UNIVERSITÉ DE STRASBOURG

ÉCOLE DOCTORALE MSII

Laboratoire ICube, Strasbourg, France

THÈSE présentée par :

Danial POUR ARAB

soutenance prévue le : **30 juin 2023**

pour obtenir le grade de : **Docteur de l'Université de Strasbourg**
Discipline / Spécialité : Informatique

Calcul et optimisation de trajectoires pour véhicules autonomes soumis aux contraintes d'un environnement agricole

"Trajectory optimisation for autonomous vehicles under agricultural environment constraints"

THÈSE dirigée par :

Mme ESSERT Caroline	Professeur, Université de Strasbourg, France
M SPISSER Matthias	Responsable Innovation Lab, Technology and Strategy Strasbourg, France

RAPPORTEURS :

M LENAIN Roland	Directeur de Recherche, INRAE, Clermont-Ferrand, France
M SØRENSEN Claus G.	Professeur, Aarhus University, Denmark

AUTRES MEMBRES DU JURY :

M BAYLE Bernard	Professeur, Université de Strasbourg, France

Abstract

This thesis delves into the development of efficient *Complete Coverage Path Planning (CCPP)* approaches for agricultural wheeled robots, providing significant contributions to the field of autonomous agricultural robotics. The research encompasses a systematic review of existing challenges and proposed solutions, highlighting knowledge gaps and promoting collaboration between researchers and the agricultural industry. By offering valuable insights into the most effective CCPP techniques and technologies, researcher and technology developers can make informed decisions about adopting autonomous agricultural robotics for various tasks.

An essential part of the research is the creation of a dataset containing 2D and 3D models of 30 diverse fields in France. This dataset serves as a valuable resource for researchers and technology developers in agricultural robotics, allowing them to evaluate and validate path planning approaches across a wide range of real-world scenarios and agricultural settings. The inclusion of both 2D and 3D terrain models not only fosters a deeper understanding of each modeling approach's strengths and weaknesses but also promotes their integration into other techniques, inspiring further research and development.

The development of an efficient CCPP approach that generates optimal coverage paths for autonomous agricultural robots is another significant contribution of this research. This approach minimizes overlaps, non-working traveled distance, and operation time, leading to increased efficiency in real-world applications. It also handles headland coverage effectively. Furthermore, the exploration of a deep learning-based approach for field decomposition in agricultural CCPP highlights the challenges and complexities of the required data. Although not yet successful, this exploration serves as a foundation for further advancements in the fields of agricultural CCPP and autonomous agricultural robotics.

An advanced 3D hybrid path planning approach with multiple objectives is also presented in this thesis, capable of considering trajectory inclinations and various other factors to optimize coverage rate, overlaps, non-working traveled distance, and operation time. This novel method combines the strengths of previous CCPP approaches and shares the ability to effectively cover headlands. The advanced approach goes a step further by examining two distinct coverage patterns and offering faster computation times, which can be crucial in real-world applications. Additionally, it addresses complex field shapes while considering the robot characteristics and field accessibility.

This thesis provides a comprehensive and balanced overview of the work accomplished in the development of efficient CCPP approaches for agricultural wheeled robots. By reflecting on the journey undertaken and the knowledge gained through this research, practical implications and applications of these findings are discussed. This work aims to chart a path forward for continued exploration and advancement in the domain of autonomous agricultural robotics, benefiting both farmers and the industry as a whole.

Contents

List of Figures

List of Tables

List of Abbreviations

ABC	**A**rtificial **B**ee **C**olony algorithm
ACO	**A**nt **C**olony **O**ptimization algorithm
AI	**A**rtificial **I**ntelligence
ASCII	**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange
AVRP	**A**gricultural **V**ehicle **R**outing **P**roblem
BI	**B**ilinear **I**nterpolation
CCPP	**C**complete **C**overage **P**ath **P**lanning
CMC	**C**overage **M**ap **C**omputing
DEM	**D**igital **E**levation **M**odel
DL	**D**eep **L**earning
F2C	**F**ields**2C**over
GIS	**G**eographic **I**nformation **S**ystem
GNSS	**G**lobal **N**avigation **S**atellite **S**ystems
IDW	**I**nverse **D**istance **W**eighting
IoT	**I**nternet **o**f **T**hings
ITS	**I**mplement **T**ransition **S**tate
KML	**K**eyhole **M**arkup **L**anguage
MWD	**M**inimum **W**orking **D**istance
PDD	**P**redefined **D**riving **D**irection
PLY	**P**olygonal **F**ile **F**ormat
RCI	**R**ay **C**asting **I**nterpolation
RUSLE	**R**evised **U**niversal **S**oil **L**oss **E**quation
SA	**S**imulated **A**nnealing
UTM	**U**niversal **T**ransverse **M**ercator
VTK	**V**isualization **T**ool**K**it

CHAPTER 1

Introduction

Agriculture is a vital component of global food systems, playing a critical role in feeding the world's population. However, the industry faces unprecedented challenges that threaten its long-term sustainability. This thesis aims at exploring the use of *Complete Coverage Path Planning (CCPP)* for wheeled robots in agriculture to assist in the development of a more efficient approach to robotic agriculture, addressing the challenges faced by the industry.

In this introduction, we set the context for our research by discussing the key challenges in farming, including the impact of the COVID-19 pandemic, climate change and population growth. We then discuss potential solutions that have been proposed for more efficient precision agriculture and smart farming, including data-driven decision making and the use of autonomous systems such as wheeled robots.

Despite the potential benefits of autonomous systems in agriculture, there are still significant challenges to be addressed in their development and deployment, particularly in the area of CCPP. In this chapter, we also discuss the different aspects and challenges of CCPP for autonomous systems in agriculture, as well as the different types of agricultural operations in which CCPP is required.

Finally, we outline the specific research questions and objectives that this thesis aims at addressing and provide an overview of the structure of the thesis.

1.1 The State of Food and Agriculture

The COVID-19 pandemic has highlighted the vulnerability of agrifood systems to shocks and stresses, leading to increased global food insecurity and malnutrition [42]. Furthermore, climate change and population growth are putting increasing pressure on the world's food and agriculture systems.

The Intergovernmental Panel on Climate Change has projected that global warming will reach 1.5$^{\circ}C$ above pre-industrial levels by 2040, with significant impacts on agriculture [24]. These impacts include more frequent and intense heatwaves, droughts, floods, and storms, which can reduce crop yields and threaten food security. At the same time, the global population is expected to reach 9.9 billion by 2050 and 10.9 billion by 2100 [93], requiring a significant increase in food production to meet demand.

To address these challenges, the agriculture industry needs to find new ways to make farming more efficient, sustainable, and resilient. One promising approach is the use of autonomous systems, such as wheeled robots, to carry out tasks such as planting, spraying, and harvesting. Autonomous systems have the potential to increase efficiency, reduce costs, and minimize environmental impacts. However, there are still significant challenges to be addressed in the development and deployment of these systems.

1.2 Precision Agriculture and Smart Farming

Precision agriculture can be defined as a farming management concept that uses information technology, data analysis tools, and various sensors to optimize crop production and efficiency by making data-driven decisions. This approach involves collecting and analyzing data on a large scale, including crop growth, weather, soil fertility, pest and disease occurrences, and crop yield, and then using this information to make precise, targeted decisions on various aspects of crop management, such as irrigation, fertilization, and pesticide application. The goal of precision agriculture is to increase crop yields while reducing input costs, minimizing environmental impacts, and improving the overall sustainability of farming operations [109].

Smart farming is an evolution of precision agriculture that incorporates advanced technologies such as robotics, automation, and the *Internet of Things (IoT)* to further optimize farming operations [90, 157]. Smart farming relies on data from various sources, including sensors, unmanned aerial vehicles, satellites, and weather stations, to enable real-time decision making and precise management of agricultural activities such as planting, irrigation, fertilization, and harvesting. Machine learning algorithms and artificial intelligence are also utilized in smart farming to analyze vast amounts of data and provide insights for better crop management and yield prediction [89]. The ultimate goal of smart farming is also to increase productivity, reduce waste, and improve the overall sustainability and profitability of farming operations.

There are numerous applications of precision agriculture and smart farming, including:

- **Autonomous farming equipment**: using autonomous vehicles to perform various farming tasks such as planting, harvesting, and crop scouting [163].
- **Crop monitoring and management**: Smart farming technologies, such as IoT sensors and drones, can provide real-time data on crop health, soil moisture levels, and nutrient content, allowing farmers to make informed decisions on irrigation and fertilizer application [61].
- **Livestock management**: IoT sensors can be used to monitor animal health and behavior, as well as track herd movements, enabling farmers to optimize feeding and breeding practices [4].
- **Precision irrigation**: using sensors and data analysis to optimize irrigation, reduce water waste, and increase crop yields [126].
- **Yield prediction**: using machine learning algorithms and artificial intelligence to analyze data and predict crop yields [153].

- **Supply chain optimization**: using data analysis to optimize logistics and reduce waste in the supply chain, from farm to market [88].

The central focus of this thesis is on the autonomous systems, specifically wheeled robots, for achieving complete coverage of agricultural fields. Specifically, we aim at addressing the challenges of CCPP for autonomous agricultural operations, which requires a comprehensive and efficient approach to navigating and covering large fields while performing a particular task. In the following section, we will outline the essential requirements needed to realize an autonomous system in agriculture, including the necessary hardware and software components.

1.3 Autonomous Vehicles in Agriculture

The development of an autonomous system in agriculture is a complex process that requires the integration of several specialized and interrelated subsystems. Among these, path planning, path following, positioning systems, and perception systems are crucial for the success of the system. In addition, safety features are also important to ensure the safe operation of the system in the presence of human workers, other vehicles, or obstacles.

Path planning is the process of generating a feasible and optimal path from a starting point to a goal point while considering the environment's constraints and the vehicle's kinematic and dynamic constraints. This process can also involve generating a path that covers the entire field, ensuring that the entire area is covered while considering and optimizing a set of constraints.

Path following, also known as path tracking, is the process of accurately tracking the planned path with the vehicle's actuators, which can be achieved by different control techniques, including proportional-integral-derivative control, model predictive control, or nonlinear control [78, 155]. More recently, advancements in steer control techniques, including the application of reinforcement learning, have further improved path following capabilities [44, 56].

Accurate positioning is also essential for the reliable operation of autonomous systems. *Global Navigation Satellite Systems (GNSS)* can provide position information with sub-meter accuracy, but their performance can be degraded in some scenarios, such as in dense vegetation. In these cases, alternative positioning systems, such as visual odometry [97], inertial measurement units [156], or LiDAR-based localization [94], can be used to complement or replace GNSS.

Perception systems are responsible for sensing and understanding the environment, including detecting and tracking obstacles, identifying crops, and estimating terrain characteristics [23, 114]. These systems typically employ various sensors, including cameras, LiDAR, radar, and ultrasonic sensors, in conjunction with algorithms for object detection, segmentation, and tracking [16]. To achieve a robust perception system, sensor fusion methods are crucial, as they enable the seamless integration of data from multiple sensors, enhancing the overall understanding and representation of the environment [3, 9].

Safety features, such as obstacle avoidance, collision detection, and emergency stop mechanisms, are essential to ensure the safety of human workers and other vehicles in the vicinity of the autonomous system. These features can be achieved by combining data from different sensors and using advanced algorithms to detect and react to potential hazards [154].

1.3.1 Different Types of Autonomous Vehicles

Autonomous systems in agriculture can take different forms depending on the application and the specific requirements. On one hand, it is possible to retrofit traditional farm machinery, such as tractors or sprayers, with sensors, actuators, and controllers to enable both manual and autonomous operation. This approach offers the advantage of leveraging existing equipment and infrastructure, as well as the possibility of switching between manual and autonomous modes as needed. Fig. 1.1a shows an example of a tractor equipped with required sensors.

