Norwegian University of Life Sciences
Faculty of Science and Technology

Philosophiae Doctor (PhD)
Thesis 2020:48

Fusion of a minimalistic set of sensors for mapping and localization of autonomous agricultural robotic systems

Integrasjon av et minimalistisk sett av sensorer for kartlegging og lokalisering av landbruksroboter

Tuan Dung Le

Fusion of a minimalistic set of sensors for mapping and localization of autonomous agricultural robotic systems

Integrasjon av et minimalistisk sett av sensorer for kartlegging og lokalisering av landbruksroboter

Philosophiae Doctor (PhD) Thesis

Tuan Dung Le

Norwegian University of Life Sciences
Faculty of Science and Technology

Ås (2020)

Thesis number 2020:48
ISSN 1894-6402
ISBN 978-82-575-1714-4

To my lovely wife Iris and my little angel Hannah,

Supervisory team

Professor **Pål Johan From** (main supervisor)
Faculty of Science and Technology, Norwegian University of Life Sciences, Norway
Professor **Jon Glenn Omholt Gjevestad** (co-supervisor)
Faculty of Science and Technology, Norwegian University of Life Sciences, Norway

Summary

Robots have recently become ubiquitous in many aspects of daily life. For in-house applications there is vacuuming, mopping and lawn-mowing robots. Swarms of robots have been used in Amazon warehouses for several years. Autonomous driving cars, despite being set back by several safety issues, are undeniably becoming the standard of the automobile industry. Not just being useful for commercial applications, robots can perform various tasks, such as inspecting hazardous sites, taking part in search-and-rescue missions. Regardless of end-user applications, autonomy plays a crucial role in modern robots. The essential capabilities required for autonomous operations are mapping, localization and navigation. The goal of this thesis is to develop a new approach to solve the problems of mapping, localization, and navigation for autonomous robots in agriculture. This type of environment poses some unique challenges such as repetitive patterns, large-scale sparse features environments, in comparison to other scenarios such as urban/cities, where the abundance of good features such as pavements, buildings, road lanes, traffic signs, etc., exists.

In outdoor agricultural environments, a robot can rely on a Global Navigation Satellite System (GNSS) to determine its whereabouts. It is often limited to the robot's activities to accessible GNSS signal areas. It would fail for indoor environments. In this case, different types of exteroceptive sensors such as (RGB, Depth, Thermal) cameras, laser scanner, Light Detection and Ranging (LiDAR) and proprioceptive sensors such as Inertial Measurement Unit (IMU), wheel-encoders can be fused to better estimate the robot's states. Generic approaches of combining several different sensors often yield superior estimation results but they are not always optimal in terms of cost-effectiveness, high modularity, reusability, and interchangeability. For agricultural robots, it is equally important for being robust for long term operations as well as being cost-effective for mass production.

We tackle this challenge by exploring and selectively using a handful of sensors such as RGB-D cameras, LiDAR and IMU for representative agricultural environments. The sensor fusion algorithms provide high precision and robustness for mapping and localization while at the same time assuring cost-effectiveness by employing only the necessary sensors for a task at hand. In this thesis, we extend the LiDAR mapping and localization methods for normal urban/city scenarios to cope with the agricultural environments where the presence of slopes, vegetation, trees render the traditional approaches to fail. Our mapping method substantially reduces the memory footprint for map storing, which is important for large-scale farms. We show how to handle the localization problem in dynamic growing strawberry polytunnels by using only a stereo visual-inertial (VI) and depth sensor to extract and track only invariant features. This eliminates the need for remapping to deal with dynamic scenes. Also, for a demonstration of the minimalistic requirement for autonomous agricultural robots, we show the ability to autonomously traverse between rows in a difficult environment of zigzag-liked polytunnel using only a laser scanner. Furthermore, we present an autonomous navigation capability by using only a camera without explicitly performing mapping or localization. Finally, our mapping and localization methods are generic and platform-agnostic, which can be applied to different types of agricultural robots.

All contributions presented in this thesis have been tested and validated on real robots in real agricultural environments. All approaches have been published or submitted in peer-reviewed conference papers and journal articles.

Sammendrag

Roboter har nylig blitt standard i mange deler av hverdagen. I hjemmet har vi støvsuger-, vaske- og gressklippende roboter. Svermer med roboter har blitt brukt av Amazons varehus i mange år. Autonome selvkjørende biler, til tross for å ha vært satt tilbake av sikkerhetshensyn, er udiskutabelt på vei til å bli standarden innen bilbransjen. Roboter har mer nytte enn rent kommersielt bruk. Roboter kan utføre forskjellige oppgaver, som å inspisere farlige områder og delta i leteoppdrag. Uansett hva sluttbrukeren velger å gjøre, spiller autonomi en viktig rolle i moderne roboter. De essensielle egenskapene for autonome operasjoner i landbruket er kartlegging, lokalisering og navigering. Denne type miljø gir spesielle utfordringer som repetitive mønstre og storskala miljø med få landskapsdetaljer, sammenlignet med andre steder, som urbane-/bymiljø, hvor det finnes mange landskapsdetaljer som fortau, bygninger, trafikkfelt, trafikkskilt, etc.

I utendørs jordbruksmiljø kan en robot bruke Global Navigation Satellite System (GNSS) til å navigere sine omgivelser. Dette begrenser robotens aktiviteter til områder med tilgjengelig GNSS signaler. Dette vil ikke fungere i miljøer innendørs. I ett slikt tilfelle vil reseptorer mot det eksterne miljø som (RGB-, dybde-, temperatur-) kameraer, laserskannere, «Light detection and Ranging» (LiDAR) og propriopsjonære detektorer som treghetssensorer (IMU) og hjulenkodere kunne brukes sammen for å bedre kunne estimere robotens tilstand. Generisk kombinering av forskjellige sensorer fører til overlegne estimeringsresultater, men er ofte suboptimale med hensyn på kostnadseffektivitet, moduleringingsgrad og utbyttbarhet. For landbruksroboter så er det like viktig med robusthet for lang tids bruk som kostnadseffektivitet for masseproduksjon.

Vi taklet denne utfordringen med å utforske og selektivt velge en håndfull sensorer som RGB-D kameraer, LiDAR og IMU for representative landbruksmiljø. Algoritmen som kombinerer sensorsignalene gir en høy presisjonsgrad og robusthet for kartlegging og lokalisering, og gir samtidig kostnadseffektivitet med å bare bruke de nødvendige sensorene for oppgaven som skal utføres. I denne avhandlingen utvider vi en LiDAR kartlegging og lokaliseringsmetode normalt brukt i urbane/bymiljø til å takle landbruksmiljø, hvor hellinger, vegetasjon og trær gjør at tradisjonelle metoder mislykkes. Vår metode reduserer signifikant lagringsbehovet for kartlagring, noe som er viktig for storskala gårder. Vi viser hvordan lokaliseringsproblemet i dynamisk voksende jordbær-polytuneller kan løses ved å bruke en stereo visuel inertiel (VI) og en dybdesensor for å ekstrahere statiske objekter. Dette eliminerer behovet å kartlegge på nytt for å klare dynamiske scener. I tillegg demonstrerer vi de minimalistiske kravene for autonome jordbruksroboter. Vi viser robotens evne til å bevege seg autonomt mellom rader i ett vanskelig miljø med polytuneller i sikksakk-mønstre ved bruk av kun en laserskanner. Videre presenterer vi en autonom navigeringsevne ved bruk av kun ett kamera uten å eksplisitt kartlegge eller lokalisere. Til slutt viser vi at kartleggings- og lokaliseringsmetodene er generiske og platform-agnostiske, noe som kan brukes med flere typer jordbruksroboter.

Alle bidrag presentert i denne avhandlingen har blitt testet og validert med ekte roboter i ekte landbruksmiljø. Alle forsøk har blitt publisert eller sendt til fagfellevurderte konferansepapirer og journalartikler.

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of *Philosophiae Doctor* (Ph.D.) at the Norwegian University of Life Sciences (NMBU).

The research presented here was conducted at NMBU, under the supervision of Professor Pål Johan From and Professor Jon Glenn Omholt Gjevestad during my doctoral study in the period of August 2017 through July 2020.

This thesis is structured as a cumulative thesis, combining previously-published-or-submitted works, which are appended at the end of the thesis.

Acknowledgements

I alone would never be able to finish this thesis, and I would like to thank all those that helped and supported me all along my journey.

First, I want to express my heartfelt thank you to my supervisory team, Professor Pål Johan From and Professor Jon Glenn Omholt Gjevestad for letting me conduct my doctoral study at the Robotic and Control group for the past years. Professor From has been a constant source of advice and it is my privilege to have gotten to work with him. I would like to thank him for the freedom to pursue what excited me for better or worse, and the knowledge, advice, and resources to make those ideas much more appealing. I would like to express my gratitude to Professor Gjevestad for his unwavering support in helping me with equipment, fieldwork and study courses. I would not be able to finish my study requirements without your help. I am eternally thankful for the hours spent with my supervisory team, discussing topics of my research, and how we should address them. Those were some of the most valuable and educational moments of my Ph.D.

I had the distinct pleasure to work directly with many people during this thesis. I would like to thank the SAGA robotics team that helped me with the Thorvald robot, which I used for all of my fieldwork. I would like to specifically thank Marius and Øystein for teaching me mechanic skills that saved me more than once when working out in the field with the robot. My special thanks also go to Ellen, our wonderful project coordinator for handling all the paperwork, setting up workshops and taking good care of our strawberry polytunnel. I'd also like to thank my Ph.D. colleagues, especially Vignesh, for the fun times, good times, and down times together during past years.

And finally, I am forever in debt to my family, my parents, especially my wife Iris and my little angel Hannah. I could not have finished without your constant support and patience throughout this journey, even though it has meant that I am very far from home.

Tuan Le
Ås, July 2020

Contents

Chapter 1

Introduction

Agriculture has been undergoing drastic changes during the past years. The modernization of agricultural product manufacturing has adopted autonomous robots with increasing demands. One of the main reasons for this wide acceptance is the heavy burden of amplifying production in the near future. By 2050, the human population is expected to peak at 9.8 billion according to the United Nations [1]. Food supply security is a non-negotiable matter for each and every country. The agriculture industry may have to double its production to keep up with demands [7]. The industry also faces several problems. Changes in diet in diverse cultures require a high amount of investments in different sectors. Farmland allocated to agriculture worldwide has almost saturated. Resources such as water, energy and greenhouse gas emission constrain place another challenge for agricultural activities. Last but not least, climate change exacerbates and threatens all of the human efforts.

All of those challenges lead to this future of farming and agriculture in general: precision, efficiency, and sustainability. Traditional agricultural machinery simply can not satisfy these requirements. One example is the exceeding usage of pesticide/herbicide has led to exposure impact on human, water and soil contamination, the evolution of pesticide/herbicide-resistant weeds and harmful insects [1]. Agricultural robots equipped with advanced sensing devices, on the other hand, provide a higher level of precision in weed detection and classification [2, 9, 12]. Even more, they offer an alternative physical method to chemical solution for weeding managements [4]. On efficiency, agricultural robots might be able to solve labor shortages and high production costs problems [3]. A particular example is strawberry harvesting. The global strawberry market was reported at 9.2 million tons in 2016, which saw a 5% increase in comparison to 2015. The numbers were collected by the market research company IndexBox. Yet, nowadays most strawberry harvesting operations are done manually by human pickers. As the cost of labor work increases and other factors, such as human pickers can only work in daylight, it is clear that relying on labor forces for harvesting strawberry would soon become cost-prohibitive. Hence, autonomous agricultural robots present a noteworthy solution to the human labor shortage problem in agricultural production.

Agricultural robots have been researched and developed for the past several years. The current market offers different kinds of robots in all shapes and forms. They include both unmanned ground vehicles (UGV) and unmanned aerial vehicles (UAV). While the ground mobile platform is suitable for heavy duties with additional equipment such as plows, arm manipulators, spraying systems,

[1] https://www.un.org/development/desa/en/news/population/world-population-prospects-2017.html

(a) Multipurpose Robotti. Image courtesy of https://agrointelli.com//

(b) Crop monitoring and mapping Tom robot. Image courtesy of https://smallrobotcompany.com

(c) E-series strawberry harvesting robot. Image courtesy of https://agrobot.com/

(d) Scouting UAV. Image courtesy of https://american-robotics.com/

(e) Autonomous tractors. Image courtesy of https://bearflagrobotics.com/

(f) Autonomous weed control. Image courtesy of http://bluerivertechnology.com/

Figure 1.1: Some available agricultural robots.

etc., aerial platforms are mainly used for crop monitoring. Some available agricultural robots are shown in Fig.1.1.

Regardless of the platform and assigned tasks, agricultural robots or autonomous robots, in general, must always be able to answer three questions:

1. What does the world look like?

2. Where is the robot in that world?

3. How can the robot get to its goal in that world?

These questions constitute the fundamental functionalities of an autonomous robot. They relate to the core technologies of modern robots: mapping, localization and motion planning. Mapping and localization provide spatial awareness and knowledge of the global pose to the robotic platforms, a *sine qua non* requirement for motion planning.

This thesis aims to address these questions by designing mapping, localization and planning solutions that work together to allow fast and robust operations on autonomous agricultural robots. Furthermore, we specifically tackle the unique challenges in agricultural environments: sparse features, repetitive patterns, mixed-mode indoor-outdoor activities and more importantly we want to build a minimalistic system that does not incur any heavy cost for agricultural production.

1.1 Motivation and Objectives

Full autonomy has been achieved in several agricultural robotics systems for outdoor environments such as some commercialized products depicted in Fig.1.1. However, agricultural activities are not exclusive to outdoors but also including greenhouses, polytunnels, farmhouses, etc., Most available systems currently depend on GNSS solutions, which is not universal for every task. Hence, a more complex task at hand would require additional sensors, which may, in turn, require a custom-designed solution for each robot. This makes extensions to different platforms difficult. Hence, the main objective of this thesis is to develop autonomous agricultural robots that have such properties:

GNSS-independent operations Since our targeted working environments are a mix of indoor and outdoor scenes, and some may be completely GPS-denied such as polytunnels, we cannot rely on external global positioning. Our goal is to build a pose estimation system that can complement existing GNSS solutions that enable robots to work seamlessly in various scenarios and also work independently.

Minimalistic system Stacking up different sensor modalities may yield better pose estimation. However, it also complicates the whole platform and makes it difficult to extend to other robots. It also places a cost burden on the applicability of a robot. We want to use as few sensors as possible and a common sensor that can be used in various environments should be supported.

Scalability We want our system to apply to general platforms. We want to build a system that can support both ground and aerial robots. Even though the usage of aerial robots in agriculture is still limited, we believe a good system should be able to operate regardless of the platform it is attached to. Therefore, we specifically target 6 degree-of-freedom (DOF) pose estimation, which can be applied to wheeled robots (moving on rough/uneven terrains) and aerial robots. We do not tailor to any specific sensors. All proposed algorithms can work with different brands of sensors (LiDARS, cameras, RGB-D cameras, IMUs, etc.,). We also guarantee the compatibility by developing on Robot Operating System (ROS). ROS is widely adopted and supported by robotics companies around the world. Compatibility with this framework will greatly extend our system usage.

Assistance to human operator Finally, the system should be able to complement and assist human operators if necessary. This is ensured by having map representations that are easy for a human to inspect, infer and intervene so that high-level decision making from a human can be given to robots.

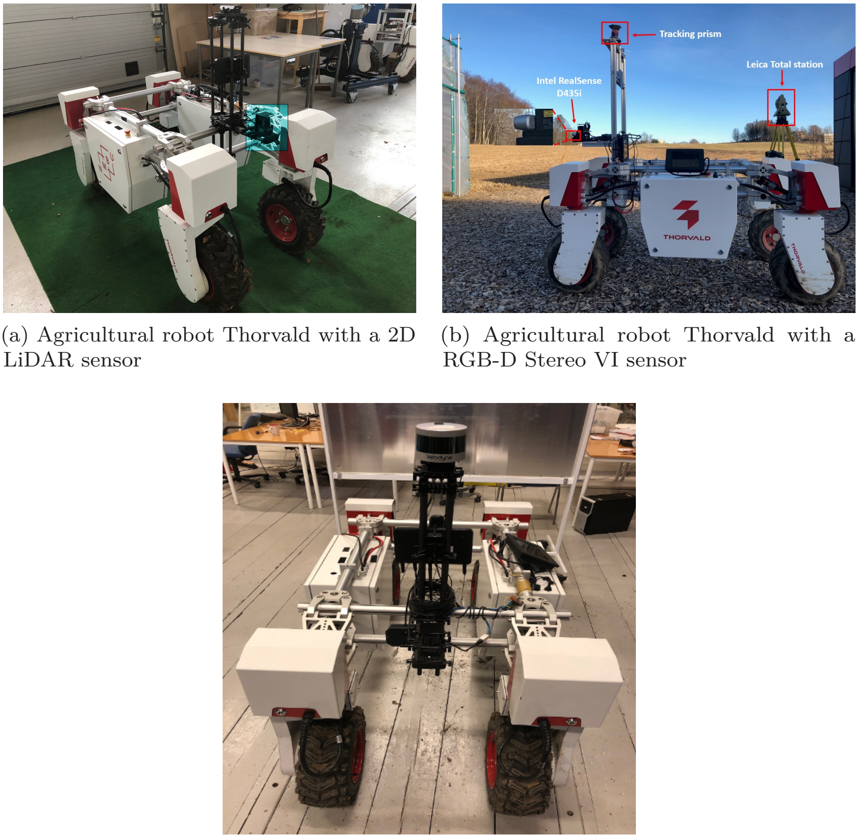
Fig. 1.2 shows examples of three sensor setups on an agricultural robot, Thorvald [8]. We advocate for the selective usage of sensors for specific tasks so that it is unnecessary to stack too many sensor modalities on one robot. Specifically, a 3D LiDAR is suitable for tasks such as a robot moving from an open field to storage on a large-scale farm. For cluttered environments such as polytunnels, we show that we can use only one RGB-D stereo VI sensor for localization.

1.2 Approach

Our approach presents contributions in three main areas: a complete online 3D SLAM system for agricultural robots that can handle sparse features, repetitive patterns problems in agricultural environments, a minimalistic global localization system for agricultural robots in cluttered polytunnel environments, and a simple yet efficient system for autonomous navigation in a challenging polytunnel.

For mapping, we aim to solve a problem of mapping sparse feature and/or repetitive pattern environments, for which, traditional methods of 3D LiDAR mapping such as [14] cannot be applied directly. Moreover, we also want to have an easy-to-read map so that a human operator would not struggle to understand a current situation that a robot is currently in. Octomap [10] is a popular occupancy grid mapping representation. However, it is hard to interpret from a voxel-like map since a voxel obscures fine details of objects it represents. Noted that, by having a detail map does not mean we need to suffer a high-memory consumption for map storage. By leveraging the semantic segmentation of 3D pointcloud [5], we are able to reduce the memory footprint for map storage by 60%, in comparison to state-of-the-art method [14].

For localization, we developed two methods: an NDT-based method [13] to localize a robot in a prior large-scale 3D map, such as a farm, where it is reasonable to assume most features (storage, barns, fences, big trees, etc.,) are static; a machine learning (ML) based approach [11] to extract invariant features from a dynamic environment in a polytunnel with growing strawberry. The



(a) Agricultural robot Thorvald with a 2D LiDAR sensor

(b) Agricultural robot Thorvald with a RGB-D Stereo VI sensor

(c) Agricultural robot Thorvald with a 3D LiDAR sensor

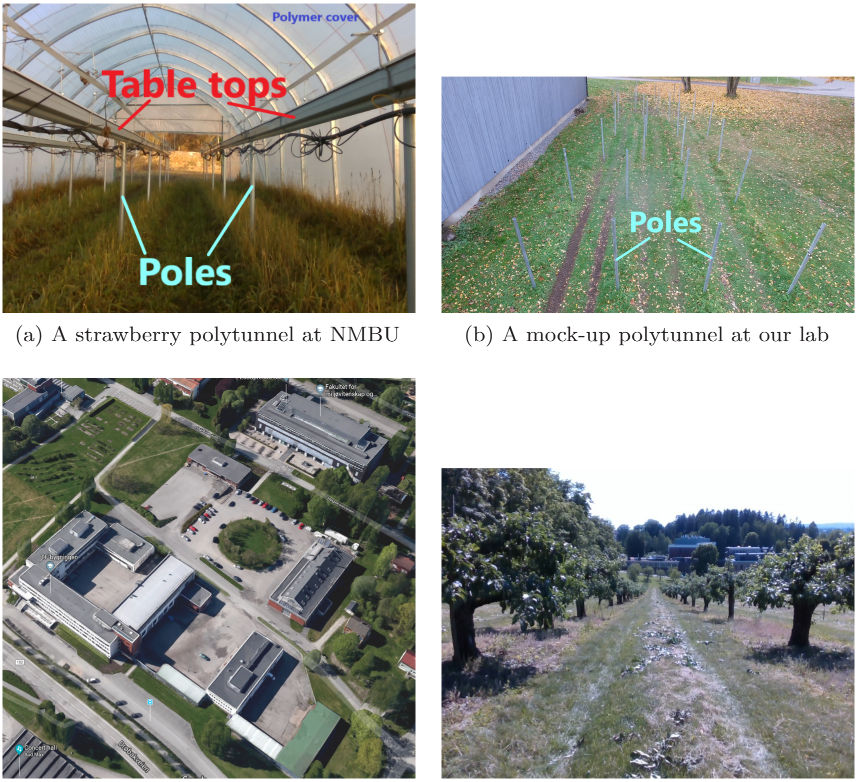
Figure 1.2: Three minimalistic sensor setups used for this thesis.

extracted features helps the robot localize inside a polytunnel, where traditional methods relied on visual features would likely fail.

For navigation, we demonstrated that using only one 2D laser scanner and exploiting the geometric structure of a polytunel, a simple pure pursuit algorithm [6] can safely navigate a robot autonomously between rows in a challenging zigzag-like polytunnel. This also emphasizes our minimalist design strategy.

1.3 Description of experimental environments

For all of our works, we conducted experiments on real-world fields. These fields include a mock-up polytunnel, a strawberry polytunnel, NMBU's campus Ås, and NMBU's orchards. A strawberry polytunnel is a plant-growing area covered by polymer material. A common structure of a strawberry polytunnel consists



(a) A strawberry polytunnel at NMBU

(b) A mock-up polytunnel at our lab

(c) A test area of NMBU's campus at Ås. Image is taken from Google Map

(d) A test area of NMBU's orchards at Ås

Figure 1.3: Environments for experiments used in this thesis.

of several evenly-spaced sets of poles, on top of which hold table-trays. A pole usually has a cylinder shape and made of steel. Poles are firmly inserted into the ground. Strawberry plants are grown in plastic pots and placed on top of those table-trays. An illustration of our strawberry polytunnel at NMBU is shown in Fig. 1.3a. A mock-up polytunnel is an over-simplified version of a polytunnel. A mock-up polytunnel consists only of poles, which are evenly-spaced to mimic a real polytunnel. We built our mock-up polytunnel for fast prototype-testing. Our mock-up polytunnel is shown in Fig. 1.3b. We also use our NMBU's campus at Ås for testing as shown in Fig. 1.3c. A Google Map image of our testing area at NMBU's campus is shown in Fig. 1.3. And finally, we utilize NMBU's orchards for experiments. An image of our orchards is shown in Fig. 1.3d.

References

[1] Aktar, Wasim, Sengupta, Dwaipayan, and Chowdhury, Ashim. “Impact of pesticides use in agriculture: their benefits and hazards”. In: *Interdisciplinary toxicology* vol. 2, no. 1 (2009), pp. 1–12.

[2] Bawden, Owen et al. “Robot for weed species plant-specific management”. In: *Journal of Field Robotics* vol. 34, no. 6 (2017), pp. 1179–1199.

[3] Bechar, Avital and Vigneault, Clément. “Agricultural robots for field operations: Concepts and components”. In: *Biosystems Engineering* vol. 149 (2016), pp. 94–111.

[4] Blackmore, Simon et al. “Robotic agriculture–the future of agricultural mechanisation”. In: *Proceedings of the 5th European conference on precision agriculture.* 2005, pp. 621–628.

[5] Bogoslavskyi, I. and Stachniss, C. “Efficient Online Segmentation for Sparse 3D Laser Scans”. In: *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science* (2017), pp. 1–12.

[6] Conlter, R Craig. “Implementation of the Pure Pursuit Path’hcking Algorithm”. In: (1992).

[7] FAO, Food. *The future of food and agriculture—Alternative pathways to 2050.* 2018.

[8] Grimstad, Lars and From, Pål Johan. “The Thorvald II agricultural robotic system”. In: *Robotics* vol. 6, no. 4 (2017), p. 24.

[9] Hall, David et al. “Towards unsupervised weed scouting for agricultural robotics”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA).* IEEE. 2017, pp. 5223–5230.

[10] Hornung, Armin et al. “OctoMap: An efficient probabilistic 3D mapping framework based on octrees”. In: *Autonomous Robots* vol. 34, no. 3 (2013), pp. 189–206.

[11] Milioto, A., Mandtler, L., and Stachniss, C. “Fast Instance and Semantic Segmentation Exploiting Local Connectivity, Metric Learning, and One-Shot Detection for Robotics ”. In: *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA).* 2019.

[12] Perez, Tristan et al. “A bayesian framework for the assessment of vision-based weed and fruit detection and classification algorithms”. In: ICRA. 2015.

[13] Stoyanov, Todor et al. “Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations”. In: *The International Journal of Robotics Research* vol. 31, no. 12 (2012), pp. 1377–1393.

[14] Zhang, Ji and Singh, Sanjiv. “LOAM: Lidar Odometry and Mapping in Real-time.” In: *Robotics: Science and Systems.* Vol. 2. 2014, p. 9.

Chapter 2

Background

This chapter contains a survey of prior works on robotics mapping and localization and applications for agricultural robots. We start with a review of recent achievements in robotics for mapping and localization, where we also discuss in detail relevant techniques that are similar to what we have developed in our papers. We then focus on sensor fusion applications in agricultural robotics, including notable research projects and relevant applications.

2.1 Robotics mapping and localization

Mapping and localization are the two core functions of autonomous robots. A map can be seen as a distinct representation of an environment. The representation may be in the form of landmark positions, semantic segmentation of objects such as obstacles, walls, ceiling, etc., A purpose of a map is to describe an environment, in which the robot is currently in. It may also provide visualization to a human supervisor so that one can provide high-level commands to a remote robot. Localization, on the other hand, is the robot's capability to recognize where it is in an environment. For the majority of use-cases, knowledge of the precise location of a robot is of utmost importance. For example, robots interact with humans on a factory floor. Usually, a robot is localized inside a map. That map may be *a priori* or may be built online. Due to the complementary nature of mapping and localization processes, they are often combined in a process called Simultaneous Localization and Mapping or SLAM.

There is a vast collection of SLAM work about robotics mapping in literature. Thrun *et al.* [89] provide the classical textbook about mapping among other topics. Durrant-Whyte and Bailey provide a thorough review of probabilistic approaches and data association methods in [25]. More recently, Stachniss *et al.* describe three main SLAM formulations in [85]. Also, other works focus on selective aspects of the SLAM problem. Aulinas *et al.* offer a review of filtering approaches [1]. Grisetti *et al.* discussed about SLAM back-end [37]. Huang and Dissanayake analyze consistency, and convergence problems of EKF-based SLAM in [47]. Scaramuzza and Fraundofer describe advances in visual odometry methods [29, 81]. Saeedi *et al.* discuss the challenges in multi-robot SLAM [79]. Lowry *et al.* focus on the visual place recognition problems [58]. Huang and Dissanayake give an updated review of the theoretical understanding of fundamental SLAM problems in [46].

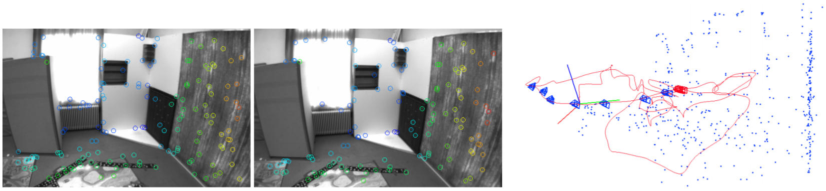


Figure 2.1: An example of a *feature*-based map. Features are represented as 3D points in space (right most image). Those feature are detected and matched between left and right images (left most and middle images).

2.1.1 Map representations

We start by discussing a simple map representation in 2D environments. For this type of environment, one can represent a map as a set of landmark locations or in a form of occupancy grids. The former is trivial to understand, while the latter discretizes the world into cells and each cell is marked by a probability of being occupied. Note that, a landmark-based representation can also be conceived as a feature map. Features can be lines, corners for pure 3D features to other invariant forms detected by a sensor such as a camera or a 3D sensor. Both representations have advantages and disadvantages.

Feature (landmark) maps are mostly sparse and useful for localization by feature (landmark) matching. They are less visually intuitive for environment reconstruction and not easy for human perception. An example of a feature map is shown in Fig.2.1.

On the other hand, occupancy grid maps are more popular. They can represent both static and dynamic environments. An occupancy grid map is originally built in 2D [89], but it can be extended to 3D in a form of *Octomap* [44]. In this thesis, we mainly use the 2D version of an occupancy grid map in Paper III. Hence, we discuss in-depth the 2D occupancy grid map next.

Occupancy grid maps describe a world in a set of cells. Each cell is assigned a number that represents the probability of being occupied. The map then gets updated with new incoming measurements, where the probability of being occupied can be changed. An occupancy grid map is built given known poses of a robot and sensor measurements in those poses. Fig.2.2 shows an example of an occupancy grid map.

The target environment is then categorized into empty space or occupied by obstacles. A probabilistic value of being occupied $p(c)$ is assigned to each cell s. A sequence of observations $z_{1:t}$ obtained from the robot at poses $x_{1:t}$ is used to update the map as the robot traverses the environment. The subscript $1:t$ denotes a time series. The map is then updated by the Bayes' rule:

$$p(c|x_{1:t}, z_{1:t}) = \frac{p(z_t|c, x_{1:t}, z_{1:t-1})p(c|x_{1:t}, z_{1:t-1})}{p(z_t|x_{1:t}, z_{1:t-1})} \tag{2.1}$$

To compute the posterior probability of occupancy, one can assume that all the cells are independent. This allows us to simply formulate the probability of

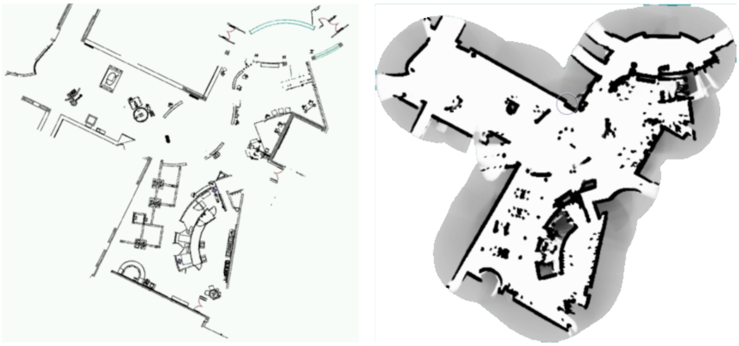


Figure 2.2: An example of an occupancy grid map. *Left*: a floor plan; *right*: a built occupancy grid map of the same floor using 2D range sensor. Black color denotes cells being occupied, white color denotes empty spaces and gray colors mean unknown. Image courtesy of Thrun *et al.* [89].

the map m is a product overall probabilities of individual cells c:

$$p(m) = \prod_{c \in m} p(c) \tag{2.2}$$

Assuming that z_t is independent from $x_{1:t-1}, z_{1:t-1}$, we can rewrite Eq.2.1 as:

$$\begin{aligned} p(c|x_{1:t}, z_{1:t}) &= \frac{p(z_t|c, x_t)p(c|x_{1:t}, z_{1:t-1})}{p(z_t|x_{1:t}, z_{1:t-1})} \\ p(z_t|c, x_t) &= \frac{p(c, x_t, z_t)p(z_t|x_t)}{p(c|x_t)} \end{aligned} \tag{2.3}$$

Besides, we can also assume an arbitrary pose x does not affect the map cells c if there is no observation z at the the same timestamp. It is trivial to make this assumption as we can only infer about the environment only if we have new observation about it. It reflects the fact that if the robot does not move but continue to receive measurements of its current location, other parts of the environment (that are not contained in the current measurements) should not be updated. Hence, we can write:

$$p(c|x_{1:t}, z_{1:t}) = \frac{p(c|x_t, z_t)p(z_t|x_t)p(c|x_{1:t}, z_{1:t-1})}{p(c)p(z_t|x_{1:t}, z_{1:t-1})} \tag{2.4}$$

Each cell of the map is assumed to be either free or occupied, therefore, we can also write the negation of Eq.2.4:

$$p(\neg c|x_{1:t}, z_{1:t}) = \frac{p(\neg c|x_t, z_t)p(z_t|x_t)p(\neg c|x_{1:t}, z_{1:t-1})}{p(\neg c)p(z_t|x_{1:t}, z_{1:t-1})} \tag{2.5}$$

Combining Eq.2.4 and Eq.2.5, we have:

$$\frac{p(c|x_{1:t}, z_{1:t})}{p(\neg c|x_{1:t}, z_{1:t})} = \frac{p(c|x_t, z_t)p(\neg c)p(z_t|x_{1:t}, z_{1:t-1})}{p(\neg c|x_t, z_t)p(c)p(z_t|x_{1:t}, z_{1:t-1})} \tag{2.6}$$

With $p(\neg c) = 1 - p(c)$, Eq.2.6 becomes:

$$
\begin{aligned}
\frac{p(c|x_{1:t}, z_{1:t})}{p(\neg c|x_{1:t}, z_{1:t})} &= \frac{p(c|x_{1:t}, z_{1:t})}{1 - p(c|x_{1:t}, z_{1:t})} \\
= \frac{p(c|x_t, z_t)}{1 - p(c|x_t, z_t)} \cdot \frac{1 - p(c)}{p(c)} &\cdot \frac{p(c|x_{1:t-1}, z_{1:t-1})}{1 - p(c|x_{1:t-1}, z_{1:t-1})}
\end{aligned}
\tag{2.7}
$$

Combining Eq.2.4, 2.5, 2.6, 2.7, we have the occupancy update formula as follows:

$$
p(c|x_{1:t}, z_{1:t}) = \left[1 + \frac{1 - p(c|x_t, z_t)}{p(c|x_t, z_t)} \cdot \frac{p(c)}{1 - p(c)} \cdot \frac{1 - p(c|x_{1:t-1}, z_{1:t-1})}{p(c|x_{1:t-1}, z_{1:t-1})}\right]^{-1} \tag{2.8}
$$

Eq.2.8 shows us how the occupancy probability of a grid cell map is updated given observations. In practice, one can initialize an occupancy prior of 0.5 to all the map cells.

