Improving robotic active perception in agriculture using online self-supervised learning



Jianchao Ci

Propositions

1. Active perception is the most effective way for resolving occlusions in unstructured environments.
   (this thesis)

2. The transition from robotic automation to autonomy requires continuous learning.
   (this thesis)

3. Generative AI weakens publication count as a measure of scientific contribution.

4. Academic systems overvalue novelty and undervalue practical implementation.

5. Completion of work is more important than its perfection.

6. Traveling requires more structured planning than programming.

Propositions belonging to the thesis, entitled

Improving robotic active perception in agriculture using online self-supervised learning

Jianchao Ci
Wageningen, 02 September 2026

Improving robotic active perception in agriculture using online self-supervised learning

Jianchao Ci

Thesis committee

Promotor
Prof. Dr ir. E.J. van Henten
Professor of Agricultural Biosystems Engineering
Wageningen University & Research

Co-promotor
Dr G.W. Kootstra
Associate Professor in Agricultural Biosystems Engineering
Wageningen University & Research

Other members
Prof. Dr Bedir Tekinerdogan, Wageningen University & Research
Prof. Dr Lazaros Nalpantidis, Technical University of Denmark, Denmark
Dr Marija Popovic, Delft University of Technology
Dr Jochen Hemming, Wageningen University & Research

This research was conducted under the auspices of the C.T. de Wit Graduate School of Production Ecology & Resource Conservation (PE&RC)

Improving robotic active perception in agriculture using online self-supervised learning

Jianchao Ci

Thesis

submitted in fulfilment of the requirements for the degree of doctor at
Wageningen University
by the authority of the Rector Magnificus
Prof. Dr C. Kroeze,
in the presence of the
Thesis Committee appointed by the Academic Board
to be defended in public
on Wednesday 2 September 2026
at 10:30 a.m. in the Omnia Auditorium.

Jianchao Ci
Improving robotic active perception in agriculture using online self-supervised learning
191 pages.

PhD thesis, Wageningen University, Wageningen, the Netherlands (2026)
With references, with summary in English and Chinese

DOI: https://doi.org/10.18174/681762

Contents

Page

Contents v

Chapter 1 Introduction 1

Chapter 2 3D Pose Estimation of Tomato Peduncle Nodes using Deep Keypoint Detection and Point Cloud 15

Chapter 3 SSL-NBV: A Self-Supervised-Learning-Based Next-Best-View algorithm for Efficient 3D Plant Reconstruction by a Robot 43

Chapter 4 Robotic Self-Supervised Local View Planning for Efficient and Scalable 3D Plant Reconstruction Across Varying Plant Sizes 79

Chapter 5 SSL-Semantic-NBV: A Self-Supervised Learning-Based NBV for Target-Aware 3D Reconstruction in Agricultural Robotics 117

Chapter 6 General discussion 137

References 157

Summary 173

中文摘要 177

Acknowledgements 179

About the author 185

PE&RC Training and Education Statement 187

Authorship statement 189

Chapter 1

General Introduction

1.1 Greenhouse cultivation: benefits and challenges

The human practice of crop production in protected facilities has a long history. One of the earliest records dates back to 14–37 CE in Rome, when Emperor Tiberius ordered cucumbers (*Cucumis*) to be planted on carts and moved into protected houses, called *Specularia*, during cold nights, enabling year-round production (Paris et al., 2008). In the 1450s in Korea, an underfloor heating system (*Ondol*) was used to grow citrus plants during winter (Rajesh et al., 2020). By the 17th and 18th centuries, tall masonry structures with glass windows, known as *Orangeries*, were constructed in France to protect fruit trees such as oranges from frost (Wearn & Mabberley, 2016). In the 19th century, Europe saw the development of large glass-covered wrought-iron structures, or glasshouses, designed to cultivate exotic tropical plants (Nemali, 2022).

Commercial-scale greenhouse production began to expand in the 20th century in the Westland region of the Netherlands. Although many greenhouses were damaged during the Second World War, the post-war reconstruction period led to a major innovation: engineers in the city of Venlo designed the Venlo-style greenhouse, a tall, multispan, glass-covered and insulated structure that became a milestone in modern greenhouse design (Berkers & Geels, 2011).

The terms greenhouse horticulture, protected cultivation, and Controlled Environment Agriculture (CEA) are often used to describe crop production systems that enable manipulation of environmental conditions to create optimal growing environments independent of the external climate (Heuvelink et al., 2024). While CEA initially emerged in colder regions to raise temperatures and enable the cultivation of exotic or out-of-season crops, its scope has since broadened considerably. Advances in structural design, glazing materials, climate control, artificial lighting, and fertigation now position CEA as a comprehensive solution for mitigating extreme weather events (e.g., droughts, heat waves, floods), reducing pest and disease pressures, limiting soil degradation, enhancing yields, and improving resource-use efficiency (Heuvelink et al., 2024; Nemali, 2022; Kooten et al., 2008).

For example, tomatoes, one of the most widely grown greenhouse crops, can easily achieve yields exceeding 500 t/ha in Dutch year-round cultivation greenhouses, compared to 50–100 t/ha in open-field (Heuvelink & Dorais, 2005). At the same time, advanced recirculating irrigation technologies reduce water consumption by about 90% relative to open-field systems, while the enclosed greenhouse environment lowers pest and disease incidence and thereby decreases pesticide use (Kooten et al.,

2008).

In the context of increasing population and growing pressure on natural resources such as land and water (FAO, 2018; Long et al., 2015), CEA has emerged as a promising approach to produce high-value horticultural crops with improved resource-use efficiency and year-round productivity.

Although CEA has achieved significant progress in improving yield, resource efficiency, and environmental control, large-scale production, particularly of high-wired fruit crops, is highly labor-intensive, and labor costs are high. In developed countries, labor can account for more than 30% of total production costs in greenhouse crops such as tomatoes, cucumbers, and sweet peppers (Manzano-Agugliaro & Cañero-Leon, 2010; van Henten et al., 2002). At the same time, the harsh working environment inside greenhouses exacerbates labor scarcity. Workers are often required to perform repetitive and physically demanding tasks such as harvesting and pruning under conditions of high temperature, and elevated humidity (Kootstra et al., 2021). These problems make greenhouse work increasingly unattractive. Surveys also indicate that the average age of greenhouse workers is rising, raising concerns about long-term workforce sustainability (Noguchi et al., 2004). Furthermore, the dependency on seasonal and migrant workers further adds to instability in the labor supply. To alleviate the problems of labor shortages and high labor costs, increasing levels of automation and robotization appear to be a promising solution (Bac et al., 2014; van Henten et al., 2003).

1.2 Agri-food robotics and challenges

A typical CEA production cycle consists of several key stages: field preparation, breeding, transplanting, crop management, harvesting, sorting, and packing. Depending on the crop type, varying levels of automation have been achieved in certain stages of production through the use of machinery, for instance in breeding, sorting, and packing, which primarily take place during the early and late phases of the production cycle. However, daily maintenance and selective harvesting still rely heavily on manual labor (Bergerman et al., 2016; van Henten, 2006).

This is because early and late phase operations usually do not require advanced human intelligence or precise eye–hand coordination. Conventional mechanization, grounded in industrial automation principles, can efficiently perform these repetitive tasks along fixed paths using simple control systems, without the need for perception or adaptation capabilities (van Henten, 2006).

In contrast, daily maintenance tasks in CEA, such as harvesting and pruning, involve

selective actions in unstructured and cluttered environments. These tasks demand that machines perceive their surroundings, identify relevant targets, make decisions, plan motions, and manipulate objects accurately. Under such conditions, conventional mechanization is insufficient, and robotic systems that integrate perception, decision-making, motion planning, mobility, and manipulation capabilities are required (van Henten, 2006; Kootstra et al., 2021; Bechar & Vigneault, 2016).

Over the past five decades, research interest in agricultural robotics has increased rapidly, as documented in several reviews (Bac et al., 2014; Oliveira et al., 2021; Kootstra et al., 2021). Nevertheless, despite significant progress, large-scale commercial deployment of robotics in CEA remains limited. According to Bac et al. (2014), several requirements must be met before robotic systems can be successfully adopted in practice.

Most critically, robotic performance must be improved to cope with the complexity of agricultural environments. A major challenge arises from severe occlusions, where the parts of the plant to be manipulated, such as fruits or nodes, are often partially or completely occluded by leaves, other plant structures, or neighboring plants. Such occlusions result in incomplete or unreliable sensory information, increasing the uncertainty of robotic operations and the risk of failure (Kootstra et al., 2021). In addition, plants exhibit substantial variation in size, shape, color, and texture due to genetic differences among cultivars and environmental factors such as temperature, light, humidity, nutrient availability, and cultivation systems. Consequently, even plants of the same species can differ considerably in morphology within a single facility (Bac et al., 2014). Moreover, plant structures are dynamic and change over time as a result of growth and routine crop management. Together, these variations and dynamics further complicate robotic operation and require systems to continuously adapt to changing plant geometries.

Addressing these challenges requires improvements in robotic performance. In particular, perception systems must become more robust and accurate in detecting and localizing targets and/or in constructing reliable 3D representations of the environment and crops. Robots must also incorporate stronger reasoning and learning capabilities to adapt to changing conditions (Hagras et al., 2002), as well as motion and path-planning strategies that ensure safe and reliable task execution. Appropriate hardware design, including specialized manipulators, mobility platforms, and end-effectors, further support effective operation across different tasks (Afonso et al., 2019; Bechar & Vigneault, 2016; van Henten et al., 2002).

Beyond robotic performance, additional factors influence their successful deployment in CEA. Task complexity can be reduced by adapting cultivation systems or selecting

crop cultivars that are more compatible with robotic operation (van Henten et al., 2002, 2006). Clear performance metrics and transparent reporting are necessary to support system evaluation and comparison (Bonsignorio et al., 2009; Bac et al., 2014). In addition, economic feasibility, safety, logistics, and societal acceptance must be satisfied for large-scale adoption (Pedersen et al., 2008; Kootstra et al., 2021).

As these requirements have not yet been fully met, the commercial deployment of robotics in CEA remains limited.

1.3 Technical solution: toward smarter robotic perception

This thesis focuses on solutions to enhance robotic performance, particularly through the development of more intelligent robotic perception systems.

Robotic perception is an essential step for enabling robots to sense their surroundings, identify targets, and extract task-relevant information to support downstream operations. In robotic harvesting, for example, a perception system detects the cutting point, such as the peduncle, and estimates its location and orientation. This information is then used to guide the robot' s end-effector for fruit detachment. However, when operating in complex CEA environments, current perception systems remain limited in accuracy and robustness, mainly due to severe occlusions and variations.

A robotic perception system integrates sensors and algorithms to perceive and interpret the environment (Bac et al., 2014; Fountas et al., 2022). Depending on the application, commonly used sensors include RGB cameras for capturing visual appearance and geometry, multispectral and hyperspectral cameras for inferring chemical properties, and depth cameras or LiDAR for acquiring three-dimensional information (Kootstra, 2024). In this thesis, data acquisition relies on cost-effective, off-the-shelf sensors, while the primary contribution lies at the algorithmic level.

Perception algorithms process sensory data to extract task-relevant information, including object attributes such as color, texture, spectral signatures, and ripeness; spatial information such as object location, pose, and 3D plant geometry; and temporal information such as motion or gradual changes over time (Kootstra, 2024). Common perception tasks for information extraction include object detection, segmentation, tracking, pose estimation, and three-dimensional reconstruction. However, reliably performing such tasks in CEA environments remains challenging.

First, plant appearance can change significantly under varying environmental condi-

tions, such as changes in illumination, while plant structures often exhibit irregular and diverse geometries. These variations make reliable object detection, pose estimation, and tracking over time difficult. Over the past decade, advances in computer vision, particularly through the adoption of deep learning, have significantly improved perception accuracy and robustness in handling such variations by automatically learning representative features from raw data. Reviews across a wide range of agricultural vision tasks show that deep learning–based methods consistently outperform traditional approaches based on hand-crafted features, although further improvements are still required (Fountas et al., 2022; Kamilaris & Prenafeta-Boldú, 2018).

Despite these advances, occlusion remains a major challenge for robotic perception. In CEA environments, plant parts relevant to the task, such as fruits or nodes, are frequently partially or fully occluded by leaves, stems, or other plant organs. Such occlusions prevent perception systems from reliably detecting targets and estimating the information required for subsequent operations, leading to missing or inaccurate inputs and increased operational uncertainty. While deep learning–based methods demonstrate some level of ability to handle partial occlusion (Zhang et al., 2018; Kortylewski et al., 2021), they inherently lack the capability to cope with heavy or complete occlusion, which is common in densely structured and cluttered CEA environments.

Therefore, this thesis focuses on developing solutions to more effectively address occlusion in robotic perception under CEA conditions. In practice, occlusions can arise in several forms. They may occur due to self-occlusion within a plant, where leaves or stems block the view of fruits or nodes, or due to occlusion from neighboring plants in densely planted rows. Moreover, the degree and pattern of occlusion change over time as plants grow, are pruned, or move slightly due to environmental factors such as airflow. These variations require perception methods that are robust and adaptive to different occlusion patterns and plant structures.

1.3.1 Multi-view perception system

A common strategy to address occlusion is the use of multiple observation viewpoints, where a set of camera poses (e.g., varying locations, angles, or both) are employed to observe the target from different perspectives, and the final result is obtained through a comprehensive analysis of all observations. In this way, information that is not available from a single camera view can be aggregated to support robotic operations. For example, early work by van Henten et al. (2003) reported that the success rate of robotic sweet pepper harvesting increased from 39.5% to 74.4% when the number of harvesting attempts was increased from one to three. Although this

approach did not explicitly formulate viewpoint planning as a perception problem, it implicitly demonstrated the advantage of observing targets from multiple perspectives. Similarly, Hemming et al. (2014) reported an improvement of over 30% in the detectability (i.e., the visible part of a target above a defined threshold) of sweet peppers when the number of viewpoints was increased from one to five. Boogaard et al. (2020) similarly demonstrated that employing multiple viewpoints improved the accuracy of internode detection in cucumbers under occlusion. Golbach et al. (2016) developed a ten-camera system for 3D plant reconstruction in seedling phenotyping, which was later applied by Shi et al. (2019) to tomato seedling phenotyping and showed improved node detection accuracy compared with single-view results.

While multi-view systems help reduce occlusion to some extent, most rely on a fixed set of predefined viewpoints determined through expert knowledge. As a result, they tend to perform well for specific plant types with similar size and morphology but lack robustness in handling variations among plants, since the predefined viewpoints do not dynamically adapt to actual plant morphology or occlusion conditions. Consequently, occlusions may still occur. Increasing the number of viewpoints could further improve perception accuracy and information completeness but also increases device complexity and computational demand.

Ideally, viewpoint planning should consider the actual plant morphology and occlusion conditions to maximize information collection with the fewest possible camera viewpoints. Furthermore, instead of developing systems that rely on multiple coordinated cameras, using a single camera mounted on the manipulator for viewpoint transitions is widely accepted as a flexible and practical design (van Henten et al., 2003; Lehnert et al., 2017; Zaenker et al., 2021b).

1.3.2 Active perception system

The above concept forms the foundation of active perception (AP) (Bajcsy et al., 2018). Unlike passive systems that collect data from fixed viewpoints, active systems assess how well the currently collected information satisfies the task objective and then take actions to acquire additional data (Gibbs et al., 2020; Atefi et al., 2021). These actions may involve adjusting camera parameters (e.g., aperture or focal length) (Tenenbaum, 1971) or changing the camera pose (Scott et al., 2003; Delmerico et al., 2018) to reveal previously unseen regions of the scene. This perception–action cycle repeats until the perception goal, such as complete 3D reconstruction or sufficient target visibility, is achieved.

In this thesis, this cycle of evaluation and data acquisition is implemented through Next-Best-View (NBV) planning, where the system iteratively repositions the camera

to collect new information.

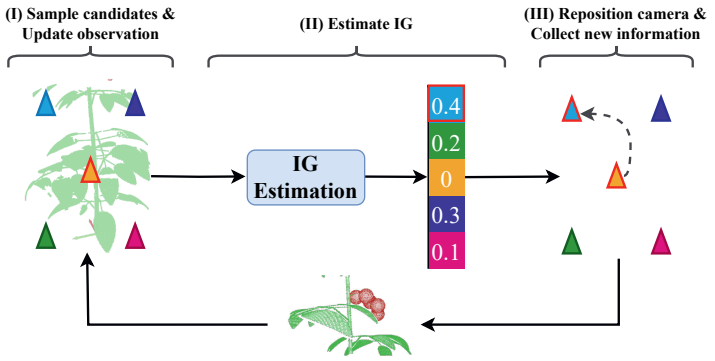


Figure 1.1: Illustration of a NBV iteration. Step (I) generates candidate viewpoints, step (II) evaluates IG of each candidate, step (III) selects the candidate with the highest IG as the next camera pose

NBV methods for viewpoint planning typically follow a generation-evaluation-selection pipeline (Tarabanis et al., 1095). As shown in Fig. 1.1, starting from an initial camera pose: (I) a set of candidate viewpoints is generated to represent possible camera poses; (II) each candidate is evaluated using an information gain (IG) metric that estimates how much novel information it could provide; and (III) the candidate with the highest IG is chosen as the next camera viewpoint, referred to as the next best view. This iterative process continues until the perception goal is achieved. Since viewpoint selection is guided by the data collected from the actual object, NBV provides a more effective solution for resolving occlusions in plants with complex and variable structures (Atefi et al., 2021; Burusa et al., 2024a; Gibbs et al., 2020; Zaenker et al., 2023; Zapotezny-Anderson & Lehnert, 2019).

Depending on the way IG is estimated, NBV approaches can be broadly categorized into explicit and implicit approaches. In explicit methods, the scene is typically represented as a volumetric voxel map, where each voxel is assigned an occupancy probability indicating whether it is occupied, free, or unexplored. Extended versions may also include a class label to specify whether a voxel belongs to the target (e.g., fruits). The IG of a viewpoint is then computed by casting rays from that viewpoint through the map and counting the traversed voxels. Intuitively, the IG value reflects how much additional information about the target can be obtained by observing that region. Representative works in the agricultural robotics include Burusa et al. (2024a,c,b) and Zaenker et al. (2021a,b). While these methods provide accurate and interpretable IG estimation, they are computationally and memory intensive due to

extensive ray-casting operations.

Implicit methods address this limitation by training neural networks to fit a regression between the current observation and the corresponding IG of candidate viewpoints. During inference, the trained network rapidly predicts IG from current observations, avoiding the need for extensive ray-casting operations on volumetric representations. For example, Mendoza et al. (2020) proposed NBV-Net, a 3D convolutional network that predicts IG from occupancy grids, achieving a 1,800−3,000 times speedup over explicit approaches. Subsequent works (Zeng et al., 2020; Han et al., 2022) further improved prediction efficiency by using point clouds as network inputs.

1.4 Knowledge gap

Despite their computational advantages, implicit NBV methods need to meet several requirements to be effectively applied in CEA environments, specifically the adaptability, real-world feasibility, scalability, and efficiency. These requirements are interrelated to some extent; for example, real-world feasibility depends on performance across the other dimensions. Nevertheless, clearly distinguishing these aspects is necessary for systematic evaluation and meaningful comparison of methods. To this end, the requirements are defined below.

Adaptability requires implicit NBV methods that can adapt to novel environments and previously unseen plant types. In CEA settings, where environments are highly complex and dynamic, this requirement is essential but absent from existing implicit NBV approaches. Existing implicit NBV systems usually rely on offline, pre-collected datasets. When the method fails under new environments, additional data collection and network retraining become necessary. In a dynamic environment such as CEA, where plant structures and environmental conditions change frequently, this reactive cycle of data recollection and retraining is impractical. Although current methods show some robustness to environmental and object variation (Burusa et al., 2024a; Han et al., 2022; Mendoza et al., 2020), the substantial differences in plant morphology, growth stages, and environmental conditions make it difficult for a trained network to generalize across all real-world scenarios. For this reason, improving the adaptability of implicit NBV systems became a central focus of this thesis. The goal is to enable the system to continuously adapt to new contexts during task execution. This training paradigm is commonly referred to as online learning. As illustrated by Fig. 1.2, during NBV execution the system continuously collects newly generated data and stores them in a buffer, from which training samples are selected to update the network online.

In real-world deployment, collecting large amounts of training data for implicit NBV

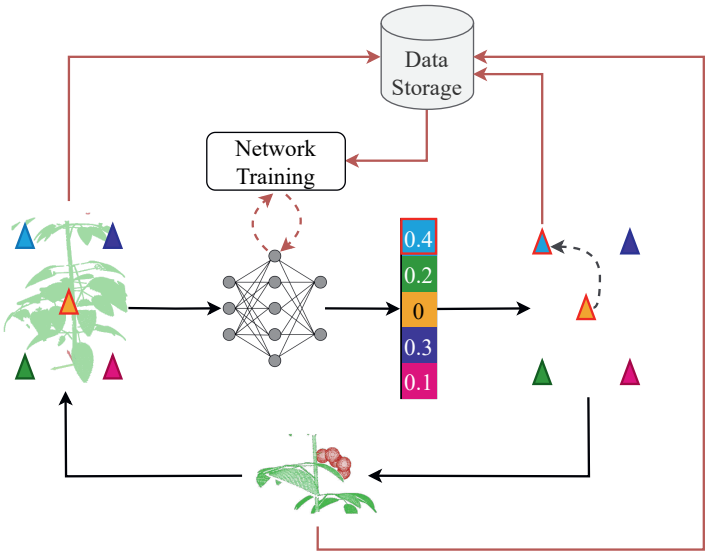


Figure 1.2: Illustration of an NBV iteration with online learning. During NBV execution, the generated data are stored in a data buffer, from which training samples are drawn to train the network online during task execution.

methods remains a major challenge, primarily due to the difficulty of efficiently and accurately calculating ground-truth IG for candidate viewpoints required by existing methods. IG quantifies the novel information gained from a new viewpoint. Zeng et al. (2020) and Han et al. (2022) estimated ground-truth IG by comparing new observations from a viewpoint with the complete 3D model of the object. However, this approach is impractical in agricultural scenarios, where plants exhibit high morphological variability and 3D models are rarely available in real-world setting. This limitation constrains most implicit NBV methods to simulation environments and hinders their transfer to real-world applications. To enable real-world deployment, a key requirement is the development of an IG metric that allows efficient and accurate ground-truth IG estimation directly from the observable data collected by the system itself.

Scalability requires NBV methods to extend the viewpoint planning strategy to different plant sizes and perception targets, such as fruit and node, depending on the robotic task. A key limitation of many implicit NBV approaches is that their network design necessitates a fixed number of candidate viewpoints, with each candidate viewpoint corresponding to an output entity of the network. This design inherently assumes similar target sizes and viewpoint distributions to fully cover the target. In CEA environments, however, plants can vary significantly in size and geometry and

continue to grow over time, requiring different numbers and distributions of viewpoints. As a result, the applicability of such implicit NBV methods is limited, and adapting them typically requires modifying the network architecture and retraining. In addition, most existing methods treat the entire scene as the target, aiming to capture all details and assigning equal importance to every region during exploration (Mendoza et al., 2020; Jin et al., 2023). While this strategy is suitable for tasks requiring full-scene information, such as plant growth monitoring, it is inefficient for tasks like harvesting or de-leafing, where only specific plant parts (e.g., fruits or nodes) are relevant (Burusa et al., 2024c; Bartolomei et al., 2021; Zeng et al., 2022; Jin et al., 2024). Ideally, a scalable NBV-based perception system should adapt to varying plant sizes and different plant parts without requiring changes to the network architecture or retraining from scratch.

Efficiency in NBV planning generally refers to how quickly an NBV task can be completed while satisfying a predefined perception criterion, such as reconstruction completeness in 3D reconstruction tasks. It is a fundamental requirement for NBV methods. To successfully deploy NBV methods in CEA environments, several factors related to NBV efficiency must be considered. First, the speed of IG prediction determines how rapidly the next camera viewpoint can be selected (IG prediction efficiency). Many implicit NBV methods rely on neural networks for IG prediction but still depend on computationally expensive ray-casting operations during data preprocessing, such as the work of (Zeng et al., 2022), which reduces overall efficiency. Second, selected viewpoints should provide the maximum amount of useful information (viewpoint efficiency), for example by maximizing incremental object coverage in 3D reconstruction. While existing implicit NBV methods demonstrate promising performance on simplified or synthetic targets (Wang et al., 2024; Zeng et al., 2020), their effectiveness in complex CEA environments involving plants and specific plant parts needs further evaluation. Third, efficiency also depends on the ability to maximize information acquisition while minimizing robotic motion (trajectory efficiency). Many existing methods select viewpoints solely based on information gain, which often leads to large and unnecessary camera movements and increased robotic motion, thereby reducing overall efficiency. Instead, viewpoint planning should be performed at the trajectory level to reduce redundant motion and consequently decrease execution time and energy consumption (Jin et al., 2025; Pan et al., 2022). In practice, a balance is often needed between selecting the most informative viewpoint (high viewpoint efficiency) and minimizing camera movement (high trajectory efficiency). For implicit NBV methods, efficiency also depends on how quickly the network can be trained and how effectively training samples are utilized.

Collectively, these requirements limit the ability of current implicit NBV methods to

satisfy the demands of robotic perception systems for downstream tasks.

1.5 Objective, main hypothesis, and thesis outline

The main objective of this thesis is to advance active perception in agricultural robotics by improving existing implicit NBV methods through robotic self-supervised learning that effectively address occlusion challenges in complex CEA environments. The objective is achieved by specifically targeting the key requirements of adaptability, real-world feasibility, scalability, and efficiency in implicit NBV-based perception, thereby improving robotic perception to handle occlusions more effectively and robustly in agricultural tasks during real-world operations.

The central hypothesis of this thesis is:

> *"Online self-supervised learning enables implicit NBV methods to adjust their viewpoint planning strategy to novel scenarios, resulting in more efficient environment perception in previously unseen conditions compared to offline methods."*

The central hypothesis is evaluated by comparing the efficiency of online adaptive implicit NBV with offline implicit NBV in 3D reconstruction tasks under occlusion. The evaluation also focuses on the methods' adaptability when applied to novel environments, scalability across different plant sizes, and feasibility in real-world settings. Fig. 1.3 provides an overview of the studies conducted in each chapter and the corresponding key requirements addressed.

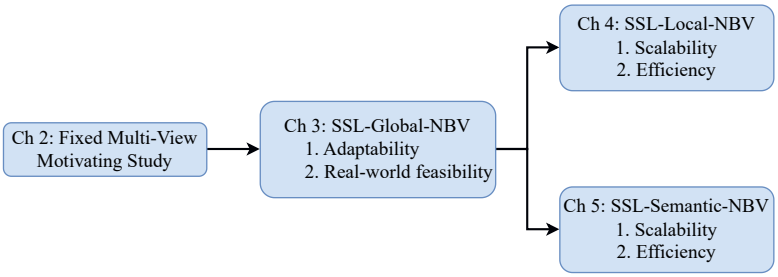


Figure 1.3: Overview of the studies conducted in this thesis and the key requirements addressed in each chapter.

Our exploration of the importance of viewpoint selection for the overall performance of robotic perception begins in Chapter 2. In this chapter, a deep learning-based

3D pose estimation system was proposed for tomato peduncles, with the detection and measured pose being essential for downstream tasks such as robotic harvesting, and phenotyping, by providing structural trait data. The accuracy of the proposed method was evaluated across five viewpoints: a top-down view, a bottom-up view, two side views, and one perpendicular view relative to the plant row. The pipeline consisted of two components: (1) estimating pose in 2D images, and (2) projecting the 2D pose into 3D by incorporating spatial data from a depth sensor. Results in both 2D and 3D pose estimation showed that the top-down and perpendicular views produced the highest accuracy, emphasizing the critical role of viewpoint selection in robotic perception. This work serves as a preliminary test and does not involve active viewpoint selection for improving robotic perception accuracy. In the later chapters, active robotic perception is introduced and further optimized.

In Chapter 3, an implicit NBV-based robotic active perception system is presented, where a camera mounted on the robot end-effector in an eye-in-hand setup allows flexible viewpoint changes. This work directly confronts the requirement of adaptability and real-world feasibility by integrating implicit NBV into a self-supervised learning (SSL) framework, allowing the robot to autonomously collect training data and adapt to novel environments and previously unseen plant types. Since the method determines the next camera pose by evaluating the entire set of candidate viewpoints, it follows a global planning strategy and is therefore referred to as SSL-Global-NBV. The SSL-Global-NBV method was evaluated on 3D plant reconstruction tasks, aiming to achieve complete reconstructions with as few viewpoints as possible, and its reconstruction efficiency was compared with that of offline implicit and explicit NBV methods. This chapter mainly investigates whether online self-supervised learning can improve the adaptability and perception efficiency of implicit NBV methods under occlusion in novel agricultural settings.

A key limitation of SSL-Global-NBV is its reliance on a predefined set of viewpoints with a fixed number of candidate views, which requires prior assumptions about plant size. Moreover, its global planning strategy often leads to extensive camera traversal across the entire viewpoint space, resulting in substantial robotic motion and low trajectory efficiency. To address these limitations, Chapter 4 introduced SSL-Local-NBV as an extension of SSL-Global-NBV, with modifications for local camera movements. This work targets key requirements of scalability and efficiency. Instead of selecting viewpoints from a predefined global set, SSL-Local-NBV determines the next camera pose through relative motions from the current pose by adjusting position, orientation, or both. This design significantly reduces robotic motion and improves trajectory efficiency. Moreover, by removing the dependence on a fixed viewpoint set, SSL-Local-NBV enhances scalability across different plant sizes. This chapter mainly

investigates the gains in trajectory efficiency and scalability achieved by local viewpoint planning compared with global viewpoint planning in three-dimensional plant reconstruction tasks.

Despite the advances achieved by SSL-Global-NBV and SSL-Local-NBV, a key limitation of both methods is that they focus on reconstructing the entire scene, treating all regions as equally important. This approach is suitable for tasks such as plant monitoring, where observations about the entire plant or scene are required. However, for tasks such as harvesting and deleafing, which require information only about specific plant parts, allocating sensing effort to irrelevant regions is inefficient. To address this limitation, Chapter 5 introduces SSL-Semantic-NBV, which enables viewpoint planning focused on specific plant parts. This approach, which prioritizes viewpoint selection based on task-relevant plant parts, is referred to as target-aware NBV planning. This work targets the requirements of scalability and efficiency in implicit NBV by enabling adaptation across different perception tasks and reducing exploration of irrelevant scene regions. Compared with perception for the whole scene, target-aware NBV is considerably more challenging due to the sparse visibility of task-relevant targets. For example, fruits are often observable from only a limited set of viewpoints, making reliable perception more difficult. In the context of implicit NBV, this sparsity further complicates network training, as most candidate viewpoints contribute little or no information gain. To address these challenges, the training pipeline was redesigned to enable effective learning from sparse information gain signals associated with sparsely observable targets. This chapter evaluates the benefits of the target-aware mechanism in terms of reconstruction efficiency across different plant parts.

Chapter 6, finally, concludes the thesis with a general discussion, drawing on the findings of the previous chapters and their contributions to the overall objectives. This chapter discusses the added value of learning-based active perception for agricultural robotics, examines its current limitations, and provides recommendations for future research. It then closes with a summarized conclusion that reflects the central hypothesis.

Chapter 2

3D Pose Estimation of Tomato Peduncle Nodes using Deep Keypoint Detection and Point Cloud

This chapter has been published as:

Ci, J., Wang, X., Rapado-Rincón, D., Burusa, A. K., & Kootstra, G. (2024). 3D pose estimation of tomato peduncle nodes using deep keypoint detection and point cloud. *Biosystems Engineering*, 243, 57–69. https://doi.org/10.1016/j.biosystemseng.2024.04.017

Abstract

Greenhouse production of fruits and vegetables in developed countries is challenged by labour scarcity and high labour costs. Robots offer a good solution for sustainable and cost-effective production. Acquiring accurate spatial information about relevant plant parts is vital for successful robot operation. Robot perception in greenhouses is challenging due to variations in plant appearance, viewpoints, and illumination. This paper proposes a keypoint-detection-based method using data from an RGB-D camera to estimate the 3D pose of peduncle nodes, which provides essential information to harvest the tomato bunches.

Specifically, this paper proposes a method that detects four anatomical landmarks in the colour image and then integrates 3D point-cloud information to determine the 3D pose. A comprehensive evaluation was conducted in a commercial greenhouse to gain insight into the performance of different parts of the method. The results showed: (1) high accuracy in object detection, achieving an Average Precision (AP) of AP@0.5=0.96; (2) an average Percentage of Detected Joints (PDJ) of the keypoints of PhDJ@0.2=94.31%; and (3) 3D pose estimation accuracy with mean absolute errors (MAE) of 11° and 10° for the relative upper and lower angles between the peduncle and main stem, respectively. Furthermore, the capability to handle variations in viewpoint was investigated, demonstrating the method was robust to view changes. However, canonical and higher views resulted in slightly higher performance compared to other views. Although tomato was selected as a use case, the proposed method has the potential to be applied to other greenhouse crops, such as pepper, after fine-tuning.

2.1 Introduction

Growing vegetables in greenhouses can significantly extend the production period of plants, which increases yields and brings economic benefits to the owner of the greenhouse. Most greenhouse crops, such as tomato, cucumber, and bell pepper require selective maintenance and harvesting, making production activities labour-intensive. This poses a huge challenge for production because the labour cost is high and labour is scarce (Benavides et al., 2020). A higher level of automation and robotization is seen as a good solution to replace human labour in greenhouse production, and many studies have been proposed to more effectively involve robotics in greenhouse operations, such as harvesting (Bac et al., 2014; Ji et al., 2012; Yoshida et al., 2018), monitoring (Halstead et al., 2018), and phenotyping (Boogaard et al., 2020; Virlet et al., 2017; Vit et al., 2020).

Most greenhouse operations require the robotic perception system to detect the relevant parts of the plant (e.g., fruits, peduncles, and main stem) and to collect sufficient information about them to perform manipulations, such as harvesting, deleafing, and pruning. The perception process is adversely affected by variation in the appearance of the plant and occlusions that are significantly present in complex greenhouse environments (Afonso et al., 2019; Kootstra et al., 2021). Variation results from natural variation in the growth of plants, creating differences in the morphology and appearance of the plant parts, as well as from environmental influences, such as changing illumination. Occlusions happen frequently in the cluttered greenhouse environments, where plant parts are frequently entirely or partially obstructed by other plant parts. To overcome the challenge of variation and improve perception performance, in recent years, deep-learning-based vision systems have been extensively used in agricultural scenarios. Compared to traditional methods that use handcrafted features, deep-learning methods include the feature extraction as part of the end-to-end learning process, which has been shown to perform much better in terms of accuracy and robustness to variation. Kamilaris & Prenafeta-Boldú (2018) reviewed 40 studies using deep learning in agricultural applications and concluded that deep neural networks are superior to other methods. Bargoti & Underwood (2017), for instance, used a deep neural network for object detection of several different fruits, including mangoes, apples, and almonds in orchards, while Sa et al. (2016) used a similar method on a combination of colour and near infra-red images to successfully detect seven different fruit types in greenhouse environments. Boogaard et al. (2020) used deep object detection to detect the leaf and fruit nodes of cucumber plants. A deep neural network for instance segmentation of grape bunches in orchards was used in the work of (Santos et al., 2020). The same method was used by Shi et al. (2019) to segment images of tomato seedlings in stem, node, and the individual leaf. Kang & Chen

(2020) proposed a new network for instance segmentation of apples. These detection algorithms are able to locate different plant parts in 2D images. However, to perform a robotic operation, 3D information about the position and orientation of the objects is needed so that the robot can bring a tool to the desired location and orientation. Often, the 3D pose of an object is defined by the 3D position and the 3D orientation of the object. This was used, for instance by Eizentals & Oka (2016) and Lehnert et al. (2016) to represent the pose of bell peppers. In both studies, the pose of the fruit was estimated by fitting a 3D template to the acquired partial point cloud of the fruit. Li et al. (2018) estimated the pose of peppers by finding the symmetry axes. A deep neural network was proposed in (Wagner et al., 2021) to estimate the 3D pose of strawberries directly from the colour-and-depth (RGB-D) image. Kang & Chen (2020) proposed a point-based neural network to learn to estimate the grasp pose directly from the point cloud of the fruit.

In the animal domain, a different definition of object pose is used. There, computer-vision methods, inspired by human-pose detectors, estimate the pose of the animal by a set of keypoints, representing anatomical landmarks on the animal (Li et al., 2019; Mathis et al., 2020; Pereira et al., 2019; Russello et al., 2022). The pose provides information on the location of these keypoints, as well as the relations between the keypoints. The advantages of this representation are that it can be used for articulated objects and that they are more robust to occlusion, as the non-occluded keypoints can still be detected to represent the object pose. This makes it interesting for pose estimation of plant parts but it has hardly been explored to date, with the exception of Zhang et al. (2022) who used keypoint detection to get the pose of tomato bunches using a combination of stacked hourglasses and an object-detection network. The method localized a set of 2D keypoints on the tomatoes, peduncle and stem and then converting them into 3D keypoints using spatial information provided by the depth camera. The accuracy of the 3D pose estimation, however, was not quantitatively evaluated, so it is unclear if the method can be used for robotic operation.

This study focusses on the 3D pose estimation of peduncle nodes on tomato plants to provide information for fruit harvesting. The peduncle connects the fruit with the main stem. To avoid fruit damage and extend shelf life, tomato trusses are normally detached from the plant by cutting the peduncle. Some other studies tried to detect and localize the peduncle relying on the detection of other plant parts (such as the fruits and branches), then determining the peduncle according to specific morphological relationships between them (Sa et al., 2017; Yoshida et al., 2020). The downside of these methods is that they pose assumptions on the pose of the peduncle. Some applications, for instance, assumed the peduncle to be on top of the fruit, requiring the fruit cluster to be downwards and vertical (Liang et al., 2020; Luo et al., 2016). This

limits the application, as the tomato peduncles can be oriented in any direction. In the work of Boogaard et al. (2020), nodes on cucumber plants were directly detected in the images by a deep neural network (DNN) without assumptions on the relations to other plant parts. In the work of Rong et al. (2022), a DNN for instance-segmentation method was used to predict an image mask for tomato peduncles, after which a hand-crafted algorithm was used to determine the 3D cutting position. However, both approaches focused solely on the peduncle without considering the relationship to the main stem, which can lead collisions between the robot and the plant during harvesting (Lin et al., 2019). Kim et al. (2023) used a more elaborate definition of the pose of a tomato truss, by detecting keypoints at the centre of all tomatoes on the truss, at each calyx, and at the pedicels. They developed a method to detect these points in cropped 2D images of a truss, but did not evaluate the accuracy in 3D. Furthermore, their work is of interest for harvesting individual tomatoes, while the focus of this study is harvesting tomato trusses. Zhang et al. (2023a) developed the Tomato Pose Method (TPMv2), utilizing a deep neural network for simultaneous 2D and 3D keypoint detection on tomato trusses. Their method includes the pose of the peduncle and local direction of the stem in 3D, needed for robotic harvesting of the truss. However, the direct prediction of 3D positions requires additional effort in the annotation of the training data and the more complex neural network requires additional data to train the model. Because the results were not compared to ground-truth geometrical measurements in the real world, it is unclear what the achieved pose accuracy was. If sufficient training data is available and when the additional information about the location of the tomatoes on the truss is needed, the work of Zhang et al. (2022) offers a useful approach. However, in the light of variation in the appearance of plants of different cultivars, the detection system will likely need to be retrained for specific situations, requiring less data-hungry methods.

In this paper, a keypoint-detection-based method was proposed, combining the benefits of a 2D keypoint detection network with point-cloud processing techniques to estimate the full 3D pose of peduncle nodes of tomato plants, enabling robotic operations. As shown in Fig. 2.1, the method was built upon three steps: (1) a deep keypoint detector was used for robust detection of the peduncle node and its pose in 2D images, (2) the detected keypoints were projected to the aligned 3D point cloud, and (3) the 3D pose was estimated. To provide relevant information about the pose and nearby stem segments needed for robotic harvesting, four keypoints was defined: one on the node, one on the peduncle, and two on the main stem. The method was thoroughly evaluated on three aspects: the node detection, the 2D pose estimation, and the 3D pose estimation, each aspect impacting the accuracy of the final estimated 3D pose. The first two analyses were done on the image level, while the last analysis evaluated the accuracy of estimated 3D pose in real-world coordinates compared to

manual measurements. Furthermore, because the success of peduncle node detection might depend on the viewpoint, the result for different camera poses relative to the target object was analysed. This provides insight for optimal viewpoint planning, such as implemented in, e.g., Lehnert et al. (2019) and Burusa et al. (2024b), who aimed to find the optimal viewpoint and utilized a robot arm with a sensor mounted on the end effector to reach this viewpoint for data collection in downstream tasks. All images used in this project (both training and validation) are raw images directly obtained from the camera, which provided a realistic evaluation of the model's performance in real-world applications.

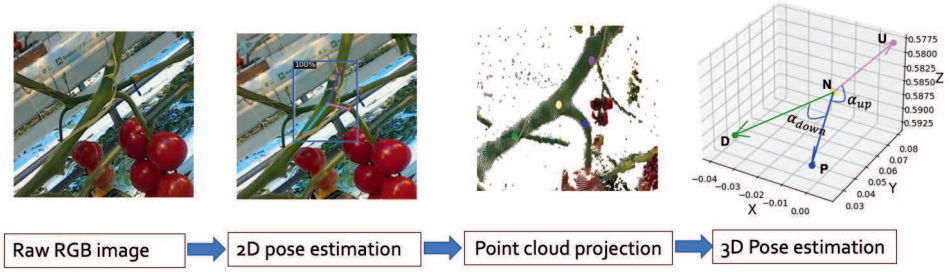


Figure 2.1: Three stages, 2D pose estimation, 3D point cloud projection, and 3D pose estimation.

2.2 Materials and Method

2.2.1 Materials

2.2.1.1 Camera setup and dataset acquisition

The data were collected at a commercial tomato greenhouse located in Beek en Donk, Netherlands. This greenhouse is equipped with a fully automatic climate control system, including temperature, humidity, and lighting control. The tomato plants were grown in a high-wire pattern and the main stem was trained to grow vertically along the wire. The tomato variety used in this research was 'Tasty Tom', which has around 8 fruits per tomato truss, a peduncle diameter of around 7mm at maturity. Standard operational management was applied, including the removal of lower leaves around the ripe tomato trusses (de-leafing) to reduce pests and diseases, and to remove leaves that are no longer contributing to photosynthesis. No modifications of the crop were made for the purpose of this project. Although de-leafing reduces the level of occlusion, the task to detect the pose of the peduncle nodes is still challenging due to different viewing poses, variation in the appearance of the nodes, self-occlusion, and clutter of other neighbouring plants.

A RealSense L515 LiDAR camera was used to capture both RGB images and aligned point clouds, both of which had a resolution of 1280x720. The camera was connected to an Alienware M15 R3 laptop with an Intel i7 10750H CPU and Nvidia GeForce RTX-2070s GPU, running on Windows 10 using the Pyrealsense2 package for image acquisition. Data was collected under ambient lighting, with the camera being hand-held between 50cm-80cm from a peduncle node, which was usually centred in the image. In addition to this node, a large number of peduncle nodes from other plants were present in the background of each image. Data collection occurred between October to December, in the 6th, 10th, and 11th weeks after planting.

The dataset included 648 RGB images, out of which 503 were used for model training and 145 for testing (Table 2.1). The training set included 614 annotated peduncle nodes, with 1-3 nodes per image, while the test set contained 145 annotated peduncle nodes, with a single peduncle node annotated per image. Although the test images regularly contain multiple nodes and the proposed method is also able to detect multiple peduncle nodes, the test focused on the detection of the peduncle closest to the robot, hence only one peduncle was annotated. Apart from the peduncle nodes, also leaf nodes (petiole node) were visible in the images. The detection algorithm had to ignore those and detect only the peduncle nodes.

The test set contained a subset of 95 images of 19 nodes from five different viewpoints to be able to analyse the detection performance for different poses of the peduncle in the image. These viewpoints were named V1 (canonical view), V2 (higher view), V3 (lower view), V4 (leftwards view), and V5 (rightwards view), all directed toward the peduncle node (see Fig. 2.2). V1 is horizontal and perpendicular to the direction of the plant rows, while the positions of V2-V5 were approximately determined, each having an angle of around $60° \pm 15°$ with respect to V1. Although the positions of the five viewpoints were determined approximately in this study, they still provide insights into the impact of camera pose relative to the peduncle node on the model' s performance.

Table 2.1: The composition of the data set.

	Total RGB	Total Nodes	Align point cloud
Training set	503	614 (1-3 per image)	No
Test set	145	145	Yes

2.2.1.2 Ground-truth annotation in 2D

The RGB images were annotated using COCO annotator, an open-source web-based image annotation tool for object detection, segmentation, and keypoint detection

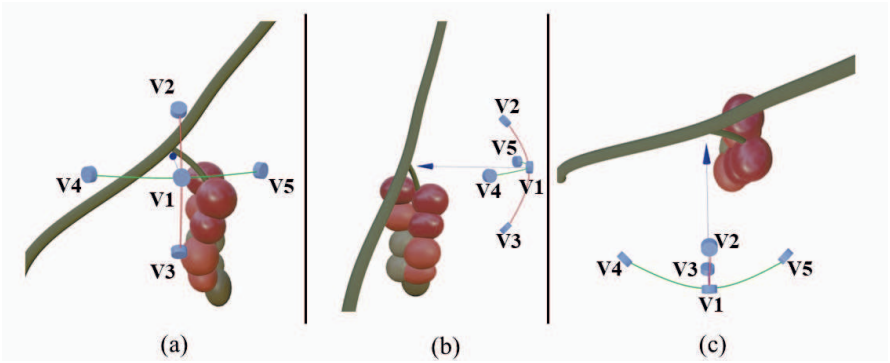


Figure 2.2: The distribution of the 5 viewpoints from (a) the front view, (b) the side view, (c) the top view. V1 represents the canonical view, V2 represents the higher view, V3 represents the lower view, V4 represents the leftwards view, and V5 represents the rightwards view. All viewpoints are directed towards the peduncle node.

(Brooks, 2019). As shown in Fig. 2.3, four keypoints were labelled to express the pose of the peduncle node, where three points 'U', 'N', and 'D' were set at the upper, node, and lower positions of the main stem, respectively, and one point 'P' was set on the peduncle where it begins to bend. These four keypoints provide comprehensive pose information about the peduncle and the stem. The position and orientation of the peduncle are derived from the vector $\vec{NP}$, while those of the stem are determined by the vectors $\vec{DN}$ and $\vec{NU}$. Only the visible keypoints were labelled. It is important to note that due to the lack of visible features, the positions of points 'U' and 'D' were determined based on Euclidean distance to 'N' at 80 pixels. The distance was determined arbitrarily and did not consider the distance from the camera lens to the object. The images shown in Fig. 2.3 are not raw images with a resolution of 1280 x 720, but are cropped images of single peduncle nodes to give a clear presentation of the annotation results.

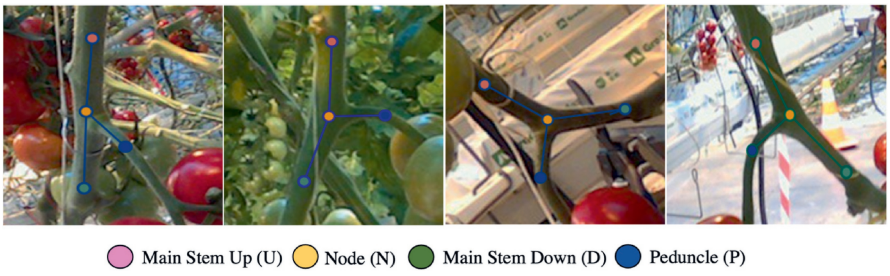


Figure 2.3: Samples of annotated images. Four keypoints were annotated to express the pose of the peduncle node. The 3 points 'U', 'N', and 'D' were set at the upper, node, and lower positions of the main stem, while 1 point 'P' was set on the peduncle.

2.2.1.3 Ground-truth measurement in 3D

To evaluate the pose-estimation method, the ground-truth angles between peduncle and main stem were manually measured. As shown in Fig. 2.1, the upper angle between the peduncle and the main stem was defined as α_{up}, which was calculated as $\alpha_{up} = \angle(\vec{NP}, \vec{NU})$, while the lower angle was defined as α_{down}, calculated as $\alpha_{down} = \angle(\vec{NP}, \vec{ND})$. Three people measured the angles with protractors, and the average value was used. The standard deviations between the observers for all node measurements ranged between 1.73°-7.64° for α_{up}, and between 0°-8.66° for α_{down}.

2.2.2 Peduncle node pose estimation

As shown in Fig. 2.1, the pose estimation system was designed with three stages: (1) 2D pose estimation, (2) point cloud projection, and (3) 3D pose estimation. The system takes the 2D colour images and the aligned point cloud as input and estimates the 3D pose of the peduncle node as output. The first stage uses a deep keypoint detector on colour images to localize the 4 pre-defined keypoints ('U', 'N', 'D', 'P'). In the second stage, the 2D keypoints are projected onto the point cloud to convert them into 3D Cartesian space. Finally, the 3D poses are estimated in the third stage. More details about the three stages are provided in the next subsections.

2.2.2.1 Keypoints R-CNN architecture

This study employed the Keypoints R-CNN network (He et al., 2020), based on the Detectron2 implementation (Wu et al., 2019b), for pose estimation on color images. Keypoint R-CNN takes a camera image as input and predicts the bounding box of each peduncle node, including the location of the four keypoints in the image. As shown in Fig. 2.4, the network is comprised of two stages. In the first stage, a ResNet50-FPN (50-layer Residual Network) (He et al., 2016) combined with a Feature Pyramid Network (Lin et al., 2017) was used as backbone to extract and fuse features from the input image. These feature maps were then processed by a region proposal network (RPN) (Ren et al., 2017) to generate a large set of regions of interest (ROI) that possibly contain target objects. In the second stage, the ROI proposals were used to crop the feature maps, which were subsequently processed by two independent networks: (1) an object-detection network featuring multiple Fully Connected (FC) layers to predict the object's bounding box, along with a class label and confidence score; and (2) a Fully Convolutional Network (FCN) with multiple convolutional layers to predict the location of the keypoints associated with the detected object.

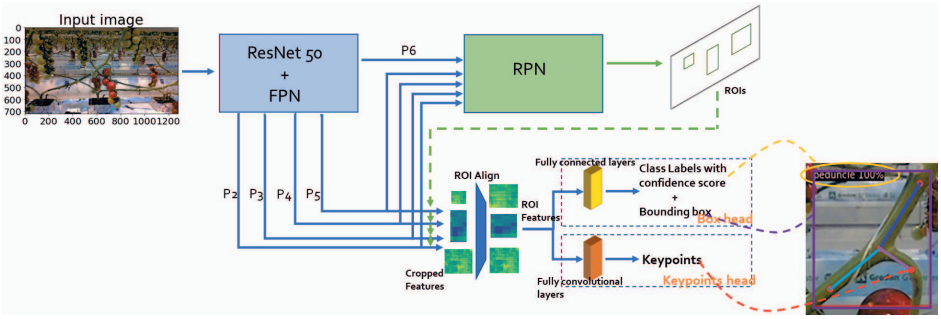


Figure 2.4: The structure of the Keypoints-RCNN algorithm. ResNet50-FPN is used to extract features, and RPN is utilized to generate ROI proposals by taking feature maps as input. The cropped ROI features are fed into two branches to predict class labels, bounding boxes, and keypoints.

2.2.2.2 Training of the network

The model was trained on a PC with an Nvidia-GPU RTX-3090 and Intel-CPU i9-10920X on Linux Ubuntu 18.04, utilizing a training set of 503 images, as described in section 2.2.1.1. To speed up the process, transfer learning was applied by using a pre-trained network on the COCO dataset (Lin et al., 2014) and fine tuning on collected training set. The training process consisted of 30,000 iterations with a batch size of 8, resulting in approximately 477 epochs. The Stochastic Gradient Descent (SGD) optimizer was used with a learning rate of 0.005 and a decay rate of 0.1 to optimize the network weights and biases to reduce the loss. During training, an anchor was considered positive if it had the highest intersection-over-union (IoU) or an IoU higher than 0.7, while anchors with IoU less than 0.3 were considered negative. To reduce the duplication of detections corresponding to the same ground truth, the Non-Maximum Suppression (NMS) was set to 0.7. During training, augmentation methods were employed to enhance the robustness of the model, including brightness with a random factor between 0.8-1.5, contrast with a random factor between 0.6-1.3, saturation with a random factor between 0.8-1.4, lighting through adding colour jittering with a random degree sampled from normal distribution with a standard deviation of 0.7, and horizontal flipping with a probability of 50%. The default anchor sizes of 32 and 512 were removed as the peduncle nodes in collected images were typically around 200 pixels in height and 140 pixels in width.