(A) John Deere [69]

(B) Naïo [92]

(C) Combined Powers [25]

(D) Nexus Robotic [95]

FIGURE 1.1: Different types of autonomous robots in agriculture

On the other hand, it is also possible to design and build custom autonomous robots tailored to specific tasks and environments. These robots can be smaller and more agile than traditional farm machinery, allowing them to navigate in challenging terrain with greater ease. Figs. 1.1b, 1.1c and 1.1d show some examples of small autonomous robots designed for a specific or multiple tasks.

In the interest of simplicity, we will refer to all the autonomous vehicles discussed in this section as *autonomous robots*. However, the focus of this study is on autonomous robots that are specifically designed to carry equipment or implements for a particular task.

1.3.2 Prototype Autonomous Robot

To design, test, and validate the different subsystems necessary for achieving full autonomy, the Innovation Lab at *Technology & Strategy* in Strasbourg developed a prototype robot. As shown in Figure 1.2, the robot is equipped with a range of sensors, including LiDAR, RGB-D camera, and GNSS, as well as advanced algorithms for path following, navigation, obstacle detection, and safety. The robot is designed to navigate autonomously in deformable terrain.

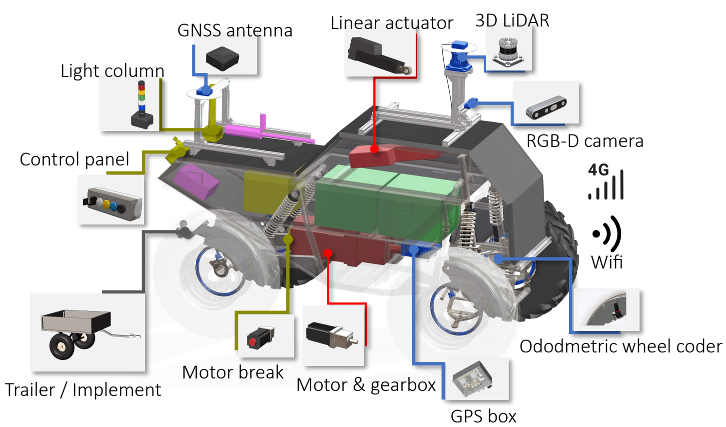


FIGURE 1.2: Prototype of an autonomous robot designed and developed at Innovation Lab at Technology & Strategy in Strasbourg

The primary objective of this thesis is to focus on the path planning subsystem, with a specific emphasis on developing a CCPP approach for an autonomous robot equipped with a task-specific implement. The goal is to obtain a planning algorithm compatible with real-world use on a field, in conjunction with other subsystems that have been developed and improved by our team, to demonstrate the effectiveness and practicality of the proposed approach.

In the next section, we will discuss different types of agricultural operations, specifically those that require complete coverage of the field using a robot equipped with a specialized implement to perform the task.

1.4 Agricultural Operations and Required Machinery

As highlighted earlier, creating an autonomous system in agriculture is a challenging task that involves integrating multiple specialized and interconnected subsystems. The successful integration of these subsystems is critical to ensure efficient and effective operation of the system in the field. Furthermore, the requirements of each subsystem might vary depending on the specific agricultural operation being performed. Additionally, different types of farming practices, such as arable farming and orchard farming, may have distinct requirements for their respective operations. In this section will explore various types of agricultural operations that could benefit from the use of autonomous systems, specifically wheeled robots.

1.4.1 Orchard Farming

Orchard farming involves cultivating fruits and nuts, such as apples, pears, cherries, and almonds, in closely planted trees, which can make it challenging to navigate and perform tasks using traditional farm machinery.

Autonomous systems can address these challenges by providing a solution to navigate between the trees and perform tasks such as pruning, spraying, and fruit picking. These systems can also help reduce labor costs and improve efficiency by using sensors and machine learning algorithms to optimize irrigation and fertilization schedules and predict disease outbreaks.

The use of autonomous systems in orchard farming is facilitated by the location of the existing trees. The trajectories are thus already predefined and it is sufficient to determine the optimal sequence of trajectories to plan the path. However, in this study, our focus is primarily on operations performed in arable farming, which typically involve large fields where generating a complete coverage path is necessary and the direction of parallel trajectories is not predefined by such constraints.

1.4.2 Arable farming

Arable farming is the cultivation of crops on a large scale in fields. It is one of the most common types of farming and is characterized by crops that are planted in rows and grown in a particular pattern. The use of autonomous systems in arable farming can be particularly advantageous due to the large size of fields and the repetitive nature of tasks such as:

- **Plowing and tillage**: preparing the soil for planting
- **Seeding**: planting seeds in the soil to grow crops
- **Planting**: planting crops in the soil
- **Fertilizing**: adding nutrients to the soil to support crop growth
- **Irrigation**: providing water to the crops to ensure proper growth
- **Spraying**: applying pesticides or herbicides to crops to protect them
- **Harvesting**: collecting crops when they are ready for harvest
- **Mowing**: cutting down grass or other vegetation in a field

Autonomous systems can be used to perform these tasks efficiently and precisely, reducing the need for manual labor and minimizing the risk of human error. For example, autonomous seed drills can plant seeds at precise intervals and depths, ensuring optimal plant growth. Autonomous spraying systems can also apply pesticides and herbicides precisely, reducing the amount of chemicals used and minimizing the risk of environmental damage.

(A) Seeding (B) Fertilizing

(C) Harvesting (D) Spraying

FIGURE 1.3: Some examples of operations in arable farming

1.5 Path Planning

After contextualizing the use of autonomous systems in agriculture and identifying the different agricultural operations that can benefit from their deployment, we can now delve into a more comprehensive definition of path planning.

Path planning is an essential component in the field of robotics and autonomous systems, as it deals with the problem of finding a feasible path for a robot to move from an initial position to a goal position while avoiding obstacles and fulfilling certain requirements or constraints. Path planning can be broadly classified into two main categories: point-to-point path planning and complete coverage path planning.

Point-to-point path planning focuses on finding a path from a starting point to a destination point, usually optimizing for specific criteria such as distance, energy consumption, or time. This type of planning is often employed in domains like autonomous vehicles [66], drones [170], and robotic arms [159]. Some popular algorithms used for point-to-point path planning include A* [52], Dijkstra's algorithm [30], Rapidly-exploring Random Trees [76], and Probabilistic Roadmaps [70].

Complete coverage path planning, on the other hand, deals with the problem of covering an entire region or workspace while satisfying certain constraints or requirements [165]. The concept of generating a path that covers a given area has been studied for decades and has been applied in various domains requiring different sorts of environments and operation requirements such as geophysical surveys [8], model reconstruction and mapping [18], vacuum cleaning [141], forest monitoring [161], marine growth removal [53], water sampling [84], 3D printing [167], or demining [32], to name a few.

In agriculture, CCPP is an essential aspect of the majority of agricultural operations. It involves navigating the robot through the entire field in an optimized and efficient manner, covering all areas with minimal overlaps. CCPP requires generating a feasible and optimal path that considers the robot's kinematic and dynamic constraints, the characteristics of its implement, and optimizing a set of objectives.

However, as illustrated in Fig.1.3, different operations require different types of implement and have distinct constraints that need to be taken into consideration which make it challenging to propose a generic CCPP. For example, seeding requires an implement that works with precision and at a specific depth, which requires the robot to move straight forward while lowering or raising it. In contrast, the implement required for spraying does not touch the ground, but the robot must avoid damaging the germinated crops.

Additionally, CCPP becomes even more challenging when the inclination and slopes of a field need to be considered, as these can have a direct impact on the performance of the robot and the quality of the results. This requires a 3D model of the field to be constructed, which adds an extra layer of complexity to the CCPP problem. Nevertheless, taking into account these factors is crucial for optimizing the efficiency and effectiveness of the generated path, particularly in terms of energy consumption and soil erosion. Therefore, it is important to develop CCPP approaches that consider the specific requirements of each operation and implement, as well as both the 2D and 3D characteristics of a field to ensure optimal performance.

1.6 Motivations

The primary motivation of this thesis is to develop a generic and efficient CCPP approach for autonomous agricultural robots that can address the complexities of various agricultural operations detailed in Section 1.4.2. Specifically, we aim at designing CCPP methods that can consider the specific requirements of each operation and implement, while also taking into account the 3D characteristics of the field and optimizing multiple objectives such as coverage and overlap area, non-working traveled distance, energy consumption, and soil erosion. Our approach should be applicable to a wide range of agricultural applications and field shapes, ensuring optimal performance. Ultimately, the objective of this thesis is to contribute to the advancement of CCPP approaches for autonomous agricultural robots, providing a foundation for further research and development in this area.

1.7 Contributions

The thesis proposes several significant contributions to the field of agricultural robotics, which are outlined in the following paragraphs.

The first contribution is a systematic review of the various algorithms proposed in the literature to address the challenges of CCPP for wheeled agricultural robots. This review provides a comprehensive analysis of the solutions proposed in the literature and the important aspects of the problem that may affect and improve the final result of a CCPP approach. A total of 48 articles were categorized and analyzed from different perspectives, including terrain modeling, constraints modeling, and path planning.

The second contribution provides a detailed discussion of the approaches for generating both 2D and 3D models of a field, along with a dataset of 30 fields located in France. This contribution can serve as a guide for researchers in the field of agricultural robotics, as it provides a comprehensive overview of the data required for creating accurate and efficient solutions for agricultural operations. The dataset, that is publicly available on Zenodo [115], can provide valuable insights into the evaluation of future path planning approaches, helping researchers to better understand the advantages and limitations of both 2D and 3D terrain modeling.

The third major contribution is the development of an efficient CCPP approach for generating optimal paths for autonomous agricultural robots to cover fields with high accuracy while minimizing overlaps, non-working path length, and overall travel time. This approach incorporates a tree-based intelligent search algorithm, which takes into account the geometry of the robot and its implement, as well as other important factors, resulting in efficient and optimal solutions. The approach is designed to handle complex field shapes and sizes, while also providing automated headland coverage.

The fourth contribution is an exploration of a deep learning-based approach for field decomposition in agricultural CCPP, which aimed to enhance the adaptability and efficiency of existing methods. Although the results of this study were not as convincing as hoped, it highlighted the challenges and complexities of the dataset and the need to reconsider certain assumptions about farmers' preferences, operation, and machinery requirements. This study also emphasized the importance of developing a robust CCPP approach that can handle multiple dividing lines and diverse constraints efficiently.

The fifth and final contribution is the development of an advanced 3D hybrid path planning approach with multiple objectives for complete coverage. This approach combines the strengths of the previous approach and an open-source algorithm to generate coverage paths for agricultural robots that consider working trajectory inclinations, which have a direct impact on soil erosion and energy consumption. The method optimizes several other objectives, including coverage rate, overlap rate, non-working traveled distance, and operation time. The approach is capable of exploring all possible driving directions and considers two different coverage patterns (sequential and row-skip). The method's performance is evaluated and compared to that of the original proposed algorithm, demonstrating its effectiveness and efficiency.

1.8 Thesis Structure

Chapter 1 - Introduction: This chapter sets the context for the thesis by discussing the importance of agriculture, the challenges faced by the industry, and the potential of precision agriculture, smart farming, and autonomous vehicles to address these challenges. It also introduces the concept of CCPP and highlights the motivations and contributions of this research.

Chapter 2 - State of the Art: This chapter provides a comprehensive overview of the current state of the art in CCPP for autonomous wheeled agricultural robots, submitted to the Journal of Field Robotics. It discusses the challenges involved in agricultural CCPP, such as terrain modeling, constraints modeling, and path planning. The chapter reviews proposed solutions in the literature, factors that affect CCPP results, and different validation methods for evaluating CCPP algorithms in agriculture. The chapter concludes by emphasizing the need for further research to develop more accurate models to account for all the constraints required in CCPP.

