run rp_subscriber_telemetry_pkg rp_subscriber_telemetry_node
```

2. Package implementation in Python (this_rp_device_telemetry_ws/src)

The telemetry subscriber node is implemented in Python and is located at:

```
Contents of
./rp_subscriber_telemetry_pkg/rp_subscriber_telemetry_pkg/rp_subscriber_telemetry
_node.py

(here is the source code: https://tinyurl.com/297vnam9 )
```

3. Package launch

The telemetry subscriber can also be launched using the following shell script:

```
~/go_subscribe_telemetry.sh
(content: ros2 run rp_subscriber_telemetry_pkg
rp_subscriber_telemetry_node)

During startup, the node reports successful initialization and creates
subscriptions for all configured telemetry topics:

* rp_subscriber_telemetry_node STARTED.
* Subscriptions created for rp_telemetry_a, rp_telemetry_b, rp_telemetry_c,
rp_telemetry_d, rp_telemetry_e, rp_telemetry_f, rp_telemetry_g, and
rp_telemetry_h.
```

This confirms the scalability of the telemetry collection framework and its capability to support multiple Raspberry Pi devices operating concurrently within the ROS 2 network.

4.4.9 LCD devices electrical connections

The integration of multiple distributed Raspberry Pi devices, sensors, and control modules within a ROS 2-based architecture requires effective tools for system supervision, diagnostics, and runtime monitoring.

Building upon the hardware architecture described in Sections 4.3.3 (propulsion system), 4.3.6 (multispectral camera management), and 4.3.7 (motion management ROS 2 node), the present section illustrates the implementation of a human-readable monitoring layer based on the Overall Master Control (OMC) and real-time ROS 2 visualization tools. The combination of local LCD-based feedback and network-level ROS 2 introspection enables continuous monitoring of system status, telemetry data, sensor availability, and inter-node communication, providing both low-level robustness and high-level situational awareness during experimental operation.

Four LCD displays (20 columns × 4 rows) are connected to the Arduino Mega 2560 acting as the Overall Master Control (OMC) using the I²C communication interface. The I²C bus is implemented through the following Arduino pins: **SCL** (pin **21**) and **SDA** (pin **20**) (Figure 4.35).

Figure 4.35. LCD-based system status visualization implemented on the Overall Master Control (OMC). The four I²C-connected displays provide real-time feedback on system initialization, device availability, ROS 2 communication status, power conditions, and safety flags, enabling local supervision of the distributed robotic architecture.

Each LCD module is assigned a unique I²C address through the configuration of the address selection pins: the addresses used are **0x27** (**LCD1**, A0 open, A1 open, A2 open), **0x26** (**LCD2**, A0 **closed**, A1 open, A2 open), **0x25** (**LCD3**, A0 open, A1 **closed**, A2 open), **0x24**

(**LCD4**, A0 **closed**, A1 **closed**, A2 open). In addition, the +5V and GND pins of the Arduino MEGA 2560 device (OMC) are used to power the 4 LCD.

The Arduino Mega 2560 (OMC) program

The Overall Master Control (OMC) is managed by a program written in C (here is the latest developed source code: https://tinyurl.com/2cyypbja).

The OMC firmware coordinates are scheduled as: **RP1** (telemetry) and **RP2** (Micasense camera) and Efento environmental sensors (Figure 4.36).

Figure 4.36. Initialization and runtime status screens displayed on the four I²C-connected LCD modules of the Overall Master Control (OMC), showing system version, device checks, ROS 2 communication status, power supply information, and safety-related flags.

During system operation, the ROS 2 network and active topics are monitored using the **Foxglove debugger**, which provides a real-time overview of data exchange across the distributed architecture. An example of the system running with RP1, RP2, and environmental sensors is shown in the figure below (Figure 4.37).

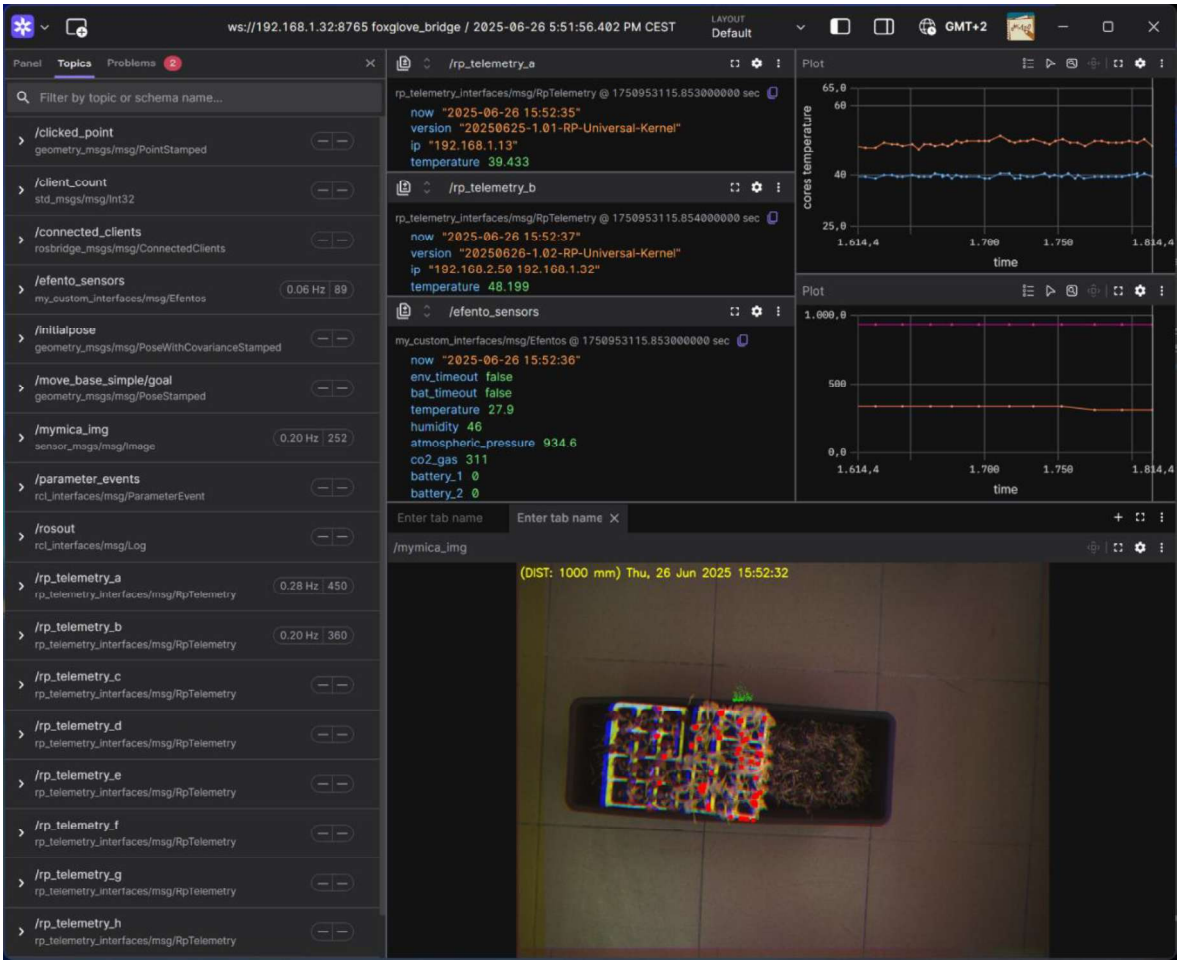


Figure 4.37. ROS 2 network visualization using the Foxglove debugger. The interface shows active telemetry topics from multiple Raspberry Pi devices, environmental data from Efento sensors, and multispectral image streams, confirming correct end-to-end data flow across the distributed system.

4.4.10 RP5: Joystick management for robot guidance

The joystick-based control represents a practical and safe solution for the initial testing, calibration, and supervision phases of the robotic platform, prior to the activation of fully autonomous navigation strategies. For these reasons, joystick-based teleoperation was selected as the primary interface for initiating and supervising robot motion during experimental activities.

Robot teleoperation is implemented using **Twist-based ROS 2 messages**, which represent the standard abstraction for mobile robot velocity control. The script **go_joy.sh** will allow tele-operating the robot using Twist-based ROS 2 messages.

The source files have been downloaded from https://github.com/ros2/teleop_twist_joy.

The purpose of this package is to provide a generic facility for tele-operating Twist-based ROS 2 robots with a standard joystick. It converts joy messages to velocity commands.

The joystick interface is activated through the following launch script:

go_joy.sh

```
#!/bin/bash
ros2 launch teleop_twist_joy teleop-launch.py
config_filepath:=ps3_config.yaml
```

ps3_config.yaml

```
teleop_twist_joy_node:
  ros__parameters:
    axis_linear:
      x: 4
    scale_linear:
      x: 0.7
    scale_linear_turbo:
      x: 1.5
    axis_angular:
      yaw: 0
    scale_angular:
      yaw: 0.4
    enable_button: 4
    enable_turbo_button: 5
```

Figure 4.38 displays the mapping of buttons and analog axes on the PlayStation 3 Sixaxis controller, used for robot teleoperation via the teleop_twist_joy ROS 2 package. Axis directions and button numbers are overlaid, showing how linear and angular velocities are assigned (e.g., Axis 4 → linear x, Axis 0 → angular yaw). Buttons 4 and 5 are configured as the enable and turbo buttons, respectively. This configuration enables real-time proportional control of robot motion using standard ROS 2 Twist messages.

Figure 4.38. Buttons and Axes mapping for the PS3 controller.

The package comes with the **teleop_node** that republishes **sensor_msgs/msg/Joy** messages as scaled **geometry_msgs/msg/Twist** messages. The message type can be changed to **geometry_msgs/msg/TwistStamped** by the **publish_stamped_twist** parameter. This design ensures full compatibility with ROS 2 navigation and motion control pipelines, while maintaining flexibility for future extensions.

4.4.11 The Universal Kernel Image (UKI)

The Linux system management of the overall RP devices is produced and maintained as a compressed image called Universal Kernel Image (**UKI**) (here are the developed images available until now: https://tinyurl.com/28swqkhc).

The **UKI** is self-adapting to the RP device number (as hard coded using the Raspberry PI 4 GPIO). GPIO26, GPIO22, GPIO27 are used as inputs (with pullup active) to hardcode the number identifying the RP device (1 to 8). 0, 0, 0 identifies RP1, 0, 0, 1 identifies RP2, and so on.

4.4.12 The ABOTx1 software ROS2 node development

DeepSeek-V3 R1 has been asked about the following question.

"*I want to implement a two-wheeled robot using ROS 2 framework 'humble'. A PS3 joystick publishes /cmd_vel topic. I want to use a Raspberry PI 4 and an Arduino Mega 2560. The Arduino Mega 2560 is powered by Raspberry PI 4. I have a 2D Lidar SLD PRPLD-C1M1 (RPLIDAR-C1M1) connected using USB to the Raspberry PI 4. I have an IMU sensor WTGAHRS2 connected using a TTL-USB converter to the Raspberry PI 4. The PS3 joystick is connected to the Raspberry PI 4. I have already developed the IMU system. I have already developed the Lidar system. The IMU emits the topic /imu_data as a /sensor_msgs/msg/Imu. The Lidar emits the topic /scan as a /sensor_msgs/msg/LaserScan. The joystick emits the topic /cmd_vel as a /geometry_msg/msg/Twist. The motors are two brushed motors of 250W 24VDC managed using a Sabertooth 2x32A and a Kangaroo interface. The Kangaroo device is powered by the Sabertooth.. The Kangaroo device is connected to the Arduino Mega 2560 by serial port. I want to use a CALT GHS52-8G3600BMP526 encoder with power supply 5-26 VDC, PNP Push Pull with pulse output signal of 3600 pulses per rotation. The encoder is connected to the Kangaroo device. The connection between motors, wheels and encoders is provided by chain using sprockets. Each wheel is 20'' diameter and has a 32 teeth sprocket, each motor has a 9 teeth sprocket and*

each encoder has a 10 teeth sprocket. The encoder is connected to the motor by chain. The motor is connected to the wheel by chain. The motors are two. The wheels are two. The encoders are two. I want to use also the Z output of each encoder to the Kangaroo device. The encoders are electrically connected to the Kangaroo device. We have two 50Ah LiFePO4 batteries connected in series delivering 25.6 V to the Sabertooth. A step-down converter (30V to 5 V, 3 A) powers the Raspberry PI 4. I want the more detailed as possible software implementation using Python and C++, the more detailed as possible electrical schematics possibly graphical, and a detailed layout of the implementation more detailed as possible. The system must be fully configurable with all the required constant parameters held into separate yaml files, as for example wheels diameter, distance between wheels, geometry of the robot, identifications of the serial ports used and speeds of the serial ports. The node related to PS3 joystick, IMU and Lidar are already implemented and are started manually. The IMU node can be started using the bash command /home/ubuntu/go_myimu.sh. The Lidar node can be started using the bash command /home/ubuntu/go_mylidar.sh. The joystick node can be started using the bash command /home/ubuntu/go_joy.sh. The software implementation must use Python whenever possible. I want an in depth and detailed description of the system and related software components. I want to fuse the sensors to increase the movement precision tracking. I want an unique file in bash to create directory structure and populate all the content of the related files required by the implementation. I want to use the directory my_x1_ws as primary workspace. All the files must reside in this workspace. All files and package names must start with "x1_". Even a software and hardware security stop of the robot must be implemented."

The AI responded with the following directory structure. Some files needed some changes as they showed errors at compile time. The workspace structure is available here: https://tinyurl.com/ytvvpdat .

`colcon build --symlink-install` now correctly compile the workspace, and, after sourcing it by **`source install/setup.bash`**, the run command (**`ros2 launch x1_control x1_bringup.launch.py`**) now correctly starts the node.

```
ubuntu@ubuntu:~/my_x1_ws$ tree src/
src/
├── x1_bringup
│   ├── launch
│   ├── package.xml
│   ├── resource
│   │   └── x1_bringup
```

```
│   ├── scripts
│   │   └── master_launch.sh
│   ├── setup.cfg
│   ├── setup.py
│   ├── test
│   │   ├── test_copyright.py
│   │   ├── test_flake8.py
│   │   └── test_pep257.py
│   └── x1_bringup
│       └── __init__.py
├── x1_control
│   ├── arduino
│   │   └── x1_kangaroo_driver
│   │       └── x1_kangaroo_driver.ino
│   ├── config
│   │   ├── ekf_params.yaml
│   │   └── x1_params.yaml
│   ├── launch
│   │   └── x1_bringup.launch.py
│   ├── package.xml
│   ├── resource
│   │   └── x1_control
│   ├── setup.cfg
│   ├── setup.py
│   ├── test
│   │   ├── test_copyright.py
│   │   ├── test_flake8.py
│   │   └── test_pep257.py
│   └── x1_control
│       ├── __init__.py
│       ├── __pycache__
│       │   └── __init__.cpython-310.pyc
│       └── scripts
│           ├── __init__.py
│           ├── __pycache__
│           │   ├── __init__.cpython-310.pyc
│           │   ├── x1_arduino_bridge.cpython-310.pyc
│           │   └── x1_safety_monitor.cpython-310.pyc
│           ├── x1_arduino_bridge.py
│           └── x1_safety_monitor.py
└── x1_description
    ├── CMakeLists.txt
    ├── include
    │   └── x1_description
    ├── package.xml
    ├── src
    └── urdf
        └── x1_robot.urdf.xacro