Obvious, to compute the occupancy probability $p(c|x_t, z_t)$, one needs to apply to a specific sensor model that is being used. This model needs to be defined manually for each type of sensor. Interested readers are referred to [89] for more details on sensor models.

Occupancy grid maps enjoy widespread usages for indoor robotics applications due to its simple yet compact representation. However, as the world is not only in 2D, different forms of maps in 3D are required to better describe the environment. Before the arrival of cost-effective 3D LiDAR sensors such as Velodyne VLP-16 [1] and more recently Ouster [2], cameras are the most common sensor that is used for building 3D map, in the form of feature-based maps.

Feature maps as in Fig.2.1 even though they are less perception-friendly, but has been researched for a long time, especially in the computer vision community. The obvious bottleneck of the feature-based map is the computational burden of feature detection, tracking, and matching process. Davidson is the first to present a real-time visual SLAM (VSLAM), called MonoSLAM, in [20]. MonoSLAM uses a mono camera to detect and match sparse keypoints and recovers the scene geometry in an Extended Kalman Filter (EKF) based framework. Civera *et al.* later extend it by including a parametrization in inverse depth in [15]. Klein and Murray paved the way for Parallel for Tracking and Mapping, or PTAM, in [52], which is the first to parallelize the tracking and mapping tasks in different threads, demonstrating the viability of using a bundle adjustment (BA) scheme is to maintain a persistent map. Since then, feature-based mapping attracted a tremendous amount of research in the form of VSLAM problems. LSD-SLAM in [27] is the first direct monocular visual odometry for large-scale environments. LDSO in [31] extend the direct sparse odometry (DSO) [26] by enabling loop closure property. Here, one might notice that the difference between a state estimation problem and a full VSLAM problem is whether or not one includes a loop closure function to achieve a persistent map.

[1] https://velodynelidar.com/

[2] https://ouster.com/

A lot of effort has been put into bridging the gap between computer vision and robotics mapping. For example, Signed Distance Fields (SDFs) have been used extensively for representing 3D volumes in computer vision [30, 32]. They have also been used for offline 3D object reconstructions [18] from the mid-90s. A derived form of SDF, namely Truncated Signed Distance Fields (TSDFs), has spurred a new wave of research with the new RGB-D Kinect sensor and the pioneering work KinectFusion by Newcombe *et al.* [69]. This method enabled real-time running, high-resolution, and accurate 3D reconstructions from an RGB-D camera. It relied on a GPU for fast computation and used TSDF as the main map representation. The authors also introduced a SLAM framework, which estimated the pose of the camera at the same time as reconstructing the scene. Since then, various adaptations have been made to extend this approach. Whelan *et al.* proposed Kintinuous, which allowed scanning much larger spaces in [99]. Steinbrucker *et al.* eliminated the need for GPU for online reconstruction by leveraging an octree-style voxel grid representation. They called it FastFusion [87]. Henry *et al.* used a different 3D representation, called surfels, which are small planar units with size, color, and surface normal), to produce a high resolution online mapping approach using RGB-D camera [42]. Surfel is a volumetric analog to sparse point clouds. ElasticFusion in [98] also used surfel and leverage SLAM techniques for sparse keypoint-based maps to deal with distorting geometry. More recently, Wang *et al.* introduced an online dense surfel mapping for a large scale environment [95]. Most notably, Oleynikova *et al.* introduced Voxblox as a replacement for Octomap, where the authors employed Euclidean Signed Distance Fields (ESDF) for a fast, compact online mapping and planning framework for aerial robots.

In general, feature-based mapping approaches can be considered mature with a long history of success. The current new direction of research for feature-based mapping moves beyond primitive geometric representation to high-level object-based representations or semantic-aware mapping. Early techniques in object-based reasoning were described by Salas-Moreno *et al.* [80] in SLAM++; Civera *et al.* in Semantic SLAM [16] and Dame *et al.* focusing in 3D object shapes [19]. Recently, Grinvald *et al.* introduced a framework built upon *Voxblox* for incrementally building volumetric object-centric maps during online scanning with an RGB-D camera. The built maps contain information about the individual object instances observed in the scene with accurate geometry. The proposed framework can retrieve the dense shape and pose of recognized semantic objects, as well as of newly discovered, previously unobserved object-like instances. The authors assume that the camera pose estimation is provided. Rosinol *et al.* propose a complete semantic-aware SLAM system in [76]. The system is called *Kimera-Semantics*. It includes four key components: a module for visual-inertial odometry estimation (VIO) for camera pose estimation, a module of robust pose graph optimizer for global trajectory estimation, a module of lightweight 3D mesher for fast mesh reconstruction and finally, a module for dense 3D metric-semantic reconstruction.

As shown in Fig. 2.3, semantic-aware mapping offers a magnitude of advantage over classical methods, which are based on primitive geometric

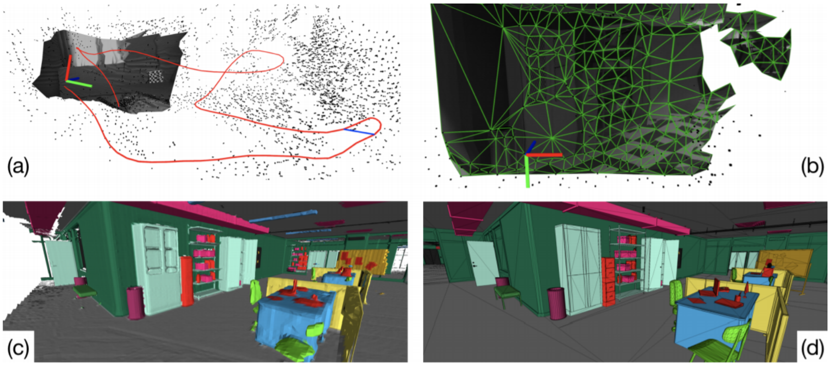


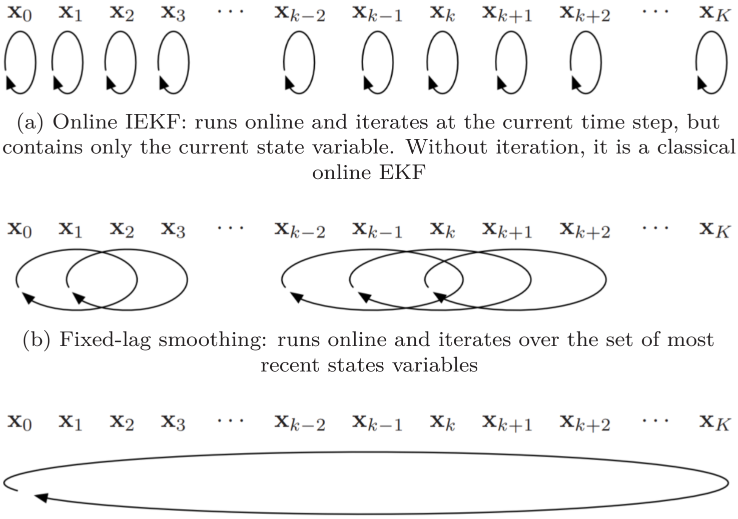
Figure 2.3: An example of Kimera-Semantics: (a) - camera pose estimation; (b) - low latency local mesh of the scene; (c) - a global semantically annotated 3D mesh; (d) - scene ground truth model. Image courtesy of [76].

representations. It enhances robustness for robot operations by moving from path-planning to high level of task-planning. A robot may be able to “perceive” a difference between a wall and a chair, not just a plain classification of “obstacles” as before. Certainly, a high level of task-driven concepts demands more complex semantics concepts. For example, an agricultural robot may be given a task to “going into polytunnel number 3”. The coarse concept of roads, doors might suffice for performance. However, a more complex task such as “harvesting strawberry” demands finer categories of table-top, tray, ripeness, etc., The joint SLAM and semantics inference research have spawned a significant and ongoing body of work in both computer vision and robotics communities.

2.1.2 Sensor fusion for localization

A robot can only perceive its surrounding environment via its sensors measurements. More often, those measurements are noisy and imperfect. Hence, the fundamental challenge is to correctly model and infer from sensory measurements. Khalegi *et al.* categorized sensor imperfections into uncertainty, outliers, conflicting data, correlated errors, and other effects [51]. As mentioned in the previous section, probability theory can be used to deal with uncertainty for localization as well as mapping [89]. Note that, not only a probabilistic theory can be applied for sensor fusion. Murphy proposed Dempster-Shafer evidence theory, which focuses on ambiguity in sensory data [68]. Goodman *et al.* offer a different approach to data ambiguity based on random sets [34]. The probabilistic approach, however, is much more intuitive to understand and easier to implement. Hence, in this thesis, we focus on probabilistic approach for localization in Paper I, Paper II.

Generic sensor fusion has been applied to other fields rather than localization. Munz *et al.* proposed a sensor fusion method for multi-target tracking in [67]. In this thesis, we mainly concern with sensor fusion for pose estimation. Moore


$\mathbf{x}_0$ $\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\cdots$ $\mathbf{x}_{k-2}$ $\mathbf{x}_{k-1}$ $\mathbf{x}_k$ $\mathbf{x}_{k+1}$ $\mathbf{x}_{k+2}$ $\cdots$ $\mathbf{x}_K$
$\mathbf{x}_0$ $\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\cdots$ $\mathbf{x}_{k-2}$ $\mathbf{x}_{k-1}$ $\mathbf{x}_k$ $\mathbf{x}_{k+1}$ $\mathbf{x}_{k+2}$ $\cdots$ $\mathbf{x}_K$
$\mathbf{x}_0$ $\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\cdots$ $\mathbf{x}_{k-2}$ $\mathbf{x}_{k-1}$ $\mathbf{x}_k$ $\mathbf{x}_{k+1}$ $\mathbf{x}_{k+2}$ $\cdots$ $\mathbf{x}_K$

(a) Online IEKF: runs online and iterates at the current time step, but contains only the current state variable. Without iteration, it is a classical online EKF

(b) Fixed-lag smoothing: runs online and iterates over the set of most recent states variables

(c) Offline batch estimation: iterates over all state variables

Figure 2.4: Example of iterative state estimation methods. Image courtesy of [2].

and Stouch proposed an EKF-based for pose estimation in [65]. Ratasich *et al.* introduced a general EKF-based pose estimation using unsynchronized sensors in [75]. Lynen *et al.* proposed a famous generalized pose fusion framework, called Multi-Sensor-Fusion or MSF in [59]. MSF uses an Iterative Extended Kalman Filter or IEKF for state propagation based on IMU measurements. Cucci and Matteucci discussed a multi-sensor pose tracking and calibration in [17]. Their system uses a pose-graph based approach. Recently, a factor graph framework, named GTSAM, has been widely used for pose estimation [21, 83]. Chiu *et al.* model sensor readings as constraints in a factor graph and develop a strategy for selecting a subset of optimal sensor readings in [14]. Hertzberg *et al.* introduce a pose representation on manifold for generic sensor fusion algorithms [43]. Interestingly, on manifold computation attracted subsequent important work for IMU preintegration [28], which in turn, leads to an impressive real-time optimization-based approach for visual-inertial pose estimation [72].

Pose estimation can be related to localization tasks in the sense of a state estimation problem with respect to a reference frame and its uncertainty over time. Filtering-based approaches for pose estimation are the most widespread usage in literature. It follows by smoothing-based approaches. Both directions seek to estimate the maximum likelihood of a state, given noisy sensor measurements.

In Fig.2.4, we illustrate some iterative scheme of state estimation techniques. These techniques along with other variants are discussed in-depth by Barfoot [2]. Interested readers are therefore referred to this excellent textbook for more details. Next, we will briefly discuss filtering-based methods and smoothing-based methods.

Most filtering-based approaches are derived from the Kalman filter (KF). The classical KF consists of two steps: prediction and correction, which can happen in any order. To better model real-world physical properties, the Extended Kalman filter (EKF) is used. The EKF is the extension of KF to a nonlinear system, where the linearization of non-linear observation and dynamic equations about the current state to fit the linear KF. Kubelka *et al.* propose an error state EKF. The filter fuses IMU, wheel encoders, visual odometry, and 2D laser scanner for pose estimation in [53]. Weiss *et al.* introduced an EKF framework for IMU, GPS, and visual odometry fusion for pose estimation in [97]. The state propagation relies on IMU measurements. The authors claim that their filter can keep up to state estimation at a very high rate, approximately 1 kHz, while it also can deal with various time delays between sensors. Note that in case of delayed measurements arrival, one needs to repropagate all affected states. While this is feasible to state vector, it is intractable for the state covariance matrix. The IEKF is another popular variant of EKF, where the prediction and correction steps are iterated until converged to minimize the influence of linearization errors. The MSF framework by Lynen *et al.* [59] was built upon an Iterative EKF (IEKF). Another variant of EKF is the Extended Information Filter (EIF). The EIF is the dual of an EKF, in which the state belief is represented as information vector and information matrix. The textbook by Dan Simon [84] provides more in-depth discussions about KF and its variants. Kalman filter-based approaches have enjoyed several decades of development and implementation. Recently, a drastic change to the EKF structure was introduced by Bourmaud *et al.* [10]. The authors introduce a new EKF on Lie groups, called LG-EKF. Instead of the traditional Euclidean state representation, the state is now projected on Lie groups. State predictions and corrections are now performed on the pertaining Lie algebra. The reason for moving from classical Euclidean space to Lie groups is because robot poses (and landmark poses) in 3D inherently reside on manifolds, specifically a Special Euclidean group or $SE(3)$:

$$\left\{ \begin{pmatrix} R & T \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \right. \quad , \mathrm{R} \in SO(3) \text{ and } \mathrm{T} \in \mathbb{R}^3 \tag{2.9}$$

where R, T is rotation matrix and translation vector, respectively. $SO(3)$ is the Special Orthogonal group that a rotation matrix lives on while the translation vector belongs to the Euclidean space $\mathbb{R}^3$. Details on mathematical operations on groups are beyond the scope of this work. Interested readers are referred to this primer and references therein [7]. Armed with the correct pose representation in Lie groups, LG-EKF is able to respect the geometry of the state space, thus achieving greater estimation accuracy of both the mean and the covariance. A similar approach using the Unscented Kalman filter (UKF) is proposed by Hertzberg *et al.* [43]. It is then followed by the continuous-discrete EKF on Lie groups by Bourmaud *et al.* [9], and invariant filters on Lie groups in [3] by Barrau and Bonnabel. More recently, Ćesić *et al.* introduce Extended Information Filter on Lie groups or LG-EIF (a dual of LG-EKF), which the authors claim that has boosted up the performance of the SLAM framework to the same level as existing state-of-the-art methods.

Note that for all the mentioned method of KF-based pose estimation, one must assume that all state variables lie in a unimodal distribution. The particle filter (PF) resolves this restriction by using a non-parametric distribution representation to approximate the multimodal distributions [89]. The main advantage of PF is that we do not have to worry about linearization errors, while its main drawback is the high dimensionality of the state space. Montemerlo *et al.* propose FastSLAM in [64] using PF for the SLAM back-end and resolve the high dimensional state space by applying Rao-Blackwell marginalization. Mattern *et al.* introduce a PF-based pose estimation system fusing wheel odometry, GPS data, and landmark matching from a camera and a given digital map.

We continue with a discussion about smoothing-base approaches for pose estimation. Similar to filtering-based methods, smoothing-based approaches consider the state as a sequence of state variables and they try to solve the maximum likelihood estimation of the state given noisy sensor measurements. In contrast to filtering-based methods, smoothing approaches use non-linear least square optimization to estimate the state maximum likelihood. The optimization problem may be in the form of a Markov random field or a factor graph. In robotics, factor graphs are used dominantly. As illustrated in Fig.2.4c an optimization problem can consider several measurements at a time for optimization or all the measurements. Obviously, optimize over all measurements - or offline batch optimization can only be done offline. A slightly different version of batch optimization that can be done online, is to continuously add arrival measurements for optimizing. Certainly, in this way, the optimization problem grows unbounded over time and quickly becomes intractable for online computation. One strategy to alleviate the burden of repeatedly optimizing over the whole state vectors whenever a new measurement arrives is to only consider recompute the parts of the state vector that are affected by new measurements. This technique is called *incremental smoothing*. Kaess *et al.* first introduce the incremental smoothing and mapping technique, called iSAM in [49]. Later, they propose an upgrade version called iSAM2 [50], in which a new Bayes tree data structure is used. Indelman *et al.* apply this incremental technique to a factor graph with multiple odometry and pose sources in [48].

Another smoothing method is to only consider a restricted number of variables of a state vector at a time for optimization to bound the computational requirement. This is called a fixed-lag smoothing technique [60]. Note that, this is different from filtering methods. In filtering methods, the state vector is restricted to the most **recent** state, which means all the previous (computed) states are marginalized. This prevents new measurements to change previous states, or it is impossible to relinearize past states. Also, the current state is usually not relinearized and its Jacobians are evaluated only once. This is pointed out by Strasdat *et al.* in [88] as the main reasons why filtering approaches are suboptimal to optimization methods for pose estimations. For fixed-lag smoothing techniques, a state is estimated over a sliding window of time as shown in Fig.2.4b. The number of state vectors in the optimization problem defines the size of the sliding window. One might choose to use the (computed) state vector at the beginning or the end of the sliding window as the final estimation. Note that this smoothing technique is not only for non-linear least square optimization problem. Ranganathan *et al.* in [74] apply the fixed-lag smoothing principle to a forward-backward smoothing EKF [74]. Barfoot considers this similar to an IEKF with an augmented state vector [2]. Indeed, mathematically speaking, iterated prediction, and correction steps of an IEKF are similar to the Gauss-Newton problem [5].

For keeping a fixed size of the sliding window, old states must be marginalized out. One can totally ignore the old states and only integrate new ones. This usually leads to overconfidence. A more common approach is to convert the marginalized states into a prior for the next optimization cycle, equivalently removing old states from the estimation but still keep their information. Marginalization strategy can be either exact or approximate. In computer vision, **Schur complement** are the most commonly used for exact marginalization [90]. In term of probability densities, if we have a joint density $p(x, y)$ over two variables x, y, marginalize out the variable x is equivalent to integrate over x, which gives us a density $p(y)$ over the remaining variable y:

$$p(y) = \int_x p(x, y) \tag{2.10}$$

If the density is given in information form with information vector η and information matrix Λ as follows:

$$p(x, y) = \mathcal{N}\left(\Lambda^{-1} \begin{pmatrix} \eta_x \\ \eta_y \end{pmatrix}, \begin{pmatrix} \Lambda_{xx} & \Lambda_{xy} \\ \Lambda_{xy}^T & \Lambda_{yy} \end{pmatrix}^{-1} \right) \tag{2.11}$$

then the information matrix for y after marginalization is given by the Schur complement of Λ_{xx} in the matrix Λ, which is equivalent to $\Lambda_t = \Lambda_{yy} - \Lambda_{xy}^T \Lambda_{xx}^{-1} \Lambda_{xy}$. Note that Λ_t is the target information prior *after marginalization* for the next optimization cycle. Unfortunately, exact marginalization with Schur complement introduces fill-in, a phenomenon in which the original sparse information matrix becomes dense because of additional non-zero entries in the otherwise zero entries. A dense prior poses as a computational burden to optimization problems. Marginalization hence degrades computational efficiency.

Leutenegger *et al.* [56] and Qin *et al.* [72] work around this issue by selectively discard measurements to keep the sparsity structure of the pose graph. For their keyframe-based visual-inertial odometry optimization methods, namely OKVIS and VINS-MONO, respectively, landmarks that are not observed in the recent frames are marginalized out altogether with the marginalized IMU states. This approximation strategy, while being able to maintain the sparsity of the graph, potentially loses information. The marginalized variables are no longer optimizeable. The solution to the *after marginalization* problem is no longer optimal with respect to the original problem. In order to preserve the marginalized information, Vial *et al.* propose a conservative sparsification scheme for the information matrix [94]. This technique tries to minimize the Kullback-Leibler divergence while enforcing certain edges to be removed. Later, Wang *et al.* formulate the sparsification problem also by minimizing Kullback-Leibler divergence in a lase-based SLAM application [96]. Carlevaris-Bianco *et al.* propose a generic linear constrain which utilizes the Chow-Liu tree to approximate the information of the Markov blanket, which is a collection of incident states variables to marginalized variables. Mazuran *et al.* introduce a general framework called *nonlinear factor recovery* [61], which uses specified nonlinear factors to approximate the dense prior by Kullback-Leibler divergence optimization. This spurred the pioneering work of information sparsification in fixed-lag visual-inertial odometry by Hsiung *et al.* [45]. The authors claim that the proposed method maintains the original visual-inertial odometry problem while preserving most of the information and sparse structure. Most recently, Usenko *et al.* introduce a complete visual-inertial odometry and mapping framework with nonlinear factor recovery [91].

We have surveyed several distinct methods of sensor fusion for pose estimations. Note that, if the pose estimation is relative to some fixed reference frame, this is also a localization problem. A consecutive of estimated poses constitutes the robot trajectory while it is moving. On the other hand, localization also means the ability to recognize *previous* visited places. In a SLAM context, this is a *loop closure* detection capability. Although vision-based methods have advantages in loop closure detection as it is being extensively researched in the computer vision community, they are suffered in degeneration cases such as dark environment or strong viewpoint changes. These disadvantages make vision-only methods less reliable. Hence, we explore LiDAR-based localization methods, which using a high-resolution 3D LiDAR sensor to capture fine details of an environment at a long-range. This is also suitable for agricultural robots where they usually work on large scale environments.

The most classical method for finding a transformation between two LiDAR scans is called iterative closest point (ICP) method [6]. This method tries to align the two point clouds at point-wise level iteratively until converged. Inherently, the efficiency of the ICP method is disproportional to the number of points in the target and source point cloud. Various adaptations to the original ICP method have been proposed. Chen and Medioni introduced the point-to-plane ICP method, which matches points to local planar patches. Segal *et al.* proposed generalized-ICP that matches local planar patches from both point clouds. On computational efficiency, Nücher uses parallelized computation for accelerating scan matching. Qiu *et al.* [73] and Bauer *et al.* [4] use GPU for improving computational efficiency. Moving from the point-wise level, feature-based approaches tackle the alignment problem without demanding high computational resources. Rusu *et al.* proposed Point Feature Histograms (PFH) [78] and Viewpoint Feature Histograms (VFH) [77] as feature detectors. They extract such features from targeted point clouds using simple and efficient methods. Li and Olson propose a Kanade-Tomashi corner detector for extracting general-purpose features from point clouds in [57]. Sefarin *et al.* introduced line and plane features detector from point clouds in [82]. Many techniques for point cloud registration using features have also been studied. Bosse and Zlot in [8] propose a keypoint selection algorithm that calculates point curvature in a local cluster. The selected keypoints are then used for matching and place recognition methods. Steder *et al.* in [86] also select high curvature feature points for matching and place recognition but doing so by projecting the original point cloud onto a range image and analyzing the second derivative of the depth values. Another method exploits the characteristic of a specific environment such as plane dominant to propose a plane-based registration algorithm [35]. A collar line segments method is proposed by Velas *et al.* in [93]. This method randomly generates lines using points from two consecutive "rings" of a scan. This generates two line clouds and can be used for registration. This is illustrated in Fig.2.5. Most recently, Yang *et al.* introduce a fast and certifiably-robust point cloud registration called TEASER++ [101]. It leverages a powerful siamese deep learning architecture and fully convolutional layers 3DSmoothNet [33] to detect correspondences between point clouds. Given those correspondences, an optimization-based method is used to solve a rigid body transformation problem with a reformulated *truncated least square* cost. The authors provide theoretical bounds on estimation errors, hence the certifiably-robust solution.

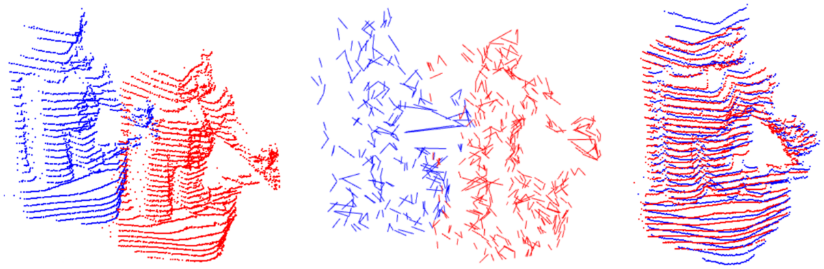


Figure 2.5: An examples of collar line segment registration. *Left* - two unaligned scans, *middle* - sampled by line segments to produce line clouds and *right* - alignment results. Image courtesy of [93].

In a SLAM context, Dubé *et al.* in [24] propose a 3D pointcloud-based modular for place recognition, called *SegMatch*. Incoming 3D LiDAR point

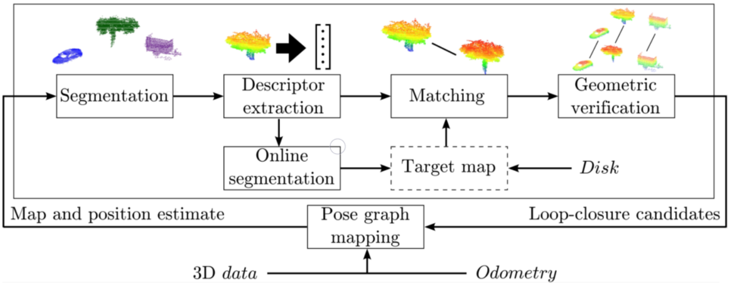


Figure 2.6: Block diagram of Segmatch, a modular algorithm for 3D place recognition. Image courtesy of [23].

clouds are first segmented into distinct clusters. Then for each cluster, a feature vector is calculated based on the cluster eigenvalues and shape histograms. For matching, a random forest algorithm is used to match a source and a target cloud. By matching between consecutive scans, one can estimate a robot's odometry. While matching between a scan and a prior 3D map, one can perform a place recognition for localization and/or for loop closure detections. Zhang *et al.* on the other hand, propose a low drift and real-time LiDAR odometry and mapping (LOAM) framework in [102, 103]. LOAM assigns points to either edge or plane features to find correspondences between scans. Features are calculated based on point "roughness score" in its local region. The points with high roughness scores are classified as edge features, while points with low roughness scores are considered planar features. Real-time performance is achieved by dividing the estimation problem into two sub algorithms. One algorithm runs at high frequency and estimates sensor velocity with low accuracy. The other algorithm runs at low frequency but returns high accuracy motion estimation. The two estimations, fast sensor velocity estimation and slow motion estimation, are fused to produce a single motion estimate at both high frequency and high accuracy.

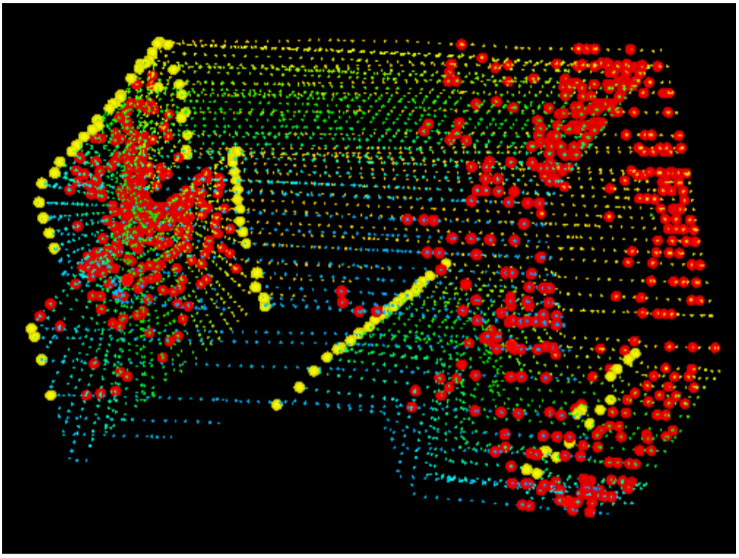


Figure 2.7: An example of extracted edge points (yellow) and planar points (red) from a 3D LiDAR scan. Image courtesy of [102]

2.2 Applications in agricultural robotics

The application of autonomous robotics systems in agriculture has been drastically increased over the past years. Autonomous agricultural robots have shown promising impacts on food security, sustainability, resource use efficiency, reduction of chemical treatments, minimization of human effort, and maximization of yield. As agricultural activities are varied: crop fields, poultry, animal farms, aquaculture, etc., it is impossible to survey all aspects of applications of autonomous robots in those fields. Therefore, we will focus only on the following areas, where we believe the usage of autonomous robots is demanding: *crop monitoring, weed detection, and harvesting.*

Using images for crop monitoring has been studied for decades. Moulin *et al.* in [66] use multi-spectral and hyper-spectral satellite images for field monitoring. However, relying on satellite images are unreliable as they are limited in term of resolution as well as coverage in space and time. One solution is to manually collect color and spectral images from aircraft flying at low altitudes. Nonetheless, this method shares some similar problems satellite images as images are infrequently updated and do not provide important information such as crop height, maturity, yield estimation, etc., Recently, Carlone *et al.* propose a 4D crop analysis framework in [11]. The system provides a 3D reconstructions of a crop field with the ability to associate other data such as shape and plant appearance overtime into one unified 3D map. The system does not only provide a 3D view but also information on plant growth. Potena *et al.* introduce AgriColMap, which is an aerial-ground collaborative 3D mapping solution for precision farming in [71]. The system relies on both ground and aerial robots for 3D mapping of the environment. An effective map registration pipeline that averages a multimodal field representation and casts the data association problem as a large displacement dense optical flow estimation is developed. An example of such maps is shown in Fig.2.8. Chebrolu *et al.* tackle the visual ambiguity problem of strong appearance changes in crop fields for robust long term mapping registration in [12]. The authors propose a scale-invariant, geometric feature descriptor that encodes the local plant arrangement geometry and uses these descriptors for image registrations.

We continue our review on crop and weed detection. Given a detailed map of a crop field, the next step is to efficiently monitor it, namely, we must be able to classify weeds from crops for suitable treatments. For classification problems, a majority of solutions leverage the recent power of machine learning/deep learning (ML/DL) techniques. Lottes and Stachniss propose a semi-supervised online approach using camera images for classification of crops and weeds by exploiting additional arrangement information of the crops in order to adapt the visual classifier. Later, they present a convolutional neural network (CNN) for plant classification in [63], called *Bonnet.* The network is able to segment plants, weeds, and background using only RGB images. Similarly, Sa *et al.* propose *weedNet*, a fully convolutional neural network (FCN) for image classification to detect weeds from aerial images. The widespread usage of ML/DL techniques demands a huge amount of labeled data for training the networks. Di Cicco *et al.* propose an automatic, model-based dataset generation solution in [22]. The proposed model can generate large synthetic training datasets by randomizing the key features of the required environment. The authors model a leaf of a selected plant by means of kinematic chains that span from the stem toward the leaf's principal veins, applied over RGB textures taken from real-world plant

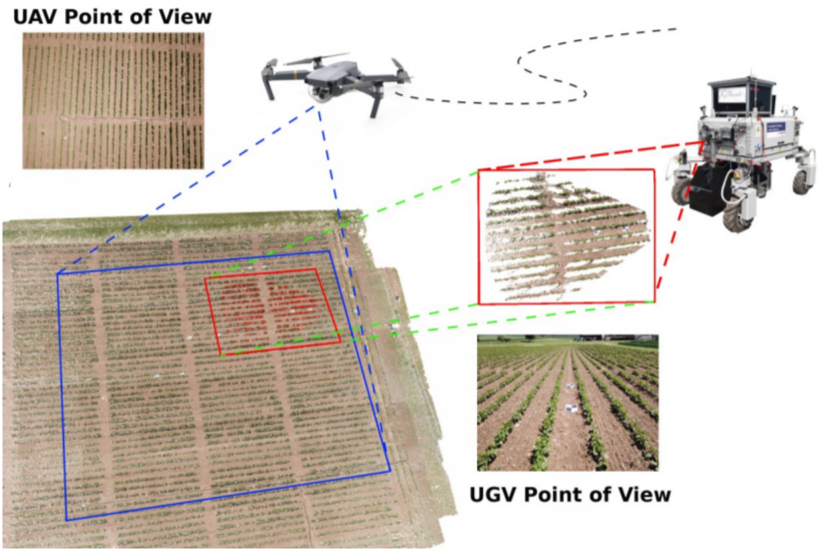


Figure 2.8: An example of AgriColMap, an aerial-ground collaborative mapping solution for precision farming. The final map (left corner) is merged from an aerial view (blue area) and ground view (red area) via an affine transformation. Image courtesy of [71].