Chapter 3 - Terrain Modeling: Data Acquisition and Dataset Construction: In this chapter, the process of terrain modeling is explored, discussing data acquisition methods and the construction of both 2D and 3D models. The chapter introduces a publicly available dataset of thirty fields located in France for evaluating various applications and understanding the advantages and limitations of different terrain modeling approaches. The dataset can be found on Zenodo [115].

Chapter 4 - Problem Statements and Challenges: In this chapter, the challenges and constraints associated with CCPP for various agricultural operations are discussed. The chapter proposes an exhaustive CCPP approach, evaluates its performance and limitations through simulations, and sets the foundation for future, more efficient CCPP approaches.

Chapter 5 - Intelligent Tree-based Search: This chapter presents a novel CCPP approach for efficiently generating optimal paths for mobile robots in agricultural fields, which has been accepted for publication in the Journal of Field Robotics [116]. By exploiting tree exploration, the method successfully adheres to hard constraints while optimizing soft constraints such as worked area, overlaps, non-working traveled distance, and operation time. The chapter delves into the methodology, offers analytical results, and provides an in-depth discussion on the impact of various considered factors.

Chapter 6 - Extensions: Row-Skip Pattern and Deep Learning-based Field Decomposition: This chapter presents two extensions to the original CCPP approach, proposed in Chapter 5, to improve its efficiency and versatility. The first extension incorporates a row-skip coverage pattern, which was accepted and presented at the 9th International Conference on Automation, Robotics and Applications (ICARA 2023). This pattern allows for more efficient field coverage while minimizing overlaps, non-working traveled distance, and operation time. The second extension introduces a Deep Learning-based method for automating field decomposition, adapting it to various field characteristics and enhancing the overall efficiency of the CCPP approach. The chapter provides a detailed description of both extensions, discussing their methodologies, results, and impact on agricultural robot path planning.

Chapter 7 - Advanced 3D Hybrid Path Planning with Multiple Objectives for Complete Coverage: This chapter introduces a sophisticated hybrid CCPP approach for autonomous agricultural robots, seamlessly merging the advantages of prior methods with the F2C algorithm. Taking into account multiple driving directions, coverage patterns, and field inclination, the approach elegantly addresses soil erosion and energy consumption optimization in an indirect manner.

Chapter 8 - Conclusion & Perspectives: This final chapter synthesizes and reflects upon the research conducted throughout the thesis, summarizing major findings, contributions, and practical implications in the development of efficient CCPP approaches for agricultural wheeled robots. It acknowledges the study's limitations and outlines potential avenues for future research, providing suggestions for refining, optimizing, and expanding the CCPP approaches to address a broader range of scenarios and applications in agriculture.

CHAPTER 2

State of the art

In this chapter, we provide an overview of the state of the art in CCPP for wheeled robot performing agricultural operations.

We begin by discussing the challenges involved in agricultural CCPP, including terrain modeling, constraints modeling, and path planning. We then review the proposed solutions in the literature and highlight some of the important factors that may affect the final result of a CCPP approach. Additionally, we examine different types of validation that can be used to evaluate the effectiveness of CCPP algorithms in agriculture.

The primary goal of this chapter is to provide a comprehensive understanding of the current state of the art in CCPP for autonomous agricultural robots. By examining the challenges, proposed solutions, and validation methods, we hope to offer insight into the development of effective CCPP algorithms for autonomous agricultural robots.

2.1 Introduction

CCPP is a key challenge for developing an autonomous system for the majority of field operations such as ploughing, harrowing, seeding and plowing, to name a few. In general, CCPP is the process of generating a path that completely covers the area of interest in a precise and feasible manner while avoiding obstacles and reducing operating expense and time [48].

A CCPP approach consists of three different parts; 1) terrain modeling, 2) constraints modeling and 3) path planning. These components might be adapted for various robotic applications according to the environment, the robot's capabilities and the intended application but in general, if these procedures are followed, a complete coverage of an area without overlapping should be achieved in a limited amount of time. The generated path should also be collision-free and sufficiently smooth for the robot to travel on.

The terrain can be represented using a 2D or 3D model, depending on the field's specifications. A 2D depiction may be adequate for a perfectly flat area in order to carry out an exact path planning. However, to accurately design a path planning algorithm for non-flat fields that include slopes and height variations of altitude, a

3D model is necessary. Regarding energy consumption, soil compaction, soil erosion, or robot operability, a number of restrictions that rely on the field's relief can be modeled and taken into account.

In agriculture, path planning is often performed at two separate levels. A coverage map is first generated using the field data including the position of the static obstacles by a **Coverage Map Computing (CMC)** algorithm. The best path is then allocated to one or more robots using the resultant coverage map. This second challenge is referred to as **Agricultural Vehicle Routing Problem (AVRP)**. In order to replan paths, a routing algorithm has to be robust and flexible. For instance, if a robot is blocked by an animal or stops functioning, the path assignment may need to be revised to ensure a high quality execution of the path.

It is well known that the CCPP in agriculture using a non-holonomic robot is highly challenging to solve optimally. Nearly majority of the agricultural robots in use today are far from holonomic. It takes time to turn an agricultural robot. It makes the procedure much more difficult [100]. Most surveys on this subject address CCPP algorithms in general [43, 72]. Almadhoun et al. [5] conducted a survey on multi-robot CCPP for model reconstruction and mapping. Edan, Han, and Kondo [36] published a more comprehensive survey on agricultural automation systems, that comprised field equipment, irrigation systems, greenhouse automation, animal automation systems, and automated fruit production systems. Santos et al. [129] most recently published a brief review on path planning for ground robots in agriculture. They classified path planning approaches into two categories: Point-to-Point and complete coverage path planning. In these surveys, several CCPP algorithms in agriculture have been briefly discussed. However, certain additional factors that have a significant impact on path planning for agricultural applications, such as terrain modeling and constraints unique to agriculture, haven't been taken into account. Therefore, it is necessary to study various CCPP applications in agriculture, including their pro's and cons, and to provide a perspective for further research.

The rest of this chapter is structured as follows: Section 2.2 goes through how this survey was conducted. In Section 2.3 we review various methods for terrain modeling and explain about the data required to produce an accurate model of the field. Section 2.4 discusses various constraints that should be taken into account in agricultural CCPP. Different path planing methods are detailed in Section 2.5. In Section 2.6 various types of validation are discussed. Finally Section 2.7 concludes this chapter.

2.2 Methodology

The primary aim of this chapter is to identify and review in a systematic manner: Different challenges of CCPP, the proposed solutions in the literature, and also some important aspects of the problem that may affect and improve the final result of a CCPP approach.

To achieve this goal, several publications were gathered using the following search terms from online research databases including IEEE Xplore, Science Direct, and Google Scholar:

- ("agricultural" OR "agriculture") AND ("trajectory planning" OR "path planning" OR "coverage path planning" OR "CPP")

- ("agricultural" OR "agriculture") AND ("routing problem" OR "vehicle routing problem: OR "VRP")

The top 300 articles for each of the two search terms and for each of the three search engines were chosen after sorting the results using the default relevance criterion. In total 1800 articles were selected. 1800 papers in total were chosen. By using the following exclusion criteria, irrelevant publications were eliminated:

- Article is a duplicate
- Article is not related to the agricultural sector
- Article is not about CCPP
- Article is about CCPP for aerial robots in agriculture
- Article is about indoor agriculture
- Article is about positioning and navigation systems
- Article is not written in English
- Article is published before 2007
- Article is a review or survey, paradigm or benchmarking
- Full text of the article is not available

Once these exclusion criteria were applied 49 articles remained. The publication year of these papers is depicted in Fig. 2.1 and their country of origin is shown in Fig. 2.2. Base on the following aspects, an overall analysis of the remaining articles was performed: terrain modeling, constraints modeling and path planning. Depending on the articles, these categories, which typically stand for the three CCPP algorithm steps, are discussed in various degrees of detail. They served as inspiration for this chapter's organization as well.

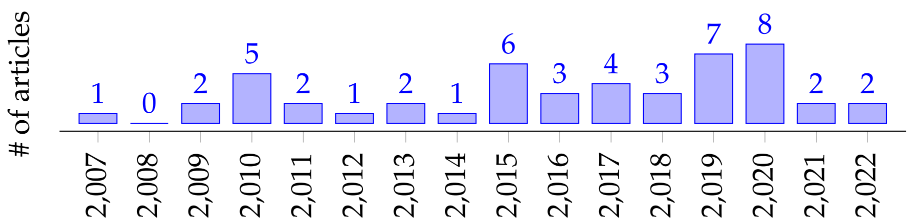


FIGURE 2.1: Year of publication of selected articles.

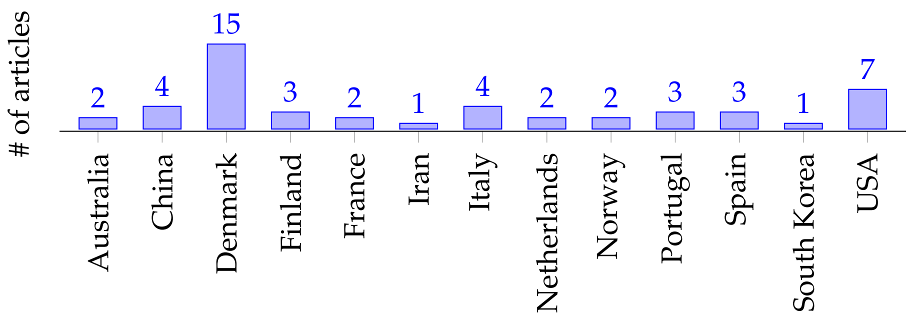


FIGURE 2.2: The country where algorithms were tested or reported country for first author of selected articles.

The articles were then carefully examined to determine their response to each challenge by answering to the following questions:

- Which techniques have been used to represent the terrain ?
- Which constraints have been considered and how have they been modeled ?
- Which approach has been used to obtain a coverage map of the field?
- How the coverage map was assigned to one or several robots ?
- How the study has been validated ?

The following sections detail the result of this analysis.

2.3 Terrain Modeling

The initial stage in implementing CCPP is modeling the terrain, which may also include the locations of the accessible edges of the field to enter and exit and the location of service unites. A field could be divided into three zones: Workable zone, passable zone and impassable zone or obstacle. A workable zone is the area that the robot is authorized to traverse while its implement is either on or off. For instance the main part of the field that is needed to be cover by the robot's implement is a workable area. A passable zone is defined as an area that the robot is authorized to cross only while its implement is off or its not in touch with the ground or crops. For instance uncultivable part of a farmland is a passable area that could be used for performing half-turns but crossing them while the implement is on may damage the implement and/or the robot. A zone that the robot cannot traverse is referred to as an impassable zone. This includes any obstacles located inside the field, such as trees, lakes, and waterways.

Table 2.1 summarizes the data used for terrain modeling, as well as some included features. In the rest of this section different considerations and approaches of field representation based on 2D and/or 3D data are reviewed.

2.3.1 2D Representation

The majority of CCPP approaches only work with 2D field data. In these works, the field is typically shown as a polygon that, depending on the field borders, may be convex or concave. The obstacles and impassable zones inside the field borders are represented by small polygons inside the field polygon.

By driving around the field and obstacles boundaries, Zhou et al. [169] stored their coordinates. They described a ring as a collection of connected, ordered line segments where its start and the end point are the same. The field is then represented by a combination of zero or several inner rings and one outer ring. The segments of an inner ring are ordered counterclockwise, whereas the segments of an outer ring are ordered clockwise. It is also possible to register and export these coordinates by using a **Geographic Information System (GIS)** software. The second strategy is more frequent and simpler.