23 directories, 31 files
```

The **/home/ubuntu/my_x1_ws/src** directory is here: https://tinyurl.com/ytvvpdat

4.5 Laboratory testing and prototype validation

4.5.1 Final prototype integration and system overview

To provide an overview of the final result of this work, all the hardware and software implementations described in the previous sections are part of a fully integrated smart agricultural cart prototype, designed as a flexible experimental platform for precision farming applications, with a particular focus on supporting the distribution of agrochemicals and harvesting assistance. From a systemic point of view, the prototype is not a single device, but a distributed robotic architecture, in which mechanical design, integrated electronics, sensing, communication and software intelligence are closely interconnected. The resulting platform embodies the concept of a mobile, modular, sensor-driven smart cart capable of functioning as a supervised robotic assistant in agricultural environments.

The prototype integrates the following main subsystems:

A. Mechanical structure and mobility

The cart is built around a modular aluminum profile frame composed by struts, chosen for its mechanical robustness, reconfigurability, and ease of expansion. The structure supports the installation of electronic enclosures, sensors, power units, and future payloads (e.g., sprayers or harvesting tools). Mobility is provided by a differential-drive propulsion system, comprising two powered wheels and auxiliary caster wheels, enabling precise low-speed motion and maneuverability in constrained agricultural settings such as crop rows or greenhouse environments. This mechanical configuration forms the physical basis for controlled motion and future autonomous navigation.

B. Embedded control and propulsion management

The motor actuation is handled by a dedicated Motor Control Interface (MCI) based on an Arduino Mega 2560, coupled with industrial-grade motor drivers and closed-loop motion controllers. Encoder feedback ensures accurate speed and displacement control, while regenerative braking management enhances electrical efficiency and system safety. This is the basic layer of control.

On top of this layer, a ROS 2-based motion management node (RP3) executes runtime coordination of velocity commands, sensor feedback, and safety constraints, clearly separating hardware enablement from operational control logic.

C. Distributed computing and ROS 2 architecture

A key feature of the prototype is its distributed computing architecture, based on multiple Raspberry Pi units (RP1–RP5), each assigned a specific functional role:

- RP1: centralized telemetry collection and aggregation;
- RP2: multispectral camera management and image processing;
- RP3: motion management and actuation control;
- RP4: 2D Lidar mapping management;
- RP5: joystick-based teleoperation interface.

All units communicate through a ROS 2 network, enabling standardized message exchange, modularity, and scalability. This architecture allows each subsystem to operate independently while remaining fully integrated within a coherent system.

D. Sensing and perception capabilities

The prototype incorporates a heterogeneous set of sensors, enabling both environmental awareness and crop-level perception:

- A multispectral camera system, managed through a hybrid Web Server–ROS 2 pipeline, supports distance-aware image alignment and vegetation index computation (e.g., NDVI, MCARI).
- Environmental sensors provide real-time measurements of temperature, humidity, atmospheric pressure, and CO_2 concentration.
- Positioning and orientation sensors (GPS, IMU, AHRS, LiDAR) supply the data required for motion supervision and future localization and navigation algorithms.

Together, these sensing modalities transform the cart from a simple mobile platform into a data-driven perception system.

E. Human–machine interaction and supervision

To ensure safe operation and usability during experimental activities, the prototype includes multiple levels of human–machine interaction: 1) a joystick-based teleoperation interface (RP5) allows manual control of the robot during setup, testing, and supervised operation. 2) a set of LCDs connected to the Overall Master Control (OMC) provides local, human-readable feedback on system status, device availability, power conditions, and safety flags and 3) Foxglove-based ROS 2 visualization enabling remote, network-level monitoring of telemetry, sensor data, and image streams. This multi-layered supervision approach ensures transparency, robustness, and operator confidence during system operation.

The platform validates a system integration strategy, in which sensing, motion, communication, and supervision coexist within a unified architecture. The final prototype demonstrates that it is possible to build a practical, modular, and extensible smart cart by combining accessible hardware components with a modern ROS 2 software framework.

This integration is visually summarized in Figure 4.39 below.

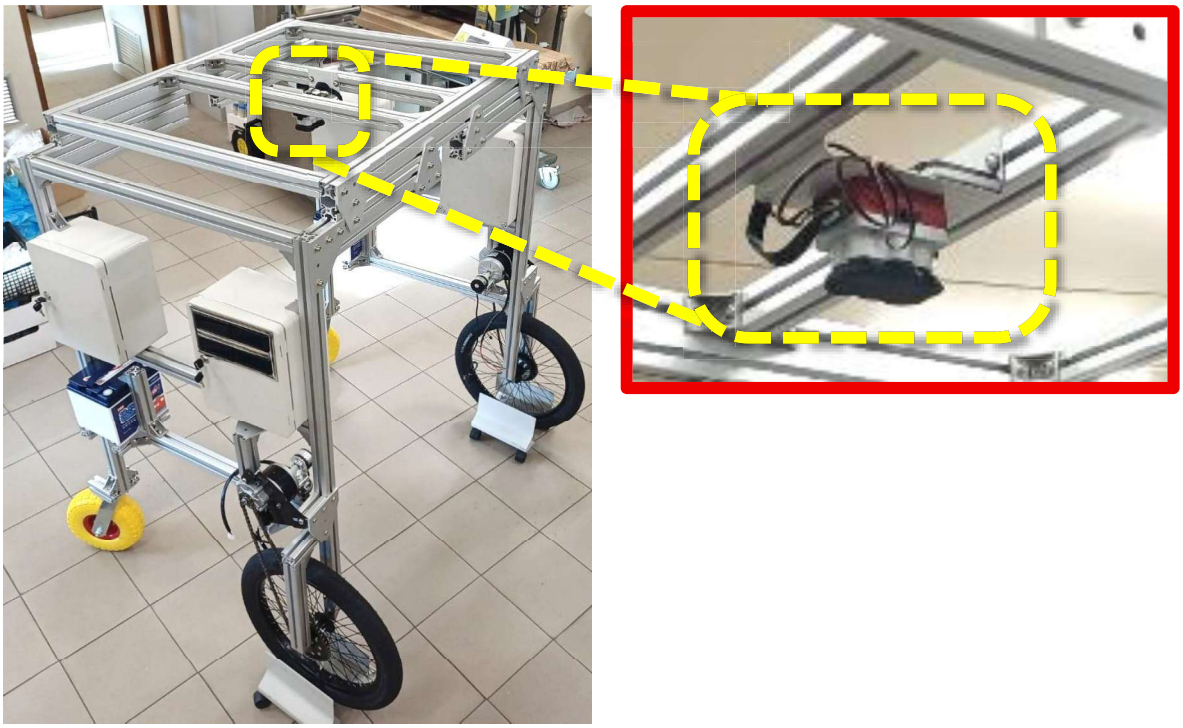


Figure 4.39. Final prototype of the smart agricultural cart developed in this work. The platform integrates the propulsion system, embedded control units, sensor interfaces, and modular mechanical structure within a ROS 2–based distributed architecture.

4.5.2 Simultaneous Localization and Mapping (SLAM) in agriculture: main application in greenhouse environment

Simultaneous Localization and Mapping (SLAM) is a key technology in mobile robotics, enabling robots to localize and map unknown environments in the absence of GPS signal[222] .In precision agriculture, and especially in greenhouse contexts, SLAM has become increasingly important to tackle the unique challenges posed by repetitive structures, visually ambiguous environments, and dynamic obstacles.

Several recent studies have adapted SLAM algorithms to the specific characteristics of greenhouse settings. The European GRAPE project, for instance, explored 2D/3D SLAM variants (such as Gmapping and KartoSLAM), combining LiDAR and stereo cameras for mapping and navigation in real vineyards[223] While these systems show promising performance in structured environments, they still struggle under poor visibility and low-contrast visual conditions [223][224]. A particularly targeted solution is VineSLAM,

developed by *[225]* for vineyard and greenhouse corridors. This method extracts robust geometric features (points and semiplanes) from 3D LiDAR data to reduce drift in long, symmetric vegetative corridors. VineSLAM achieved sub-decimeter localization accuracy, outperforming standard SLAM approaches. However, its application is still limited by the high cost of 3D LiDAR sensors and a lack of validation in horticultural greenhouses. [226] addressed the issue of glass-induced reflections typical of greenhouses by combining 2D and 3D LiDARs. Their system preprocesses 3D point clouds to filter out reflections and project them onto a 2D plane, subsequently processed using Cartographer. The robot maintained average positional deviations under 8 cm and angular errors below 3° at a speed of 0.4 m/s, offering improved computational efficiency compared to full 3D SLAM approaches. Other targeted approaches as showed from [227]that introduced the AF-PCP algorithm, optimized for greenhouses with hanging plants and low foliage density. The system expanded the mapped area by 155% compared to Google Cartographer. Also, [228] enhanced LIO-SAM-based SLAM with YOLOv8 networks to detect rail paths in multi-span greenhouses. On a rail-guided robot, the system achieved lateral deviation of ~4.1 cm, angular deviation of ~1.8°, and improved rail-switching efficiency by 25% over traditional methods. However, it requires substantial computational resources.

Visual-inertial odometry-based SLAM systems have also shown strong potential; [229] proposed a stereo system equipped with image enhancement techniques to recover ORB and LSD features in low-light and foggy agricultural environments. Validated on standard datasets (KITTI and EuRoC), it outperformed VINS-SLAM, PL-SLAM, and ORB-SLAM2 in both localization and mapping accuracy. Its ability to function under degraded visual conditions makes it promising for poorly lit greenhouses, though it remains sensitive to strong texture variation and vegetative occlusion. [222] introduced a multi-sensor SLAM system combining Realsense T265 (VIO), IMU, and wheel odometry with an Extended Kalman Filter (EKF), using a modified ORB-SLAM2 for real-time 3D mapping. In real greenhouses (closed loops, intersections, soilless cultivation, metallic tunnels), it drastically reduced localization error—e.g., absolute error dropped from ~1.0 m to ~0.25 m at 0.4 m/s; RMSE as low as 0.023 m in tight curves versus 0.564 m using VIO alone. A dense plant model was generated in real time using CPU, although some visual artifacts remained. Strengths include high robustness and multi-sensor accuracy; limitations involve the need for calibrated odometry and occasional holes in the dense maps. □

According to the reviewed studies, current research trends strongly emphasize the use of 2D/3D LiDAR and multi-sensor fusion in greenhouse environments due to limited

GPS coverage and spatial constraints[226]. As example, **[230]** released the GREENBOT dataset, specifically developed for agricultural robotics in Mediterranean greenhouses. Using a mobile platform equipped with a stereo camera (Bumblebee), two 3D LiDAR sensors (Velodyne VLP-16 and Ouster OS0), a high-precision GNSS receiver (NovAtel), and an IMU, the authors collected real-world data on perception and navigation in a tomato greenhouse. This dataset enables the validation of 3D SLAM algorithms under realistic trajectories, providing detailed crop maps.

As highlighted in this brief overview, SLAM applications in greenhouse scenarios reveal that the integration of heterogeneous sensors (3D LiDAR, stereo cameras, IMU, and odometry) is crucial to enhance robustness and localization accuracy. Despite several successful implementations [222,225,226], which demonstrate that advanced data fusion techniques can mitigate typical greenhouse challenges, such as glass reflections, symmetrical vegetation layouts, and texture-poor environments, important limitations persist. These include the high computational demands of 3D SLAM methods, the need for precise and expensive sensors, and the difficulty in generalizing models to unstructured or highly variable agricultural environments. Therefore, while SLAM-based approaches provide a solid foundation for the advancement of autonomous agricultural robotics, further optimization is necessary to ensure operational reliability in real-world, dynamic, and low-supervision scenarios.

4.5.2.1 SLAM in MATLAB

MATLAB offers a powerful ecosystem of toolboxes designed for the simulation, development, and deployment of SLAM (Simultaneous Localization and Mapping) and robotic navigation algorithms. Among these, the Navigation Toolbox provides an integrated suite for sensor modeling, localization, mapping, and path planning, making it particularly suitable for rapid prototyping in agricultural robotics[231]

The toolbox enables realistic simulation of sensor data—including IMU, GPS, and wheel encoders—by incorporating noise models and calibration routines. It supports sensor fusion through filters such as EKF, and allows generation of both 2D occupancy grids and 3D voxel maps. Additionally, planners such as A*, Hybrid A*, RRT, and RRT* are available to compute navigation paths, while the SLAM Map Builder App streamlines map construction and visualization tasks[232].

For example, with the `lidarSLAM` object, users can add laser scans via the `addScan` function to incrementally build a pose graph, apply loop closure and scan matching, and finally generate an occupancy map using `buildMap`. These capabilities allow for both online SLAM and offline map optimization [233].

The Robotics System Toolbox further enhances the modeling of robotic platforms by providing detailed kinematics and dynamics via the `rigidBodyTree` class, importing URDF/SDF robot descriptions, and supporting sensor models like lidarScan. It also includes SLAM-relevant functions such as `buildMap` and `slamLoopClosure`, and supports realistic robot motion spaces like `stateSpaceDubins` for non-holonomic vehicles [232].

The SLAM workflow in MATLAB typically involves:

1. **Mapping**: Creating or updating a map from LiDAR scans. For instance, `lidarSLAM` builds a pose graph, which can be converted into a 2D occupancy map using `buildMap`. MATLAB also supports layered 3D mapping.
2. **Localization**: Estimating robot pose through scan matching or Monte Carlo Localization (e.g., `monteCarloLocalization`), using fused data from odometry and sensors like IMU or LiDAR.
3. **Path Planning and Navigation**: Given a mapped environment, trajectory planning is executed using state spaces (e.g., Dubins) and planners such as `plannerRRT*` or `plannerAStar`, facilitating robot motion in constrained agricultural layouts

MATLAB represents a highly valuable resource for the development and testing of SLAM algorithms applied to protected agriculture, owing to its ability to integrate sensor modeling, data fusion filters, visualization, and advanced interactive tools within a single environment. One of its main strengths lies in rapid prototyping: MATLAB allows for the relatively straightforward simulation of complex scenarios and facilitates the deployment of developed models to real platforms through MATLAB Coder and Simulink Coder, which enable automatic code generation compatible with embedded hardware and physical robots. Additionally, graphical interfaces such as the SLAM Map Builder, Scope Viewer, and RViz-like tools greatly simplify debugging and enhance the understanding of system dynamics during development.

However, the MATLAB environment presents several critical limitations when transitioning from simulated contexts to real-world applications. Simulations often simplify the inherent complexity of agricultural environments such as greenhouses or cultivation tunnels, where conditions like glass reflections, condensation, fog, or variable lighting are difficult to replicate accurately. Furthermore, being a high-level interpreted language, MATLAB can be computationally demanding: maintaining real-time responsiveness frequently requires down sampling data or reducing map resolution, which may compromise localization accuracy and map fidelity.

In addition, although MATLAB supports the generation of ROS nodes and C++ code, full portability to physical systems often requires manual interventions, including sensor calibration, data synchronization, and communication error handling factors that can undermine the overall robustness of the final system [234].

4.5.3 Proof of concept of the implementation of the algorithm of Simultaneous Localization And Mapping (SLAM) with Lidar Scans using a ROS2 prototype small robot (TurtleBot3 Burger)

In order to validate, under realistic conditions, the software solutions developed for the modular ABOTx1 platform, a small-scale proof-of-concept experiment was conducted in a controlled laboratory environment. The objective of this experiment was to test the distribution and interoperability of the software modules designed for autonomous navigation by deploying them on the TurtleBot3 Burger (identified as RP8 by the Universal Kernel Image 1.15, codename Lazarus), a compact and accessible ROS2-based platform widely used in academic settings for SLAM applications and robotic control (Figure 4.40). During the test, the main functionalities of the ABOTx1 system were simulated using the TurtleBot3 Burger and its topics (i.e. /scan, /odom, /imu, /cmd_vel). The RP8 module was responsible for managing locomotion, specifically the control of the two DYNAMIXEL motors via the OpenCR board, which integrates an ARM Cortex-M7 microcontroller dedicated to real-time command execution. The RP8 also handled the acquisition and management of data from the LDS-02 LiDAR sensor, capable of generating 360° scans of the surrounding environment, which are essential for localization and map building. Finally, the RP5 module enabled manual teleoperation of the robot via joystick, facilitating environment exploration prior to autonomous mapping.

Communication between the TurtleBot and the control workstation took place over a distributed ROS2 network for real-time monitoring and data logging. The TurtleBot3, running UKI (codename Lazarus), acted as a ROS node, synchronously sending and receiving control, sensor, and actuation data.

The TurtleBot3 Burger proved to be an ideal platform for this validation thanks to its compact yet complete architecture. The robot features a Raspberry Pi as its onboard computer, an 11.1V 1800 mAh Li-Po battery rechargeable via a 12V/5A adapter, and an integrated 9-DOF IMU combining a gyroscope, accelerometer, and magnetometer. It is also equipped with multiple interface ports (GPIO and Arduino-compatible), facilitating integration with external modules.

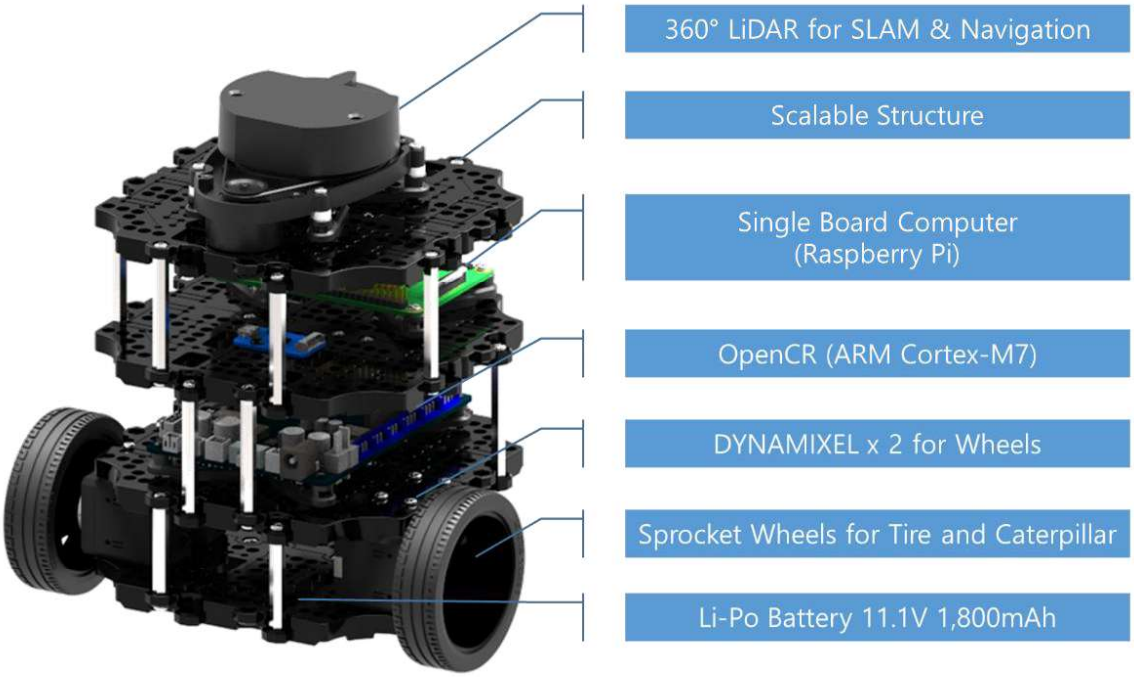


Figure 4.40. Modular structure of the TurtleBot3 Burger(source:https://turtlebot-3-blockly-wiki.readthedocs.io/en/latest/about.html).

4.5.3.1 Experimental Layout

The experimental layout (Figure 4.41) was designed to simulate a miniature greenhouse environment under controlled laboratory conditions. The test area was enclosed using non-woven fabric (TNT) panels supported by vertical rods, while seven vertical poles were positioned at the center of the scene with an inter-distance of approximately 30 cm. This configuration was intentionally selected to reproduce structurally symmetric geometries and reduced-visibility conditions, which are commonly encountered in cultivated

greenhouse environments and represent a challenging scenario for robotic perception and localization

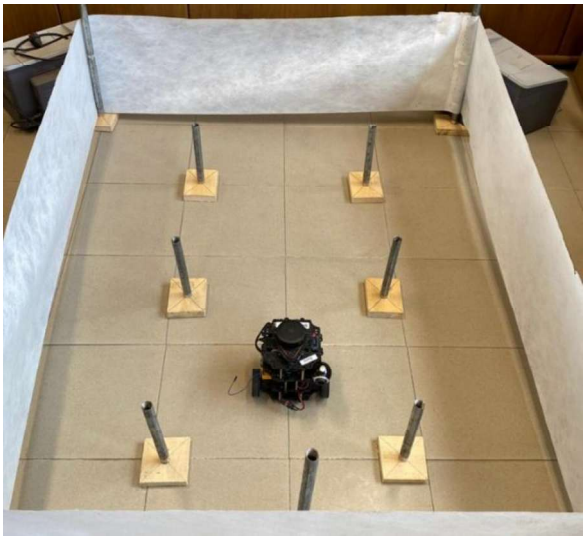


Figure 4.41. Simulated environment layout.

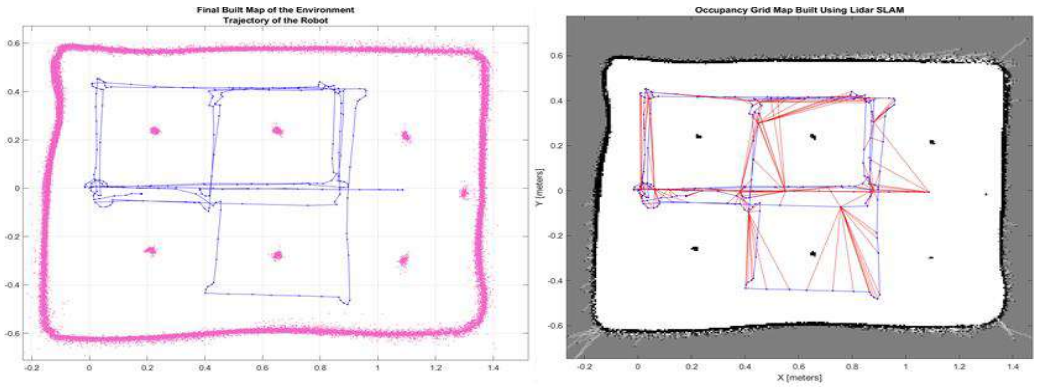
During the experiment, the robot trajectory was teleoperated and manually controlled through a joystick interface (RP5), while LiDAR data were continuously acquired in real time by the onboard sensor module (RP8). The collected scans were subsequently processed offline in MATLAB using the SLAM toolbox in order to reconstruct the environment map and estimate the robot trajectory. This experimental strategy allowed the validation of the perception and mapping components independently from full autonomous navigation, thereby isolating the performance of the SLAM pipeline. Therefore, as a preliminary step toward the implementation of an Autonomous Navigation for Mobile Robots (ANMR) system on the smart agricultural cart under development, a proof-of-concept SLAM experiment was conducted using a small-scale robotic platform (TurtleBot3 Burger). The test successfully produced both a global environment map and the corresponding occupancy grid map of the delimited experimental area, confirming the feasibility of LiDAR-based SLAM under greenhouse-like conditions. The computational effort required for the map reconstruction was found to be strongly dependent on the selected map resolution and precision parameters. On average, a processing time ranging from approximately 1 to 3 minutes per square meter was required to obtain an acceptable map, highlighting the trade-off between spatial accuracy and computational load. These aspects are particularly relevant when considering real-time deployment on embedded platforms. The script to make the

SLAM using the Lidar scans is reported below. A short video of the teleoperated TurtleBot3 Burger is here: https://sites.google.com/unibas.it/smartagri/a-bot-x1#h.z8z0azdqb2ml.

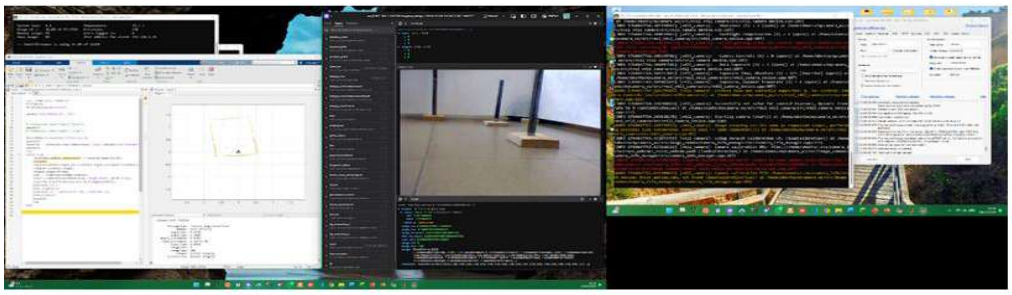
```
load('slam.mat');
scans=data.scan;
odom=data.odom;

maxLidarRange = 3;
mapResolution = 150;

slamAlg = lidarSLAM(mapResolution, maxLidarRange);

slamAlg.LoopClosureThreshold = 100;
slamAlg.LoopClosureSearchRadius = 2;

for i=1:10
    ascan=scans{i};
    ascan.Ranges=ascan.ranges;
    ascan.Angles= [ascan.angle_min:ascan.angle_increment:ascan.angle_max]';
    [isScanAccepted, loopClosureInfo, optimizationInfo] = addScan(slamAlg, ascan);
    if isScanAccepted
        fprintf('Added scan %d \n', i);
    end
end
show(slamAlg);
title({'Map of the Environment','Pose Graph for Initial 10 Scans'});
drawnow

firstTimeLCDetected = false;

hf=figure;
hf.WindowStyle="docked";
for i=10:length(scans)
    i
    drawnow
    ascan=scans{i};
    ascan.Ranges=ascan.ranges;
    ascan.Angles= ascan.angle_min:ascan.angle_increment:ascan.angle_max;
    [isScanAccepted, loopClosureInfo, optimizationInfo] = addScan(slamAlg, ascan);
    if ~isScanAccepted
        continue;
    end
    % visualize the first detected loop closure, if you want to see the
    % complete map building process, remove the if condition below
    if optimizationInfo.IsPerformed && ~firstTimeLCDetected
        show(slamAlg, 'Poses', 'off');
        hold on;
        show(slamAlg.PoseGraph);
        hold off;
        firstTimeLCDetected = true;
        drawnow
    end
end
title('First loop closure');
drawnow

hf=figure;
hf.WindowStyle="docked";
show(slamAlg);
title({'Final Built Map of the Environment', 'Trajectory of the Robot'});
drawnow

[scans, optimizedPoses]  = scansAndPoses(slamAlg);
map = buildMap(scans, optimizedPoses, mapResolution, maxLidarRange);
```

```
hf=figure;
hf.WindowStyle="docked";
show(map);
hold on
show(slamAlg.PoseGraph, 'IDs', 'off');
hold off
title('Occupancy Grid Map Built Using Lidar SLAM');
drawnow
```

Figure 4.42 shows the results of the LiDAR-based Simultaneous Localization and Mapping (SLAM) experiment conducted as a proof of concept for the implementation of Autonomous Navigation for Mobile Robots (ANMR). The final reconstructed environment map, together with the estimated trajectory of the robot during the exploration phase (Figure 4.42a), highlights the continuity and consistency of the trajectory, indicating stable localization performance despite the presence of structurally symmetric elements and limited visual features in the simulated greenhouse environment. Meanwhile, the corresponding occupancy grid map generated from the optimized LiDAR scans (Figure 4.42b) provides a discretized representation of free and occupied areas and constitutes a fundamental input for subsequent navigation tasks, such as path planning, obstacle avoidance, and pose initialization.



(a)



(b)

Figure 4.42. Teleoperated LiDAR-based SLAM results obtained as a proof of concept for autonomous navigation: (a) final reconstructed environment map with estimated robot trajectory; (b) occupancy grid map generated from optimized LiDAR scans.

In addition to the SLAM implementation, a dedicated MATLAB class (https://it.mathworks.com/matlabcentral/fileexchange/182934-manage-a-real-turtlebot3-burger-using-ros2-humble) was developed to enable full management of the TurtleBot3 Burger through the ROS 2 interface. This control framework supports robot initialization, odometry reset, velocity command execution, and sensor data handling, and was specifically designed with the explicit objective of being transferable to the full-scale ABOTx1 platform.

A custom scouting and mapping script, coupled with a graphical dashboard, was implemented to support autonomous scouting exploration using a boundary patrol like algorithm, real-time visualization, collision detection, and supervised motion control during mapping operations.

To this aim the dedicated MATLAB class was further improved with the calculation of the front (-2° to +2°), left (+2° to +30°) and right (-2° to -30°) obstacle distance using the lidar scan data. The minimum collision distance was set to 0.45 m. The forward speed and rotation speed were set to 0.15 m/s and 0.15 rad/s respectively.

Moreover, a simulated TurtleBot3 Burger was implemented acting precisely as its real counterpart. An environment test map was used using 20 cells/meter resolution.

The core of the autonomous scout & mapping Matlab script is listed below.

```
% SCOUT & MAP
% ********* MAIN LOOP
while true
if stop, break, end
    go_test_3_tb3_show_scout2map % display info
% CHECK IF THE LIDAR SCAN DETECTED A COLLISION
    collision_detected= 0;
    if min([ tb.dist_now,...
             tb.collision_dist_now]) <= dist_collision
        collision_detected= 1;
    end
    % CHECK IF MAXIMUM TRAVEL REACHED
    if tb.planpathdist_now >= max_travel_dist
        collision_detected = 1;
    end
% ********* DRIVING SECTION
    if collision_detected == 0
    % GO FORWARD
        msg= 'GO FORWARD';
        tb.set_cmd_vel_xzvel(forw_speed,0);
    else
        msg= 'F-STOP';
        tb.set_cmd_vel_stop;
        drawnow
        % pause(1)
    % TURN BASED ON OBSTACLE AVOIDANCE
        if tb.collision_left_now>tb.collision_right_now
            k=1; % TURN LEFT
        else
            k=-1; % TURN RIGHT
        end
        % ********* INNER ROTATION LOOP
        while min([ tb.dist_now,...
```