2.2.2.3 Peduncle-node detection

The trained keypoint detector has the capability to localize multiple peduncle nodes in the image. However, considering the harvesting robot can only harvest one tomato truss at a time in real application, when multiple peduncles were detected, the most

prominent node was selected as the target. This node was also used for comparison with the annotated ground truth during evaluation. The final peduncle node was determined based on the confidence value corresponding to the detection (Fig. 2.5a).



Figure 2.5: In figure (a), the detection with the highest confidence value was determined as the final target. In figure (b), the predicted keypoints of the selected target were extended to circular masks to include more pixels to be projected in the point cloud.

2.2.2.4 2D Keypoints Extension

After a peduncle node was detected, its keypoints were projected into 3D by using the spatial information from the aligned point cloud. However, the collected point clouds are usually sparse, the projection of only the four keypoints might fail due to missing corresponding points in the point cloud. To deal with this, the 2D keypoints were extended with circular masks (Fig. 2.5b) to include more pixels and all pixels inside the masks were projected to the point cloud (Fig. 2.6a-b). The use of a fixed circular mask radius for all peduncle nodes is arbitrary, as the diameter of the peduncle usually varies from node to node and is affected by different camera shooting distances. In some cases, a fixed radius can produce a suitable circular mask, but it can be too large for some nodes, extending beyond the range of the peduncle pixels and resulting in the background pixels being segmented, which can significantly affect the 3D pose estimation. According to Equation (2.1), to avoid segmenting extra pixels from the background, the radius r_m of the circular masks was determined by applying a weighting factor w to the diagonal d of the predicted bounding box of the peduncle node. This ensured the radius of the circular mask was always determined relative to the size of the peduncle node. w was set to 0.03 in the experiments through a preliminary test, resulting in an as large as possible radius without extending beyond the boundaries of the node/stem width. A higher ratio often resulted in background points being segmented.

$$r_m = w \cdot d \quad \text{[pixel]} \tag{2.1}$$

2.2.2.5 Point cloud projection

The colour and depth images captured by the camera were aligned. Using the depth data, the 3D coordinates, X_i, Y_i, Z_i, were determined for all pixels p_i that had valid depth data. Thus, the relation between the image coordinates, $p_i = \{x_i, y_i\}$, and the 3D coordinates was known. Figure 6a displays an example of a point cloud where the pixels within the keypoint masks in the colour image were projected into the point cloud. However, it can be seen in Fig. 2.6b-c that the raw point clusters were influenced by the outliers in the background. This caused the centroid of some clusters to deviate, see Fig. 2.6c for an example. Outlier points were removed in the next step.

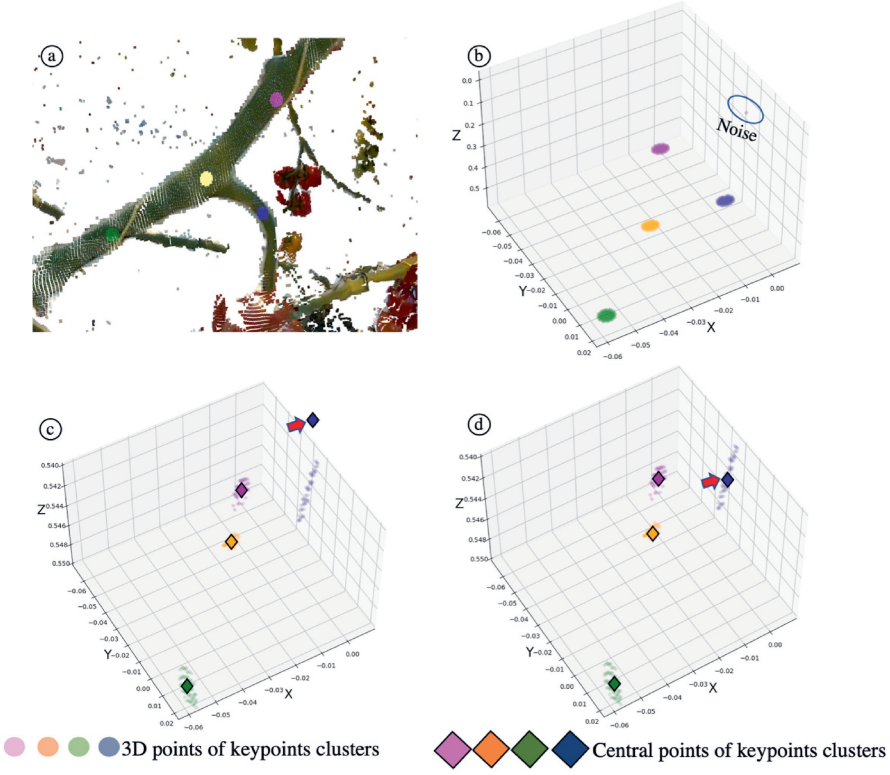


Figure 2.6: From 2D keypoints to 3D pose. a) show the masked keypoints in the colour image projected into the 3D point cloud, with each keypoint marked in a different colour. b) show the 3D points belonging to the keypoint clusters, containing outliers. c) show a zoom-in of b) with the centroids of each noisy clusters marked by diamonds. d) shows the centroids of the four clusters after removal of outliers.

2.2.2.6 Outliers Removal

The outliers were determined based on the density of the surrounding points. A point was considered an outlier using Equation (2.1). For each point i, the number of neighbours within a 3D sphere of radius r_f was counted. If there were fewer than the minimum required number of neighbours n_{min}, the point was handled as an outlier and removed. Consider n_p as the total number of points and d_{ij} as the distance from neighbouring point j to point i. The function ρ equals 1 when $d_{ij} < r_f$, otherwise it equals 0. During the experiments, the radius $r_f = 0.005$ [m] and the threshold $n_{min} = 10$ were used. These values were determined by a simple pilot experiment and by manually checking the result of noise removal.

$$\text{Outlier}_i = \begin{cases} 1, & \sum_{j=1}^{n_p} \rho(d_{ij} < r_f) < n_{min} \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

2.2.3 Evaluation

A test set containing 145 RGB-D images was used to evaluate the performance of the algorithm from 3 perspectives: (1) object detection, (2) keypoint detection, and (3) pose estimation. For each aspect, the overall performance of the model was evaluated on the entire test set (with images from all viewpoints), while its performance at a specific viewpoint was evaluated on images from the corresponding viewpoint. In evaluations (1) and (2), only the RGB images were used, and in evaluation (3), the aligned point clouds resulting from the depth images were used. The configurations of the thresholds for NMS and IoU in the testing phase were identical to those in the training phase. But the maximum-detection-per-image parameter was adjusted to 1 corresponding to the detection with the highest confidence. In other words, the evaluation was done for the most prominent peduncle node in the image.

2.2.3.1 Evaluation of object detection

The objective of this experiment was to evaluate the algorithm' s performance in detecting the peduncle nodes. A commonly used metric for object detection, the Average Precision (AP) and F1-socre, was applied to evaluate the performance. To judge if the detections were correct, a criterion of Intersection-over-Union (IoU) was employed to calculate the similarity between the predicted bounding box (B_p) and the ground-truth box (B_{gt}). It is by dividing the area of the intersection of the two bounding boxes by their union area:

$$IoU(B_P, B_{gt}) = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})} \tag{2.3}$$

The IoU ranges from 0 to 1, where a higher value represents a better match. Based on the calculated IoU, it is determined if a detection is a true positive (TP) or a false positive (FP), and if there are false negatives (FN). Specifically, if $IoU(B_P, B_{gt}) > \theta$, where θ is a predefined threshold, then the prediction is TP. A detection is FP if the $IoU(Bp, Bgt) \leq \theta$ or the annotated object is already detected by another prediction with a higher IoU value. If an annotated object is not detected, in other words, the IoU with all the predictions is lower than θ, then it is an FN. Based on the number of TP, FP, and FN, the value of precision and recall under θ can be calculated. The value of precision represents what fraction of the detections were correct out of all the detections while recall reflects what fraction of the targets were correctly detected.

$$\text{Precision}^{\theta} = \frac{\text{TP}^{\theta}}{\text{TP}^{\theta} + \text{FP}^{\theta}} \tag{2.4}$$

$$\text{Recall}^{\theta} = \frac{\text{TP}^{\theta}}{\text{TP}^{\theta} + \text{FN}^{\theta}} \tag{2.5}$$

Based on the value of precision and recall, the AP and F1-score can be calculated. According to Equation (2.6), AP is determined as the area under the interpolated precision-recall curve, where $R_1, R_2, \ldots, R_n$ are the recall levels, and interpolated precision $P_{interp}(R)$ is the maximum precision found at any recall level $\geq R$. F1-score can be calculated using Equation (2.7), which computes the harmonic mean of the precision and recall giving a more intuitive evaluation for the performance. Both the evaluations were conducted using three IOU thresholds θ at 0.5, 0.75, and 0.5:0.95.

$$AP@\theta = \sum_{i=1}^{n} (R_i^{\theta} - R_{i-1}^{\theta}) P_{interp}^{\theta} \tag{2.6}$$

$$\text{F1-score}@\theta = 2 \cdot \frac{\text{Precision}^{\theta} \cdot \text{Recall}^{\theta}}{\text{Precision}^{\theta} + \text{Recall}^{\theta}} \tag{2.7}$$

2.2.3.2 Evaluation of keypoint detection

This experiment aimed to evaluate the performance of the keypoint detector in detecting pre-defined keypoints of the peduncle nodes. Four keypoints were expected to be detected for each node. The detection performance was evaluated using the Percentage of Detected Joints (PDJ) metric, also known as Percentage of Correct Keypoints (PCKh) in some studies (Andriluka et al., 2014; Li et al., 2019; Russello et al., 2022). Equation (2.8) was used to calculate the PDJ. A predicted keypoint was considered correctly detected if the Euclidean distance (d_i) between it and the ground truth was smaller than a certain fraction f of the diagonal d of the predicted bounding box. Accounting for the object diagonal allowed the method to handle nodes of arbitrary

sizes. The term n referred to the number of keypoints on a node. And $\sigma(x) = 1$ if $x \leq 0$, otherwise $= 0$.

$$PDJ@f = \frac{1}{n}\sum_{i=1}^{n}\sigma(d_i - f \cdot d) \cdot 100 \quad [\%] \tag{2.8}$$

In this analysis, the PDJ values were calculated for three fractions f: 0.05, 0.1, and 0.2. The smaller the value of f, the closer the predicted points need to be to the ground truth keypoints to be considered correct. The analysis was conducted from two perspectives: (1) detection-wise and (2) keypoint-wise. The detection-wise evaluation calculated the number of detections that achieved different PDJ values. The PDJ of a detection was 100% all 4 keypoints were correctly detected, and 0% if none of the keypoints were detected correctly (Equation (2.8)). The keypoint-wise evaluation calculated success rate per keypoint type, with a higher ratio indicating a higher successful detection rate.

2.2.3.3 Evaluation of pose estimation

Using the 3D keypoints generated in section 2.2.2.5 after removing outliers, the 3D pose of the peduncle node was estimated. The 3D pose was defined to include the node position, orientation, and relative angles. However, measuring the complete ground truth was challenging, so only the relative angles α_{up} and α_{down} were evaluated. The angels were calculated using Equation (2.9):

$$\alpha = \arccos\left(\frac{\vec{A} \cdot \vec{B}}{|\vec{A}| \cdot |\vec{B}|}\right) \tag{2.9}$$

for any two given vectors $\vec{A}$ and $\vec{B}$ in 3D space, their included angle α equals their dot product divided by their magnitudes. For 4 keypoints 'U', 'N', 'D', and 'P', the α_{up} was calculated using vectors $\vec{NU}$ and $\vec{NP}$, while α_{down} was calculated by $\vec{ND}$ and $\vec{NP}$.

Two primary sources of error were considered in the estimated angles: (1) pose estimation in 2D, resulting from incorrect localization of keypoints, (2) pose estimation in 3D, including 3D keypoints projection, noise filtering, and pose estimation. The errors resulting from limitations in the sensor's capability were not considered, such as insufficient depth measurements leading to bad quality of generated point cloud, as these would not reflect the performance of the algorithm. After removing the data where 3D information was not adequately captured, 134 test samples remained for analysis.

To gain a deeper understanding of the source of errors, the accuracy of three types of predicted angles was evaluated: (1) $\hat{\alpha}^{anno}$ is the predicted angle based on the

annotated keypoints, so with perfect keypoint detection, which reflect errors caused in 3D, (2) $\hat{\alpha}^{kp}$ is the predicted angle based on the predicted keypoints, which includes errors in keypoint detection and 3D calculation , and (3) $\hat{\alpha}^{4kp}$ represents the predicted angles only when all 4 keypoints were correctly detected, which reflects the optimal performance that the proposed method can achieve. To compare to, α represents the ground-truth angle manually measured in the greenhouse.

Two evaluation metrics, Mean Absolute Error (MAE) and Mean Relative Error (MRE), were used to evaluate the accuracy of the estimated angles. The MAE is calculated as the mean of the absolute errors between the predicted angle, $\hat{\alpha}_t$, and the ground-truth angle, α_t, for peduncle t, as shown in Equation (2.10). The MRE was calculated as the mean of the relative errors obtained by dividing the absolute errors with the corresponding ground-truth angle, as shown in Equation (2.11). Here, T denotes the total number of detections.

$$MAE = \frac{1}{T}\sum_{i=0}^{t}|\hat{\alpha}_t - \alpha_t| \quad [^\circ] \tag{2.10}$$

$$MRE = \frac{1}{T}\sum_{i=0}^{t}\frac{|\hat{\alpha}_t - \alpha_t|}{\alpha_t} \quad [\%] \tag{2.11}$$

2.3 Results

The results of the evaluation in terms of object detection, keypoint detection, and 3D angle estimation were displayed in section 2.3.1, section 2.3.2, and section 2.3.3 respectively.

2.3.1 Node detection

Table 2.2 presents the overall AP and F1-score across all viewpoints for different IoU thresholds (0.5, 0.75, and 0.5:0.95). The method achieved outstanding results for both metrics on IoU0.5, with AP@0.5=0.96 and F1-score@0.5=0.98, indicating the model can successfully detect most of the peduncles under a relatively tolerant intersection-over-union criterion. However, both metrics experienced a significant decrease for a stricter threshold (IoU0.75), suggesting that the bounding-box predictions of the complete peduncle were not very accurate with a mismatch to the ground-truth bounding box. It should be noted that the peduncle node, as we defined it, is poorly represented by a bounding box, so it is not unexpected that the accuracy of the bounding-box prediction is not so high. The bounding-box predictions are not use for further operations, but instead use the keypoint detections discussed in the next subsection.

Table 2.2: Overall results of AP and F1-score across all viewpoints.

Metrics	IoU0.5	IoU0.75	IoU0.5:0.95
AP	0.96	0.37	0.45
F1-score	0.98	0.53	0.54

According to Table 2.3, the algorithm accurately detected nodes from all 5 viewpoints, with an AP@0.5 greater than 0.91 and an F1-score@0.5 greater than 0.95 for all viewpoints, reaching 1.0 for three of the five viewpoints. This suggests the algorithm can deal with the variations caused by different viewpoints. Specifically, Viewpoints 1, 2, and 3 achieved perfect node detection, while Viewpoints 4 and 5 had lower accuracy (refer to Fig. 2.7 for visual samples cropped from the raw images). This is likely because the lateral views can introduce more shape variations in the peduncle node, making it more difficult for the algorithm to accurately detect. When looking at IoU0.75 and IoU0.5:0.95 thresholds for both the metrics, larger differences can be seen. Viewpoint 2 exhibited the highest values across all metrics, indicating that the top view is the optimal viewpoint for tasks such as harvesting. Viewpoint 3 exhibited a significantly lower value. One possible reason could be the impact of lighting on the images, as this viewpoint is more susceptible to lighting variations, which affects the algorithm's ability to detect the bounding-box of the peduncle nodes accurately. To provide a clearer illustration, Fig. 2.8 displays a sample with detection results from Viewpoint 3.

Table 2.3: The AP and F1-score obtained from 5 different viewpoints.

Viewpoint	AP			F1-score		
	IoU0.5	IoU0.75	IoU0.5:0.95	IoU0.5	IoU0.75	IoU0.5:0.95
View1	1.0	0.33	0.48	1.00	0.47	0.54
View2	1.0	0.86	0.67	1.00	0.89	0.71
View3	1.0	0.08	0.30	1.00	0.16	0.37
View4	0.94	0.47	0.46	0.95	0.58	0.52
View5	0.91	0.17	0.35	0.95	0.21	0.38

2.3.2 Keypoint detection

The evaluation of detected keypoints was performed on three fractions (f) of 0.05, 0.1, and 0.2. Table 2.4 displays the count of detections achieving different PDJ values, a higher PDJ value indicates more keypoints were correctly detected. The PDJ@0.2 and PDJ@0.1 display a similar distribution, with the most of detections achieving a PDJ of 100% (124 detections for PDJ@0.2 and 68 detections for PDJ@0.1), while

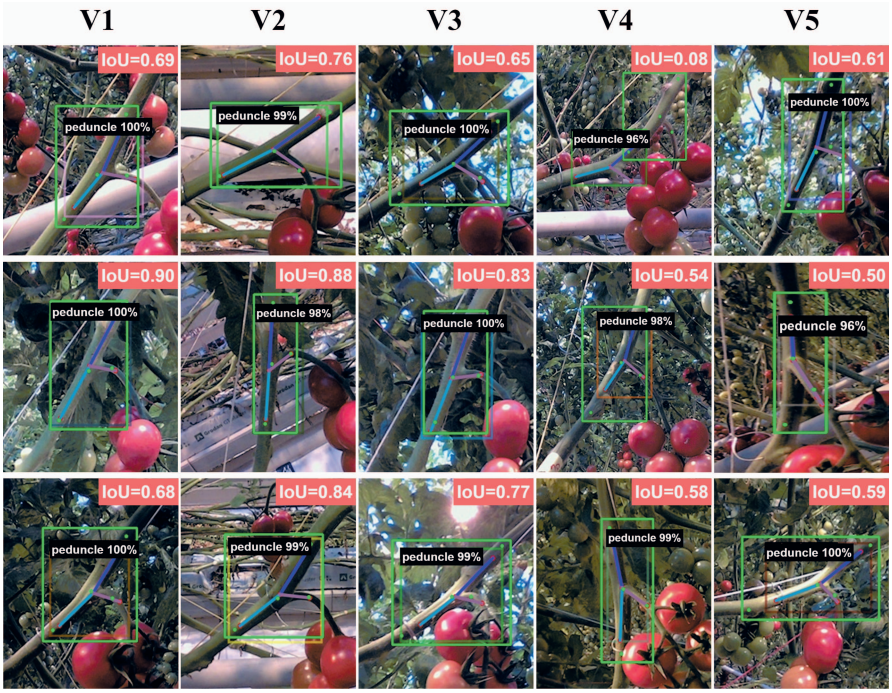


Figure 2.7: Illustration of node detection results for 3 peduncle nodes (rows) from 5 viewpoints (columns). All images are cropped from the raw images to give a better illustration. The keypoints and bounding box in green colour indicate the ground truth. The IoU value between the predicted and ground truth bounding boxes of the detection is calculated and plotted in the pink box.

the smallest number of detections achieved a PDJ smaller than 50% (3 detections for PDJ@0.2 and 9 detections for PDJ@0.1). In contrast, for PDJ@0.05, only 19 detections achieved PDJ=100%, while a large number of detections (45) achieved a PDJ smaller than 50%. The significant shift in the ratio when reducing f from 0.2 to 0.05 is that the smaller fraction f requires the detected keypoints be closer to the ground truth to be considered correct, which resulted in most of the detected keypoints considered correct under $f = 0.2$ being considered as failed under a more stringent criterion (f = 0.05).

The results of PDJ@0.2 and PDJ@0.1 indicate the feasibility of the model, which essentially located the keypoints on the right locations. But some detections are not accurate enough in specific keypoints or scenarios to fulfil a more stringent criterion. To investigate the sources of error, further analysis was carried out by calculating the PDJ for each individual keypoint.

As shown in Table 2.5, the 4 keypoints 'U', 'N', 'D' and 'P' achieved PDJ@0.2 at



Figure 2.8: Illustration of the raw image used for 2D pose estimation, captured from viewpoint V3. The model faces challenges including accurately identifying peduncle nodes among similar plant parts (petiole nodes), distinguishing between foreground and background, and dealing with changing light. The keypoints and bounding box in green colour indicate the ground truth.

Table 2.4: The number of images that achieved different PDJ values are counted at three fractions, 0.05, 0.1, and 0.02 independently.

Metrics	<50%	50% (2/4)	75% (3/4)	100% (4/4)
PDJ@0.05	31.03%(45)	37.35%(44)	25.52%(37)	13.10%(19)
PDJ@0.1	6.21%(9)	23.45%(34)	23.45%(34)	46.90%(68)
PDJ@0.2	2.07%(3)	2.07%(3)	10.34%(15)	85.52%(124)

91.03%, 97.93%, 91.72%, and 96.55% respectively, suggesting the model was capable of accurately localizing the keypoints and achieving a success rate above 90% for all the keypoints. The corresponding PDJ values decreased at varying levels by 60%, 8.96%, 57.93%, and 35.17% when reducing f from 0.2 to 0.05. The slight reduction of keypoint 'N' indicates it could be consistently localized close to the ground truth, and hence, its results were not largely affected even under a stricter criterion. In contrast, the significant decrease of keypoints 'U' and 'D' at PDJ@0.05 suggests these two keypoints usually deviated largely from the ground truth, and thus were no longer considered correct when f was decreased to 0.05. The keypoint 'N' achieved the highest PDJ in all three fractions, implying the most accurate detection, while the keypoint 'U' and 'D' showed significantly lower success rates, and their gap with the keypoint 'N' became even larger when decreasing f.

As described in section 2.2.1.2, keypoints 'N'and 'P'were determined based on unique

visual features ('N' linking the peduncle and the main stem; 'P' normally bending), while keypoints 'U' and 'D' were annotated based on the Euclidean distance. Based on the results, it is suggested that the visual features are more effective for the model to learn and recognize compared to the distance features.

Table 2.5: The PDJ of different keypoints for the entire testset with 145 images under 3 fractions of 0.05, 0.1, and 0.2.

Metrics	U	N	D	P
PDJ@0.05	31.03%	88.97%	33.79%	61.38%
PDJ@0.1	57.93%	97.93%	60.69%	92.41%
PDJ@0.2	91.03%	97.93%	91.72%	96.55%

Fig. 2.9 displays some samples in which the localizations of the keypoints 'U' or 'D' were considered incorrect due to large deviations from the ground truth. However, all these localizations were found to be correctly assigned to the corresponding plant parts. Based on this, it is suggested that the results can be further improved if the system is tasked with accurately locating the different landmarks without necessarily requiring proximity to the ground truth.

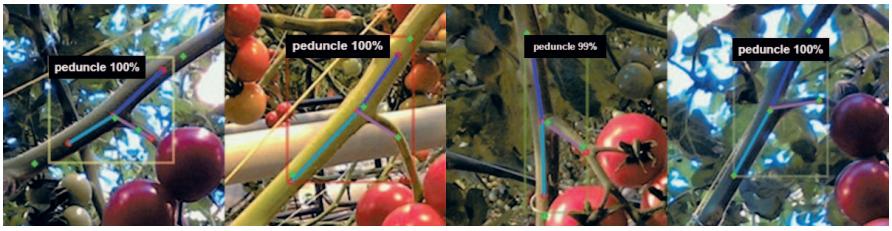


Figure 2.9: Illustration of some detections failed to localize the keypoints 'U' and 'D' due to their larger deviations from the ground truth. The keypoints and bounding box in green colour indicate the ground truth.

Table 2.6 presents keypoints detection results from different viewpoints, with an average PDJ@0.2 of over 89.47% for all views and two views reaching above 97.37%. View1 and View2 outperformed the other views, making them ideal for keypoints detection in practical applications. Keypoints 'N' and 'P' consistently showed significantly higher PDJ values compared to 'U' and 'D' across all views. Notably, View5 displays lower PDJ for 'P'. This is because in the greenhouse used in this study, the peduncles tend to grow towards the same direction, which is roughly View5, leading to significant shape variations and challenging keypoints detection. The reduced detection accuracy of 'P' can largely impact angle estimation, as its precise 3D position is crucial for estimating both angles, α_{up} and α_{down}.

Table 2.6: The PDJ obtained by the algorithm at different viewpoints under fraction equals 0.2.

Viewpoints	U	N	D	P	Mean
View1	100%	100%	94.74%	100%	98.68%
View2	94.74%	100%	94.74%	100%	97.37%
View3	73.68%	100%	94.74%	100%	92.11%
View4	89.47%	94.74%	78.95%	94.74%	89.47%
View5	94.74%	94.74%	89.47%	89.47%	92.11%
Mean	90.53%	97.89%	90.53%	96.84%	

2.3.3 3D Pose estimation

Based on the results presented in Fig. 2.10, $\hat{a}^{anno}$ achieved a MAE of 8° and 7° for α_{up} and α_{down}, respectively, with a MRE of 12% and 6%. These values show that even when all keypoints are accurately localized, errors can still occur due to inaccuracies in the 3D pose estimation, relating to, for instance, noise in the depth data. These errors serve as a minimum level of error that can be achieved by the full algorithm. When including the 2D pose estimation and evaluating those cases where all 4 keypoints were correctly estimated, it can be seen that $\hat{a}^{4kp}$ shows a slightly higher error with MAE of 8° and 7° for α_{up} and α_{down}, respectively, and an MRE of 13% and 6%. This increase in error results from small inaccuracies in predicting the keypoint locations. Including all cases of 2D pose estimation, the error for $\hat{a}^{kp}$ increases further to an MAE of 11° and 10° for α_{up} and α_{down}, respectively, and an MRE of 17% and 8%. The larger errors in $\hat{a}^{kp}$ result from some additional inaccuracy in the 2D keypoint estimation. The figure suggests that the largest error in 3D pose estimation stems from the final step in the method.

Compared to α_{down}, α_{up} showed a slightly larger MAE but a significantly larger MRE. This is because the actual angle of α_{up} is generally much smaller, and as a result, the MRE can be amplified when the MAE is similar to that α_{down}.

It is important to note that not all 134 detections were considered in the evaluation of $\hat{a}^{kp}$, as there were cases where α_{up} or α_{down} were incomputable. For example, keypoints localized on the background due to incorrect keypoints detection usually lack 3D information, which can make the angles incomputable. In this evaluation, only one case was excluded for this reason, so a total of 133/134 cases were included. For $\hat{a}^{4kp}$, 115 detections were used.

Table 2.7 displays the results of angle estimation acquired from different viewpoints. View1 showed the lowest errors, followed by View2 and View3, which had a vertical

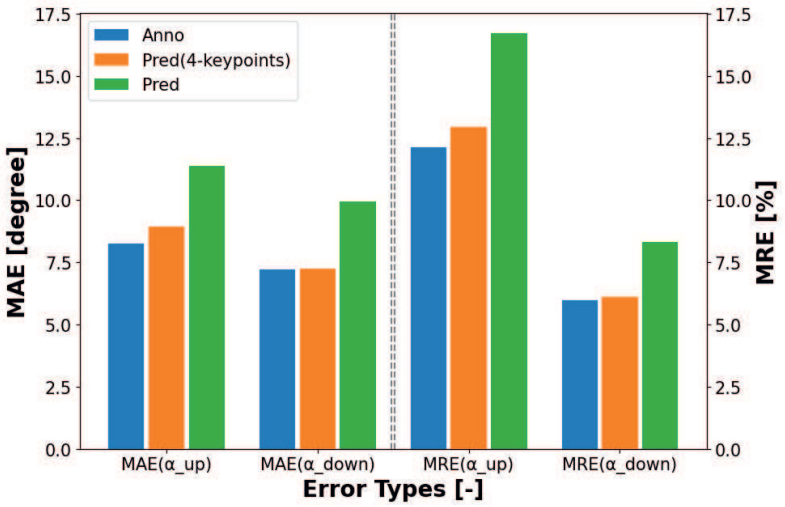


Figure 2.10: Three kinds of angles were evaluated on MAE and MRE. The 'Anno' represents the errors contained in $\hat{a}^{anno}$, the 'Pred' represents the errors contained in $\hat{a}^{kp}$, while 'Pred(4-keypoints)' represents the errors contained in $\hat{a}^{4kp}$ only when 4 keypoints are correctly detected under f=0.2.

deviation from View1, lower than View4. However, View5 exhibited significantly higher errors, mainly due to the low detection accuracy of 'P' at this view (refer to section 2.3.2), suggesting that this view should be avoided in real applications.

Table 2.7: Errors achieved by $\hat{a}^{kp}$ in angle estimation at different viewpoints. The values were kept to integers considering the precision of manual measurements in real environments.

Viewpoint	MAE (up)	MAE (down)	MRE (up)	MRE (down)
View1	9°	6°	7%	5%
View2	11°	9°	10%	8%
View3	9°	7°	8%	6%
View4	12°	11°	9%	9%
View5	21°	23°	18%	19%

2.4 Discussion

2.4.1 General evaluation of proposed method

The proposed method demonstrated a high success rate in detecting the peduncle nodes. Although the nodes were detected well, the exact bounding-box coordinates

often did not accurately match the ground truth. This is mainly because keypoints 'U' and 'D' were annotated based on distance and not based on a clear image feature, which the model struggled to effectively learn. This was further evident in the evaluation of the keypoint detection. While the method displayed high accuracy across all types of keypoints, 'U' and 'D' showed significantly lower PDJ compared to 'N' and 'P'. Since the proposed method directly performs 3D pose estimation based on the detected keypoints, the accuracy of the 3D pose estimation relied on the accuracy of the 2D pose detection.

The proposed method demonstrated capability in handling variations caused by different viewpoints with respect to the peduncle node. The node detection results showed an AP ranging between 0.94-1.00, while the F1 score ranged between 0.95-1.00, indicating high detection performance for all viewpoints, with View 1-3 showing the best performance. Similarly, the results of the keypoint detection showed a high consistency over viewpoints, with PDJ between 89.47% and 98.68%, with View 1 and 2 resulting in the best performance and View 4 showing the lowest performance. The accuracy of the pose estimation showed more variation between viewpoints, with errors ranging between 6° and 12° for Views 1-4 and errors between 21° and 23° for View 5. Overall, View 1 and 2 showed to provide the best view on the node. While the node and keypoint detection for View 5 were good in general, it did show a lower PDJ for the node point 'P'. As the 3D location of this point is crucial to estimate the two pose angles, this explains the lower pose-estimation accuracy for this view. View 5 is the rightwards view, which relates to the view where the tomato truss is closer to the camera than the node point 'P', possibly partially blocking a good view on the node.

2.4.2 Future improvements

To evaluate the algorithm's performance in a realistic scenario, all test data were collected from a natural greenhouse. However, the complex environment posed challenges for object annotation and ground-truth measurement, as it was impossible to cover all the nodes in the background. To find a solution, only the main node in each image was considered, matching the prediction that gained the maximum confidence value. Focusing only on the most confident prediction ignored other predictions based by the method. However, for the use of robotic harvesting, that is justified, as the robot will harvest the trusses one by one anyway. Occasionally, a node in the background was detected, which was evaluated as a FP. A similar problem was seen in the experiment of Rong et al. (2022), where the models were trained to detect only the fruits and peduncles of tomato plants in the foreground, but targets in the background were still likely to be detected because both had quite similar traits. In future

work, depth information can be integrated as input for the pose-estimation method to ignore the nodes in the background. This would force the model to only focus on objects in a certain depth range.

As in normal in commercial production, the leaves surrounding the peduncle node were removed before the fruit ripened. Consequently, situations where the peduncle node was occluded by the leaves were not considered. The deep-keypoint-detection method can inherently handle a certain level of occlusions and estimate the object' s pose based on a subset of the visible keypoints. However, if the nodes are significantly occluded, the current method may encounter some limitations. In such cases, utilizing the information from multi-views is a popular solution, it makes information that is not perceivable from a single view become available, and reduces perception uncertainty (Hemming et al., 2014; Rapado-Rincón et al., 2023; Shi et al., 2019; van Henten et al., 2003). To increase efficiency, next-best-viewpoint (NBV) methods can be used that propose camera poses to maximize the information gain (Burusa et al., 2024a; Mendoza et al., 2020; Zeng et al., 2020). As the results showed, some viewpoints are more favourable for a good peduncle-node detection than others. In future work, a method that integrates NBV planning with the keypoint-based pose estimator can be proposed to change camera poses that maximize the view on the peduncle.

Due to the lack of visual features, the keypoints 'U' and 'D' were determined based on pixel distance, which was expected to be learned by the model. However, as shown in Table 5, these two keypoints presented significantly lower PDJ values compared to keypoints 'N' and 'P'. Figure 9 also suggests that keypoints 'U' and 'D' were likely considered incorrect due to their large deviations from the ground truth, even though they were localized on the correct plant part and could be used to calculate the angles. So, it is considered that the traditional PDJ metric may not the most suitable way to evaluate these two keypoints. On the other hand, keypoints are possible not the best possible representation for these points. Instead, the DNN could be adapted to predict a vector up the main stem and a vector down the main stem instead of two keypoints.

In section 2.2.2.3, circular masks with a radius of r_m (Equation 2.1) were used to expand the keypoints to include more pixels. The value of r_m was self-adjusted based on the diagonal (d) of the predicted bounding box and a constant weighting factor (w) to avoid segmenting the pixels in the background. This approach was found to significantly reduce errors in 3D pose estimation compared to using a single pixel. However, beyond a certain range (0-0.05), increasing the value of w caused a negative effect, as pixels in the background were added. The sensitivity of the model to w will reduce its robustness. This problem is considered can be solved by employing a better strategy for expanding keypoints including 3D distance information, or by growing

only inside the mask of the node.

In Fig. 2.10, the errors of the estimated angles did not reduce to very low even when all keypoints were accurately located ($\hat{a}^{anno}$ and $\hat{a}^{4kp}$), suggesting potential errors in the ground truth measurements. The measurement of a was subjective as three measurers manually located keypoints and measured angles with a protractor. The large standard deviations in α_{up} and α_{down} (section 2.2.1.3) also highlight significant deviations in measurements. To improve accuracy and reduce time consumption, using $\hat{a}^{anno}$ directly as the ground truth is recommended in future experiments.

The analyses of the influence of the viewpoint with respect to the node suggests that some viewpoints are better than others. Particular V1 (canonical view) and V2 (the higher view) resulted in higher detection and pose-estimation performance. In future work, active-vision methods should be explored to actively guide the robot to the right viewpoint to make the most accurate detection.

2.4.3 Comparison with related work

Boogaard et al. (2020) utilized YOLO v3 (Redmon & Farhadi, 2018) to detect cucumber internodes using multi-view imaging. The comparable results were achieved if it comes down to the node detection, although a direct comparison is not possible due to differences in experimental conditions, tasks, and datasets. It is worth noting that their experiments were conducted in a controlled environment, while ours occurred in a commercial greenhouse, posing more challenges for node detection. They investigated the benefits of multiple viewpoints in object detection, which is what can be explored for keypoints detection method in the future work. While their work only detected the nodes, the method proposed in this study also estimated the pose of the peduncle node.

The keypoint detection results obtained in this study are consistent with the work of Zhang et al. (2022), who used a DL-based keypoint detector for tomato bunches. Both studies found that keypoints 'U' and 'D' had significantly lower PDJ values compared to 'N' and 'P'. While our study used a stricter evaluation criterion based on the weighted size of the node, they used the weighted size of the entire tomato bunch, allowing for greater deviation in their predicted keypoints to still be considered correct. Nevertheless, our method demonstrated higher PDJ values for 'N' and 'P', confirming its effectiveness in keypoints detection. In comparison to the method of Kim et al. (2023), the proposed method in this study achieved better results with an average PDJ@0.2 = 94.31% across all types of keypoints, outperforming their results of PDJ@0.5 = 92.9%, even under a more stringent criterion. However, it must be noted that a one-on-one comparison of results is not possible due to differences

in targets, criteria, and datasets. In addition, compared to their work, which used zoomed-in images with only one tomato bunch per image, the images used in this research were originally captured by the camera, containing many tomato plants in the background and changing light, making the task more challenging (refer to Fig. 2.8). On the other hand, their method included also the location of the individual tomatoes in the truss, which makes their task more challenging. Similar to Kim et al.' s method, the proposed method also has the capability to detect the pose of multiple peduncle nodes simultaneously. However, only the most prominent nodes were used in the evaluation, as a robot would harvest trusses one by one.

The proposed method for 3D pose estimation was compared with the work of Luo et al. (2022), who used Mask R-CNN for target detection in RGB images and estimating peduncle orientations by mapping the detections to the aligned point clouds. Direct comparison of the results is not possible due to differences in evaluation metrics and targets. Their method focused solely on peduncle orientation, ignoring relative angles. Furthermore, The proposed method estimated the pose by processing only the keypoint clusters, making it more efficient and simpler than their method, which processed the entire point cloud. However, their method, by using more complete point cloud can be less sensitive to 2D object detection accuracy and point cloud quality. Zhang et al. (2023a) evaluated the accuracy of their predicted 3D keypoints compared to human annotations in the point cloud, but they did not evaluate their method with real-world measurements as this work did. By doing so, valuable insights into the error sources in different steps in the pipeline were allowed to be acquired in this research. The contribution to the 3D pose error of the 2D keypoint detection and the integration of the 3D data could be shown, as displayed in Fig. 2.10. Although their method to predict 3D keypoints directly from the image data using a deep neural network is promising, it requires the availability of sufficient amount training data. In future work, this end-to-end approach will be investigated and compare the performance of both approaches as a function of the amount of available training data.

To the best of knowledge, there are no other studies investigating the impact of different viewing angles on the accuracy of pose detection. According to the results, for this crop, views 1 and 2 provide a higher performance, which is valuable input for future work on robotic harvesting of tomatoes.

The proposed method demonstrated good performance in 3D pose estimation of peduncle nodes, making it suitable for acquiring comprehensive information for robotic harvesting. However, defining precise accuracy requirements for robotic harvesting poses challenges due to limited research and variations in crops, harvesting mechanisms, and other factors. For example, the design of grippers and cutters in a robotic

harvester can tolerate some level of error, reducing the accuracy requirements for the vision system significantly (Zhang et al., 2022).

The impact of post-harvest peduncle node structure on keypoint detection was not considered in this project. This is because, to minimize plant infection, tomato trusses are typically detached as close as possible to the main stem, between the 'N' and 'P' keypoints. Consequently, the residual peduncle length is shorter than the annotated distance from 'N' to 'P', and thus does not affect keypoint detection. If this becomes a concern in other application scenarios, the issue can be addressed by including post-harvest nodes in the training set.

A test set comprising 145 images with aligned point clouds was used to evaluate the proposed method. The size of the test set is similar to Kim et al. (2023)), who used 136 images for evaluation, but smaller than that of Zhang et al. (2022), with 400 images. The test set used in this study, however, consist of representative set of images from an actual commercial tomato greenhouse, including cluttered scenarios, variation in the appearance and pose of the peduncle nodes, different viewing angles and natural variations in illumination. In addition, and different from related work, the manual reference measurements performed in the greenhouse were included to evaluate the accuracy of the method in the real world.

2.5 Conclusions

In this study, an innovative method was proposed to estimate the 3D pose of tomato peduncle nodes from RGB-D images. The method provided complete 3D pose information, including the position, orientation, and relative angles to the main stem, which are needed for robotic operations. Meanwhile, this method is expected to benefit other agricultural operations using robotics, such as phenotyping and monitoring, with the output of comprehensive information. After being analysed from the three aspects of (1) object detection, (2) keypoint detection, and (3) pose estimation, the feasibility of the proposed algorithm was shown. For each evaluation, the impact of viewpoint angles was investigated, indicating the model can handle a level of variations caused by view change. The evaluation resulted in the following conclusions:

- For object detection, the proposed algorithm achieved outstanding results in both AP@0.5 and F1-score@0.5, suggesting its good capability in node detection. However, there was a significant drop in both metrics for the IoU threshold of 0.75, indicating that most of the bounding-box predictions do not accurately match the ground truth.

- Keypoint detection resulted in above 90% of PDJ@0.2 for all keypoints, demon-

strating the model's capability to accurately localize the keypoints. However, there is a significant drop in PDJ@0.05 for 'U' and 'D' keypoints, indicating these two keypoints are localized with less accuracy. Nevertheless, as discussed earlier, some predicted keypoints considered incorrect can still be localized on the correct plant parts.

- For pose estimation, the errors of $\hat{a}^{4kp}$ displayed a significant decrease compared to $\hat{a}^{kp}$, indicating a large proportion of the errors in 3D pose estimation were due to incorrect keypoint detection in 2D. However, even if all the keypoints were accurately detected, there was still a level of error, suggesting the measurement of a might not be very accurate. Therefore, in future experiments, the $\hat{a}^{anno}$ could be directly used as the ground-truth, which can save significant time in actual angle measurement.

- The methods achieved high AP@0.5 (>0.91) in object detection and PDJ@0.2 (>0.89) in keypoint detection for all viewpoints, indicating robustness to variations caused by view changes. View1 (canonical view) and view2 (higher view) yielded the best results and are recommended for practical applications.

While the proposed method was tested for the 3D pose estimation of tomato peduncles, the method has potential to be applied to other similar crops, such as cucumber, zucchini, and bell pepper and for other plant parts, such as the pedicel (e.g., for deleafing) and fruit clusters (e.g., for monitoring tasks). Besides robotic applications, the method can also provide valuable information for plant monitoring and plant phenotyping.

Chapter 3

SSL-NBV: A Self-Supervised-Learning-Based Next-Best-View algorithm for Efficient 3D Plant Reconstruction by a Robot

This chapter has been published as:

Ci, J., van Henten, E. J., Wang, X., Burusa, A. K., & Kootstra, G. (2025). SSL-NBV: A self-supervised-learning-based next-best-view algorithm for efficient 3D plant reconstruction by a robot. *Computers and Electronics in Agriculture*, 233, 110121. https://doi.org/10.1016/j.compag.2025.110121

Abstract

The 3D reconstruction of plants is challenging due to their complex shape causing many occlusions. Next-Best-View (NBV) methods address this by iteratively selecting new viewpoints to maximize information gain (IG). Deep-learning-based NBV (DL-NBV) methods demonstrate higher computational efficiency over classic voxel-based NBV approaches but current methods require extensive training using ground-truth plant models, making them impractical for real-world plants. These methods, moreover, rely on offline training with pre-collected data, limiting adaptability in changing agricultural environments. This paper proposes a self-supervised learning-based NBV method (SSL-NBV) that uses a deep neural network to predict the IG for candidate viewpoints. The method allows the robot to gather its own training data during task execution by comparing new 3D sensor data to the earlier gathered data and by employing weakly-supervised learning and experience replay for efficient online learning.

Comprehensive evaluations were conducted in simulation and real-world environments using cross-validation. The results showed that SSL-NBV required fewer views for plant reconstruction than non-NBV methods. It achieved IG prediction in 0.0038 seconds, making it over 800 times faster than a voxel-based NBV, and an online learning iteration in 0.099 seconds. SSL-NBV reduced training annotations by over 90% compared to a baseline DL-NBV. Furthermore, SSL-NBV could adapt to novel scenarios through online fine-tuning. Also using real plants, the results showed that the proposed method can learn to effectively plan new viewpoints for 3D plant reconstruction. Most importantly, SSL-NBV automated the entire network training and uses continuous online learning, allowing it to operate in changing agricultural environments.

3.1 Introduction

Greenhouse production is a widely used method for vegetable cultivation, offering substantial benefits such as an extended plant production period, improved quality, and quantity, and finally leading to increased profitability for growers. The selection of suitable cultivars, i.e. phenotyping, as well as the crop production process itself, including tasks like harvesting and de-leafing, require a significant amount of labor. This limits large-scale greenhouse production because experienced labor is scarce and labor costs are high. Automating these labor-intensive tasks using robotics can help reduce reliance on human labor and potentially increase greenhouse productivity and production efficiency.

Robotic phenotyping and greenhouse operations like harvesting and leaf removal, are essentially based on the same key functionality, i.e. perception. However, accurate robotic perception in complex greenhouse environments is challenging due to the presence of occlusion and variation. Occlusion is mainly caused by leaves, preventing sufficient information collection, thus increasing uncertainty in robotic operations. Variations arise from dynamic growing environment (e.g., lighting) and inherent plant properties (e.g., shape, size, and texture), making systems that may work under certain conditions but lack robustness to variation. The objective of this study was to develop a robust robotic perception system to capture sufficient information of the plants under the presence of occlusion. The system was specifically developed and evaluated in plant phenotyping scenario for plant reconstruction tasks, but with the potential to be used for other tasks such as robotic harvesting after small modifications and fine-tuning.

Plant phenotyping is a set of methodologies to measure the plant traits, such as leaf angle, node length, and leaf area, and subsequently link these measurements to plant genotypes and growing environments, which is a key technology in plant breeding for efficient selection of target cultivars (Poland & Rife, 2012). Compared to traditional manual assessment of plant traits, robotic phenotyping methods have received significant interests by integrating machine vision techniques and robotics, showing potential of plant phenotyping in an automatic, accurate, non-destructive, and high-throughput manner (Atefi et al., 2021; Hartmann et al., 2011; Polder & Hofstee, 2014; van der Heijden et al., 2012). Two-dimensional (2D) robotic phenotyping methods have been widely studied (Jansen et al., 2009; Minervini et al., 2014; Tisné et al., 2013). However, a significant disadvantage of these methods is they cannot accurately measure plant traits that are expressed in three dimensions (3D) such as plant volume. Alternatively, three-dimensional (3D) methods perform measurement in 3D data, offering more comprehensive and accurate information of plant traits (Boogaard

et al., 2023; Shi et al., 2019). An essential first step of 3D phenotyping methods is generating a 3D digital reconstruction of the plants, which can be represented as a point cloud (Boogaard et al., 2022, 2021), or a mesh (Thapa et al., 2018; Vázquez-Arellano et al., 2018). However, accurate 3D plant reconstruction is extremely challenging due to significant plant occlusion. A common approach to solve this problem is using multiple viewpoints (Golbach et al., 2016; Lu et al., 2017; Shi et al., 2019), whereby the information that is occluded in a single view becomes available. However, as these methods employ a passive paradigm, where data collection is constrained to predefined views and trajectories, they suffer from missing data, redundant information collection, and the need for continuous fine-tuning for novel plants.

Recently, Next-Best-View (NBV) methods in agricultural robotics have gained significant interest (Gibbs et al., 2020, 2018; Zaenker et al., 2021b; Zapotezny-Anderson & Lehnert, 2019). NBV adopts an active perception paradigm to actively reposition and reorient the camera. The selection of the next viewpoint relies on analyzing the observed data so far, considering the objective at hand, with the aim of maximizing the information gain (IG). IG is a metric that quantifies the increase in information achievable by choosing a new viewpoint. This approach enables more efficient and flexible data collection compared to passive perception methods (Atefi et al., 2021). Accurate and efficient estimation of IG is crucial for NBV methods. Traditional NBV methods require maintaining an occupancy grid representing space in voxels and corresponding occupancy probabilities indicating whether a voxel is free or occupied, we call this Voxel-NBV. IG estimation of a candidate viewpoint is performed by casting rays from the viewpoint into the occupancy grid and counting the voxels (both free and occupied) that the rays traverse. This method has been employed in various agricultural applications (Burusa et al., 2024a; Gibbs et al., 2020; Zaenker et al., 2021b). This method is useful as it estimates IG explicitly, making the planning process interpretable, and provides information about both occupied and unoccupied space. However, it can be computationally and memory-intensive due to casting millions of rays and examining the transitions of each ray at a voxel level.

Deep-Learning-based Next-Best-View (DL-NBV) methods aim to address these downsides. These methods use neural networks to directly predict the IG based on the observed data so far, eliminating the need for computationally expensive ray-casting and voxel-wise operations. For example, Mendoza et al. (2020) proposed NBV-Net, which utilizes a 3D convolutional neural network to directly predict the IG of a set of pre-defined candidates using the accumulated occupancy grid as input. Similarly, Zeng et al. (2020) proposed a point-cloud-based neural network called PC-NBV, which predicts the IG values of all the candidates by taking the accumulated point cloud as input. PC-NBV offers higher planning efficiency by eliminating the step of converting

the point cloud to a volumetric map. While both methods display high reconstruction efficiency and robustness to man-made objects with simple geometries, their performance in reconstruction complex plant structure has not been evaluated. Moreover, both methods require separate training steps and complete object models for ground-truth IG estimation, posing challenges for use in dynamic agricultural environments. In this paper, we will develop a DL-NBV method based on the existing PC-NBV method but with the capability of online self-supervised learning and evaluate its performance in an agricultural scenario focusing on plant phenotyping.

Training DL-NBV networks for plant reconstruction poses a significant challenge due to the difficulty of acquiring large amounts of IG-labeled training data. Manual calculation of ground-truth IG is impractical as it is subjective and difficult to accurately measure IG values. However, using a robot, this process can be automated by having the robot explore the environment. Thus, the robot can autonomously collect and annotate IG for data for training even during execution of the task, enabling to improve performance in a life-long and self-supervised manner. This learning approach is termed Self-Supervised Learning (SSL) (Deng et al., 2020). A challenge of SSL is the automatic annotation of IG. Mendoza et al. (2020), Wang et al. (2019a), and Zeng et al. (2020) compute ground-truth IG of a candidate view by comparing to the complete model. However, these methods require prior knowledge of the object shape, making them infeasible for agricultural applications as plant models are not available in advance. To solve this problem, this research proposes an improved IG calculation metric based on the method of Zeng et al. (2020) but enables ground-truth IG calculation solely based on the data collected by the robot, thereby achieving self-supervised learning.

Another problem that prevents the application of current DL-NBV methods in agriculture is their reliance on offline training, i.e., initially collecting large amounts of training data, training the method and then execution in a separate step. This limits the method to continue to improve or to adapt to novel targets and environments (Kahn et al., 2021). Conversely, online learning algorithms integrate both processes into a continuous feedback loop, allowing the network to learn and optimize while executing the task, facilitating adaptation to new targets and environments. However, online learning methods often struggle with low sample-efficiency, as they focus on the most recent data and discard historical data (Zhang & Sutton, 2017). To address the challenges of online learning, Mnih et al. (2013) employed an off-policy training method called ‘experience replay’ in the field of Reinforcement Learning (RL), demonstrating improved sample-efficiency and generalization (refer to step 5 of section 3.2.1.1 for detailed explanation). This inspired our work. Additionally, existing DL-NBV methods typically undergo training with strongly-supervised data, wherein

each training input requires IG-annotation for all candidates as ground-truth, leading to inefficient data collection. To address this issue, we employed a weakly-supervised learning technique to reduce the number of required IG annotations, facilitating efficient training data collection and online learning. Further discussion is provided in section 3.2.1.4.

The objective of this paper is to propose and evaluate a novel SSL-NBV method for 3D robotic plant reconstruction tasks. It utilizes a deep neural network for IG prediction, significantly improving view planning efficiency compared to voxel-based NBV methods. In comparison to state-of-the-art learning-based NBV methods, such as PC-NBV, the proposed method offers several key contributions: (1) it integrates SSL to enable online network training, allowing the model to adapt to novel plants during task execution; (2) it introduces an IG metric to eliminate the need for complete plant models in ground-truth IG calculation; (3) it employs an experience replay technique to enhance sample efficiency; and (4) it adopts weakly-supervised learning to improve training data collection efficiency and reduce annotation demands. The method was tested in both simulation and real-world scenarios, providing a comprehensive evaluation in comparison with various baseline planners. The real-world experiments demonstrated that the proposed method is also capable to learn view planning for the reconstruction of real plants. The study addressed the following questions:

- What reconstruction quality and efficiency can the proposed SSL-NBV achieve in comparison with other NBV (PC-NBV and Voxel-NBV) and non-NBV (Random and Pre-defined) methods?
- How does the proposed SSL-NBV, using online and weakly-supervised learning, compare to PC-NBV, using offline and strongly-supervised learning?
- Can SSL-NBV adapt to a new environment during online self-supervised learning?
- Can the proposed SSL-NBV approach be applied in a noisy real-world scenario?