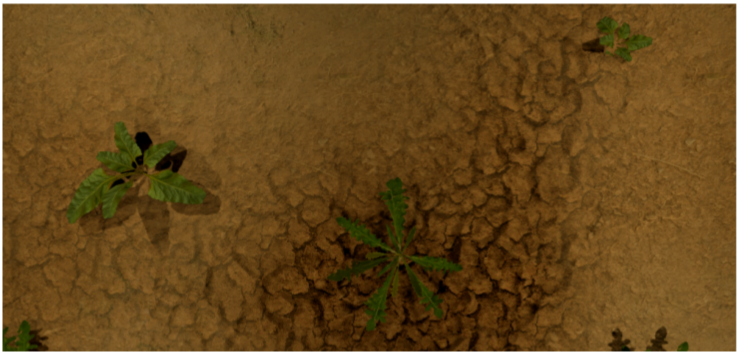
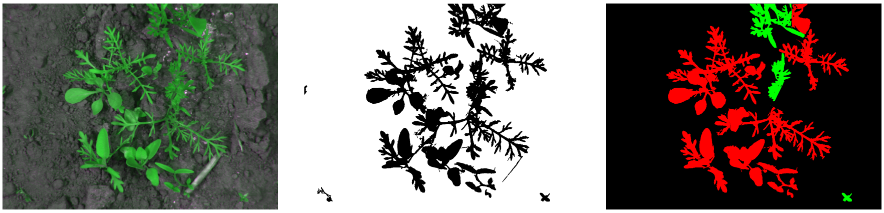


Figure 2.9: An example of a generated RGB image of plants and weeds. Image courtesy of [22].

pictures. Several other information is also incorporated to produce photorealistic images including ambient occlusions, normals and height maps, etc., An example of such images is shown in Fig.2.9. Chebrolu *et al.* publish a dataset for plant classification, localization, and mapping on sugar beat fields in [13]. The dataset contains multi-spectral images, RGB-D images, 3D point cloud, wheel encoder, RTK-GNSS system from Leica, and a low price GNSS receiver from uBlox. The dataset was recorded over a period of three months. A small portion of RGB



(a) RGB image (b) Mask image (c) Annotated image

Figure 2.10: An example of annotated data for crop/weed classification from a public dataset of [38]. Red color denotes plants and green color denotes weeds.

images is manually labeled for sugar beet plants and different types of weed. The authors also provide a terrestrial scan of a whole sugar beet field using a FARO X130 scanner. Haug and Ostermann publish a crop/weed filed image dataset of carrot plants and weeds in [38]. However, the dataset is rather small, containing only sixty images. All images are annotated for the classification of carrot plants and weeds. The authors also provide a binary mask for background/foreground segmentation. Pire *et al.* publish dataset for localization and mapping on soybean fields in [70]. This dataset is mostly similar to the one published by Chebrolu *et al.* [13]. However, it does not provide any annotated images for classification. Overall, public datasets for agricultural robotics are very few, in comparison to those for autonomous driving, point cloud registrations, visual-inertial odometry, etc.,

Traditional fruit productions demand high manual labor costs and contain mostly tedious tasks. Hayashi *et al.* in [40] shows that in Japan, protected horticulture production in small greenhouses has been moving toward large-scale greenhouses. This change requires substantial changes in productivity and employment. With the advance of technology, harvesting robots has been raised as a research focus. Although harvesting robots show encouraging results in increased productivity, the overall performance is still insufficient in comparison to manual operations [36]. Bac *et al.* evaluate the performance of a sweet pepper harvesting robot, in which the robot achieved a success rate between $26\% - 33\%$. Lehnert *et al.* develop a different sweet pepper harvesting system, called *Harvey* in [54]. The robot achieved a 46% success rate of harvesting for unmodified crops and 58% for protected crops with an average picking time per pepper around $35 - 40s$. Later, an upgraded version of **Harvey** is introduced in [55] with an improved success harvesting rate at 76.5%. Autonomous harvesting has also been introduced to different types of fruit. Henten *et al.* design an autonomous cucumber harvesting system in [92]. Mehta and Burks introduce a robotic manipulator system for citrus harvesting in [62]. Hayashi *et al.* and Xiong *et al.* propose autonomous strawberry picking system in [39] and [100], respectively. The mentioned strawberry, sweet pepper, and cucumber harvesting systems rely on vision-based sensors such as RGB camera, depth camera, etc., to detect and localize a fruit target. Then an end-effector needs to approach the target for cutting and retrieving. For different types of fruits such as apples, a vibratory harvesting system is more suitable. He *et al.* propose an adaptive vibratory system for apple harvesting in [41]. The system identifies the optimal shaking frequencies for different tree branches. Zhang *et al.* develop a shake-and-catch system for apple harvesting. The proposed system can detect and identify target for the shaker with 72.7% acceptance rate in comparison to a human expert's input.

References

[1] Aulinas, J. et al. "The SLAM Problem: A Survey". In: *Proceedings of the International Conference of the Catalan Association for Artificial Intelligence*. IOS Press, 2008, pp. 363–371.

[2] Barfoot, Timothy D. *State estimation for robotics*. Cambridge University Press, 2017.

[3] Barrau, Axel and Bonnabel, Silvere. "Intrinsic filtering on Lie groups with applications to attitude estimation". In: *IEEE Transactions on Automatic Control* vol. 60, no. 2 (2014), pp. 436–449.

[4] Bauer, Sebastian et al. "Real-time RGB-D Mapping and 3-D Modeling on the GPU using the Random Ball Cover". In: *Consumer Depth Cameras for Computer Vision*. Springer, 2013, pp. 27–48.

[5] Bell, Bradley M and Cathey, Frederick W. "The iterated Kalman filter update as a Gauss-Newton method". In: *IEEE Transactions on Automatic Control* vol. 38, no. 2 (1993), pp. 294–297.

[6] Besl, Paul J and McKay, Neil D. "Method for registration of 3-D shapes". In: *Sensor Fusion IV: Control Paradigms and Data Structures*. Vol. 1611. International Society for Optics and Photonics. 1992, pp. 586–607.

[7] Bloesch, Michael et al. *A Primer on the Differential Calculus of 3D Orientations*. 2016. arXiv: 1606.05285 [cs.RO].

[8] Bosse, Michael and Zlot, Robert. "Keypoint design and evaluation for place recognition in 2D lidar maps". In: *Robotics and Autonomous Systems* vol. 57, no. 12 (2009), pp. 1211–1224.

[9] Bourmaud, Guillaume et al. "Continuous-discrete extended Kalman filter on matrix Lie groups using concentrated Gaussian distributions". In: *Journal of Mathematical Imaging and Vision* vol. 51, no. 1 (2015), pp. 209–228.

[10] Bourmaud, Guillaume et al. "Discrete extended Kalman filter on Lie groups". In: *21st European Signal Processing Conference (EUSIPCO 2013)*. IEEE. 2013, pp. 1–5.

[11] Carlone, Luca et al. "Towards 4D crop analysis in precision agriculture: Estimating plant height and crown radius over time via expectation-maximization". In: *ICRA Workshop on Robotics in Agriculture*. 2015.

[12] Chebrolu, Nived, Läbe, Thomas, and Stachniss, Cyrill. “Robust long-term registration of UAV images of crop fields for precision agriculture”. In: *IEEE Robotics and Automation Letters* vol. 3, no. 4 (2018), pp. 3097–3104.

[13] Chebrolu, Nived et al. “Agricultural robot dataset for plant classification, localization and mapping on sugar beet fields”. In: *The International Journal of Robotics Research* vol. 36, no. 10 (2017), pp. 1045–1052.

[14] Chiu, Han-Pang et al. “Constrained optimal selection for multi-sensor robot navigation using plug-and-play factor graphs”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 663–670.

[15] Civera, Javier, Davison, Andrew J, and Montiel, JM Martinez. “Inverse depth parametrization for monocular SLAM”. In: *IEEE transactions on robotics* vol. 24, no. 5 (2008), pp. 932–945.

[16] Civera, Javier et al. “Towards semantic SLAM using a monocular camera”. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2011, pp. 1277–1284.

[17] Cucci, Davide Antonio and Matteucci, Matteo. “A Flexible Framework for Mobile Robot Pose Estimation and Multi-Sensor Self-Calibration.” In: *ICINCO (2)*. 2013, pp. 361–368.

[18] Curless, Brian and Levoy, Marc. “A volumetric method for building complex models from range images”. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 303–312.

[19] Dame, Amaury et al. “Dense reconstruction using 3D object shape priors”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 1288–1295.

[20] Davison, Andrew J et al. “MonoSLAM: Real-time single camera SLAM”. In: *IEEE transactions on pattern analysis and machine intelligence* vol. 29, no. 6 (2007), pp. 1052–1067.

[21] Dellaert, Frank. *Factor graphs and GTSAM: A hands-on introduction*. Tech. rep. Georgia Institute of Technology, 2012.

[22] Di Cicco, Maurilio et al. “Automatic model based dataset generation for fast and accurate crop and weeds detection”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 5188–5195.

[23] Dubé, Renaud et al. “SegMap: 3D Segment Mapping using Data-Driven Descriptors”. In: *arXiv preprint arXiv:1804.09557* (2018).

[24] Dubé, Renaud et al. “Segmatch: Segment based place recognition in 3d point clouds”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 5266–5272.

[25] Durrant-Whyte, H. F. and Bailey, T. “Simultaneous Localisation and Mapping (SLAM): Part I”. In: vol. 13, no. 2 (2006), pp. 99–110.

[26] Engel, Jakob, Koltun, Vladlen, and Cremers, Daniel. "Direct sparse odometry". In: *IEEE transactions on pattern analysis and machine intelligence* vol. 40, no. 3 (2017), pp. 611–625.

[27] Engel, Jakob, Schöps, Thomas, and Cremers, Daniel. "LSD-SLAM: Large-scale direct monocular SLAM". In: *European conference on computer vision.* Springer. 2014, pp. 834–849.

[28] Forster, Christian et al. "On-Manifold Preintegration for Real-Time Visual–Inertial Odometry". In: *IEEE Transactions on Robotics* vol. 33, no. 1 (2016), pp. 1–21.

[29] Fraundorfer, F. and Scaramuzza, D. "Visual Odometry : Part II: Matching, Robustness, Optimization, and Applications". In: *IEEE Robotics Automation Magazine* vol. 19, no. 2 (2012), pp. 78–90.

[30] Frisken, Sarah F et al. "Adaptively sampled distance fields: A general representation of shape for computer graphics". In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques.* 2000, pp. 249–254.

[31] Gao, Xiang et al. "LDSO: Direct sparse odometry with loop closure". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE. 2018, pp. 2198–2204.

[32] Gibson, Sarah F Frisken. "Using distance maps for accurate surface representation in sampled volumes". In: *IEEE Symposium on Volume Visualization (Cat. No. 989EX300).* IEEE. 1998, pp. 23–30.

[33] Gojcic, Zan et al. "The Perfect Match: 3D Point Cloud Matching with Smoothed Densities". In: *International conference on computer vision and pattern recognition (CVPR).* 2019.

[34] Goodman, Irwin R, Mahler, Ronald P, and Nguyen, Hung T. *Mathematics of data fusion.* Vol. 37. Springer Science & Business Media, 2013.

[35] Grant, W Shane, Voorhies, Randolph C, and Itti, Laurent. "Finding planes in LiDAR point clouds for real-time registration". In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on.* IEEE. 2013, pp. 4347–4354.

[36] Grift, Tony et al. "A review of automation and robotics for the bio-industry". In: *Journal of Biomechatronics Engineering* vol. 1, no. 1 (2008), pp. 37–54.

[37] Grisetti, G. et al. "A tutorial on graph-based SLAM". In: vol. 2, no. 4 (2010), pp. 31–43.

[38] Haug, Sebastian and Ostermann, Jörn. "A crop/weed field image dataset for the evaluation of computer vision based precision agriculture tasks". In: *European Conference on Computer Vision.* Springer. 2014, pp. 105–116.

[39] Hayashi, Shigehiko et al. "Evaluation of a strawberry-harvesting robot in a field test". In: *Biosystems engineering* vol. 105, no. 2 (2010), pp. 160–171.

[40] Hayashi, Shigehiko et al. “Robotic harvesting technology for fruit vegetables in protected horticultural production”. In: *Information and Technology for Sustainable Fruit and Vegetable Production* (2005), pp. 227–236.

[41] He, Leiying et al. “In-situ identification of shaking frequency for adaptive vibratory fruit harvesting”. In: *Computers and Electronics in Agriculture* vol. 170 (2020), p. 105245.

[42] Henry, Peter et al. “RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments”. In: *The International Journal of Robotics Research* vol. 31, no. 5 (2012), pp. 647–663.

[43] Hertzberg, Christoph et al. “Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds”. In: *Information Fusion* vol. 14, no. 1 (2013), pp. 57–77.

[44] Hornung, Armin et al. “OctoMap: An efficient probabilistic 3D mapping framework based on octrees”. In: *Autonomous Robots* vol. 34, no. 3 (2013), pp. 189–206.

[45] Hsiung, Jerry et al. “Information sparsification in visual-inertial odometry”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1146–1153.

[46] Huang, Shoudong and Dissanayake, Gamini. “A critique of current developments in simultaneous localization and mapping”. In: *International Journal of Advanced Robotic Systems* vol. 13, no. 5 (2016), p. 1729881416669482.

[47] Huang, Shoudong and Dissanayake, Gamini. “Convergence and consistency analysis for extended Kalman filter based SLAM”. In: *IEEE Transactions on robotics* vol. 23, no. 5 (2007), pp. 1036–1049.

[48] Indelman, Vadim et al. “Factor graph based incremental smoothing in inertial navigation systems”. In: *2012 15th International Conference on Information Fusion*. IEEE. 2012, pp. 2154–2161.

[49] Kaess, Michael, Ranganathan, Ananth, and Dellaert, Frank. “iSAM: Incremental smoothing and mapping”. In: *IEEE Transactions on Robotics* vol. 24, no. 6 (2008), pp. 1365–1378.

[50] Kaess, Michael et al. “iSAM2: Incremental smoothing and mapping using the Bayes tree”. In: *The International Journal of Robotics Research* vol. 31, no. 2 (2012), pp. 216–235.

[51] Khaleghi, Bahador et al. “Multisensor data fusion: A review of the state-of-the-art”. In: *Information fusion* vol. 14, no. 1 (2013), pp. 28–44.

[52] Klein, Georg and Murray, David. “Parallel tracking and mapping for small AR workspaces”. In: *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE. 2007, pp. 225–234.

[53] Kubelka, Vladimır et al. “Robust data fusion of multimodal sensory information for mobile robots”. In: *Journal of Field Robotics* vol. 32, no. 4 (2015), pp. 447–473.

[54] Lehnert, Christopher et al. “Autonomous sweet pepper harvesting for protected cropping systems”. In: *IEEE Robotics and Automation Letters* vol. 2, no. 2 (2017), pp. 872–879.

[55] Lehnert, Chris et al. “A sweet pepper harvesting robot for protected cropping environments”. In: *arXiv preprint arXiv:1810.11920* (2018).

[56] Leutenegger, Stefan et al. “Keyframe-based visual–inertial odometry using nonlinear optimization”. In: *The International Journal of Robotics Research* vol. 34, no. 3 (2015), pp. 314–334.

[57] Li, Yangming and Olson, Edwin B. “Structure tensors for general purpose LIDAR feature extraction”. In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 1869–1874.

[58] Lowry, S. et al. “Visual Place Recognition: A Survey”. In: vol. 32, no. 1 (2016), pp. 1–19.

[59] Lynen, Simon et al. “A robust and modular multi-sensor fusion approach applied to mav navigation”. In: *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE. 2013, pp. 3923–3929.

[60] Maybeck, Peter S. *Stochastic models, estimation, and control*. Academic press, 1982.

[61] Mazuran, Mladen, Burgard, Wolfram, and Tipaldi, Gian Diego. “Nonlinear factor recovery for long-term SLAM”. In: *The International Journal of Robotics Research* vol. 35, no. 1-3 (2016), pp. 50–72.

[62] Mehta, SS and Burks, TF. “Vision-based control of robotic manipulator for citrus harvesting”. In: *Computers and Electronics in Agriculture* vol. 102 (2014), pp. 146–158.

[63] Milioto, Andres, Lottes, Philipp, and Stachniss, Cyrill. “Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in CNNs”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2229–2235.

[64] Montemerlo, Michael et al. “FastSLAM: A factored solution to the simultaneous localization and mapping problem”. In: *Aaai/iaai* vol. 593598 (2002).

[65] Moore, Thomas and Stouch, Daniel. “A generalized extended kalman filter implementation for the robot operating system”. In: *Intelligent autonomous systems 13*. Springer, 2016, pp. 335–348.

[66] Moulin, S, Bondeau, Alberte, and Delecolle, R. “Combining agricultural crop models and satellite observations: from field to regional scales”. In: *International Journal of Remote Sensing* vol. 19, no. 6 (1998), pp. 1021–1036.

[67] Munz, Michael, Dietmayer, Klaus, and Mählisch, Mirko. “Generalized fusion of heterogeneous sensor measurements for multi target tracking”. In: *2010 13th International Conference on Information Fusion*. IEEE. 2010, pp. 1–8.

[68] Murphy, Robin R. “Dempster-Shafer theory for sensor fusion in autonomous mobile robots”. In: *IEEE Transactions on Robotics and Automation* vol. 14, no. 2 (1998), pp. 197–206.

[69] Newcombe, Richard A et al. “KinectFusion: Real-time dense surface mapping and tracking”. In: *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. IEEE. 2011, pp. 127–136.

[70] Pire, Taihú et al. “The Rosario dataset: Multisensor data for localization and mapping in agricultural environments”. In: *The International Journal of Robotics Research* vol. 38, no. 6 (2019), pp. 633–641.

[71] Potena, Ciro et al. “AgriColMap: Aerial-ground collaborative 3D mapping for precision farming”. In: *IEEE Robotics and Automation Letters* vol. 4, no. 2 (2019), pp. 1085–1092.

[72] Qin, Tong, Li, Peiliang, and Shen, Shaojie. “Vins-mono: A robust and versatile monocular visual-inertial state estimator”. In: *IEEE Transactions on Robotics* vol. 34, no. 4 (2018), pp. 1004–1020.

[73] Qiu, Deyuan, May, Stefan, and Nüchter, Andreas. “GPU-accelerated nearest neighbor search for 3D registration”. In: *International Conference on Computer Vision Systems*. Springer. 2009, pp. 194–203.

[74] Ranganathan, Ananth, Kaess, Michael, and Dellaert, Frank. “Fast 3D pose estimation with out-of-sequence measurements”. In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2007, pp. 2486–2493.

[75] Ratasich, Denise et al. “Generic sensor fusion package for ROS”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 286–291.

[76] Rosinol, Antoni et al. “Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping”. In: *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. 2020.

[77] Rusu, Radu Bogdan et al. “Fast 3d recognition and pose using the viewpoint feature histogram”. In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE. 2010, pp. 2155–2162.

[78] Rusu, Radu Bogdan et al. “Learning informative point classes for the acquisition of object model maps”. In: *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*. IEEE. 2008, pp. 643–650.

[79] Saeedi, S. et al. “Multiple-Robot Simultaneous Localization and Mapping: A Review”. In: vol. 33, no. 1 (2016), pp. 3–46.

[80] Salas-Moreno, Renato F et al. “Slam++: Simultaneous localisation and mapping at the level of objects”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2013, pp. 1352–1359.

[81] Scaramuzza, D. and Fraundorfer, F. “Visual Odometry [Tutorial]. Part I: The First 30 Years and Fundamentals”. In: vol. 18, no. 4 (2011), pp. 80–92.

[82] Serafin, Jacopo, Olson, Edwin, and Grisetti, Giorgio. “Fast and robust 3d feature extraction from sparse point clouds”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE. 2016, pp. 4105–4112.

[83] Shan, Tixiao and Englot, Brendan. “LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE. 2018, pp. 4758–4765.

[84] Simon, Dan. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches.* John Wiley & Sons, 2006.

[85] Stachniss, C., Thrun, S., and Leonard, J. J. “Simultaneous Localization and Mapping”. In: *Springer Handbook of Robotics.* Ed. by Siciliano, B. and Khatib, O. 2nd. 2016. Chap. 46, pp. 1153–1176.

[86] Steder, Bastian, Grisetti, Giorgio, and Burgard, Wolfram. “Robust place recognition for 3D range data based on point features”. In: *2010 IEEE International Conference on Robotics and Automation.* IEEE. 2010, pp. 1400–1405.

[87] Steinbrücker, Frank, Sturm, Jürgen, and Cremers, Daniel. “Volumetric 3D mapping in real-time on a CPU”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA).* IEEE. 2014, pp. 2021–2028.

[88] Strasdat, Hauke, Montiel, José MM, and Davison, Andrew J. “Visual SLAM: why filter?” In: *Image and Vision Computing* vol. 30, no. 2 (2012), pp. 65–77.

[89] Thrun, Sebastian. “Probabilistic robotics”. In: *Communications of the ACM* vol. 45, no. 3 (2002), pp. 52–57.

[90] Triggs, Bill et al. “Bundle adjustment—a modern synthesis”. In: *International workshop on vision algorithms.* Springer. 1999, pp. 298–372.

[91] Usenko, V. et al. “Visual-Inertial Mapping With Non-Linear Factor Recovery”. In: *IEEE Robotics and Automation Letters* vol. 5, no. 2 (2020), pp. 422–429.

[92] Van Henten, Eldert J et al. “An autonomous robot for harvesting cucumbers in greenhouses”. In: *Autonomous robots* vol. 13, no. 3 (2002), pp. 241–258.

[93] Velas, Martin, Spanel, Michal, and Herout, Adam. “Collar Line Segments for fast odometry estimation from Velodyne point clouds.” In: *ICRA.* 2016, pp. 4486–4495.

[94] Vial, John, Durrant-Whyte, Hugh, and Bailey, Tim. “Conservative sparsification for efficient and consistent approximate estimation”. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2011, pp. 886–893.

[95] Wang, Kaixuan, Gao, Fei, and Shen, Shaojie. “Real-time scalable dense surfel mapping”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 6919–6925.

[96] Wang, Yue et al. “Kullback-leibler divergence based graph pruning in robotic feature mapping”. In: *2013 European Conference on Mobile Robots*. IEEE. 2013, pp. 32–37.

[97] Weiss, Stephan et al. “Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV”. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 31–38.

[98] Whelan, Thomas et al. “ElasticFusion: Dense SLAM without a pose graph”. In: Robotics: Science and Systems. 2015.

[99] Whelan, Thomas et al. “Kintinuous: Spatially extended kinectfusion”. In: (2012).

[100] Xiong, Ya et al. “An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation”. In: *Journal of Field Robotics* (2019).

[101] Yang, Heng, Shi, Jingnan, and Carlone, Luca. “TEASER: Fast and Certifiable Point Cloud Registration”. In: (2020). arXiv: 2001.07715 [cs.RO].

[102] Zhang, Ji and Singh, Sanjiv. “LOAM: Lidar Odometry and Mapping in Real-time.” In: *Robotics: Science and Systems*. Vol. 2. 2014, p. 9.

[103] Zhang, Ji and Singh, Sanjiv. “Low-drift and real-time lidar odometry and mapping”. In: *Autonomous Robots* vol. 41, no. 2 (2017), pp. 401–416.

Chapter 3

Contribution

This chapter details the contributions of each of the papers presented as parts of this cumulative thesis. We will describe the context of the work, contributions, and finally how the paper relates to the rest of the thesis.

We present the papers in the following order: mapping algorithm, localization algorithm, and autonomous navigation. We consider our last paper on supervised learning solution for row following task in horticulture as a part of autonomous navigation.

3.1 Paper I

Tuan Le, Jon Glenn Omholt Gjevestad and Pål Johan From, "Online 3D Mapping and Localization System for Agricultural Robots", *6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture*, Sydney, December, 2019.

Context

This paper attempted to solve the problem of large-scale mapping in agriculture using a 3D LiDAR sensor. Many methods have been studied for LiDAR mapping in the literature [4, 6, 8, 10]. One popular and is considered as the state-of-the-art method is LiDAR odometry and mapping (LOAM) by Zhang et al., [8]. While this method works well in urban/city scenarios, it likely fails to work in agricultural environments. An agricultural environment is inherently different from an urban/city scene. The latter contains an abundant amount of visual features such as planes (buildings, walls, roads, etc.,), edges (windows, street lights, lane signs, etc.,). These features are easy to detect and track, hence mapping in urban/city environments is most problematic with dynamic object detection (pedestrians, moving vehicles, etc.,). On a farm, the existence of those features is scarce. Moreover, the scene is likely dominated by trees, leaves, grass, etc., whereas they might offer corner or edge features, they are extremely hard to track. We aim to develop a 3D LiDAR mapping and localization method that can specifically handle agricultural environments.

Contribution

We highlight the contributions of this work as follows:

- a complete online 3D LiDAR mapping and localization system for autonomous agricultural robots
- high quality built map for a human operator and subsequent reuse

- an evaluation of the proposed system on both simulation and real experiments

We notice that existing methods such as LOAM can store its built map and use it for relocalization purposes. However, the authors of LOAM does not focus on this functionality. Hence, our proposed system fills in the gap for agricultural applications. Even though our method is not suitable for crop field environments, where the appearance of plants gradually changes, it is still applicable for other agricultural tasks such as product transportation between fields and storage, or between polytunnels/greenhouses. Therefore, we argue that our proposed system is still useful.

Interrelation

This paper suggests a better mapping and localization approach for agricultural robots on large-scale farms. This paper exhibits our desired properties of agricultural robots discussed in Chapter 1. Specifically, the robot can operate in GNSS-denied environments since it does not rely on any external localization system. It demonstrates a minimalistic design by using only one 3D LiDAR sensor and an IMU. It is platform agnostic. Even though we did not explicitly test our system on aerial robots, the fact that we always estimate 6-DOF poses means our SLAM system can directly apply to aerial robots without any modification. It is scalable. Our approach was developed on the ROS framework. Finally, it can assist a human operator by providing a finer map than traditional methods.

3.2 Paper II

Tuan Le, Jon Glenn Omholt Gjevestad and Pål Johan From, “A Cost-Effective Global Localization System for Precision Agriculture Tasks in Polytunnels”, *IEEE 16th International Conference on Automation Science and Engineering*, Hongkong, August, 2020.

Context

This paper focused on the localization problem for agricultural robots in dynamical agricultural environments such as growing strawberry in polytunnels. A strawberry polytunnel is a plant-growing area covered by polymer material. A common structure of a strawberry polytunnel consists of several evenly-spaced sets of poles, on top of which hold table-trays. We aim to build a global localization system that can handle challenging environments such as strawberry polytunnel. In this type of environment, highly repetitive patterns from plants, structures, etc., dominates the scene. They render traditional methods relying on visual features such as SIFT, ORB, BRISK [5, 7] fail to work. Moreover, even though there might be some distinct features detected from the scene such as a specific leave or fruit, those features are unstable due to the growing process of the plants. Hence, we tackle this problem of ambiguity by leveraging the power of object recognition using machine learning. In a polytunnel, the most invariant objects are the poles. We train a convolutional neural network (CNN) to segment the poles from a captured image and use them for localizing. Also, we target to build a cost-effective and platform-agnostic system. We only require an Intel Realsense camera D435i, which gives us a stereo visual-inertial system and an RGB-D camera in a compact form. The localization system provides a full 6DOF pose estimation in a prior global map.

Contribution

The main contribution of this paper is a cost-effective global localization system for agricultural robots deployed in polytunnels. Our system is able to:

- localize with the required accuracy for the robot to navigate between table-top rows in strawberry polytunnels
- provide an alternative method to GNSS-based localization system which might suffer from signal outages in GNSS-denied environments such as e.g. polytunnels
- perform robust localization over extended periods of time across plant season without remapping the environment.

Interrelation

This paper built the second block of an autonomous agricultural robot, which is the localization functionality. The combination of the mapping system in the first paper and the localization system in this paper provides a robot with the full perception capability. The robot now knows where it is in an environment and what that environment looks like. The localization system again exhibits our desired properties of agricultural robots.

3.3 Paper III

Tuan Le, Ponnambalam, V. R., Jon Glenn Omholt Gjevestad and Pål Johan From, "A low cost and efficient autonomous row following robot for food production in polytunnels", *Journal of Field Robotics 37.2 (2020): 309-321.*

Context

This paper aimed to solve a navigation problem in a tightly space-constraint agricultural environment such as a strawberry polytunnel. We wanted to build a system that is simple to set up and easy to operate. A 2D laser scanner is chosen as the only perception sensor. The reason for this selection is that we want a robust system that tackles degenerated environments such as low-light, sparse features, etc., A camera must rely on illumination to capture a scene. Hence, cameras suffer to work in harsh environments or in tasks that have to be operated at night time, for example, UV-light treatment. Besides, the target environment is more challenging than those that are commonly found in agriculture, namely the rows are curved or zigzagged-like. Traditional methods of row following have been well-studied [1, 2, 3, 9]. However, they did not explicitly deal with this challenging environment of curved rows.

Contribution

The main contributions of this paper are as follow:

- a minimal system consists of one 2D laser scanner that can freely navigate an agricultural robot inside a polytunnel
- a robust navigation system that can handle challenging situations in polytunnel environments such as curved/zigzagged rows

Interrelation

This paper completed the final piece of an autonomous system: navigation. After perceiving about the surrounding environment and know where it is in that environment, an agricultural robot can now safely move to its goals while performing its task, such as providing UV-light treatment to plants. The navigation strategy is simple but yet efficient for the task at hand. The system also follows our desired properties. Noted that in this paper, the motion planning algorithm is platform-specific. However, this is common to have different motion plannings for different robotic platforms. Hence, we argue that this paper is still related to our desired properties of autonomous agricultural robots.

3.4 A supervised learning solution for row following tasks in horticulture

Tuan Le, Vignesh Raja Ponnambalam, Jon Glenn Omholt Gjevestad, Pål Johan From, "A supervised learning solution for autonomous row following tasks in horticulture" ***Submitted to*** *IROS 2020 Workshop on Perception, Planning, and Mobility in Forestry Robotics (WPPMFR 2020)*, Las Vegas, Nevada, USA, October, 2020.

Context

Row following is one of the key activities in horticulture. Regardless of specific agricultural environments, e.g indoor (polytunnels, greenhouses) or outdoor (crop fields, orchards) and activities (UV light treatment, weeding, pollination, harvesting), a robot should be able to follow rows. A lot of work have been done to enable this capability, including two of our published works. However, most of them are tailored to some specific setups, including environments (indoor or outdoor), activity (centerline following or sideline following), and robot size. We aim to develop a method that can be used as an alternative solution to existing ones for row following, regardless of targeted environments and robot platform.

Contribution

This work attempts to solve a key task in horticulture - row following. We propose a solution using only RGB-D images. We train a convolutional neural network (CNN) for segmenting parts of an image suitable for robot movements. We label those segmented areas as *traversable*. As different from existing methods, we train our network on an inclusive dataset, which contains both indoor and outdoor horticultural environments. Hence, it can be used for a variety of row following tasks (centerline or sideline following) in different environments. This work serves as a baseline comparison for our future work on releasing a data set on autonomous navigation for agricultural robots to the agricultural robotics community.

Interrelation

In this work, we explore one tool that has been widely used in computer vision and robotics communities - supervised learning method, on solving a key task for autonomous agricultural robots in horticulture - row following. As supervised learning-based techniques are now ubiquitous, one can consider them as *off-the-shelf* products. Hence, we are intrigued to explore the possibility of applying them to solve row following tasks in horticulture. We have created an inclusive dataset for autonomous row following tasks and have achieved promising results. We plan to continue working on releasing our data set to the agricultural research community.

References

[1] Åstrand, Björn and Baerveldt, Albert-Jan. “A vision based row-following system for agricultural field machinery”. In: *Mechatronics* vol. 15, no. 2 (2005), pp. 251–269.

[2] Bergerman, Marcel et al. “Robot farmers: Autonomous orchard vehicles help tree fruit production”. In: *IEEE Robotics & Automation Magazine* vol. 22, no. 1 (2015), pp. 54–63.

[3] Biber, Peter et al. “Navigation system of the autonomous agricultural robot Bonirob”. In: *Workshop on Agricultural Robotics: Enabling Safe, Efficient, and Affordable Robots for Food Production (Collocated with IROS 2012), Vilamoura, Portugal.* 2012.

[4] Kohlbrecher, Stefan et al. “A flexible and scalable slam system with full 3d motion estimation”. In: *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics.* IEEE. 2011, pp. 155–160.

[5] Leutenegger, Stefan, Chli, Margarita, and Siegwart, Roland Y. “BRISK: Binary robust invariant scalable keypoints”. In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2548–2555.