```
                    tb.collision_dist_now]) <= dist_collision
        if stop, break, end
            go_test_3_tb3_show_scout2map % display info
            msg= 'TURN';
            tb.set_cmd_vel_xzvel(0,rotz_speed*k);
        end % ********* INNER ROTATION LOOP
        msg= 'T-STOP';
        tb.set_cmd_vel_stop;
        drawnow
        % pause(1)
    end
end % ********* MAIN LOOP
```

The main idea is the use of the left and right sensed distance to turn the robot with the aim to avoid the detected obstacle. This allows the robot to perform like a boundary patrol, in addition, the lidar scans are collected and the occupancy map is built in real-time. A short video of the autonomous simulated TurtleBot3 Burger is here: https://sites.google.com/unibas.it/smartagri/a-bot-x1#h.c5y4l6exfwen. As shown in Figure 4.43, the dashboard provides real-time visualization of the robot pose, LiDAR scans, distance to obstacles, battery status, and navigation messages, while enabling manual control of the script through start, stop, and refresh commands.

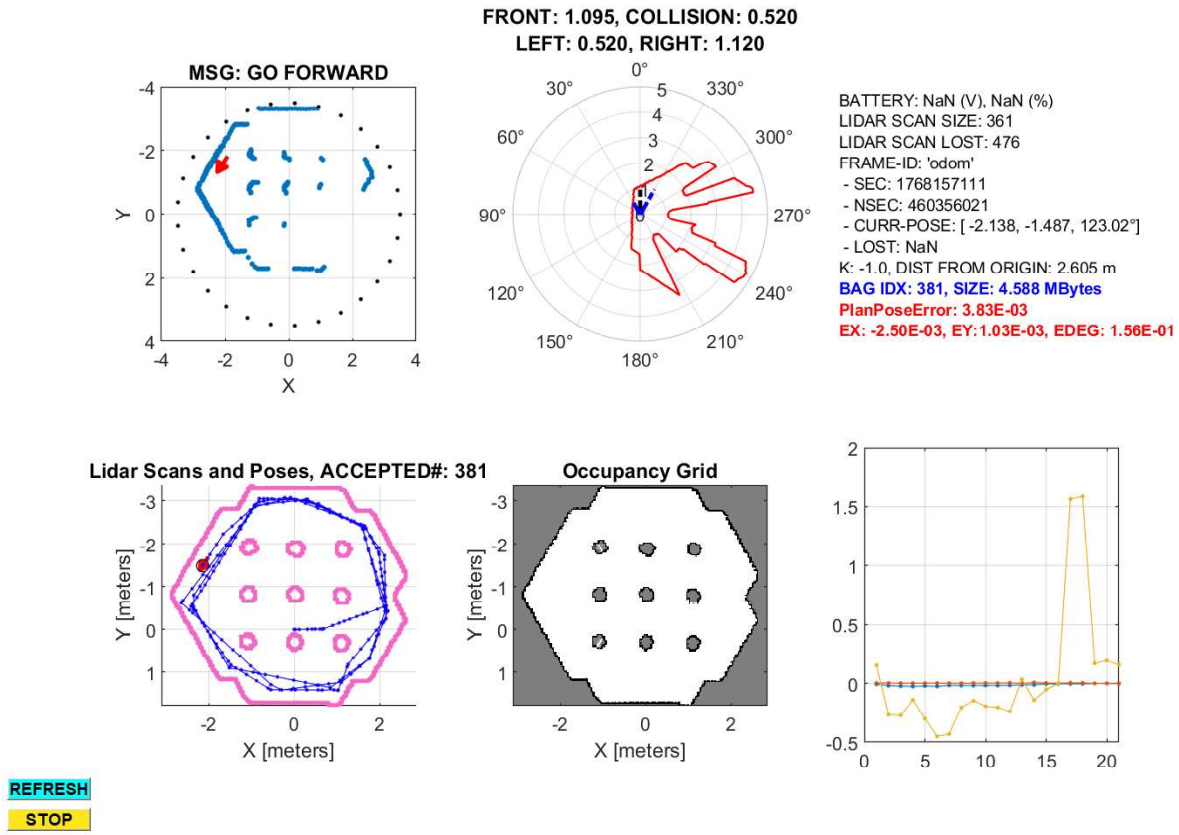


Figure 4.43. MATLAB-based dashboard for scouting, mapping, and supervisory control of the TurtleBot3 Burger.

Based on the maps generated by the SLAM algorithm, the robot pose can be initialized with respect to the reconstructed environment, enabling the execution of motion sequences defined by a series of start-and-stop vector points based on the environment map. Between two consecutive motion commands, the robot can perform additional timed operations, such as acquiring multispectral images and analyzing crop health indices. This operational logic represents the basis for future integration of perception-driven decision-making and targeted actuation, such as site-specific agrochemical spraying using a robotic arm (needed to be implemented).

Finally, the mechanical assembly of the ABOTx1 robotic platform has been completed and fully documented, while the electrical system integration remains to be finalized and set up. The results presented in this section, therefore, represent a validated software and algorithmic foundation for the subsequent implementation of autonomous navigation and perception–action loops on the full-scale agricultural robot.

CHAPTER 5

5 NIR (Near-Infrared) models for harvest assistance

5.1 Introduction: role of Near-Infrared (NIR) sensors on quality and pest management in harvest and post-harvest.

The globalization of agricultural trade has significantly increased the movement of fresh horticultural products across geographic and political boundaries, intensifying the risk of spreading plant pests and diseases and posing serious threats to food security worldwide. In this context, the presence of insect infestation and internal quality deterioration in fruits represents a major challenge for postharvest management, storage, processing, and international trade. Border control procedures and zero-tolerance regulations for quarantine pests have therefore been strengthened, placing increasing pressure on producers and supply chains to ensure high-quality, pest-free products before shipment [235]. Among invasive insect pests, Drosophila suzukii (Matsumura), commonly known as spotted wing drosophila (SWD), has emerged as one of the most economically damaging species affecting berries and stone fruits in Europe and the Americas since the late 2000s [236] .Its ability to oviposit in intact, ripening fruits using a serrated ovipositor makes crops such as sweet cherries particularly vulnerable [237,238][239,240]. Infestation often remains hidden during the early stages, as larvae develop inside the fruit without visible external symptoms, leading to rapid postharvest decay, secondary microbial infections, and severe economic losses [241,242][243]. The difficulty of detecting early infestation stages using conventional visual inspection or destructive methods represents a critical bottleneck in packing houses, storage facilities, and export chains [244,245][246].

At the same time, fruit quality management in climacteric species such as kiwifruit (Actinidia chinensis) poses a different but equally complex challenge. Kiwifruits are typically harvested unripe and stored under controlled conditions for extended periods, during which ethylene-mediated ripening processes and preharvest factors contribute to large variability in quality attributes such as soluble solids content (SSC) and flesh firmness (FF) [247] [248]. Accurate assessment of the ripening stage is essential to optimize harvest timing, storage allocation, and market destination. However, traditional quality evaluation

methods are destructive, time-consuming, and unsuitable for large-scale or continuous monitoring [249].

In response to these challenges, near-infrared (NIR) spectroscopy has gained widespread attention as a rapid, non-destructive, and environmentally friendly analytical technique for both quality assessment and pest detection in horticultural products. NIR spectroscopy is based on the interaction between electromagnetic radiation and molecular vibrations of organic compounds, producing characteristic absorption and reflectance patterns related to chemical composition and structural properties [250][251] Commercial spectrometers operating in the Vis/SWNIR, NIR, or Vis/NIR ranges have been extensively applied for qualitative and quantitative analyses in agriculture, requiring minimal or no sample preparation and enabling on-line or in-line implementation [252–258]

Over the past decades, NIR-based techniques have proven effective for assessing internal fruit quality attributes, including SSC, dry matter, and firmness, in a wide range of fruits such as apples, citrus, berries, and kiwifruit [259][260][261–264][265], In kiwifruit specifically, Partial Least Squares Regression (PLSR) models have consistently demonstrated high predictive accuracy for SSC and FF across different cultivars and measurement configurations [266][267][268]More recently, the integration of portable NIR sensors with advanced machine learning approaches, including Support Vector Machines (SVM), artificial neural networks (ANN), and convolutional neural networks (CNN), has further improved prediction and classification performance, particularly under postharvest storage conditions where nonlinear quality dynamics become more pronounced [269][270]. Parallel to quality assessment, NIR spectroscopy has also been explored for the non-destructive detection of internal insect infestation in fruits such as cherries, blueberries, mangoes, citrus, and pomegranate [271][272]. These studies demonstrated that spectral changes associated with insect feeding, tissue degradation, and metabolic alterations can be captured and exploited using chemometric and pattern recognition techniques, including PLS-based classifiers, linear discriminant analysis (LDA), and genetic algorithms for wavelength selection. However, detection performance has been shown to depend strongly on fruit species, ripening stage, infestation severity, insect developmental stage, and acquisition conditions [273].

A common limitation in spectroscopic modeling lies in the difficulty of statistically correlating high-dimensional spectral data with the property of interest while providing reliable estimates of prediction uncertainty on unknown samples [274][275–277]. To address this issue, robust chemometric strategies combining cross-validation, external

validation, wavelength selection, and Monte Carlo-based randomization have been proposed. In particular, the coefficient of variation algorithm (CVA) has proven effective in identifying and removing unstable or uninformative wavelengths, thereby improving model robustness and interpretability [278][279–281]. The increasing availability of high-performance computing resources has further enabled the extensive use of Monte Carlo approaches to assess model stability under repeated random sampling conditions [282,283].

Within this framework, the present chapter addresses two complementary case studies that exemplify the potential of portable NIR spectroscopy combined with advanced chemometric and machine learning approaches for harvest and postharvest management in fruit supply chains.

The first case study focuses on the non-destructive early detection of Drosophila suzukii egg infestation in intact sweet cherries, with the objective of supporting sorting, grading, and quarantine decision-making during postharvest handling. The second case study investigates the prediction of key quality attributes, namely soluble solids content (SSC) and flesh firmness (FF), as well as the classification of ripening stages in kiwifruit, by integrating spectral and physicochemical data collected from the pre-harvest phase through short- and medium-term cold storage. These case studies provide experimentally validated examples of how portable NIR spectroscopy, when coupled with advanced data-driven modelling techniques, can be effectively applied as a decision-support tool to enhance fruit quality management, reduce postharvest losses along the supply chain, and improve consumer safety. The corresponding methodology and results have been reported in:

Altieri, G., Avaei, M. R., Matera, A., Genovese, F., Verrastro, V., Admane, N., Mammadov, O., Laveglia, S., & Di Renzo, G. C. (2025). *Non-Destructive Early Detection of Drosophila suzukii Infestation in Sweet Cherries (cv. Sweet Heart) Based on Innovative Management of Spectrophotometric Multilinear Correlation Models*. **Applied Sciences**, 15(1), 197. https://doi.org/10.3390/app15010197.

Altieri, G., Laveglia, S., Rashvand, M., Genovese, F., Matera, A., Mininni, A. N., Calabritto, M., & Di Renzo, G. C. (2025). *Portable NIR Spectroscopy Combined with Machine Learning for Kiwi Ripeness Classification: An Approach to Precision Farming*. **Applied Sciences**, 15(11), 6233. https://doi.org/10.3390/app15116233.

5.2 Case study 3: Prediction of quality parameters and maturity classification in kiwifruit

5.2.1 Materials & Methods

5.2.1.1 Sample preparation, NIR data acquisition and quality analysis

Kiwifruits (*Actinidia chinensis cv. Zesy002,* commercial name Zespri™ Sungold) were grown in the Metaponto area (Matera Province, Basilicata, Southern Italy). Fruits were harvested in September 2024 at two maturity stages: approximately two weeks before commercial ripeness and at commercial harvest. Immediately after harvesting (day 0), fruits were placed in cold storage at an initial temperature of 10 °C. Throughout the storage period, fruits were subjected to a daily ozone treatment consisting of a 30-minute exposure at a flow rate of 5 ppm/min. The storage temperature was progressively lowered from day 0 at a controlled rate of about 1 °C per day until reaching −0.5 °C on day 10. After this cooling phase, fruits were maintained at −0.5 °C for the remaining storage duration.

Near-infrared (NIR) spectroscopy measurements were carried out using a portable PoliSPEC-NIR spectrophotometer (IT-Photonics S.r.l., Fara Vicentino, Italy), operating in the spectral range of 900–1700 nm with a spectral resolution of 3.2 nm. The instrument is equipped with a halogen light source, an InGaAs detector with a 256-pixel array, and an integrated thermal control system with a front heat sink to reduce signal variability due to temperature fluctuations during measurements. Spectral data were acquired at three different sampling times: 10 days before harvest (pre-harvest, PH), at harvest (day 0, D0), and after 60 days of refrigerated storage (day 60, D60). Overall, 200 fruit samples were analyzed. For each fruit, two spectra were collected on opposite equatorial sides, and the resulting spectra were averaged to obtain a representative profile for each sample.

At each sampling time, quality attributes were also assessed. Soluble solids content (SSC) was measured using a portable digital refractometer (Atago ATC-1E, Japan), with results expressed in °Brix. Fruit firmness (FF) was determined using a texture analyzer (Instron Model 3343, Instron Co., MA, USA) fitted with a flat-ended cylindrical stainless-steel probe (8 mm diameter). Penetration tests were performed at a constant speed of 4 mm min^{-1}, and firmness was expressed as the force (N) required to penetrate the peeled fruit flesh to a depth corresponding to 20% of the equatorial fruit height.

5.2.1.2 Data analysis

Prediction task:

Predictive models for soluble solids content (SSC) and fruit firmness were developed using Partial Least Squares Regression (PLSR), a multivariate statistical approach widely applied to relate spectral information to physicochemical quality attributes. To improve model accuracy and enhance the signal-to-noise ratio, several spectral preprocessing techniques were applied prior to model development. Multiplicative Scatter Correction (MSC) and Standard Normal Variate (SNV) were used to compensate for scattering effects caused by surface irregularities and sample heterogeneity. n addition, Savitzky–Golay smoothing and derivative transformations were applied to calculate first- and second-order derivatives (SG1 and SG2), aiming to reduce high-frequency noise while preserving relevant spectral features. Dimensionality reduction was further performed using Principal Component Analysis (PCA), retaining components accounting for at least 95% of the total spectral variance. Model optimization was achieved through an iterative procedure involving the identification and removal of outliers and non-informative wavelengths. Outliers were detected based on cross-validation residuals, normalized using z-scores, and excluded when associated with anomalous deviations from the overall distribution. Relevant spectral regions were selected by examining PLSR loading coefficients, which were normalized with respect to their standard deviation and the inherent variability of each wavelength. PLSR model performance was evaluated using 80% of the dataset for calibration, followed by k-fold cross-validation. Model accuracy and robustness were assessed using the coefficient of determination (R^2), the root mean square error of calibration (RMSE), and the residual predictive deviation (RPD) (see the equations in section 3.2.6)

Classification task:

Several supervised classification approaches were implemented to discriminate kiwifruit samples according to ripening stage using spectral information. The selected methods included Linear Discriminant Analysis (LDA), Decision Tree–based classifiers (DT), Artificial Neural Networks (ANN), and Support Vector Machines (SVM), allowing a comparative evaluation of linear and non-linear modeling strategies.

Linear Discriminant Analysis was applied to enhance class separation by projecting the original n-dimensional feature space onto a reduced m-dimensional space that maximizes between-class variance while minimizing within-class variance. This method models class membership as a linear combination of predictor variables and is closely related to variance analysis and classical classification techniques.

Decision Tree–based models were employed due to their robustness to noisy data and ability to handle discrete-valued features. Predictions are generated through a hierarchical set of decision rules derived from the statistical properties of the training data. In this study, three DT algorithms were evaluated: J48, an enhanced version of the traditional decision tree aimed at reducing classification error; CART, which recursively partitions the feature space into homogeneous subregions; and Logistic Model Tree (LMT), a hybrid approach combining decision tree structure with logistic regression at the terminal nodes.

Artificial Neural Networks were developed to capture complex and non-linear relationships between spectral inputs and ripening classes. The network architecture and learning parameters were optimized by defining the number of layers, neurons, transfer functions, and training algorithm. A supervised learning strategy based on backpropagation was adopted, and the Levenberg–Marquardt optimization algorithm was used to enhance training efficiency. To balance predictive performance and computational complexity while limiting overfitting, the ANN consisted of two hidden layers with 20 neurons. The dataset was divided into training, validation, and test subsets, and a 10-fold cross-validation procedure was applied to ensure model robustness.

Support Vector Machines were also implemented as non-parametric classifiers capable of mapping input features into a higher-dimensional space and defining optimal separating hyperplanes through kernel functions. Four kernel types were evaluated: Radial Basis Function (RBF), Polynomial, Gaussian, and Pearson Universal Kernel. Model performance was optimized by tuning the regularization parameter (C), which controls the trade-off between margin width and classification error, and the kernel parameter (γ), which defines the degree of non-linearity in the transformed feature space. Five values of C (0.01, 0.1, 1, 10, and 100) and three values of γ (0.01, 0.1, and 1) were tested through an iterative optimization procedure to identify the most effective hyperplane configuration.

The performance of the classification models was assessed through a comparative analysis using the correlation coefficient (R), mean absolute error (MAE) and root mean squared error (RMSE) which were measured by following equations:

$$R = \sqrt{1 - \frac{\sum_{i=1}^{n}(C_i - C_{ii})^2}{\sum_{i=1}^{n}(C_i - C_m)^2}} \quad (5.1)$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|C_i - C_{ii}| \quad (5.2)$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(C_i - C_{ii})^2} \quad (5.3)$$

where *Ci*, *Cii*, *Cm* and *n* were the discriminated Kiwi, predicted Kiwi, the mean value of the Kiwi samples and the total number of samples, respectively. In addition, out-of sample validation was used to analyze the prediction performance of the models

5.2.2 Results and discussions

5.2.2.1 Firmness and SSC prediction results

The results of the PLSR model under the different configurations analyzed, after the selection of relevant wavelengths and the removal of outliers, are summarized in Table 5.1, following a Monte Carlo simulation with 100 iterations.

The regression models developed for predicting firmness generally demonstrated good predictive performance in cross-validation (CV), with R^2CV values ≥ 0.85. The best results in CV were obtained using the SG2 (R^2CV = 0.880; RPDCV = 2.92) and MSC (R^2CV = 0.872; RPDCV = 2.82) preprocessing methods. These were followed by the Raw and SNV models, which showed similar performances (R^2CV = 0.856), with RPDCV values around 2.65. However, in external prediction, model performance decreased significantly, with R^2P values ranging from 0.562 (SG1) to 0.749 (SNV). The SNV-PLS model achieved the best trade-off between accuracy and robustness (R^2P = 0.749; RMSEP = 12.342; RPDP = 2.02), standing out for its stability across the tested configurations. In contrast, the SG1-PLS model yielded the poorest results (R^2P = 0.562; RMSEP = 16.312; RPDP = 1.53).

Regarding the soluble solids content (SSC), the models showed greater consistency between CV and external prediction. During training, all models achieved R^2CV > 0.96 and RPDCV > 5, with RMSECV values ranging from 0.744 to 0.803 °Brix, indicating strong model fitting and reliable performance during cross-validation.

This trend was also confirmed in external prediction, where the models maintained excellent performance, with $R^2P > 0.91$ and RPDP > 3, confirming their strong predictive reliability. Specifically, the Raw-PLS model achieved the best overall results ($R^2P = 0.935$; RMSEP = 1.142; RPDP = 3.98), closely followed by the SG1-PLS model ($R^2P = 0.934$; RMSEP = 1.151; RPDP = 3.95). These results indicate that SSC strongly correlates with spectral features, independently of the applied preprocessing method.

Table 5.1.PLSR Model performance in Cross-Validation and External Prediction over 100 Monte Carlo runs.

Parameter	Model	R2CV	RMSECV	RPD	Outliers	N° Samples	Selected Wavelength	R2P	RMSEP	RPDP
Firmness	Raw-PLS	0.856 (0.021)	10.451 (0.752)	2.66 (0.18)	8	152	30	0.728 (0.023)	12.854 (0.533)	1.95 (0.08)
	SNV-PLS	**0.856 (0.014)**	**10.501 (0.524)**	**2.65 (0.13)**	**8**	**152**	**16**	**0.749 (0.011)**	**12.342 (0.274)**	**2.02 (0.05)**
	MSC-PLS	0.872 (0.016)	9.795 (0.596)	2.82 (0.17)	8	152	**22**	0.662 (0.013)	14.336 (0.276)	1.74 (0.03)
	SG1-PLS	0.853 (0.020)	10.463 (0.697)	2.64 (0.17)	8	152	23	0.562 (0.022)	16.312 (0.415)	1.53 (0.04)
	SG2-PLS	0.880 (0.015)	9.462 (0.579)	2.92 (0.17)	8	152	25	0.625 (0.014)	15.088 (0.278)	1.66 (0.03)
SSC	**Raw-PLS**	**0.967 (0.004)**	**0.753 (0.042)**	**5.57 (0.31)**	**4**	**156**	**22**	**0.935 (0.003)**	**1.142 (0.022)**	**3.98 (0.08)**
	SNV-PLS	0.964 (0.004)	0.781 (0.046)	5.33 (0.31)	**0**	**160**	**28**	0.918 (0.005)	1.289 (0.039)	3.53 (0.11)
	MSC-PLS	0.965 (0.004)	0.770 (0.048)	5.37 (0.34)	4	156	23	0.929 (0.003)	1.194 (0.027)	3.81 (0.09)
	SG1-PLS	0.968 (0.004)	0.744 (0.042)	5.60 (0.32)	0	160	22	0.934 (0.003)	1.151 (0.022)	3.95 (0.08)
	SG2-PLS	0.962 (0.004)	0.803 (0.042)	5.18 (0.26)	3	157	17	0.931 (0.002)	1.177 (0.017)	3.86 (0.06)

These results are clearly illustrated in Figure 5.1, which shows the predictive performance of the best PLS models for estimating firmness (SNV-PLS) and soluble solids content (SSC, Raw-PLS), after the selection of relevant wavelengths and removal of outliers, along with the relative trend of the RPD metric during the model optimization cycle. Compared to the firmness model, the SSC model shows greater consistency between cross-validation results and external prediction outcomes, suggesting a lower risk of overfitting. This behavior can be partly attributed to the narrower range of the SSC target variable (from 5.2 to 18 °Brix) compared to firmness (from 2.36 to 92.26 N), as well as to variability related to the presence of different maturation stages at the sampling times used in this study. Specifically, two ripening classes (related to the pre-harvest and post-harvest periods) appear to contribute to this variability, highlighting a marked difference in post-harvest (D60) kiwifruits. The more diverse data distribution introduces complexity in establishing consistent predictive relationships, which may reduce the performance of external validation. In contrast, the distribution of SSC values across the three data acquisition periods (pre-harvest, harvest, and post-harvest) is more uniform, resulting in better