3.2 Methods and materials

This section describes the method and experimental setup. Section 3.2.1 describes the online self-supervised learning process for plant reconstruction. The experimental setup, including simulation and real-world scenarios, is detailed in section 3.2.2. Finally, section 3.2.3 presents the metrics used for evaluating the performance of the proposed method.

3.2.1 Online self-supervised learning for plant reconstruction

The entire learning process involves continuous iterations. At each iteration, the robot performs next-best-view planning using a deep neural network to select a new camera pose from a set of candidate viewpoints to collect new data for 3D plant reconstruction, which simultaneously generates new training data to update the network. This online learning process allows the network to be updated during task execution and without the need for a separate training phase. This allows the robot to continuously improve and to adapt to a changing environment. The algorithm was first developed and validated in the simulation for rapid development and repeated experiment, then evaluated in the real-world application.

The entire learning process consists of a maximum number of T iterations. The candidates set $C = \{c_1, \ldots, c_M\}$ is defined before the entire learning process, where $c_i \in \mathbb{R}^6$ is a camera pose consisting of position and orientation $\{x_i, y_i, z_i, \alpha_i, \beta_i, \gamma_i\}$, and M is the total number of candidate views. Briefly, the learning process works by iteratively selecting views from C as camera pose for plant reconstruction and collecting data to update the network. View change and data collection are conducted using a robot arm with a depth camera mounted as end effector. From each view, a partial point cloud is collected, and the point clouds collected from all views are fused as a digital plant reconstruction.

Fig. 3.1 provides an overview of an iteration. An iteration t consists of seven steps: (1) Using a DL-NBV network to predict the IG, $\hat{G}_t$ by taking as input the current accumulated point cloud P_t^a and a view-selection state V_t keeping track of previously visited viewpoints; (2) selecting the next view v_{t+1} based on $\hat{G}_t$; (3) moving the camera to v_{t+1} and collecting a new partial point cloud P_{t+1}^c; (4) calculating the actual IG G_t based on P_{t+1}^c and P_t^a; (5) storing supervision pair with P_t^a and V_t as the input, and G_t as the ground-truth; (6) updating the accumulated point cloud P_{t+1}^a and view-selection state V_{t+1}; and (7) training the network weights W_{t+1} of the DL-NBV based on a batch from the training data. The steps are explained in detail in the next subsection.

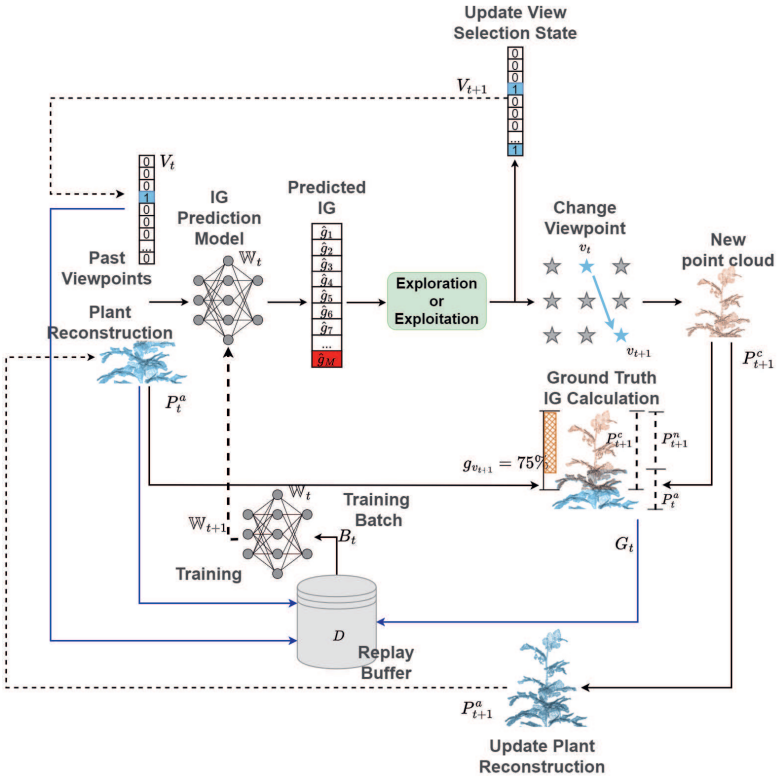


Figure 3.1: Illustration of an iteration step in self-supervised learning. At each iteration step t, next-best-view planning is conducted for plant reconstruction, and generated data, including the accumulated point cloud, view selection states, and ground-truth IG, are collected to optimize the network' s weights. Blue lines represent the collection of training samples, dotted black lines represent data updates, and solid black lines represent data transformation after the operation.

3.2.1.1 The iteration steps in detail

In step 1, the current accumulated point cloud P_t^a and view-selection state V_t are fed into the DL-NBV network to predict $\hat{G}_t = \{\hat{g}_1, \ldots, \hat{g}_M\}$, which contains IG for all candidates, where $\hat{g}_i \in \mathbb{R}$ represents the predicted IG for a viewpoint. $P_t^a = \{p_1, \ldots, p_K\}$ is the accumulated point cloud containing all collected points, where $p_i = \{x_i, y_i, z_i\} \in \mathbb{R}^3$ is a 3D point with xyz-coordinates. $V_t = \{s_1, \ldots, s_M\}$ is a vector containing the selection state for all candidate views, with $s_i \in \{0, 1\}$ equaling 1 if a viewpoint was selected in the past and 0 otherwise. At the beginning of the learning process, $P^a = \emptyset$ and $V = \{0, \ldots, 0\}$. In this research, we used the PC-NBV network (Zeng et al., 2020) as the DL-NBV planner. A detailed explanation of the PC-NBV network structure and IG prediction is provided in section 3.2.1.2 and

section 3.2.1.4.

In step 2, based on $\hat{G}_t$, the estimated next-best view of iteration t is determined as $\hat{v}_{t+1} = \arg\max_{j=1}^{M} \hat{g}_j$, corresponding to the view acquiring the highest predicted IG. However, instead of using $\hat{v}_{t+1}$ directly as the next camera view v_{t+1}, an exploration mechanism is employed to maintain an exploration-exploitation balance during online learning. Exploitation involves selecting $\hat{v}_{t+1}$ as the next camera view, which prevents wasteful exploration, while exploration entails randomly picking a viewpoint v^r from the candidate set to avoid the network getting stuck in a suboptimal view selection policy. We devised an exploration mechanism following a commonly used paradigm as in Wang et al. (2016). The mechanism is governed by the exploration rate $\epsilon_t \in [\epsilon_{min}, \epsilon_{ini}]$. At the beginning of the entire training process, when $t = 1$, $\epsilon_t = \epsilon_{ini}$ was set as the initial exploration rate. Then, ϵ_t is decayed by a decay rate ρ every iteration until reaching a minimum ϵ_{min}, according to $\epsilon_t = \max(\epsilon_{min}, \rho^{t-1} \cdot \epsilon_{ini})$. In this work, we used $\rho = 0.95$ as a constant. The next viewpoint is then determined as follows:

$$v_{t+1} = \begin{cases} \hat{v}_{t+1}, & \text{if } \epsilon_t < x \sim U(0,1) \\ v^r \in_R C, & \text{otherwise.} \end{cases} \tag{3.1}$$

If a randomly generated number x from a uniform distribution with bounds 0 and 1 is greater than ϵ_t, the next viewpoint v_{t+1} is set to $\hat{v}_{t+1}$, otherwise, it is randomly selected from the candidate set C. This exploration mechanism enables the system to perform extensive exploration in the beginning and gradually shifts towards using self-collected information as the network improves from experiences.

In step 3, after the next view is determined, the camera pose is changed to this view, allowing for the collection of a new partial point cloud P^c_{t+1}. A detailed explanation of point cloud collection and processing is provided in section 3.2.1.3.

In step 4, based on P^a_t and P^c_{t+1}, the ground-truth IG $g_{v_{t+1}}$ for viewpoint v_{t+1} is calculated and is then converted into $G_t = \{0, \ldots, g_{v_{t+1}}, \ldots, 0\}$ as the target vector for network training. To achieve continuous online learning in real agricultural scenarios, the robot should be able to calculate $g_{v_{t+1}}$ from its own collected data. To this end, we propose a method enabling ground-truth IG calculation of a viewpoint based solely on P^c_{t+1} collected from this view by robot and the previously collected data P^a_t. Briefly, $g_{v_{t+1}}$ is calculated as the proportion of points in P^c_{t+1} that provide new information that was not captured in P^a_t. As shown in Fig. 3.2, the subset of newly observed points, denoted as P^n_{t+1} (marked orange), within P^c_{t+1} is obtained by removing the intersection (marked green) with P^a_t, calculated as $P^n_{t+1} = P^c_{t+1} - (P^c_{t+1} \cap P^a_t)$. Then, the ground-truth IG for view v_{t+1} is calculated as $g_{v_{t+1}} = |P^n_{t+1}|/|P^c_{t+1}|$, where $|\cdot|$ denotes the size of the set of points.

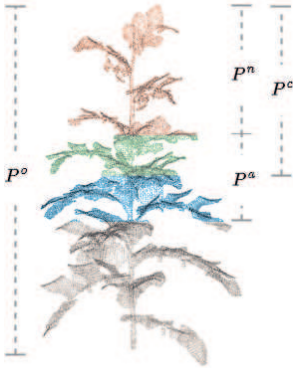


Figure 3.2: Illustration of different types of point clouds. The ground-truth IG of a viewpoint equals the proportion of novel points with respect to the observed data from this view. P^a (blue and green points) refers to the accumulated point cloud. P^c (orange and green points) refers to the newly collected partial point cloud. P^n (orange points) refers to the newly collected points within P^c. Green points refer to the intersection between P^a and P^c. P^o (grey points) refers to the complete point cloud, which is used in the evaluation phase.

However, intersection $P^c_{t+1} \cap P^a_t$ cannot strictly be calculated using the intersection of the sets, because the corresponding points in P^c_{t+1} and P^a_t will not have the exact same coordinates. Instead, we determine $P^c_{t+1} \cap P^a_t$ by calculating the Euclidean distance between each point in P^a_t for each point in P^c_{t+1}. If the smallest distance is below a threshold δ, the point is added to the intersection. So, the intersection is defined as:

$$P^c_{t+1} \cap P^a_t = \{p_k | p_k \in P^c_{t+1} \wedge \min_{p_j \in P^a_t} \|p_k - p_j\|_2 \leq \delta\} \tag{3.2}$$

where $\|\cdot\|_2$ is the Euclidean distance between two points. δ was set at 0.003m in the simulation, while set at 0.01m in the real word. This aligned with the voxel size used for point cloud downsampling (refer to section 3.2.1.3). This method allows for the calculation of ground-truth IG solely based on the data collected by the robot, facilitating an autonomous training data collection when integrated with the flexibility of a robotic system.

The formulated target vector G_t for network training containing only the sparse ground-truth IG for a single view is called weakly-supervised annotation. The detailed explanation of using weakly-supervised data in network training is provided in section 3.2.1.4.

In step 5, the accumulated point cloud P^a_t and the view-selection state V_t are collected as inputs, along with the target vector G_t, to formulate a training sample

$b_t = \{P_t^a, V_t, G_t\}$, which is then stored in a circular buffer $D = \{b_{t-l+1}, \ldots, b_{t-1}, b_t\}$, where $l \in \mathbb{N}$ is the maximum number of samples that can be stored. When the capacity of D is reached, the oldest data is replaced by the newest data, maintaining a fixed-size representation of past experiences. D is not cleared at the end of each plant reconstruction cycle, allowing historical experiences to be reused in the future reconstruction cycles.

In step 6, the accumulated point cloud and the view-selection state are updated. The accumulated point cloud is updated by adding the newly collected point cloud, $P_{t+1}^a = P_t^a \cup P_{t+1}^c = P_{t+1}^c - (P_{t+1}^c \cap P_t^a) + P_t^a$ (the calculation of $P_{t+1}^c \cap P_t^a$ refers to Eq. (3.2)), and the view-selection state is updated to V_{t+1} by setting the $s_{v_{t+1}}$ to 1.

Finally, in step 7, a subset $B_t \subseteq_R D$ is randomly sampled to optimize the weights W_t of the network, where the subscript R indicates a random selection of training samples. $B_t = \{s_1, \ldots, s_N\}$ contains N training samples, where $N \in \mathbb{N}$ is the batch size, and network training starts when the stored number of samples in D exceeds the batch size. Since D is not cleared after each reconstruction cycle, so B_t comprises data from both the current and previous reconstruction cycles. This method, collecting historical data in a buffer and then sampling data from the buffer to train the network, is called experience replay. Experience Replay is a key technique utilized in online learning to improve sample efficiency through the reuse of historical data and improve training stability by breaking temporal correlations between consecutive steps.

3.2.1.2 Deep-learning network structure for information gain prediction

Fig. 3.3 shows the architecture of the network used in this research, which follows the original paper in which PC-NBV was presented (Zeng et al., 2020). At any iteration step t, the network takes P_t^a and V_t as inputs. First, the point cloud P_t^a is processed by a feature extraction network, extracting local features to generate the point-wise feature F0 with 264 features per point. F0 is further processed using max pooling to obtain the global feature G0 of length 264. Subsequently, both G0 and V_t are duplicated to match the dimension of F0 along the vertical axis. This duplication facilitates the concatenation with F0, resulting in the generation of point-wise features F1, containing both global and local features of the point cloud, as well as the view-section state. F1 is then input to a self-attention unit (Zhang et al., 2019a) to further integrate these features, yielding the attention feature F2. Following this, a multi-layer perceptron (MLP1) module with 2 layers of 1024 neurons each following max pooling are applied, producing G1 as the final global feature. G1 represents P_t^a and V_t, which is finally fed into another MLP2 with 4 layers of 1024, 512, 256, and 33

neurons each. MLP2 predicts $\hat{G}_t = \{\hat{g}_1, \ldots, \hat{g}_M\}$, containing the predicted IG for all candidate viewpoints.

Notably, the original PC-NBV is designed for offline learning and tested in a simulation environment. We modified the training approach, IG calculation metric (step 4 of section 3.2.1.1), and loss function (section 3.2.1.4), enabling continuous online learning with weakly-supervised data, showing its capability in real agricultural scenarios.

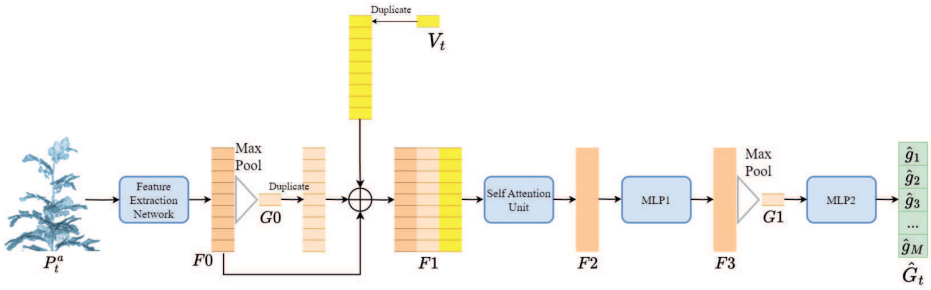


Figure 3.3: The architecture of PC-NBV network. The network initially extracts the global feature of the accumulated point cloud and then combines it with the local feature and view selection state to predict the IG for each candidate viewpoint. The figure is adopted from (Zeng et al., 2020).

3.2.1.3 Point cloud collection and processing

In the simulation environment, a simulated Intel Realsense L515 RGB-D camera was utilized to capture both color and depth information from the viewpoint, which were then combined to form a partial point cloud. Point cloud downsampling was subsequently performed using the VoxelGrid[1] filter with a voxel size of 0.003m, generating P^c. The choice of voxel size aligned with the work of Burusa et al. (2024a), who employed a Voxel-based NBV method for plant reconstruction, enabling direct comparison with their approach.

In the real-world experiment, an Intel Realsense L515 RGB-D camera was used to capture point clouds. Real-world point clouds commonly contain various sources of noise due to ambient lighting fluctuations, sensor-specific artifacts, and idiosyncrasies in point cloud generation algorithms. To address this, a three-step noise reduction process was implemented. Initially, a RangeFilter was employed to eliminate points falling outside a specified range. The cropped point cloud then underwent further refinement through the application of the Statistical Outlier Removal (SOR) filter,

[1] `https://www.open3d.org/docs/0.6.0/python_api/open3d.geometry.voxel_down_sample.html`.

which identified and removed points significantly deviating from their neighbors when compared to the point cloud' s average. Finally, the VoxelGrid filter with a voxel size of 0.01m was applied to further enhance the cleanliness of the point cloud data.

3.2.1.4 Loss calculation using weakly-supervised data

In the original PC-NBV, a classic mean square error (MSE) loss function was utilized to calculate the loss L^s (superscript s represents the loss for strongly-supervised learning) between $\hat{G}_t$ and G_t. The G_t was formulated as $G_t = \{g_1, \ldots, g_M\}$, encompassing the ground-truth IG for all candidate viewpoints. The L^s was computed as follows:

$$L^s = \sum_{i=1}^{M} (g_i - \hat{g}_i)^2 \tag{3.3}$$

This approach requires each $\hat{g}_i$ to be associated with a corresponding g_i for supervision, or in other words, it demands fully ground-truth labels, using strongly-supervised learning. While valid in simulation environments, this method is inefficient for robotic online learning in real-world scenarios, as robotic motion is time-consuming and obtaining G_t necessitates the robot traversing all candidate viewpoints in C.

To address this issue, we propose a weakly-supervised learning method, enabling training based on coarse-grained ground-truth labels, a specific technique called inexact supervision (Zhou, 2018). We defined ground-truth vector $G_t = \{0, \ldots, g_{v_{t+1}}, \ldots, 0\}$, where v_{t+1} is the next viewpoint. The weakly-supervised loss L^w is then computed as follows:

$$L^w = \sum_{i=1}^{M} a_i (g_i - \hat{g}_i)^2 \tag{3.4}$$

where

$$a_i = \begin{cases} 1, & \text{if } i = v_{t+1} \\ 0, & \text{otherwise.} \end{cases} \tag{3.5}$$

This method allows the loss to be calculated only based on the ground truth of the next viewpoint v_{t+1}, significantly enhancing the efficiency of training data collection. However, it may lead to reduced sample efficiency, as only information associated with the next viewpoint is utilized for loss calculation and network optimization. To improve sample efficiency, experience replay is utilized (details refer to step 5 of section 3.2.1.1).

3.2.2 Experimental setup

Three experimental scenarios were designed to evaluate the performance of our SSL-NBV algorithm:

- Simulated Scenario 1 (experiment S1): This scenario evaluates the network' s performance in reconstruction efficiency and quality, IG prediction speed, and training efficiency. We compared our SSL-NBV algorithm with other NBV and non-NBV methods. (Q1).
- Simulated Scenario 2 (experiment S2): Building on S1, this scenario modifies viewpoint settings and target plants to test the method' s generalization and adaptability to novel view settings and target plants. In this experiment, we compared weakly-supervised and strongly-supervised learning (Q2), and analyzed performance improvements after online fine-tuning (Q3). The network weights trained in S1 was used as the starting point for fine-tuning in this experiment to retain knowledge from previous plants. Additionally, the experiment examined the model' s ability to handle catastrophic forgetting—a common issue in online learning where adapting to new data causes the loss of previously learned information. To test this, the fine-tuned network from S2 was reapplied to the S1 scenario to evaluate its ability to retain earlier knowledge.
- Real-World scenario (experiment RW): This scenario involved testing the algorithm in real-world conditions using a robot equipped with an RGB-D camera and real plants (Q4). The neural network trained in S1 was continuously fine-tuned in this scenario.

To improve generalization, multiple plants were used during training. The entire training process was separated into many plant reconstruction cycles. Each cycle, a plant with random pose was positioned as the target, and the robot consecutively selects n views from C to reconstruct the plant, collecting training data and updating the network simultaneously. Once a reconstruction was complete after n iterations, a new plant was positioned, the accumulated point cloud and the view-selection state were re-initialized, and a new cycle began. Through preliminary tests, n=10 was set for simulation experiments, where we systematically increased the value, and 10 viewpoints typically allowed good plant reconstruction for both SSL-NBV and baseline methods. It was also sufficient for comparisons between methods without unnecessary computational overhead. For the real-world experiment, n=15 was set due to the higher complexity of real plants. The same value of n was used during testing. While increasing the number of viewpoints may slightly improve reconstruction ratios for all methods, the overall performance trends and comparative results between methods will be unchanged.

3.2.2.1 Simulation scenarios

The simulation was developed using Gazebo (Koenig & Howard, 2004), with data collection and exchange via the Robot Operating System (ROS) (Quigley et al., 2009). The simulation ran on a ThinkPad P15 laptop with an Intel Xeon W-11855M CPU and an Nvidia GeForce RTX A500 GPU with 16 GB memory, operating on Ubuntu 20.04.

In experiment S1 and S2, as illustrated in Fig. 3.4, the candidate viewpoints set C comprised 33 viewpoints (M=33), arranged in a cylindrical pattern around the origin of the global frame, providing observations of the plant from 11 angles $(a_1, \ldots, a_{11})$ and 3 heights (h_1, h_2, h_3). These viewpoints were all horizontal and oriented to face the z axis of the origin. The camera moved freely between viewpoints, providing a 360° view of the plant. To create additional variation, each time a plant was created, a plant model was randomly selected from the plant model set and positioned randomly with coordinates d_x and d_y along the x and y axes, and a rotation θ.

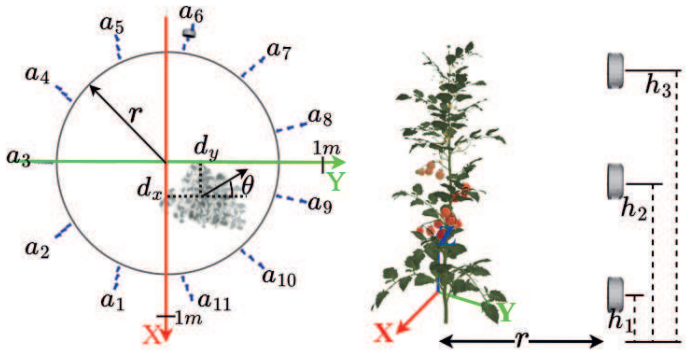


Figure 3.4: Illustration of viewpoint sampling and plant creation. The left image depicts a top view and the right image depicts a side view. Viewpoints (blue arrows) were sampled in a cylindrical distribution to observe the plant from 11 angles $(a_1, \ldots, a_{11})$ and 3 heights (h_1, h_2, h_3). Each plant is created around the origin with random positions d_x and d_y in x and y axes, and a rotation θ.

In experiment S1, the candidate viewpoints featured a radius r of 0.6m, with different heights $h_1 = 0.04$m, $h_2 = 0.25$m, and $h_3 = 0.46$m. For the random translation, the range for d_x and d_y selections were set at $d_x, d_y \in U(-0.1, 0.1)$ with a 0.02m interval, and for the rotation was $\theta \in U(0, 360)$ with a 20° interval. Ten 3D tomato plant models[2] (T1-T10 as shown in Fig. 3.5) were used as targets, exhibiting variations in architecture, size, height, the number of tomato trusses, and leaf nodes. Eight plants

[2] `https://www.cgtrader.com/3d-models/plant/other/xfrogplants-tomato`.

were used for training and two for testing. Due to the limited number of plants, k-fold cross-validation was employed to evaluate the method. Those ten tomato plants were divided into two classes, A (T1-T5) with simpler structures and B (T6-T10) with more complex structures. The ten plants were then randomly split into 5 sets (k=5), each containing one plant from each class: set 1 (T3, T8), set 2 (T4, T6), set 3 (T5, T9), set 4 (T1, T7), and set 5 (T2, T10). This setup resulted in five rounds of validation, where each round used one set for testing and the others for training. Each validation round involved 50 repetitions of plant reconstruction for the testing set, totaling 250 repetitions (5 sets $\times$ 50 repetitions). Each repetition randomly selected a plant from the test set, placed it in a random location and orientation, and had the trained network reconstruct the plant starting from a random viewpoint. We chose 50 repetitions to ensure sufficient variability in the reconstruction process to reflect the method's overall performance.

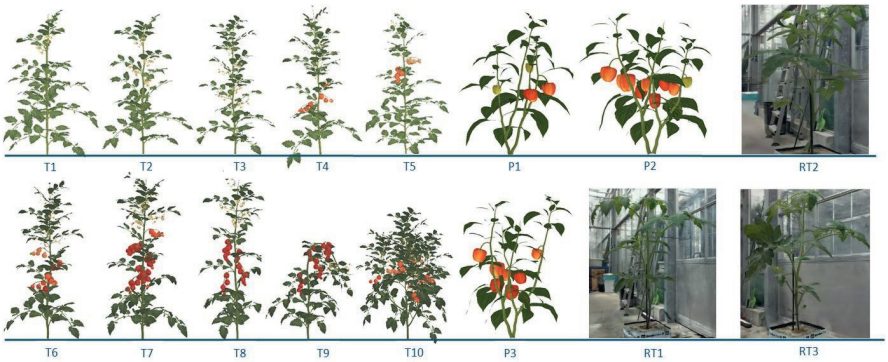


Figure 3.5: Plants used in experiments. T1-T10 are simulated tomato plants used in experiment S1. P1-P3 are simulated pepper plants used in experiment S2. RT1-RT3 are real tomato plants used in experiment RW.

In experiment S2, the viewpoint settings and target plants were modified to create a novel environment, testing the online learning and fine-tuning capabilities of the proposed method in a completely novel scenario. The radius (r) was set at 0.5m, and a larger range of random plant positions $d_x, d_y \in U(-0.3, 0.3)$ was used to introduce more variation in the relative pose between the camera and the plant compared to experiment S1. Three pepper plants[3] (P1-P3 as shown in Fig. 3.5), significantly different in morphology from tomato plants were used. A k-fold cross-validation method with k=3 was employed during the evaluation. In each round of validation, one plant was used for testing while the remaining two were used for fine-tuning. Each validation round involved 50 repetitions, totaling 150 repetitions (3 plants $\times$ 50

[3]`https://www.cgtrader.com/3d-models/plant/other/chili-pepper`.

repetitions).

3.2.2.2 Real-world scenario

The real-world setup involved an ABB IRB 1200 robot equipped with an Intel Realsense L515 RGB-D camera attached to its end effector (refer to Fig. 3.6a), facilitating flexible data collection from various viewpoints. The system was controlled by ROS for robot control and data communication between devices, using the same ThinkPad P15 laptop as simulation.

Due to the motion limitations of the robot, we sampled M=33 viewpoints in a semi-cylindrical distribution, spanning from -60° to 60° relative to the x-axis of the robot frame (see Fig. 3.6b). The radius of the cylindrical sector was 0.45m, with viewpoints at heights of $h_1 = 0.75$m, $h_2 = 1.0$m, and $h_3 = 1.25$m relative to the origin of robot frame. We selected three 40-day-old tomato plants (RT1-RT3, as shown in Fig. 3.5) as targets. These plants were approximately 55cm in height and had 6-7 composite leaves. The data from the real plants was pre-collected, allowing us to repeat experiments and compare the results with baseline methods.

For each plant, the position was fixed at 0.9m, 0m, and 0.65m along the x, y, and z axes of the robot frame, respectively. Four rotations of 0°, 90°, 180°, or 270° were applied, and the point clouds for all 33 views for each rotation were collected. A k-fold cross-validation method was used to test the approach, with each round of validation using two plants as the training set and one plant as the testing set, resulting in a total of three rounds (k=3) of validation. Each round involved 50 repetitions, leading to a total of 150 repetitions (3 plants × 50 repetitions) of plant reconstruction. In each repetition, the test plant, set to one of the four rotations, was selected as the target, and the method reconstructed the plant from a randomly selected initial view.

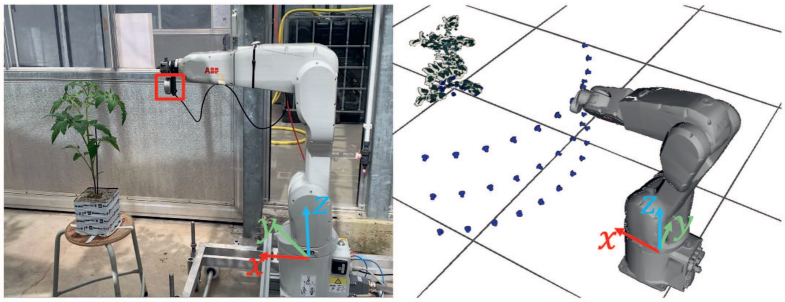


Figure 3.6: Illustration of real-world setup. The real-world environment includes a real robot equipped with an RGB-D camera for flexible plant reconstruction.

3.2.2.3 Implementation details of Self-Supervised Learning in experiments

In all three experiment scenarios, during network training, a batch size N=32 for experience replay was set, and the buffer size was set to l=1000. Before feeding the accumulated point cloud P_t^a to the network, a down sampling procedure was conducted to resize the accumulated point cloud P_t^a to 512 points, independent of the size of the plants, with each point being randomly chosen from P_t^a. The purpose of this down sampling procedure was to ensure a consistent number of points for network IG prediction while maintaining computational efficiency. The original P_t^a was continuously maintained and updated as the final 3D reconstruction. Based on rough estimations, the final 3D reconstruction of P_t^a typically contained 30,000 to 60,000 points for simulated plants and approximately 7,000 to 13,000 points for real-world plants, depending on the size of the target plants. This choice of 512 was based on a preliminary test which indicated that using more points did not improve the network' s accuracy in IG prediction (or increase loss) and reduced prediction speed and increasing computation and memory demands linearly due to the increased complexity of the network architecture.

In experiment S1, the exploration parameters were set to $\epsilon_{ini} = 1.0$ and $\epsilon_{min} = 0.2$, with the maximum number of training iterations set to T=50,000. In experiment S2, ϵ_{ini} was reduced to 0.2 since the network was already pre-trained in experiment S1, and T was set to 12,400 to maintain the same number of iterations per plant in the training set (8 for tomato and 2 for pepper). The replay buffer, which was maintained and updated throughout S1, was cleared before the start of online learning in S2 to enable efficient fine-tuning in this novel environment. In Experiment RW, ϵ_{ini} was set back to 1.0 to allow more exploration due to the substantial differences between simulation and real-world conditions. The maximum iterations were set to T=25,000, which was higher than in experiment S2, considering the increased complexity of real-world plants. The replay buffer was cleared prior to online learning to facilitate efficient fine-tuning in RW.

3.2.2.4 Implementation of baseline planners for experiments

We compared the performance of our SSL-NBV with other methods:

- Traditional Voxel-based NBV (Burusa et al., 2024a): This method converts the point cloud into an occupancy grid and uses ray casting from each candidate viewpoint to calculate the IG for the next-best view. To enhance the speed of view planning, the occupancy grid updates and ray casting processes are executed using GPU acceleration. The parameters used for ray casting, IG

calculation and occupancy calculation follow the original paper.

- PC-NBV (Zeng et al., 2020): This approach has a training setup similar to our SSL-NBV in terms of batch size, training iterations, and epochs. However, PC-NBV relies on strongly-supervised data, requiring the collection of ground-truth IG by traversing all candidate viewpoints, a process that is highly time-consuming and dependent on complete plant models, which are not available in real-world scenarios.

- Random planner: In this method, each next viewpoint is randomly selected from the candidate set C.

- Pre-defined planner: This planner selects 11 viewpoints from the candidate set C and distributed in a zigzag pattern across 3 heights and 11 angles, alternating heights at each angle. The set of views is formulated as $\{a_1h_1, a_2h_2, a_3h_3, a_4h_2, a_5h_1, a_6h_2, a_7h_3, a_8h_2, a_9h_1, a_{10}h_2, a_{11}h_3\}$ (refer to Fig. 3.4). During each cycle of plant reconstruction, 10 viewpoints were randomly selected without repetition from this set and visited in random order to remove the impact of viewpoint order on plant reconstruction efficiency.

3.2.3 Evaluation metrics

Two main metrics were employed to evaluate the proposed method: (a) Reconstruction quality and efficiency; (b) Required number of ground-truth annotations for network training.

3.2.3.1 Reconstruction quality and efficiency

Reconstruction quality was assessed using the reconstruction ratio (R) for the ground-truth point cloud P^o. In the simulation environment, the ground-truth point clouds of plant models were generated by uniformly sampling points on their mesh surfaces. These points were then down sampled using the VoxelGrid filter, resulting in a spatial resolution of 0.003m. This process generated point clouds containing approximately 30,000 to 40,000 points for tomato plants and 50,000 to 60,000 points for pepper plants. In the real-world scenario, where complete 3D models were unavailable, the ground-truth model was approximated by merging point clouds collected from all viewpoints. Subsequently, the same noise filtering process described in section 3.2.1.3 was applied, and the ground-truth point cloud was down sampled to a resolution of 0.01m. This process produced point clouds with approximately 7,000 to 9,000 points for real-world plants. While the ground-truth generated using this method is not perfectly accurate, it provides a reasonable approximation, as all planners were implemented and evaluated under consistent criteria.

The reconstruction ratio at any current reconstruction round was computed as the proportion of points within P^o that were reconstructed in the current accumulated point cloud, expressed as $R = |P^o \cap P^a|/|P^o|$, where the intersection $P^a \cap P^o$ was calculated using Eq. (3.2). In experiment S1 and S2, the threshold δ was set at 0.003m to determine if a point in P^o is reconstructed, while in RW, the threshold was set to 0.01m due to the additional sensor noise. Reconstruction efficiency was assessed by counting the number of views required to achieve specific thresholds $\tau = 0.8$ and 0.9 of R in a plant reconstruction cycle.

The IG prediction speed can significantly impact reconstruction efficiency. A faster IG prediction method enables faster next-best-view selection, leading to faster plant reconstruction. IG prediction speed was measured as the time taken to predict the IG for all 33 candidate views, with and without GPU acceleration. Additionally, learning-based NBV methods require additional time for data collection and network training, which affects overall plant reconstruction efficiency. Assuming A plants need to be reconstructed, the total time required for the proposed method, including network training and plant reconstruction, was calculated and compared to baseline methods.

The performance of our SSL-NBV algorithm was evaluated in comparison to baseline planners using k-fold cross-validation. During each round of validation, all planners were applied to the same set of test plants and performed plant reconstruction with the same number of repetitions (details are provided in sections 3.2.2.1 and 3.2.2.2). The results for each method were obtained by averaging the outcomes from these k-fold cross-validation repetitions.

3.2.3.2 Number of ground-truth annotations

The measurement of each ground-truth IG required the robot to move its end-effector to a candidate view, collect a partial point cloud, and calculate IG based on the accumulated point cloud. As a result, the number of ground-truth annotations necessary for network training significantly impacted the adaptability of online learning methods. A method that requires fewer annotations can reduce robotic motion, accelerate data collection, and facilitate more frequent network updates.

The required number of annotations for SSL-NBV, denoted as $A^{ssl} \in \mathbb{N}$, was calculated equaling the total number of robotic motions, as it used online weakly-supervised learning, each robotic motion allowed the collection of a training sample and its corresponding ground-truth IG. To give a direct insight about the results, the value of A^{ssl} was compared with PC-NBV, which was trained using offline strongly-supervised learning. The required number of annotations for PC-NBV was calculated

as $A^{off} = F \times M$, where F represents the total number of offline training samples and M denotes the number of candidate views.

Since the training data for PC-NBV was pre-collected, A^{off} remained fixed throughout the entire training process. We used A^{off} as the maximum allowable number of annotations for SSL-NBV during online training. As training progressed, SSL-NBV' s performance in plant reconstruction improved, and A^{ssl} increased until reaching A^{off}. Practically, SSL-NBV' s training could be terminated when its plant reconstruction performance converged to a maximum level, as the network' s weights stabilize and optimize. However, determining the exact convergence point can be subjective. To address this, we proposed a statistical approach where training was considered complete when there was no significant difference (p-value>0.05) between the reconstruction ratio of SSL-NBV and PC-NBV. This point was then used to calculate the A^{ssl} for SSL-NBV. The p-value between the two sets of results was calculated using a Two-Sample t-Test, under the assumption that both sets follow a normal distribution.

In experiment S1, 8,000 samples were collected for PC-NBV training, resulting in $A^{off} = 264,000$(8000 samples $\times$ 33 views). In experiment RW, 1,920 samples were collected for fine-tuning PC-NBV, resulting in $A^{off} = 63,360$(1920 samples $\times$ 33 views).

3.3 Results

3.3.1 Simulated scenario 1

The results of experiment S1 addressed Question 1 by comparing the plant reconstruction efficiency and quality of SSL-NBV with baseline methods. Fig. 3.7 shows the average reconstruction ratio of SSL-NBV compared to other methods over cross-validation (visualization examples are provided in Fig. 3.14). All NBV methods outperformed non-NBV methods, confirming that NBV approaches are more effective for plant reconstruction. Our method achieved a final reconstruction ratio of approximately 0.95, corresponding to 30,000 to 60,000 points in the final 3D reconstruction, depending on the size and structural complexity of the target plants. The threshold $\tau = 0.8$ and 0.9 were reached after 5 and 6 viewpoints, respectively. This was faster than the non-NBV methods, where the Pre-defined planner required 5 and 7 views, and the Random planner needed 7 and 10 views, indicating that SSL-NBV can more efficiently reconstruct plants with fewer views. Compared to PC-NBV, which was trained using strongly-supervised learning, SSL-NBV showed a slightly lower reconstruction efficiency, with a 0.02 reduction in the final reconstruction ratio.

However, SSL-NBV uses weakly-supervised learning, requiring only sparse IG labels, can significantly reduce the need for ground-truth IG annotations during training compared to PC-NBV (further analysis is provided subsequently). None of the planners achieved 100% plant reconstruction because the viewpoints were restricted to horizontal views, missing parts of the plant only visible from other angles, such as looking up or down.

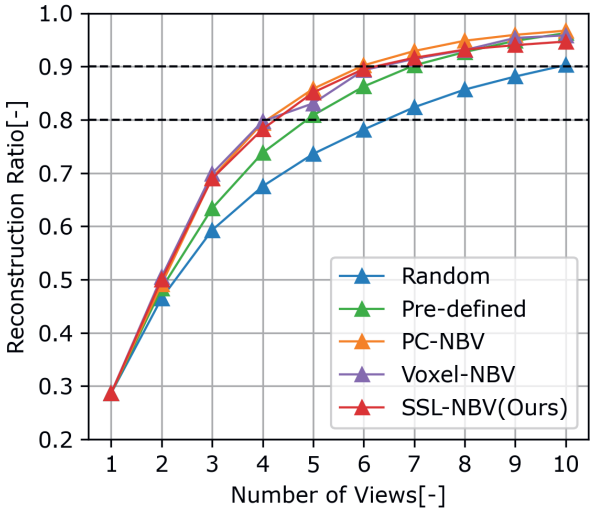


Figure 3.7: Average reconstruction ratio of SSL-NBV in experiment S1 across cross-validation (250 plant reconstruction cycles) compared to baseline methods.

Fig. 3.8 shows the plant reconstruction ratio of SSL-NBV as a function of the number of annotations used during online learning. This is a representative curve with T1-T7 as test plants, and a similar pattern was observed in other validation rounds. The figure compares weakly-supervised (SSL-NBV) and strongly-supervised (PC-NBV) learning in terms of sample efficiency, partially addressing Question 2. To compare to PC-NBV, which used 264,000 annotations during training. In this comparison, the training of SSL-NBV was also extended to 264,000 training iterations but the plot shows that already with fewer annotations, SSL-NBV reaches similar performance. The results indicate that SSL-NBV achieved a similar plant reconstruction ratio to PC-NBV (p-value>0.05) with only 9% of the annotations, indicating a significantly higher sample efficiency. This increased sample efficiency can be attributed to the different training approaches. PC-NBV used offline strongly-supervised learning, updating the network to improve IG prediction accuracy for all views simultaneously. While this approach could enhance generalization and reduce fluctuations in training, it could also introduce redundant or less informative data, reducing sample efficiency.

Conversely, SSL-NBV employed weakly-supervised learning with a balance between exploration and exploitation, which likely results in the selection of more informative views.

While SSL-NBV exhibited a slightly lower average reconstruction ratio than PC-NBV when $A^{ssl} = A^{off}$, the difference was not statistically significant, suggesting that, given the same number of annotations, SSL-NBV can achieve similar plant reconstruction performance with PC-NBV. And consider the significant advantages of SSL-NBV in sample efficiency and online learning adaptability, it is well-suited for automatic plant reconstruction tasks.

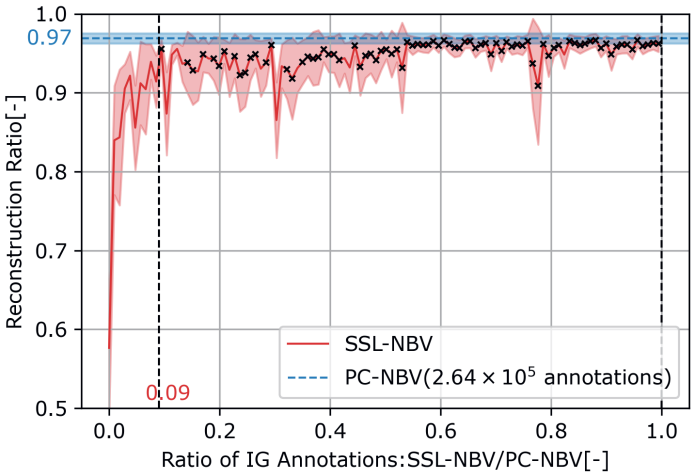


Figure 3.8: The plant-reconstruction ratio as a function of the number of training annotations used during online learning, using T1-T7 as test plants. The x-axis shows the ratio of used IG annotations used by SSL-NBV comparing to the 2.64×10^5 annotations used by PC-NBV. The red line shows the trend for the SSL-NBV method, while the blue line for the PC-NBV method is plotted as a reference. The shaded regions represent the 95% confidence interval over 50 plant reconstruction cycles. Crosses (x) indicate reconstruction ratios not significantly different (p-value>0.05) between SSL-NBV and PC-NBV. The vertical black dashed line indicates the first ratio of IG annotation where the two methods do not show a significant difference in the reconstruction ratio.

Furthermore, we compared the IG prediction speed between our SSL-NBV, PC-NBV, and Voxel-NBV, as displayed in Table 3.1. This speed was measured by the total time required by each method to predict the IG for all candidate views. Since SSL-NBV and PC-NBV used the same network architecture, they achieved a similar prediction speed. When the GPU was utilized, our SSL-NBV required only 0.0038 seconds for IG prediction, representing a substantial 842-times improvement over Voxel-NBV (3.2s).

This highlights the efficiency gain of neural-network-based NBV methods compared to voxel-based approaches that use ray-casting for IG prediction. When only the CPU was available, our method required 0.039 seconds for IG prediction, achieving a 564-times improvement over Voxel-NBV (22s). Importantly, these time measurements considered only IG prediction. For Voxel-NBV, which requires additional time for converting point cloud data to voxels, the difference in speed between the two methods would be further increased.

Additionally, SSL-NBV involved extra steps, such as ground-truth IG calculation, training data storage, and online network training during NBV planning, to fine-tune for novel plants. Under GPU acceleration, one iteration of NBV planning with online learning required 0.099s, making it 32-times faster than Voxel-NBV. On a CPU, the time increased to 3.7s due to computations of ground-truth IG and network training, but it was still 6-times faster than Voxel-NBV. Notably, online network training is only necessary when the method is applied to novel plants. For the plants the method has already been trained, online training could be deactivated.

Table 3.1: Comparison of IG prediction speed between the learning-based NBV methods and the voxel-based method with ray-casting.

GPU/CPU	Planner	NBV Planning (s) (Average±Std)	NBV Planning + Training (s) (Average±Std)
GPU	SSL-NBV	0.0038±0.00029	0.099±0.00024
	PC-NBV	0.0037±0.00042	-
	Voxel-NBV	3.2±0.0038	-
CPU	SSL-NBV	0.039±0.0037	3.7±1.2
	PC-NBV	0.040±0.0038	-
	Voxel-NBV	22±1.8	-

Based on the data presented in Table 3.1 and Fig. 3.8, we estimated the time required to deploy different methods for 3D plant reconstruction. In our simulations, the robot took an average of 0.47 seconds to transition between two viewpoints. For Voxel-NBV, which consumed time solely on plant reconstruction, the total time to reconstruct A plants was calculated as $(3.2\,\mathrm{s} + 0.47\,\mathrm{s}) \times 10 \times A$, where 10 represents the number of views needed per plant. For PC-NBV, additional time was required for data collection and network training. The total time was computed as $264{,}000 \times 0.47\,\mathrm{s} + 1{,}800\,\mathrm{s} + (0.0037\,\mathrm{s} + 0.47\,\mathrm{s}) \times 10 \times A$, where 264,000 corresponds to the minimum number of annotations needed to train PC-NBV for a specific plant type (align with Fig. 3.8), and 1,800 seconds accounts for the network training process. For SSL-NBV, which employed online learning, the annotation requirements

were significantly reduced compared to PC-NBV (see Fig. 3.8). Its total computation time was calculated as $264,000 \times 0.09 \times (0.47\text{s} + 0.099\text{s}) + (0.0038 + 0.47\text{s}) \times 10 \times A$, where 0.09 represents the proportion of annotations required for SSL-NBV compared to PC-NBV to achieve similar performance.

Fig. 3.9 illustrates the time consumption trends for the three methods as the number of plants increases. SSL-NBV consistently required approximately 35.84 hours less than PC-NBV, due to its reduced need for annotations, which reduced robot motions. Voxel-NBV is more efficient than SSL-NBV when reconstructing less than 422 plants, as it does not require training. However, for larger-scale reconstructions, SSL-NBV's efficiency becomes increasingly evident, leading to a substantial reduction in overall time consumption compared to Voxel-NBV.

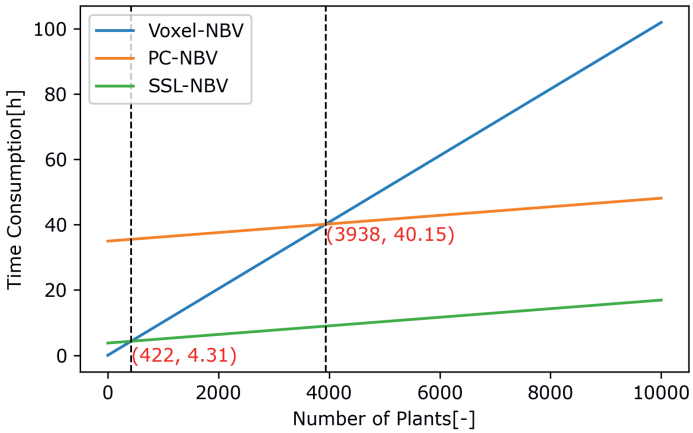


Figure 3.9: Approximate time required for various methods to be deployed in a 3D plant reconstruction scenario. X-axis represents the number of reconstructing plants, and y-axis represents the required time.

3.3.2 Simulated scenario 2

The results of experiment S2 addressed Question 2 by comparing the generalization capability of weakly- supervised learning and strongly-supervised learning, and Question 3 by testing the adaptability of the proposed method in a novel environment. Fig. 3.10 shows the average reconstruction ratio of SSL-NBV compared to other NBV methods over cross-validation (visualization examples are provided in Fig. 3.14). Since the comparison with non-NBV methods was clearly illustrated in experiment S1 and is not relevant to Questions 2 and 3, these methods were excluded from Fig. 3.10. In this experiment, the SSL-NBV network trained in experiment S1 with T=50,000 iterations on tomato plants was continuously fine-tuned for pepper plants. The fine-tuned SSL-NBV demonstrated improved reconstruction efficiency

and quality compared to the non-fine-tuned SSL-NBV network, reaching $\tau = 0.8$ and 0.9 with 1 and 2 fewer views, respectively, showing its adaptability to automatically optimize for new scenarios, addressing Question 3. The fine-tuned network reached $\tau = 0.8$ with 1 fewer viewpoint than PC-NBV. However, both methods achieved comparable reconstruction at the end (containing around 50,000 to 60,000 points in the final 3D reconstruction depending on the size of plants), indicating that PC-NBV has a high level of generalization in this novel environment and can produce good plant reconstruction with sufficient viewpoints.

The non-fine-tuned SSL-NBV showed lower reconstruction efficiency than PC-NBV, indicating that weakly-supervised learning offers lower generalization compared to strongly-supervised learning. However, SSL-NBV was trained with significantly fewer annotations (50,000 for SSL-NBV compared to 264,000 for PC-NBV) than PC-NBV, still achieved a similar final reconstruction ratio (addressing Question 2). After automatic online fine-tuning, SSL-NBV outperformed PC-NBV. Although the improvement was small, we attribute this to the fact that experiment S2 still shared some similarities with experiment S1, enabling PC-NBV to generalize to this environment. The Voxel-NBV yielded the poorest results, possibly because the grid space was enlarged to allow more variation in plant position in this scenario (refer to section 3.2.2.1). This introduced more empty voxels, impacting the performance of Voxel-NBV, as this method aimed to reconstruct the entire grid space rather than focusing solely on the plant.

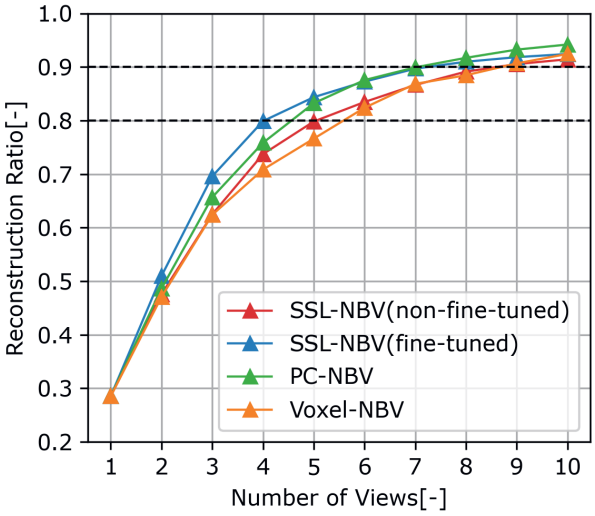


Figure 3.10: Average reconstruction ratio of SSL-NBV in experiment S2 across cross-validation (150 plant reconstruction cycles) compared to baseline methods.

Furthermore, we evaluated the model's ability to handle catastrophic forgetting. This evaluation consisted of three phases: (1) initial training on tomato plants (consistent with experiment S1), (2) fine-tuning on pepper plants (aligned with experiment S2), and (3) re-training on tomato plants, using the network weights obtained after phase (2). As shown in Fig. 3.11, a slight decrease in the reconstruction ratio was observed at the beginning of phase (3), indicating that prolonged adaptation to novel plants poses challenges for SSL-NBV in fully retaining knowledge of previously learned plants. However, the method still achieved a reconstruction ratio above 90%, demonstrating its feasibility for tomato plant reconstruction tasks. But for scenarios requiring higher reconstruction accuracy, network re-training may be necessary to ensure optimal results for tomato plants.

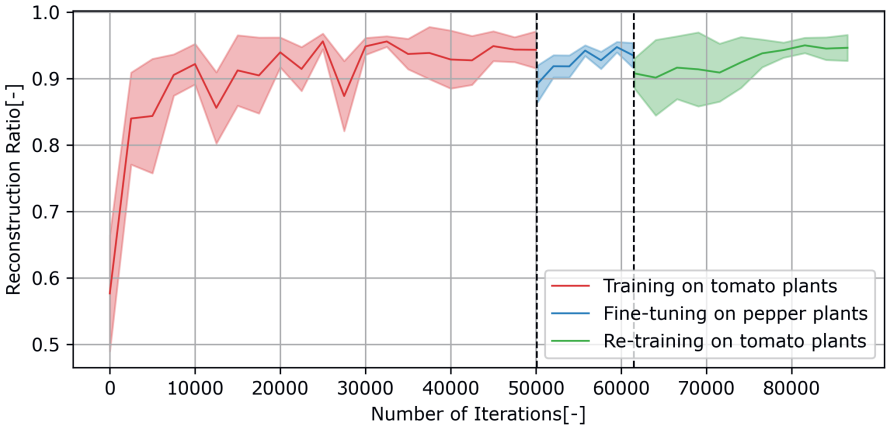


Figure 3.11: Reconstruction ratio of SSL-NBV during online training across three phases: initial training on tomato plants (red), fine-tuning on pepper plants (blue), and re-training on tomato plants (green).