As summarized in Table 2.1, The assumption that the field is perfectly flat serves as the foundation for the majority of research projects. This may not be the case depending on the region where the field is located. The distance between computed trajectories on the topographic surface changes when the outcome of a 2D planning is projected to a 3D terrain, which causes skipped and/or overlapping areas between adjacent trajectories on the slopes [51]. Besides, a discrepancy between the computed total area based on 2D data and the actual area of the field may arise in cases of significant elevation changes across the field. This discrepancy will thus have an impact on all other metrics and measures, including the worked area and overlaps. For instance, as illustrated on Fig. 2.3, the distance between two adjacent trajectories on the 2D surface is not the same comparing to the distance between their projections on the 3D surface.

Ref.	2D data	3D data	Entrances	Obstacles	Concave field
[12, 13, 14]	✓				✓
[15]	✓		✓		✓
[17]	✓				✓
[19, 20]	✓			✓	✓
[21]	✓		✓		
[22]	✓				✓
[26, 28, 27]	✓				
[31, 33]	✓	✓		✓	✓
[37]	✓		✓		✓
[40]	✓				✓
[47]	✓			✓	✓
[50]	✓				✓
[49]	✓	✓		✓	✓
[51]	✓	✓		✓	✓
[48]	✓		✓	✓	✓
[62]	✓		✓		✓
[64, 63]	✓				✓
[65]	✓		✓		
[68]	✓			✓	✓
[67]	✓	✓			✓
[71]	✓				✓
[86]	✓				✓
[87]	✓				
[96]	✓		✓		✓
[100, 101]	✓			✓	✓
[112]	✓		✓		✓
[110]	✓			✓	✓
[111]	✓		✓	✓	✓
[130]	✓			✓	✓
[136, 135, 134, 137]	✓		✓		✓
[139]	✓	✓	✓		✓
[147, 148]	✓				✓
[151]	✓				✓
[168]	✓		✓		✓
[169]	✓			✓	✓
[170]	✓			✓	✓

TABLE 2.1: Data used for terrain modeling

Erosion of the soil and water conservation, which are a major concern when planning trajectories on a field, may not be optimized when elevation variations across the field are ignored [67]. Van Doren, Stauffer, and Kidder [152] demonstrated that plots planted with the direction perpendicular to the slope had lower soil losses and surface runoffs than those planted with the same direction as the slope.

Dogru and Marques [31] stated that climbing increases significantly the consumption of energy. As a result, disregarding elevation variations also results in inaccurate energy cost estimates, which in consequence leads to incorrect optimization of energy consumption on non-flat fields. As a result, CCPP algorithms built on a 3D model of the field have a huge potential to generate more accurate and better paths.

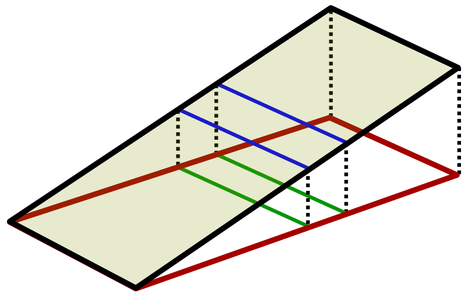


FIGURE 2.3: The black polygon represents the field and the red polygon is its 2D projection. The green trajectories are planned based on the 2D representation of the field and the blue trajectories are their projections on the 3D model of the field

2.3.2 3D Representation

To generate a 3D model of a field, usually the *Digital Elevation Model (DEM)* are used in addition to the field's 2D polygon. DEM data are structured as a grid of squares or cells. A DEM file is arranged as an *American Standard Code for Information Interchange (ASCII)* grid file containing in its header the file id, cell length, number of grid lines along x-axis, number of grid lines along y-axis, minimum and maximum x values of the grid, minimum and maximum y value of the grid and minimum and maximum elevation values of the grid in the *Universal Transverse Mercator (UTM)*. Then elevation values of the grid cells (*i.e.*, z values) are ordered in rows in the rest of the file representing the elevation matrix [49].

Hameed [49] proposed an approach to constructed a set of parallel tracks using the 2D field polygon. Afterwards, they estimated the elevation across each track. Hameed, Cour-Harbo, and Osen [51] also proposed an approach to constructed a set of parallel tracks using the 2D field polygon. Afterwards, they generated a 3D representation of the field using DEM files and applying bi-linear interpolation in order to estimate accurately skips and overlaps. As a hypothesis, considering that the surface of the field is linear, Shen et al. [139] used DEM data to generate a linearized 3D structure of the field surface.

After acquiring the sparse altitude of the field by driving the robot over it, Dogru and Marques [31, 33] applied the Kriging method as an interpolate method to estimate the altitude of any points on the field's surface. Kriging can provide a linear unbiased prediction of the intermediate values [102]. They acquired also a dense altitude of the field to compare and validate their terrain modeling approach. Jin and Tang [67] implemented B-form splines interpolation method by tensor product splines to describe the topographic surfaces in 3D space by using DEM grid and 2D field shape.

2.3.3 Discussion

As described in Section 2.3.1 and illustrated on Fig. 2.3, terrain modeling may have an impact on the metrics and consequently it may lead to an inaccurate outcome. This is mainly due to the projection of a non-flat field to a 2D plane where distances and angles may not remain intact. Besides the covered area and skips/overlaps, it might significantly affect the computations in term of soil erosion and energy consumption.

The complexity of the computations, however, may increase by integrating a 3D model of the field. This could be the reason that, as shown in Table 2.1, only few researchers have included 3D data in their approaches. Using 2D surfaces to simplify some aspects of the algorithms and then 3D surfaces to refine the path or compute the metrics is perhaps an efficient solution to decrease computational cost without significantly lowering accuracy. As summarized in Table 2.1, few authors already integrated such a hybrid terrain modeling in their approaches. Another strategy might be to use GPUs or parallelization to speed up the computations.

In the context of agricultural 3D modeling, the effects of the various interpolation methods used to construct a 3D model have not been thoroughly investigated. The quality of the terrain modeling may vary depending on the applied interpolation method. For instance some methods may result a smoother surface comparing to other methods. Therefore, it could be useful to compare the effectiveness of interpolation and 3D reconstruction methods for agricultural terrain modeling.

In other domains or even for other applications of field robotics such as point to point path planning and ground crops surveying, however, Several 3D modeling methods have already been proposed. The scope of this thesis does not allow for a detailed discussion on these researches. Therefore, we refer interested readers to [164] to discover more about the impact of various interpolation methods on 3D modeling approaches for object reconstruction, to [106] which reconstructed a 3D model of ground crops using airborne LiDAR technology, to [161] for a triangulation-based approach of 3D modeling of a general terrain, and to [127, 128, 131] for agricultural terrain modeling by the robot's laser scan.

The field's accessibility is a crucial factor to consider for constructing a realistic model of a field and consequently finding a feasible path. It is naive to assume that a field can be accessible from all of its edges, which is not true for the majority of fields. For most of the fields, this assumption may damage the robot when it cross an edge of the field or even may damage the neighboring field. It could also lead to a solution that the farmer can't apply in practice. An approach that was frequently used to address this issue was to consider an inner offset for a field polygon as headlands and perform the headland coverage afterward. Therefore, as illustrated in 2.1, most of researchers considered the initial location of the robot somewhere inside the field while others considered one or two points as entrance. Only Nørremark, Nilsson, and Sørensen [99] considered the the field's accessibility as line segments on the field borders.

Considering one or two points as entrance for the robot would strongly limit the possible solutions. Therefore, an ideal scenario is to consider the accessibility of a field, to determine several entry for the robot and to authorize the robot to finish its path anywhere on an accessible edge of the field.

2.4 Constraints

CCPP is inherently a multiple objective problem. Numerous constraints come into play due to the nature of a field such as slopes and static obstacles inside the field, as well as due to the features of the robot and the implement attached to it. The implement can be either on or off.

One of the main goals of every CCPP approaches is to maximize the coverage rate which is also known as the worked area. The worked area is computed as the area covered by the implement while it is on. Consequently, the distance traveled while the implement is on and off are known respectively as working and non-working traveled distances. Typically, the worked area is calculated as the working traveled distance multiplied by the length of the implement. The length of the implement is also referred to as the working width.

Ref.	Operation time	Non-working distance	Obstacle Avoidance	Skips/Overlaps
[12, 13, 14]		✓		
[15]		✓		
[17]		✓		✓
[19, 20]	✓	✓	✓	✓
[21]		✓		
[22]		✓		
[26, 28, 27]		✓		
[31, 33]			✓	
[37]		✓		✓
[40]		✓		
[47]			✓	
[50]		✓		✓
[49]		✓	✓	
[51]			✓	✓
[48]		✓	✓	
[62]	✓	✓		
[64, 63]		✓		
[65]	✓	✓		✓
[68]		✓	✓	
[67]		✓		✓
[71]		✓		
[86]		✓		
[96, 99]	✓	✓		
[100, 101]	✓	✓	✓	
[112]		✓		
[110]		✓	✓	
[111]		✓	✓	
[130]			✓	
[136, 135, 134, 137]	✓	✓		
[139]		✓		
[147, 148]		✓		
[151, 149, 150]		✓		
[168]		✓		
[169]		✓	✓	✓
[170]		✓	✓	

TABLE 2.2: Common constraints

As summarized in Table 2.2, constraints that are most frequently used, most evident, and perhaps easiest to consider are the minimization of non-working traveled distance and the obstacle avoidance. However, other significant and more complicated factors also need to be taken into account, such as energy consumption, soil condition and constraints due to the machinery features.

In the rest of this section, the different types of constraints that have been considered in the literature are detailed.

2.4.1 Common Constraints: Traveled Distance and Obstacle Avoidance

The most prevalent constraint that is included in almost all of the research in the literature is the minimization of non-working traveled distance. It is frequently assumed that minimizing only non-working traveled distance will reduce the path length while preserve the working traveled distance and consequently the total coverage rate. Indeed, when minimizing the path length, chances to reduce the operation time, energy consumption and soil compaction are higher. This simplification may, however, result in less accuracy. As summarized in Table 2.2, some researchers, however addressed directly the minimization of the operation time.

The second most prevalent constraint is obstacle avoidance. Obstacles in a farm field might be stationary, such trees, lakes, canals, or transmission towers. As described in Section 2.3, static obstacles were always considered in the terrain modeling phase by simply excluding the obstacle's location from the field model. Obstacles may also be dynamic such as domestic animals, humans and other robots in a multi-robot system. It is obviously challenging in this situation to foresee their position in advance and determine a course of action to avoid them and it requires a robust CCPP approach for replanting the path if it is needed. This case is usually addressed through sensors and decision making system of autonomous robots, or simply by human decision in traditional agriculture.

As described in Section 2.3.1, considering a 2D model for a non-flat fields may lead to skips and/or overlaps between two adjacent tracks. Overlaps, on the other hand, may also occur between trajectories used for covering headlands and the parallel tracks used for covering the main part of the field. This second case occurs if the parallel tracks are not perpendicular to the corresponding headland. Some researcher attended to minimize skips and overlaps but non of them simultaneously took both cases into account.

Besides these common constraints, some more complex factors need to be taken into account. The studies that take into account more complex factors are described in the next sections.

2.4.2 Soil Condition

A significant global issue that frequently leads to insufficient roots and low yield in crops is soil compaction [29]. It significantly reduces the effectiveness of fertilizers and water (from irrigation and rainfall). It raises also the risk of runoff and soil erosion [10]. Traffic is the primary cause of soil compaction in agricultural fields. The main reason of soil compaction in agricultural field is traffic [10]. In the literature, the soil compaction is addressed indirectly by minimizing non-working traveled distance or using multiple light weight robots instead of a heavy tractor.

Compacted soil can also accelerate soil erosion. Lands damaged by soil erosion are vulnerable to the loss of nutrients and organic matter in the topsoil that results in poor agricultural productivity, higher water pollution, and the destruction of wildlife habitats. Some of the primary causes of soil erosion are improper farming methods combined with heavy precipitation, and rough slopes with few vegetation [122].