[6] Magnusson, Martin, Lilienthal, Achim, and Duckett, Tom. “Scan registration for autonomous mining vehicles using 3D-NDT”. In: *Journal of Field Robotics* vol. 24, no. 10 (2007), pp. 803–827.

[7] Rublee, Ethan et al. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2564–2571.

[8] Zhang, Ji and Singh, Sanjiv. “LOAM: Lidar Odometry and Mapping in Real-time.” In: *Robotics: Science and Systems*. Vol. 2. 2014, p. 9.

[9] Zhang, Ji et al. “3d perception for accurate row following: Methodology and results”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 5306–5313.

[10] Zhen, Weikun et al. “LiDAR Enhanced Structure-from-Motion”. In: *arXiv preprint arXiv:1911.03369* (2019).

Chapter 4

List of Publications

The following four papers, listed in chronological order, are included in this thesis:

- Tuan Le, Jon Glenn Omholt Gjevestad, and Pål Johan From. Online 3D Mapping and Localization System for Agricultural Robots. In *6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture*. IFAC-PapersOnLine 52.30 (2019): 167-172. DOI: 10.1016/j.ifacol.2019.12.516

- Tuan Le*, Vignesh R. Ponnambalam*, Jon GO Gjevestad, and Pål J. From. A low-cost and efficient autonomous row-following robot for food production in polytunnels. *Journal of Field Robotics* 37, no. 2 (2020): 309-321. DOI: 10.1002/rob.21878. (*authors have contributed equally)

- Tuan Le, Jon Glenn Omholt Gjevestad, and Pål Johan From. A Cost-Effective Global Localization System for Precision Agriculture Tasks in Polytunnels. **Accepted, to be presented at** *IEEE 16th International Conference on Automation Science and Engineering*, Hongkong, August, 2020.

- Tuan Le, Vignesh R. Ponnambalam, Jon Glenn Omholt Gjevestad, and Pål Johan From. A supervised learning solution for autonomous row following tasks in horticulture. **Submitted to** *IROS 2020 Workshop on Perception, Planning, and Mobility in Forestry Robotics (WPPMFR 2020)*, Las Vegas, Nevada, USA, October, 2020.

The author has also contributed to a research paper not included in this thesis:

- L. Grimstad, R. Zakaria, T.D. Le, P.J. From. A Novel Autonomous Robot for Greenhouse Applications. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018. DOI: 10.1109/IROS.2018.8594233

Chapter 5

Conclusion and Outlook

This thesis provides three parts that as a sum, constitute a complete autonomous agricultural system. We aimed to solve the three fundamental problems of autonomous robots, including mapping, localization, and navigation. We are specifically tackling the challenge of highly repetitive patters, sparse visual features and dynamically changing agricultural environments. Note that, even though we used different sensors for different tasks, we did not explicitly require any specific sensor configuration. All developed algorithms can work with general range sensors, such as 2D laser scanner, RGB-D cameras, stereo VIO, and any 360°LiDAR without adaptation to individual sensor models. The core contributions of this thesis revolve around the three basic functions of an autonomous robot:

3D mapping in agricultural environments We proposed a 3D mapping system that can be substituted for any existing GNSS solutions. Our system leverages a 3D LiDAR sensor to build a large-scale map of agricultural environments with a minimum memory-footprint for storage. Our system is different from other 3D mapping methods, which mostly focus on urban/city scenarios. In comparison to state-of-the-art methods, our system provides a finer map quality and lower size for map storage. Also, our system can provide a good localization using the previously built 3D map, which makes it completely independent of any external global positioning system.

Localization in a highly dynamic agricultural environment A polytunnel is a challenging environment for a robot. It exhibits highly repetitive patterns for visual features, dynamic change over a short time. For example, a strawberry polytunnel undergoes rapid changes during a season. We proposed a system that can extract an "invariant" feature from such an environment and be able to localize robustly over a long period without the need to remap the environment. This is beneficial for agricultural production such that a prior map can be used over time in a dynamic scene without updating.

Autonomous navigation in a challenging polytunnel Row following is one of the most basic tasks an agricultural robot has to perform. We developed a navigation system that allows a ground agricultural robot traversing along rows inside a polytunnel. We addressed the corner case of a typical agricultural polytunnel, curved or zigzagged rows. This challenging environment makes existing methods unsuitable.

And finally, all designed and developed systems in this thesis support our desired properties for an agricultural robot: GNSS-independent operations, minimalistic system, platform-agnostic (except for the motion planning, which must be platform-specific), scalability and assistance to a human operator.

5.1 Future work

There are many avenues for future work to truly enable autonomous robots in agriculture. We will discuss some future work that can be built directly upon our current system.

Human safety As agricultural robots would often share the same workspace with human workers, safety is a non-negotiable issue. The current designed systems have been well-tested but only in non-human interaction environments. Hence, it is important and non-trivial to incorporate human safety features into agricultural robots before deploying to productions. One might adapt well-established safety guidelines for autonomous driving cars into agricultural scenarios.

Motion planning in tightly constraint environment Motion planning is a hard problem for autonomous robots. It is even more troublesome in agricultural domains. For example, in a polytunnel, the free space for trajectory generation is severely limited. On the other hand, for open fields, it is more relaxed when a robot moves off-field and again becomes more restricted when moving on-field (row following). Hence, it is difficult to have a generic motion planning for agricultural robots.

Exploration Regardless of operation modes (performing online or offline), mapping is the first step that one should perform before letting a robot doing tasks. The current mapping system currently requires a human supervisor for the whole process even though the mapping is done online. The robot should be able to perform exploration into the assigned agricultural environment and return the complete map of that environment autonomously. Exploration is not new for autonomous robots. However, in agricultural domains, special safety checks must be enforced.

Coordinations of multiple robots It is most likely that several robots would be needed for tasks on large farms. For example, harvesting on multiple-hectare fields or UV-light treatments for multiple polytunnels. Multiple robot coordination offers speed-up in production. Deploying multiple robots requires research works on communication, safety, production costs, etc., Our desired properties directly benefit for building fleets of robots.

Public data sets for research communities In computer vision community and autonomous driving sectors, the readiness of publicly available data sets has brought several benefits for the research communities. In the agricultural robotics community, the number of public data sets is still small. We plan to work on and release our own data sets focusing on safety and autonomous navigation for agricultural robots.

Papers

Paper I

Online 3D Mapping and Localization System for Agricultural Robots

Tuan Le, Jon Glenn Omholt Gjevestad, Pål Johan From

Published in *6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture*, December 2019, Volume 52, Issue 30, pp. 167–472. DOI: 10.1016/j.ifacol.2019.12.516.

Abstract

For an intelligent agricultural robot to reliably operate on a large-scale farm, it is crucial for the robot to accurately estimate its pose. In large outdoor environments, 3D LiDAR is a preferred sensor. Due to the inherent difference in characteristic of urban and agricultural scenarios, where the latter contains many poorly defined objects such as grass and trees with leaves that will generate noisy sensor signals. While state-of-the-art methods of state estimation using LiDAR, such as Lidar odometry and mapping (LOAM), works well in urban scenarios, they will fail in the agricultural domain. Hence, we propose a mapping and localization system to cope with challenging agricultural scenarios. Our system maintains a high quality global map for subsequent reuses of relocalization or motion planning. This is beneficial as we avoid the unnecessary repetitively mapping process. Our experimental results show that we achieve comparable or better performance in state estimation, localization, and map quality when compare to LOAM.

I.1 Introduction

For the last couple decades, we have witnessed an unprecedented advance in technologies like mobile robotics and artificial intelligence. These technologies bring positive effects on our daily lives: autonomous cars, service robots for elderly care and precise agricultural robots for food production. Among various applications, agricultural robots currently attract a lot of attention due to

All authors are with Faculty of Science and Technology, Norwegian University of Life Sciences

its important role in solving a vital problem: the demand for increased food production.

Different types of agricultural robots have been developed in recent years. Multi-wheel mobile robots [2, 10, 22, 24], is widely adopted due to its high capacity of transportation and outfit with multiple sensors. Multi-rotor flying robots are also used in agriculture [1, 20, 21], though their utility are limited to short-term operations because of short battery life and low computation capacity.

LiDAR mapping and localization has been widely studied in the literature [16, 18, 28]. However, most focus on indoor, urban or city scenarios. The difference in characteristic of urban and agricultural scene is significant. In an urban scene such as a city, sufficient features such as lines, planes, corners from houses, pavements, etc., can be extracted for scan registration. In an agricultural scene, objects such as grass, tall trees, tree leaves can not provide reliable features for detection to the same extent. For example, a tree leaf is unlikely to be observed twice in two consecutive scans. The ground in a farm is more likely to be rugged and not flat as a city street. These challenges prevent directly applying conventional method such as LOAM.

In this work, we propose a complete *online 3D mapping and localization* for our agricultural mobile robotic platform Thorvald II [11]. The robot is capable of *i)* incrementally build and localize in a 3D map using 3D point cloud data, *ii)* the global built map can be stored for subsequent reuse. Specifically, an optimization-based approach is used for estimating the robot odometry. We also employ loop-closure detection to ensure the large built 3D map is consistent and usable for later tasks without rebuilding it every time. For relocalization in a pre-built 3D map, we employ a normal distribution transformation (NDT) scan matching method. Both processes (map building and relocalization) are guaranteed to run online on the robot onboard computer. In summary, we highlight the contributions of this work as follows:

- a complete online 3D LiDAR mapping and localization system for autonomous agricultural robots
- high quality built map for human operator and subsequent reuse
- an evaluation of the proposed system on both simulation and real experiments

We notice that existing methods such as LOAM can store its built map and use it for relocalization purposes. However, the authors of LOAM does not focus on this functionality. Hence, our proposed system fills in the gap for agricultural applications. Even though our method is not suitable for crop field environments, where the appearance of plant gradually changes, it is still applicable for other agricultural tasks such as product transportation between fields and storage, between polytunnels. Therefore, we argue that our proposed system is still useful.

The paper is organized as follows: In section II, we review related work. Section III depicts our hardware system overview. Section IV and V discuss the

3D LiDAR map building and localization. Experimental results are presented in Section V and conclusions are discussed in Section VI.

I.2 Related Work

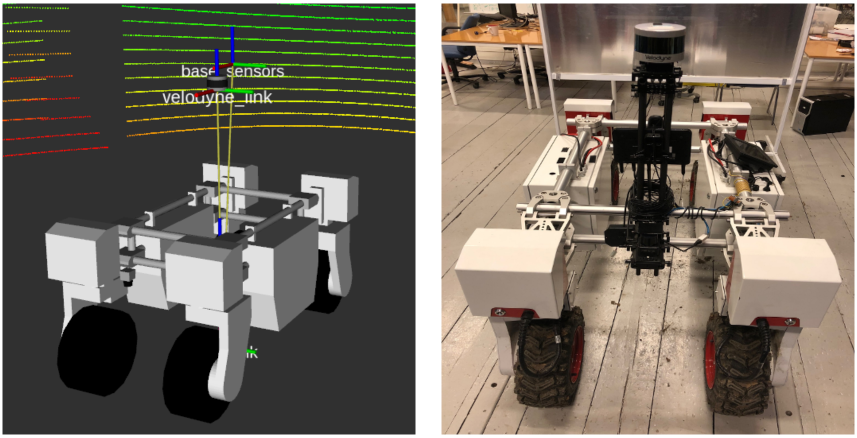
Several works on mapping and localization in agricultural domain have been focused on crop field environment. Early work by Khanna et al., [15] proposed a simple mapping solution by using a stereo camera for generating 3D pointcloud but using a commercial software. Albani et al., [1] proposed a decentralized multi-UAV system for crop field mapping and weed detection. However, the system was only tested in simulation without any validation from real field. Popović et al., [21] proposed a Gaussian Process model for generating a multiresolution map for biomass monitoring. More recently, Chebrolu et al., [19] combined aerial images and ground images for localizing in a prebuilt-aerial map of a sugar beet field. The aerial map is continuously updated after each session to maintain a high localization accuracy.

Beside crop fields, a robot might need to travel to other parts of a farm. For example, the robot might need to transport harvested products from crop fields to storage. For this task it also requires a good 3D map since the terrain on a farm is unlikely to be globally flat. Therefore, in this work, we aim to solve a 3D mapping and localization problem using 3D LiDAR for agricultural logistics application.

We focus on geometry approach for LiDAR odometry estimation. The state-of-the-art method, LOAM, is presented in [28, 29]. The method leverages point feature to edge/plane scan-matching for scan registrations. The state estimation is further divided into a cascade system: velocity is estimated with low accuracy but at high frequency and motion estimation runs at low frequency but returns high accuracy estimation. The fused output of the system is a high frequency and high accuracy motion estimation. The result of odometry estimation by LOAM is still by far the best on the KITTI odometry benchmark[1].

We notice a couple drawbacks that prevent us from directly implementing the original LOAM method. First, LOAM needs to iterate through every point in a given point cloud to compute features for scan matching. This poses a computational bottle neck. Second, an agricultural robot is likely to work in an environment filled with trees, grass, which makes detected features unreliable. For example, an edge feature from a tree leaf is unlikely to be observed twice for matching. Or grass with uneven height on the ground might give inconsistent planar features. And lastly, since LOAM focuses solely on odometry estimation, no loop closure or saving built map functionality is implemented. This prevents an agricultural robot from operating efficiently since it needs to rebuild a map of a large scale environment every time it is turned on. The work in [25] is the most similar to ours, however, like the original LOAM, the authors focus on odometry estimation only.

[1] http://www.cvlibs.net/datasets/kitti/eval_odometry.php



(a) Simulated robot

(b) Real robot

Figure I.1: Hardware system overview.

CSIRO group proposes several handcraft descriptors for place recognition such as DELIGHT [6], ISHOT [12] and keypoint voting [4]. These descriptor can be used to localize in a prior 3D map with LiDAR point cloud. However, they are computationally expensive and require the robot to stand still for localization. Caselitz et al. [5] utilize a monocamera to reconstruct a local 3D map and match it against a prior 3D map for localization. Even though the localization result is promising, the use of camera limits a robot to operate only at daylight time. We are inspired by an NDT-based approach for localization in [23]. However, the authors in [23] use a 2D-3D matching while we directly perform a 3D-3D matching. We argue that for agricultural environments, where features are sparse, the use of 2D LiDAR would severely limit the matching process for localization. Hence, we prefer a 3D-3D matching method.

I.3 System Overview

I.3.1 Hardware system overview

The robotic system used in this work is an agricultural mobile platform Thorvald II [11]. The robot is four wheel drive with a modular design. Unlike other fixed size agricultural robots such as the BonniRob [22] or Harvey [17], Thorvald II is easily size-reconfigurable for different tasks and domains [9, 27].

The robot is equipped with a 3D LiDAR Velodyne VLP-16 and a commercial grade IMU Xsens MTi-30. The complete hardware system is shown in Fig.I.1b.

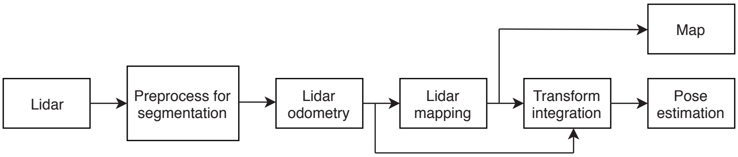


Figure I.2: Block diagram of map building.

I.3.2 Software system overview

The 3D mapping process is divided into three steps. First, incoming LiDAR measurement is preprocessed to separate a set of ground points from non-ground points. The set of non-ground points is further segmented into different clusters, each cluster contains points from one single object. Both set of ground points and object clusters are used for extracting edge and planar features. Second, extracted features are then used to match and estimate pose between consecutive LiDAR scans at scan rate. Pose estimations are further refined at a lower rate by registering those features to a global map. Finally, both pose estimations are fused to give the final pose estimation. Loop closure detection is also executed to guarantee a consistent global map. When the mapping process is done, the final global map is saved for later use.

For localization in a pre-built 3D map, we iteratively perform 3D-3D scan matching between LiDAR scan and the 3D map using the NDT representation of the map. Details of mapping and localizing procedure are further discussed in the next section.

I.4 3D LiDAR Mapping

I.4.1 Data Preprocessing

The original LOAM method [28, 29] works well in indoor environments. The authors also confirmed that feature matching is less reliable in outdoor environment due to worse feature extraction [29]. We adopt the approach in [3, 25] to preprocess the raw point cloud data before extracting its features. In particular, the ground points are first removed from the point cloud and the remaining points are segmented into clusters where each cluster contains points of one object. The whole segmentation process is based on the projected range image from the raw 3D point cloud for fast performance. We notice that Shan et al. [25] employs the ground removal strategy from [13] and require a heuristic predefined number of ground scans to perform ground detection. We find that the ground removal method by Bogoslavskyi et al. [3] is more robust and implement this approach.

Let $\mathcal{P}_k$, $k \in Z^+$ be set of point cloud at measurement k. After preprocessing, a set of ground point G_k and non-ground point Q_k $(G_k, Q_k \subset \mathcal{P}_k)$ are obtained for feature extraction. Notice that, P_k, Q_k also contain label for their points, i.e, ground label for ground points and unique label for each cluster and its points. We also eliminate clusters containing less than forty points. The idea of separating and labelling points is to further improve the feature matching process by matching only points with corresponding labels. For example, ground points are *never* used to match with edge features, which most likely come from non-ground points.

I.4.2 Odometry Estimation and Mapping

The LiDAR odometry estimation process is executed in the following order.

First, we extract features from the currently received LiDAR scan $\mathcal{P}_k$. Following [29], we also use a threshold to identify edge and planar features. However, to avoid iterating through 3D points, we perform this process using the projected range image as in [3, 25]. Let S be the set of all points p_i on the same row of the range image of $\mathcal{P}_k$. The roughness c of p_i is evaluated in Eq.I.1, where $\|\cdot\|$ is the Euclidean distance and $|\cdot|$ is the number of points:

$$c = \frac{\|\sum_{j \in S, j \neq i}(r_j - r_i)\|}{|S| \cdot \|r_i\|} \tag{I.1}$$

The point p_i is classified as edge feature if its roughness c score is greater than a threshold, or else it is considered as planar feature. Let $\mathcal{E}_k, \mathcal{H}_k$ be the sets of all extracted edge and planar features, respectively. [29] performs several condition checks to reject outliers feature points for scan matching. In contrast, by leveraging the label associated with each point, we still can ensure reliable scan matching result between scan $\mathcal{P}_k$ and $\mathcal{P}_{k-1}$ as follows. For each type of extracted features, we select a small subset of edge features $E_k, E_k \subset \mathcal{E}_k$ with maximum c score a small subset of planar features $H_k, H_k \subset \mathcal{H}_k$ with minimum c score. Then for finding correspondences, we only match points from E_k with points of the same label from $\mathcal{E}_{k-1}$ and similarly for H_k and $\mathcal{H}_{k-1}$.

Second, after finding the correspondences of the feature points, the distance between a point in the k^{th} scan and its correspondence is used to estimate the LiDAR motion, denoted as

$$\begin{aligned}\mathbf{x}_k =& [\mathbf{R}, \mathbf{T}] \\ \mathbf{T}_k =& [t_x, t_y, t_z]^T \\ \mathbf{R}_k =& [roll, pitch, yaw]^T\end{aligned} \tag{I.2}$$

where $\mathbf{T}_k$ and $\mathbf{R}_k$ is translational and rotational part, respectively. Stacking all the equations describe the geometric relationship between an edge points p_i and its corresponding edge line, we have:

$$f_{\mathcal{E}}(\mathbf{x}_{k,i}) = d_{\mathcal{E}}, i \in \mathcal{E}_k \tag{I.3}$$

Similarly, we can obtain another set of equations for planar points and their corresponding planar patches:

$$f_{\mathcal{H}}(\mathbf{x}_{k,i}) = d_{\mathcal{H}}, i \in \mathcal{H}_k \tag{I.4}$$

The detail derivation of $d_{\mathcal{E}}, d_{\mathcal{H}}$ is exactly as in [29] and omitted here for brevity. While in [29], the authors combine $f_{\mathcal{E}}(\mathbf{x}_{k,i}), f_{\mathcal{H}}(\mathbf{x}_{k,i})$ into one system of non linear equations and apply the Lavenberg-Marquardt (LM) method to solve it, we follow the approach in [25] to obtain the motion estimation in a more efficient way. We first solve Eq.I.4 using the same LM method. Notice that Eq.I.4 estimates transformation between planar patches, the estimation of roll, pitch angles and translation in z direction is more accurately estimated than other components. We then use the three components as constraints to solve Eq.I.3. Again, for edge lines in Eq.I.3, translation in x, y and yaw angles are estimated more robustly and we selectively choose these components. Finally, we fuse these six components together to achieve the final 6-DOF pose estimation.

Let $\mathcal{G}_{k-1}$ be the set of point clouds in the global map accumulated up to the LiDAR $(k-1)^{th}$ scan . We implement the similar method in [29] to match the points in $\mathcal{E}_k, \mathcal{H}_k$ to $\mathcal{G}_{k-1}$ to further refine the pose estimation. Readers are referred to [29] for the details. We notice the difference here is that we explicitly aim for a consistent and reusable large-scale map, not just accurate odometry estimation. Hence, we implement a pose-graph SLAM with loop closure detection in [7, 14] to obtain the fine map. Specifically, the pose obtained in the odometry estimation step is considered a node in the graph. A loop is detected by matching between $\mathcal{E}_k, \mathcal{H}_k$ and $\mathcal{E}_{k-1}, \mathcal{H}_{k-1}$. If a match is found, it is added as a new constraint to the graph. The graph is efficiently updated using iSAM2 library [14].

I.5 Localization in A Prior 3D LiDAR Map

Given a 3D LiDAR map built in the previous section, the robot can estimate its pose as the sensor ego-motion. We adopt an NDT-based scan matching for localization. In comparison to the ICP method, 3D NDT scan matching is faster and at least as accurate as the state-of-the-art ICP method [26]. It is especially beneficial for the robot to localize itself on a large scale map. Instead of performing heavy computation scan matching by iterating through every point, the robot only needs to compare between the much smaller estimated Gaussian components, which represent the map and the received LiDAR scans. In addition, the robot might experience abrupt changes on uneven terrain, which in turn causes a large displacement between consecutive scans.

Let $\mathbf{x}_t = [\mathbf{p}_t, \mathbf{q}_t, \mathbf{v}_t, \mathbf{b}_t^{\omega}]^T$ be the state vector at time t that we need to estimate, where $\mathbf{p}_t$ is the position, $\mathbf{q}_t$ is the rotation vector in quaternion representation, $\mathbf{v}_t$ is the velocity and $\mathbf{b}_t^{\omega}$ is a constant bias for raw gyroscope measurements $\hat{\boldsymbol{\omega}}_t$ from an IMU, that is rigidly attached to the LiDAR sensor frame. Since the robot normally runs at low speed, we can assume a constant translational velocity for the motion model. Employing a standard Extended Kalman filter,

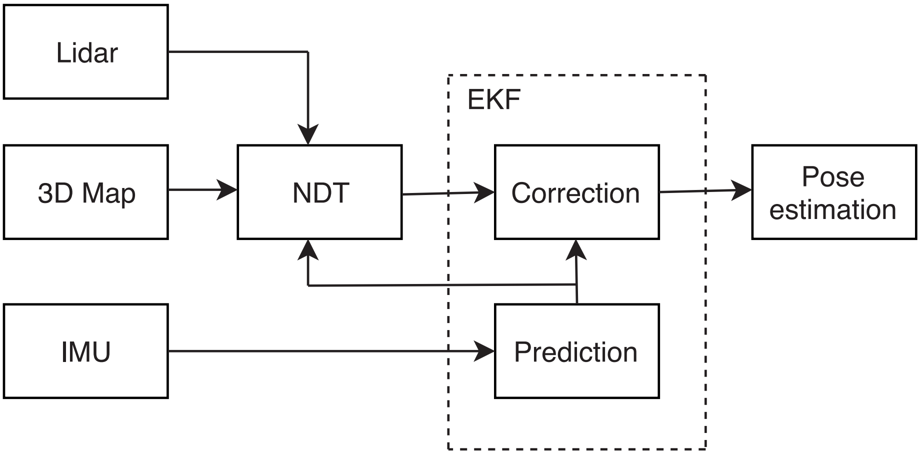


Figure I.3: Block diagram of localization system.

the prediction step is defined as follows:

$$\begin{aligned}\mathbf{x}_t =& [\mathbf{p}_{t-1} + \mathbf{v}_{t-1} \cdot \delta t, \mathbf{q}_{t-1} \cdot \delta \mathbf{q}_t, \mathbf{v}_{t-1}, \mathbf{b}^{\omega}_{t-1}]^T \\ \delta \mathbf{q}_t =& [\frac{\delta t}{2}\omega^x_t, \frac{\delta t}{2}\omega^y_t, \frac{\delta t}{2}\omega^z_t, 1] \\ \boldsymbol{\omega}_t =& \hat{\boldsymbol{\omega}}_t - \mathbf{b}^{\omega}_{t-1}\end{aligned} \tag{I.5}$$

where δt is a time step, $\delta \mathbf{q}_t$ is the rotation during δt with the bias-compensated angular velocity $\boldsymbol{\omega}_t$. The predicted pose $\mathbf{x}_t, \mathbf{q}_t$ are used as initial guess for the NDT process to match the observed point cloud to the global map. The correction step then uses the NDT estimation result to correct the final state estimation.

I.6 Experiments

We validate our proposed system on both simulated and real datasets. Here, we provide quantitative evaluations on: position drift while mapping, relocalization on previously built 3D LiDAR map and map quality comparison.

We first validate the proposed system using the simulation built on *gazebo*[2] for our project[3]. The simulated scene consists of two polytunnels and food processing storage. The scene is shown in Fig.I.6a. The simulated Thorvald robot is configured to physically match the real one. It is equipped with a simulated Velodyne VLP-16[4] and a 2D LiDAR Hokuyo. Currently, 2D LiDAR with *gmapping* SLAM[5] is used for building map. The de-facto AMCL[6] is

[2] http://gazebosim.org/

[3] https://rasberryproject.com/

[4] https://github.com/LCAS/velodyne_simulator

[5] http://wiki.ros.org/gmapping

[6] http://wiki.ros.org/amcl

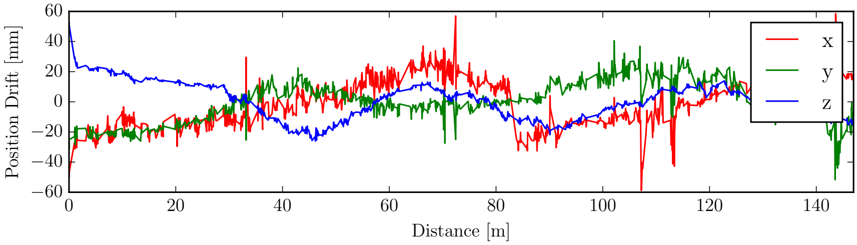


Figure I.4: Position drift when mapping in simulation.

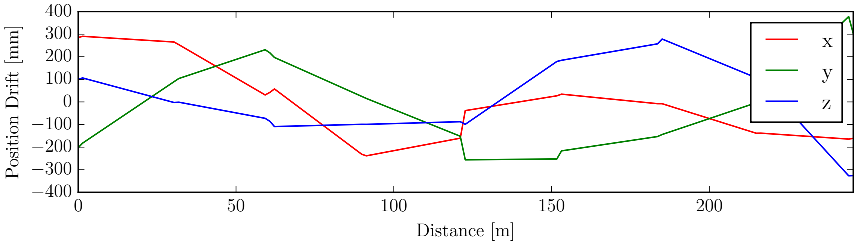
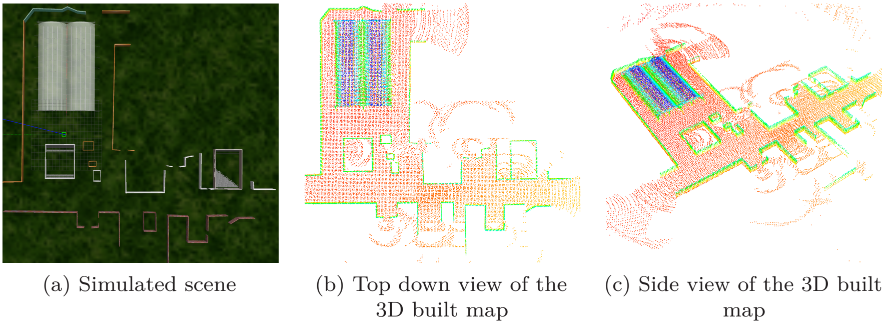


Figure I.5: Position drift when mapping in real scene.

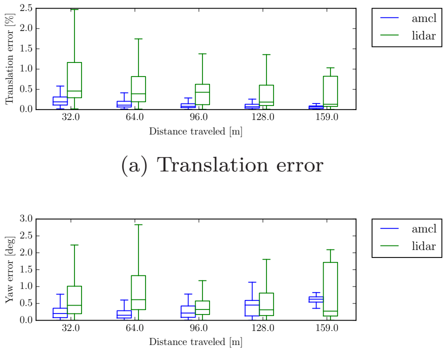


(a) Simulated scene

(b) Top down view of the 3D built map

(c) Side view of the 3D built map

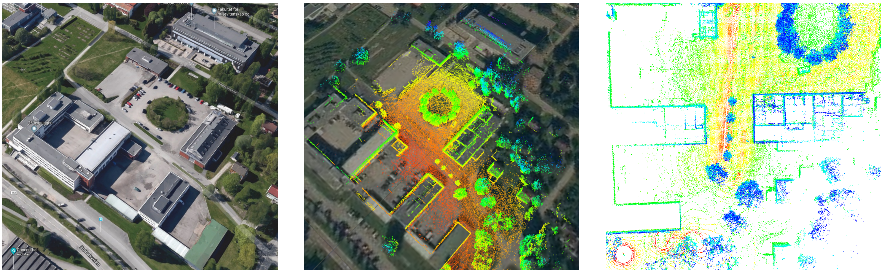
Figure I.6: Built map in simulation. Color indicates the height. Best view in color.



(a) Translation error

(b) Rotation error

Figure I.7: Quantitative localization comparison between *amcl* and proposed method in simulation. The ground truth is taken from *gazebo*.



(a) 3D view from Google map

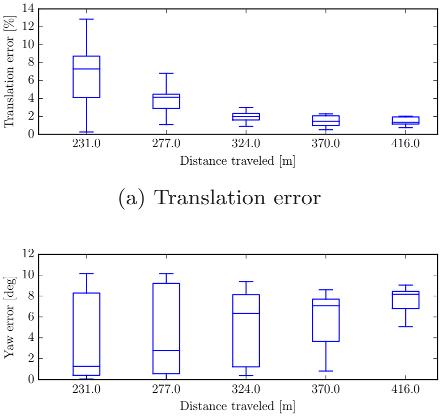
(b) Built 3D map overlay on top of Google map. Color indicates the height.

(c) Top down view of sharp corners of buildings. Color indicates intensity for viewing.

Figure I.8: Result of 3D mapping. Best view in color.

used for localization in a pre-built 2D map. Hence, we directly compare the localization results from two different sensor modalities and show that we can achieve comparable or better results. The ground truth is taken from *gazebo*.

In the simulation test, the robot is first manually driven around the scene while both *gmapping* and the proposed 3D LiDAR mapping are running to build the 2D and 3D map of the scene, respectively. The built maps are then saved for localization test. The 3D built map is shown in Fig.I.6b,I.6c. The 2D map is omitted here due to space constraint. After building maps, the robot is again driven manually through the scene using the previously built maps for localization. Both *AMCL* and the proposed localization method are running to estimate the robot pose. Both estimation results are recorded and analyzed following [30]. Position drift when mapping with LiDAR is shown in Fig.I.4. The relative errors in translation and rotation (yaw) are shown in Fig.I.7a,I.7b,



(a) Translation error

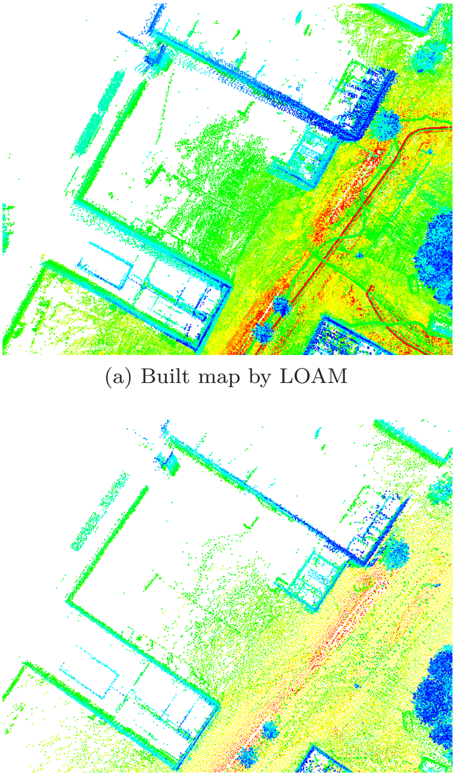
(b) Rotation error

Figure I.9: Quantitative localization comparison of proposed method with a real dataset.

respectively. The relative errors also show a consistently low median translational error of the proposed method (less than 0.5%). For relative rotation (yaw) errors, the proposed method shows a smaller median error in comparison with *amcl* for long trajectories.