3.3.3 Real world

The results of experiment RW addressed Question 4 by evaluating the SSL-NBV method in a real-world scenario. Fig. 3.12 illustrates the average reconstruction ratio of SSL-NBV compared to other methods over cross-validation. SSL-NBV showed effectiveness in this real-world scenario, outperforming non-NBV planners in both reconstruction quality and efficiency. Specifically, SSL-NBV achieved a $\tau = 0.8$ within 7 viewpoints, which is 2 and 3 views faster than the Random and Pre-defined planners, respectively. SSL-NBV reached a $\tau = 0.9$ after 12 views, a threshold that the non-NBV methods failed to reach. SSL-NBV reconstructed approximately 92.5% of the plant (corresponding to 7,000 to 13,000 points in 3D reconstruction depending on the size of plants), representing a 3.5% and 8.5% improvement over the Random and

Pre-defined planners, respectively.

Notably, in this experiment, we approximated the ground-truth plant model by merging point clouds from all viewpoints, which made the ground-truth IG calculation and offline fine-tuning of PC-NBV possible. However, in real-world applications, ground-truth plant models are not available for PC-NBV fine-tuning. In contrast, our SSL-NBV does not require ground-truth plant models and automates the entire training/fine-tuning process to collect its own training data and to continuously optimize the network to adapt to novel plants and environments in a self-supervised manner.

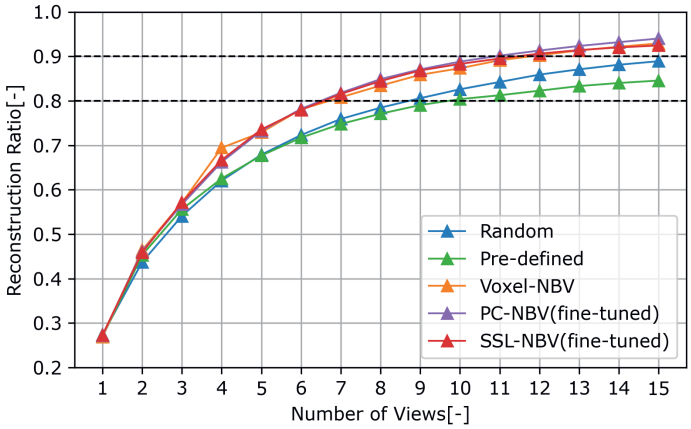


Figure 3.12: Average reconstruction ratio of SSL-NBV in experiment RW evaluated using cross-validation (150 plant reconstruction cycles) compared to the baseline methods.

Fig. 3.13 shows the trend of SSL-NBV' s reconstruction ratio as an increased number of annotations were used during online learning, with RT3 as the test plant. During fine-tuning, 1,920 samples were recollected to retrain PC-NBV, resulting in $A^{off} = 63,360$ IG annotations. To align with PC-NBV (with 63,360 annotations), the training of SSL-NBV was extended to 63,360 training iteration. The figure shows SSL-NBV reached a similar reconstruction ratio (p-value>0.05) with PC-NBV using only 21% of the annotations, demonstrating a 79% reduction in the need for ground-truth annotations.

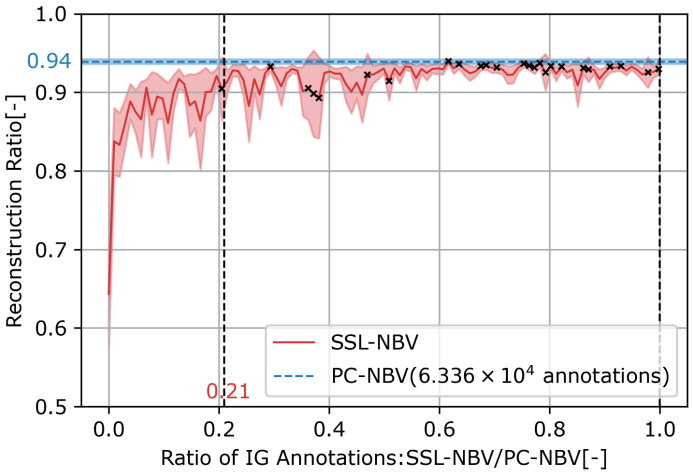


Figure 3.13: The plant-reconstruction ratio as a function of the number of training annotations used during online learning, using RT3 as the test plant. The shaded regions represent the 95% confidence interval over 50 plant reconstruction cycles. The x-axis shows the ratio of IG annotations used by SSL-NBV over PC-NBV. Crosses (x) indicate reconstruction ratios not significantly different (p-value>0.05) between SSL-NBV and PC-NBV.

Fig. 3.14 illustrates examples of 3D reconstruction results for various plant types, including simulated tomato plants (experiment S1), simulated pepper plants (experiment S2), and real-world tomato plants (experiment RW). The point clouds of simulated and real-world plants exhibit differences in point density, due to the use of varying voxel sizes for point cloud downsampling (see section 3.2.1.3 for details). All plants achieved a reconstruction ratio exceeding 90% by the end of the reconstruction cycle, with 10 views allowed for simulated plants and 15 views for real-world plants, demonstrating the effectiveness of SSL-NBV in both simulated and real-world environments. Notably, the ground-truth point cloud for real-world plants contains noise, particularly at the edges, which may reduce the reconstruction ratio as the noise is challenging to reconstruct accurately.

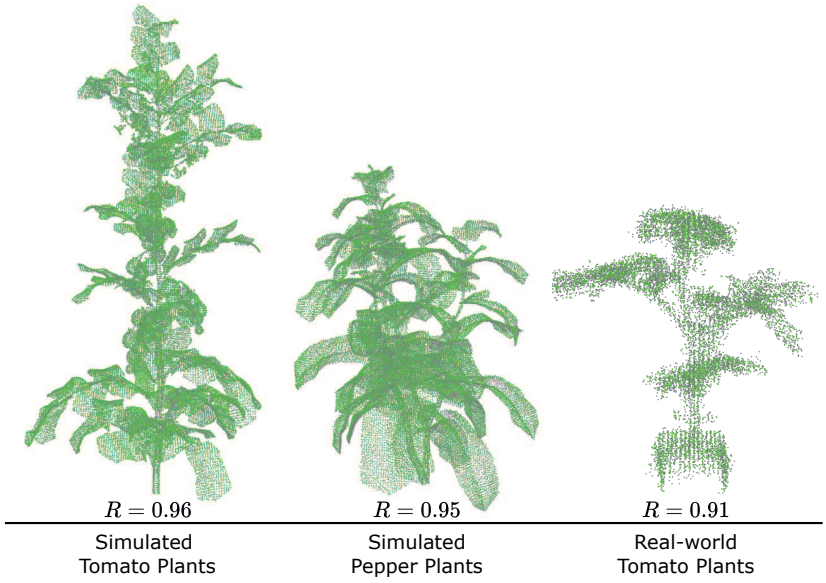


Figure 3.14: 3D reconstruction results across experiments S1 (10 views), S2 (10 views), and RW (15 views). Ground-truth point clouds are shown in grey, and reconstructed point clouds are displayed in green. The reconstruction ratio for each reconstruction is indicated at the bottom. Simulated point clouds were downsampled using a 0.003m voxel size, while real-world point clouds used a 0.01m voxel size, resulting in differences in point density.

3.4 Discussion

3.4.1 Comparison with relevant studies

The results demonstrated that learning-based NBV methods achieved plant reconstruction performance comparable to the classic Voxel-based NBV method, while significantly improving IG prediction speed. This aligns with Han et al. (2022), who developed a Double Branch NBV Network (DB-NBV) for 3D object reconstruction and evaluated it on synthetic datasets. Our findings also support Zapotezny-Anderson & Lehnert (2019), who showed that using a deep neural network for direct IG prediction improved viewpoint planning efficiency. Although SSL-NBV required online training during NBV planning, slightly reducing overall efficiency, it remained more efficient than voxel-based methods. Importantly, online learning is only necessary for novel plants; for previously fine-tuned targets, this step is unnecessary, preserving high efficiency.

Compared to PC-NBV, which relied on strongly-supervised learning with fully labelled IG data, our method required only sparse IG labels and achieved similar plant reconstruction efficiency while significantly reducing the number of ground-truth IG annotations. Our findings are consistent with Stutz & Geiger (2020), who demonstrated that using 3-10% of annotations can yield comparable results to fully annotated training data in 3D shape completion tasks. Similarly, Cheng et al. (2023) showed that using weakly-supervised labels can achieve comparable performance to fully labelled data in image segmentation tasks. To our knowledge, the application of weakly-supervised learning to NBV systems has not been explored, with existing approaches like those of Han et al. (2022), Zeng et al. (2020) and Mendoza et al. (2020) relying on extensive IG labels.

Notably, our method has the capability to automatically fine-tune for novel plants and environments without human intervention. When integrated with robotics, it enabled life-long and continuous online learning, which is crucial for robotic agricultural operations where plants and environments are constantly changing. This capability is absent in existing learning-based NBV methods, which typically rely on pre-collected offline data for training.

We conducted a quantitative analysis of the method' s performance in real-world plant reconstruction, an important evaluation lacking in similar studies. While Han et al. (2022) applied their method to real-world plant reconstruction, their evaluation was qualitative and lacked quantitative comparison. Furthermore, their approach required complete object models for ground-truth IG calculation, limiting its ability to be fine-tuned for real-world plants where such models are unavailable. This issue was also present in the works of Zeng et al. (2020) and Mendoza et al. (2020). Instead, our method addressed this by improving the IG metric and eliminating the need for complete plant models in IG calculation, enabling continuous fine-tuning for real-world plants.

In summary, this study presented SSL-NBV, a method capable of automatically adapting to novel plants and environments, allowing for life-long and continuous improvement of the robot during task execution without human intervention. Through comprehensive evaluation in both simulation and real-world experiments, we demonstrated its efficiency in IG prediction and its ability to significantly reduce the need for IG annotations. This work offers valuable insights for future applications of NBV methods in agriculture, particularly in reducing the reliance on extensive annotations and enhancing scalability in dynamic environments.

3.4.2 Limitations and future improvements

3.4.2.1 Flexible viewpoint sampling for different-sized plants

Our method used a global viewpoint sampling approach, where a set of candidate viewpoints were pre-sampled, and the NBV planner iteratively selected from this set to reconstruct the plant. This approach is efficient, as it enables a global search for the next best viewpoint. However, it restricted the selection to fixed viewpoints, potentially missing optimal views and reducing reconstruction quality. Additionally, the use of a fixed set of candidate views can limit the method' s scalability for different plant sizes. Zeng et al. (2022) and Lehnert et al. (2019) addressed this challenge with relative viewpoint planning, where the next camera pose was determined based on incremental movements from the current view. These methods are not restricted to fixed viewpoints and are adaptable to various plant sizes. However, local view planning methods often face issues with local maxima. In future work, we aim to integrate SSL-NBV with local NBV methods, facilitating global planning for large-scale search (e.g., entire plant or row of plants) while enabling detailed reconstruction of plant parts not visible from fixed viewpoints.

3.4.2.2 Trajectory optimization of camera view

Our approach performed NBV planning on a viewpoint-wise manner, selecting the next best view solely based on the maximization of IG. However, this can lead to low view trajectory efficiency, requiring extensive camera movement to achieve a similar level of plant reconstruction. Ideally, NBV planning should consider view trajectory optimization, i.e., minimizing camera movement while maintaining high reconstruction efficiency. To address this, we propose integrating a trajectory optimization mechanism into the existing NBV planning process using a DRL algorithm. DRL is widely used in path planning as it considers long-term impacts, learning a policy about task execution that maximizes cumulative rewards. In the NBV problem, incorporating both reconstruction efficiency and camera movement into the reward function could enable efficient plant reconstruction while reducing camera movement. Another potential solution is to add constraints to the current view selection process, allowing only nearby views to be selected after weighing camera movement against the predicted IG.

3.4.3 Impact of experimental conditions

3.4.3.1 Size and distribution of set of candidate viewpoints

The size of the candidate set can affect the network's training efficiency and the quality of plant reconstruction. Increasing the candidate set size can improve re-

construction by capturing previously unseen plant parts but also reduces training efficiency and increases the need for training data. In this study, we used a set of 33 viewpoints, which balanced plant reconstruction and training efficiency. However, for larger plants or row of plants reconstruction tasks, expanding the candidate set may be necessary to fully capture all plant parts. A larger candidate set can amplify the advantages of NBV methods over non-NBV methods in plant reconstruction, as it becomes more difficult for non-NBV methods to select high-IG viewpoints, while NBV approaches, given sufficient training (for DL-NBV) or time (for Voxel-NBV), can effectively identify the next-best viewpoint, improving plant reconstruction results.

In this study, we employed cylindrical and semi-cylindrical distributions in simulation and real-world environments, respectively, similar to Golbach et al. (2016). This configuration facilitated plant reconstruction because the plants were centrally located so that each viewpoint had a high probability of capturing a plant part. However, in terms of evaluation, this setup may narrow the gap between NBV and non-NBV methods by making it easier for non-NBV methods to select viewpoints with high IG.

3.4.3.2 Limited real-world plants

Three real plants were used in this study to evaluate the method's performance in real-world scenarios. While testing on only three plants may not fully evaluate the method's overall performance due to the limited coverage of plant variations, we believe it still demonstrated the feasibility of the method for real plant reconstruction. Firstly, data for each plant were collected from four different poses, adding variations to the training and testing sets. Secondly, comprehensive evaluations were conducted in simulations with a wide range of plant variations, which helped validate and support the real-world results. Thirdly, all results were obtained through k-fold cross-validation, which enhanced the robustness of the evaluation. We anticipate that testing on more real plants, the methods would yield similar patterns to those observed in our study. However, to thoroughly evaluate the method's performance in the real world, we recommend conducting further experiments that include a broader range of plants and plant variations.

Additionally, although we performed a three-step noise removal process on the point cloud, there was still significant noise, especially at the edges of the ground-truth point cloud (Fig. 3.14 provides visual examples). This noise reduced the reconstruction ratio, as noise present in the ground-truth point cloud may be absent in the reconstruction. To minimize the effect of noise, we set the threshold δ to 0.01m (refer to Eq. (3.2)) for detailed explanation). Although this relatively large threshold may

overestimate reconstruction performance by easily counting a point in the ground truth as reconstructed, it ensured fairness from a comparative perspective and did not affect the comparison between methods, as the same threshold was applied to all methods. We believe the impact of camera noise can be decreased in the future as technology advances and RGB-D cameras become more accurate and cheaper. Furthermore, since complete 3D models of real-world plants were unavailable, the ground-truth model was approximated by merging point clouds from all viewpoints. Although this method introduced imperfections into the ground-truth, it maintained fairness in evaluation, as all methods were assessed using the same criteria. The effect of these imperfections on the reconstruction ratio is difficult to quantify, as it is affected by multiple factors like noise, and evaluation metric. However, we believe that even with a perfect ground-truth, the comparative performance between methods would remain consistent.

3.4.3.3 SSL-NBV training iterations

During SSL-NBV online training, we set T=50,000, 12,400, and 25,000 iterations for experiments S1, S2, and RW, respectively. These settings allowed the method to achieve comparable performance in plant reconstruction to PC-NBV while significantly reducing training time. However, as observed in Fig. 3.8 and Fig. 3.13, with additional training iterations, the plant reconstruction performance of SSL-NBV could be further improved, bringing it closer to that of PC-NBV. Thus, we believe that the results in Fig. 3.7, Fig. 3.10, and Fig. 3.12 of SSL-NBV could also show improved performance if extended training iterations were applied, potentially reaching similar results with PC-NBV.

Furthermore, the number of self-supervised learning iterations can be reduced or even eliminated when SSL-NBV is applied to plants or environments it has already learned. In our experiments, self-supervised learning was consistently enabled to allow the method to adapt to novel plants and environments. While this ensures flexibility, it also introduces additional time for network optimization. This process can introduce additional time for network optimization. Results from experiment S2 demonstrated SSL-NBV' s generalization capability for a different plant type, suggesting that self-supervised learning can be deactivated for familiar scenarios. Disabling this functionality for known plants and environments would significantly improve NBV planning efficiency and prevent time consumption on plants that do not require fine-tuning, enhancing the method' s practicality.

3.5 Conclusion

This paper proposed an SSL-NBV algorithm to enhance learning-based NBV methods through self-supervised learning, enabling efficient and automatic training of neural networks to predict the IG of next viewpoints, which allows to optimize 3D plant reconstruction. Using a robot to actively change viewpoints, the method facilitated life-long and continuous online learning without requiring human annotations. To ensure efficient online learning, an improved IG metric, experience replay, and weakly-supervised learning techniques were incorporated. A comprehensive evaluation with k-fold cross-validation was conducted to address the research questions.

The results of experiment S1 showed the reconstruction quality and efficiency for different multi-view reconstruction methods (answering Question 1). The proposed SSL-NBV method outperformed non-NBV methods, achieving the level of 90% plant reconstruction with fewer views, respectively 1 view and 4 views for the Pre-defined planner and the Random planner. While SSL-NBV method achieved similar reconstruction quality to other NBV methods, it excelled in computational speed. Specifically, SSL-NBV achieved IG prediction in 0.0038s, making it 842 times faster than the classic Voxel-based NBV. When online learning was enabled, SSL-NBV required additional steps for network training, taking 0.099s per training iteration, which was still 32 times faster than Voxel-based NBV. Compared to PC-NBV, which used offline strongly-supervised learning, the proposed method achieved comparable plant reconstruction performance with only 9% of the IG annotations (answering Question 2). Moreover, this was achieved purely in a self-supervised way, without the need of human annotation or a ground-truth plant model. In terms of deployment efficiency, Voxel-based NBV was more efficient for small-scale plant reconstruction tasks (less than 422 plants), as it eliminates the need for data collation and training steps. However, SSL-NBV's efficiency advantages became increasingly pronounced as the number of plants grew, making it more suitable for larger-scale applications. Question 3 was addressed in experiment S2. After online self-supervised learning, the method achieved 80% and 90% plant reconstruction 1 and 2 views faster than the non-fine-tuned network, respectively, demonstrating the method' s adaptability to new environment though online self-supervised learning (answer Question 3). Also, a clear improvement in plant reconstruction during online learning (as shown in Fig. 3.8, Fig. 3.11, and Fig. 3.13) further confirmed the method' s adaptability. Finally, the results of experiment RW addressed showed that the proposed method could also successfully be applied in the real world (Question 4), achieved 92.5% plant reconstruction after online fine-tuning on real plants. It outperformed the Random and Pre-defined planners by around 3.5% and 8.5%, respectively, while reducing the need for IG annotations by 79% compared to PC-NBV.

In conclusion, the proposed SSL-NBV method was capable of efficient 3D plant reconstruction in simulated and real-world environments and could adapt to novel environments through online self-supervised learning without the need of any human intervention. Although this work was demonstrated using single plants, we believe that the proposed method can be adapted in future research for robotic plant reconstruction in more complex agricultural environments. Adaptations should overcome the current limitation of having a small number of fixed candidate views, and instead use more flexible view sampling and planning of view trajectories.

Chapter 4

Robotic Self-Supervised Local View Planning for Efficient and Scalable 3D Plant Reconstruction Across Varying Plant Sizes

This chapter has been published as:

Ci, J., van Henten, E. J., Wang, X., & Kootstra, G. (2025). Robotic self-supervised local view planning for efficient and scalable 3D plant reconstruction across varying plant sizes. *Computers and Electronics in Agriculture*, 238, 110839. https://doi.org/10.1016/j.compag.2025.110839

Abstract

Accurate perception in complex agricultural environments is challenging due to significant plant occlusion, primarily from leaves, which hinder data collection and increase uncertainty in robotic operations. Deep-learning-based Next-Best-View (DL-NBV) methods address this by using neural networks to predict information gain (IG) for potential camera views and actively repositioning the camera to maximize data collection with minimal views. However, training DL-NBV models requires extensive IG-labeled data. A self-supervised learning-based NBV method, SSL-Global-NBV, enables robots to collect training data autonomously and improve themselves for global NBV planning.

Despite its advantages, SSL-Global-NBV has two key limitations: (1) it requires a fixed number of views, limiting scalability across different plant sizes, and (2) it selects views globally, resulting in inefficient view transitions across the entire view space, reducing trajectory efficiency. To overcome these limitations, this paper introduces SSL-Local-NBV, which incorporates local view planning for scalable and efficient view selection. To prevent redundant visits to the same views, a View Trajectory Network (VTN) was proposed to memorize the view trajectory information of visited views.

Comprehensive evaluations in simulation and real-world plant reconstruction demonstrated that SSL-Local-NBV reduced trajectory distance by 56% – 70% per reconstruction cycle, achieving 267% – 300% higher trajectory efficiency than global NBV methods. Compared to SSL-Global-NBV, SSL-Local-NBV improved plant reconstruction efficiency by 5.2% across varying plant sizes, demonstrating greater scalability. For real plants, SSL-Local-NBV achieved over 80% reconstruction, confirming its feasibility in practical applications. Notably, SSL-Local-NBV fully automated training through self-supervised learning, enabling continuous and lifelong robotic learning.

4.1 Introduction

In agriculture, robotics has gained significant interest as a solution for automating labor-intensive tasks, reducing reliance on human labor, and addressing labor shortages. A typical workflow for robotic operations in agriculture involves first using a robotic perception system to gather sufficient information about the target, such as position, orientation, or 3D reconstructions, which then guides subsequent tasks (Arad et al., 2020; Ci et al., 2024; Sa et al., 2017; Wang et al., 2025). However, achieving accurate robotic perception in complex agricultural environments is challenging due to substantial plant occlusion, primarily caused by leaves (Kootstra et al., 2021). This occlusion hinders data collection for robotics, increasing operational uncertainty and reducing success rates (Burusa et al., 2024c). The main objective of this work was to develop an effective method to address plant occlusion using active perception. The method was quantitatively tested on 3D plant reconstruction tasks, with the resulting reconstruction expected to support automatic plant phenotyping in later stages. With minimal modifications, this method could be extended to other agricultural tasks by robotics, such as de-leafing, monitoring, and harvesting, which encounter similar occlusion challenges.

Plant phenotyping is a research field that involves measuring various plant traits, such as leaf angle, internode length, and leaf area, and linking these phenotypic data to genotypes, thus bridging the gap between plant phenotype, genotype, and the environment. Automated plant phenotyping methods combine machine vision with robotics to provide automated, accurate, and high-throughput measurements of phenomics data. The first step in automatic plant phenotyping is generating an accurate 3D digital reconstruction of the plant, which is then used for trait measurement. However, due to significant plant occlusion, accurate 3D plant reconstruction is challenging. Some methods address this issue by using multiple views, enabling the collection of data that is unavailable from a single view (Boogaard et al., 2020; Golbach et al., 2016; Hemming et al., 2014; Shi et al., 2019). For example, Hemming et al. achieved an approximately 30% increase in sweet pepper fruit detectability when the number of viewpoints increased from one to five. However, most of these methods use a fixed set of views which does not guarantee to effectively deal with occlusion.

Next-Best-View (NBV) methods, which follow an active perception paradigm, iteratively reposition the camera to poses expected to capture the most novel information, aiming to achieve efficient 3D plant reconstruction using the fewest views (Burusa et al., 2024a; Gibbs et al., 2018; Wu et al., 2019a). This approach offers high flexibility and robustness to plant variations. In NBV methods, the selection of the next viewpoint is typically based on the expected information gain (IG) for a number of

candidate views, with the most informative viewpoint chosen as the next camera view, called the next-best viewpoint. IG estimation is an essential step in NBV methods. Traditional methods, such as that by Isler et al. (2016), use ray-casting for IG estimation by casting millions of rays into an occupancy grid and counting the uncertainty per voxel (such a method is hereafter referred to as voxel-based NBV), which is time-consuming and computationally intensive. Alternatively, deep-learning-based NBV (DL-NBV) methods employ deep neural networks to directly predict IG based on the current 3D reconstruction of the scene, demonstrating significantly higher efficiency (Han et al., 2022; Mendoza et al., 2020; Zapotezny-Anderson & Lehnert, 2019; Zeng et al., 2020). For example, Mendoza et al. reported that DL-NBV achieved approximately 3,000 times faster IG prediction than voxel-based NBV. Similarly, Hen et al. observed a 120-fold increase in IG prediction speed using DL-NBV. And However, DL-based NBV methods face several challenges when applied in real-world scenarios: (1) they require large amounts of IG-annotated training data, and (2) calculating ground-truth IG necessitates digital 3D models of the plants. Consequently, these methods are primarily feasible in simulations, but the neural networks cannot be effectively trained or fine-tuned for real plants.

Ci et al. (2025) addressed this issue by enhancing the state-of-the-art DL-NBV method with self-supervised learning (SSL), called SSL-NBV. In robotics, SSL enables robots to autonomously collect and annotate their own training data during task execution, which is then used to train models and improve robotic operational performance (Zeng et al., 2017a). This process allows model training in an online and self-supervised manner, an important capability for enhancing the method's adaptability to dynamic environments. However, Ci et al.'s method faced two challenges. First, they employed a global viewpoint planning approach (hereafter referred to as SSL-Global-NBV), in which a set of candidate views around the plants was pre-sampled. The method then selected the view with the highest IG from this set for NBV planning. This often led to viewpoint transitions that traversed the entire view space, resulting in longer camera movements, a problem referred to as low trajectory efficiency. Additionally, this approach of view selection was restricted to a fixed number of candidate views, limiting the method's scalability across different plant sizes, which typically require varying numbers of views for optimal reconstruction.

These challenges can be addressed by shifting to local viewpoint planning, where the next view involves selecting an action with defined direction and distance to move the camera relative to the current view, ensuring smoother trajectories compared to global viewpoint planning methods. Burusa et al. (2024b) for instance, proposed a method that iteratively optimized camera pose using gradient ascent to maximize IG, achieving comparable reconstruction performance while reducing trajectory distance

by 80%. However, as this method relied on ray-casting and occupancy grid, it was slow in IG estimation. Furthermore, local viewpoint planning selects actions based on collected information and visited views, enabling next view selection not limited to a predefined set of views, making it scalable to various plant sizes.

The objective of this work was to address the limitations of SSL-Global-NBV by introducing SSL-Local-NBV, a local viewpoint planning method aimed at improving trajectory efficiency and scalability in 3D plant reconstruction. This approach builds on SSL-Global-NBV (Ci et al., 2025), retaining its ability for online self-supervised learning, allowing the model to update itself autonomously during reconstruction. The contributions of this work are as follows: (i) a local NBV planning framework, SSL-Local-NBV, that improves trajectory efficiency and scales effectively across different plant sizes and view configurations; (ii) a View Trajectory Network (VTN) that encodes a latent representation of visited views to avoid redundant observations and integrates with PC-NBV to achieve online self-supervised learning for IG prediction; (iii) a viewpoint selection strategy that chooses the next view from local changes in various directions and distances, supporting effective exploration and fine plant structure capture; and (iv) a comprehensive evaluation of SSL-Local-NBV in both simulation and real-world environments against baseline methods.

This study addressed the following research questions: (1) What is the gain in trajectory efficiency of SSL-Local-NBV in 3D plant reconstruction compared to SSL-Global-NBV using global views planning and other baseline methods (Voxel-based NBV and non-NBV methods)? (2) How well does the SSL-Local-NBV deal with plants of various sizes in 3D reconstruction tasks? (3) How well does SSL-Local-NBV perform on 3D plant reconstruction tasks in real-world scenarios?

4.2 Materials and Methods

This section outlines the methods, materials, and experimental setup used to evaluate the proposed SSL-Local-NBV. Section 4.2.1 details the self-supervised learning process for 3D plant reconstruction. Section 4.2.2 describes the experimental setup used to address the research questions.

4.2.1 Self-supervised learning for local view planning

4.2.1.1 Overview of Self-supervised learning during plant reconstruction

Fig. 4.1 provides an overview of the learning-based local NBV planning process for 3D plant reconstruction. Starting from an initial camera pose, a deep neural network

predicts information gain (IG) for all possible actions (the blue arrows) to select the best action for adjusting the camera pose. The camera then moves to the new pose, captures additional data to update the 3D reconstruction. In this paper, we call this an "NBV iteration". This process repeats for a number of iterations until the plant is reconstructed, or the maximum allowed number of iterations is reached.

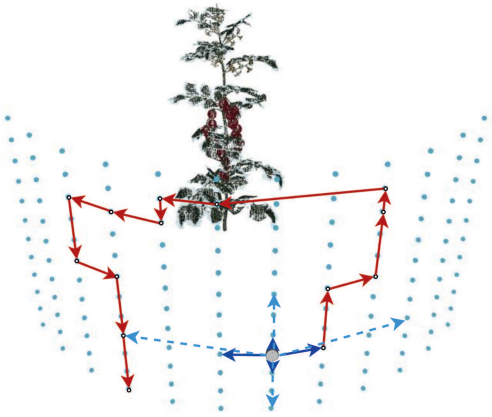


Figure 4.1: An illustration of the local NBV planning process for 3D plant reconstruction. Red arrows indicate the trajectory of camera movement, while blue arrows represent the possible actions. Solid blue arrows represent small movements, and dashed blue arrows represent large movements in the same directions. In this example, the plant is manually modeled, with details provided in section 4.2.2.1

.

In this work, the NBV planning process was improved by incorporating a self-supervised learning framework, enabling robotic continuous online learning for 3D plant reconstruction. As illustrated in Fig. 4.2, the proposed SSL-Local-NBV consists of three components: (a) 3D plant reconstruction, where a deep neural network predicts IG values for all possible actions and selects the best action to adjust the camera position for collecting new data; (b) ground-truth IG calculation, where the system automatically gathers ground-truth IG based on the data captured by the robot itself, eliminating the need for human annotations; and (c) data collection and network training, where training data is continuously collected and used to update the neural network in an online, self-supervised manner.

A single NBV iteration of the online self-supervised learning process during 3D reconstruction involves seven steps. (1) The neural network predicts IG for all possible actions based on the current reconstruction and information about past views (as introduced in the figure). (2) The system selects the best action and adjusts the camera pose. (3) A partial point cloud is collected from the new camera pose. (4) This point cloud is compared to the current reconstruction to compute the amount of new

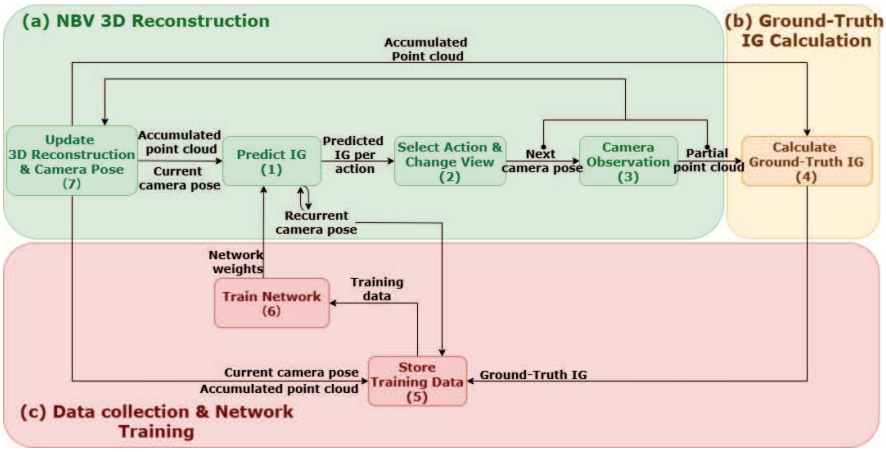


Figure 4.2: An illustration of a single iteration of the online self-supervised learning process during 3D reconstruction. Module (a) shows the NBV planning process, Module (b) illustrates the automatic calculation of ground-truth IG, and Module (c) illustrates the self-supervised training of the neural network. The numbered blocks (1) to (7) correspond to the seven steps of an NBV iteration.

information in the new observation (the ground-truth IG). (5) The resulting data is stored for network training. (6) The neural network is trained for one iteration using a batch of data randomly sampled from the collected training dataset. At every NBV iteration, the network continues to train. (7) The new observation and viewpoint are used to update the 3D reconstruction and current camera pose. Detailed explanations of each step are presented in section 4.2.1.2, and section 4.2.1.3 illustrates the network architecture and loss function in detail.

The NBV iterations continue until the reconstruction is complete or the maximum number of iterations $i^{\max} \in \mathbb{N}$ is reached. The collection of NBV iterations resulting in a reconstruction is called a "reconstruction cycle". The self-supervised learning process consists of continuous reconstruction cycles on multiple plant instances in the environment, where in each cycle, a scene with a plant is set as the reconstruction target. The neural network weights are continuously fine-tuned to enhance IG prediction accuracy and improve reconstruction efficiency in the target environment.

Alg. 1 formalizes the notations, integrating NBV iterations into reconstruction cycles, offering a comprehensive representation of the entire online learning process for the target environment. For an environment with a set of plants M, the plant reconstruction scene is initialized with a view space $V \subset \mathbb{R}^3$, which specifies potential camera poses. The robot then performs NBV iterations for plant reconstruction, while simultaneously collecting training data for self-supervised learning. The total number of

executed iterations over the lifespan of the system is counted using $t \in \mathbb{N}$, and the total number of executed reconstruction cycles is counted using $c \in \mathbb{N}$.

Algorithm 1 Self-supervised learning process

Input: $\mathcal{M}, \mathcal{A}, n, i^{max}, \epsilon^{min}, \epsilon_0, q$

$t \leftarrow 0$
$\mathcal{W}_t \leftarrow InitialNet()$ ▷ Initialize the network
$\mathcal{D}_t \leftarrow \emptyset$ ▷ Initialize the replay buffer
$c \leftarrow 0$
while true **do**
  $\mathcal{V} \leftarrow InitializeScene(\mathcal{M})$ ▷ Initialize scene of plant reconstruction
  $v_0 \leftarrow RandomView(\mathcal{V})$
  $v_0^r \leftarrow (0, \ldots, 0)$
  $P_0^{acc} \leftarrow ObserveEnvironment(v_0)$ ▷ Initial camera observation
  for $i \leftarrow 0 : i^{max}$ **do**
    $\hat{G}_i, v_{i+1}^r \leftarrow PredictIG(P_i^{acc}, v_i, v_i^r)$ ▷ 1. Predict IG
    $a_i \leftarrow SelectAction(\hat{G}_i, \mathcal{V}, c, \epsilon^{min}, \epsilon_0)$ ▷ 2. Select action
    $v_{i+1} \leftarrow Execute(a_i)$ ▷ Change view
    $P_{i+1}^{part} \leftarrow ObserveEnvironment(v_{i+1})$ ▷ 3. Camera observation
    $G_i \leftarrow \mathcal{F}(P_i^{acc}, P_{i+1}^{part})$ ▷ 4. Calculate ground-truth IG
    $\mathcal{D}_{t+1} \leftarrow UpdateTrainingSet(\mathcal{D}_t, (P_i^{acc}, v_i, v_i^r, G_i))$ ▷ 5. Store data
    if $Size(\mathcal{D}_{t+1}) \geq n \ \wedge \ (t \mid q)$ **then**
      $\mathcal{B}_{t+1} \leftarrow SampleBatch(\mathcal{D}_{t+1}, n)$ ▷ Sample training data
      $\mathcal{W}_{t+1} \leftarrow TrainNet(\mathcal{W}_t, \mathcal{B}_{t+1})$ ▷ 6. Train network
    end if
    $P_{i+1}^{acc} \leftarrow Merge(P_i^{acc}, P_{i+1}^{part})$ ▷ 7. Update 3D Reconstruction
    $t \leftarrow t + 1$
  end for
  $c \leftarrow c + 1$
end while

4.2.1.2 The NBV iteration

Step (1): For an iteration i, the deep neural network predicts IG $\hat{G}_i = \{\hat{g}_1, \ldots, \hat{g}_A\}$ for all A possible actions in the action space $\mathcal{A} = \{a_1, a_2, \ldots, a_A\}$, where each $a_j \in \mathbb{N}$ corresponds to a specific camera movement. The predictions are based on three inputs: a) the accumulated point cloud $P_i^{acc} = \{p_1, \ldots, p_K\}$, consisting of K points, where $p_k = \{x, y, z\}$ contains the x-y-z coordinates; b) the current camera pose $v_i = \{x, y, z\}$; and c) v_i^r, a view-trajectory representation of all past views. Details on

the network structure, IG prediction, and view-trajectory representation are given in section 4.2.1.3.

Step (2): Based on $\hat{G}_i$ and V, the next-best action $\hat{a}_i$ is determined by selecting the action with the highest IG, as described by the following equation:

$$\hat{a}_i = \arg\max_{a \in \mathcal{A}} \hat{g}_a \tag{4.1}$$

After execution, the camera is moved to the corresponding new view $v_{i+1} \in V$. To enable efficient and stable online learning and prevent the network from getting stuck in a sub-optimal policy (e.g., consistently selecting the same action), instead of always executing the best action, an exploration-exploitation mechanism is employed. An exploration rate, ϵ_c, determines the probability of exploration by the robot. If a randomly selected value from a uniform distribution $U(0,1)$ is larger than ϵ_c, $\hat{a}_i$ is executed; otherwise, an explorational action a^r is randomly selected from $\mathcal{A}^p \subseteq \mathcal{A}$ to explore the world. $\mathcal{A}^p$ is a subset of $\mathcal{A}$ with only the actions that can generate a valid next view v_{i+1} that is within V to prevent moving outside the workspace. The process of determining the best action a_i is thus defined as follows:

$$a_i = \begin{cases} \hat{a}_i, & \text{if } x \sim U(0,1) > \epsilon_c \\ a^r \in_R \mathcal{A}^p, & \text{otherwise} \end{cases} \tag{4.2}$$

To maintain an exploration-exploitation balance, an exploration decay mechanism is used. Starting from an initial value ϵ_0, the exploration rate decays exponentially every reconstruction cycle by multiplying it with a decay rate $\gamma \in [0,1]$, calculated as $\epsilon_c = \epsilon_0 \gamma^c$, until reaching the minimum value $\epsilon^{\min}$. This method allows extensive environment exploration during the early stages of training, avoiding the network getting stuck in suboptimal planning strategies. Over time, it transitions to more efficient exploitation based on the learned knowledge, reducing wasteful exploration and improving learning efficiency.

Step (3): From the new view v_{i+1}, a new partial point cloud P_{i+1}^{part} is collected using an RGB-D camera.

Step (4): The ground-truth IG G_i for action a_i is calculated using the accumulated point cloud P_i^{acc} and the newly collected P_{i+1}^{part}, as proposed in (Ci et al., 2025). G_i equals the ratio of novel points within P_{i+1}^{part} compared to P_i^{acc}, which is calculated as follows:

$$G_i = \frac{|P_{i+1}^{part} - (P_{i+1}^{part} \cap P_i^{acc})|}{|P_{i+1}^{part}|} \tag{4.3}$$

where $|\cdot|$ represents the number of points of a point cloud, and $P_{i+1}^{part} \cap P_i^{acc}$ represents the intersection between the two point clouds. This approach allows the ground-truth

IG to be estimated solely based on the data collected by the robot, facilitating robotic online self-supervised learning. As the coordinates of the points in P_{i+1}^{part} and P_i^{acc} are not exactly matched, their intersection cannot be directly calculated. To address this, the Euclidean distance between each point in P_i^{acc} for each point in P_{i+1}^{part} was calculated. If the closest point in P_i^{acc} to a point in P_{i+1}^{part} had a Euclidean distance smaller than a threshold δ, that point in P_{i+1}^{part} was considered an intersected point; otherwise, it was classified as a novel point.

Step (5): A new training sample is constructed based on the accumulated point cloud P_i^{acc}, the current viewpoint v_i, and the view-trajectory representation v_i^r as input feature, along with the ground-truth G_i as output target. This training sample is then stored in a circular buffer D, which stores the most recent $l \in \mathbb{N}$ training samples. l is referred to as the buffer size. In step (6), training data is sampled from D to train the network. This method, which stores historical training data for later network training, is known as experience replay. Experience replay is crucial in online learning to improve data efficiency and the model's generalization (Mnih et al., 2013).

Step (6): A number $n \in \mathbb{N}$ of training samples is randomly selected from D to construct a training batch B for a training iteration of the current network weights W_t, leading to W_{t+1}. The training process is triggered when the number of training samples in D exceeds n and the number of executed NBV iterations t is a multiplier of q, where q is a hyperparameter that controls network update frequency. More details of network architecture and loss function are provided in section 4.2.1.3.

Step (7): The 3D reconstruction of P_i^{acc} is updated with the newly collected point cloud P_{i+1}^{part}. Specifically, P_{i+1}^{acc} is calculated as follows:

$$P_{i+1}^{acc} = P_i^{acc} \cup (P_{i+1}^{part} - (P_i^{acc} \cap P_{i+1}^{part})) \tag{4.4}$$

Where $P_{i+1}^{part} - (P_i^{acc} \cap P_{i+1}^{part})$ represents the set of novel points in P_{i+1}^{part}, and $P_i^{acc} \cap P_{i+1}^{part}$ denotes the intersection between the two point clouds, which is calculated similarly to step (4). This prevents adding already reconstructed points multiple times.

4.2.1.3 Network architecture and loss function

Fig. 4.3 illustrates the architecture of the proposed network for IG prediction, which integrates two components: (1) the View Trajectory Network (VTN), introduced in this work, and (2) the IG Prediction Network, adapted from PC-NBV (Zeng et al., 2020) and modified to support local view planning in combination with the VTN.

In learning-based local NBV planning, maintaining a memory of past views is important for avoiding redundant visits to previously observed regions. To address this, the VTN was proposed. VTN incorporates a Gated Recurrent Unit (GRU) (Chung et al., 2014), a type of recurrent neural network that can store and utilize historical information to improve predictions for current and future inputs. At iteration i, VTN takes the current camera view v_i and a view-trajectory representation v_i^r as inputs. v_i^r is a latent representation predicted by the VTN encodes information from all past views and is updated with each new input. First, VTN encodes the current view v_i using position encoding (Mildenhall et al., 2021), transforming the camera pose from low-frequency (3 features) to a high-frequency one, v_i^e, with 63 features. Given a view $v_i = [x, y, z]$ with 3D coordinates, the v_i^e is computed as:

$$v_i^e = [x, y, z, \sin(2^0\pi x), \cos(2^0\pi x), \ldots, \sin(2^{L-1}\pi z), \cos(2^{L-1}\pi z)] \tag{4.5}$$

Where L is set to 10, representing the number of frequency bands. This high-frequency representation helps neural networks better distinguish between close views by capturing fine-grained spatial variations (Rahaman et al., 2019). The encoded view v_i^e is then passed through a two-layer Multi-Layer Perceptron (MLP0) with 128 and 64 neurons, respectively, producing output L0. This output is concatenated with v_i^e to generate feature L1, which, together with the previous view-trajectory representation v_i^r, is processed by the GRU to produce v_{i+1}^r, the updated representation incorporating the new view.

The original PC-NBV framework uses a 'View State' vector to track visited candidate views, where each view is marked as 1 if visited, and 0 otherwise. It helps prevent redundant view selection in global view planning. However, in local view planning, the number of views varies with plant size, making the construction of a fixed-size view state vector impractical. In this work, the latent view-trajectory representation v_{i+1}^r is used to replace the view state vector, effectively encoding historical view information in a scalable way. The IG prediction network takes the accumulated point cloud P_i^{acc} and the view-trajectory representation v_{i+1}^r as inputs. First, P_i^{acc} is processed by a feature extraction unit (Wang et al., 2019b), producing point-wise feature F0 with 264 features per point, which is then max-pooled to generate a global feature G0 (length 264), representing the whole point cloud. F0, G0 and a duplicated v_{i+1}^r are concatenated to form point-wise feature F1, representing local and global point cloud representation and past visited views. Then, F1 is fed into a self-attention unit (Zhang et al., 2019a) to fuse the information from different sources and produce F2. A two-layer MLP (MLP1), each with 1024 neurons, processes F2 and generates F3, which is then max-pooled to produce G1, the final feature capturing the combined context from the point cloud and past views. Finally, G1 is processed by a four-layer MLP (MLP2) with 1024, 512, 256, and A neurons, where A is the size of the action

space size, to predict the information gain for each possible action.

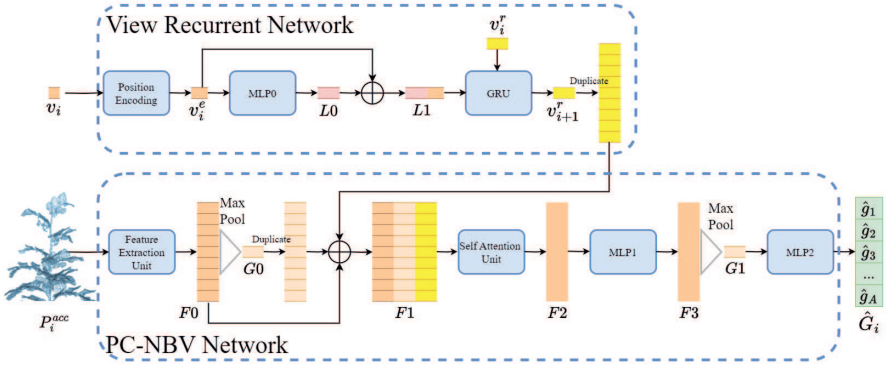


Figure 4.3: The structure of the proposed neural network. The view trajectory network predicts a latent representation of the visited views to avoid view revisiting. The IG prediction network extracts features from the accumulated point cloud, which are then fused with the view trajectory information to predict IG for possible actions.

Theoretically, building a supervision for learning-based next best view selection requires ground-truth IG for all candidate views at each step, as implemented in previous works (Caldeira & Moreno, 2024; Mendoza et al., 2020; Zeng et al., 2020). This is known as strongly-supervised learning. However, collecting strongly-supervised IG annotations is extremely time-consuming in real-world scenarios, as it requires the robot to traverse all candidate views for each training sample.

To address this issue, Ci et al. (2025) introduced weakly-supervised learning into the training paradigm. This approach only requires ground-truth IG for the selected view, significantly reducing the required number of ground-truth IG annotations and the needed time. In this work, this method was adapted for local NBV planning. A weakly-supervised loss function, denoted $\mathcal{L}^w$, was proposed based on the Mean Square Error (MSE) function. $\mathcal{L}^w$ only needs ground-truth IG for the action a_j that was selected to move the camera. $\mathcal{L}^w$ is calculated as follows:

$$\mathcal{L}^w = \sum_{j=1}^{A} \mu_j (g_j - \hat{g}_j)^2 \tag{4.6}$$

$$\mu_j = \begin{cases} 1, & \text{if } a_j \text{ is selected} \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

In this setup, only the loss associated with the selected action for camera movement was used for backpropagation. This reduces the need for dense annotations and makes the approach more feasible for online learning for changing environment.

4.2.2 Experimental setup

A comprehensive evaluation was conducted in both simulation and real-world environments to address the research questions. Q1 and Q2 were answered through simulation experiments, which tested (1) the trajectory efficiency compared to baseline planners and (2) the scalability of the proposed SSL-Local-NBV on plants of varying sizes, respectively. Q3 was addressed in the real-world experiment to assess the method' s feasibility in practical scenarios. The primary motivation for using simulation was to enable efficient, repeatable, and controlled evaluation with access to perfect plant ground truth. These conditions are difficult to achieve in real-world settings. While the simulated environments cannot exactly replicate real-world conditions, they still provide valuable insights into the performance of SSL-Local-NBV through comparisons with baseline methods, as all methods are evaluated under the same criteria. Combined with the real-world results, these findings allow concrete conclusions to be drawn.

This section is outlined as follows: section 4.2.2.1 describes the simulation experiment setup, followed by the real-world experiment setup in section 4.2.2.2. The action space used for all experiments is described in section 4.2.2.3. Section 4.2.2.4 provides implementation details for self-supervised learning. The baseline methods are described in section 4.2.2.5, and the evaluation metrics are detailed in section 4.2.2.6.

4.2.2.1 Simulation scenarios

The simulation environments were built using Gazebo (Koenig & Howard, 2004), with data exchange facilitated by the Robot Operating System (ROS) (Quigley et al., 2009). The setup operated on a ThinkPad P15 laptop with an Intel Xeon W-11855M processor, an Nvidia GeForce RTX A5000 GPU (16 GB VRAM), and Ubuntu 20.04.

Two simulation experiments, hereafter called experiment S1 and experiment S2, were designed, to address Q1 and Q2, respectively. Experiment S1 used ten manually modeled digital tomato plants (T1–T10, Fig. 4.4a) as reconstruction targets[1]. These plants exhibited morphological variations, such as the number of trusses and leaves. Additionally, varying levels of occlusion were included, making them sufficient for evaluating the method' s performance in plant reconstruction under occlusion. Eight plants (T1, T2, T4, T5, T6, T7, T9, T10) were used for network training, while two (T3, T8) for testing, ensuring the test set included plants both with and without fruits. The plants showed similar sizes (height: 1.43m ± 0.19m), ensuring they could be covered by the same set of views.

[1] `https://www.cgtrader.com/3d-models/plant/other/xfrogplants-tomato`.

Experiment S2 utilized ten modified plants (T11–T20, Fig. 4.4b) to evaluate method's scalability across different plant sizes. These plants derived from T1–T10 by randomly adding or removing elements (e.g., leaves, branches, fruits), resulting in varied plant heights (1.48m ± 0.51m). Smaller plants were created by removing parts from the bottom or middle of the plants and scaling down the entire structure along the x, y, and z axes. Conversely, larger plants were produced by adding parts and scaling up the structure. The plants were grouped by size: small (T11, T12, T14; 0.92m–1.07m) were approximately 25% smaller, medium plants (T13, T16, T19, T20; 1.09m-1.49m) retained their original structure and size, and large plants (T15, T17, T18; 2.12m-2.34m) were approximately 35% larger than the original. Seven plants (T11, T13, T14, T18, T18, T19, T20) were used for network training, and three (T12, T15, T16) were used for testing, with one plant from each size group.

During each reconstruction cycle, the plant model was randomly positioned around the origin of the world frame with random translations d^x, d^y (sampled from U(-0.1,0.1) at 0.05m interval), along the x and y axes, respectively, and a random rotation θ around the vertical axis (sampled from U(0,360) at 60° intervals), as shown in Fig. 4.4c.

During testing, online learning and exploration were disabled. For each repetition, a plant from the test set was randomly selected and positioned with random offsets and rotation. The method then performed view planning to reconstruct the plant from a random initial view. In both experiments S1 and S2, 25 repetitions were conducted, and the final results were averaged across all repetitions.

The view space V defined the sampling region for camera views. As shown in Fig. 4.4c, three semi-cylindrical view spaces ($V_{\text{small}}, V_{\text{med}}, V_{\text{large}}$) were designed for the simulated plants, corresponding to small, medium, and large plants, respectively. V_{med} was used in experiment S1, while all three were used in experiment S2. The height of $V_{\text{small}}, V_{\text{med}}$, and V_{large} were 35 cm, 55 cm, and 85 cm, respectively, with the lowest views positioned 5cm above the ground. These view spaces were divided into grids of 8×12, 12×12, and 18×12 views, respectively. All views oriented horizontally toward the z-axis, spanning from -55° to 55° relative to the x-axis with a radius of 60cm (see Fig. 4.4c for a top-down view). This design enabled accurate 3D reconstruction from a wide range of viewpoints while ensuring safe robotic motion during camera movement. During viewpoint transitions, collision avoidance was not explicitly implemented. If the robot operates at a closer range (e.g., with a reduced radius), a collision-avoidance mechanism may be required to ensure safe operation (Ye et al., 2021).

An Intel RealSense L515 RGB-D camera was used to capture RGB and depth images (960×540) from each selected view. These images were then combined to generate 3D

point clouds. A VoxelGrid[2] filter with a voxel size of 0.003m was applied to downsample each point cloud, reducing memory usage while preserving plant structure.