With the assumption that planning paths perpendicular to slopes not only significantly lowers energy consumption but also reduces soil losses and surface runoffs, only a few researchers took the slopes into account. To directly address the minimization of soil erosion, only Jin and Tang [67] integrated the *Revised Universal Soil Loss Equation (RUSLE)* in their approach. RUSLE is an erosion model designed to predict the longtime average annual soil loss carried by runoff from specific field slopes in specified cropping and management systems as well as from rangelands [121].

2.4.3 Robots and Machinery Constraints

Finding a feasible solution requires taking into account factors other than the implement width, such as the minimum turning radius of the robot, its capacity for carrying agricultural materials and its fuel consumption.

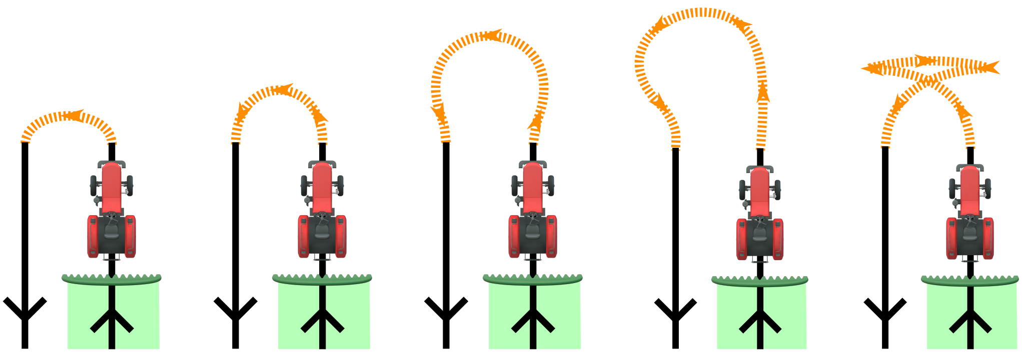
The minimum turning radius of the robot must be considered to generate feasible half-turns. Fig. 2.4 illustrates all potential half-turns that were found in the literature. In general, Dubins curves can be applied for generating half-turns with no reverse moves (Half-turns illustrated on Figs. 2.4a, 2.4b, 2.4c and 2.4d) [35]. For half-turns containing reverse moves (Fig. 2.4e), the method proposed by Reeds and Shepp [120] can be applied. Both methods compute the shortest curve from the starting point to the destination point given a minimum turning radius, the starting and destination coordinates, and the direction of the robot at these coordinates. The Reeds-Shepp method also takes into account reverse moves, whereas all turns generated by the Dubins method only include forward moves. Table 2.3 summarizes the various type of half-turns integrated in the reviewed approaches.

The majority of researchers assumed that the Dubins vehicle description could adequately characterize the robot. Cariou et al. [21, 22] integrated a bicycle model of a car-like vehicle. According to this model, the two front wheels (respectively the two rear wheels) of the vehicle are lumped into a unique wheel located at the center of the front axle (respectively of the rear axle) [113]. Considering the steering rate capacity of the robot, its speed, and a maximum transverse acceleration, Cariou et al. [21, 22] proposed an approach based on Euler spirals to generate feasible half-turns.

For some operations such as seeding sparing and harvesting, the capacity of the robot to transport agricultural materials such as herbicides and seeds must be also considered. In general, for big size fields, considering this factor requires also one or several service units for loading and/or unloading materials even with a multi-robot system. As summarized in Table 2.3, all researcher who took into account the robot's capability also took into account one or more service units. Service units can be either stationary or mobile. In Section 2.5.2, methods that take into account one or more service units are described.

Ref.	EC	Multi-robot	Service unit	Robot capacity	Turning type
[12, 13, 14]		✓	✓	✓	U, Bulb, Fishtail, Flat
[15]					U, Flat
[17]		✓			U, Flat
[19, 20]		✓			U, Flat
[21]					U, Bulb, Fishtail
[22]		✓			U, Bulb, Flat, Hook
[26, 28, 27]	✓	✓	✓	✓	U, Bulb
[31, 33]	✓				In-place rotation
[37]					U, Bulb, Flat, Hook
[40]		✓	✓	✓	U, Bulb, Hook
[47]					
[50]					U, Bulb, Hook
[49]	✓		✓	✓	U, Bulb
[51]					
[48]					U, Flat
[62]		✓	✓	✓	
[64, 63]			✓	✓	U, Bulb, Flat, Hook
[65]					U, Fishtail
[68]					U, Bulb, Fishtail, Flat, Hook
[67]					U, Bulb, Fishtail, Flat, Hook
[71]		✓	✓	✓	Flat
[86]					U, Flat
[87]					U, Bulb, Fishtail, Flat, Hook
[96, 99]		✓	✓	✓	U, Bulb
[100, 101]					U, Bulb
[112]			✓	✓	Flat
[110]					
[111]					Flat
[130]					
[136, 135, 134, 137]		✓			Bulb, Flat, Hook
[139]	✓				U, Bulb, Fishtail, Flat
[147, 148]		✓			U, Bulb, Flat
[151, 149, 150]			✓	✓	U, Bulb, Hook
[168]			✓	✓	U, Bulb, Flat
[169]					
[170]					

TABLE 2.3: Characteristics of the robot and implement. *EC* stands for Energy Consumption



(A) Flat turn (B) U-turn (C) Bulb-turn (D) Hook-turn (E) Fishtail-turn

FIGURE 2.4: Different type of half-turns

2.4.4 Energy Model

A key objective is to reduce energy consumption for variety of reasons, such as to reduce costs for farmers, to protect the environment and resources, and to reduce pollution. To minimize the energy consumption, it is crucial to precisely predict how much energy a robot will need to follow a certain path. This requires a complex model that can account for all variables that affect the energy consumption including the environment and soil condition, the feature of the robots and its implement and the interaction between the implement and soil. For more detail about the different energy models for wheeled mobile robots based on 2D and 3D data of the field, we refer the readers to the survey performed by Zhang, Zhang, and Yang [166].

The main presumption in many works is that a robot has a limitless power supply and/or that there is a small area that one or more fully charged robots can entirely cover. However, recharging often may be required for a large field or when the goal is to utilize some small and compact robots to reduce soil compaction. As summarized in Table 2.3 some researchers took into account one or more service units in their approach, but only for refilling agricultural material or unloading harvested products. Only Conesa-Muñoz et al. [26, 27] considered both the robot capacity and its energy level.

Based on a 3D model of a field, Dogru and Marques [31, 33] provided an analysis energy model that was based on friction and gravity. They reported that driving in the direction of slopes, where there is an excessive variation of elevation, increases the energy consumption. Hameed [49] and Shen et al. [139] proposed an energy consumption model that took into account slopes and the total mass of the robot and its implement.

2.4.5 Discussion

As was previously stated, the CCPP is a multiple objective problem that demands for the best compromise possible between a variety of constraints. It is challenging to model all of these restrictions in a dynamic environment in which an interaction between the robot and soil or crops is needed. A non-flat field might make this task considerably more difficult. None of the papers included in this state of the art explicitly took into account all the listed constraints due to the complexity of the problem. However some constraints are only required for certain type of operations. For instance, since no agricultural materials is required for ploughing, considering the robot's capacity for this operation is irrelevant.

The minimization of non-working traveled distance was the most common factor took into account in cited papers because it is assumed that minimizing non-working traveled distance can indirectly minimize the energy consumption, the soil compaction and total operation time. As summarized in Table 2.3, energy consumption was addressed directly only by four research groups. However non of them took into the account the interaction between the the implement and soil.

Non of the presented studies addressed directly the soil compaction. Besides the minimization of non-working travel distance, replacing heavy tractors with a fleet of little and lightweight robots was also proposed. However, smaller robots have also smaller storage and fuel capacity. Therefore, this solution demands for a high number of service units for loading/unloading and refueling. Consequently, it may increase the non-working travel distance and soil compaction. Furthermore, a smaller robot also has less power than a large tractor, and it might not be suitable for all types of soil and climatic circumstances. Hence, including a soil compaction map, acquired from previous operations on the field, into CCPP might be a proper solution to minimize the soil compaction. We refer readers to [131, 41, 54] for further information about compaction maps.

Soil erosion received only slightly more attention. As described in Section 2.4.2, Only Jin and Tang [67] integrated a soil erosion model to their approach.

Other constraints could also be mentioned but haven't been researched in the literature lately, such as weather circumstances that can affect the state of the soil. For instance, on a sunny day the most optimal solution may be acquired by a path but the same path may not be a good choice on deformable soil due to rain. Another constraint may arise only for operations in which the implement is in contact with the ground while being used. In such operations, raising and lowering the implement to the ground can not be done instantly. It must be done gradually while the robot travels straight forward a few meters. The area covered by the implement in these few meter should not be considered as worked area. In such operation, the robot may also have less degree of freedom when the implement is on and lowered to the ground as contrast to when it is off and raised. Therefore, another constraint is to forbid a tight turn while the implement is in contact with the ground. These kind of turns may damage the implement and/or the robot. Neglecting these constraints would lead to impractical paths or to an overestimation of coverage rate.

Clearly, successfully performing CCPP involves much more than just minimizing the operation time and the total path length while avoiding obstacles. Directly satisfying all constraints necessitates the integration of complex models, which is a challenging process that frequently results in extensive calculation times. In addition, the outcome of an operation depends on earlier operations when taking soil compaction into account. A multi-robot CCPP would make all of these tasks more difficult.

2.5 Path Planning

CCPP is a challenging and complex problem. Consequently, a strategy is to split it into two smaller problems. Therefore, CCPP is always addressed in the literature as two separate tasks. Based on field data, the first task (CMC) is to create a set of parallel trajectories to cover the main part of the field. These parallel trajectories may also be known as parallel tracks or back and forth trajectories. The second task (AVRP) is to assign the parallel tracks to one or multiple robots and connect them by half-turns performed inside headlands.

However for some operations such as spraying and crop monitoring, parallel tracks are predefined based on a tramline farming system which is usually done during sowing or seeding. To prevent soil compaction caused by wheels in the cultivated area, permanent parallel wheel tracks (tramlines) are created in the field [11]. Therefore, to determine the optimal sequence of tramlines for these operations, only an AVRP approach is required.

The core component of CCPP approaches is path planning. They can be categorized using a variety of criteria, whether they are offline or online, multi-robot or single robot, grid-based, graph-based, or based on cellular decomposition. Additionally, they may be categorized based on the optimization method they use, such as greedy algorithms, dynamic programming, or evolutionary algorithms. We refer the reader to [43, 72] for further information regarding CCPP algorithms for robotics in general. The review of Almadhoun et al. [5] on multi-robot path planning is another resource we suggest to the reader.

Table 2.4 gives an outline of inputs of approaches reviewed in this chapter. It also precise which approach addressed CMC and/or AVRP. The remainder of this section describes CCPP approaches in context of two sub-problems, CMC and AVRP.