We conduct another test with a real dataset. The robot (Fig.I.1b) is driven manually around our campus. Starting in front of our lab, which is served as a storage, the robot moves to our mockup polytunnel and back. The total trajectory is 500 meters. The ground truth in this test is obtained via a RTK-GNSS Septentrio AsteRx4 system. Notice that, we solely use the RTK-GNSS for ground truth comparison. To further challenge the proposed localization method, only *half* of the dataset is used for mapping. For localization in a built 3D map, the robot uses the whole dataset, in which half of the dataset contains LiDAR scans from the opposite moving direction when mapping. This mimics a scenario where we want to perform a fast mapping process and the robot can reliably use the built map for localization. We also achieve a low drift in position as shown in Fig.I.5. The relative pose error is shown in Fig.I.9. The robot achieves a small median translation error ($< 2\%$) for the whole trajectory. However, we encounter accumulating drift in rotation estimation, which is contributed by our assumption about constant bias of angular velocity and inconsistency of the EKF filter.

Finally, we compare the quality of the built map between our proposed method and the original LOAM. We follow [8] to calculate the *mean map entropy* (MME) from the mapped points $\mathcal{P} = \{p_1, ..., p_n\}$. The mean map entropy $H(\mathcal{P})$ is used as the *crispness/sharpness* metric of the map. A map with lower entropy



(a) Built map by LOAM

(b) Built map by our method

Figure I.10: Qualitative comparison of built maps. Color indicates intensities for viewing. Notice the difference of the building walls. Our method produces sharper map than LOAM's. Best view in color.

has higher quality. The mean map entropy is defined in Eq.I.6.

$$\begin{aligned} h(p_k) =& \frac{1}{2} \ln |2\pi e \Sigma(p_k)| \\ H(\mathcal{P}) =& \frac{1}{n} \sum_{k=1}^{n} h(p_k) \end{aligned} \tag{I.6}$$

where $h(p_k)$ is the entropy of the mapped point p_k, $\Sigma(p_k)$ is the sample covariance of the mapped point p_k in a local radius $r = 0.3m$ around p_k, and H($\mathcal{P}$) is averaged over all n mapped points.

The result of MME of built maps is listed in Table.I.1.

We illustrate the differences in quality of the built maps by two methods in Fig.I.10. Notice that, for fair comparison, we apply the same configuration for both methods, including using the same half of the dataset for mapping, the same downsample constants for point cloud filtering. LOAM retains more points

Table I.1: MME of built maps (lower is better)

Method	**MME**	**Map size**
LOAM	-0.19	27.7 MB
Ours	-0.22	8.2 MB

in its map but the quality of the map is lower than ours. Our method produces a sharper map with less memory consumption for storage.

The video of experiments is available online: https://youtu.be/05sTYF8AKaY

I.7 Conclusions

In this work, we propose a complete online 3D mapping and localization system for intelligent agricultural robots. Existing method, such as the state-of-the-art LOAM, primarily focuses on odometry estimation. We provide an additional localization method to make use of an accurate 3D built map, which is vital for an agricultural robot to work on a large scale farm without remapping before operating. The proposed system is tested using simulated and real datasets.

We notice, that by applying segmentation on input point clouds, we achieve more robust and better point cloud registration. Hence, future work involves further exploitation of point cloud segmentation to deal with dynamic environment. In addition, we plan to improve the localization system to further reduce drifts in rotation estimation caused by the inconsistency of the EKF filter.

I.8 Acknowledgement

We would like to thank Tixiao Shan and Igor Bogoslavskyi for fruitful discussions and code base.

References

[1] Albani, Dario, Nardi, Daniele, and Trianni, Vito. “Field coverage and weed mapping by UAV swarms”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 4319–4325.

[2] Bergerman, Marcel et al. “Robot farmers: Autonomous orchard vehicles help tree fruit production”. In: *IEEE Robotics & Automation Magazine* vol. 22, no. 1 (2015), pp. 54–63.

[3] Bogoslavskyi, I. and Stachniss, C. “Efficient Online Segmentation for Sparse 3D Laser Scans”. In: *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science* (2017), pp. 1–12.

[4] Bosse, Michael and Zlot, Robert. “Place recognition using keypoint voting in large 3D lidar datasets”. In: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE. 2013, pp. 2677–2684.

[5] Caselitz, Tim et al. “Monocular camera localization in 3d lidar maps”. In: *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE. 2016, pp. 1926–1931.

[6] Cop, Konrad P, Borges, Paulo VK, and Dubé, Renaud. “Delight: An efficient descriptor for global localisation using lidar intensities”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3653–3660.

[7] Dellaert, F. and Kaess, M. “Factor Graphs for Robot Perception”. In: *Foundations and Trends in Robotics, FNT* vol. 6, no. 1-2 (Aug. 2017). http://dx.doi.org/10.1561/2300000043, pp. 1–139.

[8] Droeschel, David, Stückler, Jörg, and Behnke, Sven. “Local multi-resolution representation for 6D motion estimation and mapping with a continuously rotating 3D laser scanner”. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 5221–5226.

[9] Fentanes, Jaime Pulido et al. “3D Soil Compaction Mapping through Kriging-based Exploration with a Mobile Robot”. In: *IEEE Robotics and Automation Letters* (2018).

[10] Grimstad, L. et al. “On the design of a low-cost, light-weight, and highly versatile agricultural robot”. In: *2015 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO 2015)*. 2015.

[11] Grimstad, Lars and From, Pål Johan. “Thorvald II - a Modular and Re-configurable Agricultural Robot”. In: *IFAC 2017 World Congress*. 2017.

[12] Guo, Jiadong et al. “Local Descriptor for Robust Place Recognition using LiDAR Intensity”. In: *arXiv preprint arXiv:1811.12646* (2018).

[13] Himmelsbach, Michael, Hundelshausen, Felix V, and Wuensche, H-J. “Fast segmentation of 3d point clouds for ground vehicles”. In: *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE. 2010, pp. 560–565.

[14] Kaess, Michael et al. “iSAM2: Incremental smoothing and mapping using the Bayes tree”. In: *The International Journal of Robotics Research* vol. 31, no. 2 (2012), pp. 216–235.

[15] Khanna, Raghav et al. “Beyond point clouds-3D mapping and field parameter measurements using UAVs”. In: *2015 IEEE 20th conference on emerging technologies & factory automation (ETFA)*. IEEE. 2015, pp. 1–4.

[16] Kohlbrecher, Stefan et al. “A flexible and scalable slam system with full 3d motion estimation”. In: *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*. IEEE. 2011, pp. 155–160.

[17] Lehnert, Christopher F., McCool, Christopher, and Perez, Tristan. “Lessons Learnt from Field Trials of a Robotic Sweet Pepper Harvester”. In: *CoRR* vol. abs/1706.06203 (2017). arXiv: 1706.06203.

[18] Magnusson, Martin, Lilienthal, Achim, and Duckett, Tom. “Scan registration for autonomous mining vehicles using 3D-NDT”. In: *Journal of Field Robotics* vol. 24, no. 10 (2007), pp. 803–827.

[19] Nived, Chebrolu et al. “Robot localization based on aerial images for precision agriculture tasks in crop fields”. In: *Robotics and Automation (ICRA), 2019 IEEE International Conference on.* IEEE. 2019.

[20] Pfeifer, Johannes et al. “Towards automatic UAV data interpretation for precision farming”. In: *CIGR-AgEng conference. Aarhus, Denmark.* 2016.

[21] Popović, Marija et al. “Multiresolution Mapping and Informative Path Planning for UAV-based Terrain Monitoring”. In: *Intelligent Robots and Systems (IROS), 2017 IEEE International Conference on.* Vancouver, 2017.

[22] Ruckelshausen, Arno et al. “BoniRob: an autonomous field robot platform for individual plant phenotyping”. In: *Precision agriculture* vol. 9, no. 841 (2009), p. 1.

[23] Sakai, Takahiro et al. “Large-scale 3D outdoor mapping and on-line localization using 3D-2D matching”. In: *System Integration (SII), 2017 IEEE/SICE International Symposium on.* IEEE. 2017, pp. 829–834.

[24] Sammons, Philip J., Furukawa, Tomonari, and Bulgin, Andrew. “Autonomous Pesticide Spraying Robot for use in a Greenhouse”. In: *Proceedings of the Australian Conference on Robotics and Automation, Sydney, Australia.* 2005.

[25] Shan, Tixiao and Englot, Brendan. “LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE. 2018, pp. 4758–4765.

[26] Stoyanov, Todor et al. “Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations”. In: *The International Journal of Robotics Research* vol. 31, no. 12 (2012), pp. 1377–1393.

[27] Xiong, Y., From, P. J., and Isler, V. “Design and Evaluation of a Novel Cable-Driven Gripper with Perception Capabilities for Strawberry Picking Robots”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA).* 2018, pp. 7384–7391.

[28] Zhang, Ji and Singh, Sanjiv. “LOAM: Lidar Odometry and Mapping in Real-time.” In: *Robotics: Science and Systems.* Vol. 2. 2014, p. 9.

[29] Zhang, Ji and Singh, Sanjiv. “Low-drift and real-time lidar odometry and mapping”. In: *Autonomous Robots* vol. 41, no. 2 (2017), pp. 401–416.

[30] Zhang, Zichao and Scaramuzza, Davide. “A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry”. In: *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*. 2018.

Paper II

A Cost-Effective Global Localization System for Precision Agriculture Tasks in Polytunnels

Tuan Le, Jon Glenn Omholt Gjevestad, Pål Johan From

Accepted, to be presented at IEEE 16th International Conference on Automation Science and Engineering, Hongkong, August, 2020.

Abstract

Precision agriculture tasks demand highly efficient and accurate actions performed by autonomous robots. In order to carry out such actions, a prerequisite for a robot is to accurately localize itself in its working environments, such as crop fields, greenhouses, polytunnels, etc. Agricultural environments usually present unique challenges to localization tasks such as the highly repetitive structure of a polytunnel or a crop field leading to visual aliasing and changing appearance over time. This makes it challenging for the robot to localize. In this paper, we develop a 6DoF localization system for precision agriculture tasks in polytunnels. The system only requires a cost-effective stereo RGB-D camera and a prebuilt 3D map. The system allows an agricultural robot to robustly localize over multiple stages of a strawberry season despite the strong dynamic and changes in the appearance of the environment caused by growing plants. Experiments are carried out on a real strawberry polytunnel over a period of several weeks to evaluate the system. The results show that our system provides adequate localization accuracy for agricultural robots to perform tasks that require a high level of accuracy.

II.1 Introduction

Multi-wheel mobile robots [3, 9, 22, 23] and multi-rotor flying robots [2, 18, 19] are widely adopted in agriculture for a variety of precision tasks such as weeding, harvesting and crop monitoring. Ground robots are suitable for heavy-power tasks such as plowing and carrying UV lights, while aerial robots are more used for monitoring and surveillance.

All authors are with Faculty of Science and Technology, Norwegian University of Life Sciences.

Regardless of the platform, an autonomous robot has to localize itself accurately to navigate and perform its assigned tasks in a given environment. Agricultural environments usually exhibit highly repetitive structures such as similar crop rows on an open field or similar table-top rows in a polytunnel. Fig. II.2 depicts an example of one such environment. Repetitive structures give rise to the well-known computer vision problem of visual aliasing. This causes strong ambiguity in determining the robot location. Another inherent problem of agricultural scenes is that the appearance of the scene is gradually changed over time as plants grow. These problems make the localization task more challenging over extended periods of time, which is a requirement of autonomous agricultural robot - robust long term operations.

Currently, high precision real time kinematic GNSS systems can provide highly accurate localization in open fields. However, such systems might suffer in GNSS-denied environments such as polytunnels or greenhouses. Other localization methods relies on visual features such as ORB [21] or other hand-crafted features [12, 13] tends to fail when dealing with appearance-changing environments.

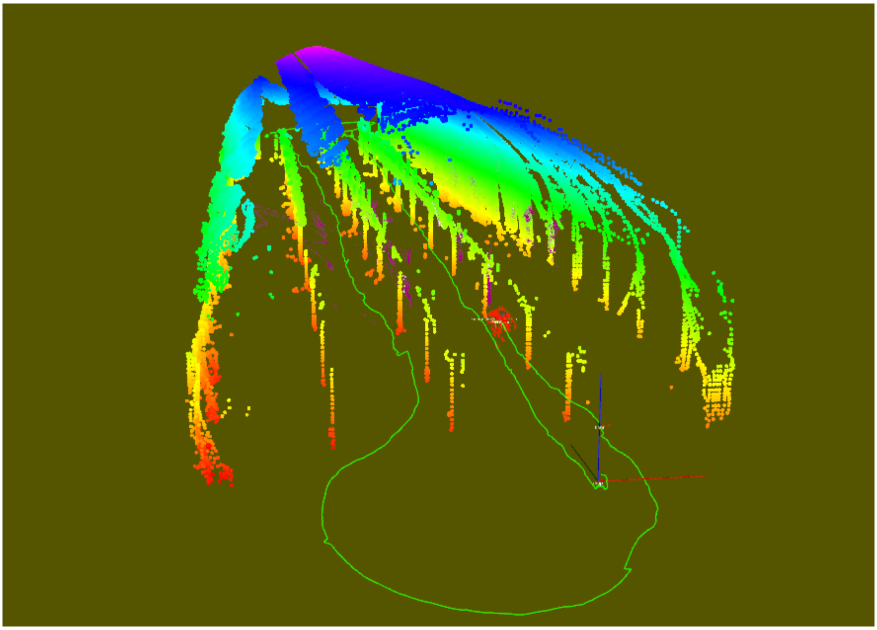


Figure II.1: Localization in polytunnels. The red dots denote a particle set. Green line denotes the robot trajectory.

A common structure of a typical strawberry polytunnel consists of several evenly-spaced sets of poles, on top of which hold table-trays. Strawberry plants are grown in plastic pots and placed on top those table-trays. The plant-growing area is covered by polymer material. The polytunnel provides optimum conditions for strawberry plants to grow [10] and therefore, their demand is growing on the market.

In this paper, we propose a localization approach for autonomous agricultural robots operating in strawberry polytunnels. Our system only requires a cost-effective stereo depth camera (Intel RealSense D435i), which is a stereo visual inertial odometry sensor with additional depth sensing module. We also assume that a 3D reference map of the polytunnel is available. A reference map can be obtained by performing an offline mapping process once. The main idea of our method is to exploit the semantic information of the polytunnel to serve as invariant features across a strawberry season, even in scenario where the environment undergoes substantial changes through-out the growing season. We extract the poles' shape to capture the inherent geometry of the polytunnel and use it as the invariant feature for measuring.

The main contribution of this paper is a cost-effective global localization system for agricultural robots deployed in polytunnels. Our system is able to:

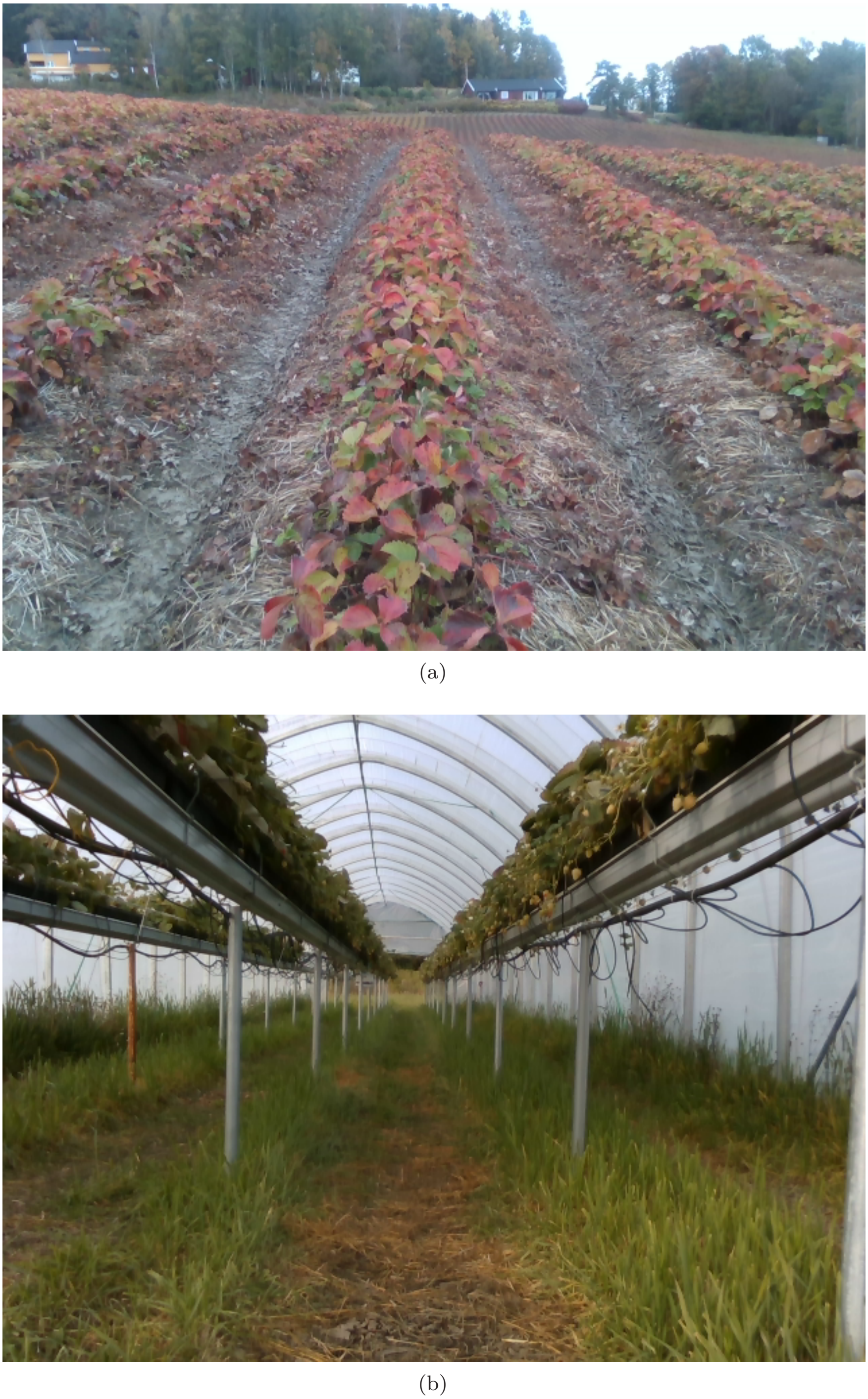
- localize with the required accuracy for the robot to navigate between table-top rows in strawberry polytunnels
- provide an alternative method to GNSS-based localization system which might suffer from signal outage in GNSS-denied environments such as indoor polytunnels
- perform robust localization over extended periods of time across plant season without remapping the environment.

The paper is organized as follows: In section II, we review related work. Section III presents our localization method. Experimental results are showed in Section IV and conclusions are discussed in Section V.

II.2 Related Work

A particle filter (PF), usually referred as Monte-Carlo localization (MCL), is a well-studied method for mobile robots. Thrun et al. [24] introduced MCL and analyzed its performance for planar, laser-based equipped mobile robot using a 2D occupancy grid maps. While most MCL method is intended to use with laser-base scanner sensor, Dellaert et al. [5] was the first to introduced vision-based MCL for mobile robots. Wolf et al. [26] proposed a MCL framework for a vision guided robot that extracts and matches invariant features from images but also requires a 2D occupancy grid maps for visibility computations.

Incorporating RGB-D cameras into a MCL framework enables 6DoF localization in a 3D map. Fallon et al. [7] proposed a 6DoF MCL system using RGB-D camera. This method heavily relied on planar features of the environment for the PF to converge. An interesting work by Winterhalter et al. [25] also proposed a 6DoF MCL system for indoor localization leveraging a RGB-D camera with a 2D floor plan. The authors generate a 3D map by constructing 3D walls as vertical planes, floors and ceilings as horizontal planes whose geometric characteristics are available from a given 2D floor plan. Other



(a)

(b)

Figure II.2: Different environments to grow strawberry (a) on an open field, (b) in a polytunnel.

work such as [4, 14] employed a prior global 3D pointcloud map and used Iterative Closest Point (ICP) based method to match current pointcloud to the global map. [17] transformed the point-based 3D map into a Normal Distribution Transform (NDT) representation while [6] chose a Gaussian Mixture Model (GMM) representation of the 3D map. The reason behind these choices of different map representations is to overcome the inaccuracy representation of the traditional grid-based map, which are discretized by definition [24].

Localization in agricultural domain has received little attention from the robotic community. Even though MCL method can naturally deal with sparse feature indoor environments such as long corridor by considering multiple hypotheses, its application in agriculture with similar ambiguous representation (repetitive and/or sparse features) is limited. We are inspired with the recent work by Chebrolu et al., [16], where the authors incorporated a semantic exploitation scheme into a MCL framework for their mobile robot to localize on a sugar beet field over a long period. The authors explained that the locations of plant stems and weeds, as well as the gaps between plants' clusters can be considered as “invariant" features of the map to build their MCL framework. In contrast, we target a *totally* different type of agricultural field, polytunnels, where those features are much harder to detect and track (plants are grown in trays on a table top, which significantly eliminate the existence of weeds and gaps). Hence, we exploit a different type of semantic features in the polytunnel to tackle the ambiguity problem.

II.3 Global Localization in A Prior 3D Map

II.3.1 A prior 3D reference map

In this work, we do not aim to solve a SLAM problem. As it is inefficient to perform mapping and localization every time we assign tasks to a robot, we assume we already have a built 3D map by using our previous work [11] or any other existing methods such as employing a terrestrial laser scanner. The reference 3D map of our polytunnel used in experiments is shown in Fig. II.3.

II.3.2 Poles as stable landmarks

Similar to open field scenarios, visual features detected using hand-crafted descriptors such as SIFT, ORB or BRISK are not consistent over the season due to large differences in the appearance of the plants over the crop season. Hence, we deliberately choose the poles in a polytunnel as a type of consistent landmarks since they are not physically changed across seasons. In order to detect poles, we use an end-to-end trainable convolutional neural network (CNN) and fine-tune with our annotated image dataset. We use the popular framework as described in [15] and omit the detailed description here. The output of the network is a mask image of poles and we use it to extract depth information of detected poles from the corresponding-aligned depth image. Notice that depth-images are aligned and synchronized with RGB images on hardware-level. The extracted

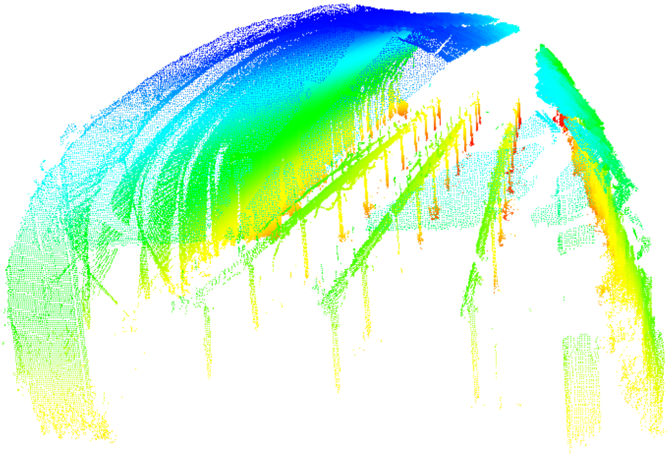


Figure II.3: The reference 3D map. Point's size is inflated for clarity.

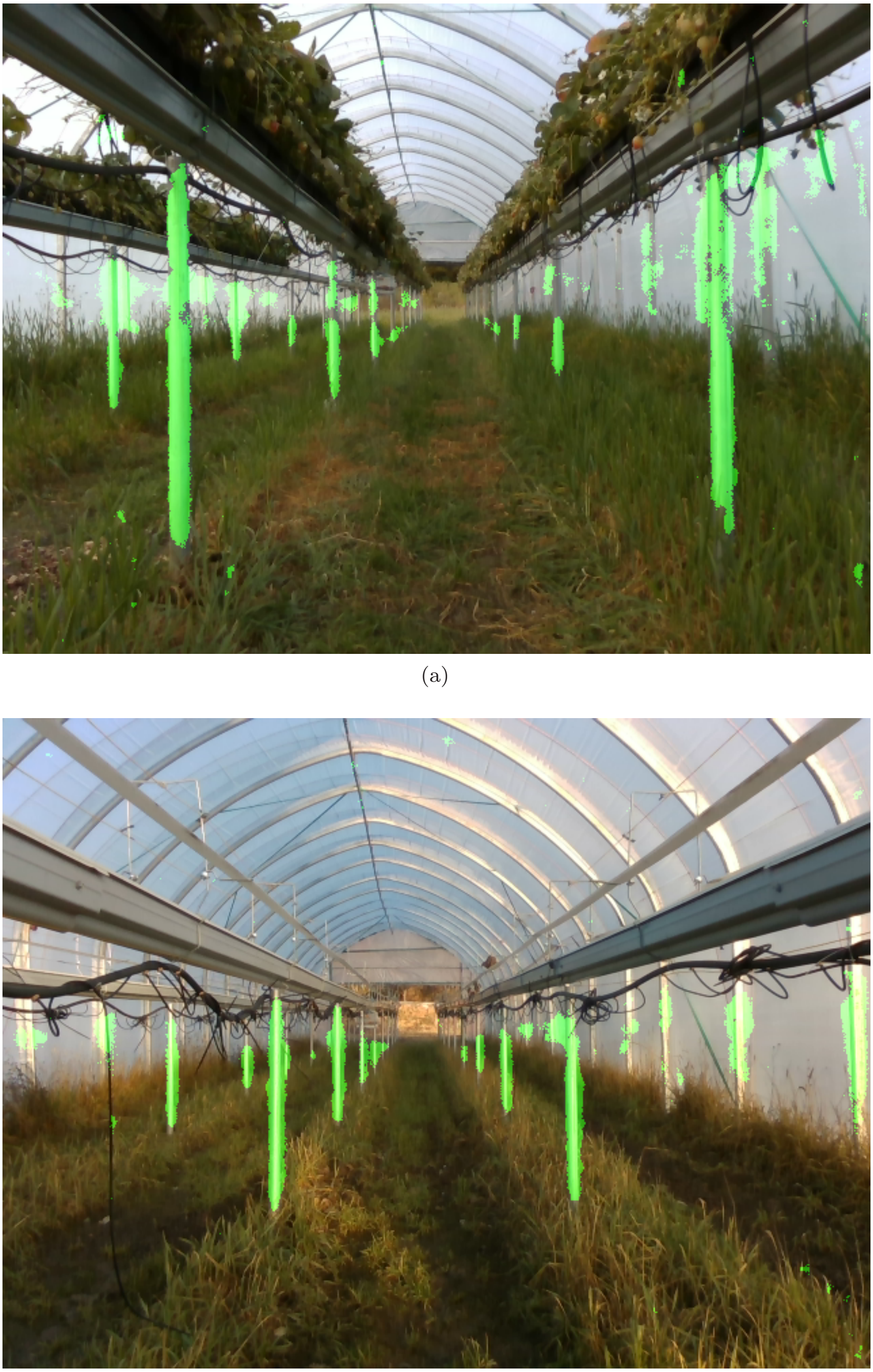
depth information of poles serve as measurements in the MCL framework that will be discussed next. Examples of poles detection are shown in Fig. II.4.

II.3.3 Monte Carlo Localization

A recursive Bayesian filtering scheme is used to estimate the robot's pose $\mathbf{x} = (x, y, z, \varphi, \vartheta, \psi)$ in its environment. We prefer this probabilistic localization to deal with the repetitive structural nature of the polytunnel. The main idea is to maintain a probability density $p(\mathbf{x}_t|\mathbf{z}_{1:t}, \mathbf{u}_{1:t}, m)$ of the robot's pose $\mathbf{x}_t$ at time t in the provided map of the environment m along with given sets of observations $\mathbf{z}_{1:t}$ and motion control commands $\mathbf{u}_{1:t}$ up to time t. This posterior is updated recursively as follows:

$$\begin{split} p(\mathbf{x}_t|\mathbf{z}_{1:t}, \mathbf{u}_{1:t}, m) \propto \eta p(\mathbf{z}_t|\mathbf{x}_t, m) \cdot \\ \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t) p(\mathbf{x}_{t-1}|\mathbf{z}_{t-1}, \mathbf{u}_{t-1}, m) d\mathbf{x}_{t-1} \end{split} \tag{II.1}$$

The motion model $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t)$ denotes the probability of state $\mathbf{x}_t$ given the motion command $\mathbf{u}_t$ in state $\mathbf{x}_{t-1}$. The sensor model $p(\mathbf{z}_t|\mathbf{x}_t, m)$ denotes the likelihood of getting the observation $\mathbf{z}_t$ with the pose $\mathbf{x}_t$ and the map m. η is the normalization constant. To implement the filter, we follow the sample-based approach described in [24]. The belief update as described in Eqn. II.1 is executed by the following two alternating steps: 1) a prediction step, where we draw for each particle with weight $w^{[i]}$ a new particle according to $w^{[i]}$ and



(a)

(b)

Figure II.4: Poles detection at different moment of time: (a) session 2, (b) session 4.

to the prediction model $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t)$ and 2) a correction step, where a new observation $\mathbf{z}_t$ arrives, a new weight $w^{[i]}$ is assigned to each particle according to the sensor model $p(\mathbf{z}_t|\mathbf{x}_t, m)$.

MCL maintains a set of weighted particles to represent the belief about the system and update the belief by sampling from the motion model when new odometry measurements arrive. The weight of each particle is calculated proportionally to the observation likelihood of the measurement given the corresponding state of the particle. The particle set need to be resampled according to the assigned weights to obtain a good approximation of the pose distribution with a finite number of particles.

Next, we discuss our choice of estimating the 6DoF absolute localization. The first reason is that we aim to develop a cost-effective system. We want it to be generic and platform-agnostic, meaning it is applicable on both ground and aerial platforms. Second, for the environment that we target it almost always exhibit non-planar terrain. Hence, a design choice of conventional 3DoF (translation in x, y and yaw rotation) is inefficient.

II.3.4 Motion model

Fig. II.5 shows our robot with the sensor setup. The Intel RealSense D435i camera is a cost-effective stereo visual inertial (VI) sensor with an additional depth sensor module. We assume that the VI sensor is calibrated and the relative transformation from the robot base to the camera $\mathbf{T}_B^C$ is known. Hence, the local pose estimation in the VI sensor frame can be transformed to the robot base. For local pose estimation, we adopt the existing joint optimization-based visual inertial odometry (VIO) algorithm described in [20].

The VIO estimates a 6DoF poses in IMU frames and features' depth within a sliding window. For the local pose estimation, we denote the states as follows:

$$\begin{aligned} \mathcal{S}_l &= [\mathbf{s}_0, \mathbf{s}_1, \cdots, \mathbf{s}_n, \lambda_0, \lambda_1, \cdots, \lambda_m] \\ \mathbf{s}_k &= [\mathbf{p}_{b_k}^l, \mathbf{v}_{b_k}^l, \mathbf{q}_{b_k}^l, \mathbf{b}_a, \mathbf{b}_g], k \in [0, n] \end{aligned} \tag{II.2}$$

where the k-th IMU state $\mathbf{s}_k$ includes the position $\mathbf{p}_{b_k}^l$, velocity $\mathbf{v}_{b_k}^l$, orientation $\mathbf{q}_{b_k}^l$ of the center of the IMU with respect to the local reference frame l, $\mathbf{b}_a$ and $\mathbf{b}_g$ are accelerometer and gyroscope biases respectively. A reference frame is the first IMU pose. Detected features in stereo images are parameterized by their inverse depth λ when first observed in the camera frame. The pose estimation is solved as a nonlinear least square problem:

$$\begin{aligned} \min_{\mathcal{S}_l} \Bigg\{ \left\| \mathbf{r}_p - \mathbf{H}_p \mathcal{S} \right\|^2 + \sum_{k \in \mathcal{B}} \left\| \mathbf{r}_{\mathcal{B}}(\hat{\mathbf{z}}_{b_{k+1}}^{b_k}, \mathcal{S}) \right\|_{\mathbf{P}_{b_{k+1}}^{b_k}}^2 + \\ \sum_{(l,j \in \mathcal{C})} \rho\Big(\left\| \mathbf{r}_{\mathcal{C}}(\hat{\mathbf{z}}_l^{c_j}, \mathcal{S}) \right\|_{\mathbf{P}_{b_l}^{c_j}}^2 \Big) \Bigg\} \end{aligned} \tag{II.3}$$

where $\mathbf{r}_{\mathcal{B}}(\hat{\mathbf{z}}_{b_{k+1}}^{b_k}, \mathcal{S})$ and $\mathbf{r}_{\mathcal{C}}(\hat{\mathbf{z}}_l^{c_j}, \mathcal{S})$ denote inertial and visual residuals respectively. $\mathbf{r}_p, \mathbf{H}_p$ are prior terms containing information about past marginalized states.