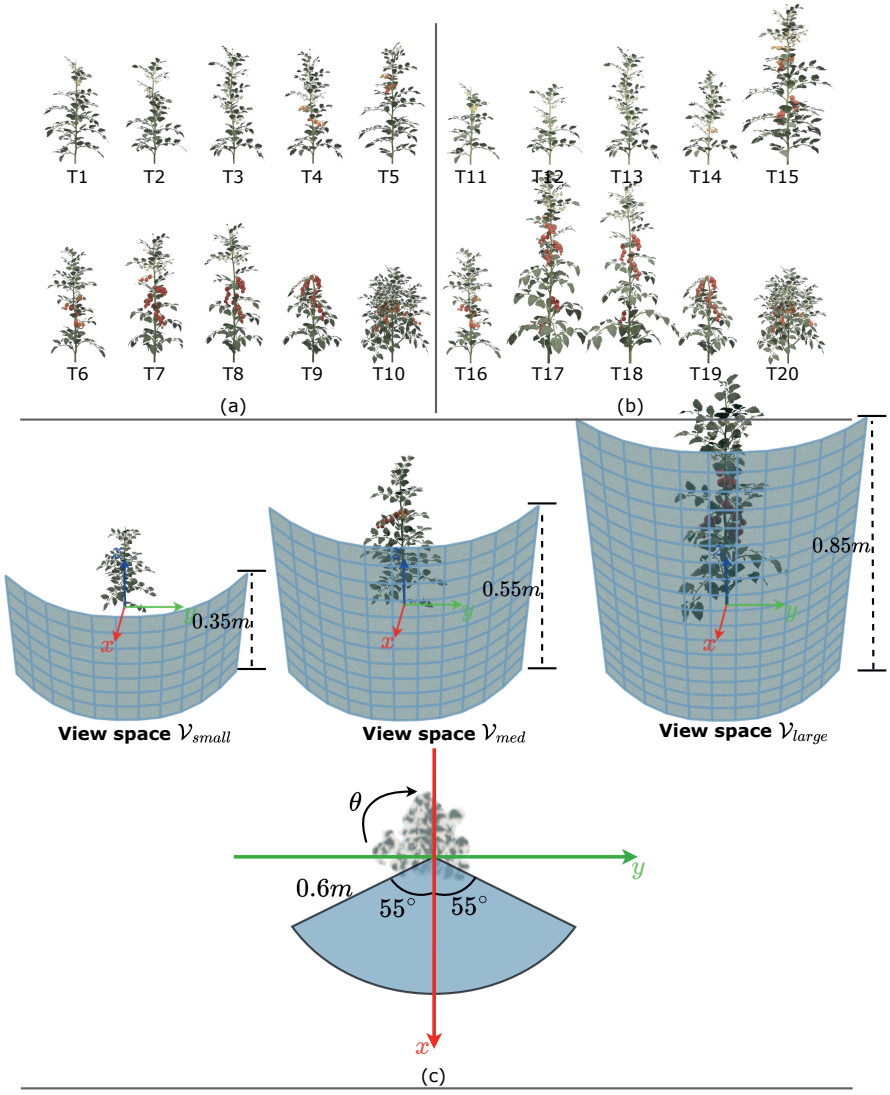


Figure 4.4: Setup for simulation experiments. (a) Ten simulated plants with similar sizes were used in Experiment S1. (b) Ten simulated plants with varying sizes were used in Experiment S2 to evaluate scalability across different plant sizes. (c) View spaces designed for small, medium, and large plants, labeled as $V_{\text{small}}, V_{\text{med}}$, and V_{large}, respectively.

To provide a more comprehensive evaluation, two additional tests were conducted based on Experiment S1: a cross-crop validation and a noise robustness test, aimed

[2] https://www.open3d.org/docs/0.6.0/python_api/open3d.geometry.voxel_down_sample.html.

at evaluating the generalization and robustness of SSL-Local-NBV.

Cross-crop validation was performed using pepper plants[3]. As shown in Fig. 4.5, compared to tomato plants, pepper plants exhibit significant morphological variations and more severe occlusion due to larger leaves. Pepper 2 and Pepper 3 were used for training, while Pepper 1 was reserved for testing. The same translation and rotation procedures used in experiment S1 were applied when creating the pepper models.

Noise robustness testing was performed by manually introducing Gaussian noise into the input point clouds of the neural network. Two noise levels were applied to the original clean data, as illustrated in Fig. 4.6. For the light noise condition, 20 percent of the points were randomly selected and added with Gaussian noise (mean zero, standard deviation 0.01) in the x, y, and z coordinates. For the heavy noise condition, 80 percent of the points were added using Gaussian noise with mean zero and standard deviation 0.05. Importantly, noise was only added to the point clouds used as input to the network, directly testing the method' s robustness during NBV planning. The reconstruction used the original clean data to ensure a fair evaluation of reconstruction quality.

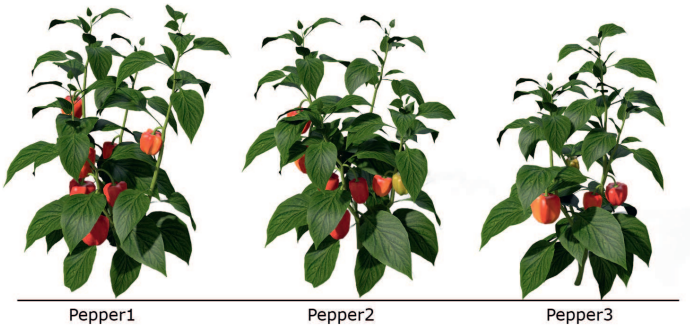


Figure 4.5: Pepper plants used for cross-crop validation.

4.2.2.2 Real-world scenario

A real-world experiment, hereafter called experiment RW, was conducted to address Q3. A dataset of 10 real tomato plants (T21-T30) (Fig. 4.7a) from Burusa et al. (2024c) was used to evaluate the method in practical scenarios. The tomato plant, cultivar 'Tasty Tom', were grown under standard commercial greenhouse conditions at Wageningen University, The Netherlands. Each plant had a single main stem, with side shoots removed as part of routine cultivation. Due to robotic reachability, only

[3] `https://www.cgtrader.com/3d-models/plant/other/chili-pepper`.

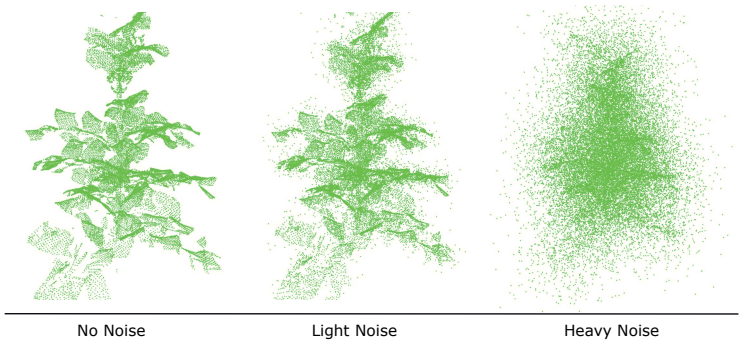


Figure 4.6: Point clouds with different levels of noise introduced in Experiment S1.

the regions between 0m and 1.3m above the growing substrate were scanned, which is also the primary region of interest for most robotic operations, such as harvesting. The plants exhibited significant variations in morphology, with fruit trusses ranging from 0 to 3 and leaf nodes ranging between 3 and 8 for different plants. The plants exhibited substantial occlusions, primarily caused by leaves.

Data collection was performed using an ABB IRB 1200 robotic arm (6 DoF) with an Intel RealSense L515 RGB-D camera mounted on the end-effector. At each viewpoint, RGB and depth images (960×540) were captured and converted into point clouds. Since the raw point clouds contained significant noise and background points, a three-step filtering process was applied: (1) removing background and non-plant points, (2) eliminating outliers using a Statistical Outlier Removal (SOR) filter, and (3) a Voxel-Grid filter with a voxel size of 0.006m was applied to reduce point cloud density. Each plant was observed from 135 evenly sampled viewpoints arranged in a 15×9 grid spanning a 0.6m (vertical) × 0.36m (horizontal) plane, located 0.5m in front of the plant (Fig. 4.7b). Viewpoints were spaced approximately 0.04m apart. At each grid position, five predefined pan-tilt angles $\{(-15°, 0°), (15°, 0°), (0°, 0°), (0°, 15°), (0°, 15°)\}$ were used to collect point clouds, resulting in 675 views per plant. However, for NBV planning, only the 135 perpendicular views were used, as angle variation was not considered during action selection. The angled views were used solely to generate ground-truth point clouds for evaluation (details in section 4.2.2.6).

Unlike the simulation study, where plant poses could be randomized in each reconstruction cycle to introduce variation, the real plant data was collected with fixed poses, limiting the evaluation of the method' s ability to handle variation to the ten plants. To address this issue, a 5-fold cross-validation approach was applied. First, T21-T30 plants were categorized into two groups by height: short plants (T21, T22, T24, T29, T30; 0.98m-1.05m above the growing substrate) and tall plants (T23, T25,

T26, T27, T28; 1.06m-1.22m). Then, these plants were randomly divided into five sets, each containing one plant from each height group: Set 1 (T21, T23), Set 2 (T22, T25), Set 3 (T24, T26), Set 4 (T29, T27), and Set 5 (T30, T28). This resulted in five cross validation runs, where in each run, one set was used for testing, and the remaining sets were used for training. In experiment RW, a total of 125 repetitions (5 runs x 25 repetitions) were conducted, and the final results were averaged across all repetitions.

View space V_{real}, shown in Fig. 4.7b, was used for the real-world plants. Additionally, to accommodate different plant heights, the view space was adjusted. For tall plants, views from row 1-15 and column 1-9 were used. For short plants, views from row 1-12 and column 1-9 were used. This also enabled evaluation of the method's scalability in reconstructing plants of different sizes.

To provide a more comprehensive evaluation of the proposed SSL-Local-NBV, additional real-world experiments were conducted to further evaluate its robustness to plant variation and high occlusion levels. These experiments used seven real plants collected using the same method as in Fig. 4.7. However, as shown in Fig. 4.8, these plants exhibited greater morphological variation and occlusion. During 3D reconstruction, the view space for plants T31–T35 included rows 1–15 and columns 1–8, as in Fig. 4.7. For the more horizontally spread plants T36 and T37, the view space was extended to columns 1–9. Plants T31–T32 and T34–T36 were used for training, while T33 and T37 were reserved for evaluation. A maximum of 15 views was allowed for T31–T35, and 20 views for the larger plants T36–T37.

4.2.2.3 Action space

A consistent action space $\mathcal{A}$={'up-1','down-1','left-1','right-1','up-3','down-3','left-3','right-3'} with eight actions (A=8) was defined for camera movement across all three experiments (see Fig. 4.1). Actions 'up-1' to 'right-1' represented small movements, moving the camera to an adjacent grid point in the up, down, left, or right direction, respectively. Meanwhile, actions 'up-3' to 'right-3' represented large movements, moving the camera by 3 grid points in the corresponding direction. This combination allowed the method to handle targets of varying sizes and avoid local optima—a common issue in local view planning, as large movements may help the camera escape such regions.

4.2.2.4 Implementation details of self-supervised learning in experiments

Table 4.1 lists the parameters and hyperparameters used for the experimental setup and self-supervised learning. SSL-Local-NBV was designed for continuous online

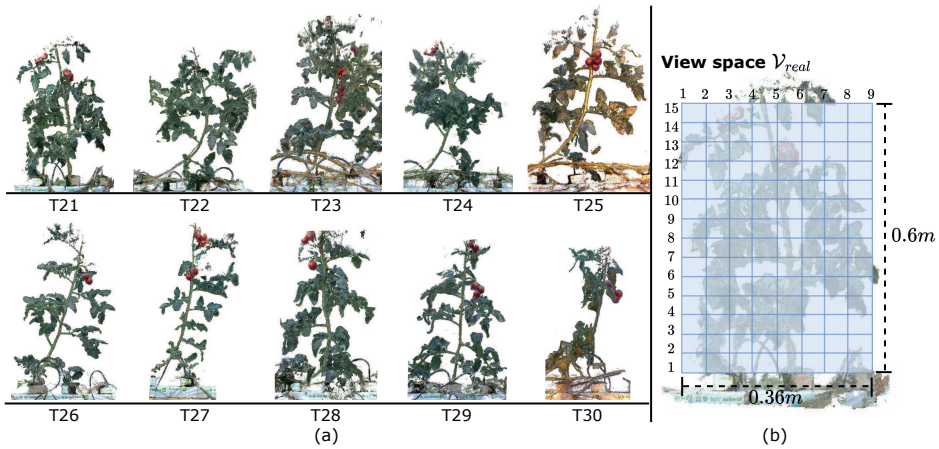


Figure 4.7: Setup for real-world experiment. The plants used in the real-world experiment are displayed in (a). The view space used for data collection is displayed in (b). For each real-world plant, 135 views (15 rows x 9 columns) arranged in the plane in front of the plant were used for point cloud collection.

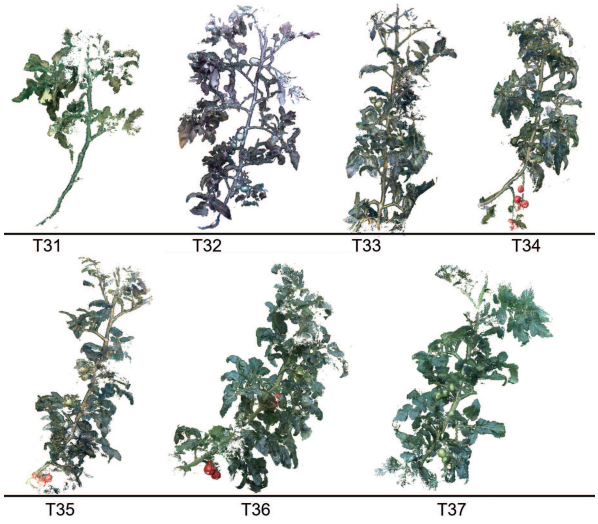


Figure 4.8: Real plants with more occlusions for real-world experiments.

learning. However, due to the finite number of target plants, a maximum of 5,000 reconstruction cycles ($c \leq 5000$) was set for experiment S1. In experiment S2, the network from S1 was fine-tuned with 2,000 additional cycles, and in experiment RW, the network from S2 was further fine-tuned with another 2,000 cycles. From an implementation perspective, the 7,000 cycles in simulation S1 and S2 can be considered a pre-training phase to support faster network convergence in the real world. Online

learning efficiency was not the focus of this study, as it was thoroughly evaluated in (Ci et al., 2025). The number of cycles used here was sufficient for convergence and could be reduced in practice using an early stopping mechanism. In cross-crop test of experiment S1 on pepper plants, the network was trained with 2000 reconstruction cycles from scratch due to a smaller plant set.

In experiment S1, each cycle allowed up to $i^{\max}$=15 iterations. In S2, $i^{\max}$ was adjusted based on plant size: 10 for small, 15 for medium, and 20 for large plants. In RW, $i^{\max}$ was set to 15 for short and 20 for tall plants. The network was updated once per reconstruction, with the update frequency matching $i^{\max}$ for each experiment. Before being fed into the network for IG prediction, the accumulated point cloud P^{acc} was downsampled to a fixed size to improve feature extraction efficiency. In experiments S1 and S2, 512 points were randomly selected from P^{acc}, as input. In experiment RW, 1,024 points were sampled to preserve more detail of the plants and reduce the impact of noise.

In step (2), parameters $\epsilon_0, \epsilon^{\min}$, and γ were set to 1.0, 0.2, and 0.98, respectively. In step (3), the point cloud from each view was pre-processed to reduce point density and filter noise (see sections 4.2.2.1 and 4.2.2.2 for details for simulation and real-world experiments). In step (4), the threshold δ was set equal to the voxel size: 0.003m in simulation and 0.006m in the real world. In step (5), the buffer size l was fixed at 10,000, and in step (6), the batch size n was set to 32.

Table 4.1: (Hyper)parameters used for experimental setup and self-supervised learning.

(Hyper) Parameters	**Experiment S1**	**Experiment S2**	**Experiment RW**
Action space	8	8	8
View space	12×12	Small:8×12, Med:12×12, Large:18×12	Small:12×9, Large:15×9
Point density	0.003m	0.003m	0.006m
Point cloud size	512	512	1024
Exploration rate	Initial:1.0, Min:0.2	Initial:1.0, Min:0.2	Initial:1.0, Min:0.2
Decay rate	0.98	0.98	0.98
Batch size	32	32	32
Buffer size	10,000	10,000	10,000
Training frequency	15	Small:10, Med:15, Large:20	Small:15, Large:20
Training cycles	5,000	2,000	2,000
Maximum views	15	Small:10, Med:15, Large:20	Small:15, Large:20

4.2.2.5 Baseline methods

The proposed SSL-Local-NBV method was compared to the following baseline methods:

- **Voxel-Global-NBV:** A voxel-based NBV method for global view planning (Burusa et al., 2024c). It selected the next best view based on an occupancy grid and ray-casting, choosing the view with the highest IG as the next view. All grid points within the view space were considered as candidate views.
- **Voxel-Local-NBV:** A voxel-based NBV method adapted from Voxel-Global-NBV for local view planning. It estimated the IG for all views corresponding to possible actions and selected the action with the highest IG to move the camera.
- **SSL-Global-NBV:** It used a deep neural network to predict IG for all candidate views, selecting the one with the highest IG as the next view (Ci et al., 2025). All grid points within the view space were considered as candidate views. As this method requires a fixed number of views, in experiment S2, the same view space V_{med} was used for all different-sized plants during online learning and testing.
- **Random global view planning:** The next view was randomly selected from the entire grid of views within the view space.
- **Random local view planning:** The next view was determined by randomly selecting an action from the action space to move the camera.

SSL-Global-NBV (Ci et al., 2025) and Voxel-based NBV (Burusa et al., 2024c) are state-of-the-art methods in plant reconstruction. SSL-Global-NBV is efficient in NBV planning and utilizes self-supervised learning, while Voxel-based NBV demonstrated high accuracy in IG estimation. Combined with the non-NBV methods, these baselines provide a comprehensive basis for evaluating the proposed SSL-Local-NBV in terms of IG prediction speed, reconstruction efficiency, scalability, and trajectory performance.

4.2.2.6 Evaluation metrics

- **Plant Reconstruction Ratio (R):** This metric measured the proportion of the plant successfully reconstructed, with higher values indicating better reconstruction quality. It was calculated as $R = |P^o \cap P^{acc}|/|P^o|$, where P^o represented the complete point cloud of the plant. In simulations, P^o was generated by uniformly sampling points with density of 0.003m from the mesh surface. In real-world experiments, where mesh models were unavailable, P^o

was constructed using COLMAP (Schonberger & Frahm, 2016), a structure-from-motion-based 3D reconstruction algorithm, taking all 675 RGB images and corresponding camera pose as inputs. The resulting model was then filtered using the three-step process described in section 4.2.2.2 The intersection $P^o \cap P^{acc}$ was computed as described in step (4) of section 4.2.1.2, with δ set to 0.003m for simulations and 0.006m for real-world experiments.

- **IG Prediction Speed:** This metric measured the time required to predict IG for all candidate views during a view planning iteration. It reflected the efficiency of the method in viewpoint planning. Details of the device used in this study are provided in section 4.2.2.1.

- **Area Under View-Reconstruction Curve (AUC@VR):** While the reconstruction ratio R provided a snapshot at a specific view step, it lacked a reflection of the entire reconstruction cycle. AUC@VR measured the area under the curve plotting R against the number of views, reflecting reconstruction efficiency across the entire trajectory with a single metric. A higher AUC@VR indicates faster, more efficient reconstruction. It is calculated as:

$$AUC@VR = \frac{1}{i^{\max} - 1} \sum_{i=1}^{i^{\max}-1} \frac{R_{i+1} + R_i}{2} \tag{4.8}$$

  where R_i represents the plant reconstruction ratio at iteration i.

- **Trajectory Distance:** This metric quantified the total camera movement by computing the Euclidean distance between consecutive camera positions along the trajectory.

- **Trajectory efficiency:** This metric evaluated how effectively camera movement contributes to plant reconstruction. It was calculated as AUC@VR / trajectory distance, indicating how much AUC@VR is achieved per meter of camera movement. Higher values represent greater efficiency in reconstruction relative to camera movement.

4.3 Results

4.3.1 Experiment Simulation 1

Experiment S1 evaluated the trajectory efficiency of SSL-Local-NBV versus global view planning methods. To provide direct insights, Fig. 4.9 illustrates representative view trajectories generated by SSL-Local-NBV (Fig. 4.9a) and SSL-Global-NBV (Fig. 4.9b), randomly selected from the test reconstruction repetitions. Metrics including

trajectory distance, reconstruction ratio, AUC@VR, and trajectory efficiency were calculated. SSL-Local-NBV produced a smoother trajectory, with 6.66m less camera movement than SSL-Global-NBV, resulting in a 335% higher trajectory efficiency. This discrepancy in SSL-Global-NBV was due to its selection of the next view from the entire view space, which caused view transitions that traversed the entire view space and resulted in unnecessary camera movements.

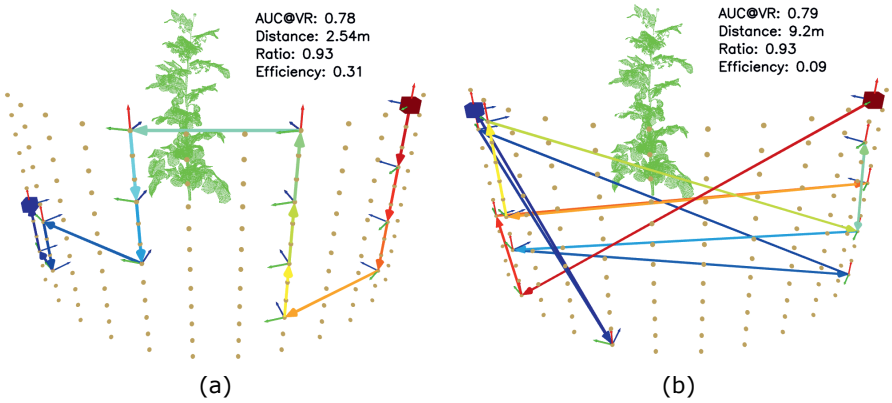


Figure 4.9: View trajectories for plant reconstruction. (a) Trajectory generated by SSL-Local-NBV. (b) Trajectory generated by SSL-Global-NBV. The view transitions from dark red to dark blue indicate the trajectory from the first to the last view. The red cube represents the starting view, while the blue cube marks the end view.

Fig. 4.10 compares the plant reconstruction ratios of SSL-Local-NBV with other local and global view planning methods. Results represent the average of 25 reconstruction repetitions on two test plants with varied poses, including 95% confidence intervals. As shown in Fig. 4.10a, SSL-Local-NBV achieved $R \geq 0.9$ after 9 views, 3 views faster than Voxel-Local-NBV method, while Random-Local failed to reach this ratio. After 15 views, SSL-Local-NBV achieved $R = 0.94$, similar to Voxel-Local-NBV (p-value=0.72) and significantly higher than Random-Local method ($R = 0.76$m, p-value=$1.05\text{x}10^{-5}$). In this work, p-values were computed using two-sample t-tests to assess statistical significance between methods. Based on Fig. 4.10a and Fig. 4.10b, global view planning methods consistently achieved higher reconstruction efficiency than their local counterparts, as they can select views from the entire space, while local methods are limited to adjacent views. However, Fig. 4.10b shows that SSL-Local-NBV performed comparably in reconstruction efficiency to SSL-Global-NBV and only slightly below the highest-performing Voxel-Global-NBV. This, combined with its high trajectory efficiency (discussed in the next paragraph), highlights SSL-Local-NBV' s suitability for plant reconstruction.

Table 4.2 presents a quantitative comparison of AUC@VR, trajectory distance, and

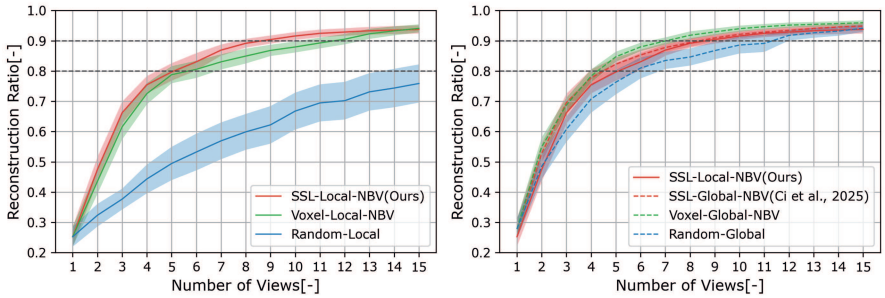


Figure 4.10: Plant reconstruction ratio of SSL-Local-NBV compared to baseline methods. (a) Comparison with local view planning methods. (b) Comparison with global view planning methods. The shaded region shows the 95% confidence interval over 25 reconstruction repetitions.

IG prediction speed between SSL-Local-NBV and other methods. SSL-Local-NBV achieved an AUC@VR of 0.82, which was higher than the other local view planning methods, and 0.03 and 0.01 lower than Voxel-Global-NBV and SSL-Global-NBV, respectively. Compared to global view planning methods, which required 6.81-8.33m of camera movement, SSL-Local-NBV reduced trajectory distance by 56%-64%. As a result, it achieved a trajectory efficiency of 0.27, meaning that each meter of camera movement contributed 0.27 AUC@VR—about 250% higher than global methods. While SSL- and Voxel-based NBV methods (both local and global) achieved similar AUC@VR, SSL-based methods were significantly faster in IG prediction. In local view planning, Voxel-Local-NBV took 0.72s to estimate IG for candidate views, about 170 times slower than SSL-Local-NBV. Notably, this IG prediction time was measured for a single NBV iteration where the network predicted IG for all candidates, with network training deactivated, as it was only needed for novel plants. In global view planning, Voxel-Global-NBV was 3,400 times slower due to estimating IG for each view individually and processing a larger candidate set (144 views). In contrast, SSL-Global-NBV used a neural network to predict IG for all views simultaneously and was minimally affected by candidate set size.

Furthermore, to provide a comprehensive evaluation of the proposed SSL-Local-NBV, a cross-crop test and a noise robustness test were conducted below.

As shown in Fig. 4.11, a similar pattern to Table 4.2 was observed in pepper plants: SSL-Local-NBV outperformed other local view planning methods and achieved reconstruction efficiency comparable to global NBV methods, demonstrating its robustness and effectiveness. All methods showed lower AUC@VR compared to on the tomato plants (Table 4.2), due to the increased occlusions in the pepper plants. This problem could be addressed in future work by incorporating views on different angels (e.g.,

look up and look down).

Table 4.2: Results from Experiment S1 comparing SSL-Local-NBV with baseline method. Bold values indicate the best results within local or global view planning categories.

Methods	AUC@VR [-]	Trajectory distance[m]	Trajectory efficiency[-]	Inference time[s]
Random-Local	0.57	2.06	**0.28**	-
Voxel-Local-NBV	0.79	2.49	**0.32**	0.72
SSL-Local-NBV	**0.82**	2.98	0.27	**0.0042**
Random-Global	0.79	**6.81**	**0.12**	-
Voxel-Global-NBV	**0.85**	8.33	0.10	15.82
SSL-Global-NBV	0.83	7.85	0.11	**0.0046**

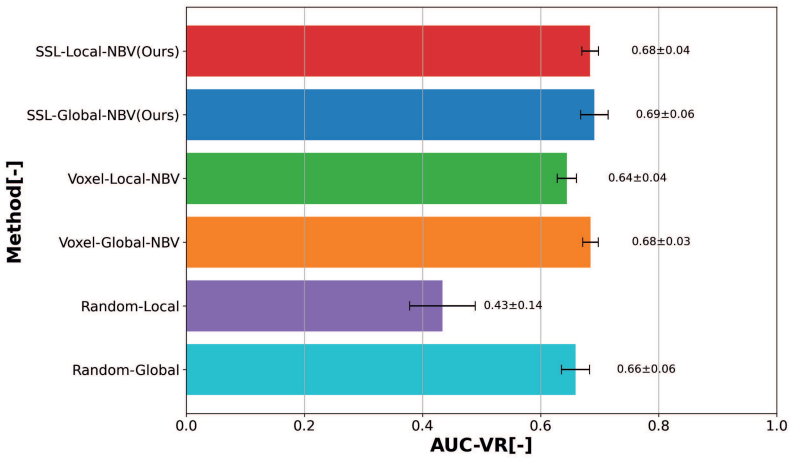


Figure 4.11: AUC@VR of SSL-Local-NBV compared to the baseline methods on pepper plants.

Based on Fig. 4.12, no significant differences were shown for the methods under varying noise levels. This indicates that both learning-based and voxel-based NBV methods could effectively handle noisy conditions, even under heavy noise. We attribute this robustness to the fact that point cloud boundaries are more critical than fine details in whole-plant reconstruction NBV tasks. However, in tasks focusing on specific plant parts (e.g., nodes or fruits), noise may impact method performance. Voxel-based NBV methods demonstrated better stability than learning-based approaches, possibly because continuously updating occupancy probabilities of grid cells allows them to better deal with noise.

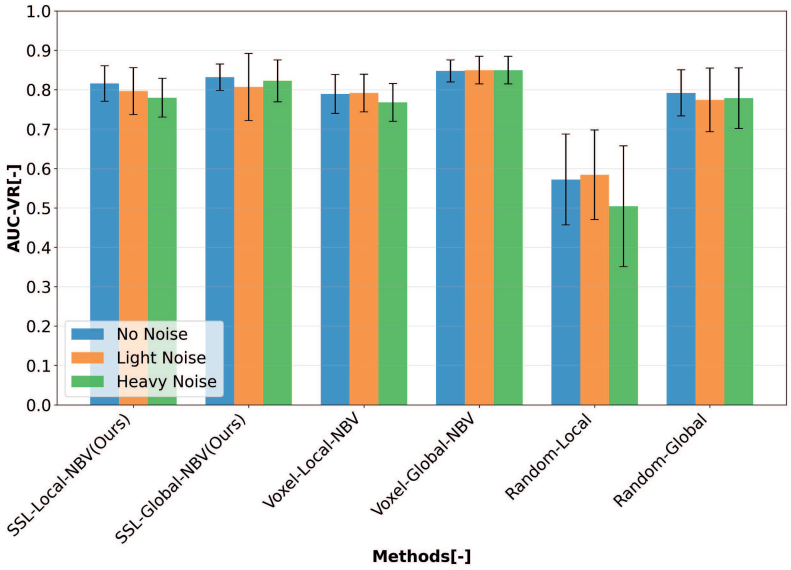


Figure 4.12: Comparison of SSL-Local-NBV and baseline methods in AUC@VR under different noise levels.

4.3.2 Experiment Simulation 2

Experiment S2 evaluated the scalability of SSL-Local-NBV for plant reconstruction across varying plant sizes and viewing settings. To provide general insights, Table 4.3 presents the average results for AUC@VR, trajectory distance, and IG prediction time (as in experiment S1) across 25 test repetitions involving plants of different sizes. The results for specific plant sizes are shown in Table 4.4 and Table 4.5. While Voxel-Local-NBV had a slower IG prediction speed, it reached a 0.80 of AUC@VR and was not limited by plant sizes. SSL-Local-NBV achieved an average AUC@VR of 0.81, which was similar to Voxel-Local-NBV (p-value=0.65) and significantly higher than Random-Local methods (p-value=$6.96\text{x}10^{-7}$), indicating strong scalability across different plant sizes. Voxel-Global-NBV was used as the upper baseline for AUC@VR. Although SSL-Local-NBV achieved an AUC@VR 0.05 lower than the Voxel-Global-NBV, it achieved a 267% higher trajectory efficiency and improved IG prediction speed by approximately 2,950 times.

SSL-Global-NBV was excluded from this table due to its limited adaptability to varying number of views required for different plant sizes. In contrast, SSL-Local-NBV adapted to different plant sizes and view settings without needing any modifications to its network architecture or hyperparameters, demonstrating higher scalability.

Table 4.4 further analyzes SSL-Local-NBV' s reconstruction performance for specific

plant sizes. It shows that SSL-Local-NBV achieved similar AUC@VR to Voxel-Local-NBV, effectively scaling across various plant sizes. Both methods showed a decrease in AUC@VR for large plants, which is likely due to the fact that large plants usually have more occlusions, making reconstruction more challenging.

Table 4.3: Results from Experiment S2 comparing SSL-Local-NBV to baseline methods. Bold values indicate the best results within local or global view planning categories.

Methods	AUC@VR [-]	Trajectory distance [m]	Trajectory efficiency [-]	Inference time [s]
Random-Local	0.58	**1.91**	0.30	-
Voxel-Local-NBV	0.80	2.39	**0.34**	0.73
SSL-Local-NBV	**0.81**	2.53	0.32	**0.0051**
Random-Global	0.81	**6.13**	**0.13**	-
Voxel-Global-NBV	**0.86**	7.27	0.12	15.05

Table 4.4: Comparison of AUC@VR between SSL-Local-NBV and baseline NBV methods across different plant sizes. Bold values indicate the best results within local or global view planning categories.

Methods	Small plants	Medium plants	Large plants	Average
SSL-Local-NBV	**0.85**	**0.81**	0.77	**0.81**
Voxel-Local-NBV	**0.85**	0.80	**0.78**	**0.81**

Table 4.5: Comparison of AUC@VR between SSL-Local-NBV and SSL-Global-NBV across different plant sizes. Bold values indicate the best results within local or global view planning methods.

Methods	Small plants	Medium plants	Large plants	Average
SSL-Local-NBV	**0.85**	0.81	**0.77**	**0.81**
SSL-Global-NBV	0.78	**0.83**	0.71	0.77

Table 4.5 compares SSL-Local-NBV and SSL-Global-NBV in plant reconstruction across various plant sizes. SSL-Global-NBV, limited by a fixed number of views, was trained and tested using the same view space V_{med} for all plant sizes. In contrast, SSL-Local-NBV used size-specific view spaces: $V_{\text{small}}, V_{\text{med}}$, and V_{large} for small, medium, and large plants, respectively. SSL-Local-NBV achieved an AUC@VR 0.02 lower than SSL-Global-NBV on medium plants but outperformed it by 0.07 and 0.06 on small and large plants, respectively. These differences were statistically significant across all plant sizes (small: p-value=1.19×10^{-7}; medium: p-value=6.1×10^{-4}; large: p-value=7.0×10^{-3}). These results indicate that while SSL-Global-NBV performed well

for specific plant sizes with fixed view configurations, it lacked scalability to handle varying plant sizes. In contrast, SSL-Local-NBV, which was not constrained by a fixed number of views, demonstrated higher overall AUC@VR and scalability for different plant sizes.

4.3.3 Experiment Real World

Simulation experiments provided insights into the method' s performance against baseline approaches, enabling efficient and comprehensive evaluation using perfect ground truth, which is difficult to obtain in real-world settings. To directly evaluate the 3D reconstruction efficiency of SSL-Local-NBV in real-world scenarios, experiments were conducted on real plants. Real-world study 1 was performed on plants maintained under standard routines (Fig. 4.7), demonstrating the method' s effectiveness in practical applications. Subsequently, additional experiments (study 1) were conducted on plants with greater morphological variation and higher levels of occlusion (Fig. 4.8) to assess the robustness of SSL-Local-NBV.

4.3.3.1 Real-World Study 1

As shown in Table 4.6, SSL-Local-NBV achieved an AUC@VR that was 0.08 lower than the top-performing Voxel-Global-NBV but reduced camera movement by 3.96m, resulting in an around 300% higher trajectory efficiency. Compared to Voxel-Local-NBV, SSL-Local-NBV had a slightly lower AUC@VR ($^{-}$0.03) and trajectory efficiency but was 76 times faster in IG prediction. SSL-Local-NBV achieved a 0.17 higher AUC@VR than Random-Local, further confirming its better capability in 3D plant reconstruction compared to non-NBV methods. The IG prediction time nearly doubled in real-world experiments compared to simulation because a larger input size of the point cloud was used. Specifically, 1024 points were sampled from the point cloud, which was twice the number used in simulation (512 points). This increased the number of neural network computations, resulting in longer prediction time. Despite this, the 9ms IG prediction time remained efficient and did not hinder real-time NBV planning.

Notably, Voxel-Local-NBV outperformed SSL-Local-NBV in AUC@VR in the real world, which contrasts with simulation results. This may be due to Voxel-based NBV' s iterative occupancy updates, which help filter out noise in real-world point clouds, leading to more accurate IG prediction. All methods showed a decreased AUC@VR compared to the simulation results. We attribute this to the difference in viewpoint sampling: in the real-world setting, viewpoints were sampled on a planar surface in front of the plants, whereas the simulation used a semi-cylindrical view space that enabled additional side-view observations, contributing to more comprehensive 3D

reconstruction. Another likely reason was the presence of noise in the real-world ground-truth point clouds, which could be absent in the reconstruction, thereby reducing reconstruction accuracy and AUC@VR compared to the simulation.

Fig. 4.13 presents representative examples of 3D plant reconstructions using SSL-Local-NBV. The method consistently achieved reconstruction ratios above 0.80 per cycle. We considered this level of reconstruction sufficient for most agricultural robotic operations like phenotyping, as it successfully reconstructed most key plant parts including leaves and fruit nodes, which are often occluded by leaves.

Table 4.6: Results of Experiment RW comparing SSL-Local-NBV with baseline methods in AUC@VR, trajectory distance, and IG planning speed. Bold values indicate the best results within local or global view planning methods.

Methods	**AUC@VR [-]**	**Trajectory distance [m]**	**Trajectory efficiency [-]**	**Inference time [s]**
Random-Local	0.48	**1.32**	0.36	-
Voxel-Local-NBV	**0.68**	1.91	0.36	0.68
SSL-Local-NBV	0.65	1.68	**0.39**	**0.0090**
Random-Global	0.67	**4.08**	**0.16**	-
Voxel-Global-NBV	**0.73**	5.64	0.13	12.68

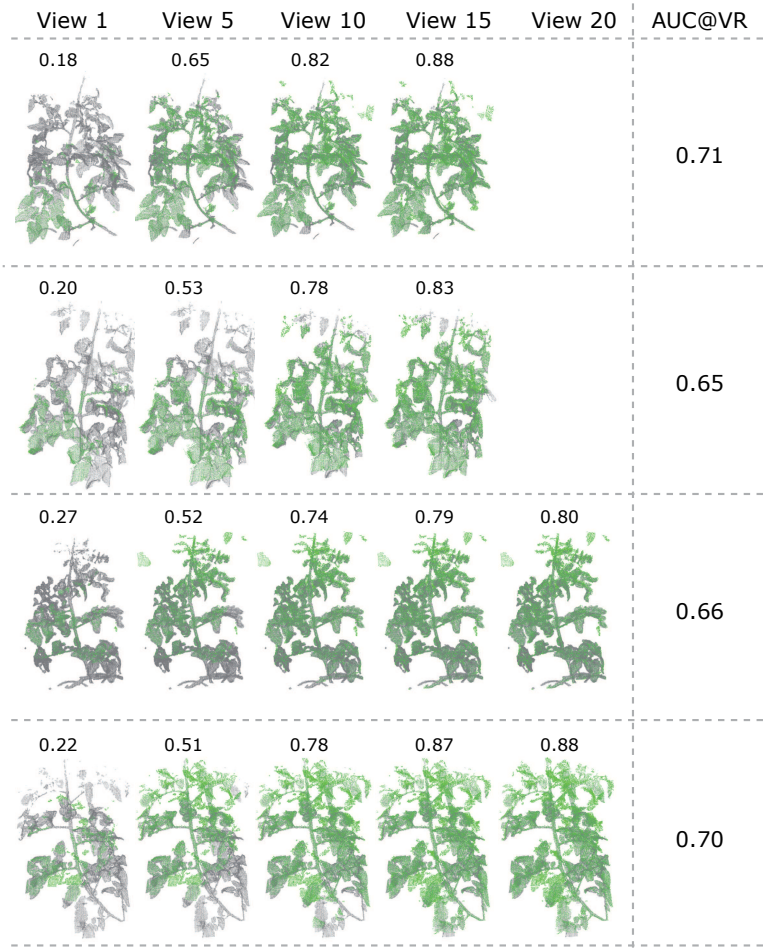


Figure 4.13: Qualitative analysis of reconstructed plants using SSL-Local-NBV. Gray points represent ground truth, and green points represent the reconstruction. Plant reconstruction ratios are shown above each reconstruction.

4.3.3.2 Real-World Study 2

Fig. 4.14 shows the reconstruction efficiency of SSL-Local-NBV and baseline methods on real plants with significant morphological complexity and higher occlusion. A similar AUC@VR pattern between the methods to Table 4.6 was observed, further confirming the consistency and reliability of the proposed method. Specifically, all NBV methods achieved higher reconstruction efficiency than non-NBV methods. SSL-Local-NBV achieved comparable reconstruction efficiency to Voxel-Local-NBV and slightly lower than the top-performing Voxel-Global-NBV. However, as shown in earlier results, SSL-Local-NBV offered faster IG prediction and significantly bet-

ter trajectory efficiency than Voxel-Global-NBV. Due to higher levels of occlusion, all methods showed a reduction in AUC@VR. Nonetheless, the comparative performance among methods remained consistent, providing support for the conclusions.

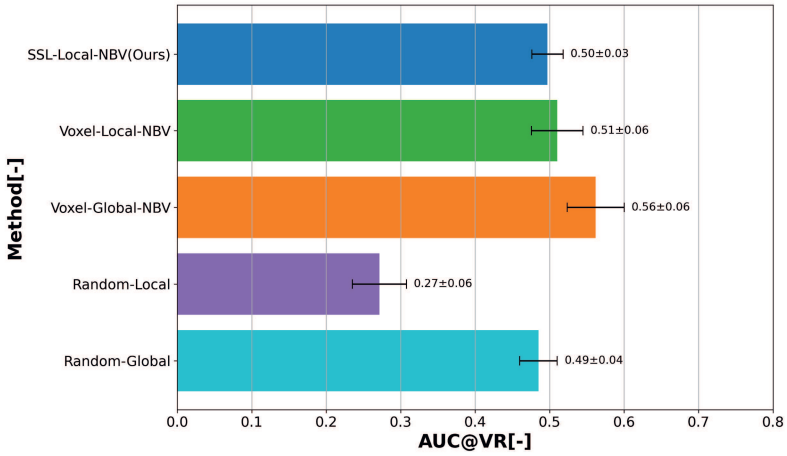


Figure 4.14: AUC@VR of SSL-Local-NBV compared to the baseline methods on real plants with significant morphological complexity and higher levels of occlusion.

According to Table 4.7, similar to Table 4.6, the proposed SSL-Local-NBV reduced trajectory distance by 5.86m compared to Voxel-Global-NBV, resulting in approximately 250% higher trajectory efficiency for heavily occluded plants. Additionally, SSL-Local-NBV achieved IG prediction speeds approximately 110 and 2230 times faster than SSL-Local-NBV and SSL-Global-NBV, respectively.

Table 4.7: Results of Experiment RW with heavy plant occlusions comparing SSL-Local-NBV with baseline methods in trajectory distance, and IG planning speed. Bold values indicate the best results within local or global view planning methods.

Methods	Trajectory distance [m]	Trajectory efficiency [-]	Inference time [s]
Random-Local	**2.00058**	0.14	-
Voxel-Local-NBV	3.03	**0.17**	0.69
SSL-Local-NBV	3.05	0.16	**0.0062**
Random-Global	**6.61**	**0.073**	-
Voxel-Global-NBV	8.91	0.063	13.86

4.4 Discussion

This work proposed SSL-Local-NBV, an algorithm for efficient and scalable 3D reconstruction of plants of varying sizes using local view planning. Utilizing self-supervised learning, SSL-Local-NBV automated both the pre-training and fine-tuning processes, enabling robotic continuous online learning during plant reconstruction. Comprehensive evaluations were conducted in both simulation and real-world scenarios.

4.4.1 Comparison with relevant studies

Compared to SSL-Global-NBV (Ci et al., 2025)), SSL-Local-NBV offered greater scalability across varying plant sizes and viewpoint settings without requiring changes to the network architecture or hyperparameters. In contrast, SSL-Global-NBV relied on a fixed number of candidate views, limiting its adaptability. While Voxel-Global-NBV was flexible in handling plant size and view settings, its IG prediction time increased linearly with the number of candidate views, making it time-intensive. Thus, SSL-Local-NBV provided a more scalable and efficient solution for plant reconstruction.

Compared to the baseline global NBV methods (Voxel-Global-NBV and SSL-Global-NBV), SSL-Local-NBV significantly improved trajectory efficiency by 180% – 300%, with only a 3% – 7% drop in plant reconstruction ratio, highlighting its practical effectiveness. This did not fully align with the work of Lehnert et al. (2019) and Zaenker et al. (2021b), who suggested that local view planning was only effective for local optimization and not suitable for larger-scale targets. SSL-Local-NBV's capability for whole-plant reconstruction stemmed from its multi-scale action strategy, where larger action steps help the robot escape local optima. Additionally, different to their works that focused on fruits where information was concentrated in local regions, our method reconstructed entire plants, with each plant part contributing IG, reducing the likelihood of getting stuck in local optima.

In this work, SSL-Global-NBV showed slightly lower reconstruction efficiency than Voxel-Global-NBV (see Fig. 4.10b and Table 4.2), differing from the results of Ci et al. (2025), where both methods performed comparably. This difference is attributed to our use of a much larger candidate view set (144 vs. 33 in Ci et al.), which made view selection more challenging for SSL-Global-NBV. In both simulation and real-world experiments, SSL-Local-NBV achieved reconstruction efficiency and quality comparable to Voxel-Local-NBV but with significantly faster IG prediction. This supports the findings of Han et al. (2022), who showed that learning-based NBV methods can substantially greatly accelerate IG prediction over traditional voxel-based ray-casting approaches. However, unlike Han et al., who only evaluated reconstructed plants qual-

itatively, our method underwent a comprehensive evaluation against baseline methods in both simulation and real-world settings.

In recent years, local NBV planning methods based on neural radiance fields (NeRF) have gained significant attention, as demonstrated in the works of Jin et al. (2024),Lee et al. (2022), and Xue et al. (2024). A comparison with these methods was excluded from this study, as most NeRF-based NBV approaches require network re-training at each NBV iteration, making them inefficient for real-time view planning. Additionally, their primary focus is on learning implicit object representations, which limits their suitability for tasks that demand accurate and detailed explicit 3D outputs, such as point clouds or voxel maps, which are crucial for agricultural applications.

Although the online learning capability of SSL-Local-NBV was not tested in this study, it was extensively evaluated by Ci et al. (2025), who showed that SSL-based NBV methods can automatically adapt to novel plants and environments while significantly reducing the need for IG annotations compared to offline strongly supervised learning. Overall, SSL-Local-NBV provides a scalable, efficient, and flexible solution for 3D plant reconstruction. With its online self-supervised learning capabilities, SSL-Local-NBV has strong potential to automate agricultural robotic tasks such as plant reconstruction, phenotyping, and harvesting in dynamic environments.

4.4.2 Impact of experimental conditions

In experiment RW, the self-supervised training used a maximum of 2,000 reconstruction cycles ($c^{\max}$) for optimal network performance and prediction accuracy. While this number may seem high for real-world deployment, the method supports continuous online learning, allowing the robot to continuously improve itself without human intervention. Moreover, this value was treated as an upper bound to ensure full network convergence. Ci et al. (2025) demonstrated that comparable performance in plant reconstruction could be achieved with just 10%-20% of the training cycles compared to using all cycles with minimal reduction on reconstruction performance, suggesting significant practical reductions of training cycles are feasible.

In the simulation experiments, nineteen manually modeled digital plants were used: ten normal sized tomato plants, six modified sized tomato plants, and three pepper plants. Each plant exhibited substantial morphological complexity and occlusion. Although these digital plants do not fully resemble real plants, the purpose of the simulation experiments was to enable safe, efficient, controlled, and repeatable evaluation of the SSL-Local-NBV method' s capability for 3D reconstruction through online learning, in comparison with state-of-the-art methods. Furthermore, given our focus on whole-plant reconstruction, where precise geometric details are less critical

than structural cues (e.g., the boundary) for guiding view planning, the simulated plants were considered sufficiently accurate for this purpose. Moreover, simulation-based evaluation is a widely adopted practice in NBV research, as seen in studies by Burusa et al. (2024b,c,a), Ci et al. (2025), Han et al. (2022), Lehnert et al. (2019), Zapotezny-Anderson & Lehnert (2019), Zeng et al. (2022), due to the challenges of acquiring perfect ground-truth. Finally, to ensure robust conclusions about real-world performance, we conducted extensive experiments on real plants. The combination of simulated and real-world evaluations provides a solid basis for drawing comprehensive conclusions about the method' s effectiveness.

In the cross-crop test, all methods showed lower AUC@VR on pepper plants compared to tomato plants. This reduction is attributed to the higher level of occlusion in pepper plants, which makes 3D data collection more challenging. These findings are consistent with the results reported in Ci et al. (2025), where all methods demonstrated reduced reconstruction efficiency on pepper plants.

In the noise robustness test, a similar performance pattern was observed under both light and heavy noise conditions, closely matching the results obtained with clean data. This suggests that the proposed method is robust to noisy point clouds and can effectively perform view planning even under significant noise levels.

Both SSL-Local-NBV and Voxel-Local-NBV exhibited a slight decrease in AUC@VR for large plants (Table 4.4), which is likely because large plants have more occlusions, making reconstruction more challenging. In this work, only the perpendicular views were used for view planning. In future work, views from different angles (e.g., looking up or down) could be included to help better view the occluded areas.

Although a three-step noise removal process was applied for the real-world ground-truth point clouds, they still contained noise. This could result in an underestimation of the methods, as noise present in the ground-truth may not appear in the reconstructions, thereby lowering the calculated reconstruction ratio. This effect is evident in Table 4.6, where all methods showed a reduction in AUC@VR in real-world experiments compared to simulations. Nonetheless, SSL-Local-NBV consistently achieved a final reconstruction ratio above 0.8 (refer to Fig. 4.13), showing its capability in real-world applications. Notably, compared to absolute reconstruction ratios, we focused more on the relative performance between methods, as absolute ratios are affected by factors such as camera FOV, viewpoint distribution, and plant occlusion levels (see comparisons between Fig. 4.14 and Table 4.6. Relative performance offers a more consistent and reliable basis for evaluating the proposed method against state-of-the-art approaches.

Moreover, the ground-truth for the real-world plants was generated from RGB im-

ages using COLMAP, while the reconstructions were based on fused point clouds captured by the camera's depth sensor. This discrepancy caused mismatches between the ground-truth and reconstructed point clouds (Fig. 4.13), potentially leading to underestimation of the reconstruction ratio, as points present in the ground-truth might be missing in the sensor-captured point clouds. This could also contribute to the consistently lower AUC@VR observed across all methods in the real-world compared to the simulation results. While perfect ground-truth data for real plants does not exist, we believe this imperfection does not affect the fairness of the evaluation since all methods were equally impacted. Additionally, comprehensive evaluations in simulation using perfect ground-truth further ensure reliable evaluation.

To reduce the number of required annotations and facilitate efficient training data collection and online network training, a weakly-supervised learning approach was adopted, in which only the selected actions for camera movement were annotated with ground-truth IG. While fully annotating all possible actions for each training sample could improve sample efficiency and training speed to some extent, it would significantly increase the time required for training data collection (approximately 8 times). Additionally, the exploration–exploitation mechanism helps prevent suboptimal policies by ensuring that all actions have a chance to be selected, thereby improving IG prediction accuracy.

4.4.3 Limitations and recommendations

This study aims to reconstruct entire plants, with the resulting digital models intended for downstream tasks like plant phenotyping. This approach is most useful when all plant parts are important. However, for tasks like harvesting and de-leafing, where only specific plant parts are relevant, full-plant reconstruction can be inefficient. Burusa et al. (2024c) and Zeng et al. (2022) addressed this by integrating a target-driven mechanism into NBV systems for efficient search and reconstruction of specific parts. Future work could adapt this approach for SSL-Local-NBV to improved efficiency in targeted applications.

Integrating prior plant knowledge into view planning could further enhance reconstruction efficiency. For example, plants in commercial greenhouses often exhibit structured growth patterns, such as consistent growth directions and predictable fruit positions. This information is valuable for target-driven NBV planning. Future research could investigate the integration of prior plant knowledge into 3D reconstruction tasks to improve efficiency.