generalization and a lower risk of overfitting, together with a likely more linear spectral response to sugar content, as also highlighted by the RPD trend during wavelength selection.

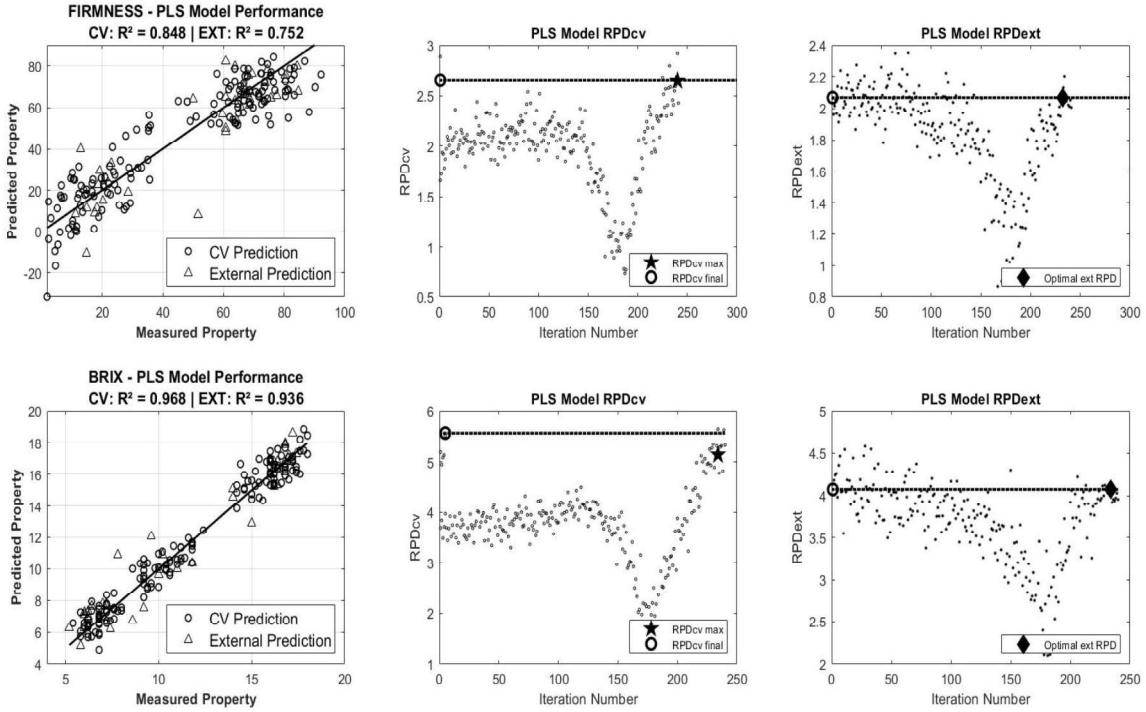


Figure 5.1.Performance of the selected PLSR models for firmness (SNV-PLS) and soluble solids content (RAW-PLS) in relation to cross-validation (CV) and external validation (EXT), after wavelength selection and outlier removal, along with the trend of RPD values in cross-validation (RPDcv) and external validation (RPDext) during the iterative optimization cycle of the models. The star point (★) on the RPDcv curve and the diamond (◆) on the RPDext curve represent the maximum RPD value reached by the model, selected as the best compromise between cross-validation accuracy and prediction. The circle on the Y-axis represents the corresponding value of the optimized model after 100 Monte Carlo runs.

As known, firmness is under the influence of several quality-related attributes of the fruit. According to [269], the spectral information acquired at harvest may not accurately reflect the firmness the fruit will exhibit after the storage period. As a result, obtaining reliable quantitative predictions is particularly challenging. Kiwi firmness is a physical property linked to the internal cellular structure, whose variability can affect both the spectral features and surface scattering. [284] hypothesized that pectin modifications, mainly responsible for changes in firmness, are more difficult to detect compared to other more abundant constituents, such as total soluble solids (TSS). Furthermore, cell turgor, which significantly contributes to perceived firmness, depends on the fruit's water content and the rate of water loss from the outer and inner pericarp tissues [285]. Finally, the intrinsic limitations of NIR, as pointed out by [269], clarify that near-infrared light penetrates only about 2 mm beneath the skin, whereas standard firmness measurements are based on penetration forces at approximately 8 mm depth The results obtained in this study confirm

that the quantitative prediction of firmness (FF) is more complex than estimating the sugar content, as previously highlighted by [269,286]. A comparison with the literature shows that, in the case of firmness, the SNV-PLS model developed achieved moderate predictive performance (R^2_p = 0,749; RMSEP = 12,342 N), which is lower than the performance reported by [286], who achieved R^2_p = 0.78 and RMSEP = 11.96 N using Vis/NIR data, but superior to the study by [269], which reported R^2_p = 0.50 and RMSEP = 4.41 N, despite considering a narrower range of firmness. In comparison, [266] reported even higher performance, obtaining the best predictive model for firmness using raw spectra (R^2_p = 0.90, RMSEP = 28.8 N). Similarly, [267] achieved significant predictive values, with R^2 ranging between 0.82 (RMSE = 14.51 N) and 0.92 (RMSE = 9.87 N).

Regarding the sugar content, the Raw-PLS model developed in the present work (R^2_p = 0,935; RMSEP = 1,142 °Brix) showed results comparable to those obtained by [286], with R^2_p = 0.859 and RMSEP = 1.45 °Brix, and superior to those reported by [287], whose models based on combinations of pretreatments (SG, SNV, MSC) and variable selection (SPA, CARS) achieved R^2 values between 0.22 and 0.69, with RMSE between 1.07 and 1.81 °Brix. Excellent performance was reported by [266], with SNV producing the best predictive model for SSC (R^2_p = 0.993, RMSEP = 0.40 °Brix), while [267] obtained predictive R^2 values ranging from 0.85 (RMSE = 1.10 °Brix) to 0.94 (RMSE = 0.73 °Brix), confirming the generally higher accuracy observed for SSC prediction compared to firmness. An RPD value between 2 and 2.5 is considered a rough but acceptable prediction, while a value above 2.5 represents a good or excellent prediction[288]. Therefore, the model developed for SSC (RPD_p = 3.98) can be considered reliable for practical applications, while the one for firmness (RPD_p = 2.02) requires further optimization to enhance its predictive capacity.

Figure 5.2 shows the trend of the average spectrum of the best models selected for predicting soluble solids content (SSC) and firmness (FF) in kiwi fruits. The curves represent the normalized average reflectance of the considered spectra. The main selected wavelengths (around 958,978, 1085,1107, 1452 and 1465 nm) are strongly consistent with those reported in the literature. Several authors have identified characteristic peaks in kiwi spectra around 974 nm, 1200 nm, 1460 nm, and 1780 nm, confirming their relevance in relation to O-H and C-H bonds, associated with sugar content, water, and the cellular structure of kiwis [266,286,289,290].The first broad peak, centered around 974 nm, is attributed to the combined absorption of water and carbohydrates [284], while [291] identified the bands at 980.39 and 1470.59 nm as corresponding to the first and second overtone of the O-H bond in water, respectively. Additionally, the bands around 1197.60 nm and 1785.71 nm are

related to the overtones of the CH group, suggesting a connection with sugars, cellulose, and cellular water [266].

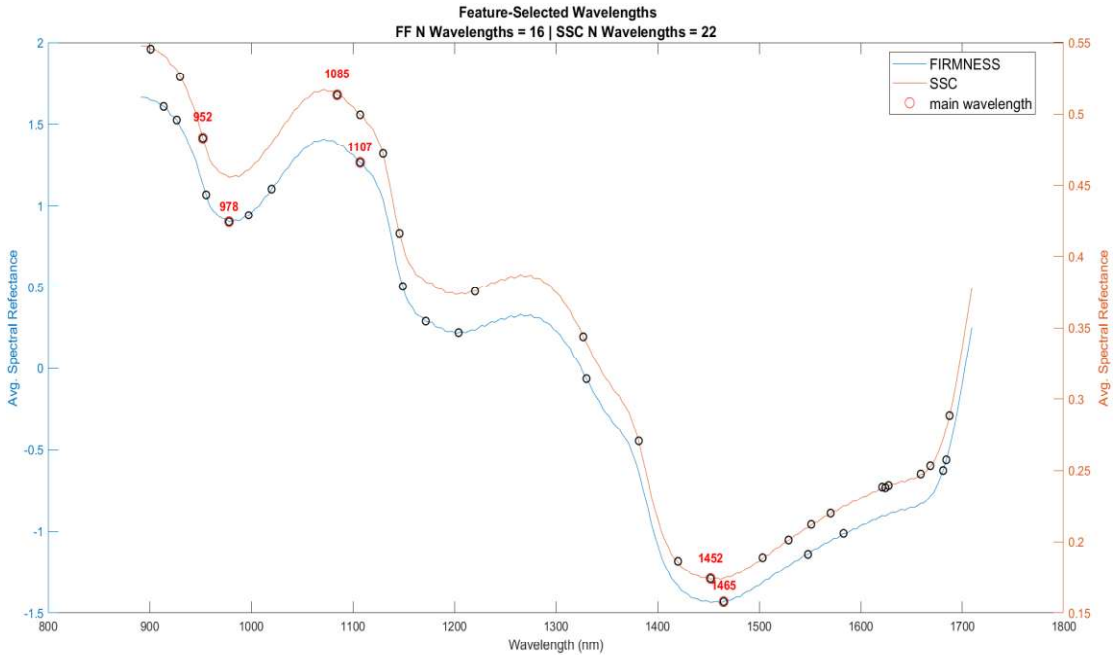


Figure 5.2. Normalized mean spectrum of the wavelengths selected for the prediction of Firmness (16 WL) and SSC (22 WL), highlighting representative peaks (within ±0.1% of the interpolated spectral peak positions) and shared wavelengths.

Vis/NIR spectroscopy can be effectively used to determine the SSC content in kiwifruit [290]. However, one of the main concerns regarding the performance and robustness of NIR models for fruits and vegetables is the influence of the "richness" of variation within the calibration sample, a topic that has been widely discussed [288]. This concern is further highlighted by the performance observed in our firmness prediction.

5.2.2.2 Ripeness classification results

Table 5.2 presents the performance analysis of different ML models for classification of the samples. LDA had relatively good performance with R^2: 0.48, RMSE: 0.46, MAE: 0.40. The smaller value of R^2 with relatively higher error measures show the poor capability of LDA in dealing with nonlinear separate ability between kiwifruit classes. Being a linear model it assumes normal distributed features and equal covariance between classes, which may not be the case for this data set. This is what causes the model to poorly classify samples, particularly when the class boundaries are not well-defined. ANN model ranked as the best model with best R^2 of 0.95, least RMSE of 0.08, and the minimum MAE of 0.03 among the other models. These findings demonstrate great model generalization and accurate detection of the right class of kiwifruit. The better performance of the network may be attributed to its capacity for modeling complex non-linear relationships and learning from higher-

dimensional data. The low value of error rates and high value of determination coefficient show that the ANN model does well with noise and overlapping features, which is the best model for this classification task.

Table 5.2. Performance metrics of ML models and their respective methods or kernel functions used for classifying three classes of kiwifruit.

Model	LDA	ANN	DTs			SVM			
			J48	CART	LMT	RBF	Polynomial	Gaussian	Pearson
R2	0.48	0.95	0.48	0.50	0.48	0.65	0.59	0.61	0.57
RMSE	0.46	0.08	0.46	0.45	0.47	0.35	0.39	0.38	0.42
MAE	0.40	0.03	0.41	0.38	0.42	0.28	0.31	0.29	0.34

We compare results for three DT techniques: J48, CART, and LMT. The best performance was done by LMT with: R^2: 0.50, RMSE: 0.45, MAE: 0.38. J48 and CART produced a little lower R^2 values (0.48) and slightly higher errors especially in MAE (J48: 0.41, CART: 0.42). These results indicate that decision trees can model patterns in structured data, their performance might be inhibited based on learning style hyper-parameters that either learns overly-simplified concepts (underfit) or has trouble generalizing the target function (overfit). Since LMT fuses logistic regression and structure of tree-based learning, it is possible to have a better generalization and error rate than those of the others.

With the SVM model, 4 kernel functions were compared. For kernels, RBF function presented the superior performance with the highest R^2 and the smallest RMSE and MAE. This demonstrates which the RBF network does well with non-linear associations. Excise and Pearson as well as Polynomial kernels presented a slightly lower performance, which means they could not adapt themselves well for the complexity of the data. However, in general, SVM models provided a better trade-off between accuracy and error when compared to LDA and standard decision trees approaches.

Figure 5.3, illustrates confusion matrix for classification of kiwifruits for each ML model. Accuracy of DTs is moderate and there were many wrong classifications between classes A and B which only 25 out of 45 samples of class A and 18 out of 45 of class B were correctly classified, showing that the model performed poorly toward especially these two classes. Class C presented worse results, but still with only 34 correct predictions the tree model seems able to at least distinguish one class better. The low-to-moderate R^2 (0.50) value confirms the low explanatory power, and the relatively high RMSE (0.45) and MAE (0.38) values reflect inaccuracy in prediction, especially for A and class B.

ANN clearly dominates the others, providing almost perfect classification with 1 misclassified data for each class. It was able to accurately predict 44 out of 45 cases in all classes, indicating the good generalization and pattern recognition ability for the classifier. This indicates that the ANN was able to learn the intricate relationships among the input feature and class label. The extremely high R^2 (0.95) and low RMSE (0.08) and MAE (0.03) indicate a high accuracy and small error for the estimation, which is in agreement with the high classification accuracy found in the confusion matrix.

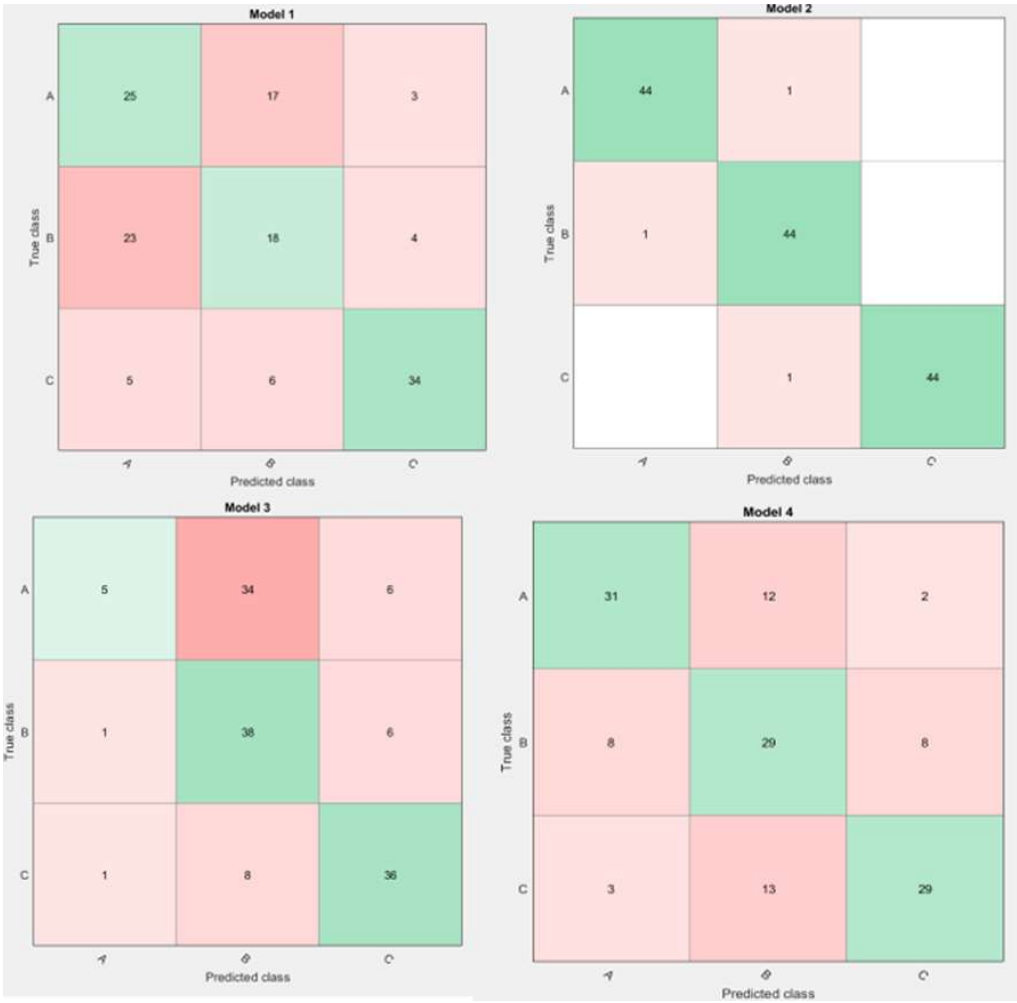


Figure 5.3. Confusion matrices of ML models used for classifying three classes of kiwifruit (A, B, and C): (a) Model 1 – Decision Tree, (b) Model 2 – Artificial Neural Network, (c) Model 3 – Linear Discriminant Analysis, and (d) Model 4 – Support Vector Machine.

Like the DT, the LDA proves to give a relatively poor performance, in particular with class A (only 5 correctly classified out of 45 samples). It is prone to classify class A as class B, which means that there are problems in dealing with non-linearly separable data or overlapping feature distributions. The results of the LDA show a poor ability to discriminate, especially for classes A, and B. The fact that the LDA is linear may lead to this loss of performance in a too complex non-linear classification frontier. The SVM was fairly

weak, yet offered a better class separation in comparison to the DTs and the LDA. It accurately predicted 31, 29 and 29 for class A, B, and C, respectively, but there is still some confusion particularly for class B, which was frequently misclassified as A or C, which suggests that the SVM was better but not entirely stable with regard to data overlap or kernel bound.

Recent studies have demonstrated the effectiveness of ML models in classifying fruit quality attributes. [292] developed four common machine learning (ML) classifiers, the k-mean, naïve Bayes, support vector machine, and feed-forward artificial neural network (FANN), all of which were aimed at classifying the ripeness stage of mangoes at harvest. Further, [293] proposed a novel non-destructive approach for classifying the sweetness of Khao Tang Kwa pomelos by integrating machine learning (ML) techniques with acoustic signal and image processing. They used deep learning and vision-based features to classify, achieving an R^2 of 0.93 and low error values. These findings closely align with our results, where the Artificial Neural Network (ANN) model yielded the best performance for kiwifruit classification (R^2: 0.95, RMSE: 0.08, MAE: 0.03). The effectiveness of ANN in our study mirrors the superior performance of complex, non-linear models highlighted in these works, reaffirming their capacity to handle subtle variations in fruit morphology and physicochemical properties.

5.2.3 Conclusions

This activity investigated the use of a portable near-infrared (NIR) spectrometer for the non-destructive prediction of soluble solids content (SSC) and fruit firmness (FF) in yellow-fleshed kiwifruit, as well as for the classification of ripening stages.

Among the spectral preprocessing strategies evaluated for PLSR-based firmness prediction, several approaches yielded satisfactory performance during cross-validation, with the highest accuracy obtained using Savitzky–Golay second derivative (SG2) and Multiplicative Scatter Correction (MSC). Nevertheless, a reduction in predictive capability was observed during external validation, where the Standard Normal Variate (SNV)-PLS model provided the most reliable compromise between accuracy and robustness, achieving an R^2P value of 0.73. In contrast, SSC prediction models exhibited consistently high performance in both cross-validation and external testing, with the raw spectra-based PLS model delivering the best results overall ($R^2P = 0.93$).

Furthermore, the selection of a limited number of informative wavelengths—16 for firmness and 22 for SSC—proved to be a critical factor for enhancing model applicability.

This reduction in spectral dimensionality supports the use of portable and low-cost NIR devices, enabling accurate predictions while relying on a compact set of spectral features suitable for field applications.

Regarding the ripening stage classification, the artificial neural network (ANN) model markedly outperformed the other classification techniques examined. The ANN achieved a correct classification rate of 97.8% across all ripening classes and demonstrated excellent predictive performance (R^2 = 0.95, RMSE = 0.08, MAE = 0.03). These results highlight the strong learning capacity and generalization ability of neural networks, particularly in modeling complex, nonlinear relationships and managing high-dimensional spectral data.

Overall, the combination of portable NIR spectroscopy with chemometric and artificial intelligence approaches proved to be a highly effective strategy for the non-destructive evaluation of kiwifruit quality and ripening stage discrimination. However, considering regression and classification tasks independently offers only a partial perspective on the decision-making potential associated with fruit quality assessment. The integration of these two components represents a promising direction for future developments, enabling the creation of unified predictive systems in which quantitative quality indicators (e.g., SSC) are directly incorporated into classification frameworks to support more accurate, automated, and informed postharvest management decisions.

5.3 Case study 4: Detection of Drosophila suzukii infestation in sweet cherries

5.3.1 Materials & Methods

Sample preparation and experimental design

The experiment was conducted on organic sweet cherries (Prunus avium L., cv. Sweet Heart), harvested at the end of June from a commercial organic orchard ("Tenute D'Onghia") located in the province of Taranto, Southern Italy. Immediately after harvest, the fruits were transported to the CIHEAM-Bari insectarium (Valenzano, Apulia, Italy), where a controlled infestation procedure was carried out to simulate natural conditions.

Cherries were randomly selected and visually inspected under a stereomicroscope (Nikon SMZ 745T) to exclude damaged or unhealthy fruits. The healthy cherries were then randomly divided into three identical batches. The first batch was introduced into a rearing cage containing Drosophila suzukii adults to allow oviposition. After a two-day exposure period, corresponding to the oviposition phase of the insect life cycle, the fruits were removed from the cage and examined under the stereomicroscope to determine the number of eggs per fruit.

A total of 100 infested cherries were classified into four groups based on egg density: (i) 1–5 eggs per fruit; (ii) 6–10 eggs per fruit, with five fruits per egg-number subgroup; (iii) 11–30 eggs per fruit, again with five fruits per subgroup; and (iv) more than 30 eggs per fruit. A second batch, consisting of 100 healthy and non-infested cherries, was used as the control group. Visible/short-wave near-infrared (Vis/SWNIR) spectral measurements were performed on both healthy and infested fruits 48 h after harvest, during which all samples were maintained under identical ambient temperature and relative humidity conditions. Subsequently, fruits were stored in a cold chamber for 24 h at 0 °C and 90–95% relative humidity, after which a second spectral acquisition was performed. This experimental design allowed the evaluation of spectral variability associated with temperature changes, with approximately 50% of the samples measured under different thermal conditions.

Spectral data were collected using a VIS–NIR spectrophotometer (AvaSpec-2048-UA, Avantes, The Netherlands), operating in the 200–1100 nm range and equipped with a 2048-pixel CCD detector. Diffuse reflectance measurements were obtained using an integrating sphere (AvaSphere-80-REFL, Avantes), coupled with a 100 W tungsten–halogen light source (ASBN-100W-L, SP Spectral Products, USA). For each fruit, four spectra were acquired by rotating the cherry by 90° around its symmetry axis between successive measurements, and the resulting spectra were averaged to obtain a representative spectral signature. The acquisition was managed using Avasoft Basic software (v. 7.3), while subsequent data processing and analysis were carried out using custom scripts developed in MATLAB (R2024b, MathWorks, USA).

Data analysis and statistical modeling

Principal Component Regression (PCR) and Partial Least Squares (PLS) algorithms were selected for model development, as implemented in the MATLAB Statistics and Machine Learning Toolbox. To accelerate computational performance, parallel processing was enabled through the Parallel Computing Toolbox, and analyses were executed on a Dell Precision T3610 workstation (Intel Xeon, 12 cores, 3.7 GHz).

The response variable (Y) was defined as a categorical parameter representing fruit health status, with healthy and infected cherries assigned values of −1 and +1, respectively. Since the regression models produced continuous outputs, classification was achieved using a decision threshold set to zero: predicted Y values equal to or above the threshold were classified as infected, while values below zero were classified as healthy. External cross-validation (EXTCV) was performed using 20% of the dataset, randomly selected, while model calibration was validated using a 2-fold cross-validation approach. Model

performance was assessed through the squared correlation coefficient for calibration (R^2CR) and cross-validation (R^2CV), based on 50 Monte Carlo simulation runs.

Additional performance indicators were considered, including the limit of confident detection at 95% probability (LODP@95%), defined as 1.96 times the root mean square error of prediction; the range of confident prediction error percentage (RANGEPERR%@95%); and the residual predictive deviation (RPD), calculated as the ratio between the standard deviation of calibration data and the standard error of prediction. Relevant spectral features were identified through an iterative wavelength selection procedure based on regression coefficients obtained during cross-validation, followed by the application of the CVA method. At each iteration, wavelengths identified as poorly informative were removed and the model recalculated using a newly randomized calibration set. This process was repeated until no further wavelengths could be excluded. The optimal wavelength subset was then determined by identifying the configuration associated with the maximum RPD value, which was statistically validated through 50 Monte Carlo runs.

Following wavelength selection, model performance was further evaluated using EXTCV by calculating the false positive rate (FPR) and false negative rate (FNR). FPR represents the proportion of healthy fruits incorrectly classified as infected, whereas FNR quantifies the proportion of infected fruits misclassified as healthy, the latter being a critical indicator due to its direct implications for consumer safety. Spectral preprocessing strategies were also investigated to enhance classification accuracy. These included the use of raw or standardized spectra (mean-centered and variance-scaled), as well as the application of Savitzky–Golay derivatives (first or second order, window width of 15 points, polynomial order of two). Outlier detection and removal were not applied in this study. Each model configuration was identified using the notation M–D–OPTN, where M indicates the regression method (PCR or PLS), D the derivative order, and OPTN the preprocessing option.

The modeling workflow consisted of three main steps. First, the complete modeling procedure was applied once to identify candidate optimal models. Second, the entire process was repeated ten times using randomized sample splits and 20% EXTCV to perform a Monte Carlo–like evaluation of model stability, enabling the statistical estimation of FPR and FNR. The best-performing model was selected based on these results. Finally, model robustness was assessed by analyzing the frequency with which individual wavelengths were selected across the ten repetitions. A selection threshold (S) was defined to identify wavelengths consistently retained across multiple runs, and robustness was evaluated by varying S from