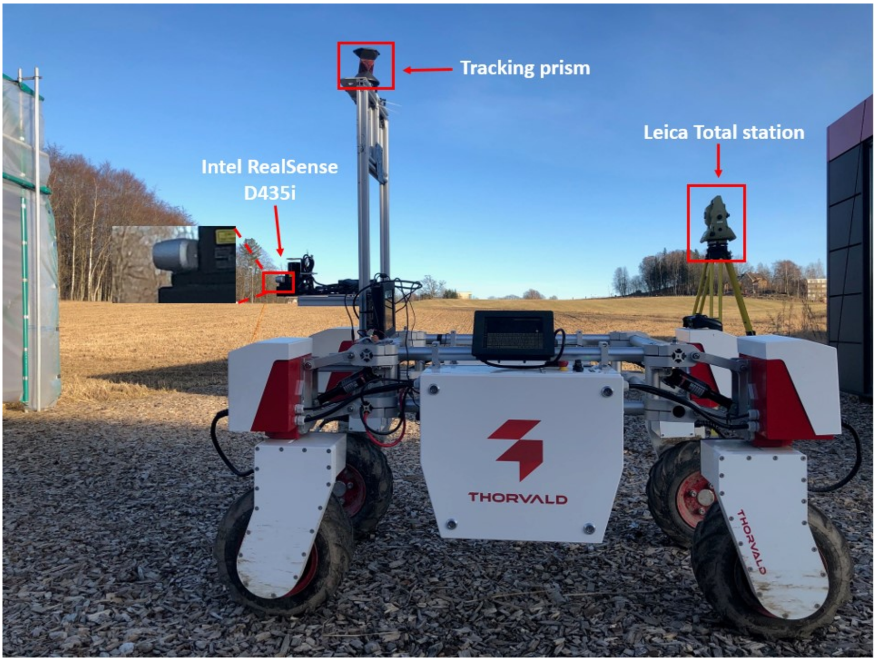


Figure II.5: Robot setup for experiments.

$\rho(\cdot)$ is a robust Huber norm and $\mathcal{C}$ is the set of features that have been observed at least twice in the current sliding window. The Ceres solver [1] is used for solving this problem. Notice that the estimated odometry is slowly drifted but it can be corrected by the global localization.

The VIO provides the odometry measurement of the robot. However, we also need to model the uncertainty of the odometry estimation, hence we directly corrupt the odometry estimation with a small amount of normally distributed noise. We note that because of our choice of joint optimization-based method of odometry estimation, we cannot directly obtain the uncertainty measurement, i.e covariance estimation. Instead, one can choose to implement a filter-based method so that the covariance estimation can be used as uncertainty measurement for the motion model.

Finally, we can express our motion model as:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{u}_t + \mathbf{e}_t \quad \mathbf{e}_t \sim N(0, \sigma_m^2) \tag{II.4}$$

where the motion control $\mathbf{u}_t$ is the relative transformation estimated from the VI sensor.

Note that in practice, depending on the type of the robot and the ground floor, the motion noise standard deviation σ_m should be chosen accordingly, i.e for smooth and continuous motions on a fairly even terrain, a small noise is adequate. While for fast motions or traversing on a rough terrain, the noise should be inflated.

II.3.5 Observation Model

In this section, we derive our sensor model to determine the likelihood of a measurement z given the pose x in the map m.

The raw range measurements likely contains many false positives, i.e from leaves, fruits, water pipes etc., Hence, we extract only the depth information of the poles, which are detected by our trained CNN network and use them as measurements. When a new measurement arrives, K range measurements are randomly sampled from the pole-depth image and converted into a measurement 3D pointcloud $\mathcal{Z}$. For computational efficiency, we apply the endpoint observation model described by Thrun *et al.* in [24].

We denote z_j as the j-th measurement of $\mathcal{Z}$. We model the likelihood of an observed measurement as a Gaussian distribution. The likelihood of a single depth measurement based on the scan point z'_j corresponding to z_j transformed into the map frame with the robot pose x and on the closest corresponding point in the map $m_j \in m$:

$$p(z_j|x, m_j) = f(z'_j, m_j) = \frac{1}{\sqrt{2\pi}\sigma_d} exp\Big(- \frac{(z'_j)^2}{2\sigma_d^2}\Big) \tag{II.5}$$

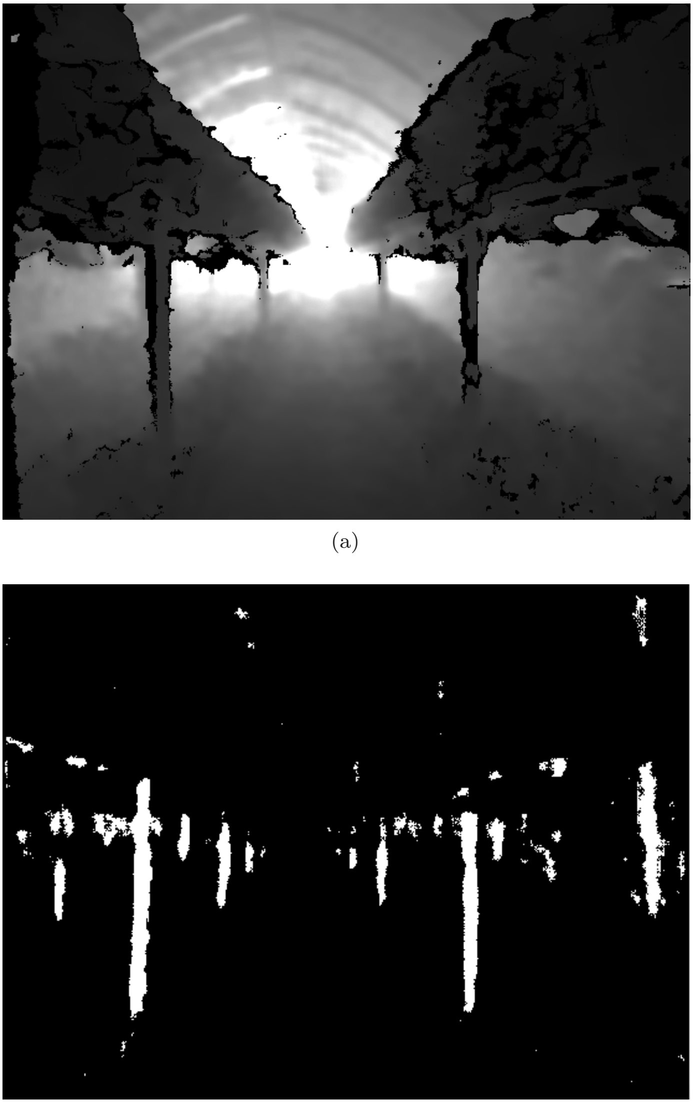
where σ_d is the standard deviation of the sensor depth noise.

We assume all beams are independent thus the integration of one full measurement is computed as the product of the each beam likelihood:

$$p(z|x, m) = f((z'_1, m_1), \cdots, (z'_K, m_K)) \propto \prod_{j=1}^{K} f(z'_j, m_j) \tag{II.6}$$

Note that unlike the conventional laser range finder sensor, which can give highly accurate range measurement, the depth measurements from a RGB-D camera are noisier with increasing measurement distances. Hence, we increase the value of σ_d along with the measurement distance to account for this characteristic in our observation model.

We want to highlight here the semantic exploitation of our approach. In other works that also use RGB-D camera for MCL [7, 25], the authors relied on dense-depth images to randomly sample depth measurements from wall, ground floors, etc., This approach is not suitable for our case since the polytunnel environment lacks those features, i.e the wall is not always visible and the ground floor is covered with grass and depth images are dominated by plant as shown in Fig. II.6. Hence, by extracting only the depth measurement from poles, we guarantee that all sampled depth measurements do not contain unreliable measurements, i.e avoiding depth measurements from leaves, fruits, etc., which are subjected to change daily.



(a)

(b)

Figure II.6: A typical example of a depth image dominated by plant (a) and its extracted poles mask in binary (b). This was recorded in session 3, mid-season.

II.4 Experiments

II.4.1 Experimental results

We performed the experiments at our research polytunnel, where we grew three rows of strawberry plants. The dimension of the polytunnel is 30 m by 5 m. The off-line built 3D reference map of the polytunnel is shown in Fig.II.3.

The robot is equipped with an Intel RealSense camera D435i. We calibrate the depth sensor module with high accuracy and medium resolution preset [1]. Depth images are aligned with RGB images on camera hardware level. Depth measurement error of the D435i is typically less than 1cm with measurement range up to 1.5m and quickly raises to 3.5cm error with measurement range up to 3m. Hence, we give a small noise σ_d for range data that is less than 1.5m and triple the noise value for range data greater than 1.5m. We simply discard range measurements that are more than 4m.

For visual inertial odometry estimation, we use two mono image streams from the infrared cameras with emitter module turned off. The integrated IMU provides synchronous measurements with the mono image streams. For training the CNN network, we collected and manually labeled images of our polytunnel for poles using the same camera. As CNN is considered as a *off-the-shelf* product due to its popularity, we omit the details of our training process here. We achieved a mean accuracy and a mean Jaccard index (mIoU) as 98.8% and 84.8%, respectively, from our trained network. We observe that our trained network has difficulty for segmenting *far-off* poles. However, since we discard depth measurements that are greater than 4 meters, as we will discuss in detail later, this level of segmentation accuracy does not affect our localization system.

We recorded four datasets of our mobile robot traversing the polytunnels at different time during the 2019 strawberry season. The first three datasets were approximately a week apart and the fourth was recorded at the end of the season. The robot was manually controlled with an average speed of 0.7m/s.

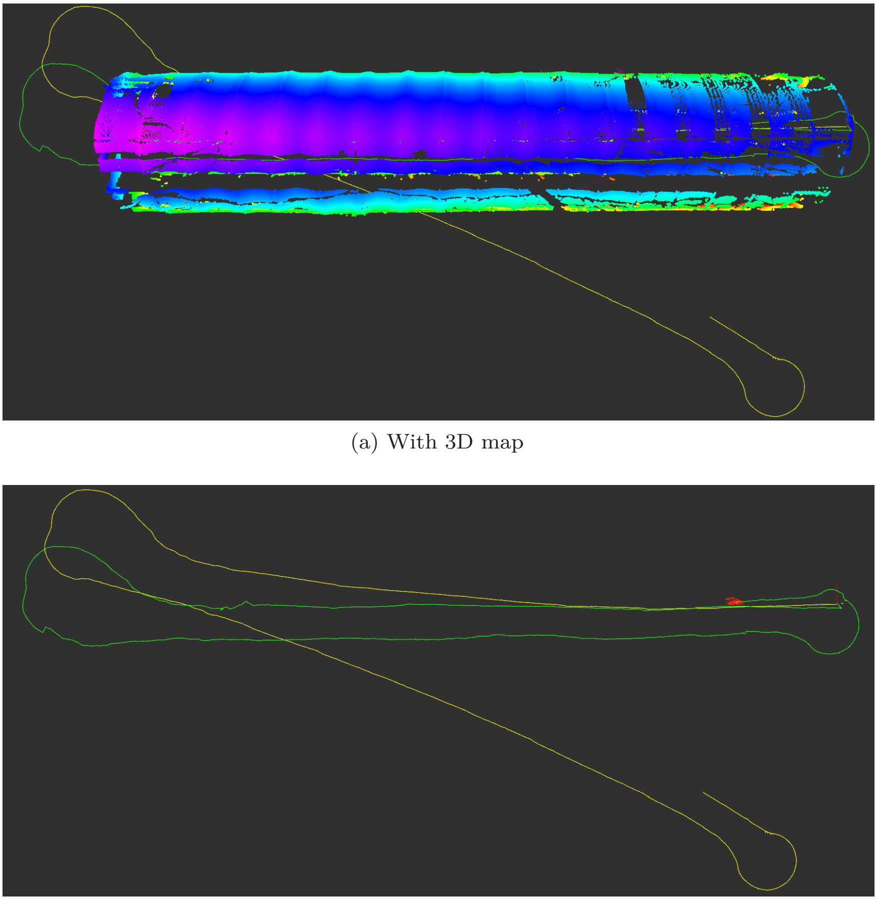
We initialize the filter with 1000 particles with an initial standard deviation of 1m around a starting position. Here we take advantage of our specific application of UV light treatment in a polytunnel, i,e our robot always starts up at a known position. Hence, we can obtain a very good estimation of the robot's first pose to initialize the MCL filter. This eliminates the needs of a high number of particles to cover the whole map, which in a 3D map may be intractable.

In Fig. II.7, II.8 we show qualitative results of a robot traversed through the polytunnel. It is easy to see that the VIO drifted quickly.

Notice that for our MCL framework, we only extracts the depth measurements of poles, hence, the particle filter diverges when the robot get out of the polytunnel, where it can not receive any meaningful measurements as shown in Fig. II.8a. But as soon as it rediscovers the poles, the particles converge again as shown in Fig. II.8b.

Unfortunately it is impossible to directly obtain 6DoF ground truth of trajectory for localization accuracy comparison. More over, the localization

[1] https://github.com/IntelRealSense/librealsense/wiki/D400-Series-Visual-Presets



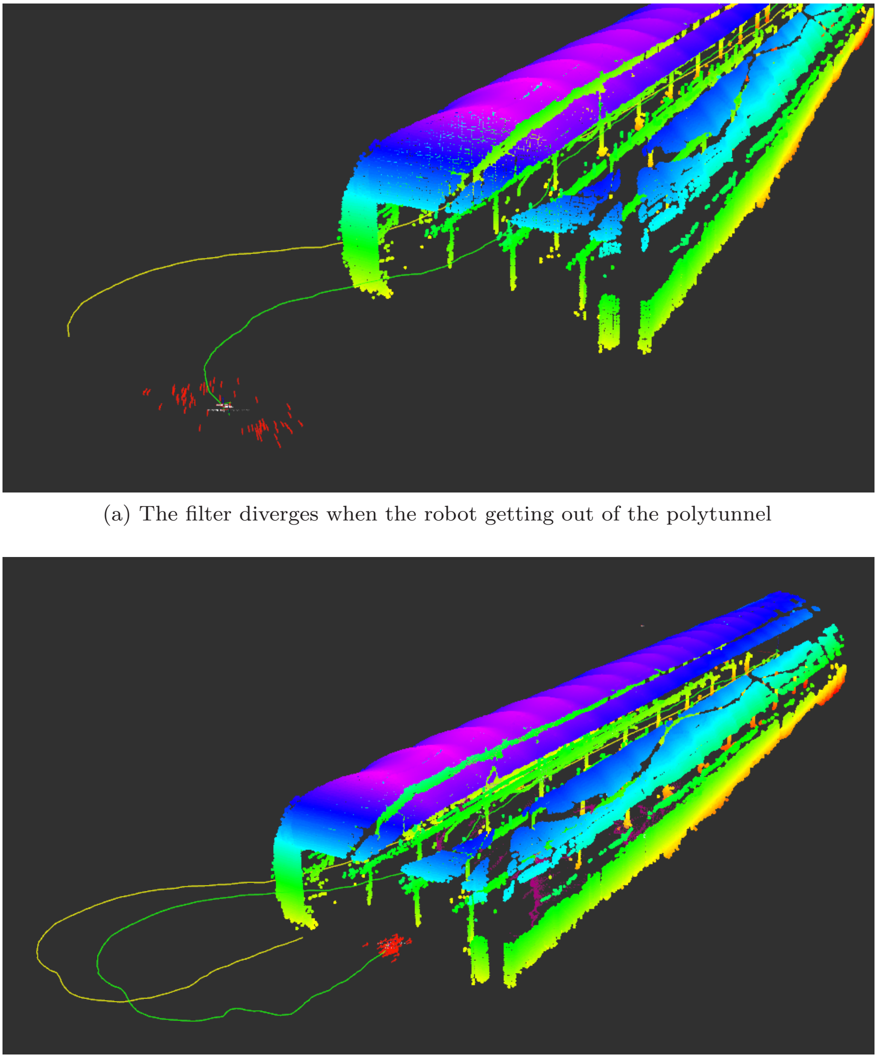
(a) With 3D map

(b) 3D map removed for clarity

Figure II.7: Odometry from VI sensor (yellow) drifts while the MCL (green) can maintain the robot's poses. Best viewed in color.

accuracy quickly degrades whenever the robot heads out of the polytunnel. Therefore, the best quantitative comparison of localization accuracy we can get is in xy-plane, where we use a Leica Total Station TCA 1100 to get the ground truth of the robot's positions inside the polytunnel.

Table II.1 shows the qualitative evaluation of localization during different recording sessions. Giving the width of a row is 1.5m and the robot's width is 1m, the robot localized within a row with a maximum error being 23cm. This error is less than the safety upper bound error, which is 25cm. It shows that the robot achieves the safety requirement to navigate inside the polytunnel.



(a) The filter diverges when the robot getting out of the polytunnel

(b) The particles converge when the robot receive measurements from poles

Figure II.8: MCL can recover the global localization.

Session	Distance	Mean error	Max error
1	75.2m	0.205m	0.225m
2	74.6m	0.194m	0.233m
3	74.1m	0.207m	0.227m
4	75.7m	0.211m	0.231m

Table II.1: Localization accuracy for different recording sessions.

II.4.2 Discussion

Even though our MCL approach shows promising results, it still has some drawbacks:

II.4.2.1 MCL implementation

The current implementation of the MCL is not optimized. A better sampling strategy such as a KLD sampling by Fox [8] would help by allowing particles generation on demand, while keeping the particle set small.

II.4.2.2 Defining noise characteristics

Relying only on depth measurements from poles gives a benefit of avoiding false measurements when sampling measurements from dense depth images. As a trade-off, we have less number of measurements and from experiments, the noise values in both motion model and observation model have greater impact on the localization accuracy. Currently, we define those noise values empirically. However, the tuning process mentioned in [25], where the authors determine the value of noise characteristics using a motion capture system can be applied.

And finally, since our CNN only detects poles, our system would not work for different types of polytunnels, such as those, where plant trays are being hung down from the ceiling.

A video of our experiments is available at: https://youtu.be/EL8uBg3nr6g

II.5 Conclusions

In this paper, we presented a cost effective localization system using only an Intel RealSense D435i camera. Our method avoids false measurements by exploiting the stable features of a polytunnels - poles. This allows us to localize successfully over multiple sessions of a strawberry season. We performed evaluations in a real polytunnel to demonstrate the effectiveness of our system. We have discussed some limitations of our current system. However, we argue that our system is still useful for precision tasks in agriculture such as UV light treatment. Future work involves developing a better initialization process for the MCL filter as well as adopting better sampling strategies.

References

[1] Agarwal, Sameer, Mierle, Keir, et al. “Ceres solver, 2013”. In: *URL http://ceres-solver. org* (2018).

[2] Albani, Dario, Nardi, Daniele, and Trianni, Vito. “Field coverage and weed mapping by UAV swarms”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 4319–4325.

[3] Bergerman, Marcel et al. “Robot farmers: Autonomous orchard vehicles help tree fruit production”. In: *IEEE Robotics & Automation Magazine* vol. 22, no. 1 (2015), pp. 54–63.

[4] Bry, Adam, Bachrach, Abraham, and Roy, Nicholas. “State estimation for aggressive flight in GPS-denied environments using onboard sensing”. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 1–8.

[5] Dellaert, Frank et al. “Using the condensation algorithm for robust, vision-based mobile robot localization”. In: *Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (cat. no pr00149)*. Vol. 2. IEEE. 1999, pp. 588–594.

[6] Dhawale, Aditya, Shaurya Shankar, Kumar, and Michael, Nathan. “Fast monte-carlo localization on aerial vehicles using approximate continuous belief representations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5851–5859.

[7] Fallon, Maurice F, Johannsson, Hordur, and Leonard, John J. “Efficient scene simulation for robust Monte Carlo localization using an RGB-D camera”. In: *2012 IEEE international conference on robotics and automation*. IEEE. 2012, pp. 1663–1670.

[8] Fox, Dieter. “Adapting the sample size in particle filters through KLD-sampling”. In: *The international Journal of robotics research* vol. 22, no. 12 (2003), pp. 985–1003.

[9] Grimstad, L. et al. “On the design of a low-cost, light-weight, and highly versatile agricultural robot”. In: *2015 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO 2015)*. 2015.

[10] Ingram, David S, Vince-Prue, Daphne, and Gregory, Peter J. *Science and the garden: the scientific basis of horticultural practice*. John Wiley & Sons, 2015.

[11] Le, Tuan, Gjevestad, Jon Glenn Omholt, and From, Pål Johan. “Online 3D Mapping and Localization System for Agricultural Robots”. In: *IFAC-PapersOnLine* vol. 52, no. 30 (2019), pp. 167–172.

[12] Leutenegger, Stefan, Chli, Margarita, and Siegwart, Roland Y. “BRISK: Binary robust invariant scalable keypoints”. In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2548–2555.

[13] Lowe, David G. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* vol. 60, no. 2 (2004), pp. 91–110.

[14] Maier, Daniel, Hornung, Armin, and Bennewitz, Maren. “Real-time navigation in 3D environments based on depth camera data”. In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE. 2012, pp. 692–697.

[15] Milioto, A., Mandtler, L., and Stachniss, C. “Fast Instance and Semantic Segmentation Exploiting Local Connectivity, Metric Learning, and One-Shot Detection for Robotics ”. In: *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*. 2019.

[16] Nived, Chebrolu et al. “Robot localization based on aerial images for precision agriculture tasks in crop fields”. In: *Robotics and Automation (ICRA), 2019 IEEE International Conference on*. IEEE. 2019.

[17] Oishi, Shuji et al. “ND voxel localization using large-scale 3D environmental map and RGB-D camera”. In: *2013 IEEE international conference on robotics and biomimetics (ROBIO)*. IEEE. 2013, pp. 538–545.

[18] Pfeifer, Johannes et al. “Towards automatic UAV data interpretation for precision farming”. In: *CIGR-AgEng conference. Aarhus, Denmark*. 2016.

[19] Popović, Marija et al. “Multiresolution Mapping and Informative Path Planning for UAV-based Terrain Monitoring”. In: *Intelligent Robots and Systems (IROS), 2017 IEEE International Conference on*. Vancouver, 2017.

[20] Qin, Tong, Li, Peiliang, and Shen, Shaojie. “Vins-mono: A robust and versatile monocular visual-inertial state estimator”. In: *IEEE Transactions on Robotics* vol. 34, no. 4 (2018), pp. 1004–1020.

[21] Rublee, Ethan et al. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2564–2571.

[22] Ruckelshausen, Arno et al. “BoniRob: an autonomous field robot platform for individual plant phenotyping”. In: *Precision agriculture* vol. 9, no. 841 (2009), p. 1.

[23] Sammons, Philip J., Furukawa, Tomonari, and Bulgin, Andrew. “Autonomous Pesticide Spraying Robot for use in a Greenhouse”. In: *Proceedings of the Australian Conference on Robotics and Automation, Sydney, Australia*. 2005.

[24] Thrun, Sebastian. “Probabilistic robotics”. In: *Communications of the ACM* vol. 45, no. 3 (2002), pp. 52–57.

[25] Winterhalter, Wera et al. “Accurate indoor localization for RGB-D smartphones and tablets given 2D floor plans”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 3138–3143.

[26] Wolf, Jürgen, Burgard, Wolfram, and Burkhardt, Hans. “Robust vision-based localization for mobile robots using an image retrieval system based on invariant features”. In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*. Vol. 1. IEEE. 2002, pp. 359–365.

Paper III

A Low-Cost and Efficient Autonomous Row-Following Robot for Food Production in Polytunnels

Tuan Dung Le*, Vignesh Raja Ponnambalam*, Jon Glenn Omholt Gjevestad, Pål Johan From

Published in *Journal of Field Robotics*, March 2020, Volume 37, Issue 2, pp. 309–321. DOI: 10.1002/rob.21878.

Abstract

In this paper, we present an automatic motion planner for agricultural robots that allows us to set up a robot to follow rows without having to explicitly enter waypoints. In most cases, when robots are used to cover large agricultural areas, they will need waypoints as inputs, either as pre-measured coordinates in an outdoor environment, or as positions in a map in an indoor environment. This can be a tedious process as several hundreds or even thousands of waypoints will be needed for large farms. In particular, we find that in unstructured environments such as the ones found on farms, the need for waypoints increases. In this paper, we present an approach that enables robots to safely traverse not only between straight rows but also through curved rows without the need for any predetermined waypoints. Most types of infrastructure found in agriculture, such as polytunnels, are built on uneven terrain, thus containing a mix of straight and curved plant rows, for which traditional methods of row following will fail. Different from traditional approaches of row following that only consider straight-line-of-sight rows, our approach identifies the rows on each side with the goal of staying in the middle of the rows, even if the rows are not straight. Waypoints are only needed on the very extreme of the rows, and these will be automatically generated by the system. With our approach, the robot can just be placed in the corner of the field and will then determine the trajectory without further input from the user. We thus obtain an approach that can reduce the installation time from potentially hours to just a matter of minutes. The final autonomous system

*These authors contributed equally to this work. All authors are with Faculty of Science and Technology, Norwegian University of Life Sciences.

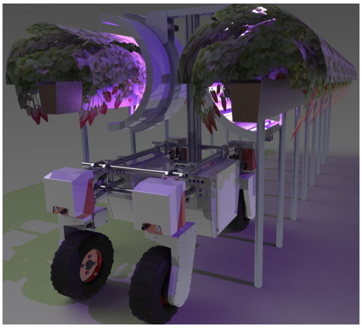


Figure III.1: A design model of the Thorvald robot carrying UV light bulbs

is low cost and efficient for various tasks that requires moving between plant rows inside a polytunnel. Several experiments were performed and the robot demonstrates 1.4% position drift over 21 meters of navigation path.

III.1 Introduction

In this paper, we address the problem of autonomous row following for an agricultural robot in a tightly constrained space such as polytunnels. This work is part of a larger project [1], in which we develop agricultural robots to automate food production [9, 10, 12]. The Thorvald II robot has been used for different purposes in food production such as phenotyping [13] and strawberry picking [26]. The Thorvald II robot is a highly versatile robot due to its unique modular design [2]. The robot for example can be retrofitted to carry UV light bulbs for UV light treatment tasks, as shown in Fig. III.1. Currently, the model robot has been actively employed at a cucumber greenhouse to provide UV-light treatment [11], in addition to strawberry polytunnels.

In this paper we address the problem of autonomous navigation in commonly found agricultural domains such as polytunnels or greenhouses. A polytunnel/greenhouse is a structured agricultural environment, where plants are grown in trays, which are organized as rows on top of several poles along the polytunnel. The rows are evenly spaced and spanned across the polytunnel and create a tightly constrained environment. For polytunnel-related tasks, the robot is usually required to navigate between plant rows. In a tightly constrained space such as polytunnels, curved rows make navigation more challenging.

We specifically aim to develop a low-cost and efficient autonomous system that is able to traverse through a polytunnel while performing assigned tasks without human intervention. The robot is equipped only with a planar laser scanner. The 2D laser scanner exploits the structured environment to provide navigation cues for the robot. In order to move along a row, a carefully designed

[1] https://rasberryproject.com/

[2] https://sagarobotics.com/

RANSAC algorithm [8] is used to filter laser scans and reliably detect two parallel straight lines, which represent a part of the plant row on both sides of the robot. Note that a row comprises of several straight lines locally, which together form a curved row. A pure pursuit controller is implemented to make the robot follow the row. When the laser scanner cannot detect any parallel lines, the robot assumes it has reached the end of a row. It then switches to row transition mode to turn and enter the next row. The proposed navigation method has been tested in both simulations and in a mock-up polytunnel.

The main contribution of this paper is a novel autonomous navigation system that allows the robot to operate freely in a polytunnel. It is a low-cost and efficient system using only one type of sensor, a planar laser scanner. Even though row following methods have been proposed in earlier work [1, 2, 4, 5, 14, 18, 22, 28], they might not be suitable for challenging constrained environments such as polytunnels.

This paper is organized as follows. In section 2, related works are discussed. Section 3 provides details about the system including line detection and navigation. Simulated and experimental results are presented in section 4. Conclusions are discussed in section 5.

III.2 Related work

Autonomous navigation systems are popular research areas, not limited to any particular fields or type of robots. Most systems, for example [5, 6, 21] depend on several types of sensors such as: inertial measurement units (IMU), high precision RTK GNSS, 3D lidar, etc. Systems with high precision RTK GNSS sensor navigate well only in open environments. Its performance will suffer inside a polytunnel because GNSS signals may be blocked. Inclusion of IMU will help with localization. Admittedly, fusion of multiple sensor types might yield better results in navigation, however it also incurs a higher budget to the end users. Hence, in this work, we aim to develop a low cost and efficient system.

There has been a lot of research on autonomous systems in agricultural applications, such as [3, 19, 25] to name a few. Among those, autonomous row following has attracted interest [1, 2, 4, 5, 14, 18, 22]. In [1, 2], even though the authors also develop autonomous systems for navigating between rows, they rely on cameras to perform a Hough transform for row detection. In [14], a different method based on a particle filter to extract lines from images is proposed to detect row lines. The usage of computer vision for robotic applications has a long history. The main draw back for camera-based navigation systems is that they are totally dependent on lighting conditions. For example, UV light treatment needs to be carried out in a dark environment so that the effect of UV radiation is not nullified by sunlight or any other white light sources. In that situation, camera-based navigation fails. Hence, a laser-based sensor is the most suitable candidate for navigation because it is independent of lighting conditions.

Navigation with 2D planar scanners has been a research topic for the last decades. One of the most extensively used solutions for autonomous navigation for ground mobile robots is *move_base*, a package that is implemented in ROS[3]. In order for the robot to move, one must provide a goal for the robot to reach. Topological navigation [17] is one way to automatically generate goal points for the robot. However, the process to produce a topological map, which contains all the necessary goal points, is tedious and time consuming because one must manually add all the goal points. Given the fact that a typical polytunnel is 60-120 by 9 meters, the total number of goal points can be easily in the hundreds, which makes topological navigation unsuitable for the task.

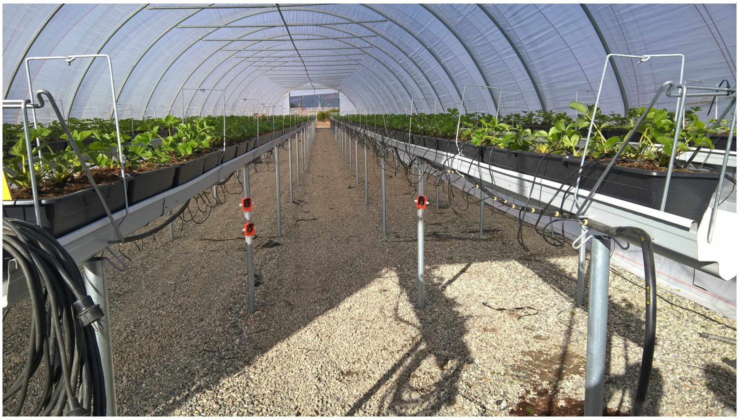


Figure III.2: A polytunnel for growing strawberry in Norway.

Our proposed solution, on the other hand, does not rely on any a priori goal points. By detecting the two parallel lines in front of the laser scan, the robot follows the path between those lines. When it reaches the end of that path, it will continue to detect another set of parallel lines in front of it to follow. In case it can not detect any more lines, the robot will try to determine if it is possible to transit to the next row. First, the robot detects the number of poles currently in the field of view of the scanner. If the number of poles are more than two, the robot will go into transition mode, which makes it enter the next row. If the number of detected poles are less than two, the robot will stop moving because it has already reached the end of the polytunnel. With this solution, the robot can automatically traverse between all the rows inside a polytunnels, for example to deliver a UV light treatment. The desired number of rows to traverse can also be predefined for the robot, so that the robot will cover only a specific area of a polytunnel.

We found that the work in [22], [4] and [20] are similar to ours. [22] also use a 2D laser scanner in combination with a camera for row following in a citrus grove. However, a challenging tightly space constrained condition like a polytunnel does not apply to their environment. [4] also use 2D lase scanner to navigate in rows in tree fruit orchards but required reflective landmarks for row

[3] http://wiki.ros.org/move_base

transition, which we do not. Similar to [22], the robot in [4] does not have to deal with tightly space constrained environment. In [28], the authors employ a spinning 2D laser scanner to detect 3D positions of tree rows and tree trunks in orchards for row following. The spinning 2D laser scanner generates 3D point cloud for registration. In comparison, a tree trunk is much bigger than a steel pole used in a polytunnel. Hence, 3D detection might not detect poles. Further more, like previously mentioned methods, orchards environment is not tightly space constraint as polytunnels. In [20], the authors developed a similar low cost system of row following using only a 2D laser scanner but did not explicitly address the problem of following curved rows.

III.3 Navigation inside a polytunnel

In order to navigate inside a polytunnel, the robot must be able to localize itself inside a given environment. We employed Adaptive Monte-Carlo Localization (AMCL)[4] [24], the de facto SLAM method for 2D laser scanner without further development. The navigation strategy for the robot inside a polytunnel is as follows. The robot is positioned in front of a row. The robot can only see the poles, and not the tables placed on the top of the poles or the plants. By detecting virtual lines between poles, the robot traverses plant rows by following the central path between them. When the robot reaches the end of a row, e.g it can not detect any more lines, the transition row module is activated to get the robot to the next row. The navigation system as in Fig.III.3 comprises of row following and row transition module that can operate seamlessly in and out of the poly-tunnel rows. The robot localizes itself relative to poles using a pre-built map. The laser scanner will monitor consistently for the static or the dynamic obstacle in front of the robot and make an emergency stop if an obstacle is detected within the boundary region. The robot will remain stopped until the detected obstacle is moved by itself or by the nearby worker since there is not an adequate area for avoiding them.