In this work, the GRU module, a lightweight recurrent neural network, was employed to store the trajectory information of previously visited views. This choice was made

to support faster online learning. In the future, other recurrent architectures such as Long Short-Term Memory (LSTM) Hochreiter & Schmidhuber (1997), or transformer-based architectures Vaswani et al. (2017) could be explored, as they may provide a stronger capacity for modeling longer viewpoint sequences, potentially leading to higher reconstruction performance.

This work focuses on 3D reconstruction under the assumption that plants remain static. However, in real-world scenarios, plant motion, such as waving caused by airflow, introduces additional challenges. Ideally, plant dynamics should be considered in both simulation and real-world experiments. To the best of our knowledge, research on 3D reconstruction for dynamic plants is still limited. Future work could explore incorporating mild plant motion into simulation and real-world settings to better reflect practical conditions.

The simulated plants used in this study do not fully capture the morphological structure of real plants. Although we believe that the use of more realistic simulated plants would not significantly change the relative performance patterns between methods, the absolute reconstruction ratios obtained may differ from those observed in real-world experiments and depends on experiments setting (e.g., slightly lower across all methods as displayed in Fig. 4.14). Future work could explore incorporating more realistic models to provide a more accurate reflection of method performance prior to real-world experiments, thereby facilitating more efficient method development.

4.5 Conclusion

The efficiency of SSL-Local-NBV compared to global view planning methods (Q1) was evaluated in Experiment S1. SSL-Local-NBV significantly reduced camera movement by 62%–64%, with only a 1%–4% reduction in reconstruction performance, yielding a 245%–270% gain in trajectory efficiency. Compared to other local view planning methods, SSL-Local-NBV improved the plant reconstruction ratio by approximately 20% over the non-NBV method. While it achieved a comparable reconstruction ratio to Voxel-Local-NBV, SSL-Local-NBV was over 170 times faster in IG prediction. In addition, SSL-Local-NBV demonstrated scalability across different plant types and maintained effectiveness under noisy conditions. Experiment S2 addressed SSL-Local-NBV' s ability to handle plants of varying sizes (Q2). SSL-Local-NBV successfully adapted to different plant sizes without modifications to the network architecture or hyperparameters. Though it slightly underperformed on medium-sized plants (0.02 lower AUC@VR), it outperformed SSL-Global-NBV on small and large plants by 0.07 and 0.06 AUC@VR, respectively—resulting in an overall 5.2% improvement in

plant reconstruction efficiency across varying plant sizes. The question of how well SSL-Local-NBV performs in real-world 3D plant reconstruction (Q3) was addressed in Experiment RW. SSL-Local-NBV achieved an AUC@VR that was 0.08 lower than the highest-performing Voxel-Global-NBV but reduced trajectory distance by approximately 4 meters, resulting in a 300% improvement in trajectory efficiency. Compared to Voxel-Local-NBV, SSL-Local-NBV achieved IG prediction speeds 76 times faster. Additionally, it consistently reached an average plant reconstruction rate of over 80%, demonstrating its feasibility for real-world applications.

In conclusion, the proposed SSL-Local-NBV enables scalable and efficient 3D plant reconstruction in both simulation and real-world environments. The reconstructed models are expected to support downstream tasks such as automated plant phenotyping and, with minor modifications, other agricultural applications facing similar occlusion challenges, such as harvesting. Future work could explore integrating target-driven mechanisms or prior knowledge to enhance efficiency in target-specific 3D reconstruction.

Chapter 5

SSL-Semantic-NBV: A Self-Supervised Learning-Based NBV for Target-Aware 3D Reconstruction in Agricultural Robotics

This chapter has been accepted for publication as:

Ci, J., Smoleňová, K., Wang, X., Streit, A., van Henten, E. J., & Kootstra, G. (2026). SSL-Semantic-NBV: A Self-Supervised Learning-Based NBV for Target-Aware 3D Reconstruction in Agricultural Robotics. In *Proceedings of the 2026 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Accepted for publication.

Abstract

3D reconstruction is critical for many robotic operations. Next-best-view (NBV) methods can efficiently reconstruct complex scenes and reduce occlusions, but most existing approaches focus on reconstructing entire scenes. In practice, many tasks require reconstructing only specific parts of a scene, while whole-scene reconstruction adds unnecessary cost for irrelevant regions. This paper proposes SSL-Semantic-NBV, a learning-based NBV framework for efficient 3D reconstruction of targeted scene regions. To address the challenge that learning-based NBV methods require large amounts of training data, SSL-Semantic-NBV enables network training through robotic self-supervised learning. We evaluate SSL-Semantic-NBV in the use case of 3D reconstruction of tomato plants, targeting fruits, nodes, and the entire plant, while the approach is expected to generalize to other tasks where occlusion poses challenges and NBV needs to be target-related. In simulation, it improves reconstruction efficiency by 47%–114% over non-NBV methods and 9%–15% over existing NBV methods. In real-world experiments, it achieves improvements of 8%–16% and 3%–7% over non-NBV and NBV methods, respectively. The framework demonstrates more efficient network training compared to the state of the art, supported by a novel IG metric, loss function, and network architecture. To support real-world deployment, functional–structural plant (FSP) modeling is used to generate synthetic data for pretraining. Importantly, SSL-Semantic-NBV automates the training process, paving the way for robotic lifelong learning with minimal human intervention.

5.1 Introduction

Robots are increasingly used in agriculture to automate tasks such as harvesting, deleafing, and phenotyping. A key component of these systems is accurate perception, where robots must detect specific parts of the plant to perform specific tasks, e.g., fruits, nodes, and whole plants, and extract their spatial information, such as 2D and 3D poses or 3D geometry (Ci et al., 2024; Zeng et al., 2022). This information can then be used to guide robotic manipulation. However, accurate perception in agriculture remains challenging due to the complex and variable structure of plants, which often leads to occlusions caused by leaves or other parts (Kootstra et al., 2021).

Next-best-view (NBV) planning, an active-perception method, addresses this challenge by iteratively repositioning the camera based on current and past observations to maximize information gain (IG) with as few views as possible (Mendoza et al., 2020; Zeng et al., 2020). NBV methods have also been investigated in agricultural robotics (Burusa et al., 2024a,c; Zapotezny-Anderson & Lehnert, 2019). Recently, learning-based NBV approaches, which use deep neural networks to predict IG, have gained interest due to their high computational efficiency of IG prediction (Zeng et al., 2020; Mendoza et al., 2020). However, their deployment in real-world scenarios is hindered by the difficulty of obtaining training sets. To address this, Ci et al. (2025) proposed SSL-NBV, a self-supervised NBV framework that enables robots to autonomously and continuously collect and annotate training data during NBV planning. Automatic ground-truth IG computation was performed by comparing the newly collected data with the data accumulated so far. This makes SSL-NBV well-suited for agricultural robotics, where environments are highly dynamic and robots must continuously adapt to changing situations. However, SSL-NBV was designed for whole-plant reconstruction, treating all plant parts as equally important. While effective for tasks such as phenotyping and whole-plant monitoring, this strategy can be inefficient for tasks that require reconstruction only of specific parts of the plant such as fruits or nodes. In this paper, we aim to extend SSL-NBV to target-aware NBV planning, focusing on the 3D reconstruction of specific plant parts, in order to minimize the number of viewpoints needed. Recent works on target-aware NBV in agriculture (Burusa et al., 2024c,a; Zeng et al., 2022) rely on occupancy grids and ray casting for IG estimation, but these methods are constrained by high computational and memory costs.

Target-aware NBV is substantially more challenging than whole-plant NBV due to target sparsity. Fruits, for instance, are sparsely visible, unevenly distributed, and often occluded, making them difficult to perceive. This sparsity also challenges training data collection, as most views are not or weakly informative, producing zero or very small IG for network training. To address this, we propose a novel IG metric de-

signed for sparse targets, meanwhile allowing automatic IG computation directly from robotic observations. Furthermore, to ensure efficient use of the collected data, we design a network architecture and loss function that enable efficient network training from limited, noisy, and weakly supervised data.

Finally, to facilitate real-world deployment, we adopt a simulation-to-real training paradigm, in which the network is first pretrained in simulation with synthetic plants and then fine-tuned in real-world conditions. The synthetic plants are generated using functional–structural plant (FSP) models (Hemmerling et al., 2008), which capture realistic 3D architectures of plants. This work focuses on self-supervised learning of NBV policies. However, the overall pipeline still relies on a target segmentation module, which may require manual annotation for training. Alternatively, pretrained foundation models, such as the Segment Anything Model 3 (SAM3) (Carion et al., 2025), enable zero-shot target segmentation via text queries, could reduce or eliminate manual labeling.

The main contributions of this paper are: (1) SSL-Semantic-NBV, a learning-based NBV framework that enables efficient 3D reconstruction of specific plant parts and whole plants, trained through robotic self-supervised learning; (2) enhanced training efficiency for learning-based NBV through a new IG metric, loss function, and network architecture; and (3) a pipeline for automated synthetic data generation to support the real-world deployment; (4) integration with zero-shot target segmentation for the entire pipeline self-supervised learning.

5.2 RELATED WORK

5.2.1 Active perception

Traditional NBV methods estimate IG by voxelizing the space and using ray-casting to count occupied voxels (Zaenker et al., 2021b). Although effective, ray-casting operations are computationally and memory intensive, limiting their planning speed. Learning-based NBV methods are significantly more efficient in IG prediction (Mendoza et al., 2020; Zeng et al., 2020). Furthermore, such methods, by using their ability to learn underlying plant features such as geometry and texture, could offer greater scalability and robustness. For effective execution of specific tasks, NBV planning should be target-aware, selecting views that prioritize specific plant parts. While target-aware NBV has been explored (Burusa et al., 2024b,c,a), most approaches still rely on ray-casting.

5.2.2 Self-supervised learning and IG annotation

Self-supervised learning (SSL) enables networks to learn from unlabeled data (Chen et al., 2020). In robotics, SSL entails robots interacting with the environment, collecting data, and annotating it automatically, thereby enabling continuous online learning without human intervention (Zeng et al., 2017b). The SSL-NBV framework (Ci et al., 2025) was introduced to enhance NBV methods in dynamic agricultural environments through online learning. A major challenge in SSL-based NBV is automatic IG annotation. Although SSL-NBV introduced an IG metric that estimates IG directly from the robot's own observations, this approach focuses on full-plant reconstruction and struggles with sparsely distributed targets. This work introduces a new IG metric for efficient and automatic IG computation for specific plant parts.

5.2.3 Synthetic data generation

Simulation-to-real training paradigms are widely adopted to improve real-world deployment efficiency of learning-based systems. In the NBV domain, Han et al. (2022) and Pan et al. (2022) trained NBV policies using hand-crafted 3D models from the ShapeNet dataset and applied the trained networks directly to real tasks, as these methods are limited in real-world fine-tuning. SSL-NBV addressed this by pre-training on simulated plants and fine-tuning to real scenarios using SSL. However, the simulated plants in their work were manually constructed and lacked realistic structural relationships between plant parts, which could contain crucial cues for target-aware NBV planning. Also, manual modeling is labor-intensive and not scalable. To address these issues, Rapado-Rincon et al. (2024) utilized GroIMP, a FSP modeling platform (Hemmerling et al., 2008), to generate morphologically realistic 3D plant models. Building on this, we developed a pipeline to automatically generate structurally accurate synthetic data for NBV pre-training.

5.3 PROBLEM FORMULATION

To obtain a 3D reconstruction of specific parts of the plant with as few views as possible, the objective of NBV planning is to select a minimal subset of views $V^{\text{select}} \subseteq V$ from the full set of candidate views $V = \{v_1, \ldots, v_M\}$, so as to maximize data collection of the targets, where $M \in \mathbb{N}$ is the total number of candidate views. Each v_m represents a candidate camera pose. Taking the current reconstruction as input, the neural network predicts the IG $\hat{G} = \{\hat{g}_1, \ldots, \hat{g}_M\}$ for all candidate views, and the next-best view $\hat{v}^*$ is selected as the one with the highest predicted IG. This process, including IG prediction and view selection, is referred to as an NBV iteration. A sequence of NBV iterations to complete a reconstruction is termed a reconstruction

cycle.

To achieve self-supervised learning, SSL-NBV continuously collects training data and updates the network during reconstruction cycles. During each NBV iteration, the ground truth IG $G = \{g_1, \ldots, g_M\}$ is calculated and used to train the network. In SSL-Semantic-NBV, G is calculated in relation to specific plant parts and directly from robotic observations.

5.4 PROPOSED METHOD

5.4.1 General pipeline

Fig. 5.1 provides an overview of the self-supervised learning process within a single NBV iteration. The framework builds on Ci et al. (2025), with modifications for target-awareness through an object segmentation module (Sec. 5.4.2), novel IG metric (Sec. 5.4.3), new loss function (Sec. 5.4.5), and network architecture (Sec. 5.4.4). Each iteration consists of eight steps, described in detail below.

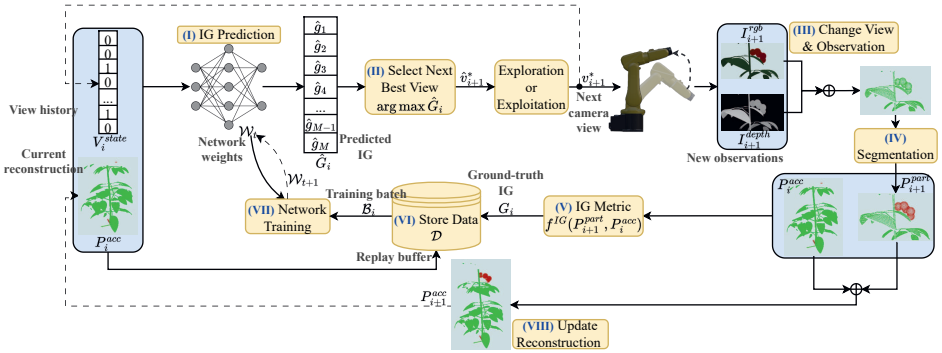


Figure 5.1: Overview of a single iteration in the self-supervised learning framework for semantic-aware NBV planning. Each iteration consists of eight steps.

Step I: At NBV iteration i, the neural network predicts $\hat{G}_i$ for all candidate views, based on the current reconstruction P_i^{acc} and the view history V_i^{state}. $P_i^{\text{acc}} = \{p_1, \ldots, p_k\}$ is an accumulated point cloud containing all points from previous views, with each point p_k having four channels: xyz for spatial position, and s for the semantic class label. For each view $v_m \in V$, its selection state is indicated by $v_m^{\text{state}} \in \{-1, 1\}$, where 1 denotes that the view has been selected and -1 otherwise. The selection states of all views are thus represented as $V_i^{\text{state}} = \{v_1^{\text{state}}, \ldots, v_M^{\text{state}}\}$.

Step (II): Based on $\hat{G}_i$, the next best view is determined as $\hat{v}_{i+1}^* = V(m^*)$, corresponding to the view with the highest predicted IG, so $m^* = \arg\max \hat{G}_i$. Instead of directly

moving the camera to $\hat{v}^*_{i+1}$, an exploration-exploitation mechanism is used to prevent the network from converging to sub-optimal view planning policies. Specifically, an exploration rate for reconstruction cycle c is calculated, representing the probability of selecting either $\hat{v}^*_{i+1}$ as the next view or a random view to explore the environment. Using this mechanism, the actual next camera view v^*_{i+1} is selected.

Step (III): From v^*_{i+1}, an RGB image I^{rgb}_{i+1} and a depth image I^{depth}_{i+1} are captured using an RGB-D camera. These images are fused to generate a partial point cloud.

Step (IV): This newly collected partial point cloud is then processed by a segmentation module to assign semantic labels to each point, producing P^{part}_{i+1}. Details about Point cloud segmentation are given in Sec. 5.4.2.

Step (V): The ground-truth IG $g_{v^*_{i+1}}$ for the selected view v^*_{i+1} is computed based on P^{acc}_i and P^{part}_{i+1} using the proposed IG metric (Sec. 5.4.3). This value is vectorized as $G_i = \{0, \ldots, g_{v^*_{i+1}}, \ldots, 0\}$ to serve as the supervision signal. Building on Ci et al. (2025), we construct supervision using only the ground truth IG of the selected view v^*_{i+1}, forming a weakly supervised learning scheme. This significantly reduces the need for robot motions to collect full IG annotations, improving the method's adaptability to new environments and tasks.

Step (VI): A training sample is constructed and stored in a replay buffer $\mathcal{D}$, using P^{acc}_i and V^{state}_i as inputs, with G_i as the supervision signal. $\mathcal{D}$ is a circular buffer storing the most recent L training samples. Once the buffer reaches its capacity, the oldest sample is replaced by the newest. This allows the network to learn from past experiences while adapting to new data.

Step (VII): A batch of training samples $\mathcal{B} \subseteq \mathcal{D}$ is randomly selected from $\mathcal{D}$ to train the network, with a constant batch size $l = 32$. This method, which first stores training samples in a buffer and then samples data from it for network training, is called experience replay (Mnih et al., 2013). Experience replay is essential for online learning, as it improves sample efficiency by reusing historical data. This step is repeated $q \in \mathbb{N}$ times, where q is a hyperparameter.

Step (VIII): The 3D reconstruction is updated as $P^{\text{acc}}_{i+1} = P^{\text{acc}}_i \cup \left(P^{\text{part}}_{i+1} \setminus \left(P^{\text{part}}_{i+1} \cap P^{\text{acc}}_i\right)\right)$, by adding new points from P^{part}_{i+1} to the current reconstruction P^{acc}_i while avoiding duplicates. The intersection $P^{part}_{i+1} \cap P^{acc}_i$ is computed similar as Sec. 5.4.3. The V^{state}_i is also updated by setting the element corresponding to v^*_{i+1} to 1.

5.4.2 Point cloud segmentation

Point cloud segmentation for plants has been widely studied (Golbach et al., 2016; van Marrewijk et al., 2025) and is therefore not the focus of this work. In simulation, segmentation is performed using point-wise ground-truth labels, which provide accurate segmentation and isolate the core challenges of target-aware NBV planning. In real-world experiments, SAM3 is employed to enable zero-shot target segmentation without requiring fine-tuning and human annotation and integrates seamlessly with the self-supervised learning pipeline. Instead of directly segmenting the point clouds, SAM3 operates on RGB images. Background pixels are first removed using depth information. The resulting 2D segmentation masks are then projected onto the corresponding 3D point clouds to obtain the 3D segmentation.

5.4.3 IG metric

We propose an IG metric $f^{\mathrm{IG}}(\cdot)$ that computes IG for specific plant parts directly from robotic observations, while accounting for target sparsity. The ground-truth $g_{v^*_{i+1}}$ for the selected view v^*_{i+1} is computed as,

$$g_{v^*_{i+1}} = w_1 \log \left|P^{\mathrm{Tnew}}_{i+1}\right| + w_2 \log \left|P^{\mathrm{new}}_{i+1}\right| \tag{5.1}$$

$$P^{\mathrm{new}}_{i+1} = P^{\mathrm{part}}_{i+1} \setminus \left(P^{\mathrm{part}}_{i+1} \cap P^{\mathrm{acc}}_{i}\right) \tag{5.2}$$

$$P^{\mathrm{Tnew}}_{i+1} = \left\{ p_k \mid p_k \in P^{\mathrm{new}}_{i+1} \wedge p^{(s)}_k = 1 \right\} \tag{5.3}$$

where, P^{new}_{i+1} represents the set of points in P^{part}_{i+1} not yet observed in P^{acc}_{i}. The semantic label of each point p_k is denoted as $p^{(s)}_k$, where $p^{(s)}_k = 1$ indicates a target point and 0 otherwise. P^{Tnew}_{i+1} includes only the target points from P^{new}_{i+1}. The notation $|\cdot|$ denotes the number of points in a point cloud.

The first term in Eq. 5.1 quantifies IG for the target, while the second term quantifies overall plant-level IG to encourage exploration when no target-relevant information has been collected. This is crucial in target-aware NBV planning where targets are sparse and unevenly distributed. The weighting factors w_1 and w_2 are set to 1 and 0.1, respectively, to balance these competing interests.

Since point coordinates in P^{part}_{i+1} and P^{acc}_{i} do not exactly match, the Euclidean distance between each point in P^{part}_{i+1} and its nearest neighbor in P^{acc}_{i} is computed. A point is considered new if the minimum distance exceeds a threshold σ; otherwise, it is treated as already reconstructed.

5.4.4 Network architecture

This work proposes a network architecture based on PC-NBV (Zeng et al., 2020) for IG prediction, with modifications to better extract features from one-hot view history V^{state}. The network takes the current reconstruction P^{acc} and V^{state} as inputs.

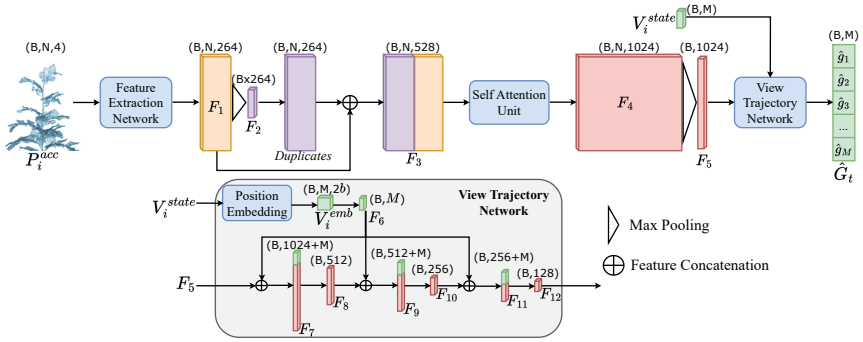


Figure 5.2: Network structure. The network takes the current reconstruction and view history as inputs and outputs the information gain for each candidate view. The View Trajectory Network is designed to extract information from the view history; details are provided in Sec. 5.4.4.

First, P^{acc} is processed by a PointNet-based network (Qi et al., 2017) to extract local features, producing F_1, which is then max-pooled to obtain a global feature F_2. F_2 is duplicated and concatenated with F_1 to form F_3, encoding both local and global point cloud features. F_3 is passed through a self-attention unit (Zhang et al., 2019a) to further fuse these features, resulting in F_4, which is max-pooled to produce F_5. To incorporate view history and avoid repetitive view selection, PC-NBV concatenates V^{state} with F_3 in a point-wise manner. Although effective with a small set of candidate views, preliminary tests show that this approach makes network challenging to learn view selection patterns for large view sets.

To solve this issue, we propose the View Trajectory Network (VTN). Each element $v_m^{\text{state}} \in V^{\text{state}}$ is embedded using a positional embedding function $f^{\text{pe}}(\cdot)$, producing the embedded view history V^{emb},

$$V^{\text{emb}} = \left\{ f^{\text{pe}}(v_m^{\text{state}}) \mid v_m^{\text{state}} \in V^{\text{state}},\ m = 1, \ldots, M \right\} \tag{5.4}$$

where

$$f^{\text{pe}}(v_m^{\text{state}}) = f^{\text{idx}}(\tilde{m}) \cdot v_m^{\text{state}}, \quad \tilde{m} = \frac{m}{M} \tag{5.5}$$

For each view v_m, its normalized index $\tilde{m}$ relative to the total number of candidate views M is computed. $\tilde{m}$ is then passed through a sinusoidal embedding function $f^{\text{idx}}(\cdot)$, following (Mildenhall et al., 2021), to encode it into a high-frequency representation,

$$f^{\text{idx}}(\tilde{m}) = \left[\sin(2^0\tilde{m}),\ \cos(2^0\tilde{m}),\ \ldots,\ \sin(2^{b-1}\tilde{m}),\ \cos(2^{b-1}\tilde{m})\right] \tag{5.6}$$

This embedding is then multiplied with v_m^{state}, injecting view selection state. This embedding injects both sequential and selection information into each view state, enhancing the model' s ability to capture patterns in the view history V^{state}.

V^{emb} is processed by a three-layer Multi-Layer Perceptron (MLP), yielding F_6. Finally, F_5 (point cloud feature) and F_6 (view history feature) are concatenated and passed through another three-layer MLP to predict $\hat{G}$. To retain view history information across layers, F_6 is concatenated with the output of each layer in the final MLP.

5.4.5 Loss function

To maximize training efficiency with weakly supervised IG annotations (ground-truth IG calculation detailed in Sec. 5.4.3), we propose a loss function $\mathcal{L}$ consisting of two components: (1) $\mathcal{L}^{\text{nbv}}$, which penalizes prediction errors for the selected next best view v^* of the current NBV iteration, and (2) $\mathcal{L}^{\text{visited}}$, which penalizes non-zero IG predictions for views that have been visited. $\mathcal{L}^{\text{nbv}}$ is computed same as Ci et al. (2025). For each current view, indices of past views in the reconstruction cycle, denoted as $M^{visited}$, are expected to have zero IG. So, in $\mathcal{L}^{\text{visited}}$, any non-zero prediction for these views is counted.

$$\mathcal{L} = \mathcal{L}^{\text{nbv}} + \mathcal{L}^{\text{visited}} \tag{5.7}$$

$$\mathcal{L}^{\text{nbv}} = \left(\hat{G}(v^*) - G(v^*)\right)^2 \tag{5.8}$$

$$\mathcal{L}^{\text{visited}} = \text{MSE}\Big(\hat{G}(M^{\text{visited}}), 0\Big)^2,$$

$$\text{with} \quad M^{\text{visited}} = \{m \mid v_m^{\text{state}} = 1,\ m \in \{1, 2, \ldots, M\}\} \tag{5.9}$$

This combined loss provides richer supervision signals to facilitate network training.

5.5 EXPERIMENTAL SETUP

Experiments were conducted in both simulated and real-world environments. The evaluation was performed from three perspectives, aligned with the contributions

stated in the introduction: (1) evaluating the 3D reconstruction efficiency of SSL-Semantic-NBV for specific plant parts and the whole plant compared to baseline methods; (2) conducting an ablation study to analyze the contributions of the proposed components to network training; and (3) evaluating the SSL-Semantic-NBV in real-world practice.

5.5.1 Simulation experiments

The simulation environment was built using Gazebo (Koenig & Howard, 2004), with data communication handled through the Robot Operating System (ROS) (Quigley et al., 2009). The experiments were run on a ThinkPad P15 laptop with Ubuntu 20.04, an Intel Xeon W-11855M processor, and an Nvidia GeForce RTX A5000 GPU (16 GB VRAM).

5.5.1.1 Digital Plant Creation

A GroIMP-based pipeline was developed to generate 3D digital tomato plants using measured traits as inputs, producing 3D meshes, point clouds, and phenotypic data. Ten tomato plants (*Solanum lycopersicum* L.), cultivar 'Merlice', were manually measured in the greenhouse in weeks 9 to 11 after sowing. Traits including internode length and diameter, leaf length, width, inclination angle, and phyllotaxis angle, truss length and angle, number of fruits and flowers per truss were recorded. Selected traits, e.g. traits related to leaf inclination and phyllotactic angles, were recorded only for a sample of plants, because of time constraints, and estimated values based on these measurements were used for the remaining plants. During plant creation, uncertainties were added to the measured values, based on expert knowledge, to introduce morphological variation.

Two sets of digital plants were generated for simulation experiments: 10 normal plants (examples shown in Fig. 5.3a), each corresponding to a real plant, and 10 plants with heavy occlusions derived from the normal set (examples shown in Fig. 5.3b). The occluded plants were created by doubling the leaf width and reducing the angles between stems and leaves, causing the leaves to wrap tightly around the stem.

For each set, seven plants were used for training and the remaining three for testing. In each 3D reconstruction cycle, a plant was randomly selected from the set and reconstructed for specific plant parts from a random initial view. To introduce uncertainties resembling real practice, the selected simulated plant was randomly translated along the x, y, and z axes within ranges of $[-0.2\text{m}, +0.2\text{m}]$, $[-0.2\text{m}, +0.2\text{m}]$, and $[0, +0.2\text{m}]$, respectively, and rotated within $[0°, 360°]$ around the vertical axis (Fig. 5.3c). A planar view space with 9×9 views over a $1.2\text{m} \times 1.2\text{m}$ area at 0.6m distance was used for data capture.

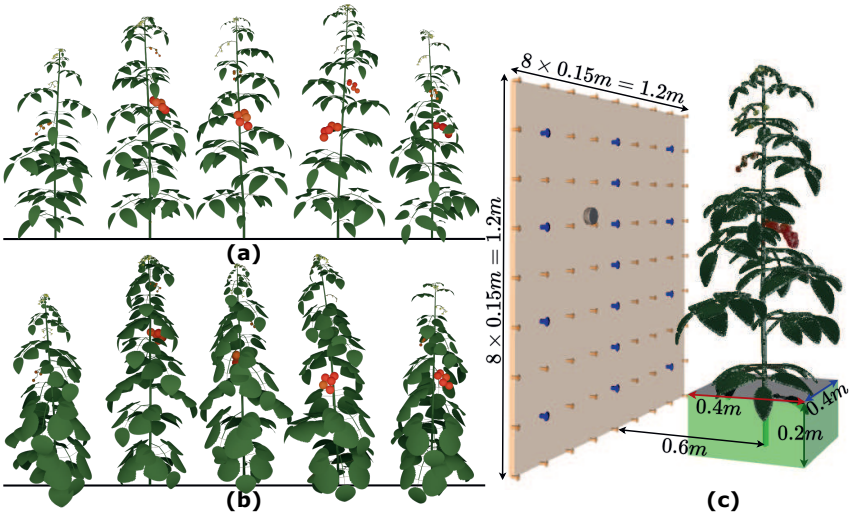


Figure 5.3: Examples of digital plants used in simulation experiments. (a) Simulated plants with normal occlusion. (b) Simulated plants with heavy occlusion of plant parts. (c) View sampling space and plant position uncertainties in the simulation experiments.

5.5.1.2 Data Collection

A simulated Intel RealSense L515 camera was used to capture RGB and depth images at a resolution of 540 × 960. These images were fused to generate point clouds, which were then downsampled using a VoxelGrid filter to maintain a resolution of σ=0.003m. In simulation, the ground-truth point clouds were generated by uniformly sampling points on the mesh surface and then downsampled to a resolution of 0.003m.

5.5.1.3 Parameter settings for self-supervised learning

For each plant set, a total of 2,000 reconstruction cycles were performed. The training frequency was set to $q = 1$, allowing one training iteration per NBV iteration. The experience buffer size L was set to 7,500. The same parameters were used for SSL-NBV, except that the number of reconstruction cycles was extended to 10,000 due to lower training efficiency. The maximum number of views allowed in a reconstruction cycle was fixed at 15.

5.5.2 Real-world experiments

Ten real-world plants collected by Burusa et al. (2024c) were used for real-world experiments. Data collection was performed using an Intel RealSense L515 camera

mounted on the end-effector of a 6-DoF ABB IRB1200 robotic arm. Each plant was observed from 150 positions (Fig. 5.6). At each position, five pan-tilt angles $[(-15°, 0°),\ (15°, 0°),\ (0°, 0°),\ (0°, 15°),\ (0°, -15°)]$ were defined, resulting in a total of 750 views and RGB-D images (960×540 for both RGB and Depth) per plant. For NBV planning, only the 150 views perpendicular to the plane were used, while all 750 views were used to generate the ground-truth using COLMAP, with the RGB images and corresponding camera poses as inputs. The resulting ground truth was then downsampled to a voxel size of σ=0.006m, which was twice that used in simulation to better filter noise. Three plants were randomly selected for testing and seven for training. At the beginning of each reconstruction cycle, a random view was selected as the starting point for NBV. The network, pre-trained on normal simulated plants, was further fine-tuned on real plants using the same parameter settings as in the simulation.

5.5.3 Baseline methods

The proposed SSL-Semantic-NBV was evaluated against the following baseline methods:

1. SSL-NBV: A self-supervised learning method (Ci et al., 2025) similar to SSL-Semantic-NBV but without target-awareness. It treats the entire plant as the target.

2. Voxel-NBV: A voxel-based NBV method (Burusa et al., 2024c) using ray-casting for IG prediction. It performs target-aware NBV by localizing the main stem and defining a region of interest (ROI) around it to guide exploration.

3. Random: A method that randomly selects the next view from the full view space at each NBV iteration.

4. Pre-defined: A method selects views from a predefined set of candidate views arranged around the central vertical axis of the view space (Fig. 5.3c). Since reconstruction efficiency depends on the visiting order, the view sequence was randomized in each cycle to ensure generalizable results.

5.5.4 Evaluation metrics

Two commonly used metrics, as in Ci et al. (2025) and Zeng et al. (2020), were applied to evaluate reconstruction quality and efficiency with respect to the number of views used in a reconstruction cycle:

1. Reconstruction Ratio (R): This metric measures the completeness of the current reconstruction by comparing P^{acc} and the ground-truth point cloud P^o,

computed as $\frac{|P^o \cap P^{\text{acc}}|}{|P^o|}$. The same method described in Sec. 5.4.3 was used to identify the reconstructed points in P^o. In simulation, P^o was generated by uniformly sampling points from the surface of digital plant meshes. In the real world, P^o was reconstructed using COLMAP (Schonberger & Frahm, 2016), a structure-from-motion tool that takes RGB images and corresponding camera poses from all views.

2. Area Under View-Reconstruction Curve (AUC@VR): Metric AUC@VR calculates the area under the R curve with respect to the number of views used, reflecting the overall reconstruction efficiency across the full view trajectory.

5.6 RESULTS

5.6.1 Simulation results

5.6.1.1 Comparison With Baseline Methods

As shown in Fig. 5.4, SSL-Semantic-NBV achieved the highest 3D reconstruction efficiency across all targets of plant, node, and fruit, compared to baseline methods. The gap in R between SSL-Semantic-NBV and other NBV methods was smaller for plant and node. This is because plants and nodes are more evenly distributed in the view space, making them easier to reconstruct and narrowing the performance gap. In contrast, SSL-Semantic-NBV was significantly more efficient for fruit reconstruction, reaching $R \geq 0.4$ with only five views, 10 fewer than voxel-NBV, and this level of R was not reached by SSL-NBV and non-NBV methods. This difference arises because fruits are sparsely and unevenly distributed, which poses greater challenges and requires target-aware planning. These results highlight the effectiveness of SSL-Semantic-NBV in target-aware NBV tasks, particularly for targets that are sparsely and unevenly distributed. No method achieved $R > 0.9$, primarily because the planar view sampling strategy limited the ability to capture the plant' s backside. Nevertheless, SSL-Semantic-NBV reached R values close to the upper-bound baselines.

Table 5.1 presents AUC@VR for all methods on both normal and heavily occluded plants, using a single metric to evaluate reconstruction efficiency over the entire view trajectory. On normal plants, SSL-Semantic-NBV improved average AUC@VR across targets by $0.15 - 0.25$ over non-NBV methods and by $0.04 - 0.06$ over existing NBV methods, corresponding to relative gains of $47\% - 114\%$ and $9\% - 15\%$, respectively. On heavily occluded plants, the comparative trends were similar. SSL-Semantic-NBV consistently achieved the highest AUC@VR across the different plant parts, demonstrating strong tolerance to occlusion.

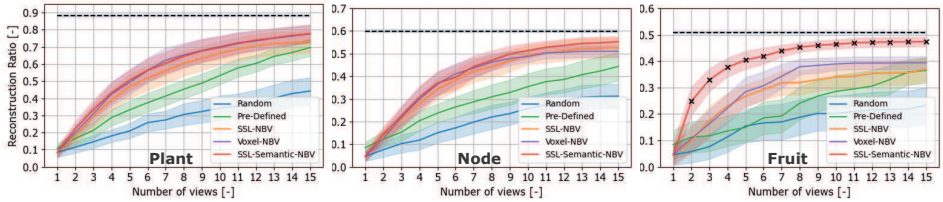


Figure 5.4: Reconstruction ratio for different methods across Plant, Fruit, and Node classes. Dashed lines represent the maximum achievable results by fusing point cloud from all views. Solid lines represent the average and shaded regions indicate the 95% confidence interval over 25 repetitions, each with a random starting view, plant model and pose.

Table 5.1: AUC@VR of various methods on normal (Norm) and heavily occluded (Heavy) simulated plants. Bold values indicate the best results. Inference speed denotes the time required for one NBV iteration (excluding robot motion time).

	Plant		Node		Fruit		Average		Speed
Method	Norm	Heav	Norm	Heav	Norm	Heav	Norm	Heav	
Random	0.28	0.25	0.21	0.13	0.17	0.10	0.22	0.16	–
Pre-Defined	0.44	0.35	0.29	0.20	0.23	0.17	0.32	0.24	–
SSL-NBV	0.54	0.45	0.39	0.28	0.28	0.23	0.41	0.32	**0.0047s**
Voxel-NBV	**0.58**	0.48	0.40	0.29	0.31	0.26	0.43	0.34	8.83s
SSL-Semantic-NBV	**0.58**	**0.51**	**0.42**	**0.33**	**0.41**	**0.36**	**0.47**	**0.40**	0.0072s

5.6.1.2 Ablation Test

This experiment evaluated the contributions of the proposed components of the VTN, loss function $\mathcal{L}$, and IG metric $f^{\mathrm{IG}}(\cdot)$, to network training and NBV planning. According to Fig. 5.5, SSL-Semantic-NBV achieved the fastest convergence and the highest final AUC@VR, indicating that all components contribute to overall performance. Removing $f^{\mathrm{IG}}(\cdot)$ caused significantly slower convergence and a lower final AUC@VR, highlighting its critical contribution. Excluding $\mathcal{L}$ also slowed convergence across targets, but with sufficient training cycles the method reached a final AUC@VR comparable to SSL-Semantic-NBV. A similar effect was observed when the VTN was removed, with the additional impact of lowering the final AUC@VR for plant reconstruction. Although a total of 2000 training cycles were performed, SSL-Semantic-NBV converged within the first ∼600 cycles, suggesting that training cycles could be substantially reduced in practical deployments with minimal performance loss.

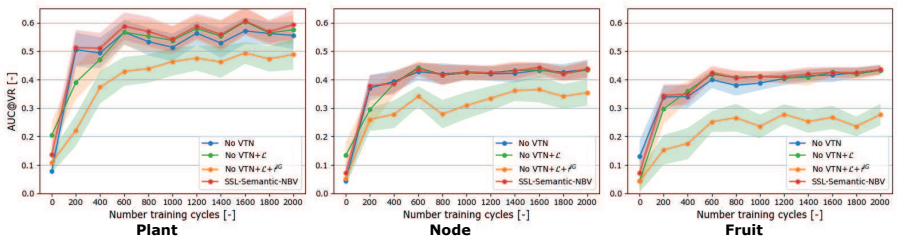


Figure 5.5: Ablation test about the contributions of the proposed VTN, loss function, and IG metric. Solid lines represent the average and shaded areas indicate the 95% confidence interval over 25 repetitions, each with a random starting view, plant model and pose. Experiments are conducted in simulation, with each reconstruction cycle consisting of 15 viewpoints. When a module is disabled, the corresponding component from SSL-NBV is used instead.

5.6.2 Real-World results

As shown in Table 5.2, SSL-Semantic-NBV achieved the highest average AUC@VR across different plant parts, with improvements of 0.03–0.05 over NBV methods and 0.06 – 0.09 over non-NBV methods, corresponding to relative improvements of 3% – 7% and 8% – 15%, respectively. While NBV methods yielded similar AUC@VR values for plants and nodes, SSL-Semantic-NBV achieved an AUC@VR of 0.66 for fruit reconstruction, which was significantly higher than other NBV methods ($(p < 0.001)$ compared to SSL-NBV). This improvement is attributed to its target-aware mechanism together with neural networks that learn plant structural features and prioritize views of the top regions (Fig. 5.6). Real-world experiments show higher AUC@VR for each method compared to simulation. This is because COLMAP generates ground-truth point clouds from RGB images captured in the view space, representing only the visible parts of the plants. SAM3 is integrated to enable zero-shot target segmentation, allowing seamless incorporation into the self-supervised learning pipeline. As shown in Fig. 5.7, the method using SAM3-based segmentation successfully learned effective NBV policies for fruit reconstruction, prioritizing relevant regions similarly to the ground-truth segmentation setting, demonstrating the applicability of SSL-Semantic-NBV in real-world self-supervised scenarios. The lower AUC@VR observed with SAM3-based segmentation is likely caused by segmentation and pixel-to-point projection errors rather than the NBV strategy itself. It should be noted that while SAM3 provides accurate segmentation for fruits, targets with more complex structures, such as nodes, may require task-specific fine-tuning.

Table 5.2: AUC@VR of methods for real-world plants. Bold values indicate the highest results. Target segmentation is based on ground-truth labels.

Methods	Plant ($\mu \pm \text{CI}_{95}$)	Node ($\mu \pm \text{CI}_{95}$)	Fruit ($\mu \pm \text{CI}_{95}$)	Average
Random	0.58±0.014	0.62±0.030	0.49±0.043	0.56
Pre-Defined	0.62±0.013	0.67±0.022	0.47±0.039	0.59
SSL-NBV	0.62±0.012	**0.67±0.019**	0.56±0.036	0.62
Voxel-NBV	0.62±0.013	0.67±0.023	0.51±0.037	0.60
SSL-Semantic-NBV	**0.62±0.0086**	0.66±0.014	**0.66±0.030**	**0.65**

5.7 Discussion and Conclusion

This paper presents SSL-Semantic-NBV for efficient 3D reconstruction of targeted plant parts and whole plants, trained through robotic self-supervised learning. Both simulation and real-world experiments demonstrate that SSL-Semantic-NBV achieves the highest reconstruction efficiency across targets. For plant and node reconstruction, the improvements over other NBV methods are relatively small, as these targets are evenly distributed in the view space, making them easier to observe and reducing the performance gap. In contrast, SSL-Semantic-NBV achieves significant improvements in fruit reconstruction because fruits are sparsely and unevenly distributed, and the target-aware mechanism enables SSL-Semantic-NBV to better handle such situations. A similar pattern to the simulation results was observed in real-world experiments. In the real-world setup, SAM3 is employed for zero-shot target segmentation. However, for more complex targets, a fine-tuned segmentation network and limited human annotations may still be required.

Based on these findings, we draw three conclusions: First, SSL-Semantic-NBV achieves the highest reconstruction efficiency among the compared methods, particularly for those that are sparsely and unevenly distributed, due to its target-aware mechanism. Second, the proposed components, including the loss function, VTN, and especially the IG metric, enhance both online learning efficiency and reconstruction efficiency. Third, SSL-Semantic-NBV is feasible for real-world deployment in target-aware NBV tasks. Future work will extend the evaluation to a wider range of plants, targets, and application scenarios.

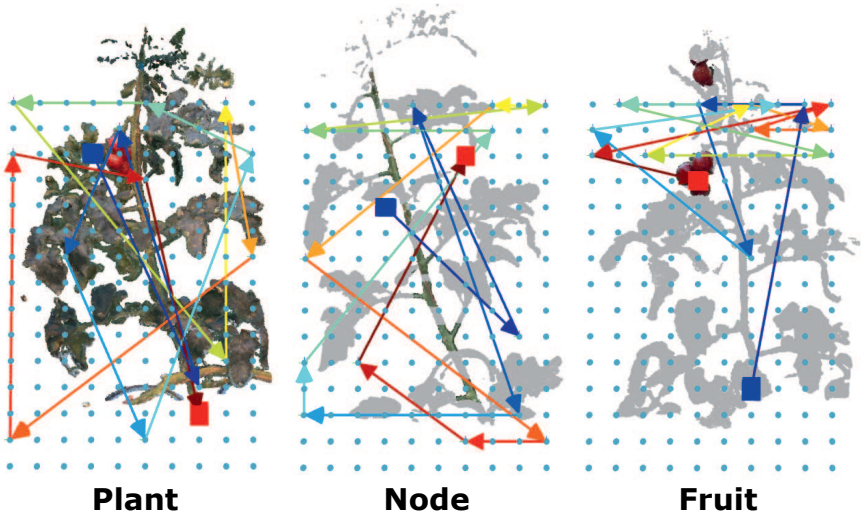


Figure 5.6: Qualitative analysis of view trajectories for different targets. Gray points represent the ground-truth point cloud, and colored points indicate target regions. Views are widely distributed for entire plant and node targets, while concentrating on upper regions when targeting fruits.

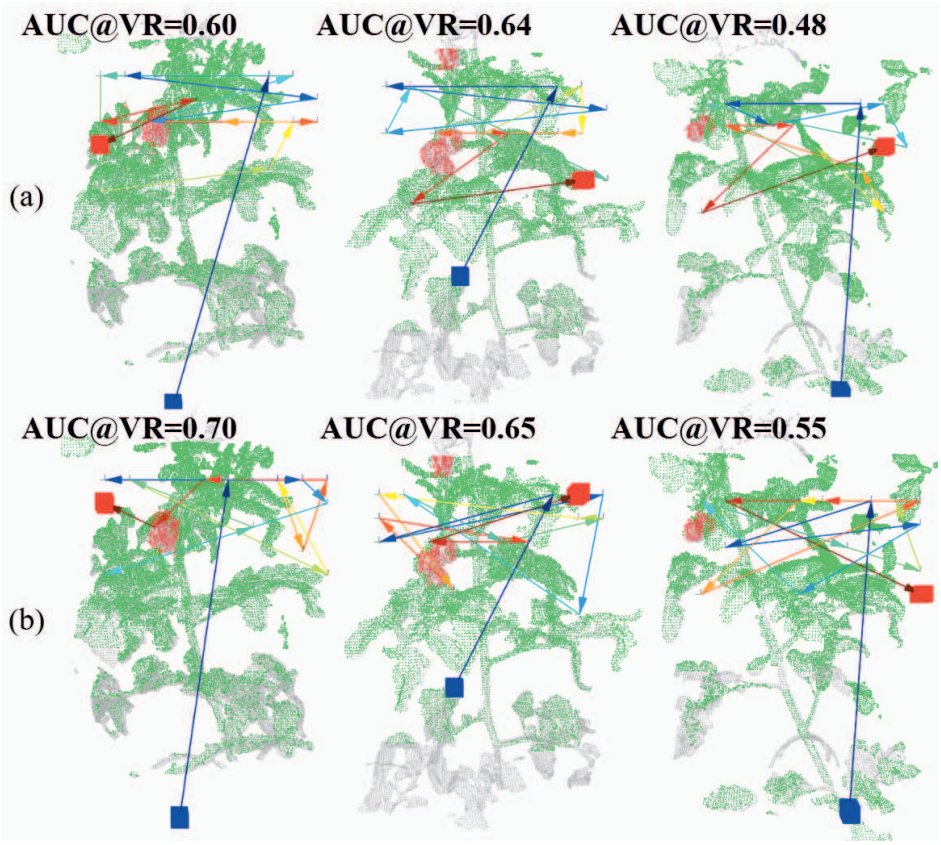


Figure 5.7: Comparison of reconstruction results using SAM3-based segmentation (a) and ground-truth bounding boxes (b). Gray points denote the complete point cloud. Green points indicate reconstructed points, and red points represent the target region.

Chapter 6

General Discussion

6.1 Conclusions and reflections on the objective and hypothesis

The main objective of this PhD thesis was to advance perception systems in agricultural robotics by developing implicit NBV methods that better handle occlusions in CEA environments and enable online self-supervised adaptation to novel environments. Designing such implicit NBV systems for CEA is challenging because they must remain adaptable, feasible to train under real operating conditions, scalable across different plant sizes and targets, and efficient, as outlined in Sec. 1.4.

The central hypothesis was that implicit NBV methods equipped with online self-supervised adaptation, enabling their viewpoint selection strategies to adjust to novel scenarios, can achieve more efficient 3D plant reconstruction under occlusion in novel environments than offline-trained implicit NBV methods.

To test this hypothesis, the thesis introduced a series of methods across four research chapters. Chapter 2 provides an important conceptual foundation by demonstrating that the accuracy of peduncle node pose estimation strongly depends on viewpoint selection, motivating the need for multi-view and active perception strategies.

Building on this insight, Chapters 3–5 progressively developed implicit NBV-based active perception methods, with each chapter targeting specific requirements identified in Chapter 1 related to adaptability, feasibility, scalability, and efficiency. Chapter 3 introduces SSL-Global-NBV, the first method proposed in this thesis to integrate self-supervised learning into the NBV pipeline. The method autonomously collects training data during NBV execution and continuously updates its NBV policy using newly acquired data. In 3D reconstruction tasks, SSL-Global-NBV improved reconstruction efficiency under occlusion in previously unseen environments after online self-supervised adaptation compared with offline-trained implicit NBV methods. This directly supports the central hypothesis. Furthermore, the proposed IG metric, self-supervised learning, and weakly-supervised learning methods improve adaptability and real-world feasibility of implicit NBV methods by eliminating the need for complete object models, offline training data collection, and extensive IG annotations, respectively.

Chapter 4 extended the framework with SSL-Local-NBV to improve scalability and reconstruction efficiency. Instead of selecting viewpoints from a predefined global set, SSL-Local-NBV determines the next viewpoint through incremental movements relative to the current camera pose. To support local viewpoint planning, a View Trajectory Network (VTN) was proposed to encode trajectory information and prevent redundant revisiting of previously observed viewpoints. This design enables the sys-

tem to handle objects of varying sizes without being constrained by a fixed candidate set, while significantly increasing reconstruction efficiency per unit of camera motion. The results demonstrate the substantial benefits of reducing camera trajectory length and robotic motion through local viewpoint planning in NBV tasks.

Chapter 5 developed SSL-Semantic-NBV, which enhances reconstruction efficiency and completeness across different perception targets, including fruits, nodes, and whole plants, through a target-aware mechanism. The work refined the IG metric to address target sparsity and integrated it with the network' s ability to implicitly encode latent representations of plant characteristics. As a result, the method is particularly effective for sparsely and unevenly distributed targets, such as fruits, by prioritizing informative regions during exploration. These findings demonstrate the advantages of incorporating target-aware mechanisms into implicit NBV, especially for tasks involving unevenly distributed perception targets.

Collectively, the proposed methods advance implicit NBV by improving adaptability to novel environments, scalability across different scene sizes, feasibility for real-world deployment, and efficiency through reduced robotic motion, fewer IG annotations, and more targeted 3D reconstruction. Together, these contributions demonstrate that the thesis objective has been achieved. A detailed discussion of how the four requirements are addressed is provided below.

Adaptability requires implicit NBV methods to continuously adapt to new environments and plant types. In this thesis, this is achieved by enabling the system to simultaneously collect training data during NBV execution and update its NBV policy using newly collected data from novel environments. To address this requirement, Chapter 3 introduced SSL-Global-NBV, an active perception framework that updates its viewpoint planning strategy through self-supervised learning. When evaluated on 3D plant reconstruction tasks, SSL-Global-NBV continuously adapted from tomato to pepper plants and achieved higher reconstruction efficiency and quality on pepper plants compared to a network without online adaptation. It reached a 90% reconstruction ratio using two fewer views than the non adapted network (Fig. 3.10). Figure 3.11 further illustrates the improved reconstruction performance achieved through online adaptation on novel plants. These results demonstrate that online adaptation allows implicit NBV methods to handle previously unseen environments and targets more effectively after a period of self-supervised learning, directly supporting the central hypothesis.

Chapter 4 presented SSL-Local-NBV, which incorporated self-supervised learning into local viewpoint planning, while Chapter 5 introduced SSL-Semantic-NBV, enabling the network to continuously optimize its NBV strategy while targeting specific plant

parts. In terms of adaptability, explicit NBV methods can be applied to different setups without requiring network training. However, their performance depends on a predefined search space, which can limit their adaptability to different targets and scene configurations.

Collectively, these developments enhanced the adaptability of implicit NBV methods at the global, local, and semantic planning levels. It is therefore concluded that online adaptation improves 3D reconstruction performance in novel plants and environments after a period of self-supervised learning, demonstrating a clear advantage over offline implicit NBV approaches.