1 to 10 and increasing EXTCV from 20% to 70%. For each wavelength subset, FPR and FNR were plotted as a function of EXTCV and fitted using an exponential model. Performance scores were derived based on predefined thresholds (0.01% for FNR and 0.1% for FPR), with higher scores indicating superior statistical robustness. While this approach required substantial computational time, it enabled the identification of a final model characterized by optimized coefficients and high statistical reliability, expressed through the coefficient of variation (CV%).

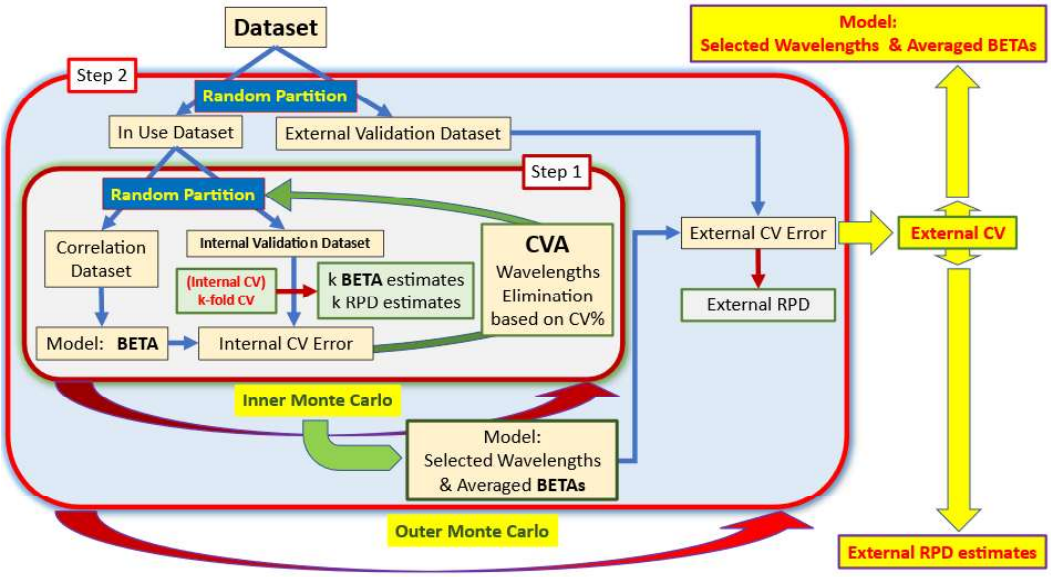


Figure 5.4. This figure depicts the general Step1 and Step 2 layout, as previously described, of the programmed script of Matlab evaluating the models.

5.3.2 Results and discussions

Model selection and statistical validation (Steps 1 and 2)

The first two steps of the modeling strategy were aimed at identifying the most reliable classification model among the tested M–D–OPTN configurations and at evaluating its statistical stability through repeated randomization. This was achieved by repeating the calibration–validation procedure ten times on randomized datasets, using an external cross-validation (EXTCV) rate fixed at 20%, thus enabling a Monte Carlo–like assessment of model performance.

Model performance was primarily evaluated in terms of false negative rate (FNR) and false positive rate (FPR), as these metrics directly reflect consumer risk and unnecessary product rejection, respectively. As shown in Figure 5.5, the comparative analysis of mean FNR and FPR values across the randomized repetitions clearly indicated that the PLS-0-0 model (PLS applied to raw spectra without derivatives or additional preprocessing) consistently outperformed the alternative configurations. Notably, this model achieved an

FNR equal to 0% in all ten randomized runs, demonstrating its robustness in minimizing the most critical classification error, namely the misclassification of infected fruits as healthy.

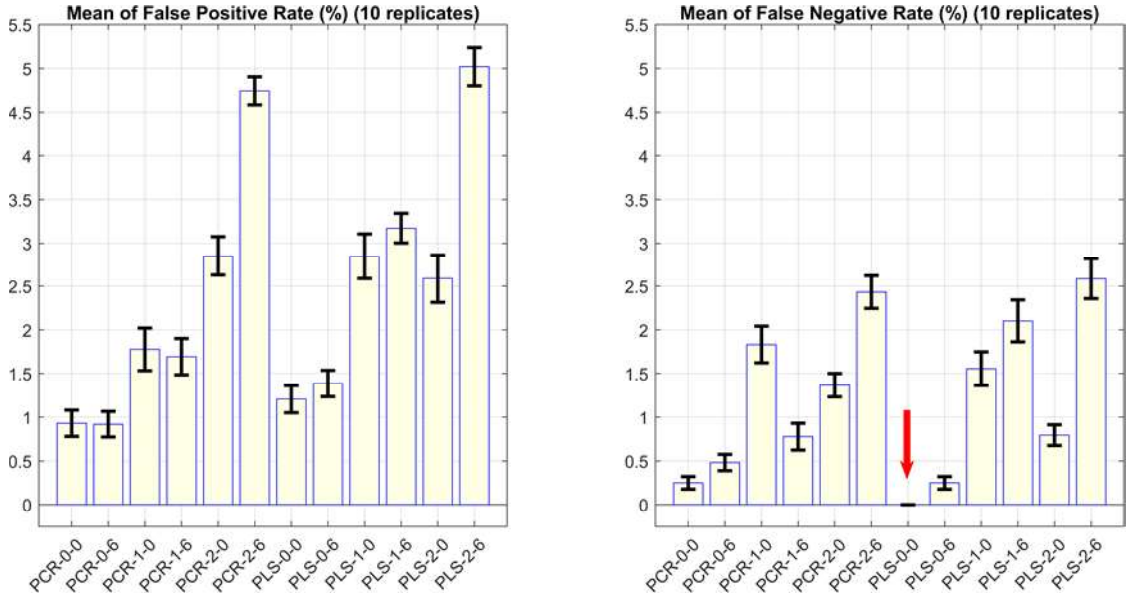


Figure 5.5. Bar plot, as the mean and standard deviation of the mean, of the results of the second step consisting of repeating 10 times the first step on randomized samples and setting EXTCV to 20%. The results are expressed as FNR and FPR in percent. The red arrow indicates the optimal model with regard to FNR (i.e. the consumer risk).

The statistical reliability of the adopted performance metrics was further investigated by analyzing the relationships among RPD, R^2CV, and RANGEPERR%@95%. The strong agreement between RPD and R^2CV is illustrated in Figure 5.6a, confirming the internal consistency of the modeling framework. However, because RPD does not directly provide information on the statistically guaranteed prediction error for unknown samples, the parameter RANGEPERR%@95% was introduced to improve interpretability. As shown in Figure 5.6b, RPD and RANGEPERR%@95% are inversely correlated through a hyperbolic relationship, indicating that higher predictive strength is associated with lower guaranteed prediction error. This behavior supports the use of RANGEPERR%@95% as a complementary and user-oriented indicator of model reliability.

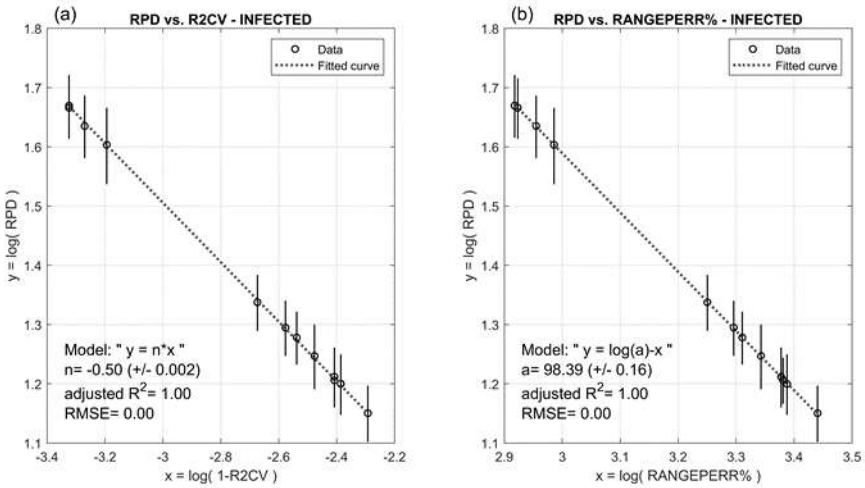


Figure 5.6.(a) Relationship between the RPD and R2CV, for the considered replicate, for all the runs performed, for all the considered models (PCR, PLS) and all the grouped Monte Carlo runs. (b) Relationship between RANGEPERR%@95% and RPD; the correlation found is hyperbolic, RPD is inversely correlated with RANGEPERR%@95%, i.e. their product is a constant.

Model robustness and wavelength subset optimization (Step 3)

The third step focused on evaluating the robustness of the selected PLS-0-0 model and identifying the most informative subset of wavelengths. This analysis was carried out by varying the wavelength selection threshold (S), defined as the minimum number of repetitions in which a wavelength must be selected, and by progressively increasing EXTCV from 20% to 70%.

As reported in Figure 5.7, the optimal compromise between robustness and model complexity was obtained for S = 5, where both the FNR and FPR scores reached their maximum values. At this threshold, the resulting wavelength subset consisted of 261 variables and was therefore selected for further analysis.

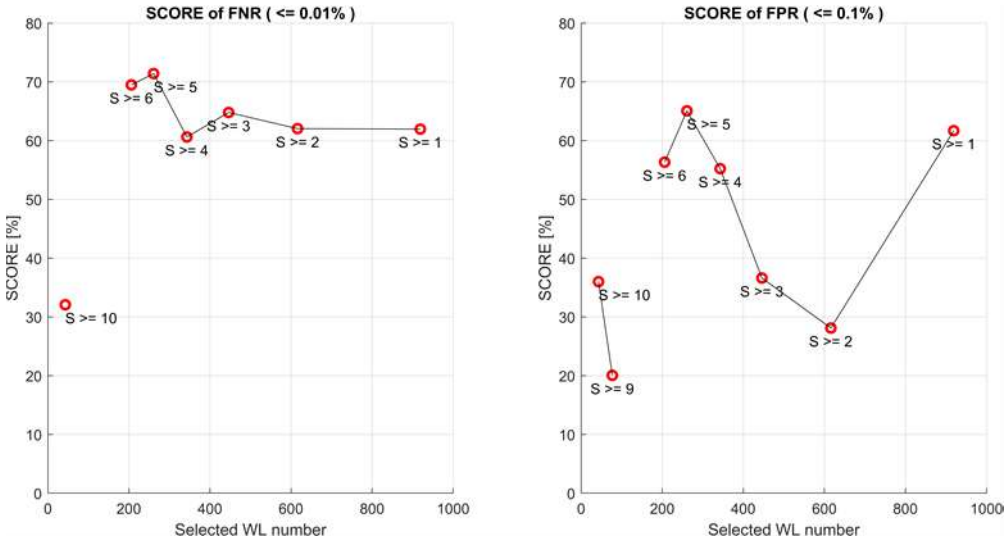


Figure 5.7. FNR and FPR SCORE for the PLS-0-0 model. The S value measures the number of replicates of the second step (from 1 to 10) in which the specific wavelength was selected.

The dependence of FNR and FPR on EXTCV for the selected wavelength subset followed an exponential trend. As shown in Figure 5.8, FNR values remained below the critical threshold even under increasingly stringent validation conditions, while Figure 5.9 illustrates the corresponding behavior of FPR. The exponential fits applied to these data allowed the derivation of quantitative SCORE values, confirming the stability and generalization capability of the selected model.

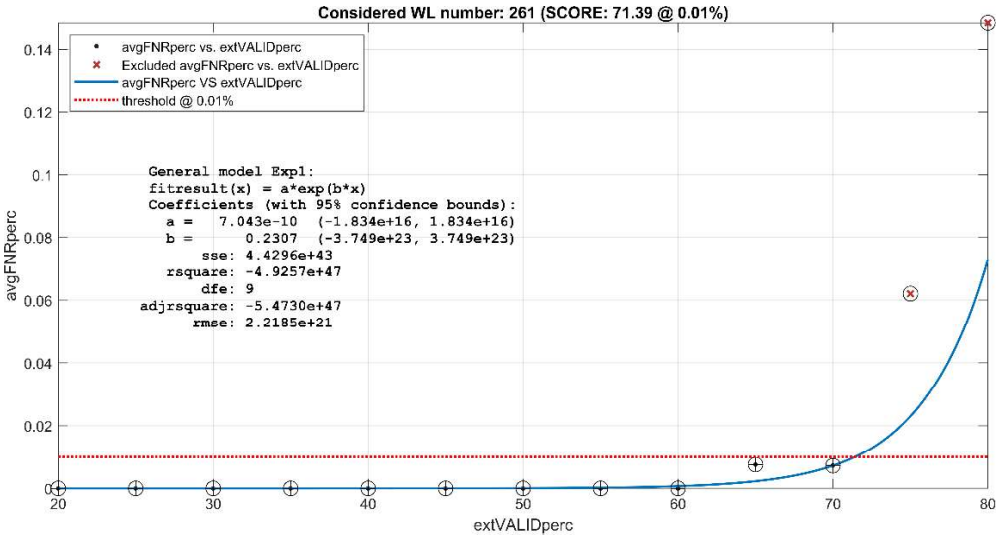


*Figure 5.8. FNR values versus EXTCV for the optimal S>=5. An exponential fit (y=a*exp(b*x)) has been considered on these data and the threshold of 0.01% has been used to express a SCORE. The SCORE for FNR represents the EXTCV value at which the FNR reaches the 0.01%.*

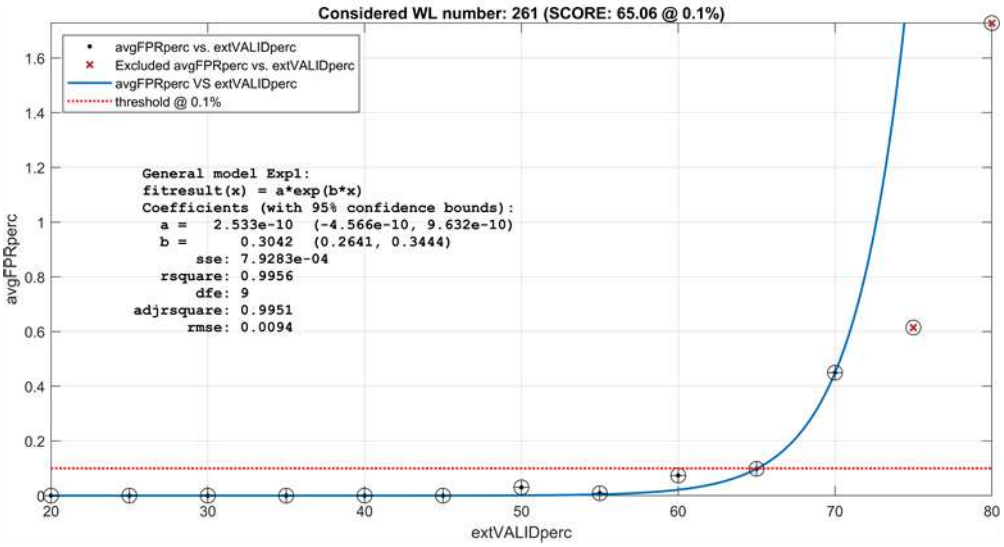


*Figure 5.9FPR values versus EXTCV for the optimal S>=5. An exponential fit (y=a*exp(b*x)) has been considered on these data and the threshold of 0.01% has been used to express a SCORE. The SCORE for FPR represents the EXTCV value at which the FPR reaches the 0.1%.*

Final model and interpretation of selected wavelengths

The final model was derived by combining the outcomes of model selection and robustness analysis. Model coefficients (BETA values) were computed as the average of

multiple estimates obtained under varying EXTCV conditions, and their uncertainty was quantified using the coefficient of variation (CV%). An initial inspection of coefficient stability revealed that a small number of wavelengths were associated with relatively high CV% values, indicating limited robustness. These wavelengths were therefore excluded, resulting in a refined subset of 247 variables. The coefficients of the final model and their associated uncertainty are shown in Figure 5.10, where all CV% values fall below 10%, indicating high statistical stability. The robustness of the final model was further confirmed through 500 Monte Carlo runs using an EXTCV of 50%, which yielded extremely low error rates (FNR = 0.004% ± 0.003; FPR = 0.02% ± 0.01).

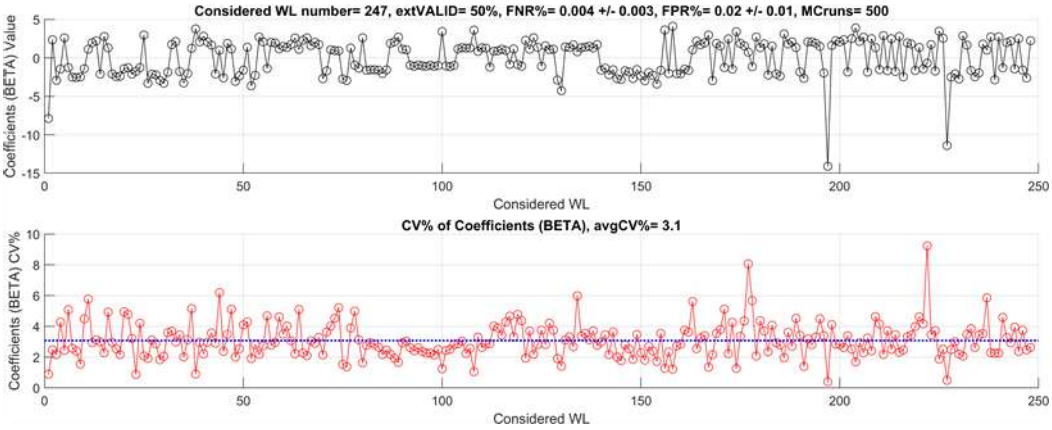


Figure 5.10. BETA coefficients of the final model and their uncertainty; FNR and FPR, after 500 Monte Carlo runs, are 0.004% +/- 0.003 and 0.02% +/- 0.01 respectively. The considered EXTCV is set to 50% of the randomized overall samples.

The spectral distribution of the selected wavelengths, together with the average normalized spectra of healthy and infected cherries, is presented in Figure 5.11. A clear absence of selected wavelengths in the 600–700 nm region can be observed, corresponding to the orange-to-dark-red spectral range. This finding indicates that this region does not contribute to the discrimination of D. suzukii egg infestation, whereas the remaining spectral intervals contain the relevant information for accurate classification.

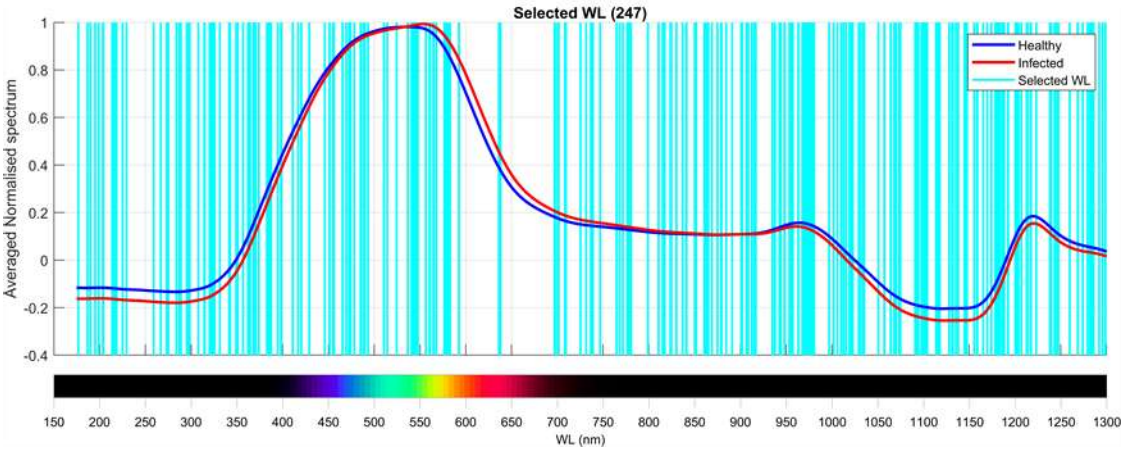


Figure 5.11. The final model selected wavelengths (247) and the average normalized spectrum of all HEALTHY and INFECTED cherries.

5.3.3 Conclusions

To date, relatively few studies have explored the application of non-destructive analytical techniques for the detection of insect infestation in sweet cherries. Given the advantages offered by spectrophotometric approaches in terms of speed, objectivity, and non-invasiveness, this study investigated the potential of Vis/SWNIR spectroscopy for the rapid detection of postharvest Drosophila suzukii infestations in intact sweet cherry fruits, with the ultimate goal of supporting fruit sorting and grading operations.

The primary objective was to assess the feasibility of Vis/SWNIR spectrophotometry as a rapid and non-destructive tool for discriminating between healthy cherries and fruits infested with SWD eggs. To this end, both Principal Component Regression (PCR) and Partial Least Squares (PLS) algorithms were implemented and systematically evaluated within a robust statistical framework. Model validation relied on an external cross-validation strategy, initially set at 20% of the available samples and progressively increased up to 50% for the final optimized model, thus ensuring a stringent assessment of predictive performance.

A key outcome of the study was the effective identification of the spectral regions most strongly associated with infestation status. This was achieved by analyzing regression coefficients derived from cross-validation and applying the CVA-based wavelength selection method, which proved highly effective in isolating the most informative wavelengths while reducing model complexity. The extensive use of Monte Carlo simulations throughout the modeling process played a central role in statistically validating model parameters, coefficients, and performance metrics, allowing a reliable estimation of model robustness and uncertainty.

The three-step modeling strategy adopted in this work enabled the selection of a highly accurate and stable classification model. After 500 Monte Carlo runs and using 50% of the dataset for external validation, the final model achieved extremely low error rates, with a false negative rate of 0.004% ± 0.003 and a false positive rate of 0.02% ± 0.01. These results highlight the strong potential of Vis/SWNIR spectroscopy, combined with advanced chemometric techniques, as an effective decision-support tool for the non-destructive detection of SWD infestation in sweet cherries. The proposed approach represents a promising step toward the implementation of automated, reliable, and consumer-safe quality control systems in postharvest fruit handling and grading chains.

CHAPTER 6

6 Conclusions and Outlooks

6.1 Conclusions

The research activity developed within this doctoral thesis successfully addressed a set of concrete and interrelated challenges that currently limit the operational adoption of smart sensing and robotic systems in precision agriculture. The work did not focus on a single technological aspect, but rather on the coordinated resolution of sensing, modelling, embedded computation, and robotic integration issues, with the explicit aim of moving from laboratory-level solutions toward field-deployable systems.

1) The first major problem tackled by the thesis concerns the practical use of multispectral cameras for proximal crop monitoring. Multispectral systems are inherently affected by band misalignment at the pixel level, which severely compromises the reliability of vegetation indices and quantitative analyses when operating at short distances. This work systematically analyzed the causes and effects of spectral misalignment and demonstrated that accurate pixel-level alignment can be achieved through sensor-less approaches, without relying on additional hardware or external calibration targets. By explicitly quantifying reprojection errors across spectral bands and distances, the thesis clarified the operational limits of short-range multispectral imaging and provided practical guidelines for reliable image alignment under greenhouse and proximal sensing conditions.

2) The second fundamental contribution lies in the management and improvement of multilinear and multivariate statistical models with high predictive capability and robustness. Across multiple case studies, ranging from crop health monitoring and disease detection to harvest assistance and postharvest quality assessment, the thesis demonstrated how chemometric and machine learning models can be strengthened through appropriate preprocessing, wavelength selection, and validation strategies. Particular attention was devoted to avoiding overfitting and ensuring model generalization by systematically using external validation and Monte Carlo approach.

This methodological rigor allowed the identification of models that are not only statistically accurate, but also physiologically coherent and operationally reliable.

3) From an embedded systems perspective, the thesis addressed a practical but often underestimated problem: the long-term reliability of low-cost computing platforms in continuous agricultural operation. The implementation of sensing and management algorithms on Raspberry Pi devices required a careful redesign of software architecture to avoid intensive write operations on SD cards. By adopting a temporary file system in RAM, the work ensured stable execution of data acquisition and processing pipelines while significantly reducing the risk of hardware/software failure and data corruption. This contribution represents a key step toward the deployment of smart agricultural devices capable of operating continuously in real environments.

4) In addition, the thesis delivered a detailed and reproducible bill of materials for the construction of a modular smart agricultural cart based on Robot Operating System 2. Rather than remaining at a conceptual level, the work translated system requirements into concrete hardware and software components, demonstrating how ROS 2 can be effectively used to integrate sensing, motion control, communication, and supervision in a distributed architecture. This aspect of the work enhances reproducibility and provides a clear reference framework for future developments and technology transfer.

5) The implementation of Autonomous Navigation for Mobile Robots (ANMR) was addressed through a structured ROS 2-based approach. The proof-of-concept validation using a TurtleBot3 platform demonstrated the feasibility of integrating SLAM, LiDAR-based perception, odometry, and IMU data within a unified navigation framework. The experiment confirmed the interoperability of the developed software modules and validated the methodological pathway required to transfer autonomous navigation capabilities to the full-scale ABOTx1 smart cart.

6) Finally, the thesis critically assessed the computational limitations of the Raspberry Pi 4 platform. Indeed, while suitable for sensor coordination, low-level control, and prototyping, the limited throughput of Raspberry Pi 4 platform was shown to be insufficient for advanced real-time image processing and data-intensive machine learning tasks. Based on these findings, the work clearly identifies the migration toward more powerful embedded platforms, such as the NVIDIA Jetson Orin Nano, as a necessary subsequent step. The expected order-of-magnitude increase in computational

capability (x200) will open new opportunities for real-time multispectral processing, on-board inference, and closed-loop perception–action systems. Therefore, the next step will consist in the complete migration of the software system set up onto one or more cooperating Jetson Orin Nano and this will allow for the management of a hyperspectral camera (e.g. Specim FX10), a point cloud lidar camera (e.g. ORBBEC Gemini Pro Depth Camera, Structured-Light, 3D Imaging) and a robotic arm (e.g. Elephant Robotics myPalletizer Compact 4-Axis Robotic Arm) onto the ABOTx1 smart cart platform.

6.2 Outlooks and future developments

Looking forward, this thesis establishes a solid foundation for several strategic research and technological developments.

From a sensing and modelling perspective, future work should focus on extending spectral coverage beyond the Vis–NIR region, particularly toward the SWIR domain, to improve sensitivity to biochemical compounds and subtle physiological changes. The integration of spectral, environmental, and structural data represents a promising direction for enhancing early stress detection and reducing ambiguity in presymptomatic stages.

On the modelling side, the availability of increased computational power will enable the deployment of lightweight deep learning architectures and hybrid models that combine the interpretability of chemometrics with the representational capacity of neural networks. Such approaches are particularly relevant for handling nonlinear responses, heterogeneous datasets, and complex decision-making tasks such as threshold-based intervention timing.

From an engineering and robotic perspective, the next step consists in completing the electrical integration of the ABOTx1 platform and transferring the validated ANMR framework to the full-scale cart. The coupling of perception with targeted actuation, such as localized spraying or selective harvesting tools, will allow the transition from monitoring to autonomous intervention. In this context, the smart cart can evolve from a sensor carrier into an intelligent robotic assistant capable of supporting site-specific and resource-efficient agricultural practices.

In terms of real-world deployment, future developments should also address system robustness, operational scalability, and integration with existing farm management infrastructures. Field validation in commercial farms, together with long-term monitoring campaigns, will be essential to assess reliability under varying environmental conditions and operational constraints. Moreover, coupling the proposed system with digital farm

management platforms could enable seamless data-driven decision support for growers, facilitating the adoption of precision agriculture technologies in real production contexts.

More broadly, the outcomes of this thesis contribute to a vision of precision agriculture in which sustainability, efficiency, and digitalization are achieved not through isolated technologies, but through integrated systems that combine sensing, intelligence, and robotics. By addressing practical constraints alongside scientific challenges, this work defines a concrete and scalable pathway toward real-world deployment of smart agricultural platforms.

References

1. FAO The Future of Food and Agriculture and Challenges. 2017.
2. Kirchmann, H.; Thorvaldsson, G. Challenging Targets for Future Agriculture. *European Journal of Agronomy* 2000, *12*, 145–161, doi:10.1016/S1161-0301(99)00053-2.
3. Qin, Y.; Horvath, A. What Contributes More to Life-Cycle Greenhouse Gas Emissions of Farm Produce: Production, Transportation, Packaging, or Food Loss? *Resour Conserv Recycl* 2022, *176*, 105945, doi:10.1016/j.resconrec.2021.105945.
4. FAO Emissions Due to Agriculture. Global, Regional and Country Trends 2000–2018. 2020.
5. Guo, L.; Zhao, S.; Song, Y.; Tang, M.; Li, H. Green Finance, Chemical Fertilizer Use and Carbon Emissions from Agricultural Production. *Agriculture 2022, Vol. 12, Page 313* 2022, *12*, 313, doi:10.3390/AGRICULTURE12030313.
6. Balafoutis, A.; Beck, B.; Fountas, S.; Vangeyte, J.; Van Der Wal, T.; Soto, I.; Gómez-Barbero, M.; Barnes, A.; Eory, V. Precision Agriculture Technologies Positively Contributing to GHG Emissions Mitigation, Farm Productivity and Economics. *Sustainability 2017, Vol. 9, Page 1339* 2017, *9*, 1339, doi:10.3390/SU9081339.
7. Bongiovanni, R.; Lowenberg-Deboer, J. Precision Agriculture and Sustainability. *Precis Agric* 2004, *5*, 359–387, doi:10.1023/B:PRAG.0000040806.39604.aa.
8. Moysiadis, V.; Sarigiannidis, P.; Vitsas, V.; Khelifi, A. Smart Farming in Europe. *Comput Sci Rev* 2021, *39*, 100345, doi:10.1016/j.cosrev.2020.100345.
9. Kamilaris, A.; Kartakoullis, A.; Prenafeta-Boldú, F.X. A Review on the Practice of Big Data Analysis in Agriculture. *Comput Electron Agric* 2017, *143*, 23–37, doi:10.1016/j.compag.2017.09.037.
10. Gallardo, M.; Elia, A.; Thompson, R.B. Decision Support Systems and Models for Aiding Irrigation and Nutrient Management of Vegetable Crops. *Agric Water Manag* 2020, *240*, 106209, doi:10.1016/j.agwat.2020.106209.
11. Gebbers, R.; Adamchuk, V.I. Precision Agriculture and Food Security. *Science (1979)* 2010, *327*, 828–831, doi:10.1126/science.1183899.
12. Zhang, N.; Wang, M.; Wang, N. Precision Agriculture—a Worldwide Overview. *Comput Electron Agric* 2002, *36*, 113–132, doi:10.1016/S0168-1699(02)00096-0.
13. Padilla, F.M.; Gallardo, M.; Peña-Fleitas, M.T.; De Souza, R.; Thompson, R.B. Proximal Optical Sensors for Nitrogen Management of Vegetable Crops: A Review. *Sensors 2018, Vol. 18, Page 2083* 2018, *18*, 2083, doi:10.3390/S18072083.
14. Ihuoma, S.O.; Madramootoo, C.A. Recent Advances in Crop Water Stress Detection. *Comput Electron Agric* 2017, *141*, 267–275, doi:10.1016/J.COMPAG.2017.07.026.
15. Ihuoma, S.O.; Madramootoo, C.A. Sensitivity of Spectral Vegetation Indices for Monitoring Water Stress in Tomato Plants. *Comput Electron Agric* 2019, *163*, 104860, doi:10.1016/J.COMPAG.2019.104860.
16. Matese, A.; Baraldi, R.; Berton, A.; Cesaraccio, C.; Di Gennaro, S.F.; Duce, P.; Facini, O.; Mameli, M.G.; Piga, A.; Zaldei, A. Estimation of Water Stress in Grapevines Using Proximal and Remote Sensing Methods. *Remote Sensing 2018, Vol. 10, Page 114* 2018, *10*, 114, doi:10.3390/RS10010114.
17. Krishna, G.; Sahoo, R.N.; Singh, P.; Patra, H.; Bajpai, V.; Das, B.; Kumar, S.; Dhandapani, R.; Vishwakarma, C.; Pal, M.; et al. Application of Thermal Imaging and Hyperspectral Remote Sensing for Crop Water Deficit Stress Monitoring. *Geocarto Int* 2021, *36*, 481–498, doi:10.1080/10106049.2019.1618922.
18. Cubero, S.; Marco-Noales, E.; Aleixos, N.; Barbé, S.; Blasco, J. RobHortic: A Field Robot to Detect Pests and Diseases in Horticultural Crops by Proximal Sensing. *Agriculture* 2020, *10*, 276, doi:10.3390/agriculture10070276.
19. Samseemoung, G.; Soni, P.; Suwan, P. Development of a Variable Rate Chemical Sprayer for Monitoring Diseases and Pests Infestation in Coconut Plantations. *Agriculture 2017, Vol. 7, Page 89* 2017, *7*, 89, doi:10.3390/AGRICULTURE7100089.