Ref.	Inputs	CMC	AVRP
[12, 13, 14]	Tracks, Initial location: Robot & Service unit		✓
[15]	Field shape, Mobile unit dimensions, Working width, Minimum turning radius		✓
[17]	Tracks, Initial location of Robots, Minimum turning radius		✓
[19, 20]	Field shape	✓	
[21]	Field shape, Working width, Robot steering rate, its speed and its maximum transverse acceleration	✓	
[22]	Field shape, Working width, Robot steering rate, its speed and its maximum transverse acceleration	✓	✓
[26, 28, 27]	Tracks, Headland paths, Robots capacity, Minimum turning radius		✓
[31, 33]	Field shape, Field's DEM	✓	
[37]	Field shape, Entry points, Working width, Minimum turning radius	✓	✓
[40]	Field shape, Working width, Minimum turning radius		✓
[47]	Field shape, Working width, Minimum turning radius	✓	
[50]	Field shape, Number of headland paths, Working width, Minimum turning radius	✓	✓
[49]	Field shape, Number of headland paths, Field's DEM, Robot speed, Fuel cost, Robot capacity, Working width, Minimum turning radius	✓	✓
[51]	Field shape, Number of headland paths, Field's DEM, Driving direction, Working width, Minimum turning radius	✓	
[48]	Field shape, Number of headland paths, Driving direction, Working width, Minimum turning radius	✓	✓
[62]	Field shape, Tracks and headlands, Road network, Mobile unit speed: on tracks; on headland and on road, Working width, Minimum turning radius	✓	✓

[64, 63]	Field shape, Reference line, Depot position, Robot capacity, Preferred field edge to start work at, Working width, Minimum turning radius	✓	✓
[65]	Field shape, Number of headland paths, Initial overlap length, Headland turning shape, Working width, Minimum turning radius, Working velocity, Turning velocity	✓	✓
[68]	Field shape, Headland width, Working width, Minimum turning radius	✓	
[67]	Field shape, Field's DEM, Headland width, Working width, Minimum turning radius	✓	
[71]	Tracks, Headlands paths, Robot capacity		✓
[86]	Occupancy grid map of the field	✓	
[87]	Field shape, Headland width, Working width, Robot width, Minimum turning radius	✓	✓
[96, 99]	Field shape, Entry points/ Access segments, Headland width, Reference direction, Working width, Minimum turning radius	✓	✓
[100, 101]	Field shape, Number of headland paths, Working width, Minimum turning radius	✓	
[112]	Field shape, Tracks, Headland width, Entry points, WW		✓
[110]	Field shape, Reference line, WW	✓	
[111]	Field shape, Headland paths, Interior lanes, Entry points, WW		✓
[130]	A satellite image of the field	✓	
[136, 135, 134, 137]	Tracks, Headlands paths, Number of robots, Minimum turning radius		✓
[139]	Field shape, Headland width, Field's DEM, Machinery mass, Rolling friction coefficient, Working width, Minimum turning radius	✓	✓
[147, 148]	Tracks, Headlands paths, Number of robots, Robot capacity, Minimum turning radius		✓
[151, 149, 150]	Tracks, Headlands paths, Robot capacity, Location of service units, Minimum turning radius		✓
[168]	Field shape, Number of headland paths, Driving direction, Robot capacity, Working width, Minimum turning radius	✓	✓
[169]	Field shape, Number of tracks in headlands, Reference line, WW	✓	
[170]	Field shape	✓	

TABLE 2.4: Inputs & Path planning

2.5.1 Coverage Map Computing

The majority of CMC algorithms work with fields that are represented as 2D terrains. Only a few studies have attempted to use the field's 3D data to increase the accuracy of CMC algorithms. This section describes in detail 2D and 3D CMC algorithms.

A common CMC method is to generate parallel trajectories based on a reference direction or the field's longest edge [21, 50, 48, 64, 63, 96, 99, 168, 170]. Additionally, constructing parallel trajectories to a curved reference line was also suggested as an improvement to this principle [47, 62, 110, 169]. For each field border, Edwards et al. [37] created parallel straight tracks, and the best candidate with the least tracks was chosen. Mier, Valente, and Bruin [87] applied a brute force algorithm to find the optimal driving direction while trying discretized angles using a given step size.

To estimate way-points of existing tracks of a vineyard, Mazzia et al. [86] used a deep learning model and A^* algorithm. As an input, their method requires an occupancy grid of the field and provides, as an output, a set of tracks. Applying a similar approach, Santos et al. [130] intended to to detect and construct a topological map of existing tracks of a field from satellite images. Their algorithm can detect and construct curve tracks.

To generate parallel straight tracks, Jeon et al. [65] applied an approach based on rotating the minimum bounding box of the field to minimize the number of tracks. Cao et al. [20, 19] presented an approach based on the *rotating calipers algorithm* and probabilistic roadmaps. In their method, the rotating calipers algorithm first generates a reference direction. Once the reference direction has been applied to the whole field, the probabilistic road maps are then utilized to connect the ends of the straight and parallel tracks. To find an optimal reference direction, Cariou et al. [22] applied a brute force algorithm. Their strategy involves rotating the field polygon in steps of 1 degree from 1 to 360 degrees. At each step, the external rectangle parallel to the x and y axes including the field is determined and its area is calculated. Finally the most optimal reference direction was chosen considering the three following criteria: 1) the longest side of the rectangle was along the y-axis, 2) the rotations leading to a discontinuity in the driving direction along the y-axis were not kept, 3) from the remaining rotations, the rectangle which had the minimal area was selected as the final solution. Their algorithm is limited to simple convex field shapes.

Oksanen et al. [100] and Oksanen and Visala [101] proposed a greedy approach for dividing the field into trapezoids and constructing blocks from parallel trapezoids by using specific rules. The optimal driving direction was determined using a heuristic algorithm. The search was repeated until the whole field has been divided and processed. Their approach can find a solution for 2D fields with variety of shapes and any number of obstacles.

Jin and Tang [68] proposed an approach based on undirected graphs to efficiently divide a 2D field into sub-fields and identify a driving direction for each of them while minimizing the number of turns and avoiding turns with high operational cost.

Hameed [49] presented a two-step approach that employs both 2D and 3D models of a field. First, the optimum driving direction that minimizes the number of field tracks was determined applying a genetic algorithm on the field's 2D model. At the second level, 3D field data was taken into account to determine the optimal sequence of tracks to ensure that the robot can explore all tracks with the least number of headland turns and the lowest fuel consumption.

Shen et al. [139] suggested a 3D-based energy model for hilly terrains. After dividing the field to sub-fields, They utilized this model to determine the driving direction that minimizes the energy consumption for each sub-field. The genetic algorithm was then used to find the best order of sub-fields based on the known location of entry point of each of them.

Using 2D data of a field, Hameed, Cour-Harbo, and Osen [51] generated parallel tracks for all directions between 0 and 180. Then they applied and approach to quantify the skips and overlaps. Finally, the driving direction that simply minimizes the skips and/or overlaps was selected.

Dogru and Marques [31, 33] proposed a CMC algorithm that employs both 2D and 3D models of a field. To save the slopes contour in a 2D map, their system first estimated the gradient of the terrain and verified it against a threshold. Afterwards, the resultant map was combined with the 2D obstacle map and the final map was then partitioned. Finally, the generic algorithm was applied to find an optimal driving direction for each partition and the order in which the partitions must be visited.

Jin and Tang [67] presented a decomposition approach to divide a terrain into slope and flat zones, and using field boundaries and slopes contour lines, they identified a reference direction that results in the lowest coverage costs. In their approach, the cost of turns in headlands, soil erosion and the curvature of trajectories were directly addressed. Finally the resultant reference line, that could be curved or straight, was applied to the entire field.

2.5.2 Agricultural Vehicle Routing Problem

To create a fully autonomous system for field operations, the agricultural routing problem must be solved using a field coverage map and the robots kinematic and/or dynamic models. The AVRP algorithm generates a sequence of tracks that can fulfill a number of optimization requirements, including minimizing the operation time, fuel consumed, soil compaction, while avoiding dynamic obstacles. Depending on the characteristics of the robot and the optimization criteria, two successive tracks in the resulting sequence may be adjacent or not.

As described by Plessen [112], An AVRP algorithm should take into consideration: 1) path following according to a field coverage path; 2) navigation from a position along the path network to a nearest service unit for recharging and loading or unloading the storage tanks; 3) navigation from the service unit back to the position along the field coverage path for resumption of work.

AVRP can be resolved for a single robot or a fleet of robots. An overview of single robot and multi-robot approaches is provided in Table 2.3. The remainder of this section provides details on single and multi robot approaches.

Single Robot

Studies focusing on a single robot operations are included in this section. With just one robot, the issue of finite storage space quickly arises. This section describes how the various studies addressed this issue. It is organized based on the number of stationary service units that are available in the field, from none (the robot has limitless storage capacity) to multiple.

Assuming one single robot with limitless storage and/or a small field, Edwards et al. [37] applied a combinatorial optimization algorithm to connect tracks in headland and connect headlands of sub-fields. Bochtis et al. [15] employed the Clarke-Wright method to provide an optimal sequence of tracks. Hameed, Bochtis, and Sorensen [50], Hameed [48], and Shen et al. [139] used evolutionary algorithms to identify the best possible track order. Plessen [111] provide an approach based on Eulerian graph for full and partial field coverage. Their approach can find an optimal sequence of tracks for fields with an arbitrary concave shape and multiple static obstacles. In their approach, the problem was first separated into nine in-field routing scenarios. Then a solution was proposed for each scenario.

Hameed [49] applied an evolutionary algorithm while considering a stationary service unit for refilling the robot tank. Under the same consideration, Jensen, Bochtis, and Sørensen [63] and Jensen et al. [64] presented a method based on the state-space search strategy, where a solution is a series of planned driving actions that minimize the non-working traveled distance. Plessen [112] presented a pattern-based approach to reduce total traveled distance. They analyzed the impact of three different coverage patterns (sequential, circular, and modified circular) for identifying an optimal sequence of tracks. Jeon et al. [65] examined sequential and gathering patterns to connect the parallel tracks in headlands. In the gathering pattern, the distance between two successive tracks is approximately half the field's width. They integrated in their approach a headland and boundary corner turning methods for efficiently covering headland and corners. Mier, Valente, and Bruin [87] also proposed a patter based approach that is able to generate different route patterns such as sequential, row-skip and spiral pattern. The spiral pattern is a variation of the row-skip pattern, that is able to skip several tracks instead of only one track.

Under the same consideration (one stationary service unit), Zhou and Bochtis [168] used the Clarke-Wright savings algorithm and the *Ant Colony Optimization algorithm (ACO)* to find an optimal sequence of tracks. The general principle of ACO is that every ant every ant leaves a trail of pheromones along its path, which over time start to evaporate and lose some of their strength as an attractant. A short path is taken frequently by ants. The maximum pheromone density is therefore found on the shortest path. After representing the set of tracks as a weighted graph, their approach uses a local search in combination with pheromone tracing to optimize the paths.

Nilsson and Zhou [96] presented a strategy built on the *Artificial Bee Colony algorithm (ABC)* while considering one stationary service unit. ABC imitates bee activity in a manner similar to how ACO does with ants. In order to make the solutions better, a local search is employed. However, ABC generates a random solution if the local search does not result in a better solution. That increases the probability of discovering the global optimum. Nørremark, Nilsson, and Sørensen [99] expanded upon this methodology for grain harvest operations by incorporating a mobile service unit for on-the-go unloading in both the headland and main field, establishing unloading timings irrespective of the harvester's full bin level, and considering the transport unit's operational time outside the field. Their main objective was to minimize time and distance costs for all vehicles engaged in harvest operations.

Considering one or several stationary service units, Vahdanjoo, Zhou, and Sørensen [151], Vahdanjoo, Madsen, and Sørensen [149], and Vahdanjoo and Sorensen [150] used the simulated annealing algorithm to identify a sequence of tracks that minimizes the non-working travel distance. The authors experimented various combinations of multiple stationary service units to determine how the number an placement of service units can affect the non-working travel distance. They claimed that relocating the service units might save the non-working travel distance by up to 40%.

They also reported that increasing the number of service units could lead to a reduction of the non-working traveled distance up to 50.9%.

Multi-Robot

The works that took into account several robots operating at once are included in this section. This section follows the same structure as the previous section with only one difference that it ends with brief description of approaches that include one or several mobile service units.