Real-world feasibility concerns whether an implicit NBV network can be trained with data that are obtainable under actual operating conditions. Existing implicit NBV methods are generally trained by taking the current 3D reconstruction and past camera viewpoints as inputs, with IG for candidate viewpoints as the target signal. While the input data are easy to record, accurate ground-truth IG values are more difficult to obtain. For example, PC-NBV (Zeng et al., 2020) and DB-NBV (Han et al., 2022) compute ground-truth IG by comparing the reconstructions with the complete digital model of the target. However, such complete digital models are not available for real plants, limiting the applicability of these methods in practical agricultural settings. This thesis addressed this limitation by proposing IG metrics that can be computed directly from the robot' s own observations. In Chapter 3, ground-truth IG was defined as the fraction of newly observed points relative to all points captured from a viewpoint. Combined with the robot' s ability to explore the scene, this enabled self-supervised training for full-plant reconstruction. Chapter 5 further refined the IG metric by emphasizing information related to specific plant parts, enabling real-world target-aware NBV planning and self-supervised learning for semantic reconstruction tasks. As a result, the proposed implicit NBV methods can be trained under realistic operating conditions. Real-world feasibility also requires the method to handle sensor noise encountered in practical conditions. In simulation experiments with manually added noise, the proposed method demonstrated greater robustness (Fig. 4.6). Furthermore, real-world experiments conducted in Chapters 3, 4, and 5 confirmed the feasibility of the proposed methods under practical operating conditions.

Scalability refers to the ability of a method to operate across different scene sizes and targets without changing the network architecture or parameters. In the context of 3D pose estimation of plant parts, Chapter 2 contributes to scalability by combining deep keypoint detection with 3D point cloud information to automatically localize relevant landmarks on plants. This approach improves accuracy and robustness under variations in plant structure and environmental conditions and has the potential to

be extended to other targets, such as petiole nodes, tomato trusses, and different crop types, without fundamental changes to the method.

For NBV-based 3D reconstruction, global implicit NBV methods such as SSL-Global-NBV rely on a predefined set of candidate viewpoints. This design limits scalability, as plants and environments of different sizes require different numbers and distributions of viewpoints to achieve full coverage. Chapter 4 addressed this limitation by introducing SSL-Local-NBV, which determines the next camera pose through small position and orientation adjustments relative to the current viewpoint. This removes the dependence on a fixed candidate set and makes the method inherently more scalable. Results in Chapter 4 show that SSL-Local-NBV adapts effectively to plants of different sizes without system modification. As shown in Table. 4.5, SSL-Local-NBV improved average reconstruction efficiency by 5.2% across varying plant sizes compared with SSL-Global-NBV, due to more flexible viewpoint selection. In contrast, SSL-Global-NBV remains constrained by its fixed candidate set, limiting its scalability to different plant sizes.

Scalability also relates to the ability to focus on different perception targets. Chapter 5 addressed this by developing SSL-Semantic-NBV, which plans viewpoints for different plant parts (e.g., fruits, nodes, or whole plants). The introduced target-aware mechanism improved reconstruction efficiency compared with non-NBV methods and state-of-the-art NBV approaches across different targets, enabling scalable perception for diverse robotic tasks.

Efficiency in implicit NBV encompasses viewpoint efficiency, trajectory efficiency, and neural network training efficiency. During NBV planning, the system must select informative viewpoints (viewpoint efficiency) while generating short and smooth camera trajectories (trajectory efficiency) that minimize unnecessary motion. Balancing these objectives requires trajectory-level optimization, which remains challenging. SSL-Local-NBV partially addressed this challenge by using local viewpoint planning, which shortens camera paths and substantially reduces camera movement while maintaining comparable viewpoint efficiency. As reported in Chapter 4, SSL-Local-NBV reduces camera movement by 56 – 70% relative to global NBV methods (Tables 4.2, 4.3, and 4.6), resulting in a 267 – 300% improvement in trajectory efficiency. When extended to target-aware tasks, SSL-Semantic-NBV further improved viewpoint efficiency by prioritizing regions that are most informative for specific plant parts, It achieves 42 – 104% higher reconstruction efficiency than non-NBV methods and 9 – 18% improvements over state-of-the-art NBV approaches. Implicit NBV offers significant advantages over explicit NBV in terms of efficiency. As shown in Table 3.1, implicit NBV achieves more than an 800-fold increase in inference speed compared with explicit NBV, and this gap widens as the number of candidate viewpoints in-

creases.

Efficiency also concerns how training data are collected and used. Chapter 3 introduced a weakly supervised learning paradigm that enabled training with sparsely labeled IG data. SSL-Global-NBV reduced the number of required IG labels by more than 90% compared to PC-NBV (Zeng et al., 2020), which relies on complete IG supervision. Because each IG label requires physically moving the robot to a specific viewpoint for measurement, this reduction significantly decreased the time needed for training data collection. Chapter 5 further refined the IG metric and training loss function, strengthening the supervision signal and reducing the required training iterations by approximately 80% relative to SSL-Global-NBV while achieving improved reconstruction performance.

6.2 Research limitations and recommendations

Despite the advances achieved, this thesis has several limitations. This section outlines the primary constraints of the research and identifies directions for future work.

This thesis conducted most evaluations in simulation, focused on specific 3D reconstruction tasks, and examined the perception module in isolation from other robotic subsystems. Accordingly, the following discussion addresses limitations related to environmental and crop variability, sensor-related challenges, robotic motion uncertainty, and the integration of perception with navigation and manipulation.

6.2.1 Environment and crop variations

CEA systems exhibit considerable variation in both plant appearance and environmental conditions. Methods developed for these settings must therefore be robust to such variation. Each chapter in this thesis considered variability in experimental design and evaluation, but it was clearly not possible to cover the full range of conditions encountered in practice. A detailed discussion on this limitation is provided per chapter below.

In Chapter 2, which focused on 3D pose estimation of tomato peduncle nodes, images were collected from plants grown in a commercial greenhouse. These plants exhibited variations in geometry and texture, and performance was evaluated from five camera viewpoints per node, introducing variations related to viewpoint changes. In Chapter 3, which addressed 3D plant reconstruction with implicit NBV, the method was tested on multiple targets: simulated tomato plants, simulated sweet pepper plants, and real tomato plants. For each reconstruction, the plant' s position and orientation were randomized so that the NBV algorithm had to adapt its viewpoint planning to the

observed scene. The same strategy for introducing variation was applied in Chapters 4 and 5.

Despite these efforts, it is important to acknowledge that real agricultural tasks involve a much broader range of variability than what was captured in this thesis. In Chapter 2, all tomato plants belonged to the same cultivar and were grown under controlled conditions with consistent pruning and maintenance. Variability arising from different species, cultivation practices, or less controlled environments was not represented. Such factors could influence the results, for example by altering the optimal camera viewpoints, affecting pose estimation accuracy, or necessitating additional network fine-tuning using domain-specific training data. However, the specific impact of these factors is difficult to predict in advance and can only be fully assessed through scenario-specific evaluations.

In Chapters 3-5, most experiments were conducted in simulation. Simulation enabled efficient, controlled, and repeatable testing with access to perfect ground-truth data, but simulated plants cannot fully replicate the complexity of real plants, including natural occlusions, motion, and morphological variability. Although real-plant experiments were also conducted, the datasets were limited in scale and diversity due to the challenges of capturing plants from many viewpoints and obtaining complete ground-truth 3D models. For instance, Chapter 3 evaluated SSL-Global-NBV on only three tomato plants of similar age and height. Chapter 4 expanded the real-plant dataset to 17 plants collected at two time points, covering different growth stages and maintenance practices. These plants were older and more complex than those in Chapter 3. A subset of 10 plants was later used in Chapter 5. Nonetheless, the dataset remained relatively small and did not include a wide range of growth stages or plant species.

The level of plant occlusion significantly influences perception performance. Higher occlusion reduces visible plant coverage and may increase the number of viewpoints required. Similarly, a larger camera–plant distance increases the observable area per view, which can improve reconstruction efficiency. Plant motion remains an open challenge for current perception methods, particularly for tasks involving physical interaction with plants, such as harvesting or deleafing. This thesis assumes static scenes and struggle to reflect plant dynamics. For applications such as 3D reconstruction or phenotyping, however, the impact of plant motion can be mitigated if data collection is performed under relatively controlled environmental conditions. The experiments in this thesis were also restricted to single-plant reconstruction and did not consider multi-plant scenes. Introducing multiple plants would substantially increase the difficulty of reconstruction. For global NBV, larger scenes require a much larger set of candidate viewpoints, increasing training time and difficulties to select

the optimal viewpoint. For SSL-Local-NBV, larger and more complex scenes may increase the risk of being stuck into local optima, reducing efficiency or even causing reconstruction failure. Works targeting efficient reconstruction of large-scale scenes or multiple targets include Zeng et al. (2022) and Vashisth et al. (2024), both of which employ deep reinforcement learning to enable long-term planning. However, these methods are evaluated only in simulation, and their feasibility in real-world settings remains to be validated. Addressing these challenges remains an important direction for future work.

These untested environmental and crop variations imply that absolute performance metrics, such as detection accuracy in Chapter 2 and reconstruction completeness in Chapters 3–5, should be interpreted in the context of the specific conditions under which the experiments were conducted. Performance in other plant types or environmental settings can only be confirmed through scenario-specific testing. However, the thesis still provides valuable insight through consistent comparative performance trends between the baseline methods. For instance, in Chapter 4, absolute reconstruction completeness differed between simulated and real plants, but the relative trends in reconstruction quality and trajectory efficiency between SSL-Local-NBV and baseline methods remained consistent.This consistency supports the robustness of the proposed methods relative to baseline approaches under the evaluated metrics.

In summary, it is inherently challenging to evaluate the proposed methods across all possible variations in plant characteristics and environmental conditions encountered in real practice. Rather than relying on absolute metric values, the performance of a method is more reliably assessed through its comparative performance patterns relative to baseline approaches.

6.2.2 Camera noise

Camera noise is a common challenge for perception systems in real-world applications and is particularly problematic in CEA environments. RGB cameras can produce inconsistent colors under changing illumination and struggle in low-light conditions. Depth and LiDAR sensors are reflective to plant surfaces, weak texture, shadows, dust, water droplets, and humidity, which can lead to noisy or inaccurate depth estimates (Kootstra, 2024). Such noise propagates through the perception pipeline, causing incorrect detections, inaccurate 3D localization and pose estimation, and poor 3D reconstructions, ultimately affecting robotic task execution. Addressing sensor noise is therefore essential for reliable robotic operation in CEA environments.

Across this thesis, specific noise-handling procedures were integrated into the perception pipeline. In Chapter 2, a deep learning–based method was used to detect 2D

landmarks representing the structure of peduncle nodes. By extracting meaningful features directly from images, the network inherently provides some resilience against noisy or imperfect inputs. During the conversion of these 2D landmarks into 3D, the aligned point cloud was used for spatial information. To address its sparsity and noise, a three-step filtering process, consisting of 2D keypoint extension, 3D point clustering, and outlier removal, was applied, improving the reliability of the final 3D landmark estimates.

In Chapter 3, noise in the point cloud lowered reconstruction quality and introduced errors in ground-truth IG calculation. Noisy points could be mistakenly classified as new observations, inflating IG values while not contributing to reconstruction completeness. To address this, a three-step noise reduction process was applied to each newly collected point cloud and iteratively to the accumulated reconstruction. The procedure removed background points, eliminated outliers, and filtered redundant points. The resulting reconstructions had cleaner surfaces and better structural detail. The same process was used in Chapters 4 and 5.

While these algorithmic strategies reduced noise and improved performance across multiple tasks, they cannot fully eliminate sensor-related limitations. For example, in Chapter 2, the sparse and noisy depth measurements remained a major bottleneck for accurate 3D pose estimation. Similarly, in the real-world 3D reconstruction tasks in Chapters 3, 4, and 5, although background clutter and outliers were largely removed, residual noise persisted on fine structures such as leaves, stems, and nodes, limiting reconstruction quality in regions needing details. This noise may also lead to an underestimation of reconstruction performance, as noise is present in the ground-truth point cloud and is difficult to accurately reproduce in the reconstructed model. These observations suggest that relying solely on post-processing of noisy sensor data may be insufficient, indicating the need for further improvements in sensing technology, including sensing hardware, data processing algorithms, sensor fusion, and data representation.

From a hardware perspective, Zhang et al. (2023b) analyzes the impact of adverse environmental conditions, such as illumination changes, fog, and rain, on RGB and LiDAR sensors used in autonomous driving, offering insights that are also relevant to CEA applications. For RGB sensing, techniques such as high Dynamic Range (HDR) imaging and event-based cameras (e.g., Dynamic Vision Sensors) can improve tolerance to strong illumination contrasts and motion blur (Battiato et al., 2003; Gallego et al., 2020). For depth sensing, LiDAR and stereo RGB-D cameras remain the most commonly used solutions in agricultural robotics due to their affordability and practical availability. However, in greenhouse environments these sensors often produce sparse and noisy point clouds as a result of humidity, water on plant surfaces, and

complex reflectance properties. To the best of the author' s knowledge, no widely adopted alternative depth sensing technology fundamentally overcomes these issues at the hardware level. As a result, much existing research focuses on algorithmic post-processing to improve 3D data quality. For example, Rakotosaona et al. (2020) introduced Point Clean Net (PCNet) to denoise point clouds by predicting point displacements toward object surfaces. Rather than relying on a single sensor or modality, Zhang et al. (2023b) highlights the benefits of fusing multi-modal sensing and data fusion, such as RGB, depth, semantic information, graph-based representations, or latent features. Combining these complementary modalities may improve robustness and reduce the impact of noise beyond what can be achieved with raw sensor data alone.

In recent years, neural scene representations have attracted growing attention, offering an alternative to explicit geometric representations such as point clouds and voxels. Representative approaches include Neural Radiance Fields (NeRF) and 3D Gaussian Splatting (3D-GS) (Mildenhall et al., 2021; Kerbl et al., 2023), with successful implementations in agriculture (Adebola et al., 2025; Smitt et al., 2024; Zhang et al., 2024). NeRF encodes scene geometry and appearance within a neural network, while 3D-GS represents the scene as a set of optimizable 3D Gaussian ellipsoids trained through minimizing the discrepancy between predicted and observed multi-view images. Both methods reconstruct 3D scenes using RGB images and their corresponding camera poses, without relying on explicit depth measurements, thereby avoiding many of the noise issues associated with depth sensors. Some recent studies have explored integrating neural scene representations into NBV pipelines by estimating uncertainty in the current representation to guide the selection of informative viewpoints (Jin et al., 2023, 2025; Jiang et al., 2024). However, these approaches present several limitations. They lack explicit depth measurements and thus require external scaling to real-world sizes, have high computational cost that increases with scene size and resolution, and require network retraining whenever new observations are added. Their suitability for NBV planning, where computational and efficiency are crucial, remains an open question that needs to be evaluated.

6.2.3 Robotic motion

In this thesis, a robot arm equipped with a camera mounted on the end-effector served as the operational platform for data collection. Camera repositioning was carried out through robotic motion, enabling the system to capture data from multiple viewpoints for 3D reconstruction of whole plants (Chapters 3 and 4) and specific plant parts (Chapter 5). Because perception depends on camera repositioning to collect data, the accuracy, safety, reliability, and efficiency of the robot' s motion for camera

movements directly influence overall system performance. For example, the robot must move the camera to planned viewpoints precisely and quickly while avoiding collisions with plant organs and greenhouse structures. Within the context of self-supervised learning, these demands become even more critical, as robots may need to move the camera many times to explore the environment and gather sufficient training data.

This thesis considers simplified scenarios in which viewpoint selection is restricted to predefined regions around the plant and the scene is assumed to be static. In Chapter 3, candidate viewpoints were constrained to a semi-cylindrical surface with a 0.45m radius surrounding the plant. In Chapters 4 and 5, viewpoints were limited to a frontal plane 0.5m from the plant. These constraints simplified motion planning, reduced collision risk, and ensured that the NBV process remained safe and practicable. For the 3D reconstruction tasks considered in this thesis, such a simplified setup was reasonable, as the objective was complete visual coverage rather than physical interaction with the plant.

In self-supervised learning, reliable and efficient robotic motion poses extra challenges, as extensive exploratory movements may be required to collect training data. Although the self-supervised pipeline was continuously optimized from Chapter 3 to Chapter 5, which significantly reduced the number of required training iterations and therefore the amount of robot motion, the time consumed by robotic movement remains a key bottleneck in the self-supervised learning process.

To reduce robotic motion while maintaining reconstruction quality, Zeng et al. (2022) formulated NBV planning as a Deep Reinforcement Learning (DRL) problem, in which the reward function is designed to take a balance between scene reconstruction efficiency and robotic motion reduction. However, training DRL-based models typically requires extensive interaction with the environment, making the approach time-consuming and costly in real robotic settings. Another potential direction is the so-called one-shot viewpoint planning paradigm, which differs from the one-step-ahead viewpoint planning strategy adopted in this thesis. One-shot viewpoint planning assumes prior knowledge of object geometry and directly generate a complete set of viewpoints from an initial observation, followed by trajectory optimization to minimize travel distance. Representative approaches include (Hu et al., 2024; Pan et al., 2022, 2024). However, their reliance on geometric priors can possibly limit their applicability in unstructured agricultural environments.

It is important to note that the impact of robotic motion on self-supervised learning and robotic perception was not fully evaluated in this thesis. In Chapters 3–5, the real-world data used for training and evaluation were pre-recorded. As a result, the

experiments did not capture how robotic motion characteristics, such as kinematic constraints, execution time, and motion precision, influence NBV performance and online learning dynamics. Motion constraints may prevent the robot from reaching selected viewpoints, leading to missing or incomplete observations. Execution time during viewpoint transitions directly affects task efficiency and limits the efficiency at which new data can be collected for online network training. Motion precision, defined as how accurately the robot end-effector reaches the defined pose, also plays a critical role: in pose estimation tasks, imprecise positioning can introduce systematic errors in the estimated target pose, while in 3D reconstruction tasks it can cause misalignment between collected point clouds, reducing reconstruction quality. Future work should therefore include a comprehensive evaluation of robotic motion under realistic operating conditions to better understand how motion accuracy, safety, reliability, and efficiency affect the feasibility and performance of self-supervised NBV systems.

6.2.4 Coordination with other robotic subsystems toward a complete CEA robot

A successfully deployed robotic system in CEA tasks requires coordination among multiple subsystems, such as perception, navigation, decision making, and manipulation (van Henten et al., 2013). For instance, a navigation system is needed to guide the robot toward the general area of the target requiring manipulation. Once the robot reaches this area, the perception system collects information about the target to support downstream tasks. When sufficient information has been gathered, a decision-making module determines the appropriate action and moves the manipulator to the correct pose for operation. This thesis specifically focused on the perception module. As the thesis focused on perception-related challenges, the scenarios were simplified to avoid the need for tight coordination with other robotic subsystems. For example, Chapter 2 focused only on the detection module and assumed that the camera could always reach planned viewpoints. In the NBV experiments in Chapters 3, 4, and 5, each trial involved a single plant placed in the scene, and its location was varied within a limited range.

In complete autonomous operation, however, the NBV planning is more complex. Plants are arranged in dense rows and partially occluded by neighboring plants and greenhouse structures. As a result, the location of the target must be reliably identified before any operation can begin. Once the target is identified, the robot must be able to approach it safely. In practice, camera or end-effector movements may fail due to collisions with leaves, stems, trellis wires, or other structural elements. These obstacles make motion execution more difficult, because the robot must avoid damag-

ing delicate plant organs while still reaching the required viewpoint or manipulation pose. In addition, workspace constraints, narrow corridors, and dynamic factors such as airflow introduce further complexity. Failures at any of these stages can prevent successful task execution, demonstrating that accurate robotic perception alone is insufficient for complete autonomous operation.

These observations highlight that complete robotic solution places substantially stricter demands on the coordination between perception, navigation, motion planning, and manipulation than those considered in the experimental setups of this thesis. A fully autonomous system must therefore integrate robust robotic navigation, target localization, active perception, online adaptation, reliable collision avoidance, precise motion execution, and manipulation. Addressing these system-level challenges lies beyond the simplified setups used in this thesis but is essential for real-world deployment.

6.2.5 Reflection on the general conclusion

Based on the identified research limitations, further evaluation and development are required to generalize the proposed implicit NBV methods to cultivation environments and task settings beyond those investigated in this thesis. In particular, large-scale and multi-plant scenarios introduce increased variation, greater spatial complexity, and more demanding robot motion, which may require adaptations in target localization, viewpoint sampling, viewpoint selection, and overall system integration. Nevertheless, insights from the literature and the experimental results presented in this thesis indicate that such adaptations are feasible. The core principles underlying the proposed methods, including implicit NBV for viewpoint selection, learning-based feature extraction, and target-focused exploration, are not inherently tied to specific plant types or scene configurations. With appropriate extensions, the proposed systems can therefore be expected to be applicable in more complex environments.

At the same time, reliable autonomous operation in agricultural environment depends not only on perception performance but also on effective integration with other robotic modules, as discussed in Sec. 6.2.4. While this thesis intentionally focused on perception to enable systematic evaluation, the proposed methods are compatible with existing approaches in these related domains. For example, the developed viewpoint selection strategies can be combined with collision-aware motion planning and robust manipulation methods. In this context, the thesis represents a meaningful step toward more robust and adaptive perception systems for autonomous agricultural robots operating under occlusion.

Finally, the development of implicit NBV methods with online adaptation capabilities

relies on advances in self-supervised learning, training paradigms, and weakly supervised learning. The techniques explored in this thesis are therefore not limited to specific perception tasks, but may also be integrated into broader agro-food robotics and other robotic applications. In the following section, a broader perspective on the developed methods is discussed.

6.3 Broader perspectives on agro-food and other robotic domains

This section discusses the broader relevance of the methods developed in this thesis from a technological perspective on agro-food and other robotic domains. The discussion focuses on two main contributions: robotic self-supervised and continuous learning, and weakly supervised network training. Although the methods were evaluated in CEA with focus on 3D reconstruction, the underlying technologies have the potential to benefit a wide range of agro-food applications and other robotic domains where viewpoint selection affects performance, data annotation is costly, occlusions limit information acquisition, and robots must adapt continuously to dynamic environments.

6.3.1 Self-supervised learning and continuous adaptation

One of the major contributions of this thesis is a perception system capable of continuously adapting to changing environments through self-supervised learning. This was demonstrated in the context of 3D reconstruction in CEA environments, where the robot continuously generated its own training data and updated its active perception strategy during task execution.

The need for continuous adaptation extends beyond CEA. Many agricultural environments, such as orchards, arable land, and forest, exhibit dynamic conditions that significantly affect robotic perception. For example, as plants grow, their size, foliage density, and branch structure change, and new fruits or shoots may emerge. Cultivation practices such as pruning and harvesting further alter plant geometry. In addition, environmental factors such as lighting, rainfall, snowfall, and other weather conditions can modify the visual appearance of the scene. These changes can substantially reduce the performance of perception systems trained on offline data collected under earlier or different conditions, affecting a wide range of tasks that rely on visual cues, such as harvesting, crop monitoring, localization, mapping, and phenotyping. In such contexts, self-supervised learning provides a way for robots to continuously update their perception systems to adapt to new visual conditions. It should be noted, however, that the self-supervised framework developed in this thesis is specifically de-

signed for 3D reconstruction. Extending it to other tasks would require task-specific modifications, such as adapting the IG metric, network architecture, and online training pipeline.

The need for continuously adapting perception models to novel scenes is not unique to agriculture; many other robotic domains face similar challenges caused by constantly changing environments and the high cost of repeatedly retraining of perception systems. For example, warehouse picking robots must adapt their grasping strategies to new products (Deng et al., 2020; Zeng et al., 2017a). Indoor service robots must handle rearranged furniture and changing object placements (Kahn et al., 2021). Aerial robots used for inspection or mapping must update their navigation strategies to cope with evolving infrastructure and seasonal changes in vegetation. In all these cases, repeatedly collecting and annotating new training datasets is costly and often impractical. Self-supervised learning reduces dependence on manual labels and enables robots to refine their perception policies based on ongoing observations, improving long-term autonomy and robustness.

6.3.2 Weakly-supervised learning

Another important contribution of this thesis is the development of a training paradigm that enables NBV networks to be trained using weakly-supervised or sparsely annotated data. While weakly-supervised learning has been investigated in computer vision to reduce reliance on dense annotations, most work focusing on tasks such as image classification (Zhou et al., 2016), object detection (Bilen & Vedaldi, 2016), segmentation (Blok et al., 2022; Zhou et al., 2023), and 3D reconstruction (Kanazawa et al., 2016; Stutz & Geiger, 2020). To the best of my knowledge, weakly-supervised learning methods have not been studied for NBV planning methods, due to their inherent needs of ground-truth IG values for every candidate viewpoint to determine the next best viewpoint.

Weakly-supervised NBV learning introduces three key requirements. The first is selecting informative viewpoints for training when full IG labels are unavailable (Blok et al., 2022; Kootstra, 2024). The second is re-assigning IG labels to selected viewpoints. The third is designing a training pipeline that supports network training from sparse supervision. This thesis addressed these requirements as follows. First, an exploration–exploitation strategy was used to select informative viewpoints as training samples while still allowing sufficient exploration to avoid convergence to suboptimal strategies. Second, the robot physically moved the camera to selected viewpoints and measured their IG values using the proposed IG metrics. Third, a loss function was designed to enable effective learning from these sparse IG labels.

For other NBV-based 3D reconstruction tasks, such as those presented by Zeng et al. (2020) and Han et al. (2022), the weakly-supervised learning paradigm developed here could be directly applied, since these tasks also rely on informative viewpoint selection and use 3D representations which would easily be compatible with the proposed IG metric. For tasks such as classification, detection, or segmentation, the criteria for selecting informative samples and the procedures for assigning labels would need to be adapted and might still require some human supervision. Nonetheless, the core idea of designing training pipelines that make effective use of sparse annotations remains broadly useful.

6.4 Societal impact

This research advanced agricultural robotic perception by developing implicit NBV-based active perception methods that improve adaptability, real-world feasibility, efficiency, and scalability under occlusion. More reliable perception enables better overall robotic performance, which in turn supports the acceptance and practical use of robotics in society. Because this thesis focuses on agricultural applications, the societal impact is examined specifically in relation to challenges in this sector, including labor shortages and food production, and job-related changes. In the longer term, improvements in robotic performance can also contribute to the growth and maturation of the broader robotics industry.

6.4.1 Labor shortages and food production

Advancements in robotic perception help address the widely recognized challenge of labor shortages in the agricultural sector by improving robotic performance for automating tasks currently performed by humans. Greenhouse production systems are rapidly scaling: the average greenhouse size increased from 0.7ha in 1990 to 1ha in 2002 and is projected to reach about 5ha by 2030 (van Henten, 2006). As production scale expands, labor shortages and rising labor costs become increasingly restrictive. At present, many agricultural tasks, such as harvesting, deleafing, and phenotyping, remain labor-intensive because they require selective, context-dependent operations. At the same time, consumer expectations for agricultural products continue to rise. Modern markets demand high-quality agricultural products and safe production processes, which require more precise, frequent, and timely interventions throughout the crop cycle. Together, these pressures make it progressively more difficult to maintain efficient and high-quality food production using traditional labor-intensive practices.

In this context, automating labor-intensive tasks becomes essential. Improved robotic

perception enables more accurate detection and localization of objects, more reliable 3D reconstructions, and adaptation to new conditions through online learning. These capabilities provide richer and more reliable information for downstream tasks, supporting continuous growth monitoring, early disease detection, and more consistent automated harvesting and deleafing. As a result, production consistency increases, product quality improves, overall yield rises, and dependence on manual labor is significantly reduced. As global food demand continues to grow, robotic systems supported by advanced perception will play an increasingly important role in ensuring stable, efficient, and resilient food production.

It is important to acknowledge that agricultural robots require substantial upfront investment. However, evidence indicates that robotization can be financially viable in the long term. The review by Lowenberg-DeBoer et al. (2020) finds that both examined domains, automation of specific horticultural operations and autonomous machinery for open-field crop production, contain scenarios in which robotic or automated systems are profitable. Profitability is most apparent where labor costs are high or rising, or where automation enhances the timeliness or precision of field operations. The authors also note potential environmental benefits, including reduced chemical use, lower soil compaction, and more targeted interventions, although these effects remain difficult to quantify and are seldom analyzed in existing studies. Together, these findings suggest that improved robotic performance, enabled by more advanced perception, can contribute not only to labor savings but also to more sustainable and economically viable agricultural production.

6.4.2 Job-related changes

The increasing integration of robotic systems into agricultural production will inevitably reshape the nature of agricultural work. As repetitive and physically demanding tasks such as selective harvesting, deleafing, and routine crop monitoring become gradually automated, demand for manual labor in these activities is expected to decline. At the same time, many existing roles can shift from manual operation toward tasks involving robot operation, supervision, basic maintenance, and simple data interpretation. Some of these jobs are accessible to current agricultural workers with additional training, particularly those related to routine robot operation and monitoring. However, more specialized functions, such as advanced maintenance, system diagnostics, or data analytics, will require higher technical expertise, which will create new employment opportunities that demand dedicated training programs. This shift in job content may also make agricultural production more attractive to younger people (Zhang et al., 2019b). Because the reduction in physical labor and the emergence of new technical roles will occur simultaneously, the long-term net

impact on total job demand is difficult to predict. What is clearer, however, is that the transition will change the skill profile of the agricultural workforce rather than eliminate the need for human labor.

6.4.3 Broader impact on the robotics industry

Beyond agriculture, the technological developments presented in this thesis may help improve societal acceptance of robotic systems have broader implications for the robotics industry. Perception is a fundamental capability in any robotic domain that relies on flexible operation and decision-making based on visual information. Advancements in object detection, localization, scene interpretation, and environment adaptation due to improved robotic perception directly strengthen the reliability and robustness of robotic systems across many applications. The methodologies developed in this thesis are transferable to other domains where robots must operate autonomously in dynamic, cluttered, or unpredictable conditions. As perception systems become more capable, robots can execute a wider range of tasks with higher success rates, improving their effectiveness in real-world deployments. These technological improvements support broader societal acceptance of robotics and contribute to the continued growth and maturation of the robotics sector.

6.5 General conclusion

This thesis investigated how implicit NBV methods can be advanced to enable adaptive, efficient, and practically deployable robotic perception in agricultural environments under occlusion. The central hypothesis stated that implicit NBV methods equipped with online self-supervised adaptation can achieve more efficient 3D plant reconstruction in novel environments than offline-trained implicit NBV methods, following a period of online adaptation.

Chapter 3 demonstrated that integrating online self-supervised learning into the NBV pipeline enables continuous updates of the viewpoint planning policy during task execution. After adaptation, SSL-Global-NBV achieved higher reconstruction efficiency on previously unseen plant types compared with offline-trained approaches, directly supporting the hypothesis. Chapters 4 and 5 further extended this framework by reducing robotic motion and improving reconstruction efficiency for specific plant parts, respectively.

This thesis addressed four key requirements for advancing implicit NBV: adaptability, real-world feasibility, scalability, and efficiency. Adaptability was achieved through online self-supervised updates that refine the viewpoint selection strategy in novel environments. Real-world feasibility was ensured by defining information gain metrics

computable from the robot' s own observations, eliminating dependence on complete digital plant models and dense IG annotations. Scalability was improved by transitioning from global candidate-based planning to local and semantic strategies that generalize across plant sizes and perception targets. Efficiency was enhanced through reduced robotic motion, fewer IG labels, and faster training convergence.

As this thesis focused primarily on perception-related challenges under simplified conditions, including single-plant reconstruction and static scenes, future work should extend these methods to large-scale, multi-plant scenarios and integrate them with other robotic modules toward a complete robotic system.

In conclusion, this thesis demonstrates that online self-supervised learning enables implicit NBV to continuously adapt to novel environments, thereby improving the robustness of robotic perception under occlusion. This contributes to the development of more autonomous and robust agri-food robotic systems operating in dynamic conditions where occlusions frequently occur.

References

Adebola, S., Kim, C. M., Kerr, J., Xie, S., Akella, P., Rincon, J. L. S., Solowjow, E., & Goldberg, K. (2025). Botany-Bot: Digital twin monitoring of occluded and underleaf plant structures with Gaussian splats. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1839–1846). doi: 10.1109/IROS60139.2025.11247474.

Afonso, M., Barth, R., & Chauhan, A. (2019). Deep learning based plant part detection in greenhouse settings. In *12th EFITA International Conference: Digitizing Agriculture* (pp. 48–53).

Andriluka, M., Pishchulin, L., Gehler, P., & Schiele, B. (2014). 2D human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition* (pp. 3686–3693). doi: 10.1109/CVPR.2014.471.

Arad, B., Balendonck, J., Barth, R., Ben-Shahar, O., Edan, Y., Hellström, T., Hemming, J., Kurtser, P., Ringdahl, O., Tielen, T., & van Tuijl, B. (2020). Development of a sweet pepper harvesting robot. *Journal of Field Robotics*, *37*, 1027–1039. doi: 10.1002/rob.21937.

Atefi, A., Ge, Y., Pitla, S., & Schnable, J. (2021). Robotic technologies for high-throughput plant phenotyping: Contemporary reviews and future perspectives. doi: 10.3389/fpls.2021.611940.

Bac, C. W., Henten, E. J. V., Hemming, J., & Edan, Y. (2014). Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. doi: 10.1002/rob.21525.

Bajcsy, R., Aloimonos, Y., & Tsotsos, J. K. (2018). Revisiting active perception. *Autonomous Robots*, *42*, 177–196. doi: 10.1007/s10514-017-9615-3.

Bargoti, S., & Underwood, J. (2017). Deep fruit detection in orchards. In *Proceedings - IEEE International Conference on Robotics and Automation* (pp. 3626–3633). doi: 10.1109/ICRA.2017.7989417.

Bartolomei, L., Teixeira, L., & Chli, M. (2021). Semantic-aware active perception for uavs using deep reinforcement learning. In *IEEE International Conference on Intelligent Robots and Systems* (pp. 3101–3108). Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/IROS51168.2021.9635893.

Battiato, S., Castorina, A., & Mancuso, M. (2003). High dynamic range imaging for digital still camera: an overview. *Journal of Electronic Imaging*, *12*, 459–469. doi:

10.1117/1.1580829.

Bechar, A., & Vigneault, C. (2016). Agricultural robots for field operations: Concepts and components. doi: 10.1016/j.biosystemseng.2016.06.014.

Benavides, M., Cantón-Garbín, M., Sánchez-Molina, J. A., & Rodríguez, F. (2020). Automatic tomato and peduncle location system based on computer vision for use in robotized harvesting. *Applied Sciences (Switzerland)*, *10*. doi: 10.3390/app10175887.

Bergerman, M., Billingsley, J., Reid, J., & van Henten, E. (2016). Robotics in Agriculture and Forestry. In B. Siciliano, & O. Khatib (Eds.), *Springer Handbook of Robotics* (pp. 1463–1492). Cham: Springer International Publishing. doi: 10.1007/978-3-319-32552-1_56.

Berkers, E., & Geels, F. W. (2011). System innovation through stepwise reconfiguration: the case of technological transitions in dutch greenhouse horticulture (1930 – 1980). *Technology Analysis & Strategic Management*, *23*, 227–247. doi: 10.1080/09537325.2011.550392.

Bilen, H., & Vedaldi, A. (2016). Weakly supervised deep detection networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2846–2854). doi: 10.1109/CVPR.2016.311.

Blok, P. M., Kootstra, G., Elghor, H. E., Diallo, B., van Evert, F. K., & van Henten, E. J. (2022). Active learning with MaskAL reduces annotation effort for training Mask R-CNN on a broccoli dataset with visually similar classes. *Computers and Electronics in Agriculture*, *197*. doi: 10.1016/j.compag.2022.106917.

Bonsignorio, F., Hallam, J., & Pobil, A. D. (2009). Defining the requisites of a replicable robotics experiment. In *RSS2009 Workshop on Good Experimental Methodologies in Robotics*.

Boogaard, F. P., van Henten, E. J., & Kootstra, G. (2021). Boosting plant-part segmentation of cucumber plants by enriching incomplete 3D point clouds with spectral data. *Biosystems Engineering*, *211*, 167–182. doi: 10.1016/j.biosystemseng.2021.09.004.

Boogaard, F. P., van Henten, E. J., & Kootstra, G. (2022). Improved Point-Cloud Segmentation for Plant Phenotyping Through Class-Dependent Sampling of Training Data to Battle Class Imbalance. *Frontiers in Plant Science*, *13*. doi: 10.3389/fpls.2022.838190.

Boogaard, F. P., van Henten, E. J., & Kootstra, G. (2023). The added value of 3D point clouds for digital plant phenotyping – A case study on internode length measurements in cucumber. *Biosystems Engineering*, *234*, 1–12. doi:

10.1016/j.biosystemseng.2023.08.010.

Boogaard, F. P., Rongen, K. S., & Kootstra, G. W. (2020). Robust node detection and tracking in fruit-vegetable crops using deep learning and multi-view imaging. *Biosystems Engineering*, *192*, 117–132. doi: 10.1016/j.biosystemseng.2020.01.023.

Burusa, A. K., van Henten, E. J., & Kootstra, G. (2024a). Attention-driven next-best-view planning for efficient reconstruction of plants and targeted plant parts. *Biosystems Engineering*, *246*, 248–262. doi: 10.1016/j.biosystemseng.2024.08.002.

Burusa, A. K., van Henten, E. J., & Kootstra, G. (2024b). Gradient-based local next-best-view planning for improved perception of targeted plant nodes. In *2024 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 15854–15860). IEEE. doi: 10.1109/ICRA57147.2024.10610397.

Burusa, A. K., Scholten, J., Wang, X., Rapado-Rincón, D., van Henten, E. J., & Kootstra, G. (2024c). Semantics-aware next-best-view planning for efficient search and detection of task-relevant plant parts. *Biosystems Engineering*, *248*, 1–14. doi: 10.1016/j.biosystemseng.2024.09.018.

Caldeira, M. P. C., & Moreno, P. (2024). Next best view for point-cloud model acquisition: Bayesian approximation and uncertainty analysis. In *2024 Eighth IEEE International Conference on Robotic Computing (IRC)* (pp. 91–96). IEEE. doi: 10.1109/IRC63610.2024.00023.

Carion, N. et al. (2025). SAM 3: Segment anything with concepts. *arXiv preprint* arXiv:2511.16719.

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *37th International Conference on Machine Learning (ICML 2020)* (pp. 1575–1585).

Cheng, Y., Lan, S., Fan, X., Tjahjadi, T., Jin, S., & Cao, L. (2023). A dual-branch weakly supervised learning based network for accurate mapping of woody vegetation from remote sensing images. *International Journal of Applied Earth Observation and Geoinformation*, *124*, 103499. doi: 10.1016/j.jag.2023.103499.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint* arXiv:1412.3555.

Ci, J., van Henten, E. J., Wang, X., Burusa, A. K., & Kootstra, G. (2025). Ssl-nbv: A self-supervised-learning-based next-best-view algorithm for efficient 3d plant reconstruction by a robot. *Computers and Electronics in Agriculture*, *233*. doi: 10.1016/j.compag.2025.110121.

Ci, J., Wang, X., Rapado-Rincón, D., Burusa, A. K., & Kootstra, G. (2024). 3d pose estimation of tomato peduncle nodes using deep keypoint detection and point cloud.

Biosystems Engineering, *243*, 57–69. doi: 10.1016/j.biosystemseng.2024.04.017.

Delmerico, J., Isler, S., Sabzevari, R., & Scaramuzza, D. (2018). A comparison of volumetric information gain metrics for active 3d object reconstruction. *Autonomous Robots*, *42*, 197–208. doi: 10.1007/s10514-017-9634-0.

Deng, X., Xiang, Y., Mousavian, A., Eppner, C., Bretl, T., & Fox, D. (2020). Self-supervised 6d object pose estimation for robot manipulation. In *Proceedings - IEEE International Conference on Robotics and Automation* (pp. 3665–3671). doi: 10.1109/ICRA40945.2020.9196714.

Eizentals, P., & Oka, K. (2016). 3D pose estimation of green pepper fruit for automated harvesting. *Computers and Electronics in Agriculture*, *128*, 127–140. doi: 10.1016/j.compag.2016.08.024.

FAO (2018). *The future of food and agriculture – Alternative pathways to 2050*. Technical Report, Food and Agriculture Organization of the United Nations Rome. URL: `https://www.fao.org/3/I8429EN/i8429en.pdf`.

Fountas, S., Malounas, I., Athanasakos, L., Avgoustakis, I., & Espejo-Garcia, B. (2022). Ai-assisted vision for agricultural robots. doi: 10.3390/agriengineering4030043.

Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A. J., Conradt, J., & Daniilidis, K. (2020). Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*, 154–180. doi: 10.1109/TPAMI.2020.3008413.

Gibbs, J. A., Pound, M., French, A. P., Wells, D. M., Murchie, E., & Pridmore, T. (2018). Plant phenotyping: An active vision cell for three-dimensional plant shoot reconstruction1[open]. *Plant Physiology*, *178*, 524–534. doi: 10.1104/PP.18.00664.

Gibbs, J. A., Pound, M. P., French, A. P., Wells, D. M., Murchie, E. H., & Pridmore, T. P. (2020). Active vision and surface reconstruction for 3d plant shoot modelling. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *17*, 1907–1917. doi: 10.1109/TCBB.2019.2896908.

Golbach, F., Kootstra, G., Damjanovic, S., Otten, G., & van de Zedde, R. (2016). Validation of plant part measurements using a 3d reconstruction method suitable for high-throughput seedling phenotyping. *Machine Vision and Applications*, *27*, 663–680. doi: 10.1007/s00138-015-0727-5.

Hagras, H., Colley, M., Callaghan, V., & Carr-West, M. (2002). Online learning and adaptation of autonomous mobile robots for sustainable agriculture. *Autonomous Robots*, *13*, 37–52. doi: 10.1023/A:1015626121039.

Halstead, M., McCool, C., Denman, S., Perez, T., & Fookes, C. (2018). Fruit Quan-

tity and Ripeness Estimation Using a Robotic Vision System. *IEEE Robotics and Automation Letters*, *3*, 2995–3002. doi: 10.1109/LRA.2018.2849514.

Han, Y., Zhan, I. H., Zhao, W., & Liu, Y.-J. (2022). *A Double Branch Next-Best-View Network and Novel Robot System for Active Object Reconstruction.* doi: 10.1109/ICRA46639.2022.9811769.

Hartmann, A., Czauderna, T., Hoffmann, R., Stein, N., & Schreiber, F. (2011). HTPheno: An image analysis pipeline for high-throughput plant phenotyping. *BMC Bioinformatics*, *12*. doi: 10.1186/1471-2105-12-148.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2020). Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*, 386–397. doi: 10.1109/TPAMI.2018.2844175.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). doi: 10.1109/CVPR.2016.90.

van der Heijden, G., Song, Y., Horgan, G., Polder, G., Dieleman, A., Bink, M., Palloix, A., van Eeuwijk, F., & Glasbey, C. (2012). Spicy: towards automated phenotyping of large pepper plants in the greenhouse. *Functional Plant Biology*, *39*, 870–877. doi: 10.1071/FP12019.

Hemmerling, R., Kniemeyer, O., Lanwert, D., Kurth, W., & Buck-Sorlin, G. (2008). The rule-based language xl and the modelling environment groimp illustrated with simulated tree competition. *Functional Plant Biology*, *35*, 739–750. doi: 10.1071/FP08052.

Hemming, J., Ruizendaal, J., Hofstee, J. W., & van Henten, E. J. (2014). Fruit detectability analysis for different camera positions in sweet-pepper. *Sensors*, *14*, 6032–6044. doi: 10.3390/s140406032.

van Henten, E. J. (2006). Greenhouse mechanization: State of the art and future perspective. In *International Symposium on Greenhouses, Environmental Controls and In-house Mechanization for Crop Production in the Tropics* (pp. 55–70). ISHS volume 710 of *Acta Horticulturae*. doi: 10.17660/ActaHortic.2006.710.3.

van Henten, E. J., Bac, C. W., Hemming, J., & Edan, Y. (2013). Robotics in protected cultivation. In A. Visala (Ed.), *Proceedings of the 4th IFAC Conference on Modelling and Control in Agriculture, Horticulture and Post Harvest Industry* (pp. 170–177). Espoo, Finland volume 46 of *IFAC Proceedings Volumes, Number 18*. doi: 10.3182/20130828-2-SF-3019.00070.

van Henten, E. J., Hemming, J., van Tuijl, B. A. J., Kornet, J. G., Meuleman, J., Bontsema, J., & van Os, E. A. (2002). An autonomous robot for

harvesting cucumbers in greenhouses. *Autonomous Robots*, *13*, 241–258. doi: 10.1023/A:1020568125418.

van Henten, E. J., van Tuijl, B. A. J., Hemming, J., Kornet, J. G., Bontsema, J., & van Os, E. A. (2003). Field test of an autonomous cucumber picking robot. *Biosystems Engineering*, *86*, 305–313. doi: 10.1016/j.biosystemseng.2003.08.002.

van Henten, E. J., van Tuijl, B. A. J., Hoogakker, G. J., van der Weerd, M. J., Hemming, J., Kornet, J. G., & Bontsema, J. (2006). An autonomous robot for de-leafing cucumber plants grown in a high-wire cultivation system. *Biosystems Engineering*, *94*, 317–323. doi: 10.1016/j.biosystemseng.2006.03.005.

Heuvelink, E., & Dorais, M. (2005). Crop growth and yield. In *Tomatoes* number 13 in Crop Production Science in Horticulture Series (pp. 85–144). Wallingford, UK: CABI. doi: 10.1079/9780851993966.0085.

Heuvelink, E., Stanghellini, C., & van 't Ooster, B. (2024). *Greenhouse horticulture: Technology for optimal crop production.* Brill. doi: 10.1163/9789004697041.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780. doi: 10.1162/neco.1997.9.8.1735.

Hu, H., Pan, S., Jin, L., Popović, M., & Bennewitz, M. (2024). Active implicit reconstruction using one-shot view planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 12477–12483). doi: 10.1109/ICRA57147.2024.10611542.

Isler, S., Sabzevari, R., Delmerico, J., & Scaramuzza, D. (2016). An information gain formulation for active volumetric 3D reconstruction. In *Proceedings - IEEE International Conference on Robotics and Automation* (pp. 3477–3484). volume 2016-June. doi: 10.1109/ICRA.2016.7487527.

Jansen, M. et al. (2009). Simultaneous phenotyping of leaf growth and chlorophyll fluorescence via Growscreen Fluoro allows detection of stress tolerance in Arabidopsis thaliana and other rosette plants. *Functional Plant Biology*, *36*, 902–914. doi: 10.1071/FP09095.

Ji, W., Zhao, D., Cheng, F., Xu, B., Zhang, Y., & Wang, J. (2012). Automatic recognition vision system guided for apple harvesting robot. *Computers & Electrical Engineering*, *38*, 1186–1195. doi: 10.1016/j.compeleceng.2011.11.005.

Jiang, W., Lei, B., & Daniilidis, K. (2024). Fisherrf: Active view selection and mapping with radiance fields using fisher information. In *European Conference on Computer Vision* (pp. 422–440). Springer. doi: 10.1007/978-3-031-72624-8_24.

Jin, L., Chen, X., Ruckin, J., & Popovic, M. (2023). Neu-nbv: Next best view planning using uncertainty estimation in image-based neural rendering. In *IEEE Interna-*

tional Conference on Intelligent Robots and Systems (pp. 11305–11312). Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/IROS55552.2023.10342226.

Jin, L., Kuang, H., Pan, Y., Stachniss, C., & Popović, M. (2024). Stair: Semantic-targeted active implicit reconstruction. In *IEEE International Conference on Intelligent Robots and Systems* (pp. 395–402). Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/IROS58592.2024.10801401.

Jin, L., Zhong, X., Pan, Y., Behley, J., Stachniss, C., & Popovic, M. (2025). ActiveGS: Active Scene Reconstruction Using Gaussian Splatting. *IEEE Robotics and Automation Letters*, *10*, 4866–4873. doi: 10.1109/LRA.2025.3555149.

Kahn, G., Abbeel, P., & Levine, S. (2021). Badgr: An autonomous self-supervised learning-based navigation system. *IEEE Robotics and Automation Letters*, *6*, 1312–1319. doi: 10.1109/LRA.2021.3057023.

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, *147*, 70–90. doi: 10.1016/J.COMPAG.2018.02.016.

Kanazawa, A., Jacobs, D. W., & Chandraker, M. (2016). Warpnet: Weakly supervised matching for single-view reconstruction. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3253–3261). doi: 10.1109/CVPR.2016.354.

Kang, H., & Chen, C. (2020). Fruit detection, segmentation and 3D visualisation of environments in apple orchards. *Computers and Electronics in Agriculture*, *171*, 105302. doi: 10.1016/J.COMPAG.2020.105302.

Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, *42*, 139:1–139:14. doi: 10.1145/3592433.

Kim, T., Lee, D.-H., Kim, K.-C., & Kim, Y.-J. (2023). 2D pose estimation of multiple tomato fruit-bearing systems for robotic harvesting. *Computers and Electronics in Agriculture*, *211*, 108004. doi: 10.1016/j.compag.2023.108004.

Koenig, N., & Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 2149–2154). IEEE volume 3. doi: 10.1109/IROS.2004.1389727.

Kooten, O. V., Heuvelink, E., & Stanghellini, C. (2008). New developments in greenhouse technology can mitigate the water shortage problem of the 21st century. In *Acta Horticulturae* (pp. 45–52). International Society for Horticultural Science volume 767. doi: 10.17660/ActaHortic.2008.767.2.

Kootstra, G. (2024). Advances in visual perception for agricultural robotics. In *Advances in agri-food robotics* (pp. 3–36). Burleigh Dodds Science Publishing. doi: 10.19103/AS.2023.0124.01.

Kootstra, G., Wang, X., Blok, P. M., Hemming, J., & Van Henten, E. (2021). Selective harvesting robotics: current research, trends, and future directions. *Current Robotics Reports*, *2*, 95–104. doi: 10.1007/s43154-020-00034-1.

Kortylewski, A., Liu, Q., Wang, A., Sun, Y., & Yuille, A. (2021). Compositional convolutional neural networks: A robust and interpretable model for object recognition under occlusion. *International Journal of Computer Vision*, *129*, 736–760. doi: 10.1007/s11263-020-01401-3.

Lee, S., Chen, L., Wang, J., Liniger, A., Kumar, S., & Yu, F. (2022). Uncertainty Guided Policy for Active Robotic 3D Reconstruction Using Neural Radiance Fields. *IEEE Robotics and Automation Letters*, *7*, 12070–12077. doi: 10.1109/LRA.2022.3212668.

Lehnert, C., English, A., McCool, C., Tow, A. W., & Perez, T. (2017). Autonomous Sweet Pepper Harvesting for Protected Cropping Systems. *IEEE Robotics and Automation Letters*, *2*, 872–879. doi: 10.1109/LRA.2017.2655622.

Lehnert, C., Sa, I., McCool, C., Upcroft, B., & Perez, T. (2016). Sweet pepper pose detection and grasping for automated crop harvesting. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2428–2434). doi: 10.1109/ICRA.2016.7487394.

Lehnert, C., Tsai, D., Eriksson, A., & McCool, C. (2019). 3D move to see: Multi-perspective visual servoing towards the next best view within unstructured and occluded environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 3890–3897). doi: 10.1109/IROS40897.2019.8967918.

Li, H., Zhu, Q., Huang, M., Guo, Y., & Qin, J. (2018). Pose estimation of sweet pepper through symmetry axis detection. *Sensors (Switzerland)*, *18*. doi: 10.3390/s18093083.

Li, X., Cai, C., Zhang, R., Ju, L., & He, J. (2019). Deep cascaded convolutional models for cattle pose estimation. *Computers and Electronics in Agriculture*, *164*. doi: 10.1016/j.compag.2019.104885.

Liang, C., Xiong, J., Zheng, Z., Zhong, Z., Li, Z., Chen, S., & Yang, Z. (2020). A visual detection method for nighttime litchi fruits and fruiting stems. *Computers and Electronics in Agriculture*, *169*, 105192. doi: 10.1016/j.compag.2019.105192.

Lin, G., Tang, Y., Zou, X., Xiong, J., & Li, J. (2019). Guava detection and pose estimation using a low-cost RGB-D sensor in the field. *Sensors*, *19*, 428. doi:

10.3390/s19020428.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2117–2125). doi: 10.1109/CVPR.2017.106.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)* (pp. 740–755). Springer. doi: 10.1007/978-3-319-10602-1_48.

Long, S. P., Marshall-Colon, A., & Zhu, X.-G. (2015). Meeting the global food demand of the future by engineering crop photosynthesis and yield potential. *Cell*, *161*, 56–66. doi: 10.1016/j.cell.2015.03.019.

Lowenberg-DeBoer, J., Huang, I. Y., Grigoriadis, V., & Blackmore, S. (2020). Economics of robots and automation in field crop production. *Precision Agriculture*, *21*, 278–299. doi: 10.1007/s11119-019-09667-5.