20. Das, B.; Manohara, K.K.; Mahajan, G.R.; Sahoo, R.N. Spectroscopy Based Novel Spectral Indices, PCA- and PLSR-Coupled Machine Learning Models for Salinity Stress Phenotyping of Rice. *Spectrochim Acta A Mol Biomol Spectrosc* 2020, *229*, 117983, doi:10.1016/J.SAA.2019.117983.
21. Kurunc, A. Effects of Water and Salinity Stresses on Growth, Yield, and Water Use of Iceberg Lettuce. *J Sci Food Agric* 2021, *101*, 5688–5696, doi:10.1002/JSFA.11223.
22. Ihuoma, S.O.; Madramootoo, C.A. Narrow-Band Reflectance Indices for Mapping the Combined Effects of Water and Nitrogen Stress in Field Grown Tomato Crops. *Biosyst Eng* 2020, *192*, 133–143, doi:10.1016/J.BIOSYSTEMSENG.2020.01.017.
23. Zhang, J.; Xie, T.; Yang, C.; Song, H.; Jiang, Z.; Zhou, G.; Zhang, D.; Feng, H.; Xie, J. Segmenting Purple Rapeseed Leaves in the Field from UAV RGB Imagery Using Deep Learning as an Auxiliary

Means for Nitrogen Stress Detection. *Remote Sensing 2020, Vol. 12, Page 1403* 2020, *12*, 1403, doi:10.3390/RS12091403.
24. Pallottino, F.; Antonucci, F.; Costa, C.; Bisaglia, C.; Figorilli, S.; Menesatti, P. Optoelectronic Proximal Sensing Vehicle-Mounted Technologies in Precision Agriculture: A Review. *Comput Electron Agric* 2019, *162*, 859–873, doi:10.1016/j.compag.2019.05.034.
25. Gitelson, A.A.; Gritz, Y.; Merzlyak, M.N. Relationships between Leaf Chlorophyll Content and Spectral Reflectance and Algorithms for Non-Destructive Chlorophyll Assessment in Higher Plant Leaves. *J Plant Physiol* 2003, *160*, 271–282, doi:10.1078/0176-1617-00887.
26. Gitelson, A.A.; Merzlyak, M.N.; Chivkunova, O.B. Optical Properties and Nondestructive Estimation of Anthocyanin Content in Plant Leaves. *Volume 74, Issue 1, Pages 38 - 45* 2001, *74*, 38, doi:10.1562/0031-8655(2001)074<0038:OPANEO>2.0.CO;2.
27. Liew, O.W.; Chong, P.C.J.; Li, B.; Asundi, A.K. Signature Optical Cues: Emerging Technologies for Monitoring Plant Health. *Sensors 2008, Vol. 8, Pages 3205-3239* 2008, *8*, 3205–3239, doi:10.3390/S8053205.
28. Omia, E.; Bae, H.; Park, E.; Kim, M.S.; Baek, I.; Kabenge, I.; Cho, B.K. Remote Sensing in Field Crop Monitoring: A Comprehensive Review of Sensor Systems, Data Analyses and Recent Advances. *Remote Sensing 2023, Vol. 15, Page 354* 2023, *15*, 354, doi:10.3390/RS15020354.
29. Zhang, X.; Fusion, H.Z.-I.; 2021, undefined Hyperspectral-Cube-Based Mobile Face Recognition: A Comprehensive Review. *ElsevierX Zhang, H ZhaoInformation Fusion, 2021•Elsevier*.
30. Gao, L.; Wang, L. V. A Review of Snapshot Multidimensional Optical Imaging: Measuring Photon Tags in Parallel. *Phys Rep* 2016, *616*, 1–37, doi:10.1016/J.PHYSREP.2015.12.004.
31. Zubler, A. V.; Yoon, J.Y. Proximal Methods for Plant Stress Detection Using Optical Sensors and Machine Learning. *Biosensors 2020, Vol. 10, Page 193* 2020, *10*, 193, doi:10.3390/BIOS10120193.
32. Xue, J.; Su, B. Significant Remote Sensing Vegetation Indices: A Review of Developments and Applications. *J Sens* 2017, *2017*, 1–17, doi:10.1155/2017/1353691.
33. Xu, S.; Xu, X.; Blacker, C.; Gaulton, R.; Zhu, Q.; Yang, M.; Yang, G.; Zhang, J.; Yang, Y.; Yang, M.; et al. Estimation of Leaf Nitrogen Content in Rice Using Vegetation Indices and Feature Variable Optimization with Information Fusion of Multiple-Sensor Images from UAV. *Remote Sensing 2023, Vol. 15, Page 854* 2023, *15*, 854, doi:10.3390/RS15030854.
34. Lipovac, A.; Bezdan, A.; Moravčević, D.; Djurović, N.; Ćosić, M.; Benka, P.; Stričević, R. Correlation between Ground Measurements and UAV Sensed Vegetation Indices for Yield Prediction of Common Bean Grown under Different Irrigation Treatments and Sowing Periods. *Water 2022, Vol. 14, Page 3786* 2022, *14*, 3786, doi:10.3390/W14223786.
35. Mahlein, A.K. Plant Disease Detection by Imaging Sensors – Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping. *Plant Dis* 2016, *100*, 241–254.
36. Oerke, E.C.; Mahlein, A.K.; Steiner, U. Proximal Sensing of Plant Diseases. *Detection and Diagnostics of Plant Pathogens* 2014, 55–68, doi:10.1007/978-94-017-9020-8_4/COVER.
37. Rouse, J.W., Jr.; Haas, R.H.; Schell, J.A.; Deering, D.W. Monitoring Vegetation Systems in the Great Plains with ERTS. *NASA. Goddard Space Flight Center 3d ERTS-1 Symp., Vol. 1, Sect. A* 1974.
38. Penuelas, J.; Pinol, J.; Ogaya, R.; Filella, I. Estimation of Plant Water Concentration by the Reflectance Water Index WI (R900/R970). *Taylor & FrancisJ Penuelas, J Pinol, R Ogaya, I FilellaInternational journal of remote sensing, 1997•Taylor & Francis* 1997, *18*, 2869–2875, doi:10.1080/014311697217396.
39. Gamon, J.; Penuelas, J.; environment, C.F.-R.S. of; 1992, undefined A Narrow-Waveband Spectral Index That Tracks Diurnal Changes in Photosynthetic Efficiency. *ElsevierJA Gamon, J Penuelas, CB FieldRemote Sensing of environment, 1992•Elsevier*.
40. Al-Gaadi, K.A.; Tola, E.; Alameen, A.A.; Madugundu, R.; Marey, S.A.; Zeyada, A.M.; Edrris, M.K. Control and Monitoring Systems Used in Variable Rate Application of Solid Fertilizers: A Review. *J King Saud Univ Sci* 2023, *35*, 102574, doi:10.1016/j.jksus.2023.102574.
41. Basso, B.; Ritchie, J.T.; Pierce, F.J.; Braga, R.P.; Jones, J.W. Spatial Validation of Crop Models for Precision Agriculture. *Agric Syst* 2001, *68*, 97–112, doi:10.1016/S0308-521X(00)00063-9.
42. Khanal, S.; Fulton, J.; Klopfenstein, A.; Douridas, N.; Shearer, S. Integration of High Resolution Remotely Sensed Data and Machine Learning Techniques for Spatial Prediction of Soil Properties and Corn Yield. *Comput Electron Agric* 2018, *153*, 213–225, doi:10.1016/J.COMPAG.2018.07.016.
43. Diacono, M.; Rubino, P.; Montemurro, F. Precision Nitrogen Management of Wheat. A Review. *Agronomy for Sustainable Development 2012 33:1* 2012, *33*, 219–241, doi:10.1007/S13593-012-0111-Z.

44. Toscano, P.; Castrignanò, A.; Di Gennaro, S.F.; Vonella, A.V.; Ventrella, D.; Matese, A. A Precision Agriculture Approach for Durum Wheat Yield Assessment Using Remote Sensing Data and Yield Mapping. *Agronomy 2019, Vol. 9, Page 437* 2019, *9*, 437, doi:10.3390/AGRONOMY9080437.
45. Serrano, J.; Shahidian, S.; da Silva, J.M.; Paixão, L.; Moral, F.; Carmona-Cabezas, R.; Garcia, S.; Palha, J.; Noéme, J. Mapping Management Zones Based on Soil Apparent Electrical Conductivity and Remote Sensing for Implementation of Variable Rate Irrigation—Case Study of Corn under a Center Pivot. *Water 2020, Vol. 12, Page 3427* 2020, *12*, 3427, doi:10.3390/W12123427.
46. Serrano, L.; Muriel, S.; Martínez-Ortega, M.; San, M.; De La Parte, E.; Serrano, S.L.; Elduayen, M.M.; Martínez-Ortega, J.-F. Spatio-Temporal Semantic Data Model for Precision Agriculture IoT Networks. *Agriculture 2023, Vol. 13, Page 360* 2023, *13*, 360, doi:10.3390/AGRICULTURE13020360.
47. Mezera, J.; Lukas, V.; Horniaček, I.; Smutný, V.; Elbl, J. Comparison of Proximal and Remote Sensing for the Diagnosis of Crop Status in Site-Specific Crop Management. *Sensors* 2022, *22*, 19, doi:10.3390/S22010019/S1.
48. Munnaf, M.A.; Haesaert, G.; Mouazen, A.M. Map-Based Site-Specific Seeding of Seed Potato Production by Fusion of Proximal and Remote Sensing Data. *Soil Tillage Res* 2021, *206*, 104801, doi:10.1016/J.STILL.2020.104801.
49. Skakun, S.; Kalecinski, N.I.; Brown, M.G.L.; Johnson, D.M.; Vermote, E.F.; Roger, J.C.; Franch, B. Assessing Within-Field Corn and Soybean Yield Variability from WorldView-3, Planet, Sentinel-2, and Landsat 8 Satellite Imagery. *Remote Sensing 2021, Vol. 13, Page 872* 2021, *13*, 872, doi:10.3390/RS13050872.
50. Munnaf, M.A.; Haesaert, G.; Mouazen, A.M. Site-Specific Seeding for Maize Production Using Management Zone Maps Delineated with Multi-Sensors Data Fusion Scheme. *Soil Tillage Res* 2022, *220*, 105377, doi:10.1016/J.STILL.2022.105377.
51. Mulla, D.J. Twenty Five Years of Remote Sensing in Precision Agriculture: Key Advances and Remaining Knowledge Gaps. *Biosyst Eng* 2013, *114*, 358–371, doi:10.1016/j.biosystemseng.2012.08.009.
52. Weiss, M.; Jacob, F.; Duveiller, G. Remote Sensing for Agricultural Applications: A Meta-Review. *Remote Sens Environ* 2020, *236*, 111402, doi:10.1016/J.RSE.2019.111402.
53. Campoy, J.; Campos, I.; Villodre, J.; Bodas, V.; Osann, A.; Calera, A. Remote Sensing-Based Crop Yield Model at Field and within-Field Scales in Wheat and Barley Crops. *European Journal of Agronomy* 2023, *143*, 126720, doi:10.1016/j.eja.2022.126720.
54. Marino, S. Understanding the Spatio-Temporal Behavior of Crop Yield, Yield Components and Weed Pressure Using Time Series Sentinel-2-Data in an Organic Farming System. *European Journal of Agronomy* 2023, *145*, 126785, doi:10.1016/j.eja.2023.126785.
55. Vizzari, M.; Santaga, F.; Benincasa, P. Sentinel 2-Based Nitrogen VRT Fertilization in Wheat: Comparison between Traditional and Simple Precision Practices. *Agronomy 2019, Vol. 9, Page 278* 2019, *9*, 278, doi:10.3390/AGRONOMY9060278.
56. Leo, S.; De Antoni Migliorati, M.; Nguyen, T.H.; Grace, P.R. Combining Remote Sensing-Derived Management Zones and an Auto-Calibrated Crop Simulation Model to Determine Optimal Nitrogen Fertilizer Rates. *Agric Syst* 2023, *205*, 103559, doi:10.1016/j.agsy.2022.103559.
57. Yuan, L.; Pu, R.; Zhang, J.; Wang, J.; Yang, H. Using High Spatial Resolution Satellite Imagery for Mapping Powdery Mildew at a Regional Scale. *Precis Agric* 2016, *17*, 332–348, doi:10.1007/S11119-015-9421-X/TABLES/4.
58. Khanal, S.; Kushal, K.C.; Fulton, J.P.; Shearer, S.; Ozkan, E. Remote Sensing in Agriculture—Accomplishments, Limitations, and Opportunities. *Remote Sensing 2020, Vol. 12, Page 3783* 2020, *12*, 3783, doi:10.3390/RS12223783.
59. Franke, J.; Menz, G. Multi-Temporal Wheat Disease Detection by Multi-Spectral Remote Sensing. *Precis Agric* 2007, *8*, 161–172, doi:10.1007/S11119-007-9036-Y/METRICS.
60. Dutta, A.; Tyagi, R.; Chattopadhyay, A.; Chatterjee, D.; Sarkar, A.; Lall, B.; Sharma, S. Early Detection of Wilt in Cajanus Cajan Using Satellite Hyperspectral Images: Development and Validation of Disease-Specific Spectral Index with Integrated Methodology. *Comput Electron Agric* 2024, *219*, 108784, doi:10.1016/J.COMPAG.2024.108784.
61. Messina, G.; Peña, J.M.; Vizzari, M.; Modica, G. A Comparison of UAV and Satellites Multispectral Imagery in Monitoring Onion Crop. An Application in the 'Cipolla Rossa Di Tropea' (Italy). *Remote Sensing 2020, Vol. 12, Page 3424* 2020, *12*, 3424, doi:10.3390/RS12203424.
62. Khan, S.; Tufail, M.; Khan, M.T.; Khan, Z.A.; Anwar, S. Deep Learning-Based Identification System of Weeds and Crops in Strawberry and Pea Fields for a Precision Agriculture Sprayer. *Precis Agric* 2021, *22*, 1711–1727, doi:10.1007/S11119-021-09808-9/TABLES/3.

63. Osorio, K.; Puerto, A.; Pedraza, C.; Jamaica, D.; Rodríguez, L. A Deep Learning Approach for Weed Detection in Lettuce Crops Using Multispectral Images. *AgriEngineering 2020, Vol. 2, Pages 471-488* 2020, *2*, 471–488, doi:10.3390/AGRIENGINEERING2030032.
64. Castrignanò, A.; Belmonte, A.; Antelmi, I.; Quarto, R.; Quarto, F.; Shaddad, S.; Sion, V.; Muolo, M.R.; Ranieri, N.A.; Gadaleta, G.; et al. Semi-Automatic Method for Early Detection of Xylella Fastidiosa in Olive Trees Using UAV Multispectral Imagery and Geostatistical-Discriminant Analysis. *Remote Sensing 2021, Vol. 13, Page 14* 2020, *13*, 14, doi:10.3390/RS13010014.
65. Guo, A.; Huang, W.; Dong, Y.; Ye, H.; Ma, H.; Liu, B.; Wu, W.; Ren, Y.; Ruan, C.; Geng, Y. Wheat Yellow Rust Detection Using UAV-Based Hyperspectral Technology. *Remote Sensing 2021, Vol. 13, Page 123* 2021, *13*, 123, doi:10.3390/RS13010123.
66. Abdulridha, J.; Ampatzidis, Y.; Qureshi, J.; Roberts, P. Laboratory and UAV-Based Identification and Classification of Tomato Yellow Leaf Curl, Bacterial Spot, and Target Spot Diseases in Tomato Utilizing Hyperspectral Imaging and Machine Learning. *Remote Sensing 2020, Vol. 12, Page 2732* 2020, *12*, 2732, doi:10.3390/RS12172732.
67. Campos, J.; Llop, J.; Gallart, M.; García-Ruiz, F.; Gras, A.; Salcedo, R.; Gil, E. Development of Canopy Vigour Maps Using UAV for Site-Specific Management during Vineyard Spraying Process. *Precis Agric* 2019, *20*, 1136–1156, doi:10.1007/S11119-019-09643-Z/TABLES/5.
68. Chia, M.Y.; Huang, Y.F.; Koo, C.H.; Fung, K.F. On-Farm Evaluation of Prescription Map-Based Variable Rate Application of Pesticides in Vineyards. *Agronomy 2020, Vol. 10, Page 102* 2020, *10*, 102, doi:10.3390/AGRONOMY10010102.
69. Pérez-Ortiz, M.; Peña, J.M.; Gutiérrez, P.A.; Torres-Sánchez, J.; Hervás-Martínez, C.; López-Granados, F. A Semi-Supervised System for Weed Mapping in Sunflower Crops Using Unmanned Aerial Vehicles and a Crop Row Detection Method. *Appl Soft Comput* 2015, *37*, 533–544, doi:10.1016/J.ASOC.2015.08.027.
70. de Castro, A.I.; Torres-Sánchez, J.; Peña, J.M.; Jiménez-Brenes, F.M.; Csillik, O.; López-Granados, F. An Automatic Random Forest-OBIA Algorithm for Early Weed Mapping between and within Crop Rows Using UAV Imagery. *Remote Sensing 2018, Vol. 10, Page 285* 2018, *10*, 285, doi:10.3390/RS10020285.
71. Jiang, J.; Wu, Y.; Liu, Q.; Liu, Y.; Cao, Q.; Tian, Y.; Zhu, Y.; Cao, W.; Liu, X. Developing an Efficiency and Energy-Saving Nitrogen Management Strategy for Winter Wheat Based on the UAV Multispectral Imagery and Machine Learning Algorithm. *Precis Agric* 2023, *24*, 2019–2043, doi:10.1007/S11119-023-10028-6/FIGURES/6.
72. Nevavuori, P.; Narra, N.; Linna, P.; Lipping, T. Crop Yield Prediction Using Multitemporal UAV Data and Spatio-Temporal Deep Learning Models. *Remote Sensing 2020, Vol. 12, Page 4000* 2020, *12*, 4000, doi:10.3390/RS12234000.
73. Garcia-Ruiz, F.; Campos, J.; Llop-Casamada, J.; Gil, E. Assessment of Map Based Variable Rate Strategies for Copper Reduction in Hedge Vineyards. *Comput Electron Agric* 2023, *207*, 107753, doi:10.1016/J.COMPAG.2023.107753.
74. Munnaf, M.A.; Mouazen, A.M. Optimising Site-Specific Potato Seeding Rates for Maximum Yield and Profitability. *Biosyst Eng* 2021, *212*, 126–140, doi:10.1016/J.BIOSYSTEMSENG.2021.10.006.
75. Shahi, T.B.; Xu, C.Y.; Neupane, A.; Guo, W. Recent Advances in Crop Disease Detection Using UAV and Deep Learning Techniques. *Remote Sensing 2023, Vol. 15, Page 2450* 2023, *15*, 2450, doi:10.3390/RS15092450.
76. Kerkech, M.; Hafiane, A.; Canals, R. Vine Disease Detection in UAV Multispectral Images Using Optimized Image Registration and Deep Learning Segmentation Approach. *Comput Electron Agric* 2020, *174*, 105446, doi:10.1016/J.COMPAG.2020.105446.
77. Tsouros, D.C.; Bibi, S.; Sarigiannidis, P.G. A Review on UAV-Based Applications for Precision Agriculture. *Information 2019, Vol. 10, Page 349* 2019, *10*, 349, doi:10.3390/INFO10110349.
78. Munnaf, M.A.; Wang, Y.; Mouazen, A.M. Robot Driven Combined Site-Specific Maize Seeding and N Fertilization: An Agro-Economic Investigation. *Comput Electron Agric* 2024, *219*, 108761, doi:10.1016/J.COMPAG.2024.108761.
79. Herrmann, I.; Berger, K. Remote and Proximal Assessment of Plant Traits. *Remote Sensing 2021, Vol. 13, Page 1893* 2021, *13*, 1893, doi:10.3390/RS13101893.
80. Anastasiou, E.; Balafoutis, A.; Darra, N.; Psiroukis, V.; Biniari, A.; Xanthopoulos, G.; Fountas, S. Satellite and Proximal Sensing to Estimate the Yield and Quality of Table Grapes. *Agriculture 2018, Vol. 8, Page 94* 2018, *8*, 94, doi:10.3390/AGRICULTURE8070094.
81. Stafford, J. V. Implementing Precision Agriculture in the 21st Century. *Journal of Agricultural Engineering Research* 2000, *76*, 267–275, doi:10.1006/jaer.2000.0577.

82. Zahir, S.A.D.M.; Omar, A.F.; Jamlos, M.F.; Azmi, M.A.M.; Muncan, J. A Review of Visible and Near-Infrared (Vis-NIR) Spectroscopy Application in Plant Stress Detection. *Sens Actuators A Phys* 2022, *338*.
83. Farber, C.; Mahnke, M.; Sanchez, L.; Kurouski, D. Advanced Spectroscopic Techniques for Plant Disease Diagnostics. A Review. *TrAC - Trends in Analytical Chemistry* 2019, *118*, 43–49.
84. Sankaran, S.; Mishra, A.; Ehsani, R.; Davis, C. A Review of Advanced Techniques for Detecting Plant Diseases. *Comput Electron Agric* 2010, *72*, 1–13.
85. Barbedo, J.G.A. Detection of Nutrition Deficiencies in Plants Using Proximal Images and Machine Learning: A Review. *Comput Electron Agric* 2019, *162*, 482–492, doi:10.1016/J.COMPAG.2019.04.035.
86. Padilla, F.M.; de Souza, R.; Peña-Fleitas, M.T.; Grasso, R.; Gallardo, M.; Thompson, R.B. Influence of Time of Day on Measurement with Chlorophyll Meters and Canopy Reflectance Sensors of Different Crop N Status. *Precis Agric* 2019, *20*, 1087–1106, doi:10.1007/s11119-019-09641-1.
87. Holland, K.; Schepers, J.; … J.S.-P. of the 7th; 2004, undefined Plant Canopy Sensor with Modulated Polychromatic Light Source. *cabdirect.orgKH Holland, JS Schepers, JF Shanahan, GL HorstProceedings of the 7th International Conference on Precision Agriculture, 2004•cabdirect.org*.
88. Bijay-Singh; Ali, A.M. Using Hand-Held Chlorophyll Meters and Canopy Reflectance Sensors for Fertilizer Nitrogen Management in Cereals in Small Farms in Developing Countries. *Sensors 2020, Vol. 20, Page 1127* 2020, *20*, 1127, doi:10.3390/S20041127.
89. Cao, Q.; Miao, Y.; Wang, H.; Huang, S.; Cheng, S.; Khosla, R.; Jiang, R. Non-Destructive Estimation of Rice Plant Nitrogen Status with Crop Circle Multispectral Active Canopy Sensor. *Field Crops Res* 2013, *154*, 133–144, doi:10.1016/J.FCR.2013.08.005.
90. Cao, Q.; Miao, Y.; Li, F.; Gao, X.; Liu, B.; Lu, D.; Chen, X. Developing a New Crop Circle Active Canopy Sensor-Based Precision Nitrogen Management Strategy for Winter Wheat in North China Plain. *Precis Agric* 2017, *18*, 2–18, doi:10.1007/S11119-016-9456-7/FIGURES/3.
91. Lu, J.; Miao, Y.; Shi, W.; Li, J.; Hu, X.; Chen, Z.; Wang, X.; Kusnierek, K. Developing a Proximal Active Canopy Sensor-Based Precision Nitrogen Management Strategy for High-Yielding Rice. *Remote Sensing 2020, Vol. 12, Page 1440* 2020, *12*, 1440, doi:10.3390/RS12091440.
92. Wang, X.; Miao, Y.; Dong, R.; Chen, Z.; Guan, Y.; Yue, X.; Fang, Z.; Mulla, D.J. Developing Active Canopy Sensor-Based Precision Nitrogen Management Strategies for Maize in Northeast China. *Sustainability 2019, Vol. 11, Page 706* 2019, *11*, 706, doi:10.3390/SU11030706.
93. Vennam, R.R.; Bheemanahalli, R.; Reddy, K.R.; Dhillon, J.; Zhang, X.; Adeli, A. Early-Season Maize Responses to Salt Stress: Morpho-Physiological, Leaf Reflectance, and Mineral Composition. *J Agric Food Res* 2024, *15*, 100994, doi:10.1016/J.JAFR.2024.100994.
94. Cotrozzi, L.; Couture, J.J. Hyperspectral Assessment of Plant Responses to Multi-Stress Environments: Prospects for Managing Protected Agrosystems. *Plants, People, Planet* 2020, *2*, 244–258, doi:10.1002/PPP3.10080.
95. Rubo, S.; Zinkernagel, J. Exploring Hyperspectral Reflectance Indices for the Estimation of Water and Nitrogen Status of Spinach. *Biosyst Eng* 2022, *214*, 58–71, doi:10.1016/J.BIOSYSTEMSENG.2021.12.008.
96. Gold, K.M.; Townsend, P.A.; Chlus, A.; Herrmann, I.; Couture, J.J.; Larson, E.R.; Gevens, A.J. Hyperspectral Measurements Enable Pre-Symptomatic Detection and Differentiation of Contrasting Physiological Effects of Late Blight and Early Blight in Potato. *Remote Sens (Basel)* 2020, *12*, doi:10.3390/rs12020286.
97. Chen, T.; Zeng, R.; Guo, W.; Hou, X.; Lan, Y.; Zhang, L. Detection of Stress in Cotton (Gossypium Hirsutum L.) Caused by Aphids Using Leaf Level Hyperspectral Measurements. *Sensors (Switzerland)* 2018, *18*, doi:10.3390/s18092798.
98. Herrmann, I.; Vosberg, S.K.; Ravindran, P.; Singh, A.; Chang, H.X.; Chilvers, M.I.; Conley, S.P.; Townsend, P.A. Leaf and Canopy Level Detection of Fusarium Virguliforme (Sudden Death Syndrome) in Soybean. *Remote Sens (Basel)* 2018, *10*, doi:10.3390/rs10030426.
99. Xu, H.R.; Ying, Y.B.; Fu, X.P.; Zhu, S.P. Near-Infrared Spectroscopy in Detecting Leaf Miner Damage on Tomato Leaf. *Biosyst Eng* 2007, *96*, 447–454, doi:10.1016/j.biosystemseng.2007.01.008.
100. Sankaran, S.; Ehsani, R. *Comparison of Visible-near Infrared and Mid-Infrared Spectroscopy for Classification of Huanglongbing and Citrus Canker Infected Leaves*; 2013; Vol. 15;.
101. Berdugo, C.A.; Zito, R.; Paulus, S.; Mahlein, A.K. Fusion of Sensor Data for the Detection and Differentiation of Plant Diseases in Cucumber. *Plant Pathol* 2014, *63*, 1344–1356, doi:10.1111/ppa.12219.
102. Mahlein, A.K.; Steiner, U.; Dehne, H.W.; Oerke, E.C. Spectral Signatures of Sugar Beet Leaves for the Detection and Differentiation of Diseases. *Precis Agric* 2010, *11*, 413–431, doi:10.1007/s11119-010-9180-7.

103. Rumpf, T.; Mahlein, A.K.; Steiner, U.; Oerke, E.C.; Dehne, H.W.; Plümer, L. Early Detection and Classification of Plant Diseases with Support Vector Machines Based on Hyperspectral Reflectance. *Comput Electron Agric* 2010, *74*, 91–99, doi:10.1016/j.compag.2010.06.009.
104. Hillnhütter, C.; Mahlein, A.K.; Sikora, R.A.; Oerke, E.C. Use of Imaging Spectroscopy to Discriminate Symptoms Caused by Heterodera Schachtii and Rhizoctonia Solani on Sugar Beet. *Precis Agric* 2012, *13*, 17–32, doi:10.1007/s11119-011-9237-2.
105. Mahlein, A.K.; Rumpf, T.; Welke, P.; Dehne, H.W.; Plümer, L.; Steiner, U.; Oerke, E.C. Development of Spectral Indices for Detecting and Identifying Plant Diseases. *Remote Sens Environ* 2013, *128*, 21–30, doi:10.1016/j.rse.2012.09.019.
106. Huang, W.; Guan, Q.; Luo, J.; Zhang, J.; Zhao, J.; Liang, D.; Huang, L.; Zhang, D. New Optimized Spectral Indices for Identifying and Monitoring Winter Wheat Diseases. *IEEE J Sel Top Appl Earth Obs Remote Sens* 2014, *7*, 2516–2524, doi:10.1109/JSTARS.2013.2294961.
107. Lu, J.; Ehsani, R.; Shi, Y.; de Castro, A.I.; Wang, S. Detection of Multi-Tomato Leaf Diseases (Late Blight, Target and Bacterial Spots) in Different Stages by Using a Spectral-Based Sensor. *Sci Rep* 2018, *8*, doi:10.1038/s41598-018-21191-6.
108. Qiao, L.; Gao, D.; Zhang, J.; Li, M.; Sun, H.; Ma, J. Dynamic Influence Elimination and Chlorophyll Content Diagnosis of Maize Using UAV Spectral Imagery. *Remote Sensing 2020, Vol. 12, Page 2650* 2020, *12*, 2650, doi:10.3390/RS12162650.
109. Qiu, Z.; Ma, F.; Li, Z.; Xu, X.; Ge, H.; Du, C. Estimation of Nitrogen Nutrition Index in Rice from UAV RGB Images Coupled with Machine Learning Algorithms. *Comput Electron Agric* 2021, *189*, 106421, doi:10.1016/J.COMPAG.2021.106421.