III.3.1 Line detection and following

The laser scanner can detect the poles that are aligned along every polytunnel's rows. This section of poles can be coupled together as virtual lines so the robot can navigate by following the generated trajectory between them as illustrated in Fig. III.4. A RANSAC algorithm [23] is implemented in order to fit a pair of poles as individual line features l_n. Unlike the solid walls, the laser scanner observes the poles in the poly-tunnels as a cluster of points at equal distant from each other. This scenario makes it challenging for the line detection algorithm and consequently, a bounding box is established with a designated boundary region R of length 6 meters and width 2 meters in size (Figure III.5b). The designated search boundary region R is constructed for the sake of eliminating the scan points from another rows as best line fits. From the laser scanner data,

[4] http://wiki.ros.org/amcl

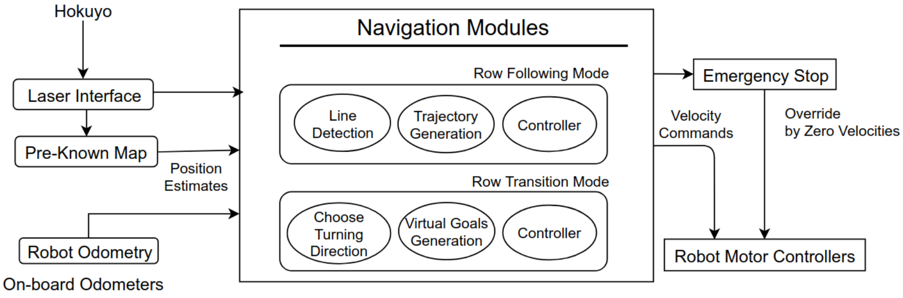


Figure III.3: Modules of navigation

a data set $a_t^n = [a_{x_t}^1, a_{y_t}^1, a_{x_t}^2, a_{y_t}^2, ..., a_{x_t}^n, a_{y_t}^n]$ is generated which contains the x and y axis position of n number of scan points that are extracted from the laser scan range and bearing values within the boundary region R.

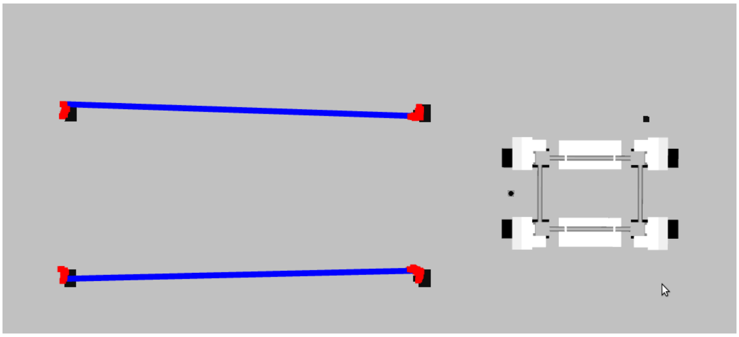


Figure III.4: The proposed line detection algorithm identifies two lines (blue) from the laser scanner points using a Hokuyo laser which is mounted on the robot. The standard RANSAC algorithm is used for line extractions. The robot detects only 2 poles on each side.

In order to execute the line detection algorithm, let us suppose that the line model L_t can be expressed as a function $f(S_t)$ which depends on randomly generated subset of points in S_t taken from the set a_t^n as:

$$L_t = f(S_t) \tag{III.1}$$

where $S_t = [p_{x_t}^1, p_{y_t}^1, p_{x_t}^2, p_{y_t}^2] \subseteq a_t^n$ comprises the position of the two randomly generated points p^1 and p^2 from the set a_t^n at time t respectively. The function $f(S_t)$ computes the model line parameters such as slope m_t and y-intercept b_t

based on the two randomly selected points $p^1_{(x_t,y_t)}$ and $p^2_{(x_t,y_t)}$ is given by:

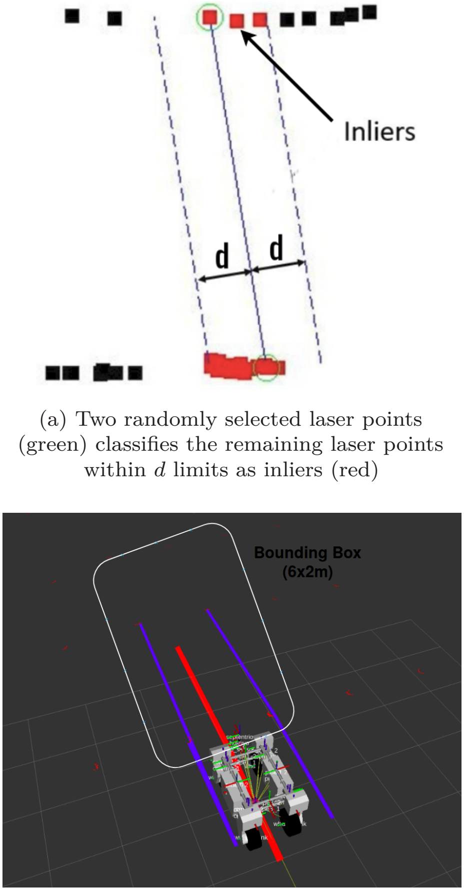
$$\begin{aligned} m_t &= \frac{p^2_{y_t} - p^1_{y_t}}{p^2_{x_t} - p^1_{x_t}}, \\ b_t &= p^1_{y_t} - m_t p^1_{x_t}. \end{aligned} \tag{III.2}$$

When the line model parameters are computed, let $E(L_t, a^n_t)$ be the objective function constructed using the least squares method for line fitting [27]. The objective function $E(L_t, a^n_t)$ proclaims the sum of all the residual values for each point belonging to the set a^n_t with respect to the estimated line model L_t. Therefore minimizing the objective function $E(L_t, a^n_t)$ will eventually minimize the residual values so that the estimated line will be close enough to most of the points from set a^n_t. The optimal set of line model parameters m_t and b_t are need to be found as per least squares method for minimizing the residuals. Hence the objective function minimizing the sum of the the squared normal distances from each point takes on the form:

$$E(L_t, a^n_t) = \sum_{p=1}^{n} \|(a^p_{y_t} - m_t a^p_{x_t} - b_t)\|^2. \tag{III.3}$$

The standard RANSAC algorithm has few parameters defined beforehand as pre-conditions that are suitable for the polytunnel environment. For minimizing the objective function $E(L_t, a^n_t)$, a threshold parameter d is introduced which represents the threshold distance from the two chosen random points for fitting the remaining scan points as inliers (see Fig. III.5a). The parameter k describes the total iterations required to determine the best line fit and therefore it will keep updating the best line fit if the better line feature with more inliers are found for the entire k^{th} number of iterations. The parameter $inliers_{min}$ represents the minimum number of inliers to be necessitated for finalizing the estimated line as a best line model. The RANSAC estimates the line after the pre-defined conditions are satisfied. Thus the parameters for the pre-conditions are tuned in such a way that it fits the parametric line model is given as:

$$\begin{aligned} &E(L_t, a^n_t) \leq d \\ &\text{where} \quad d = 0.05(m), \quad k = 100, \quad \text{inliers}_{min} = 10 \end{aligned} \tag{III.4}$$

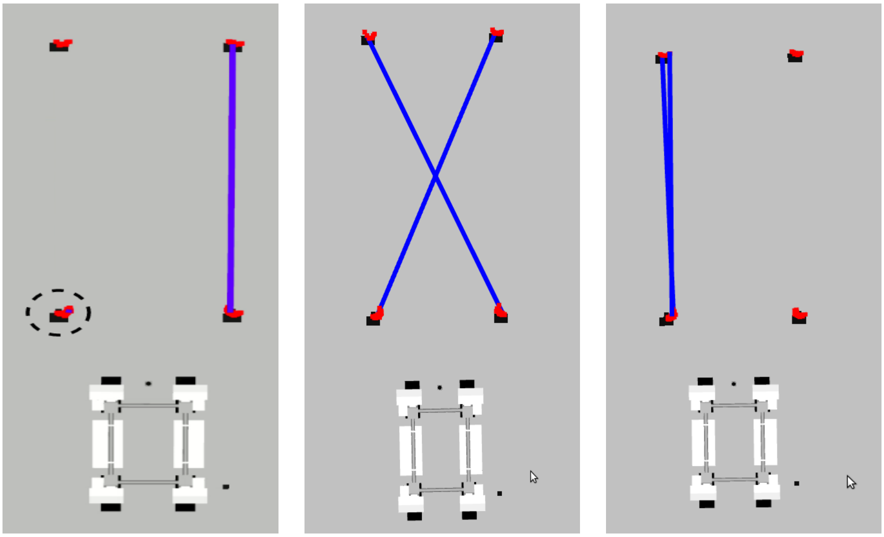


(a) Two randomly selected laser points (green) classifies the remaining laser points within d limits as inliers (red)

(b) Detected pair of lines (blue) and desired path (red) within the bounding box region (white)

Figure III.5: Row following module: line detection methodology

The RANSAC will run twice such that after satisfying the parameters in the pre-conditions, two best line fits will be estimated. Thus the points from the set S_t are incorporated as line features l_1 and l_2. For the sake of simplicity, we don't include time t in the line feature equations. As soon as the RANSAC algorithm identifies the lines $l_1 = [l_1^{x_1}, l_1^{y_1}, l_1^{x_2}, l_1^{y_2}]$ and $l_2 = [l_2^{x_1}, l_2^{y_1}, l_2^{x_2}, l_2^{y_2}]$ by satisfying the pre-defined conditions, some additional constraints are considered to avoid multiple detections, overlaps and other false positives (see Fig. III.6). The false detections are ignored. If no pair of lines is detected, the algorithm uses the previous detections until the new set of lines appears. These constraints aid in fitting the best line features for the entire navigation system. The first constraint is the minimum distance between the end points $(p^1_{(x_t,y_t)}, p^2_{(x_t,y_t)})$ for each of the two detected lines (l_1^t, l_2^t) that has to be always more than the threshold value



(a) False line detections covering not more than one pole (black circled)

(b) Cross line detections

(c) Overlaps in line detections

Figure III.6: Line detection for polytunnels: false detections.

τ as in Eq. III.5. This particular constraint avoids the possibility of detecting the incorrect best line fit when more inliers are stacked up together at one place (the black circled area in Fig.III.6a) and it is expressed as:

$$\begin{aligned} \sqrt{(l_1^{x_1} - l_1^{x_2})^2 + (l_1^{y_1} - l_1^{y_2})^2} > \tau \\ \sqrt{(l_2^{x_1} - l_2^{x_2})^2 + (l_2^{y_1} - l_2^{y_2})^2} > \tau. \end{aligned} \tag{III.5}$$

While traversing through the inclined shaped rows, the robot can also find the line features diagonally between two parallel rows as the best line fit at the same time (Fig.III.6b). For avoiding this situation, the constraint based on the angle between two end-points of the detected line is introduced. This angle is presumed to be less than Φ which is assigned as 15 degrees at maximum so that it can still detect the curved shaped poles but it can also avoid finding cross line detections at the same time. The second constraint can be written as:

$$\begin{aligned} \arctan \frac{l_1^{y_2} - l_1^{y_1}}{l_1^{x_2} - l_1^{x_1}} < \Phi \\ \arctan \frac{l_2^{y_2} - l_2^{y_1}}{l_2^{x_2} - l_2^{x_1}} < \Phi. \end{aligned} \tag{III.6}$$

There is another possibility of false detection in which the RANSAC could detect the already chosen best line fit as second best line fit again (see Fig. III.6c) because the line detection will keep finding the two best line fit l_1 and l_2 at time t and this case will also satisfy the first and second constraints as well. Therefore the third constraint is proposed as:

$$l_1^{x_1} \neq l_2^{x_1} \quad l_1^{y_1} \neq l_2^{y_1} \quad l_1^{x_2} \neq l_2^{x_2} \quad l_1^{y_2} \neq l_2^{y_2}. \tag{III.7}$$

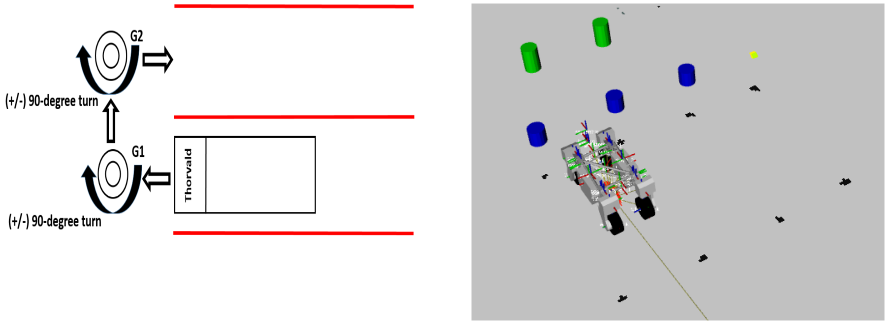
This added constraint as given in Eq. III.7 will avoid the situation where both the detected lines do not overlap each other. If the overlapping is detected using this constraint, then this pair of lines from the concerned iteration in RANSAC are rejected. After fulfilling all the three proposed constraints, the two lines l_1^t and l_2^t will be extracted on both sides of the robot in order to navigate between them. The desired trajectory has been derived as an average of the two estimated lines as in the Figure III.5b. Once the desired trajectory has been estimated, a low level controller is used for sending the necessary velocities as joint commands. The linear velocity V_t is constant and it moves at 0.3m/s for safety reasons. In order to steer the robot, a low level pure pursuit controller [7] is used to calculate the respective steering velocity ω_t for following the center line based on two estimated lines l_1^t and l_2^t by the line detection algorithm as in the Figure III.5. The steering velocity ω_t equation is written as

$$\omega_t = \arctan\big(2R_l \frac{\sin e_{\theta_t}}{e_{(x_t,y_t)}}\big) \tag{III.8}$$

where R_l, e_{θ_t}, $e_{(x_t,y_t)}$ are the total length of the robot, errors along its rotation angle and its position with respect to the current goal at the time t respectively.

III.3.2 Row transition

Once the row following module could not detect any more new lines, the robot will navigate till the end of the current desired trajectory using the last pair of detected lines. When it reaches the end of the current desired trajectory, it shifts autonomously to the row transition module. The operation of the row transition comes to an end when the robot progressed to the beginning of the next row and switches back to the row following module. In this module, the pole detection algorithm identifies the closest 3 poles which comprises of two poles on one side and another one pole on the other side of the robot based on its next course of direction. For instance, if the robot needs to transit to the new row on the right-hand side then the pole detection algorithm will give the pair of closest poles on right-hand side (Fig. III.7b) and one pole on the left-hand side or vice-versa for the turning to the next row on the left-hand side condition. Thus the virtual goal points are generated by taking the average between the three detected poles and adding a constant offset to it as seen in Fig. III.7. Then the pure pursuit controller is designed in such a way that it will navigate the robot to the first virtual goal point from the current row and makes a 90 degree-turn around that first goal point. Furthermore, it repeats the same process for the second virtual goal point (Fig. III.7a) in order to shift into the new row. Therefore the course of the turning direction should be given



(a) Robot assigned to reach goal points (G1,G2) and make (+/-) 90 degree turn around the goal points for transiting into new row.

(b) Poles at end of the row (blue) are detected and generate goal points (green) by implying an offset to it.

Figure III.7: Row transition module: row transition methodology

```
input   : 2D laser scan measurements, number of rows to traverse
while the end of the polytunnel is not reached do
    for each laser scan measurement do
        Perform RANSAC fitting for 2 lines $l_1$ and $l_2$;
        if no lines are detected then
            Check whether the robot has reached the end of the last row;
            if true then
                Stop;
            else
                Initialize row transition;
                Transits to next row;
            end
        else
            Compute the middle trajectory between $l_1$ and $l_2$;
            Follow the middle trajectory;
        end
    end
end
```

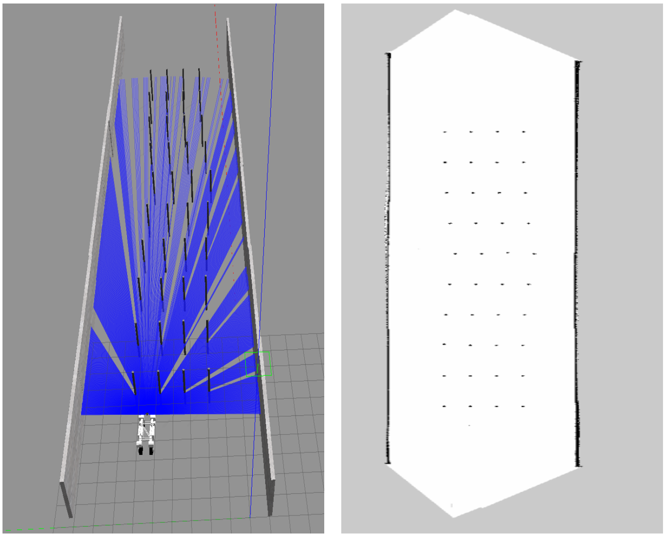
Algorithm 1: Navigation algorithm

beforehand such that the robot can navigate any polytunnels which has a larger number of rows. Moreover, the row transition module brings the integration with the row following module and makes a complete autonomous navigation system exclusively for polytunnels like environments. The pseudo-code (algorithm 1) exhibits the integration of both the navigation modules for both the straight and curved shaped poly tunnels.

III.4 Experimental Results

III.4.1 Simulations

The proposed method is verified in simulation and field trials. We show that our system can move along rows efficiently. We also discuss how our system can be extended to different environments, such as polytunnels that hang plant trays instead of using poles. The simulated environment (Fig.III.8) is created using Gazebo to mimic the real poly-tunnel environment. It consists of a plane ground and several sets of cylinders with plant trays on top. The spacing between each set of cylinders can be modified to match reality. In this environment, the robot is tasked to traverse all the rows, while in reality, it might not have to do so due to the requirement of UV light treatment, e.g. not every row requires UV light treatment. In the simulated environment, the robot is fixed at an initial known position in front of one of the rows. The row-following module in the navigation system begins traversing through the initial row that it perceives first



(a) Curved Poles Environment in Gazebo

(b) 2D Map of polytunnel Environment

Figure III.8: Experiments in simulated environments

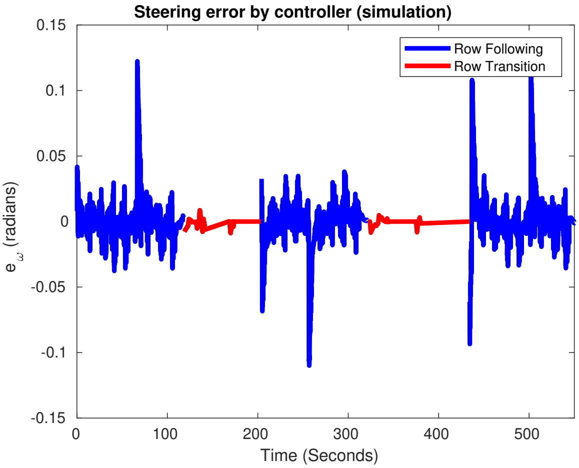


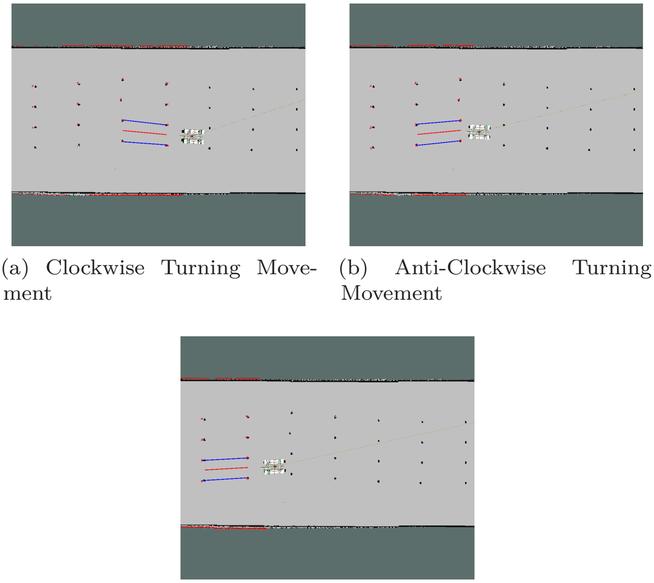
Figure III.9: Steering controller error in simulated polytunnel. Steering errors when the robot follows rows are shown in blue. Steering errors when the robot changes rows are shown in red. Best view in color.

in the environment. The robot can find the curved shaped rows and extract the trajectory lines by the line detection technique at every time step t. Then the robot can steer in both clockwise and counter-clockwise directions and can revert back to straight row following with the lesser amount of steering as shown in Fig. III.10.

The pure pursuit controller in both the row-following module and the row-transition module assist the robot to steer between and outside the rows of the poly-tunnels. The error in the graph indicates the angular difference between the current robot position and current dynamic goal position that the controller should correct at each time step t. The controller maintains the required steering angle error to be less than 0.15 radians in row-following and 0.01 radians in row-transition modules. They can maintain the error close to zero as shown in Fig. III.9 throughout the entire trajectory. The steering error increases whenever the robot needs to traverse through curved areas in the rows but it reduces again over time. In the row transition module, the controller makes the robot move along the two virtual goal points with the given steering commands and transit to the beginning of the next row

III.4.2 Navigating in a mock-up polytunnel

The robot used in the field test is shown in Fig. III.13. We constructed a mock-up polytunnel, which is 24 meters long by 9 meters wide. The mock-up polytunnel has 32 poles, which create three long rows. We deliberately added constant displacements to 12 middle poles (inside the square) as shown in Fig.III.12a. In Fig.III.12, we show how the robot detects parallel lines and moves in the center



(a) Clockwise Turning Movement

(b) Anti-Clockwise Turning Movement

(c) Return to Straight Line Movement

Figure III.10: Snapshots of the robot movement in simulation.

of a row. The curved lines are detected and shown on Fig.III.12d, III.12e, III.12f, III.12g. Transition points (two green circles) are shown in Fig. III.12i. They are automatically computed when the robot reaches the end of the row and detects three poles (blue circles) in front of it. The whole trajectory is shown in Fig. III.12j.

The robot is tasked to traverse through every row. We evaluate the robot navigation quality by two metrics, including displacement to centering lines and distances to poles on both sides of the robot. We explain these metrics in details next.

a) Translation error: all the poles' positions are carefully measured using a Leica Total Station TCA1100 with distance measurement accuracy ±1mm. These measurements are for computing virtual centering line segments on each row, which are considered ground truth. We evaluate the quality of navigation by calculating the deviation of the actual trajectory from the ground truth. For each segment of a row, we first compute the Euclidean distance of each robot position measured by the Leica to the ground truth. The average of these distances is considered as the robot translation error on that segment. For the whole row, we again average all the translation errors of all segments. The reason we choose this metric is that it provides insight into how the robot performs on each segment of a row.

b) Distance to poles error: we measure the distances from the center of the robot body to the poles on both of its sides when passing them. The distance from the robot body center to each pole on the left and right side of the robot are taken by a laser measurement Uni-T UT390B^{+} with $\pm 2mm$ accuracy. These measurements are used to evaluate how well the robot stays in-between poles.

We collect our metric measurements by letting the robot run autonomously through the mock-up tunnel ten times. The final result is averaged over these ten trials.

The result of ground truth comparison is shown in Fig. III.14. The ground truth trajectory (blue line) consists of line segments representing the ideal trajectory that stays exactly in the middle of rows.

The translation errors are shown in Table III.1. We ignore the robot trajectory that is outside rows. Given the average path of each row is 21 meters, the maximum mean error is only 29.3 cm, which yields a relative small 1.4% drift over a travelled distance.

	Translation errors (m)					
	Min		Average		Max	
	Ours	[15]	Ours	[15]	Ours	[15]
Row 1	0.052	0.024	0.117	0.128	**0.165**	0.169
Row 2	0.101	0.101	0.213	0.187	0.293	**0.292**
Row 3	0.063	0.131	0.154	0.237	**0.203**	0.386

Table III.1: Translation errors per row. Bold numbers indicate best results.

	Distance to poles (m)					
	To left poles d_ℓ		To right poles d_r		$\mid d_\ell - d_r \mid$	
	Ours	[15]	Ours	[15]	Ours	[15]
Row 1	0.257	0.272	0.226	0.215	**0.031**	0.057
Row 2	0.282	0.214	0.193	0.257	0.089	**0.043**
Row 3	0.235	0.321	0.251	0.147	**0.016**	0.174

Table III.2: Mean errors of distance from the center of the robot body to poles on both sides. Bold numbers indicate best results.

In Table III.2, the results of staying in-between poles are presented. The distance between 2 poles on each side of a robot is 1.5 meter. The robot width is 1 meter. It means the robot needs to stay at least 25 cm away from poles on each side. The maximum mean distance to poles on the right side is approximately 25.1 cm, and to the left side is 28.2 cm. This shows that the closest distances the robot gets to a left and a right pole are approximately 24.9 cm and 22.8 cm, respectively, which are well within the ideal safety distance. Also, the maximum difference between the mean distances on both sides of the robot is 8.9 cm. It shows that the robot well maintains its position at the center along rows by keeping the same distances to poles on both sides of a row.

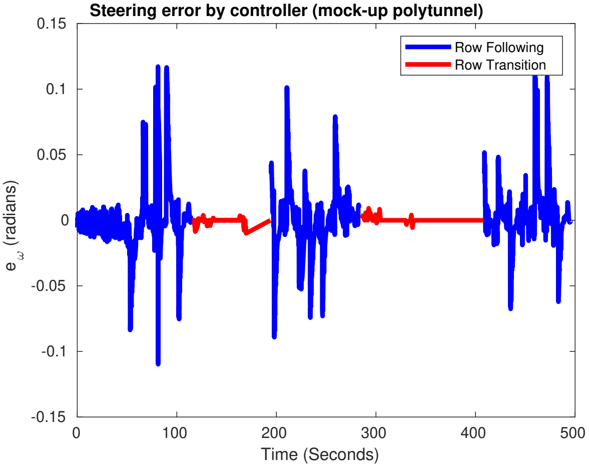


Figure III.11: Steering controller error in mock-up polytunnel. Steering errors when the robot follows rows are shown in blue. Steering errors when the robot changes rows are shown in red. Best view in color.

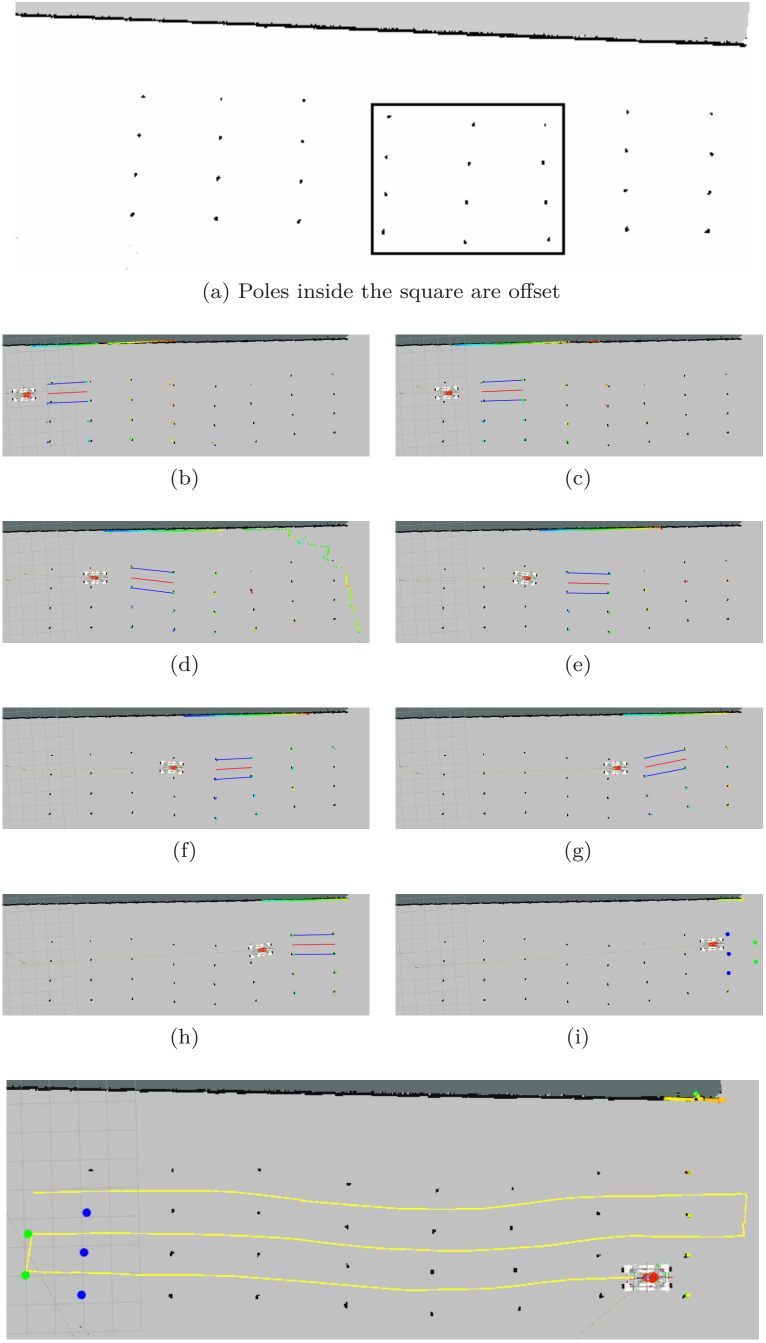
The pure pursuit controller in field tests behaves in a similar way to the simulation. Unlike the simulated environment, the poles are not aligned perfectly straight with respect to each other in real fields. The lines that are detected from line detection algorithm are not exactly straight as well, hence the steering error in the real-field tests is higher than the one in the simulation. As in Fig. III.11, the curved areas are evident in which the steering error peaks in each row along the mock-up polytunnel. In the row transition module, the steering error is kept to a low value even in the uneven terrain that are similar to simulations.

We also compare our proposed method with [15]. The topological navigation approach proposed in [15] is currently being used in our RASBerry project[5]. In order to use topological navigation, one needs to manually create *topological nodes* that connect each other to form a topological map as shown in Fig. III.15. Given a topological map, the robot can move from one arbitrary node to another. This method relies on AMCL for localization, which is similar to ours.

We run ten trial tests and collect the same metric measurements for comparison. The final result of topological navigation is averaged over ten runs and shown in Table III.1, III.2 altogether with our proposed system for comparison. Bold numbers indicate better results. Our method on average achieves better result in both metrics.

We also note that the topological navigation requires creating topological nodes, which must be done manually and therefore unsuitable for a large polytunnel. This is one of the motivations of our proposed method. One

[5] https://rasberryproject.com/



(a) Poles inside the square are offset

(b)

(c)

(d)

(e)

(f)

(g)

(h)

(i)

(j)

Figure III.12: Results of autonomous navigation between rows in a mock-up polytunnel. Blues lines are virtual lines between detected poles. Red lines are the central paths between rows. Yellow line is the complete actual trajectory.

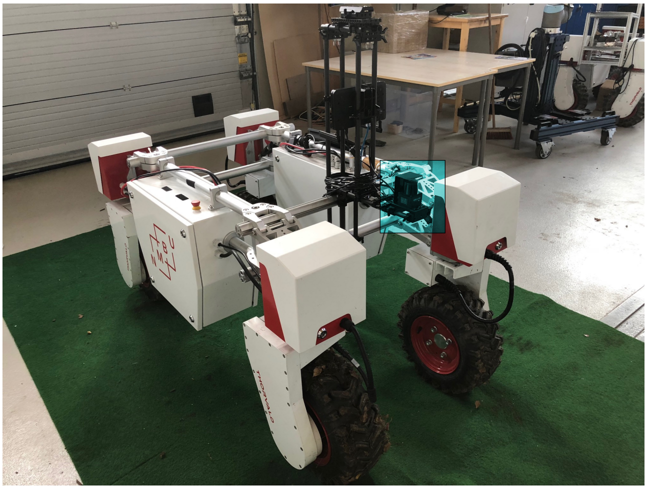


Figure III.13: Robot setup, a 2D Hokuyo laser scanner UST-20 LX is mounted in front of the robot as shown in the highlight area.

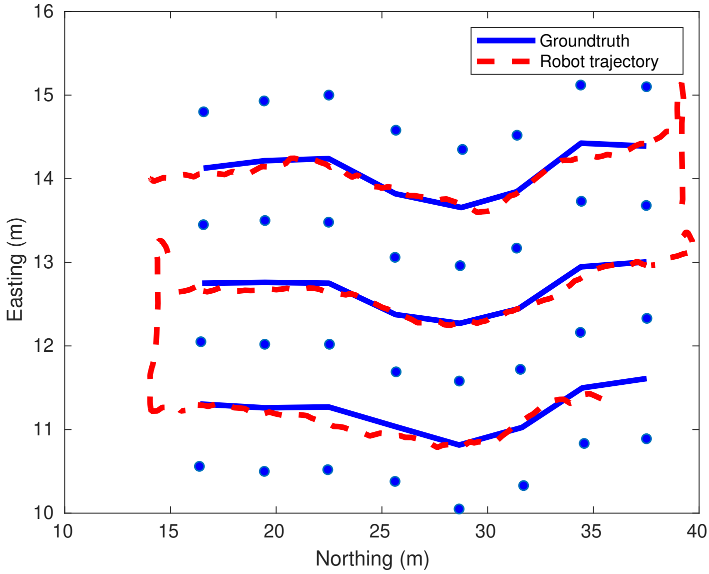


Figure III.14: Qualitative analysis of trajectories.

might argue that a *sparse* topological map would be easier to make. However, we found that in practice, a tightly constrained space requires a *dense* topological map for the robot to travel safely. In addition, by relying on a *cost map* for planning, the robot is prone to make dangerous path planning such as in Fig. III.16, whereas our method does not. Furthermore, the use of *cost map* makes the system more sensitive to faulty reading from sensor. As shown in Fig. III.16, an artificial obstacle due to noisy laser scanner was added to the *cost map* and forced the robot to move out of the row. This is a dangerous behavior. The robot is likely to collide with other poles because there is not enough space for rotation. Hence, we claim that our proposed system is better suited for polytunnel environment.