Lu, H., Tang, L., Whitham, S. A., & Mei, Y. (2017). A robotic platform for corn seedling morphological traits characterization. *Sensors (Switzerland)*, *17*. doi: 10.3390/s17092082.

Luo, L., Tang, Y., Zou, X., Ye, M., Feng, W., & Li, G. (2016). Vision-based extraction of spatial information in grape clusters for harvesting robots. *Biosystems Engineering*, *151*, 90–104. doi: 10.1016/j.biosystemseng.2016.08.026.

Luo, L., Yin, W., Ning, Z., Wang, J., Wei, H., Chen, W., & Lu, Q. (2022). In-field pose estimation of grape clusters with combined point cloud segmentation and geometric analysis. *Computers and Electronics in Agriculture*, *200*. doi: 10.1016/j.compag.2022.107197.

Manzano-Agugliaro, F., & Cañero-Leon, R. (2010). Economics and environmental analysis of Mediterranean greenhouse crops. *African Journal of Agricultural Research*, *5*, 3009–3016. doi: 10.5897/AJAR.9000192.

van Marrewijk, B. M., van Daalen, T., Xin, B., van Henten, E. J., Polder, G., & Kootstra, G. (2025). 3D plant segmentation: Comparing a 2D-to-3D segmentation method with state-of-the-art 3D segmentation algorithms. *Biosystems Engineering*, *254*. doi: 10.1016/j.biosystemseng.2025.104147.

Mathis, A., Biasi, T., Mert, Y., Rogers, B., Bethge, M., & Mathis, M. W. (2020). Imagenet performance correlates with pose estimation robustness and generalization on out-of-domain data. In *International Conference on Machine Learning (ICML) Workshop on Uncertainty and Robustness in Deep Learning*.

Mendoza, M., Vasquez-Gomez, J. I., Taud, H., Sucar, L. E., & Reta, C. (2020). Supervised learning of the next-best-view for 3d object reconstruction. *Pattern Recognition Letters*, *133*, 224–231. doi: 10.1016/j.patrec.2020.02.024.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2021). NeRF: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, *65*, 99–106. doi: 10.1145/3503250.

Minervini, M., Abdelsamea, M. M., & Tsaftaris, S. A. (2014). Image-based plant phenotyping with incremental learning and active contours. *Ecological Informatics*, *23*, 35–48. doi: 10.1016/j.ecoinf.2013.07.004.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint* arXiv:1312.5602.

Nemali, K. (2022). History of controlled environment horticulture: Greenhouses. In *HortScience* (pp. 239–246). American Society for Horticultural Science volume 57. doi: 10.21273/HORTSCI16160-21.

Noguchi, N., Will, J., Reid, J., & Zhang, Q. (2004). Development of a master-slave robot system for farm operations. *Computers and Electronics in Agriculture*, *44*, 1–19. doi: 10.1016/j.compag.2004.01.006.

Oliveira, L. F., Moreira, A. P., & Silva, M. F. (2021). Advances in agriculture robotics: A state-of-the-art review and challenges ahead. doi: 10.3390/robotics10020052.

Pan, S., Hu, H., & Wei, H. (2022). SCVP: Learning One-Shot View Planning via Set Covering for Unknown Object Reconstruction. *IEEE Robotics and Automation Letters*, *7*, 1463–1470. doi: 10.1109/LRA.2022.3140449.

Pan, S., Jin, L., Huang, X., Stachniss, C., Popović, M., & Bennewitz, M. (2024). Exploiting priors from 3D diffusion models for RGB-based one-shot view planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 13341–13348). IEEE. doi: 10.1109/IROS58592.2024.10802551.

Paris, H. S., Janick, J., & Pitrat, M. (2008). What the Roman emperor Tiberius grew in his greenhouses. *Chronica Horticulturae*, *48*, 12–14.

Pedersen, S. M., Fountas, S., & Blackmore, S. (2008). Agricultural robots—Applications and economic perspectives. In Y. Takahashi (Ed.), *Service Robot Applications* chapter 21. London: IntechOpen. doi: 10.5772/6048.

Pereira, T. D., Aldarondo, D. E., Willmore, L., Kislin, M., Wang, S. S., Murthy, M., & Shaevitz, J. W. (2019). Fast animal pose estimation using deep neural networks. *Nature Methods*, *16*, 117–125. doi: 10.1038/s41592-018-0234-5.

Poland, J. A., & Rife, T. W. (2012). Genotyping-by-Sequencing for Plant Breeding

and Genetics. *The Plant Genome*, *5*. doi: 10.3835/plantgenome2012.05.0005.

Polder, G., & Hofstee, J. W. (2014). Phenotyping large tomato plants in the greenhouse using a 3D light-field camera. In *Proceedings of the 2014 ASABE and CSBE/SCGAB Annual International Meeting* (pp. 153–159). Montreal, QC, Canada: American Society of Agricultural and Biological Engineers volume 1. doi: 10.13031/aim.20141913615.

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 652–660). doi: 10.1109/CVPR.2017.16.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A. Y. (2009). ROS: an open-source Robot Operating System. In *ICRA workshop on open source software* (p. 5). Kobe, Japan volume 3.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., & Courville, A. (2019). On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)* (pp. 5301–5310). PMLR volume 97 of *Proceedings of Machine Learning Research*.

Rajesh, T., Ramesh, D., Nagateja, K. S. S. N., & Vamshi, P. (2020). Plant protection using cold frame. In V. Rani (Ed.), *Recent Advances in Agriculture and Space Technology* chapter 1. volume 103.

Rakotosaona, M., La Barbera, V., Guerrero, P., Mitra, N. J., & Ovsjanikov, M. (2020). Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In *Computer graphics forum* (pp. 185–203). Wiley Online Library volume 39. doi: 10.48550/arXiv.1901.01060.

Rapado-Rincón, D., van Henten, E. J., & Kootstra, G. (2023). Development and evaluation of automated localisation and reconstruction of all fruits on tomato plants in a greenhouse based on multi-view perception and 3D multi-object tracking. *Biosystems Engineering*, *231*, 78–91. doi: 10.1016/j.biosystemseng.2023.06.003.

Rapado-Rincon, D., Nap, H., Smolenova, K., van Henten, E. J., & Kootstra, G. (2024). MOT-DETR: 3D single shot detection and tracking with transformers to build 3D representations for agro-food robots. *Computers and Electronics in Agriculture*, *225*. doi: 10.1016/j.compag.2024.109275.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint* arXiv:1804.02767.

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Anal-*

ysis and Machine Intelligence, *39*, 1137–1149. doi: 10.1109/TPAMI.2016.2577031.

Rong, J., Dai, G., & Wang, P. (2022). A peduncle detection method of tomato for autonomous harvesting. *Complex & Intelligent Systems*, *8*, 2955–2969. doi: 10.1007/s40747-021-00522-7.

Russello, H., van der Tol, R., & Kootstra, G. (2022). T-LEAP: Occlusion-robust pose estimation of walking cows using temporal information. *Computers and Electronics in Agriculture*, *192*. doi: 10.1016/j.compag.2021.106559.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). DeepFruits: A fruit detection system using deep neural networks. *Sensors*, *16*, 1222. doi: 10.3390/s16081222.

Sa, I., Lehnert, C., English, A., McCool, C., Dayoub, F., Upcroft, B., & Perez, T. (2017). Peduncle detection of sweet pepper for autonomous crop harvesting—Combined color and 3-D information. *IEEE Robotics and Automation Letters*, *2*, 765–772. doi: 10.1109/LRA.2017.2651952.

Santos, T. T., de Souza, L. L., dos Santos, A. A., & Avila, S. (2020). Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association. *Computers and Electronics in Agriculture*, *170*, 105247. doi: 10.1016/J.COMPAG.2020.105247.

Schonberger, J. L., & Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4104–4113). doi: 10.1109/CVPR.2016.445.

Scott, W. R., Roth, G., & Rivest, J.-F. (2003). View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, *35*, 64–96. doi: 10.1145/641865.641868.

Shi, W., van de Zedde, R., Jiang, H., & Kootstra, G. (2019). Plant-part segmentation using deep learning and multi-view vision. *Biosystems Engineering*, *187*, 81–95. doi: 10.1016/j.biosystemseng.2019.08.014.

Smitt, C., Halstead, M., Zimmer, P., Labe, T., Guclu, E., Stachniss, C., & Mccool, C. (2024). PAg-NeRF: Towards Fast and Efficient End-to-End Panoptic 3D Representations for Agricultural Robotics. *IEEE Robotics and Automation Letters*, *9*, 907–914. doi: 10.1109/LRA.2023.3338515.

Stutz, D., & Geiger, A. (2020). Learning 3D Shape Completion Under Weak Supervision. *International Journal of Computer Vision*, *128*, 1162–1181. doi: 10.1007/s11263-018-1126-y.

Tarabanis, K. A., Allen, P. K., & Tsai, R. Y. (1095). A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, *11*, 86–104. doi:

10.1109/70.345940.

Tenenbaum, J. M. (1971). *Accommodation in computer vision*. Ph.D. thesis Stanford University Stanford, CA, USA.

Thapa, S., Zhu, F., Walia, H., Yu, H., & Ge, Y. (2018). A novel LiDAR-based instrument for high-throughput, 3D measurement of morphological traits in maize and sorghum. *Sensors*, *18*, 1187. doi: 10.3390/s18041187.

Tisné, S. et al. (2013). Phenoscope: An automated large-scale phenotyping platform offering high spatial homogeneity. *Plant Journal*, *74*, 534–544. doi: 10.1111/tpj.12131.

Vashisth, A., Rückin, J., Magistri, F., Stachniss, C., & Popović, M. (2024). Deep reinforcement learning with dynamic graphs for adaptive informative path planning. *IEEE Robotics and Automation Letters*, *9*, 7747–7754. doi: 10.1109/LRA.2024.3421188.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)* (pp. 5998–6008). Curran Associates, Inc. volume 30.

Vázquez-Arellano, M., Reiser, D., Paraforos, D. S., Garrido-Izard, M., & Griepentrog, H. W. (2018). Leaf area estimation of reconstructed maize plants using a time-of-flight camera based on different scan directions. *Robotics*, *7*. doi: 10.3390/robotics7040063.

Virlet, N., Sabermanesh, K., Sadeghi-Tehran, P., & Hawkesford, M. J. (2017). Field Scanalyzer: An automated robotic field phenotyping platform for detailed crop monitoring. doi: 10.1071/FP16163.

Vit, A., Shani, G., & Bar-Hillel, A. (2020). Length phenotyping with interest point detection. *Computers and Electronics in Agriculture*, *176*, 105629. doi: 10.1016/j.compag.2020.105629.

Wagner, N., Kirk, R., Hanheide, M., & Cielniak, G. (2021). Efficient and robust orientation estimation of strawberries for fruit picking applications. In *2021 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 13857–13863). doi: 10.1109/ICRA48506.2021.9561848.

Wang, H., Zhang, G., Cao, H., Hu, K., Wang, Q., Deng, Y., Gao, J., & Tang, Y. (2025). Geometry-aware 3D point cloud learning for precise cutting-point detection in unstructured field environments. *Journal of Field Robotics*, *42*, 3063–3076. doi: 10.1002/rob.22567.

Wang, T., Xi, W., Cheng, Y., Han, H., & Yang, Y. (2024). RL-NBV: A deep rein-

forcement learning based next-best-view method for unknown object reconstruction. *Pattern Recognition Letters*, *184*, 1–6. doi: 10.1016/j.patrec.2024.05.014.

Wang, Y., James, S., Stathopoulou, E. K., Beltrán-González, C., Konishi, Y., & Del Bue, A. (2019a). Autonomous 3-D reconstruction, mapping, and exploration of indoor environments with a robotic arm. *IEEE Robotics and Automation Letters*, *4*, 3340–3347. doi: 10.1109/LRA.2019.2926676.

Wang, Y., Wu, S., Huang, H., Cohen-Or, D., & Sorkine-Hornung, O. (2019b). Patch-based progressive 3D point set upsampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 5958–5967). doi: 10.1109/CVPR.2019.00611.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016). Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)* (pp. 1995–2003). PMLR volume 48 of *Proceedings of Machine Learning Research*.

Wearn, J. A., & Mabberley, D. J. (2016). Citrus and orangeries in Northern Europe. *Curtis's Botanical Magazine*, *33*, 94–107. doi: 10.1111/curt.12128.

Wu, C., Zeng, R., Pan, J., Wang, C. C., & Liu, Y. J. (2019a). Plant phenotyping by deep-learning-based planner for multi-robots. *IEEE Robotics and Automation Letters*, *4*, 3113–3120. doi: 10.1109/LRA.2019.2924125.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019b). Detectron2. `https://github.com/facebookresearch/detectron2`. Facebook AI Research.

Xue, S., Dill, J., Mathur, P., Dellaert, F., Tsiotra, P., & Xu, D. (2024). Neural visibility field for uncertainty-driven active mapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 18122–18132). doi: 10.1109/CVPR52733.2024.01716.

Ye, L., Duan, J., Yang, Z., Zou, X., Chen, M., & Zhang, S. (2021). Collision-free motion planning for the litchi-picking robot. *Computers and Electronics in Agriculture*, *185*. doi: 10.1016/j.compag.2021.106151.

Yoshida, T., Fukao, T., & Hasegawa, T. (2018). Fast detection of tomato peduncle using point cloud with a harvesting robot. *Journal of Robotics and Mechatronics*, *30*, 180–186. doi: 10.20965/jrm.2018.p0180.

Yoshida, T., Fukao, T., & Hasegawa, T. (2020). Cutting point detection using a robot with point clouds for tomato harvesting. *Journal of Robotics and Mechatronics*, *32*, 437–444. doi: 10.20965/jrm.2020.p0437.

Zaenker, T., Lehnert, C., McCool, C., & Bennewitz, M. (2021a). Combining local and global viewpoint planning for fruit coverage. In *2021 10th European Conference on*

Mobile Robots, ECMR 2021 - Proceedings. doi: 10.1109/ECMR50962.2021.9568836.

Zaenker, T., Rückin, J., Menon, R., Popović, M., & Bennewitz, M. (2023). Graph-based view motion planning for fruit detection. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 4219–4225). doi: 10.1109/IROS55552.2023.10342532.

Zaenker, T., Smitt, C., McCool, C., & Bennewitz, M. (2021b). Viewpoint planning for fruit size and position estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 3271–3277). doi: 10.1109/IROS51168.2021.9636701.

Zapotezny-Anderson, P., & Lehnert, C. (2019). Towards active robotic vision in agriculture: A deep learning approach to visual servoing in occluded and unstructured protected cropping environments. In *IFAC-PapersOnLine* (pp. 120–125). volume 52. doi: 10.1016/j.ifacol.2019.12.508.

Zeng, A., Yu, K. T., Song, S., Suo, D., Walker, E., Rodriguez, A., & Xiao, J. (2017a). Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *Proceedings - IEEE International Conference on Robotics and Automation* (pp. 1386–1393). doi: 10.1109/ICRA.2017.7989165.

Zeng, A., Yu, K. T., Song, S., Suo, D., Walker, E., Rodriguez, A., & Xiao, J. (2017b). Multi-view self-supervised deep learning for 6D pose estimation in the Amazon Picking Challenge. In *Proceedings - IEEE International Conference on Robotics and Automation* (pp. 1386–1393). doi: 10.1109/ICRA.2017.7989165.

Zeng, R., Zhao, W., & Liu, Y. J. (2020). Pc-nbv: A point cloud based deep network for efficient next best view planning. In *IEEE International Conference on Intelligent Robots and Systems* (pp. 7050–7057). Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/IROS45743.2020.9340916.

Zeng, X., Zaenker, T., & Bennewitz, M. (2022). Deep reinforcement learning for next-best-view planning in agricultural applications. In *2022 International Conference on Robotics and Automation (ICRA)* (pp. 2323–2329). doi: 10.1109/ICRA46639.2022.9811800.

Zhang, D., Gajardo, J., Medic, T., Katircioglu, I., Boss, M., Kirchgessner, N., Walter, A., & Roth, L. (2024). Wheat3DGS: In-field 3D reconstruction, instance segmentation and phenotyping of wheat heads with Gaussian splatting. *arXiv preprint* arXiv:2412.18146.

Zhang, F., Gao, J., Song, C., Zhou, H., Zou, K., Xie, J., Yuan, T., & Zhang, J. (2023a). TPMv2: An end-to-end tomato pose method based on 3D key points detection. *Computers and Electronics in Agriculture*, *210*. doi: 10.1016/j.compag.2023.107878.

Zhang, F., Gao, J., Zhou, H., Zhang, J., Zou, K., & Yuan, T. (2022). Three-dimensional pose detection method based on keypoints detection network for tomato bunch. *Computers and Electronics in Agriculture*, *195*, 106824.

Zhang, H., Goodfellow, I., Metaxas, D., & Odena, A. (2019a). Self-attention generative adversarial networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)* (pp. 7354–7363). PMLR volume 97 of *Proceedings of Machine Learning Research*.

Zhang, Q., Karkee, M., & Tabb, A. (2019b). The use of agricultural robots in orchard management. In *Robotics and automation for improving agriculture* (pp. 187–214). Burleigh Dodds Science Publishing.

Zhang, S., & Sutton, R. S. (2017). A deeper look at experience replay. *arXiv preprint* arXiv:1712.01275.

Zhang, Y., Carballo, A., Yang, H., & Takeda, K. (2023b). Perception and sensing for autonomous vehicles under adverse weather conditions: A survey. *ISPRS Journal of Photogrammetry and Remote Sensing*, *196*, 146–177. doi: 10.1016/j.isprsjprs.2022.12.021.

Zhang, Z., Xie, C., Wang, J., Xie, L., & Yuille, A. L. (2018). DeepVoting: A robust and explainable deep network for semantic part detection under partial occlusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1372–1380). doi: 10.1109/CVPR.2018.00149.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2921–2929). doi: 10.1109/CVPR.2016.319.

Zhou, L., Xiao, Q., Taha, M. F., Xu, C., & Zhang, C. (2023). Phenotypic analysis of diseased plant leaves using supervised and weakly supervised deep learning. *Plant Phenomics*, *5*, 0022. doi: 10.34133/plantphenomics.0022.

Zhou, Z.-H. (2018). A brief introduction to weakly supervised learning. *National Science Review*, *5*, 44–53. doi: 10.1093/nsr/nwx106.

Summary

Agricultural production systems, such as controlled environment agriculture (CEA), are highly labor intensive. Daily maintenance tasks, including harvesting, leaf removal, and plant phenotyping, require selective operations that are still largely performed by human labor. Under increasing pressure to produce more food and a shortage of skilled labor, reliable agricultural production faces growing challenges. Robots and automation offer a promising solution to address these challenges. However, agricultural environments are inherently complex and cluttered. Severe occlusions caused by leaves, stems, and neighboring plants often make the plant parts that need to be manipulated partially or completely invisible to robotic vision systems. As a result, essential information required for subsequent robotic operations is frequently missing or unreliable, leading to task failure.

Active perception, and in particular Next-Best-View (NBV) planning, provides an effective approach to address these challenges. Based on the current observation, NBV evaluates a set of candidate viewpoints according to their information gain (IG) and actively repositions the camera to viewpoints that are expected to provide the most novel information. By doing so, NBV improves the completeness and accuracy of robotic perception under occlusion compared to single-view or fixed multi-view setups. However, most existing NBV methods rely on offline training, while agricultural environments are highly dynamic: plant size and structure change continuously due to growth, and environmental conditions evolve over time. These dynamics limit the applicability of NBV methods trained offline. This thesis addresses this limitation by enhancing implicit NBV methods through online self-supervised learning, enabling NBV policies to continuously adapt to novel environments during task execution. During NBV operation, the system autonomously collects training data, stores them in a buffer, and updates its viewpoint planning policy using newly acquired observations. To support real-world deployment, the thesis further considers key requirements of scalability across different environments, efficiency in NBV execution, and practical feasibility on real robotic systems.

The research begins with a preliminary study of the influence of camera viewpoints on robotic perception accuracy. In Chapter 2, a deep learning–based keypoint detection method combined with point cloud processing is developed for 3D pose estimation of tomato peduncle nodes, which connect the truss to the main stem and are critical for robotic harvesting and phenotyping. Evaluated in a commercial greenhouse, the method achieves high object detection accuracy (AP@0.5 = 0.96) and keypoint

detection accuracy (PDJ@0.2 = 94.31%), with acceptable 3D pose estimation errors. Evaluation across five camera viewpoints clearly demonstrates the strong influence of camera viewpoint on perception accuracy. This study establishes the feasibility of estimating the pose of specific plant parts using deep keypoint detection algorithms and motivates the need for robotic active perception systems.

Building on this insight, Chapter 3 introduces an online self-supervised learning framework for implicit NBV, referred to as SSL-Global-NBV. The method uses deep neural networks to predict IG and enables the system to continuously collect and label its own training data during task execution. A novel IG metric is proposed to compute ground-truth IG directly from the robot' s observations, enabling data annotation and network training in a self-supervised learning manner. Experience replay and weakly supervised learning techniques are incorporated to improve the sample efficiency of the collected training data. Evaluations in simulation and real-world plant reconstruction tasks show that SSL-Global-NBV can continuously adapt to previously unseen plants and environments through online learning, validating its adaptability in dynamic CEA settings. Compared to non-NBV methods, SSL-Global-NBV achieves more efficient reconstruction, while compared to explicit NBV approaches based on ray casting, it achieves over 800× faster IG prediction. Moreover, the weakly supervised learning strategy reduces the need for IG annotations by more than 90% compared to baseline methods relying on offline training data.

Despite these advantages, SSL-Global-NBV has limitations. It relies on a predefined set of viewpoints, which requires prior assumptions about plant size to ensure sufficient coverage. In addition, its global viewpoint selection strategy often results in extensive camera traversal across the entire viewpoint space, leading to large robotic motions and reduced efficiency. To address these limitations, Chapter 4 proposes SSL-Local-NBV, which shifts from global to local viewpoint planning. Instead of selecting viewpoints from a predefined set, SSL-Local-NBV determines the next camera pose through relative translations and rotations from the current pose. To avoid redundant exploration, a View Trajectory Network is introduced to effectively incorporate historical viewpoint information. Experimental results demonstrate that SSL-Local-NBV reduces trajectory length by 56%–70% per reconstruction cycle compared to global NBV methods, with minimal reduction in reconstruction quality. Compared to SSL-Global-NBV, SSL-Local-NBV effectively scales to plants of different sizes. In real-world experiments, it achieves over 80% reconstruction completeness, confirming its practical feasibility.

Both SSL-Global-NBV and SSL-Local-NBV focus on reconstructing the entire scene. This is suitable for tasks such as plant monitoring, where comprehensive scene information is required. However, many agricultural operations require information only

about specific plant parts, such as fruits for harvesting or nodes for leaf removal. Exploring irrelevant regions in such cases wastes time and energy. To overcome this limitation, Chapter 5 introduces SSL-Semantic-NBV, a target-aware NBV approach that prioritizes viewpoint selection for plant parts that are relevant to task execution. Target-aware NBV is considerably more challenging due to target sparsity, as most viewpoints provide little or no useful signal for supervising network training. To address this challenge, a novel IG metric and loss function are proposed to enable effective learning from sparse information signals. To accelerate real-world deployment, functional–structural plant models are used to generate synthetic data for network pretraining. In simulation, SSL-Semantic-NBV improves reconstruction efficiency by 42%–104% compared to non-NBV methods and by 9%–18% compared to existing NBV approaches. In real-world experiments, improvements of 8%–16% and 3%–7% are achieved, respectively. The semantic-aware mechanism substantially improves both training efficiency and reconstruction efficiency, particularly for sparsely distributed plant parts.

In summary, this thesis presents a series of online self-supervised learning–based implicit NBV methods that progressively address key challenges in robotic active perception for agriculture. By integrating online learning, local viewpoint planning, and target-aware perception, the proposed methods improve the adaptability of implicit NBV to novel environments, its scalability across different plant sizes, and its efficiency in perceiving task-relevant plant parts. These methods therefore enhance robotic perception systems to robustly and efficiently collect information under occlusions. By advancing robotic perception capabilities, this work can facilitate the development and deployment of robots in agricultural applications, helping to alleviate labor shortages and contribute to reliable food production.

中文摘要

农业生产系统，尤其是受控环境农业（controlled environment agriculture, CEA），具有高度劳动密集的特点。采收、打叶和作物表型测量等日常管理任务通常需要有选择性的精细操作，而目前这些工作仍在很大程度上依赖人工完成。在农业增产压力持续加大、熟练劳动力日益短缺的背景下，农业稳定生产正面临越来越严峻的挑战。机器人与自动化技术为应对这些挑战提供了有前景的解决方案。然而，农业环境通常复杂且杂乱，叶片、茎秆以及相邻植株造成的严重遮挡，常常使机器人视觉系统难以完整观测需要操作的植物结构，甚至完全无法观测。其结果是，后续机器人作业所需的关键信息往往缺失或不可靠，进而导致任务失败。

主动感知，尤其是下一最佳视角（Next-Best-View, NBV）规划，为解决上述问题提供了一条有效途径。NBV 方法基于当前观测结果，按照信息增益（information gain, IG）对一组候选视角进行评估，并主动将相机移动到预期能够提供最多新信息的视角。与单视角或固定多视角方案相比，NBV 能够在存在遮挡的情况下提高机器人感知的完整性与准确性。然而，现有大多数 NBV 方法依赖离线训练，而农业环境具有很强的动态性：植物的大小和结构会随着生长持续变化，环境条件也会随时间不断演化。这种动态特征限制了离线训练 NBV 方法在实际农业场景中的适用性。针对这一局限，本论文通过引入在线自监督学习来增强隐式 NBV 方法，使 NBV 策略能够在任务执行过程中持续适应新环境。在 NBV 运行过程中，系统自主采集训练数据并存储到数据缓冲区中，然后根据新采集到的数据持续更新视角规划策略。为了实现该方法在真实环境中的部署，论文进一步考虑了其在不同环境下的可扩展性、NBV 的执行效率以及在真实环境中部署的可行性。

本研究首先围绕相机视角对机器人感知精度的影响开展基础研究。第 2 章提出了一种结合深度学习关键点检测与点云处理的番茄花序梗节点三维位姿估计方法。花序梗节点是连接果穗与主茎的结构，也是机器人采收和表型分析任务中的关键目标。该方法在商业温室中进行了评估，取得了较高的目标检测精度（AP@0.5 = 0.96）和关键点检测精度（PDJ@0.2 = 94.31%），同时三维位姿估计误差也保持在可接受范围内。基于五个相机视角的实验结果清楚表明，相机视角的差异会显著影响感知精度。该研究验证了利用深度关键点检测方法估计特定植物部位位姿的可行性，也进一步表明了发展机器人主动感知系统的必要性。

在此基础上，第 3 章提出了一种用于隐式 NBV 的在线自监督学习框架，称为 SSL-Global-NBV。该方法利用深度神经网络预测信息增益，并使系统能够在任务执行过程中持续采集和标注训练数据。论文提出了一种新的 IG 度量方法，可直接根据机器人自己的观测计算真实信息增益，而不需要人工标注，使机器人能够以自监督的方式完成数据标注和训练。为提高所采集数据的样本利用效率，文章进一步引入了经验回放和弱监督学习策略。仿真与真实植物重建任务的结果表明，SSL-Global-NBV 能够

通过在线学习持续适应此前未见过的植物和环境，验证了其在动态 CEA 场景中的适应性。与非 NBV 方法相比，SSL-Global-NBV 实现了更高效的重建；与基于射线投射的显式 NBV 方法相比，其 IG 预测速度提高了超过 800×。此外，相较于依赖离线训练数据的基线方法，弱监督学习策略使其对 IG 标注数据的需求降低了 90% 以上。

尽管 SSL-Global-NBV 具有上述优势，但仍存在一定局限。该方法依赖预定义的视角集合，因此需要预先对植物尺寸作出假设，以保证视角覆盖范围。此外，其全局视角选择策略通常会导致相机在整个视角空间内进行大范围移动，从而增加机器人运动距离并降低执行效率。为解决这些问题，第 4 章提出了 SSL-Local-NBV，将视角规划方式由全局选择转变为局部规划。SSL-Local-NBV 不再从预定义视角集合中选择下一视角，而是基于当前相机位姿，通过相对平移和相对旋转来确定下一个相机位置。为避免冗余探索，论文进一步引入了视角轨迹网络（View Trajectory Network），以有效利用历史视角信息。实验结果表明，与全局 NBV 方法相比，SSL-Local-NBV 在每轮重建过程中可将轨迹长度缩短 56%–70%，同时仅带来很小的重建质量下降。与 SSL-Global-NBV 相比，SSL-Local-NBV 能够更有效地适用于不同尺寸的植物。在真实实验中，该方法实现了超过 80% 的重建完整度，验证了其实际应用的可行性。

SSL-Global-NBV 与 SSL-Local-NBV 均以重建整个场景为目标，这对于植物监测等需要完整场景信息的任务是合适的。然而，在许多农业作业中，机器人只需要感知特定植物部位的信息，例如采收任务中的果实或打叶任务中的节点。在这种情况下，对无关区域的探索会造成时间和能量的浪费。为克服这一局限，第 5 章提出了 SSL-Semantic-NBV，这是一种面向目标的 NBV 方法，在选择视角时给予与任务相关的植物部位更高的优先级。由于目标分布稀疏，面向目标的 NBV 在神经网络训练中更具挑战性，因为大多数视角几乎无法为网络训练提供有效的监督信号。为解决这一问题，论文提出了新的 IG 度量和损失函数，以实现对稀疏信息信号的有效学习。为加快真实场景中的部署，论文利用功能结构植物模型生成合成数据，对网络进行预训练。在仿真中，与非 NBV 方法相比，SSL-Semantic-NBV 将重建效率提高了 42%–104%；与现有 NBV 方法相比，提高了 9%–18%。在真实实验中，相应提升分别为 8%–16% 和 3%–7%。语义感知机制显著提高了训练效率和重建效率，尤其适用于目标植物部位稀疏分布的情形。

综上所述，本论文提出了一系列基于在线自监督学习的隐式 NBV 方法，目标是逐步解决农业机器人主动感知中的遮挡问题。通过融合在线学习、局部视角规划和面向目标的感知机制，所提出的方法提升了隐式 NBV 对新环境的适应能力、对不同植物尺度的可扩展性，以及对任务相关植物部位的感知效率，从而增强了机器人感知系统在遮挡条件下稳健、高效获取信息的能力。通过推动机器人感知能力的发展，本研究有望促进机器人在农业场景中的开发与应用，缓解劳动力短缺问题，并为稳定可靠的粮食生产提供技术支持。

Acknowledgements

I was born in a village in Shouguang, Weifang, Shandong Province. My hometown is known for its endless solar greenhouses and for supplying the largest share of China's vegetables, as well as exporting large quantities abroad. Although my family did not run a greenhouse, from as early as I can remember, working in greenhouses was part of the daily life of almost everyone around me. In the morning, people from the village would leave for the greenhouses, and in the evening, they would come back home after a day of work. Greenhouse production was the most common topic people talked about, and it was also one of the main activities that shaped everyday life around me. With my limited understanding as a child, I thought that vegetables were naturally grown only in greenhouses. My interest in greenhouse cultivation started from that time, together with a simple belief that improving greenhouse production technology was a responsibility shared by everyone.

When I was applying to universities abroad for a master's programme, I did not even know which keywords I should use to search for a programme that matched my interests. Then, I accidentally came across several photos showing examples of precision agriculture from the Farm Technology Group at Wageningen University & Research, the former name of my current group, Agricultural Biosystems Engineering. At that moment, I became completely certain that this was where I wanted to go.

Unfortunately, I failed in my first application, because of my English examination result. I have never been a person with a talent for languages. Fortunately, I was accepted by another university located on a picturesque island, surrounded by sunshine, long sandy beaches, endless pastures, flocks of sheep and herds of cows, and supported by a full scholarship. However, I always knew that I still held the dream of pursuing the knowledge and training that I truly wanted. Finally, one year later, after my second application, I was accepted. Without hesitation, I came to Wageningen and gave up everything I had there.

I still remember my first day walking around the Wageningen campus. It was February, cloudy and rainy. I stood alone, looking around and exploring the university by myself. At that moment, I could never have known that this was the small city in which I would live for more than seven years. The master's courses were not easy for me. I was surprised to find that one remarkable difference was that the courses here were very good at using mathematics and programming to explain almost everything. I fully agreed with this approach, and I still believe that this is how knowledge should be taught and applied. However, my limited mathematical background and complete

lack of programming experience meant that I had to spend almost all my weekends and many evenings studying, often until late at night. I never gave up because, every day, I could feel myself becoming a little stronger.

Finally, on the day I graduated from my master's programme, I was full of confidence. I believed that I had gained advanced knowledge in this field and the ability to solve problems independently. This confidence lasted until I started my PhD. During my PhD journey, I encountered many challenges, including academic writing, presentations, critical thinking, collaboration with others, and developing the ability to approach research problems independently. Step by step, I faced these challenges and eventually reached the stage of completing my PhD. Along the way, I received tremendous help from my supervisors, colleagues, friends, and family. I would like to thank them here.

Eldert, I am deeply grateful for the training you gave me in scientific writing, critical thinking, presentation, and the evaluation of existing works, from both a deep and practical perspective. This learning process was not always easy. At times, I even found it challenging and confusing. However, looking back now, what remains is my sincere appreciation. Many times, when I started working on a new topic, I realized that the training you had given me was exactly what helped me develop the capabilities needed to become a qualified researcher. I am grateful for the time, patience, and effort you invested in my development. What I learned from you will continue to support my future work, whether in academic research or in industry.

Gert, the first course I took after arriving in Wageningen was your machine learning course. I truly enjoyed it, although at that time I had no idea that I would spend most of my weekends studying for it. I also did not know that this would be the beginning of a working relationship that would last seven years. I would count my MSc period as part of this journey because, from the beginning of my master's programme, you were involved in almost every programme period. You later supervised my MSc thesis as an examiner, then became my PhD supervisor, and now also my supervisor for the postdoctoral project. We have never needed many conversations beyond work, but I am certain that there is a quiet and unspoken understanding between us. Whenever I encountered difficulties or felt uncertain about how to move forward, you somehow appeared at exactly the right moment and helped me. You are always passionate, energetic, and remarkably capable of identifying the key problems in a piece of work and immediately offering answers that are correct, or very close to correct. Therefore, I never doubt that a paper you agreed to submit deserves acceptance. From you, I learned what a good researcher should be capable of. I am sincerely grateful for your guidance and for all the help you have given me throughout these years.

Xin, you were the supervisor of my MSc thesis. I often think that if you had not been willing to supervise me at that time, I might never have entered the field I am in now: computer vision, robotics, and greenhouse applications. This is a direction that I truly enjoy and hope to continue pursuing in the future. Later, you also became one of my PhD supervisors. Throughout these years, you always gave me a great deal of trust and freedom to explore the questions that interested me. You also kept an eye out for new opportunities and actively shared them with me, which I deeply appreciate. In many ways, you helped open the door to the research path that I am still following today.

I did this research at the ABE group, where many wonderful colleagues supported me and helped create an inspiring environment for doing science. Many of you have become friends rather than just colleagues. Thank you all for your company, support, and for the many happy moments that would not have happened without you.

Laiquan, I am very glad that I could spend almost five very happy years with you. You have always been reliable, always willing to help me and others, and always ready to take care of everyone's small requests. When I think back to the first two years after I joined the group, I miss our regular weekly hotpots, parties, and drinks we had together. Although you did not drink, you were always the one who cleaned up afterwards and clearly remembered what everyone was like after getting drunk. I still miss those days of all of us talking late into the night, together with **Yuxiang**, **Mengting**, **Xiefei**, **Zhenpeng**, and **Junhan**, squeezed into my small room, even during 37°C summer days. Although the final stage of a PhD may not be the easiest time to enjoy, I sincerely hope that you can still enjoy this last part of the journey. **Yuxiang**, I miss the days when we worked hard together. You always motivated me to become fully immersed in my work. **Mengting**, you always cared about everyone and helped bring people together, which also helped me to quickly become part of the team. **Xiefei** and **Zhenpeng**, you always showed the happy side of life to others, which helped me step away from stressful plans and gave me encouragement when I needed it. My biggest regret is that you no longer seem to be seeking happiness in alcohol. **Junhan**, you have always been full of energy to explore new things, and that has also encouraged me a lot. I believe that sunny days will eventually come, and we will eventually reach the place our hearts long for.

Xindong, you have always felt like a calm and emotionally stable older brother, while still being full of energy to inspire others. The muscles you casually showed from time to time encouraged many people to start going to the gym, although most of us stopped after you left. You always took care of everyone's feelings, and you were always willing to prepare a table full of excellent food to host a perfect party.

Ziye, if Wageningen University were to hold a competition and offer an award for "the most disciplined person," I'm sure you would win it. I have always been amazed by your routine: waking up regularly at 5:30, carefully planning your day, swimming, running, going to the gym, and training your body as if it were made of iron. You have also always shown great enthusiasm for new technologies. I truly enjoyed all our discussions, because you always brought insights from different perspectives, including research, applications, companies, and society.

Akshay, our friendship began during my MSc thesis, when you were one of my supervisors. Later, when I started my PhD, we became colleagues and friends. Our PhD topics shared many similarities, and I received tremendous help from you throughout the process. I always enjoyed discussing research, life, and the future with you. You have always been patient, kind, and willing to share your thoughts and support. I am certain that, without your help, my PhD project would not have progressed as smoothly as it did. **David**, thank you so much for all the help you have given me since I started writing my master's thesis. You were always able to quickly and accurately identify problems in my experiments and give me helpful advice. Through this process, I received excellent training, which later helped me apply for a PhD position. During my PhD, I have also greatly enjoyed our discussions. You always had a sharp eye for the latest developments and emerging trends in the field, and our discussions often helped me think more deeply about my research and improve the direction of my work.

Chao, you always seemed able to make perfect plans for everything, both in research and in life. You worked with great efficiency and solved problems step by step according to your plan. Whenever I had questions, you were always able to give me direct and practical advice, which helped me find reasonable solutions. **Jingxin**, you were a pleasant surprise in my PhD life. During your one-year visit from Beijing to Wageningen as a visiting researcher, you became my office mate and friend. I truly enjoyed discussing with you the differences in research and work across different countries. You have always been full of passion for exploring new things. Thank you for the guidance, support, and encouragement you gave me in both my research and work. **Shuyi** and **Jinrui**, you are very good listeners. I always enjoyed talking with you, and our conversations gave me encouragement again. **Robert** and **Rick**, thank you for always being willing to discuss with me when I encountered problems in my experiments, and for providing valuable suggestions. **Axel**, I am very glad that I could complete the final chapter of my PhD together with you. Thank you for providing your unmatched voice for the video, which helped make the submission stronger and contributed to its acceptance. **Maedeh** and **Bart**, I am very happy to have had you as my office mates. You were always full of energy and enthusiasm, and I always

enjoyed discussing our progress with you. **Cecile**, **Homa**, **Lisa**, and **Salma**, I am very glad to have heard the interesting stories you shared. **Shutong**, you are always gentle yet strong. You show kindness to others, while also being able to face and solve all kinds of difficulties yourself. **Manuel**, I am very glad to have met you and to have played volleyball with you. Thank you for organizing everyone and helping me have such a good time through sports. I hope we will have the chance to play volleyball together again in the future. **Wentao**, I still miss our regular dinners, beers, and the conversations we had about society, history, and research. I always enjoyed listening to your thoughts and perspectives. **Yujian**, your curiosity and willingness to explore new things have always inspired me. **Thomas** and **Ashutosh**, you have always been calm and emotionally stable, and you steadily carried out your work with consistency and focus. **Stijn**, I am very glad to have you as a great office mate. I always enjoyed discussing the latest progress in our work with you, and I especially appreciated your genuine concern for my fish. **Alberto**, thank you for your active help and support in the collaborations related to the robotic arm and gripper course. **Michele**, you are an optimistic person who truly enjoys life, and I often found encouragement and positive energy in our conversations. **Xiaodong**, you have always been kind and willing to help. You could quickly understand the problems I encountered and immediately offer clear and concise advice. I am also grateful that you often shared potential career opportunities with me. **Congcong**, I am very glad to have heard the many interesting experiences and stories you shared. Thank you as well for thinking of me and sharing potential job opportunities with me. **Katarina**, I am very happy to have had you as a co-author on the final chapter of my PhD thesis. You have been a wonderful collaborator, reliable, patient, and always willing to contribute. I truly appreciated working with you and hope we will have the opportunity to collaborate again in the future. **Bolai**, I greatly miss our discussions about state-of-the-art methods in deep learning, reinforcement learning, and computer vision, as well as their practical implementation. You always offered valuable advice and inspired me with new ideas. **André**, I am very glad to have had the opportunity to collaborate with you in supervising students. You always encouraged us to explore the directions that interested us, while providing key insights whenever we encountered difficulties. **Yongran**, I am very grateful for all your help, especially your thoughtful advice regarding my future career development. **Peige**, **Zhibo**, **Zixin**, **Xin**, **Tianyi**, I am very glad to have met you all and truly enjoyed our many interesting conversations.

Beyond research, a PhD inevitably involves many practical matters, including conferences, reimbursements, hardware, software, and administrative issues. As an international student, I also had to deal with additional challenges, such as visa-related problems. These processes would not have gone nearly as smoothly without your support. **Miranda**, **Sam**, **Areerat**, and **Paul**, thank you for always being patient and

for helping me solve problems quickly and efficiently. Your support greatly minimized the time and effort I had to spend on these matters, allowing me to focus more on my research.

I would also like to thank some dear friends who came into my life, helped me temporarily set aside the pressure of PhD work, and brought me endless happiness. **Xuchuan**, I have always been amazed by your curiosity and broad interests in almost every field. I enjoyed having deep conversations with you on many different topics. **Qing**, you are brave in trying new things and always managed to plan everything. **Xiaofang**, you always cared about everyone's feelings and made people feel comfortable and included in the team. **Qing**, even under great pressure, you always managed to bring happiness to others. **Zekun**, you are such a refined and considerate person, able to take good care of both your own life and the people around you. **Yifan**, you are always full of energy, filling your life with richness and bringing surprises to both yourself and your friends.

最后，我想感谢我的家人。爸爸，妈妈，感谢你们始终如一的爱与支持。自从我开始在国外学习，从硕士到博士，你们一直给予我无限的信任，尊重并支持我的每一个选择。在压力很大或感到疲惫的时候，我总会给你们打电话，而每一次通话结束后，我都能重新获得力量，继续向前走。你们的爱一直是支撑我前进的动力，也让我知道，无论什么时候，只要我想停下来，身后总有人可以依靠。也感谢我的哥哥和弟弟。在我离家很远的这些年里，是你们替我照顾家人，让我能够安心地追求自己的目标。

To my beloved girlfriend, **Dan**, I often feel deeply grateful and lucky that I met you. Whenever I saw your smiling eyes or received your hug, all my tiredness seemed to disappear. You have always understood me, supported me, and quietly arranged our life when I was overwhelmed by work. Your love has given me warmth, peace, and courage throughout this journey. I am certain that we still have a very long way to walk together.

I am shaped by every place I have been, every challenge I have faced, every person I have met, every kindness I have received, and every moment of doubt, joy, love, and pain along the way. Together, they have made me who I am today. For all of this, and for everyone who has been part of this journey, I am deeply grateful.

Many thanks to everyone.

2026.09.02
Jianchao Ci
慈建超

About the author

Jianchao Ci was born on January 8, 1994, in Daotian Town, Shouguang City, Shandong Province, China. Growing up in Shouguang, a region well known for greenhouse vegetable production in China, he developed an early interest in horticulture, which later strongly motivated his work on robotics for greenhouse applications. In September 2013, he enrolled at Shandong Agricultural University, where he pursued a bachelor's degree in Agriculture Facility Science and Engineering and graduated in July 2017.

He continued his studies at Wageningen University, where he started the MSc programme in Biosystems Engineering in February 2019 and was awarded the degree of Master of Science (MSc) in February 2021. From May 2021 to November 2021, he worked as a computer vision engineer at Rokae, a robotics company in Beijing, China. In December 2021, he started his PhD at the Agricultural Biosystems Engineering Group, Wageningen University & Research, the Netherlands, supported by a China Scholarship Council (CSC) scholarship. His PhD research focused on active perception for agricultural robots, with particular emphasis on robotic online self-supervised learning. The results of his work are presented in this thesis.

Peer-reviewed journal publications

Ci, J., Wang, X., Rapado-Rincón, D., Burusa, A. K., & Kootstra, G. (2024). 3D pose estimation of tomato peduncle nodes using deep keypoint detection and point cloud. *Biosystems Engineering*, 243, 57–69. doi: 10.1016/j.biosystemseng.2024.04.017

Ci, J., van Henten, E. J., Wang, X., Burusa, A. K., & Kootstra, G. (2025). SSL-NBV: A self-supervised-learning-based next-best-view algorithm for efficient 3D plant reconstruction by a robot. *Computers and Electronics in Agriculture*, 233, 110121. doi: 10.1016/j.compag.2025.110121

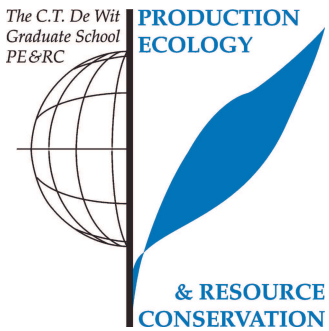
Ci, J., van Henten, E. J., Wang, X., & Kootstra, G. (2025). Robotic self-supervised local view planning for efficient and scalable 3D plant reconstruction across varying plant sizes. *Computers and Electronics in Agriculture*, 238, 110839. doi: 10.1016/j.compag.2025.110839

Conference publications

Ci, J., Smoleňová, K., Wang, X., Streit, A., van Henten, E. J., & Kootstra, G. (2026). SSL-Semantic-NBV: A Self-Supervised Learning-Based NBV for Target-Aware 3D Reconstruction in Agricultural Robotics. In *Proceedings of the 2026 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Accepted for publication.

PE&RC Training and Education Statement

PE&RC Training and Education Statement
With the training and education activities listed below, the PhD candidate has complied with the requirements set by the Graduate School for Production Ecology and Resource Conservation (PE&RC), which comprises a minimum total of 30 ECTS (= 20 weeks of activities).



Review / project proposal (6.0 ECTS)

- Learning-Based Active Perception in Robotics

Post-graduate courses (6.6 ECTS)

- Digital Plant Phenotyping and Agri-Food Robotics Summer School; WUR & Lincoln University (2025)
- Autonomous Greenhouse Challenge, 4th Edition; WUR & Tencent (2024)
- Bayesian Statistics; PE&RC (2023)
- Basic Statistics; PE&RC (2023)

Invited review of journal manuscripts and conference papers (5.0 ECTS)

- *Biosystems Engineering*: Leveraging YOLOv8 for the analysis of social network dynamics in dairy cows within automated milking systems (2024)
- *Computers and Electronics in Agriculture*: Automatic measurement method of sheep body size based on 3D reconstruction and point cloud segmentation (2025)
- *Frontiers in Plant Science*: Missed transplanting rate evaluation method for tobacco seedling (2025)
- *Computers and Electronics in Agriculture*: Branch length recognition and pruning point localization method for greenhouse tomatoes based on improved YOLOv8 (2025)
- IROS 2025 Workshop: Rethinking Viewpoint Planning in Mobile Robot Systems for Greenhouse Monitoring (2025)

Competence, skills and career-oriented activities (3.1 ECTS)

- Scientific Writing; WUR in'to Languages (2022)
- Scientific Publishing; PE&RC (2024)
- Presenting with Impact; WUR in'to Languages (2024)

Scientific Integrity / Ethics in science activities (0.6 ECTS)

- Scientific Integrity; WUR (2022)

PE&RC Annual meetings, seminars and PE&RC weekend / retreat (1.5 ECTS)

- PE&RC weekend for first years (2022)
- PE&RC weekend for midterm PhD candidates (2023)

National scientific meetings, local seminars, and discussion groups (6.8 ECTS)

- Agri-food Robotics Discussion Group (2022–2023)
- ABE Vision and Robotics Thematic Meeting (2022–2025)
- Annual Meeting WUR & Bonn (2022–2025)
- Agri-food Robotics Demonstration Event (2023)
- European Machine Vision Forum event (2023)

International symposia, workshops and conferences (4.8 ECTS)

- EurAgEng; Athens, Greece (2024)
- DigiCrop; remote (2025)
- BrIAS Agri 2025; Brussels, Belgium (2025)

Lecturing / supervision of practicals / tutorials (5.4 ECTS)

- Robotics – arms and grippers (2023)
- Robotics – arms and grippers (2024)
- Robotics – arms and grippers (2025)

MSc thesis supervision (5.0 ECTS)

- Estimating ammonia emissions from the floor in pig houses by using advanced modelling and measuring techniques
- Robotic interactive perception

Total of 44.8 ECTS

Authorship statement

Chapter 1: Introduction

I drafted and wrote this chapter. The text was revised three times following comments and discussions with my promotor, co-promotor, and advisor. ChatGPT[1] was used during the preparation of the draft to correct grammar, improve clarity, and convert the manuscript from Word to LaTeX format.

Chapter 2: 3D Pose Estimation of Tomato Peduncle Nodes Using Deep Keypoint Detection and Point Cloud

I implemented the method presented in this chapter in discussion with the co-authors. The data were collected by me and one co-author in a commercial greenhouse, and we performed the data annotation. I wrote the first draft and revised it based on comments from the co-authors. ChatGPT was used for LaTeX formatting. No AI tools were used for programming.

Chapter 3: SSL-NBV: A Self-Supervised-Learning–Based Next-Best-View Algorithm for Efficient 3D Plant Reconstruction by a Robot

The supervision team proposed the initial concept of applying self-supervised learning to NBV planning. Under their supervision, I implemented the SSL-NBV method and conducted both simulation and real-world experiments. The simulation environment was initially developed by one co-author, after which I extended it and performed the simulation experiments. The simulated plant models were obtained from CGTrader[2] and preprocessed by me. I conducted the real-world experiments in a greenhouse using a robotic arm. The robot control system was configured by the same co-author. I wrote the first draft and revised it after comments from the co-authors. ChatGPT was used for proofreading and LaTeX formatting. No AI tools were used for programming.

Chapter 4: Robotic Self-Supervised Local View Planning for Efficient and Scalable 3D Plant Reconstruction Across Varying Plant Sizes

I implemented the SSL-Local-NBV based on SSL-NBV together with the supervision team and conducted both simulation and real-world experiments. The simulation environment was based on that used in SSL-NBV, with modifications to viewpoint

[1] https://chat.openai.com/

[2] https://www.cgtrader.com/

sampling. Simulated plants were obtained from CGTrader and preprocessed by me. The real-world experiments were conducted using open-source datasets and another dataset collected by one co-author. I wrote the first draft and revised it after comments from the co-authors. ChatGPT was used for proofreading and LaTeX formatting. GitHub Copilot[3] was used as a programming assistant.

Chapter 5: SSL-Semantic-NBV: A Self-Supervised Learning–Based NBV for Target-Aware 3D Reconstruction in Agricultural Robotics

I implemented the SSL-Semantic-NBV method based on SSL-NBV together with the supervision team and conducted both simulation and real-world experiments. I further extended the simulation environment according to experimental requirements. Simulated plants were generated using functional–structural plant modeling software developed by one co-author. The real-world experiments were conducted using open-source datasets, for which I added additional annotations for each node and fruit. ChatGPT was used for proofreading and LaTeX formatting. Codex[4], GitHub Copilot[5], and Gemini 3[6] were used as programming assistants.

Chapter 6: General discussion

I drafted and wrote this chapter. The text was revised twice following comments and discussions with my promotor, co-promotor, and advisor. ChatGPT was used during the preparation of the draft to correct grammar, improve clarity, and convert the manuscript from Word to LaTeX format.

[3]https://github.com/features/copilot

[4]https://openai.com/codex/

[5]https://github.com/features/copilot

[6]https://gemini.google/

This research received funding from the China Scholarship Council (CSC) under grant number 202107720034.

Financial support from Wageningen University and the Agricultural Biosystems Engineering Group for printing this thesis is gratefully acknowledged.

Cover design by Jianchao Ci with assistance from ChatGPT.

Printed by Proefschriftmaken.nl.