110. Gitelson, A.A.; Merzlyak, M.N.; Chivkunova, O.B. Optical Properties and Nondestructive Estimation of Anthocyanin Content in Plant Leaves¶. *Photochem Photobiol* 2001, *74*, 38, doi:10.1562/0031-8655(2001)074<0038:opaneo>2.0.co;2.
111. Rigon, J.P.G.; Capuani, S.; Fernandes, D.M.; Guimarães, T.M. A Novel Method for the Estimation of Soybean Chlorophyll Content Using a Smartphone and Image Analysis. *Photosynthetica* 2016, *54*, 559–566, doi:10.1007/S11099-016-0214-X/METRICS.
112. Esau, T.; Zaman, Q.; Groulx, D.; Farooque, A.; Schumann, A.; Chang, Y. Machine Vision Smart Sprayer for Spot-Application of Agrochemical in Wild Blueberry Fields. *Precis Agric* 2018, *19*, 770–788, doi:10.1007/S11119-017-9557-Y/TABLES/6.
113. Chattha, H.S.; Zaman, Q.U.; Chang, Y.K.; Read, S.; Schumann, A.W.; Brewster, G.R.; Farooque, A.A. Variable Rate Spreader for Real-Time Spot-Application of Granular Fertilizer in Wild Blueberry. *Comput Electron Agric* 2014, *100*, 70–78, doi:10.1016/J.COMPAG.2013.10.012.
114. Taneja, P.; Vasava, H.K.; Daggupati, P.; Biswas, A. Multi-Algorithm Comparison to Predict Soil Organic Matter and Soil Moisture Content from Cell Phone Images. *Geoderma* 2021, *385*, 114863, doi:10.1016/J.GEODERMA.2020.114863.
115. Swetha, R.K.; Bende, P.; Singh, K.; Gorthi, S.; Biswas, A.; Li, B.; Weindorf, D.C.; Chakraborty, S. Predicting Soil Texture from Smartphone-Captured Digital Images and an Application. *Geoderma* 2020, *376*, 114562, doi:10.1016/J.GEODERMA.2020.114562.
116. Asefpour Vakilian, K.; Massah, J. A Farmer-Assistant Robot for Nitrogen Fertilizing Management of Greenhouse Crops. *Comput Electron Agric* 2017, *139*, 153–163, doi:10.1016/j.compag.2017.05.012.
117. Heiß, A.; Paraforos, D.S.; Sharipov, G.M.; Griepentrog, H.W. Modeling and Simulation of a Multi-Parametric Fuzzy Expert System for Variable Rate Nitrogen Application. *Comput Electron Agric* 2021, *182*, 106008, doi:10.1016/J.COMPAG.2021.106008.
118. Maleki, M.R.; Mouazen, A.M.; De Ketelaere, B.; Ramon, H.; De Baerdemaeker, J. On-the-Go Variable-Rate Phosphorus Fertilisation Based on a Visible and near-Infrared Soil Sensor. *Biosyst Eng* 2008, *99*, 35–46, doi:10.1016/J.BIOSYSTEMSENG.2007.09.007.
119. Tewari, V.K.; Pareek, C.M.; Lal, G.; Dhruw, L.K.; Singh, N. Image Processing Based Real-Time Variable-Rate Chemical Spraying System for Disease Control in Paddy Crop. *Artificial Intelligence in Agriculture* 2020, *4*, 21–30, doi:10.1016/J.AIIA.2020.01.002.
120. Dammer, K.H.; Ehlert, D. Variable-Rate Fungicide Spraying in Cereals Using a Plant Cover Sensor. *Precis Agric* 2006, *7*, 137–148, doi:10.1007/S11119-006-9005-X/METRICS.
121. Mahlein, A.K.; Steiner, U.; Hillnhütter, C.; Dehne, H.W.; Oerke, E.C. Hyperspectral Imaging for Small-Scale Analysis of Symptoms Caused by Different Sugar Beet Diseases. *Plant Methods* 2012, *8*, 1–13, doi:10.1186/1746-4811-8-3/FIGURES/7.
122. Huang, J.; Wei, C.; Zhang, Y.; Blackburn, G.A.; Wang, X.; Wei, C.; Wang, J. Meta-Analysis of the Detection of Plant Pigment Concentrations Using Hyperspectral Remotely Sensed Data. *PLoS One* 2015, *10*, e0137029, doi:10.1371/JOURNAL.PONE.0137029.

123. Li, Z.; Jin, X.; Yang, G.; Drummond, J.; Yang, H.; Clark, B.; Li, Z.; Zhao, C. Remote Sensing of Leaf and Canopy Nitrogen Status in Winter Wheat (Triticum Aestivum L.) Based on N-PROSAIL Model. *Remote Sensing 2018, Vol. 10, Page 1463* 2018, *10*, 1463, doi:10.3390/RS10091463.
124. Wang, X.; Miao, Y.; Dong, R.; Zha, H.; Xia, T.; Chen, Z.; Kusnierek, K.; Mi, G.; Sun, H.; Li, M. Machine Learning-Based in-Season Nitrogen Status Diagnosis and Side-Dress Nitrogen Recommendation for Corn. *European Journal of Agronomy* 2021, *123*, 126193, doi:10.1016/J.EJA.2020.126193.
125. Gobbo, S.; De Antoni Migliorati, M.; Ferrise, R.; Morari, F.; Furlan, L.; Sartori, L. Can Crop Modelling, Proximal Sensing and Variable Rate Application Techniques Be Integrated to Support in-Season Nitrogen Fertilizer Decisions? An Application in Corn. *European Journal of Agronomy* 2023, *148*, 126854, doi:10.1016/J.EJA.2023.126854.
126. Moshou, D.; Bravo, C.; Oberti, R.; West, J.; Bodria, L.; McCartney, A.; Ramon, H. Plant Disease Detection Based on Data Fusion of Hyper-Spectral and Multi-Spectral Fluorescence Imaging Using Kohonen Maps. *Real-Time Imaging* 2005, *11*, 75–83, doi:10.1016/j.rti.2005.03.003.
127. Zhang, J.; Pu, R.; Huang, W.; Yuan, L.; Luo, J.; Wang, J. Using In-Situ Hyperspectral Data for Detecting and Discriminating Yellow Rust Disease from Nutrient Stresses. *Field Crops Res* 2012, *134*, 165–174, doi:10.1016/j.fcr.2012.05.011.
128. Oberti, R.; Marchi, M.; Tirelli, P.; Calcante, A.; Iriti, M.; Tona, E.; Hočevar, M.; Baur, J.; Pfaff, J.; Schütz, C.; et al. Selective Spraying of Grapevines for Disease Control Using a Modular Agricultural Robot. *Biosyst Eng* 2016, *146*, 203–215, doi:10.1016/J.BIOSYSTEMSENG.2015.12.004.
129. Schor, N.; Bechar, A.; Ignat, T.; Dombrovsky, A.; Elad, Y.; Berman, S. Robotic Disease Detection in Greenhouses: Combined Detection of Powdery Mildew and Tomato Spotted Wilt Virus. *IEEE Robot Autom Lett* 2016, *1*, 354–360, doi:10.1109/LRA.2016.2518214.
130. Moshou, D.; Bravo, C.; Oberti, R.; West, J.S.; Ramon, H.; Vougioukas, S.; Bochtis, D. Intelligent Multi-Sensor System for the Detection and Treatment of Fungal Diseases in Arable Crops. *Biosyst Eng* 2011, *108*, 311–321, doi:10.1016/J.BIOSYSTEMSENG.2011.01.003.
131. Aasen, H.; Burkart, A.; Bolten, A.; Bareth, G. Generating 3D Hyperspectral Information with Lightweight UAV Snapshot Cameras for Vegetation Monitoring: From Camera Calibration to Quality Assurance. *ISPRS Journal of Photogrammetry and Remote Sensing* 2015, *108*, 245–259, doi:10.1016/J.ISPRSJPRS.2015.08.002.
132. Bienkowski, D.; Aitkenhead, M.J.; Lees, A.K.; Gallagher, C.; Neilson, R. Detection and Differentiation between Potato (Solanum Tuberosum) Diseases Using Calibration Models Trained with Non-Imaging Spectrometry Data. *Comput Electron Agric* 2019, *167*, doi:10.1016/j.compag.2019.105056.
133. Menesatti, P.; Antonucci, F.; Pallottino, F.; Giorgi, S.; Matere, A.; Nocente, F.; Pasquini, M.; D'Egidio, M.G.; Costa, C. Laboratory vs. in-Field Spectral Proximal Sensing for Early Detection of Fusarium Head Blight Infection in Durum Wheat. *Biosyst Eng* 2013, *114*, 289–293, doi:10.1016/j.biosystemseng.2013.01.004.
134. Thomas, S.; Kuska, M.T.; Bohnenkamp, D.; Brugger, A.; Alisaac, E.; Wahabzada, M.; Behmann, J.; Mahlein, A.K. Benefits of Hyperspectral Imaging for Plant Disease Detection and Plant Protection: A Technical Perspective. *Journal of Plant Diseases and Protection* 2018, *125*, 5–20.
135. Galieni, A.; Nicastro, N.; Pentangelo, A.; Platani, C.; Cardi, T.; Pane, C. Surveying Soil-Borne Disease Development on Wild Rocket Salad Crop by Proximal Sensing Based on High-Resolution Hyperspectral Features. *Sci Rep* 2022, *12*, doi:10.1038/s41598-022-08969-5.
136. Munnaf, M.A.; Haesaert, G.; Mouazen, A.M. Site-Specific Seeding for Maize Production Using Management Zone Maps Delineated with Multi-Sensors Data Fusion Scheme. *Soil Tillage Res* 2022, *220*, 105377, doi:10.1016/J.STILL.2022.105377.
137. Castaldi, F.; Pelosi, F.; Pascucci, S.; Casa, R. Assessing the Potential of Images from Unmanned Aerial Vehicles (UAV) to Support Herbicide Patch Spraying in Maize. *Precis Agric* 2017, *18*, 76–94, doi:10.1007/S11119-016-9468-3/FIGURES/7.
138. Esau, T.J.; Zaman, Q.U.; Chang, Y.K.; Schumann, A.W.; Percival, D.C.; Farooque, A.A. Spot-Application of Fungicide for Wild Blueberry Using an Automated Prototype Variable Rate Sprayer. *Precis Agric* 2014, *15*, 147–161, doi:10.1007/S11119-013-9319-4/TABLES/5.
139. Lowenberg-DeBoer, J.; Huang, I.Y.; Grigoriadis, V.; Blackmore, S. Economics of Robots and Automation in Field Crop Production. *Precis Agric* 2020, *21*, 278–299, doi:10.1007/S11119-019-09667-5/TABLES/3.
140. Berenstein, R.; Edan, Y. Human-Robot Collaborative Site-Specific Sprayer. *J Field Robot* 2017, *34*, 1519–1530, doi:10.1002/ROB.21730.

141. Gerhards, R.; Oebel, H. Practical Experiences with a System for Site-Specific Weed Control in Arable Crops Using Real-Time Image Analysis and GPS-Controlled Patch Spraying. *Weed Res* 2006, *46*, 185–193, doi:10.1111/J.1365-3180.2006.00504.X.
142. Partel, V.; Charan Kakarla, S.; Ampatzidis, Y. Development and Evaluation of a Low-Cost and Smart Technology for Precision Weed Management Utilizing Artificial Intelligence. *Comput Electron Agric* 2019, *157*, 339–350, doi:10.1016/J.COMPAG.2018.12.048.
143. Liu, J.; Abbas, I.; Noor, R.S. Development of Deep Learning-Based Variable Rate Agrochemical Spraying System for Targeted Weeds Control in Strawberry Crop. *Agronomy 2021, Vol. 11, Page 1480* 2021, *11*, 1480, doi:10.3390/AGRONOMY11081480.
144. Ghafar, A.S.A.; Hajjaj, S.S.H.; Gsangaya, K.R.; Sultan, M.T.H.; Mail, M.F.; Hua, L.S. Design and Development of a Robot for Spraying Fertilizers and Pesticides for Agriculture. *Mater Today Proc* 2023, *81*, 242–248, doi:10.1016/J.MATPR.2021.03.174.
145. Rizk, H.; Habib, M.K. Robotized Early Plant Health Monitoring System. *Proceedings: IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society* 2018, 3795–3800, doi:10.1109/IECON.2018.8592833.
146. Farooque, A.A.; Hussain, N.; Schumann, A.W.; Abbas, F.; Afzaal, H.; McKenzie-Gopsill, A.; Esau, T.; Zaman, Q.; Wang, X. Field Evaluation of a Deep Learning-Based Smart Variable-Rate Sprayer for Targeted Application of Agrochemicals. *Smart Agricultural Technology* 2023, *3*, 100073, doi:10.1016/J.ATECH.2022.100073.
147. Spaeth, M.; Sökefeld, M.; Schwaderer, P.; Gauer, M.E.; Sturm, D.J.; Delatrée, C.C.; Gerhards, R. Smart Sprayer a Technology for Site-Specific Herbicide Application. *Crop Protection* 2024, *177*, 106564, doi:10.1016/J.CROPRO.2023.106564.
148. Gai, J.; Tang, L.; Steward, B.L. Automated Crop Plant Detection Based on the Fusion of Color and Depth Images for Robotic Weed Control. *J Field Robot* 2020, *37*, 35–52, doi:10.1002/ROB.21897.
149. Baek, E.T.; Im, D.Y. ROS-Based Unmanned Mobile Robot Platform for Agriculture. *Applied Sciences (Switzerland)* 2022, *12*, doi:10.3390/APP12094335.
150. Mishra, P.; Polder, G.; Vilfan, N. Close Range Spectral Imaging for Disease Detection in Plants Using Autonomous Platforms: A Review on Recent Studies. *Current Robotics Reports 2020 1:2* 2020, *1*, 43–48, doi:10.1007/S43154-020-00004-7.
151. Gang, M.S.; Kim, H.J.; Kim, D.W. Estimation of Greenhouse Lettuce Growth Indices Based on a Two-Stage CNN Using RGB-D Images. *Sensors 2022, Vol. 22, Page 5499* 2022, *22*, 5499, doi:10.3390/S22155499.
152. Thrash, T.; Lee, H.; Baker, R.L. A Low-Cost High-Throughput Phenotyping System for Automatically Quantifying Foliar Area and Greenness. *Appl Plant Sci* 2022, *10*, e11502, doi:10.1002/APS3.11502.
153. Singh, G.; Slonecki, T.; Wadl, P.; Flessner, M.; Sosnoskie, L.; Hatterman-Valenti, H.; Gage, K.; Cutulle, M. Implementing Digital Multispectral 3D Scanning Technology for Rapid Assessment of Hemp (Cannabis Sativa L.) Weed Competitive Traits. *Remote Sensing 2024, Vol. 16, Page 2375* 2024, *16*, 2375, doi:10.3390/RS16132375.
154. Malounas, I.; Paliouras, G.; Nikolopoulos, D.; Liakopoulos, G.; Bresta, P.; Londra, P.; Katsileros, A.; Fountas, S. Early Detection of Broccoli Drought Acclimation/Stress in Agricultural Environments Utilizing Proximal Hyperspectral Imaging and AutoML. *Smart Agricultural Technology* 2024, *8*, 100463, doi:10.1016/J.ATECH.2024.100463.
155. Amitrano, C.; Junker, A.; D'Agostino, N.; De Pascale, S.; De Micco, V. Integration of High-Throughput Phenotyping with Anatomical Traits of Leaves to Help Understanding Lettuce Acclimation to a Changing Environment. *Planta* 2022, *256*, 1–19, doi:10.1007/S00425-022-03984-2/TABLES/4.
156. Tripodi, P.; Vincenzo, C.; Venezia, A.; Cocozza, A.; Pane, C. Precision Phenotyping of Wild Rocket (Diplotaxis Tenuifolia) to Determine Morpho-Physiological Responses under Increasing Drought Stress Levels Using the PlantEye Multispectral 3D System. *Horticulturae 2024, Vol. 10, Page 496* 2024, *10*, 496, doi:10.3390/HORTICULTURAE10050496.
157. Kohzuma, K.; Tamaki, M.; Hikosaka, K. Corrected Photochemical Reflectance Index (PRI) Is an Effective Tool for Detecting Environmental Stresses in Agricultural Crops under Light Conditions. *J Plant Res* 2021, *134*, 683–694, doi:10.1007/S10265-021-01316-1/FIGURES/4.
158. Polder, G.; Dieleman, J.A.; Hageraats, S.; Meinen, E. Imaging Spectroscopy for Monitoring the Crop Status of Tomato Plants. *Comput Electron Agric* 2024, *216*, 108504, doi:10.1016/J.COMPAG.2023.108504.
159. Susič, N.; Žibrat, U.; Širca, S.; Strajnar, P.; Razinger, J.; Knapič, M.; Vončina, A.; Urek, G.; Gerič Stare, B. Discrimination between Abiotic and Biotic Drought Stress in Tomatoes Using

Hyperspectral Imaging. *Sens Actuators B Chem* 2018, *273*, 842–852, doi:10.1016/J.SNB.2018.06.121.
160. Scutelnic, D.; Muradore, R.; Daffara, C. A Multispectral Camera in the VIS–NIR Equipped with Thermal Imaging and Environmental Sensors for Non Invasive Analysis in Precision Agriculture. *HardwareX* 2024, *20*, e00596, doi:10.1016/J.OHX.2024.E00596.
161. Laveglia, S.; Altieri, G. A Method for Multispectral Images Alignment at Different Heights on the Crop. *Lecture Notes in Civil Engineering* 2024, *458 LNCE*.
162. Fernández, C.I.; Leblon, B.; Wang, J.; Haddadi, A.; Wang, K. Detecting Infected Cucumber Plants with Close-Range Multispectral Imagery. *Remote Sensing 2021, Vol. 13, Page 2948* 2021, *13*, 2948, doi:10.3390/RS13152948.
163. Qin, J.; Monje, O.; Nugent, M.R.; Finn, J.R.; O'Rourke, A.E.; Wilson, K.D.; Fritsche, R.F.; Baek, I.; Chan, D.E.; Kim, M.S. A Hyperspectral Plant Health Monitoring System for Space Crop Production. *Front Plant Sci* 2023, *14*, 1133505, doi:10.3389/FPLS.2023.1133505/BIBTEX.
164. Rana, S.; Gerbino, S.; Crimaldi, M.; Cirillo, V.; Carillo, P.; Sarghini, F.; Maggio, A. Comprehensive Evaluation of Multispectral Image Registration Strategies in Heterogenous Agriculture Environment. *Journal of Imaging 2024, Vol. 10, Page 61* 2024, *10*, 61, doi:10.3390/JIMAGING10030061.
165. Wasonga, D.O.; Yaw, A.; Kleemola, J.; Alakukku, L.; Mäkelä, P.S.A. Red-Green-Blue and Multispectral Imaging as Potential Tools for Estimating Growth and Nutritional Performance of Cassava under Deficit Irrigation and Potassium Fertigation. *Remote Sensing 2021, Vol. 13, Page 598* 2021, *13*, 598, doi:10.3390/RS13040598.
166. Lee, H.; He, Y.; Isaac, M.E. Close-Range Imaging for Green Roofs: Feature Detection, Band Matching, and Image Registration for Mixed Plant Communities. *Geomatica* 2024, *76*, 100011, doi:10.1016/J.GEOMAT.2024.100011.
167. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int J Comput Vis* 2004, *60*, 91–110, doi:10.1023/B:VISI.0000029664.99615.94.
168. Mistry, D.; and, A.B.-G.J.-G.R.; 2017, undefined Comparison of Feature Detection and Matching Approaches: SIFT and SURF. *grdjournals.comD Mistry, A BanerjeeGRD Journals-Global Research and Development Journal for Engineering, 2017•grdjournals.com*.
169. Tareen, S.A.K.; Saleem, Z. A Comparative Analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. *2018 International Conference on Computing, Mathematics and Engineering Technologies: Invent, Innovate and Integrate for Socioeconomic Development, iCoMET 2018 - Proceedings* 2018, *2018-January*, 1–10, doi:10.1109/ICOMET.2018.8346440.
170. Fernández, C.I.; Haddadi, A.; Leblon, B.; Wang, J.; Wang, K. Comparison between Three Registration Methods in the Case of Non-Georeferenced Close Range of Multispectral Images. *Remote Sensing 2021, Vol. 13, Page 396* 2021, *13*, 396, doi:10.3390/RS13030396.
171. West, J.; Fitzpatrick, J.M.; Wang, M.Y.; Dawant, B.M.; Maurer, C.R.; Kessler, R.M.; Maciunas, R.J.; Barillot, C.; Lemoine, D.; Collignon, A.; et al. Comparison and Evaluation of Retrospective Intermodality Brain Image Registration Techniques. *J Comput Assist Tomogr* 1997, *21*, 554–566, doi:10.1097/00004728-199707000-00007.
172. D, R.; LI, S.; C, H.; DL, H.; MO, L.; DJ, H. Nonrigid Registration Using Free-Form Deformations: Application to Breast MR Images. *IEEE Trans Med Imaging* 1999, *18*, 712–721, doi:10.1109/42.796284.
173. Viola, P.; Wells, W.M. Alignment by Maximization of Mutual Information. *Int J Comput Vis* 1997, *24*, 137–154, doi:10.1023/A:1007958904918/METRICS.
174. Haber, E.; Modersitzki, J. Intensity Gradient Based Registration and Fusion of Multi-Modal Images. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 2006, *4191 LNCS-II*, 726–733, doi:10.1007/11866763_89.
175. Lowe, D.G. Object Recognition from Local Scale-Invariant Features. *Proceedings of the IEEE International Conference on Computer Vision* 1999, *2*, 1150–1157, doi:10.1109/ICCV.1999.790410.
176. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun ACM* 1981, *24*, 381–395, doi:10.1145/358669.358692;PAGE:STRING:ARTICLE/CHAPTER.
177. Sturm, P. Pinhole Camera Model. *Computer Vision* 2021, 983–986, doi:10.1007/978-3-030-63416-2_472.
178. Zhang, Z. A Flexible New Technique for Camera Calibration. *IEEE Trans Pattern Anal Mach Intell* 2000, *22*, 1330–1334, doi:10.1109/34.888718.
179. Longuet-higgins, H.C. A Computer Algorithm for Reconstructing a Scene from Two Projections. *Nature 1981 293:5828* 1981, *293*, 133–135, doi:10.1038/293133a0.

180. Memon, Q.; Khan, S. Camera Calibration and Three-Dimensional World Reconstruction of Stereo-Vision Using Neural Networks. *Int J Syst Sci* 2001, *32*, 1155–1159, doi:10.1080/00207720010024276.
181. Heikkila, J.; Silven, O. Four-Step Camera Calibration Procedure with Implicit Image Correction. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 1997, 1106–1112, doi:10.1109/CVPR.1997.609468.
182. Camera Calibration Toolbox for Matlab Available online: https://data.caltech.edu/records/jx9cx-fdh55 (accessed on 26 January 2025).
183. Zitová, B.; Flusser, J. Image Registration Methods: A Survey. *Image Vis Comput* 2003, *21*, 977–1000, doi:10.1016/S0262-8856(03)00137-9.
184. Geiger, A.; Moosmann, F.; Car, Ö.; Schuster, B. Automatic Camera and Range Sensor Calibration Using a Single Shot. *Proc IEEE Int Conf Robot Autom* 2012, 3936–3943, doi:10.1109/ICRA.2012.6224570.
185. Guizar-Sicairos, M.; Thurman, S.T.; Fienup, J.R. Efficient Subpixel Image Registration Algorithms. *Opt Lett* 2008, *33*, 156, doi:10.1364/OL.33.000156.
186. Birth, G.S.; McVey, G.R. Measuring the Color of Growing Turf with a Reflectance Spectrophotometer 1 . *Agron J* 1968, *60*, 640–643, doi:10.2134/AGRONJ1968.00021962006000060016X.
187. Gitelson, A.; B, M.M.-J. of P. and P.; 1994, undefined Quantitative Estimation of Chlorophyll-a Using Reflectance Spectra: Experiments with Autumn Chestnut and Maple Leaves. *ElsevierA Gitelson, MN MerzlyakJournal of Photochemistry and Photobiology B: Biology, 1994•Elsevier*.
188. Bayle, A.; Carlson, B.Z.; Thierion, V.; Isenmann, M.; Choler, P. Improved Mapping of Mountain Shrublands Using the Sentinel-2 Red-Edge Band. *Remote Sensing 2019, Vol. 11, Page 2807* 2019, *11*, 2807, doi:10.3390/RS11232807.
189. Gitelson, A.A.; Merzlyak, M.N.; Chivkunova, O.B. Optical Properties and Nondestructive Estimation of Anthocyanin Content in Plant Leaves¶. *Photochem Photobiol* 2007, *74*, 38–45, doi:10.1562/0031-8655(2001)0740038OPANEO2.0.CO2.
190. Gitelson, A.A.; Merzlyak, M.N. Signature Analysis of Leaf Reflectance Spectra: Algorithm Development for Remote Sensing of Chlorophyll. *J Plant Physiol* 1996, *148*, 494–500, doi:10.1016/S0176-1617(96)80284-7.
191. Penuelas, J.; Baret, F.; Photosynthetica, I.F.-; 1995, undefined Semi-Empirical Indices to Assess Carotenoids/Chlorophyll a Ratio from Leaf Spectral Reflectance. *researchgate.net*.
192. Wang, C.; Liu, S.; Wang, X.; Lan, X. Time Synchronization and Space Registration of Roadside LiDAR and Camera. *Electronics (Switzerland)* 2023, *12*, doi:10.3390/ELECTRONICS12030537.
193. Kim, C.; van Iersel, M.W. Image-Based Phenotyping to Estimate Anthocyanin Concentrations in Lettuce. *Front Plant Sci* 2023, *14*, 1155722, doi:10.3389/FPLS.2023.1155722/BIBTEX.
194. Ncama, K.; Sithole, N.J. The Effect of Nitrogen Fertilizer and Water Supply Levels on the Growth, Antioxidant Compounds, and Organic Acids of Baby Lettuce. *Agronomy 2022, Vol. 12, Page 614* 2022, *12*, 614, doi:10.3390/AGRONOMY12030614.
195. Paim, B.T.; Crizel, R.L.; Tatiane, S.J.; Rodrigues, V.R.; Rombaldi, C.V.; Galli, V. Mild Drought Stress Has Potential to Improve Lettuce Yield and Quality. *Sci Hortic* 2020, *272*, 109578, doi:10.1016/J.SCIENTA.2020.109578.
196. Zelinsky, A. Learning OpenCV—Computer Vision with the OpenCV Library. *IEEE Robot Autom Mag* 2009, *16*, 100, doi:10.1109/MRA.2009.933612.
197. Atkinson, N.J.; Urwin, P.E. The Interaction of Plant Biotic and Abiotic Stresses: From Genes to the Field. *J Exp Bot* 2012, *63*, 3523–3544, doi:10.1093/JXB/ERS100.
198. Altieri, G.; Genovese, F.; Tauriello, A.; Di Renzo, G.C. Models to Improve the Non-Destructive Analysis of Persimmon Fruit Properties by VIS/NIR Spectrometry. *J Sci Food Agric* 2017, *97*, 5302–5310, doi:10.1002/JSFA.8416.
199. Skendžić, S.; Novak, H.; Zovko, M.; Pajač Živković, I.; Lešić, V.; Maričević, M.; Lemić, D. Hyperspectral Canopy Reflectance and Machine Learning for Threshold-Based Classification of Aphid-Infested Winter Wheat. *Remote Sensing 2025, Vol. 17, Page 929* 2025, *17*, 929, doi:10.3390/RS17050929.
200. Sallam, N.M.A.; AbdElfatah, H.A.S.; Dawood, M.F.A.; Hassan, E.A.; Mohamed, M.S.; Khalil Bagy, H.M.M. Physiological and Histopathological Assessments of the Susceptibility of Different Tomato (Solanum Lycopersicum) Cultivars to Early Blight Disease. *European Journal of Plant Pathology 2021 160:3* 2021, *160*, 541–556, doi:10.1007/S10658-021-02263-2.
201. Sankaran, S.; Maja, J.M.; Buchanon, S.; Ehsani, R. Huanglongbing (Citrus Greening) Detection Using Visible, near Infrared and Thermal Imaging Techniques. *Sensors (Switzerland)* 2013, *13*, 2117–2130, doi:10.3390/s130202117.

202. Moshou, D.; Bravo, C.; Wahlen, S.; West, J.; McCartney, A.; De Baerdemaeker, J.; Ramon, H. Simultaneous Identification of Plant Stresses and Diseases in Arable Crops Using Proximal Optical Sensing and Self-Organising Maps. *Precis Agric* 2006, *7*, 149–164, doi:10.1007/s11119-006-9002-0.