Some researchers assumed an infinite storage capacity for the robots or a small field that several full-charge robots can cover completely. Solved a mixed-integer linear programming problem, Burger, Huiskamp, and Keviczky [17] intended to minimize total traveled distance such that the resultant path is distributed evenly among all robots. With the intention of minimizing total operation time, Seyyedhasani and Dvorak [136, 134, 135] and Seyyedhasani, Dvorak, and Roemmele [137] presented an approach for path assignment among robots using both the Clarke-Wright algorithm and a meta-heuristic algorithm (Tabu Search). Cariou et al. [22] introduced an approach for identifying preassigned parallel tracks while taking into account a convoy of homogeneous robots. In order to do this, a virtual robot whose steering and speed limitations aggregate those of all robots was first created. Then, to join a group of parallel tracks, suitable continuous curvature turns based on adaptive clothoids were constructed.

Some other researcher developed approaches that comprised a single stationary service unit for loading or unloading. Conesa-Muñoz et al. [26], Conesa-Muñoz, Pajares, and Ribeiro [28], and Conesa-Muñoz et al. [27] introduced and integrated a novel operator into the *Simulated Annealing (SA)* to create an expert system for route planning in agricultural fields. Solution generation and replacement are repeated by the SA until an acceptable solution is found or until another computational requirement, such as exceeding a predetermined time limit or number of iterations, is met. To identify the ideal sequence of tracks, Utamima, Reiners, and Ansaripoor [147, 148] applied an evolutionary algorithm and a neighborhood search. Khajepour, Sheikhmohammady, and Nikbakhsh [71] applied a route-first, clusters-second heuristic method to find an initial solution. Afterwards, the adaptive large neighborhood search was applied to find a better solution that determines the best sequence of tracks for multiple robots.

Taking into account a mobile service unit and a primary robot, Evans IV et al. [40] employed an evolutionary algorithm to identify a sequence of tracks that minimizes the non-working distances. Their approach is able to find a solution for convex fields with no obstacles. Jensen et al. [62] utilized the Dijkstra algorithm to generate optimal in-field and inter-field paths to be followed by a primary robot cooperating with a mobile service unit. They also examined on-the-go unloading. Bochtis and Sørensen [13], Bochtis, Sørensen, and Vougioukas [12], and Bochtis and Sørensen [14] introduced a breadth-first search algorithm, modified by additional heuristics, in order to minimize the non-working traveled distance. They examined different combinations of one or multiple primary robots, mobile service units and stationary service units.

Discussion

Path planning is the core problem of CCPP. The reliability and effectiveness of CCPP on a field depends not only on the path planning algorithm but also depends on the considered constraints and the model of the terrain. The complexity of the problem necessitates that CCPP be addressed at two levels of optimization, CMC and AVRP.

The primary goal of CMC algorithms, which demand for a reference direction or trajectory, was generally to generate parallel tracks. A simple strategy was to either utilize the longest field boundary or take the reference direction as an input. To improve the results, algorithms were then introduced to select the optimal reference trajectory between field boundaries. In some cases slopes contours were also considered. Decomposing a concave field shape into convex and simple shapes, then performing parallel track generation for each sub-field, was another improvement. Some researchers also proposed advanced optimization strategies including genetic algorithms, graph-based optimization, brute-force, and heuristic methods. Only two research groups have employed a deep learning technique to estimate way-points of existing tracks. It may be seen as a relatively small number of deep learning based approaches in this domain regarding the success and widespread use of deep learning in most scientific domains during the past several years.

After calculating a coverage map or using it as an input, the main goal of AVRP was to assign a set of tracks to one or more robots while taking into account one or more constraints. Using predetermined patterns such as sequential or circular patterns was a fundamental strategy. One of the most often used algorithms to perform AVRP was the genetic algorithm. Other nature-based algorithms, including ant colony and artificial bee colony algorithms, were also used in this domain. In some cases a hybrid AVRP was carried out by combining the ant colony with the Clarck-Wright saving algorithm. Some other hybrid algorithms were also experimented such as a modified form of Clarck-Wright saving algorithm combined with a tabu search or an evolutionary hybrid neighborhood search.

Divide and conquer may be an efficient strategy for resolving complex problems with numerous constraints. The division of CCPP into CMC and AVRP was frequently applied to find a solution in a reasonable amount of time. However, using this method of division could result in disregarding certain solutions that might be more beneficial than the ones that have been generated. Therefore, a solution might be to remodel it as one-step CCPP to find a wide range of potential solutions that satisfies some hard constraints. Afterwards, depending of the requirement of the operation, a set of soft constraints can be defined to select the best solutions. The efficiency to determine the optimal coverage map and assign each robot the best sequence of trajectories may also be significantly improved by deep learning, which hasn't been widely experimented in this domain yet.

2.6 Validation

Validation is a crucial step in any type of study to determine the efficacy and accuracy of a suggested approach. Validation could be challenging for an agricultural CCPP approach since grand truth is not directly available.

In any kind of research, validation is a very important aspect to assess the accuracy and efficiency of a proposed method. In agricultural path planning, validation is particularly tricky as ground truth is not directly available. To compare a novel approach to a reference, there are three choices:

- comparison with a reference provided by an expert
- comparison with previous/other methods
- comparison with an exhaustive search

The first strategy involves comparing a resultant path with one suggested by an expert. The reference path can be drawn by an expert or can be deduced from from aerial images or GPS data of a tractor used for a field operation. This strategy has the benefit of guaranteeing viability and feasibility since the expert (often the farmer) has expertise with the specific field and is aware of the best ways to operate it. The subjectivity of this reference is a limitation since there is no certainty that the path identified by the expert as a reference is the absolute optimal path.

Comparing the new method against existing methods in the literature is the second strategy. There are potentially two challenges in this case. Reproducing the algorithms of the preceding method and the experiment's settings with reliability is one of these challenges. When precise implementation of existing methods is feasible, the same metrics and experimental data may be utilized to evaluate the approaches in a rigorous and appropriate manner. Otherwise, it is crucial to conduct the new experiment under identical conditions and with the exact same input data as those described in the literature, which might be challenging. Proposing and implementing two or more approaches in order to conduct a rigorous comparison is another solution which is also challenging.

The third strategy involves carefully calculating all feasible paths and choosing the best one given the specified constraints. This strategy ensures that, given a certain combination of parameters and constraints, the absolute optimal path is discovered and used as a reference. However, this reference depends on the selected parameters and criteria, and there is no guarantee that different settings might not have produced a more advantageous path. This issue may be partially resolved by looking through several different parameter values.

Regardless of the selected validation strategy, the fields (input data) on which the validation is conducted might either be actual fields or synthetic fields. Aerial images or data, collected by a GIS software, can be used to designate actual fields. Despite the fact that synthetic fields may not accurately reflect reality, they offer the benefit of being customizable.

Table 2.6 summarizes the validation settings for each work. The validation strategy is represented in Column *Validation* of this table, where the word *EXPERT* denotes validation by an expert, and the word *COMP* stands for comparison with other approaches or evaluation of two or more novel approaches. In certain works, experiments have been conducted by changing several parameters for a single approach, such as the number of robots or service units, or considering different case studies, and comparing the outcomes. They are distinguished by the word *PARAM*. When the findings had been published without a comparison, the validation was denoted with a *NON*. The type of model used to represent the field in the method is shown in Column *Field type*, with *SYNTH* denoting a synthetic field model. *REAL* denoting models created using actual field coordinates, and *BOTH* denoting the case where the authors performed the validation using both field types. It is essential to note that not all of the identified papers had the same objectives, considerations, or even evaluations.

Ref.	Field area (ha)	Field type	Validation	Computation time	Resources
[12, 13, 14]		BOTH	PARAM		
[15]	10	BOTH	COMP		
[17]		SYNTH	NON		
[19, 20]		BOTH	NON		
[21]		SYNTH	PARAM		
[22]		REAL	COMP		
[26, 28, 27]		SYNTH	COMP		Core i5 3.3 GHz CPU 4 GB RAM
[31, 33]	0.1 - 0.6	SYNTH	COMP		
[37]	1.2 - 12.4	REAL	EXPERT		
[40]	11.5 - 27.2	REAL	EXPERT		
[47]	0.2 - 44.9	REAL	PARAM	0.1 - 402.6 s	3.2 GHz CPU 1 GB RAM
[50]	7.9 - 17.2	REAL	PARAM	18.8 - 23.7 min	2.4 GHz CPU 2 GB RAM
[49]	11.2 - 21.2	REAL	COMP	66 - 380.1 min	
[51]	21.2	BOTH	COMP PARAM		
[48]	21.2	REAL	PARAM		
[62]	6.7 - 7.6	REAL	PARAM	0.1 s	2.5 GHz CPU 3 GB RAM
[64, 63]		REAL	EXPERT	7 min	
[65]		REAL	PARAM		
[68]		BOTH	EXPERT COMP	60 s	3.2 GHz $\times 4$ 1.5 GB RAM
[67]	24 - 48	REAL	COMP		
[71]		SYNTH	EXPERT PARAM		Core i5 CPU 4 GB RAM
[86]		BOTH	EXPERT		Core i5 CPU 4 GB RAM
[87]	0.01 - 1	BOTH	EXPERT COMP	0.5 - 3.5 s	Core i7 2.8 GHz 4 cores, 8 threads
[96, 99]	5 - 26.5	BOTH	EXPERT PARAM	0.6 - 400.7 s	Core i7 2.8 GHz 16 GB RAM
[100, 101]	3.9 (average)	REAL	COMP		
[112]		BOTH	COMP		
[110]	10.3 - 62.9	REAL	COMP		
[111]	13.5 - 74.3	REAL	COMP	0.1 ms - 14 s	Core i7 4.2 GHz $\times 8$ 15.6 GB RAM
[130]	2.3 - 5.2	REAL	COMP		Core i7 2.2 GHz $\times 12$ 16 GB RAM
[136, 135, 134, 137]	2.5 - 25.6	BOTH	EXPERT COMP		
[139]	0.2 - 2	REAL	PARAM		Core i5-1035 PC
[147, 148]		BOTH	COMP		
[151, 149, 150]	0.1 - 16	BOTH	COMP	93 - 139 s	Core i5 2.5 GHz 4GB RAM
[168]	3.3 - 4.1	REAL	EXPERT		
[169]	10.3 - 75.0	REAL	PARAM	0.3 – 24.5 s	3.2 GHz CPU 4 GB RAM
[170]		SYNTH	COMP		

TABLE 2.5: Validation settings

Discussion

As summarized in Table 2.5 none of the works have been validated against an exhaustive search, although it is a frequent strategy for validating optimization methods. The main reasons behind it could be the complexity of an exhaustive approach and the limitation of resources. Additionally, we can see that some researchers have published their findings without making any comparisons. Since the optimization criteria differ depending on the operation type and the preferences of individual farmers, comparing CCPP approaches can be challenging in practice.

We can notice that almost 83% of the studies were validated on models generated from actual fields coordinates while almost 50% validated on synthetic data only. A validation on synthetic cases provides a better control over the types of shapes and their variety, and allows to experiment on particular cases, it is also important to validate on actual data. Validating on actual data often provides information about the solution chosen by the farmer, that may be considered as a ground truth or expert solution to compare the results.

2.7 Conclusion

Path planning for wheeled agricultural robots to cover a field autonomously is a complex task that requires consideration of various factors and constraints. In this chapter different challenges of CCPP and proposed approaches to address this problem were analyzed through a systematic review. 49 articles were included in this review, which have been categorized and analyzed from different perspectives: terrain modeling, constraints modeling and path planning.

Despite all of difficulties and complexity of CCPP, several researchers have provided approaches to solve the problem. The suggested solutions often use divide and conquer strategy and/or simplify the hypotheses of the problem to generate interesting paths through efficient algorithms.

Depending on the operation and required machinery to perform a CCPP, several factor and constraints must be taken into account. Estimating the efficiency of a path simply through the total time required for a complete coverage and/or through the path length may be an oversimplification for some operations such as tillage like operations. Using 2D data of non-flat field to compute skips and overlaps may not be appropriate. In presence of inclinations, the minimization of the total path length may not be sufficient for minimizing the energy consumption. Obstacle avoidance can not be limited only to static and permanent obstacles. It is also important to take into account temporary obstacles such as natural basins due to the rain and dynamic obstacles such as animals.