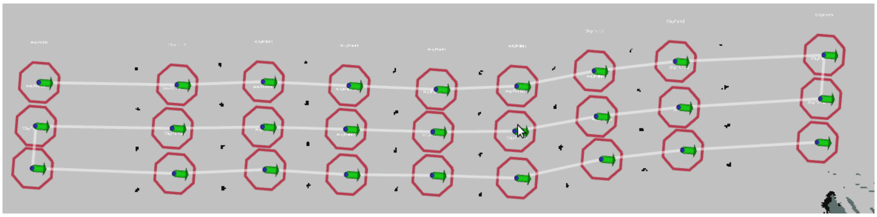


Figure III.15: Topological map for navigation. This method proposed by [15] relies on *move_base* for motion planning. All the topological nodes were manually created.

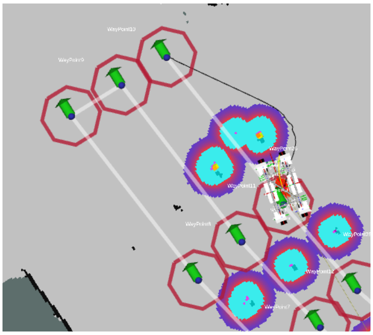


Figure III.16: Topological navigation attempts a dangerous movement. The black line shows the planned trajectory. Notice the artificial obstacle between the poles was incorrectly perceived due to noisy laser scanner readings. It forces the robot to move around.

A video of the robot moving in the mock-up polytunnel is available online : https://youtu.be/xkSpEkcBXaU.

III.4.3 Discussion

One key aspect of our system is low-cost. However, we have shown that autonomous navigation in tightly constrained agricultural domains such as polytunnels can be carried out efficiently. Our system works well in the challenging environment of polytunnels, where features for laser scanner detection are sparse. Our system can be easily adapted to different types of polytunnels without much effort. For example, plant trays might be hanged using cables instead of sitting on poles. In this case, if those cables are smalls and can not be reliably detected by the laser scanner, we can adjust to mount the laser scanner to directly detect the trays. Our system can continue to work normally without any further changes. The line detection will be easier since laser scanner detects more points from trays.

Another practical consideration is how to determine the window size for pole detection. In our implementation, the window size is fixed and its value is set upon the applied standard polytunnel structure, i.e the distance between two consecutive poles in a row is approximately 3 meters the row width is 1.5 meters. These values can be preset once with respect to the actual environment before letting the robot move. It might sound preferable to have an automatically adaptive window size, but in fact, we rarely see a polytunnel with different spacing between poles. For most cases, polytunnels are built in compliance with a standard, for which we argue that a corresponding fixed size window is adequate.

It is obvious that our system makes strong assumptions about the environment such as the distance between poles is constant, number of poles on each side of a row is equal. Our proposed system may fail to operate if those assumptions are not satisfied. However, we argue that those assumptions are reasonable, i.e it is uncommon to find a polytunnel with asymmetrical structure. Therefore our proposed system is useful for most cases.

III.5 Conclusions

In this paper, a simple but efficient navigation solution of row following for an agricultural robot is presented. Our main goal is to develop an efficient but also cost effective system that can work reliably in polytunnels, which are the typical space constrained agricultural environment. We deliberately employ one 2D laser scanner. Using only this type of sensor, the robot is able to move between rows while keeping equidistant to both sides of a row. We claim this is important for several tasks, in which the robot must stay in the middle of a row, such as delivering UV light treatment, or autonomous transporting harvested products in and out of a polytunnel.

Experimental testing in both simulation and a mock-up polytunnel were performed to evaluate the quality of navigation. The results show a small drift of 1.4% over total travelled distance per row and the robot maintains the same distance to poles on both sides.We compare our proposed row following method with an existing one in [15]. We show that our method achieves better results.

We have discussed how our system can be easily adapted to different types of polytunnels. In rare cases, where the structure of a polytunnel is irregular, i.e distances between poles are different, number of poles on each side of a row are not equal, our system will fail. However, it is unusual to have a structure like that. Our proposed solution replaces the traditional and other way point based methods such as topological navigation [16] and thus simplifies the robot operation process. For future work, we aim to develop a full scale SLAM based navigation system. More navigation and safety sensors will also be employed on the robot for the human aware navigation in the future that can cooperate along human labourers and also in respect to the safety standards in the polytunnels.

III.6 Acknowledgment

We would like to thank Øystein Sund from SAGA Robotics for designing UV light holder systems.

References

[1] Åstrand, Björn and Baerveldt, Albert-Jan. “A vision based row-following system for agricultural field machinery”. In: *Mechatronics* vol. 15, no. 2 (2005), pp. 251–269.

[2] Bakker, Tijmen et al. “A vision based row detection system for sugar beet”. In: *Computers and electronics in agriculture* vol. 60, no. 1 (2008), pp. 87–95.

[3] Bergerman, Marcel, Singh, Sanjiv, and Hamner, Bradley. “Results with autonomous vehicles operating in specialty crops”. In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on.* IEEE. 2012, pp. 1829–1835.

[4] Bergerman, Marcel et al. “Robot farmers: Autonomous orchard vehicles help tree fruit production”. In: *IEEE Robotics & Automation Magazine* vol. 22, no. 1 (2015), pp. 54–63.

[5] Biber, Peter et al. “Navigation system of the autonomous agricultural robot Bonirob”. In: *Workshop on Agricultural Robotics: Enabling Safe, Efficient, and Affordable Robots for Food Production (Collocated with IROS 2012), Vilamoura, Portugal.* 2012.

[6] Cariou, Christophe et al. “Automatic guidance of a four-wheel-steering mobile robot for accurate field operations”. In: *Journal of Field Robotics* vol. 26, no. 6-7 (2009), pp. 504–518.

[7] Conlter, R Craig. “Implementation of the Pure Pursuit Path’hcking Algorithm”. In: (1992).

[8] Fischler, Martin A and Bolles, Robert C. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. In: *Readings in computer vision*. Elsevier, 1987, pp. 726–740.

[9] Grimstad, Lars and From, Pål Johan. “The Thorvald II agricultural robotic system”. In: *Robotics* vol. 6, no. 4 (2017), p. 24.

[10] Grimstad, Lars and From, Pål Johan. “Thorvald II - a Modular and Re-configurable Agricultural Robot”. In: *IFAC 2017 World Congress*. 2017.

[11] Grimstad, Lars et al. “A Novel Autonomous Robot for Greenhouse Applications”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018.

[12] Grimstad, Lars et al. “On the design of a low-cost, light-weight, and highly versatile agricultural robot”. In: *2015 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO 2015)*. 2015.

[13] Grimstad, Lars et al. “Thorvald II Configuration for Wheat Phenotyping”. In: *IROS Workshop on Agri-Food Robotics: learning from Industry 4.0 and moving into the future*. 2017.

[14] Hiremath, S, Evert, F, Heijden, G, et al. “Image-based particle filter for robot navigation in a maize field”. In: *Workshop on Agricultural Robotics: Enabling Safe, Efficient, and Affordable Robots for Food Production (Collocated with IROS 2012), Vilamoura, Portugal*. 2012.

[15] Krajnık, Tomáš et al. “Long-term topological localisation for service robots in dynamic environments using spectral maps”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2014, pp. 4537–4542.

[16] Krajnık, Tomáš et al. “Long-Term Topological Localization for Service Robots in Dynamic Environments using Spectral Maps”. In: *IROS ’14*. to appear. 2014.

[17] Lázaro, M. T. et al. “A lightweight navigation system for mobile robots”. In: *ROBOT 2017: Third Iberian Robotics Conference*. Sevilla, Spain, Nov. 2017.

[18] Moorehead, Stewart J et al. “Automating orchards: A system of autonomous tractors for orchard maintenance”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Agricultural Robotics, Vilamoura, Portugal*. 2012.

[19] Reid, John F. “The impact of mechanization on agriculture”. In: *Bridge* vol. 41, no. 3 (2011), pp. 22–29.

[20] Riggio, Giuseppe, Fantuzzi, Cesare, and Secchi, Cristian. “A Low-Cost Navigation Strategy for Yield Estimation in Vineyards”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2200–2205.

[21] Stoll, Albert and Kutzbach, Heinz Dieter. "Guidance of a forage harvester with GPS". In: *Precision Agriculture* vol. 2, no. 3 (2000), pp. 281–291.

[22] Subramanian, Vijay, Burks, Thomas F, and Arroyo, AA. "Development of machine vision and laser radar based autonomous vehicle guidance systems for citrus grove navigation". In: *Computers and electronics in agriculture* vol. 53, no. 2 (2006), pp. 130–143.

[23] Tarsha-Kurdi, Fayez, Landes, Tania, and Grussenmeyer, Pierre. "Hough-transform and extended ransac algorithms for automatic detection of 3d building roof planes from lidar data". In: *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007.* Vol. 36. 2007, pp. 407–412.

[24] Thrun, Sebastian. "Probabilistic robotics". In: *Communications of the ACM* vol. 45, no. 3 (2002), pp. 52–57.

[25] Ting, KC et al. "Information Technology and Agriculture Global Challenges and Opportunities". In: *Bridge* vol. 41, no. 3 (2011), pp. 6–13.

[26] Xiong, Ya, From, Pal Johan, and Isler, Volkan. "Design and Evaluation of a Novel Cable-Driven Gripper with Perception Capabilities for Strawberry Picking Robots". In: *arXiv preprint arXiv:1804.09771* (2018).

[27] York, Derek. "Least-squares fitting of a straight line". In: *Canadian Journal of Physics* vol. 44, no. 5 (1966), pp. 1079–1086.

[28] Zhang, Ji et al. "3d perception for accurate row following: Methodology and results". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE. 2013, pp. 5306–5313.

Paper IV

A Supervised Learning Solution for Autonomous Row Following Tasks in Horticulture

Tuan Le, Vignesh Raja Ponnambalam, Jon Glenn Omholt Gjevestad, Pål Johan From

Abstract

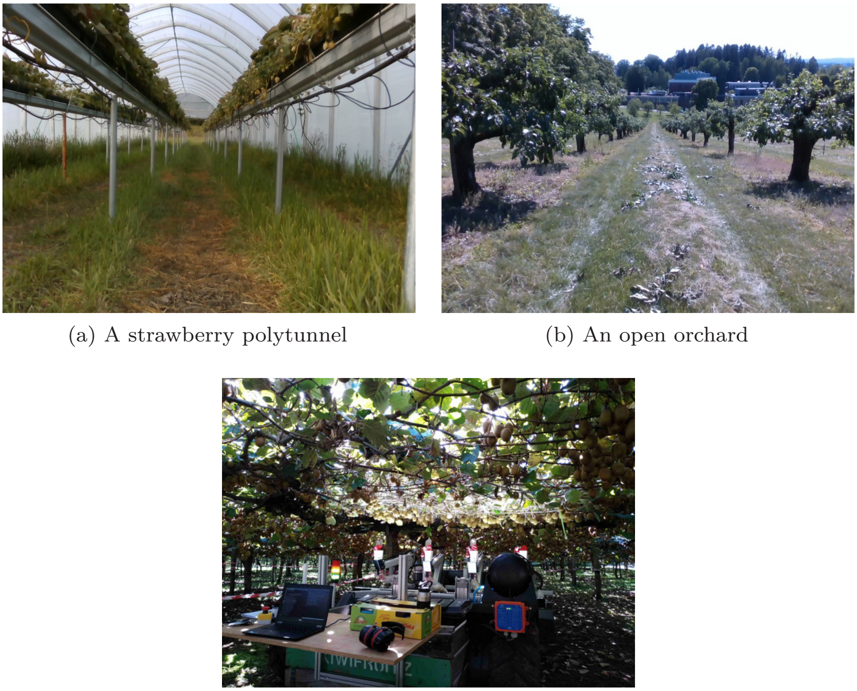
Precision agriculture is the key to sustainable farming. The usage of autonomous robotics systems in agriculture is rising. Similar to other mature areas of applied robots, agricultural robots must be able to robustly navigate in their working places (polytunnel, crop fields, etc.,). In horticulture, row following is one of the key tasks that autonomous agricultural robots must perform. Several studies had been done to address this problem. However, existing methods are tailored to their specific environments. This work aims to provide a CNN approach to row following tasks that can be used for both indoor (polytunnel-liked) and outdoor (orchard-liked) environments.

IV.1 Introduction and Motivation

A common practice for growing vegetation in horticulture is to form row-like structures. For outdoor environment, orchards mostly use row-liked structures for growing. For fruits such as apples and oranges, the most common row structure is a tree wall, e.g a row is formed by placing trees on both sides of a path. However, for fruits such as grapes, pears and kiwi, a pergola structure is more common. In a pergola, rows are formed by trees and supporting poles. For indoor environment such as polytunnel, rows are formed either by lines of table-trays placing on poles or being hung from the roof. We show three examples of polytunnel, open orchard and pergola in Fig. IV.1, respectively.

For open fields like orchards, classical navigation methods relying on external position sensors such as GNSS were fully developed [3]. For greenhouses or polytunnels, existing navigation methods from the robotics community using a 2D laser scanner can be directly applied [5]. Obviously, these classical methods

All authors are with Faculty of Science and Technology, Norwegian University of Life Sciences



(a) A strawberry polytunnel

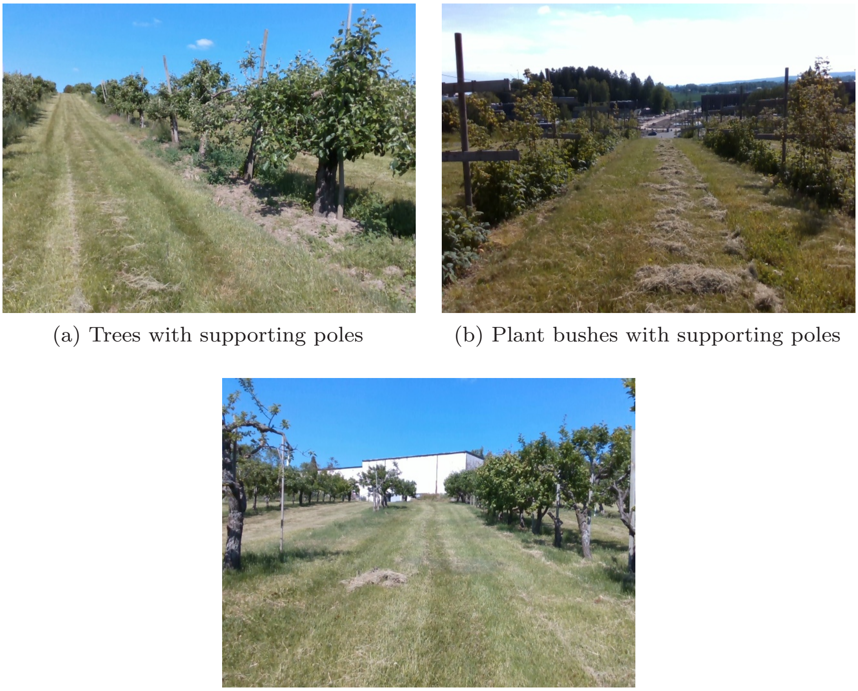
(b) An open orchard

(c) A kiwifruit orchard with pergola structure. Image courtesy of [11]

Figure IV.1: Different types of horticultural environments.

may suffer in some specific conditions: blockage of GNSS signals (in pergolas where dense canopies usually exist), uneven ground floor distorts 2D laser scanner, or in case of missing trees in a row (Fig. IV.1c) might also confuse the laser scanner reading. Several works have been done to address these problems, which specifically avoid using any external position sensor or assuming a flat terrain. Zhang *et al.* in [12] used a rotating 2D laser scanner for augmenting 3D scans to detect tree trunk and traverse along tree rows in an orchard. Bell *et al.* in [1] propose a navigation approach using a 3D LiDAR sensor to navigate inside a kiwifruit pergola, where GNSS signals are blocked by dense canopies.

We are motivated by the structural variations that we have in our test fields at NMBU. We have a strawberry polytunnel, in which three rows of tabletop trays are placed on poles. The row width is 1.5 meters (Fig. IV.1a). On the other hand, we have an open orchard where three different types of structure are utilized: a) standard rows, where trees are roughly spaced 2 meters apart, as shown in Fig. IV.1b, b) trees with supporting poles, which are roughly 2 meters apart, as shown in Fig. IV.2a, c) small trees with large supporting poles, where poles are roughly 2.3 meters apart, as shown in Fig. IV.2b. On some rows, one tree or several trees might be missed as shown in Fig IV.2c. The row width in our



(a) Trees with supporting poles

(b) Plant bushes with supporting poles

(c) A row with missing trees

Figure IV.2: Different types of orchard environments at NMBU.

orchard is much wider than the one in our polytunnel. Moreover, different types of row following tasks may be performed on these environments. For example, UV light treatment in polytunnel or tree watering on orchards are classified as *centerline following* tasks, meaning a robot needs to maintain equidistant to both sides. An example of centerline following in UV light treatment in a polytunnel is shown in Fig. IV.3.

For orchard with a wide row in harvesting season, a robot may need to stay close to one side of a row while moving along that row for fruit harvesting. This is classified as *sideline following* task.

We are inspired by the work of Bell *et al.* in [2], in which the authors trained a fully convolutional neural network (FCN) for segmenting drivable areas for row following in a kiwifruit pergola. Drivable area means the area a robot can translate to from its current position without collision. We believe this approach is more generic and applicable than existing methods relying on external position sensor (high cost for RTK-GNSS devices), artificial landmarks (burden on infrastructure for placing and maintaining) or laser scanner sensor (being confused in the presence of missing/additional objects). More over, it uses a low-cost camera sensor, which keeps the whole robotic system cost-efficient.

We argue that our work is different from the one in [2] by a magnitude of

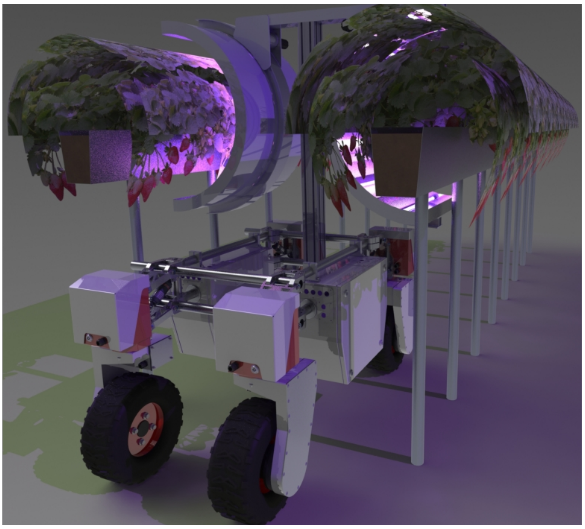


Figure IV.3: A design of Thorvald robot for UV line treatment inside a polytunnel. The robot is required to perform centerline following.

generalization. The authors in [2] were only concerned about centerline following for harvesting tasks in a specific kiwifruit pergola. We train our network for segmenting traversable ground on an inclusive dataset containing both indoor (a strawberry polytunnel) and outdoor (orchards with three different types of row structure) environment. We also cross-validate our network performance on different network architectures, including ResNet [6], Darknet [8], MobileNet [10] and ERFNet [9]. Hence, we can evaluate how our network performs in different types of environments with different network architectures. In addition, for outdoor environment, we also have three different types of structures. Hence, our network is suitable for many types of environments, which makes it more generic.

IV.2 Description of Dataset and Training Process

For data collection, we use the popular Intel Realsense Camera D435i. We mount the Realsense camera on our ground robot [4] as shown in Fig. IV.4. We manually joystick the robot along rows in our strawberry polytunnel and our open orchard. We made sure to capture as many different scenarios as possible: a) our strawberry polytunnel recordings contain our robot moving along rows with in-row rotations that are not considered dangerous b) for our open orchard, our robot undergoes different moving directions while traversing rows - straight line, rotating, diagonally c) data is being recorded under various light conditions. We select 500 images of size 640x480 pixels for training and 57 images of the same size for testing. For labeling images, we manually label each pixel either

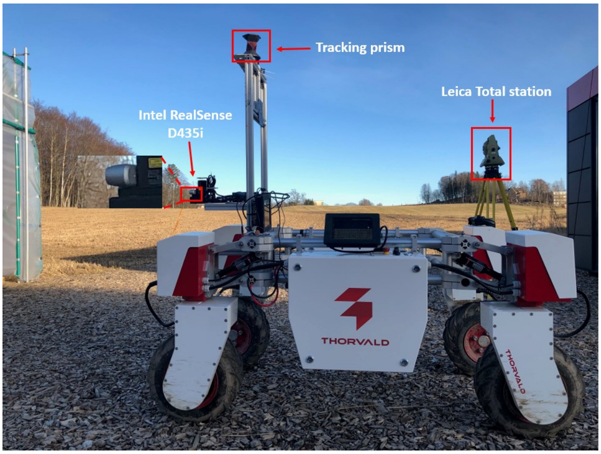


Figure IV.4: Robot setup for data collection.

traversable or non-traversable. The human expert who controlled the robot during data collection decides which pixel areas can be considered traversable. The human expert follows a similar definition of “traversable” as in [2], in which traversable area is defined as a space that the robot might get to from its current position by following a straight line and without collisions. This definition means that in cases, where the robot can observe several rows from its current position, the network should not classify neighbor row areas as traversable.

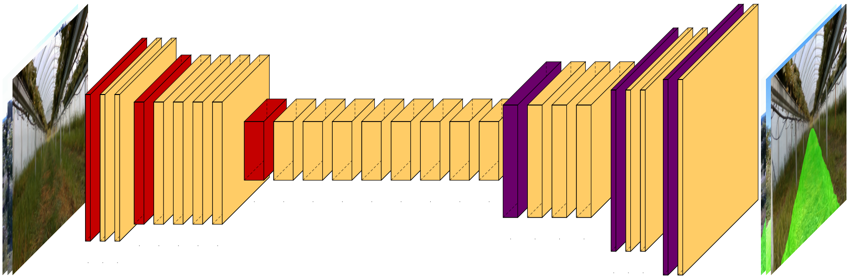


Figure IV.5: An illustration of a network architect that we use. This is similar to the structure of ERFNet in [9]. Red layers - downsample module, Yellow layers - variable receptive field, Purple layers - upsample module.

We train our network using the training tool in [7] with a Zotac Mini Gaming PC equipped with an Nvidia Geforce GTX 1070K, 16GB memory, and a quad-core Intel i5-7500T CPU. A sample architecture based on ERFNet, which we use, is shown in Fig. IV.5.

We also show examples of annotated data that we use for training in Fig.

IV.6.

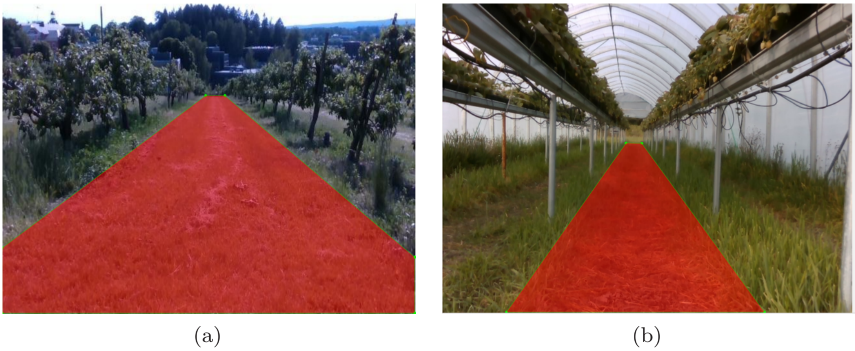


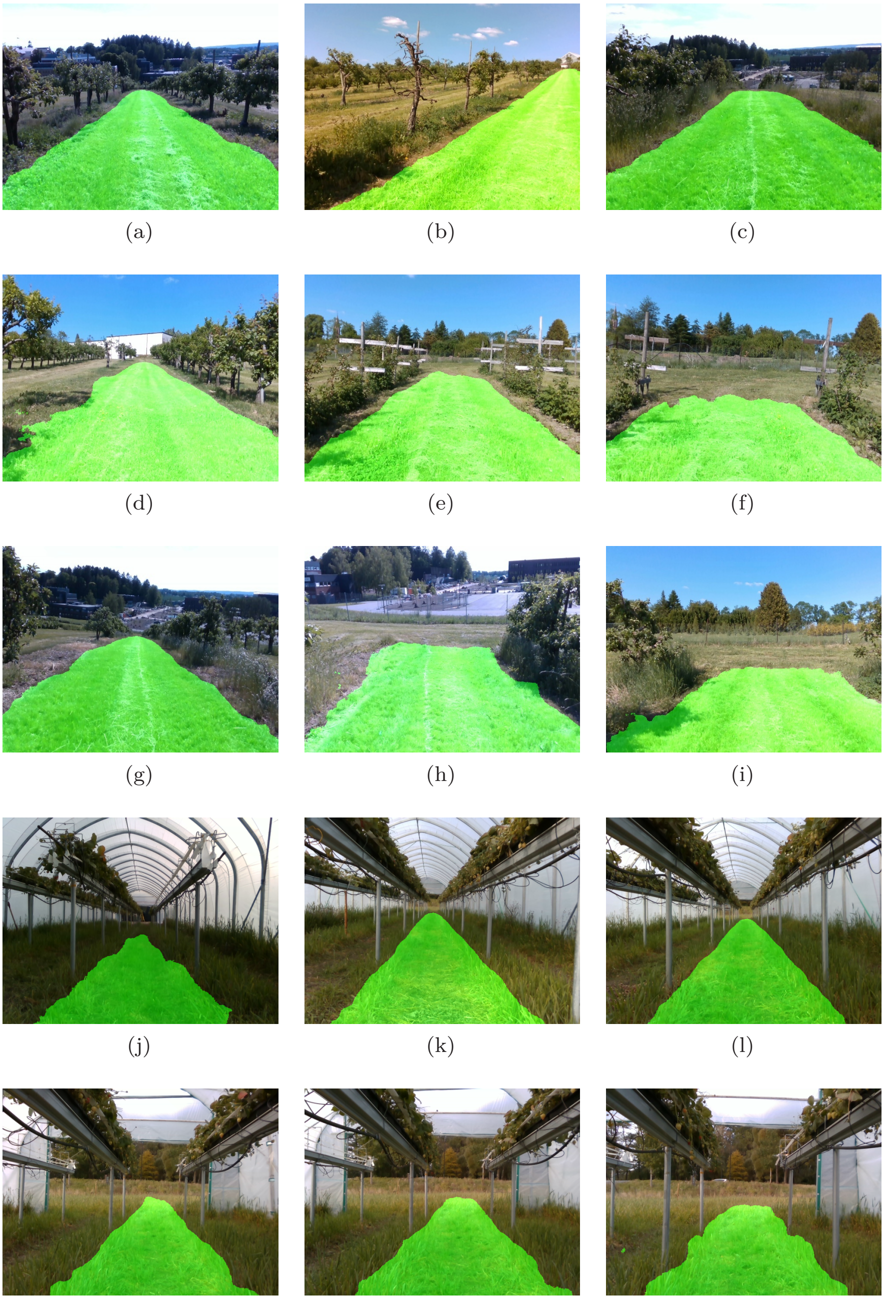
Figure IV.6: Screenshots of annotated images for training, where red areas depict traversable areas.

IV.3 Experimental Results and Discussions

IV.3.1 Results

We report our training results, including types of network architecture, the average accuracy (mAcc), mean Jaccard index (mIoU), and the mean Jaccard index of "traversable" class (mIoU of class 1) in Table IV.1. Note that for each network, we average the results of the best three trained models and report those values. As illustrated in Fig. IV.7, our trained network is able to segment traversable areas, which is the part of a row our robot is currently in and can safely translates to without collisions. Some test results including corner cases are presented in Fig. IV.7, where the network correctly ignores "traversable" areas of neighbor rows. Note that in case of a row with missing trees as in Fig. IV.7d, we explicitly do not want our robot to make a cross movement to a neighbor row, even it is safe to do so in this case. Obviously, for indoor environment, we do not want our robot to make any cross movement from one row to another. Our trained network was able to correctly identify the traversable areas inside our polytunnel (Fig. IV.7j-k).

We also report the average inference time (infer. time) per image in milliseconds by each network architecture when interfacing in ROS in Table IV.1. From experiments, we see that ERFNet gives us the fastest inference time at roughly 48 ms, which is approximately 20Hz. The slowest FPS is reported at approximately 5Hz using Darknet 53. Since our robot moves at a relatively low speed of 0.7 m/s, this inference rate is sufficient for row following performances. We do not observe significant differences in segmentation accuracy between different network architectures. Hence, it is up to an end-user to select a specific network architecture.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m) (n) (o)

Figure IV.7: Segmentation test results. Best viewed in color.

Architecture	mAcc	mIoU	mIoU of class 1	Infer. time
ResNet 18	0.986	0.941	0.899	∼ 54ms
ResNet 50	0.987	0.946	0.906	∼ 142ms
ResNet 152	0.985	0.939	0.895	∼ 190ms
Darknet 21	0.985	0.938	0.892	∼ 118ms
Darknet 53	0.987	0.947	0.908	∼ 206ms
ERFNet	0.986	0.941	0.898	∼ 48ms
Mobilenet V2	0.984	0.935	0.888	∼ 55ms

Table IV.1: Report of training

IV.3.2 Discussions

Currently, we have two main drawbacks in our work:

- We only consider traversable areas for in-row movements. We observe that headland areas are much different than in-row areas. Incorporating headland into our current network actually worsens its performance. Hence, we leave between-rows transition as a separate problem to solve.
- Ground truth determination is our bottleneck. Relying on a human expert for ground truth labeling is time-consuming and error-prone. However, to our knowledge, there are not any publicly available datasets that we can use for training or compare with. We envision a good ground truth must come from professional terrain surveying services, for which we plan to do in the future. Nonetheless, we want to stress at the current state, our network can accurately segment traversable areas on par with a human expert.

IV.4 Conclusions

In this work, we propose a unified solution for row following tasks in horticulture. Using a low cost camera, our solution is suitable for a wide range of agricultural robots. We present our approach to collect and train a fully convolutional neural network for segmenting traversable areas, which can be subsequently used for motion planning. We show that our trained networks are well generalized to different environments than existing methods. We also show that the inference time of our network is sufficiently fast for motion planning tasks. For future work, we plan to achieve a professional ground truth data for labeling traversable area using terrain surveying services and release our dataset to our agricultural robotics community.

IV.5 Acknowledgement

We would like to thank Antonio Candea Leite for helping with data collection on the orchard dataset.

References

[1] Bell, Jamie, MacDonald, Bruce A, and Ahn, Ho Seok. “Row following in pergola structured orchards”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 640–645.

[2] Bell, Jamie, MacDonald, Bruce A, and Ahn, Ho Seok. “Row Following in Pergola Structured Orchards by a Monocular Camera Using a Fully Convolutional Neural Network”. In: *Australasian conference on robotics and automation (ACRA)*. 2017, pp. 133–140.

[3] Biber, Peter et al. “Navigation system of the autonomous agricultural robot Bonirob”. In: *Workshop on Agricultural Robotics: Enabling Safe, Efficient, and Affordable Robots for Food Production (Collocated with IROS 2012), Vilamoura, Portugal.* 2012.

[4] Grimstad, Lars and From, Pål Johan. “The Thorvald II agricultural robotic system”. In: *Robotics* vol. 6, no. 4 (2017), p. 24.

[5] Grimstad, Lars et al. “A novel autonomous robot for greenhouse applications”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–9.

[6] He, Kaiming et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[7] Milioto, A., Mandtler, L., and Stachniss, C. “Fast Instance and Semantic Segmentation Exploiting Local Connectivity, Metric Learning, and One-Shot Detection for Robotics ”. In: *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*. 2019.

[8] Redmon, Joseph and Farhadi, Ali. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

[9] Romera, Eduardo et al. “Erfnet: Efficient residual factorized convnet for real-time semantic segmentation”. In: *IEEE Transactions on Intelligent Transportation Systems* vol. 19, no. 1 (2017), pp. 263–272.

[10] Sandler, Mark et al. “Mobilenetv2: Inverted residuals and linear bottlenecks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[11] Williams, Henry et al. “Improvements to and large-scale evaluation of a robotic kiwifruit harvester”. In: *Journal of Field Robotics* vol. 37, no. 2 (2020), pp. 187–201.

[12] Zhang, Ji et al. “3d perception for accurate row following: Methodology and results”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 5306–5313.

ISBN: 978-82-575-1714-4
ISSN: 1894-6402

Norwegian University
of Life Sciences

Postboks 5003
NO-1432 Ås, Norway
+47 67 23 00 00
www.nmbu.no

107944 / ANDVORDGRAFISK.NO