203. Knipling, E.B. Physical and Physiological Basis for the Reflectance of Visible and Near-Infrared Radiation from Vegetation. *Remote Sens Environ* 1970, *1*, 155–159, doi:10.1016/S0034-4257(70)80021-9.
204. Martín-Tornero, E.; de Jorge Páscoa, R.N.M.; Espinosa-Mansilla, A.; Martín-Merás, I.D.; Lopes, J.A. Comparative Quantification of Chlorophyll and Polyphenol Levels in Grapevine Leaves Sampled from Different Geographical Locations. *Scientific Reports 2020 10:1* 2020, *10*, 6246-, doi:10.1038/s41598-020-63407-8.
205. Falcioni, R.; Gonçalves, J.V.F.; de Oliveira, K.M.; de Oliveira, C.A.; Reis, A.S.; Crusiol, L.G.T.; Furlanetto, R.H.; Antunes, W.C.; Cezar, E.; de Oliveira, R.B.; et al. Chemometric Analysis for the Prediction of Biochemical Compounds in Leaves Using UV-VIS-NIR-SWIR Hyperspectroscopy. *Plants 2023, Vol. 12, Page 3424* 2023, *12*, 3424, doi:10.3390/PLANTS12193424.
206. Jayapal, P.K.; Joshi, R.; Sathasivam, R.; Van Nguyen, B.; Faqeerzada, M.A.; Park, S.U.; Sandanam, D.; Cho, B.K. Non-Destructive Measurement of Total Phenolic Compounds in Arabidopsis under Various Stress Conditions. *Front Plant Sci* 2022, *13*, 982247, doi:10.3389/FPLS.2022.982247/BIBTEX.
207. Solorio-Ramírez, J.L.; Jiménez-Cruz, R.; Villuendas-Rey, Y.; Yáñez-Márquez, C. Random Forest Algorithm for the Classification of Spectral Data of Astronomical Objects. *Algorithms 2023, Vol. 16, Page 293* 2023, *16*, 293, doi:10.3390/A16060293.
208. Chen, Z.; Xue, X.; Wu, H.; Gao, H.; Wang, G.; Ni, G.; Cao, T. Visible/near-Infrared Hyperspectral Imaging Combined with Machine Learning for Identification of Ten Dalbergia Species. *Front Plant Sci* 2024, *15*, 1413215, doi:10.3389/FPLS.2024.1413215/BIBTEX.
209. Prechsl, U.E.; Mejia-Aguilar, A.; Cullinan, C.B. In Vivo Spectroscopy and Machine Learning for the Early Detection and Classification of Different Stresses in Apple Trees. *Scientific Reports 2023 13:1* 2023, *13*, 15857-, doi:10.1038/s41598-023-42428-z.
210. Anand, R.; Parray, R.A.; Mani, I.; Khura, T.K.; Kushwaha, H.; Sharma, B.B.; Sarkar, S.; Godara, S. Spectral Data Driven Machine Learning Classification Models for Real Time Leaf Spot Disease Detection in Brinjal Crops. *European Journal of Agronomy* 2024, *161*, 127384, doi:10.1016/J.EJA.2024.127384.
211. Wang, Y.; Sun, J.; Wu, Z.; Jia, Y.; Dai, C. Application of Non-Destructive Technology in Plant Disease Detection: Review. *Agriculture 2025, Vol. 15, Page 1670* 2025, *15*, 1670, doi:10.3390/AGRICULTURE15151670.
212. Xie, Y.; Plett, D.; Evans, M.; Garrard, T.; Butt, M.; Clarke, K.; Liu, H. Hyperspectral Imaging Detects Biological Stress of Wheat for Early Diagnosis of Crown Rot Disease. *Comput Electron Agric* 2024, *217*, doi:10.1016/J.COMPAG.2023.108571.
213. Choi, J.H.; Park, S.H.; Jung, D.H.; Park, Y.J.; Yang, J.S.; Park, J.E.; Lee, H.; Kim, S.M. Hyperspectral Imaging-Based Multiple Predicting Models for Functional Component Contents in Brassica Juncea. *Agriculture 2022, Vol. 12, Page 1515* 2022, *12*, 1515, doi:10.3390/AGRICULTURE12101515.
214. Yuan, Z.; Ye, Y.; Wei, L.; Yang, X.; Huang, C. Study on the Optimization of Hyperspectral Characteristic Bands Combined with Monitoring and Visualization of Pepper Leaf SPAD Value. *Sensors 2022, Vol. 22, Page 183* 2021, *22*, 183, doi:10.3390/S22010183.
215. ROS 2 Documentation — ROS 2 Documentation: Humble Documentation Available online: https://docs.ros.org/en/humble/index.html (accessed on 15 December 2025).
216. ROS: Home Available online: https://www.ros.org/ (accessed on 15 December 2025).
217. Introduction to Robot Operating System 2 (ROS 2) - MATLAB & Simulink Available online: https://it.mathworks.com/help/ros/gs/robot-operating-system-ros2-basic-concepts.html (accessed on 15 December 2025).
218. Hachem, M.; Borrell, A.M.; Sename, O.; Atoui, H.; Morato, M. ROS Implementation of Planning and Robust Control Strategies for Autonomous Vehicles. *Electronics 2023, Vol. 12, Page 3680* 2023, *12*, 3680, doi:10.3390/ELECTRONICS12173680.
219. Integration of Matlab-Based Controllers with ROS for Autonomous Agricultural Vehicles - Webthesis Available online: https://webthesis.biblio.polito.it/25533/ (accessed on 16 December 2025).
220. Jkk_dataset_01 - JKK Research Center Available online: https://jkk-research.github.io/dataset/jkk_dataset_01/ (accessed on 16 December 2025).
221. GitHub - Foxglove/Studio: Robotics Visualization and Debugging Available online: https://github.com/foxglove/studio (accessed on 16 December 2025).

222. Yan, Y.; Zhang, B.; Zhou, J.; Zhang, Y.; Liu, X. Real-Time Localization and Mapping Utilizing Multi-Sensor Fusion and Visual–IMU–Wheel Odometry for Agricultural Robots in Unstructured, Dynamic and GPS-Denied Greenhouse Environments. *Agronomy 2022, Vol. 12, Page 1740* 2022, *12*, 1740, doi:10.3390/AGRONOMY12081740.
223. Zhang, S.; Liu, T.; Li, X.; Cai, C.; Chang, C.; Xue, X. Key Technologies of Robotic Arms in Unmanned Greenhouse. *Agronomy 2025, Vol. 15, Page 2498* 2025, *15*, 2498, doi:10.3390/AGRONOMY15112498.
224. Roure, F.; Moreno, G.; Soler, M.; Faconti, D.; Serrano, D.; Astolfi, P.; Bardaro, G.; Gabrielli, A.; Bascetta, L.; Matteucci, M. GRAPE: Ground Robot for VineyArd Monitoring and ProtEction. *Advances in Intelligent Systems and Computing* 2018, *693*, 249–260, doi:10.1007/978-3-319-70833-1_21.
225. Aguiar, A.S.; Neves dos Santos, F.; Sobreira, H.; Boaventura-Cunha, J.; Sousa, A.J. Localization and Mapping on Agriculture Based on Point-Feature Extraction and Semiplanes Segmentation From 3D LiDAR Data. *Front Robot AI* 2022, *9*, 832165, doi:10.3389/FROBT.2022.832165/BIBTEX.
226. Jiang, S.; Wang, S.; Yi, Z.; Zhang, M.; Lv, X. Autonomous Navigation System of Greenhouse Mobile Robot Based on 3D Lidar and 2D Lidar SLAM. *Front Plant Sci* 2022, *13*, 815218, doi:10.3389/FPLS.2022.815218/BIBTEX.
227. Tan, H.; Zhao, X.; Zhai, C.; Fu, H.; Chen, L.; Yang, M. Design and Experiments with a SLAM System for Low-Density Canopy Environments in Greenhouses Based on an Improved Cartographer Framework. *Front Plant Sci* 2024, *15*, 1276799, doi:10.3389/FPLS.2024.1276799/BIBTEX.
228. Ni, C.; Cai, J.; Wang, P. A LiDAR SLAM and Visual-Servoing Fusion Approach to Inter-Zone Localization and Navigation in Multi-Span Greenhouses. *Agronomy 2025, Vol. 15, Page 2380* 2025, *15*, 2380, doi:10.3390/AGRONOMY15102380.
229. Islam, R.; Habibullah, H.; Hossain, T. AGRI-SLAM: A Real-Time Stereo Visual SLAM for Agricultural Environment. *Autonomous Robots 2023 47:6* 2023, *47*, 649–668, doi:10.1007/S10514-023-10110-Y.
230. Cañadas-Aránega, F.; Blanco-Claraco, J.L.; Moreno, J.C.; Rodriguez-Diaz, F. Multimodal Mobile Robotic Dataset for a Typical Mediterranean Greenhouse: The GREENBOT Dataset. *Sensors (Basel)* 2024, *24*, doi:10.3390/S24061874.
231. Navigation Toolbox Documentation Available online: https://it.mathworks.com/help/nav/index.html?s_tid=CRUX_lftnav (accessed on 22 December 2025).
232. SLAM - MATLAB & Simulink Available online: https://it.mathworks.com/help/nav/slam.html?lang=en (accessed on 22 December 2025).
233. Mapping - MATLAB & Simulink Available online: https://it.mathworks.com/help/nav/mapping.html?lang=en (accessed on 22 December 2025).
234. Code Generation and Deployment - MATLAB & Simulink Available online: https://it.mathworks.com/help/nav/code-generation-and-deployment.html?lang=en (accessed on 22 December 2025).
235. Walsh, D.B.; Bolda, M.P.; Goodhue, R.E.; Dreves, A.J.; Lee, J.; Bruck, D.J.; Walton, V.M.; O'Neal, S.D.; Zalom, F.G. Drosophila Suzukii (Diptera: Drosophilidae): Invasive Pest of Ripening Soft Fruit Expanding Its Geographic Range and Damage Potential. *J Integr Pest Manag* 2011, *2*, doi:10.1603/IPM10010.
236. Gutierrez, A.P.; Ponti, L.; Dalton, D.T. Analysis of the Invasiveness of Spotted Wing Drosophila (Drosophila Suzukii) in North America, Europe, and the Mediterranean Basin. *Biol Invasions* 2016, *18*, 3647–3663, doi:10.1007/S10530-016-1255-6.
237. Lee, J.C.; Bruck, D.J.; Dreves, A.J.; Ioriatti, C.; Vogt, H.; Baufeld, P. In Focus: Spotted Wing Drosophila, Drosophila Suzukii, across Perspectives. *Pest Manag Sci* 2011, *67*, 1349–1351, doi:10.1002/PS.2271.
238. Ioriatti, C.; Walton, V.; Dalton, D.; Anfora, G.; Grassi, A.; Maistri, S.; Mazzoni, V. Drosophila Suzukii (Diptera: Drosophilidae) and Its Potential Impact to Wine Grapes During Harvest in Two Cool Climate Wine Grape Production Regions. *J Econ Entomol* 2015, *108*, 1148–1155, doi:10.1093/JEE/TOV042.
239. Beers, E.H.; Van Steenwyk, R.A.; Shearer, P.W.; Coates, W.W.; Grant, J.A. Developing Drosophila Suzukii Management Programs for Sweet Cherry in the Western United States. *Pest Manag Sci* 2011, *67*, 1386–1395, doi:10.1002/PS.2279.
240. Kenis, M.; Tonina, L.; Eschen, R.; van der Sluis, B.; Sancassani, M.; Mori, N.; Haye, T.; Helsen, H. Non-Crop Plants Used as Hosts by Drosophila Suzukii in Europe. *J Pest Sci (2004)* 2016, *89*, 735–748, doi:10.1007/S10340-016-0755-6/TABLES/3.
241. Cini, A.; Ioriatti, C.; Anfora, G. A Review of the Invasion of Drosophila Suzukii in Europe and a Draft Research Agenda for Integrated Pest Management. *Bull Insectology* 2012, *65*, 149–160.

242. Hamby, K.A.; Hernández, A.; Boundy-Mills, K.; Zalom, F.G. Associations of Yeasts with Spotted-Wing Drosophila (Drosophila Suzukii; Diptera: Drosophilidae) in Cherries and Raspberries. *Appl Environ Microbiol* 2012, *78*, 4869–4873, doi:10.1128/AEM.00841-12/SUPPL_FILE/AEM00841-12_SUPPLEMENTAL.PDF.
243. Ros, G. De; Anfora, G.; Grassi, A.; Bull, C.I.-I.-W.; 2013, undefined The Potential Economic Impact of Drosophila Suzukii on Small Fruits Production in Trentino (Italy). *researchgate.netG De Ros, G Anfora, A Grassi, C IoriattiIOBC-WPRS Bull, 2013•researchgate.net* 2012.
244. Ekramirad, N.; Adedeji, A.; Research, R.A.-I. in F.; 2016, undefined A Review of Non-Destructive Methods for Detection of Insect Infestation in Fruits and Vegetables. *core.ac.ukN Ekramirad, AA Adedeji, R AlimardaniInnovations in Food Research, 2016•core.ac.uk* 2016, *2*, 6–12.
245. Jamshidi, B.; Mohajerani, E.; Farazmand, H.; Mahmoudi, A.; Hemmati, A. Pattern Recognition-Based Optical Technique for Non-Destructive Detection of Ectomyelois Ceratoniae Infestation in Pomegranates during Hidden Activity of the Larvae. *Spectrochim Acta A Mol Biomol Spectrosc* 2019, *206*, 552–557, doi:10.1016/J.SAA.2018.08.059.
246. Mostafa, M.; Amor, A.I.; Admane, N.; Anfora, G.; Bubici, G.; Verrastro, V.; Scarano, L.; Moujabber, M. El; Baser, N. Reduction of Post-Harvest Injuries Caused by Drosophila Suzukii in Some Cultivars of Sweet Cherries Using a High Carbon Dioxide Level and Cold Storage. *Insects* 2021, *12*, doi:10.3390/INSECTS12111009.
247. Lee, J.-E.; Kim, M.-J.; Lee, B.-Y.; Hwan, L.J.; Yang, H.-E.; Kim, M.S.; Hwang, I.G.; Jeong, C.S.; Mo, C. Evaluating Ripeness in Post-Harvest Stored Kiwifruit Using VIS-NIR Hyperspectral Imaging. *Postharvest Biol Technol* 2025, *225*, 113496, doi:10.1016/j.postharvbio.2025.113496.
248. Prasad, K.; Jacob, S.; Siddiqui, M.W. Fruit Maturity, Harvesting, and Quality Standards. *Preharvest Modulation of Postharvest Fruit and Vegetable Quality* 2018, 41–69, doi:10.1016/B978-0-12-809807-3.00002-0.
249. Gupta, A.K.; Koch, P.; Yumnam, M.; Medhi, M.; Madufor, N.J.; Opara, U.L.; Mishra, P. Biosensors Involved in Fruit and Vegetable Processing Industries. *Biosensors in Food Safety and Quality: Fundamentals and Applications* 2022, 111–134, doi:10.1201/9780429259890-8/BIOSENSORS-INVOLVED-FRUIT-VEGETABLE-PROCESSING-INDUSTRIES-ARUN-KUMAR-GUPTA-PARISMITA-KOCH-MONICA-YUMNAM-MANISHA-MEDHI-MADUFOR-OPARA-POONAM-MISHRA.
250. Saranwong, S.; Haff, R.P.; Thanapase, W.; Janhiran, A.; Kasemsumran, S.; Kawano, S. Short Communication A Feasibility Study Using Simplified near Infrared Imaging to Detect Fruit Fly Larvae in Intact Fruit. *J Near Infrared Spectrosc* 2011, *19*, 55–60, doi:10.1255/JNIRS.915.
251. Pandiselvam, R.; Prithviraj, V.; Manikantan, M.R.; Kothakota, A.; Rusu, A.V.; Trif, M.; Mousavi Khaneghah, A. Recent Advancements in NIR Spectroscopy for Assessing the Quality and Safety of Horticultural Products: A Comprehensive Review. *Front Nutr* 2022, *9*, 973457, doi:10.3389/FNUT.2022.973457/BIBTEX.
252. Cen, H.; He, Y. Theory and Application of near Infrared Reflectance Spectroscopy in Determination of Food Quality. *Trends Food Sci Technol* 2007, *18*, 72–83, doi:10.1016/J.TIFS.2006.09.003.
253. Xia, Y.; Huang, W.; Fan, S.; Li, J.; Chen, L. Effect of Spectral Measurement Orientation on Online Prediction of Soluble Solids Content of Apple Using Vis/NIR Diffuse Reflectance. *Infrared Phys Technol* 2019, *97*, 467–477, doi:10.1016/J.INFRARED.2019.01.012.
254. Alander, J.T.; Bochko, V.; Martinkauppi, B.; Saranwong, S.; Mantere, T. A Review of Optical Nondestructive Visual and Near-Infrared Methods for Food Quality and Safety. *Int J Spectrosc* 2013, *2013*, 1–36, doi:10.1155/2013/341402.
255. Matera, A.; Altieri, G.; Genovese, F.; Di Renzo, G.C. Improved Spectrophotometric Models and Methods for the Non-Destructive and Effective Foodstuff Parameters Forecasting. *Acta Hortic* 2021, 395–402, doi:10.17660/ActaHortic.2021.1311.50.
256. Nicolaï, B.M.; Beullens, K.; Bobelyn, E.; Peirs, A.; Saeys, W.; Theron, K.I.; Lammertyn, J. Nondestructive Measurement of Fruit and Vegetable Quality by Means of NIR Spectroscopy: A Review. *Postharvest Biol Technol* 2007, *46*, 99–118, doi:10.1016/j.postharvbio.2007.06.024.
257. Qu, J.-H.; Liu, D.; Cheng, J.-H.; Sun, D.-W.; Ma, J.; Pu, H.; Zeng, X.-A. Applications of Near-Infrared Spectroscopy in Food Safety Evaluation and Control: A Review of Recent Research Advances. *Crit Rev Food Sci Nutr* 2015, *55*, 1939–1954, doi:10.1080/10408398.2013.871693.
258. Pu, Y.-Y.; O'Donnell, C.; Tobin, J.T.; O'Shea, N. Review of Near-Infrared Spectroscopy as a Process Analytical Technology for Real-Time Product Monitoring in Dairy Processing. *Int Dairy J* 2020, *103*, 104623, doi:10.1016/j.idairyj.2019.104623.
259. Hu, R.; Zhang, L.; Yu, Z.; Zhai, Z.; Technology, R.Z.-I.P.&; 2019, undefined Optimization of Soluble Solids Content Prediction Models in 'Hami'melons by Means of Vis-NIR Spectroscopy and Chemometric Tools. *Elsevier*.

260. Amoriello, T.; Ciorba, R.; Ruggiero, G.; Masciola, F.; Scutaru, D.; Ciccoritti, R. Vis/NIR Spectroscopy and Vis/NIR Hyperspectral Imaging for Non-Destructive Monitoring of Apricot Fruit Internal Quality with Machine Learning. *Foods* 2025, *14*, 196, doi:10.3390/FOODS14020196/S1.
261. Fatchurrahman, D.; Nosrati, M.; Amodio, M.L.; Chaudhry, M.M.A.; de Chiara, M.L.V.; Mastrandrea, L.; Colelli, G. Comparison Performance of Visible-NIR and Near-Infrared Hyperspectral Imaging for Prediction of Nutritional Quality of Goji Berry (Lycium Barbarum L.). *Foods 2021, Vol. 10, Page 1676* 2021, *10*, 1676, doi:10.3390/FOODS10071676.
262. Qiao, Y.; Wang, C.; Zhu, W.; Sun, L.; Bai, J.; Zhou, R.; Zhu, Z.; Cai, J. Online Assessment of Soluble Solids Content in Strawberries Using a Developed Vis/NIR Spectroscopy System with a Hanging Grasper. *Food Chem* 2025, *478*, 143671, doi:10.1016/J.FOODCHEM.2025.143671.
263. Xia, Y.; Huang, W.; Fan, S.; Li, J.; Technology, L.C.-I.P.&; 2019, undefined Effect of Spectral Measurement Orientation on Online Prediction of Soluble Solids Content of Apple Using Vis/NIR Diffuse Reflectance. *Elsevier*.
264. Xiao, Y.; Li, C.; Jin, C.; Luo, J.; Qi, H.; Zhang, C. Detection of Soluble Solid Content in Citrus Fruit Using Near-Infrared Spectroscopy with Machine Learning Regression: An Exploration of the Influence of Sampling Positions. *Journal of Food Composition and Analysis* 2025, *142*, 107554, doi:10.1016/J.JFCA.2025.107554.
265. Hu, W.; Sun, D.-W.; Blasco, J. Rapid Monitoring 1-MCP-Induced Modulation of Sugars Accumulation in Ripening 'Hayward' Kiwifruit by Vis/NIR Hyperspectral Imaging. *Postharvest Biol Technol* 2017, *125*, 168–180, doi:10.1016/j.postharvbio.2016.11.001.
266. Ciccoritti, R.; Paliotta, M.; Amoriello, T.; Carbone, K. FT-NIR Spectroscopy and Multivariate Classification Strategies for the Postharvest Quality of Green-Fleshed Kiwifruit Varieties. *Sci Hortic* 2019, *257*, 108622, doi:10.1016/j.scienta.2019.108622.
267. Benelli, A.; Cevoli, C.; Fabbri, A.; Ragni, L. Ripeness Evaluation of Kiwifruit by Hyperspectral Imaging. *Biosyst Eng* 2022, *223*, 42–52, doi:10.1016/j.biosystemseng.2021.08.009.
268. Afonso, A.M.; Antunes, M.D.; Cruz, S.; Cavaco, A.M.; Guerra, R. Non-Destructive Follow-up of 'Jintao' Kiwifruit Ripening through VIS-NIR Spectroscopy – Individual vs. Average Calibration Model's Predictions. *Postharvest Biol Technol* 2022, *188*, 111895, doi:10.1016/J.POSTHARVBIO.2022.111895.
269. Li, M.; Pullanagari, R.R.; Pranamornkith, T.; Yule, I.J.; East, A.R. Quantitative Prediction of Post Storage 'Hayward' Kiwifruit Attributes Using at Harvest Vis-NIR Spectroscopy. *J Food Eng* 2017, *202*, 46–55, doi:10.1016/j.jfoodeng.2017.01.002.
270. Qin, L.; Zhang, J.; Stevan, S.; Xing, S.; Zhang, X. Intelligent Flexible Manipulator System Based on Flexible Tactile Sensing (IFMSFTS) for Kiwifruit Ripeness Classification. *J Sci Food Agric* 2024, *104*, 273–285, doi:10.1002/JSFA.12916.
271. Jamshidi, B. Ability of Near-Infrared Spectroscopy for Non-Destructive Detection of Internal Insect Infestation in Fruits: Meta-Analysis of Spectral Ranges and Optical Measurement Modes. *Spectrochim Acta A Mol Biomol Spectrosc* 2020, *225*, 117479, doi:10.1016/J.SAA.2019.117479.
272. Zheng, Y.; Zhou, Y.; Liu, P.; Zheng, Y.; Wei, Z.; Li, Z.; Xie, L. Improving Discrimination Accuracy of Pest-Infested Crabapples Using Vis/NIR Spectral Morphological Features. *Journal of Food Measurement and Characterization* 2024, doi:10.1007/S11694-024-02841-Y.
273. Jamshidi, B. Ability of Near-Infrared Spectroscopy for Non-Destructive Detection of Internal Insect Infestation in Fruits: Meta-Analysis of Spectral Ranges and Optical Measurement Modes. *Spectrochim Acta A Mol Biomol Spectrosc* 2020, *225*, 117479, doi:10.1016/J.SAA.2019.117479.
274. Cao, N. Calibration Optimization and Efficiency in near Infrared Spectroscopy, 2013.
275. Bates, S.; Hastie, T.; Tibshirani, R. Cross-Validation: What Does It Estimate and How Well Does It Do It? *J Am Stat Assoc* 2021, *119*, 1434–1445, doi:10.1080/01621459.2023.2197686.
276. Arlot, S.; Celisse, A. A Survey of Cross-Validation Procedures for Model Selection. *Stat Surv* 2010, *4*, doi:10.1214/09-SS054.
277. Xu, Q.S.; Liang, Y.Z. Monte Carlo Cross Validation. *Chemometrics and Intelligent Laboratory Systems* 2001, *56*, 1–11, doi:10.1016/S0169-7439(00)00122-2.
278. Xiaobo, Z.; Jiewen, Z.; Povey, M.J.W.; Holmes, M.; Hanpin, M. Variables Selection Methods in Near-Infrared Spectroscopy. *Anal Chim Acta* 2010, *667*, 14–32.
279. Altieri, G.; Genovese, F.; Tauriello, A.; Di Renzo, G.C. Models to Improve the Non-Destructive Analysis of Persimmon Fruit Properties by VIS/NIR Spectrometry. *J Sci Food Agric* 2017, *97*, 5302–5310, doi:10.1002/jsfa.8416.
280. Altieri, G.; Matera, A.; Genovese, F.; Di Renzo, G.C. Models for the Rapid Assessment of Water and Oil Content in Olive Pomace by Near-Infrared Spectrometry. *J Sci Food Agric* 2020, *100*, 3236–3245, doi:10.1002/jsfa.10361.

281. Altieri, G.; Rashvand, M.; Mammadov, O.; Matera, A.; Genovese, F.; Di Renzo, G.C. Use of Wavelength Interaction Terms to Improve near Infrared Spectroscopy Models of Donkey Milk Properties. *J Near Infrared Spectrosc* 2022, *30*, 219–226, doi:10.1177/09670335221097004.
282. Sobol, I.M. *A Primer for the Monte Carlo Method*; 1st ed.; CRC Press Inc., 1994; ISBN 084938673X,9780849386732.
283. Metropolis, N.; Ulam, S. The Monte Carlo Method. *J Am Stat Assoc* 1949, *44*, 335–341, doi:10.1080/01621459.1949.10483310.
284. McGlone, V.; Technology, S.K.-P.B. and; 1998, undefined Firmness, Dry-Matter and Soluble-Solids Assessment of Postharvest Kiwifruit by NIR Spectroscopy. *Elsevier*.
285. Li, H.; Pidakala, P.; Billing, D.; Technology, J.B.-P.B. and; 2016, undefined Kiwifruit Firmness: Measurement by Penetrometer and Non-Destructive Devices. *Elsevier*.
286. Cevoli, C.; Iaccheri, E.; Fabbri, A.; Ragni, L. Data Fusion of FT-NIR Spectroscopy and Vis/NIR Hyperspectral Imaging to Predict Quality Parameters of Yellow Flesh "Jintao" Kiwifruit. *Biosyst Eng* 2024, *237*, 157–169, doi:10.1016/J.BIOSYSTEMSENG.2023.12.011.
287. Wan, C.; Yue, R.; Li, Z.; Fan, K.; Chen, X.; Li, F. Prediction of Kiwifruit Sweetness with Vis/NIR Spectroscopy Based on Scatter Correction and Feature Selection Techniques. *Applied Sciences 2024, Vol. 14, Page 4145* 2024, *14*, 4145, doi:10.3390/APP14104145.
288. Nicolaï, B.M.; Beullens, K.; Bobelyn, E.; Peirs, A.; Saeys, W.; Theron, K.I.; Lammertyn, J. Nondestructive Measurement of Fruit and Vegetable Quality by Means of NIR Spectroscopy: A Review. *Postharvest Biol Technol* 2007, *46*, 99–118, doi:10.1016/J.POSTHARVBIO.2007.06.024.
289. Zhu, H.; Chu, B.; Fan, Y.; Tao, X.; Yin, W.; He, Y. Hyperspectral Imaging for Predicting the Internal Quality of Kiwifruits Based on Variable Selection Algorithms and Chemometric Models. *Scientific Reports 2017 7:1* 2017, *7*, 1–13, doi:10.1038/s41598-017-08509-6.
290. Xia, Y.; Zhang, W.; Che, T.; Hu, J.; Cao, S.; Liu, W.; Kang, J.; Tang, W.; Li, H. Comparison of Diffuse Reflectance and Diffuse Transmittance Vis/NIR Spectroscopy for Assessing Soluble Solids Content in Kiwifruit Coupled with Chemometrics. *Applied Sciences 2024, Vol. 14, Page 10001* 2024, *14*, 10001, doi:10.3390/APP142110001.
291. Fu, X.; Ying, Y.; Lu, H.; Xu, H.; … H.Y.I. for F.Q. and; 2007, undefined FT-NIR Diffuse Reflectance Spectroscopy for Kiwifruit Firmness Detection. *SpringerX Fu, Y Ying, H Lu, H Xu, H YuSensing and Instrumentation for Food Quality and Safety, 2007•Springer* 2007, *1*, 29–35, doi:10.1007/S11694-007-9004-2.
292. Worasawate, D.; Sakunasinha, P.; Chiangga, S. Automatic Classification of the Ripeness Stage of Mango Fruit Using a Machine Learning Approach. *AgriEngineering 2022, Vol. 4, Pages 32-47* 2022, *4*, 32–47, doi:10.3390/AGRIENGINEERING4010003.
293. Sarakum, T.; Sukpancharoen, S. Non-Destructive Sweetness Classification of Khao Tang Kwa Pomelos Using Machine Learning with Acoustic and Image Processing. *Journal of Food Composition and Analysis* 2025, *142*, 107385, doi:10.1016/J.JFCA.2025.107385.