To create an effective path for one or more robots in such a complex and dynamic environment necessitates accurate models of robots, field, soil erosion, and compaction in different climates. However, no study included in this review covers all the factors and constraints of CCPP. Therefore, further efforts are required to account for all constraints through realistic and accurate models. Such models and the interaction between them could be extremely complex and demand for massive processing resources. Accelerating computations using GPUs and parallel programming may be a possible solution to achieve this goal.

Although this survey does not discuss the problem of real-time robot navigation during an operation, the precision of navigation and the ability to follow the planned path are also essential factors that can significantly affect the operation's outcome. These topics remain outside the scope of this thesis.

In conclusion, this chapter provides a comprehensive overview of the state of the art in CCPP for autonomous wheeled agricultural robots, highlighting the challenges and proposed solutions in the literature. However, further work is needed to develop more realistic and accurate models that can account for all the constraints of CCPP.

CHAPTER 3

Terrain modeling: Data Acquisition and Dataset construction

Terrain modeling is the process of creating a digital representation of the physical land surface. This representation can be either in 2D or 3D format and serves as a basis for various activities such as path planning, soil erosion analysis, and energy consumption estimation. The quality of the model may have a direct impact on the accuracy of the results of these applications.

2D terrain modeling involves creating a two-dimensional representation of the terrain surface, which is usually a flat projection of the actual terrain onto a plane. On the other hand, 3D terrain modeling involves creating a three-dimensional representation of the terrain surface, which takes into account the actual topography and elevation variations. This representation is usually created by interpolating elevation data to create a continuous surface that accurately represents the terrain in 3D.

In general, 3D terrain modeling provides a more accurate representation of the terrain surface compared to 2D modeling, as it takes into account the actual elevation variations and topography. Despite these advantages, the use of 3D models in agricultural terrain modeling is still limited. This is primarily due to the computational complexity associated with creating and processing 3D models.

In this chapter, we will provide a comprehensive overview of the data required for generating both 2D and 3D models of a field. This will include a discussion of some approaches for creating these models. Additionally, a dataset of thirty fields located in France will be provided, which can be used to evaluate the performance of different applications such as path planing, soil erosion analysis, and energy consumption estimation approaches. This will also help to provide a better understanding of the advantages and limitations of both 2D and 3D terrain modeling, and will provide a basis for further research and development in this area.

The remainder of this chapter is organized as follows: the various data required and their acquisition methods are discussed in detail in Section 3.1. Approaches for creating 2D and 3D terrain models of a field are presented in Section 3.2. The generated dataset, including thirty fields located in France, is described in Section 3.3. The chapter is concluded in Section 3.4.

3.1 Data Acquisition

Accurate data acquisition is a critical aspect in constructing a realistic field model. It is the foundation for generating a reliable representation of the terrain surface. A field model that is based on inaccurate or inconsistent data can lead to incorrect results in various applications. Furthermore, a field model that lacks precision can negatively impact the effectiveness of decision-making processes that are based on the model.

In this section, the necessary data for agricultural path planning and methods of acquiring it are discussed in detail. The required data to construct both 3D and 2D model of a field for this application include:

- a set of counterclockwise points representing the field border
- one or several pair of points representing *access segments*
- elevation data of the field, if available
- optionally some sets of clockwise points representing static obstacles

where access segments of the field are 2D line segments (*i.e.*, pair of points) overlaid on the field boundaries that specify the locations from where the robot can enter and exit the field.

FIGURE 3.1: The Géoportail [45] annotation tool. a field polygon, an obstacle inside it and one access segment are annotated in red, blue and green

The access segments, the field and the obstacles borders are acquired using Géoportail [45] annotation tool. This annotation tool provides a GUI allowing to draw geometrical forms such as points lines and polygons on satellite images and export them as a *Keyhole Markup Language (KML)*. KML is an *XML* notation for expressing geographic annotations developed by Google. Therefore, a single KML file can contains coordinate of the field and obstacles polygons as well as the coordinate of the access segments. Fig. 3.1 illustrates the annotation tool where an aerial image of the field is displayed. In this figure, a field polygon, an obstacle inside it and one access segment are also annotated in red, blue and green.

The elevation data of a field are acquired using IGN elevation calculation services [58]. It is a *Representational State Transfer API (REST API)* that takes as input a point's coordinate and returns its elevation. IGN also provides DEM of the French territory with three different resolutions: 1, 5 and 25 meters. However, downloading these data require a huge amount of disk space.

Therefore, to acquire the elevation data of a field, a grid of points with a spacing of ℓ_e, named *elevation grid*, is first generated inside the field bounding box. Afterwards, the REST API is employed to determine the elevation of each point of the elevation grid. However, depending on the number of points, this method could be extremely slow. For instance, setting ℓ_e to $0.5m$, it takes over 13 hours to determine the elevation of 136859 points for a field of $3.34ha$. For a high resolution elevation grid, this API might also produce some noises. This case is illustrated on Fig. 3.2, where only points inside the field polygon are kept and for a better visualization, it is represented as a surface.

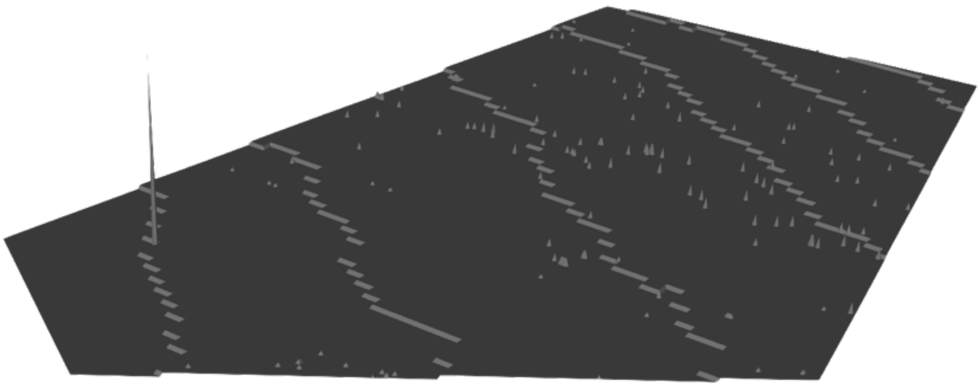


FIGURE 3.2: High resolution elevation data acquired using the REST API of IGN [58]

To address the challenges posed by acquiring high resolution elevation data using the REST API, a practical solution is to create a lower resolution elevation grid with a point spacing of $\ell_e = 5m$. The altitude of the points in this grid is then determined using the REST API. In addition, for each point in the elevation grid, a normalization is applied to set the minimum altitude to zero, while preserving the original elevation variation. Therefore, for all points $P_i(x_i, y_{,i}, z_i)$ in the elevation grid the following normalization is applied:

$$z_i = z_i - z_{min} \tag{3.1}$$

where z_{min} denote the minimum altitude of the elevation grid.

The resulting low-resolution elevation grid can then be used to construct a high-resolution 3D surface of the field through elevation interpolation methods. Section 3.2.2 examines three well-known interpolation approaches for the construction of a 3D surface of the field.

3.2 Terrain Modeling Approaches

Taking as input the required data and converting them from the geographic coordinate system to the Cartesian coordinate system, this section provides a step-by-step instruction for generating a high resolution 2D surface of a field. Afterwards, a 3D surface of the field is also generated using the 2D surface of the field, its elevation grid and a well-known interpolation method.

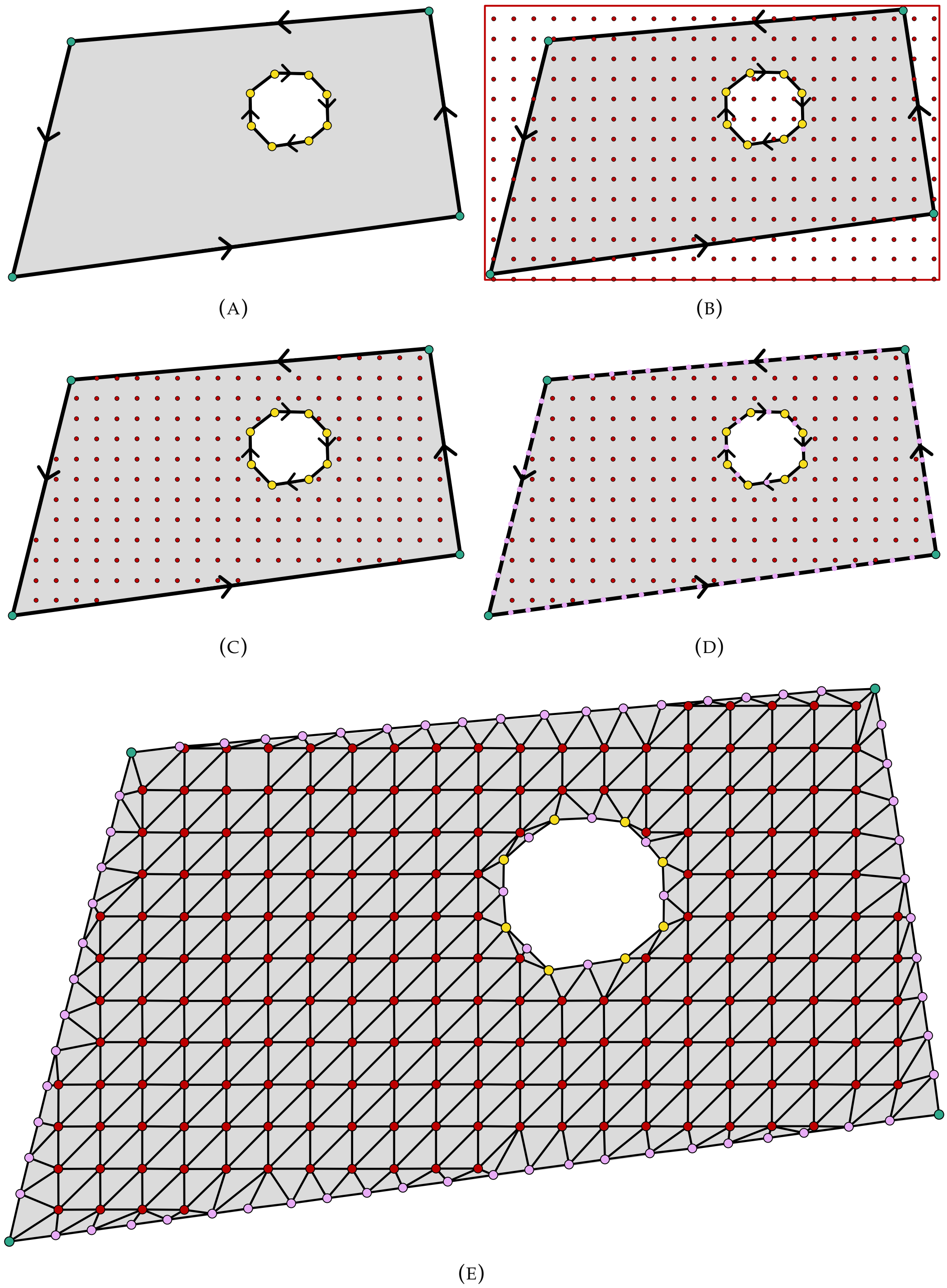


FIGURE 3.3: Different steps of 2D field construction for a convex field with one static obstacle

The implementation of the process of generating a high resolution 2D and 3D surface of a field was carried out using the *Visualization ToolKit (VTK)* library [132]. VTK is an open-source, freely available software system for 3D computer graphics, modeling, image processing, volume rendering, scientific visualization, and 2D plotting. It provides advanced modeling techniques including implicit modeling, polygon reduction, mesh smoothing, cutting, contouring, Delaunay triangulation and various interpolation methods.

3.2.1 2D Surface Construction

The generation of a 2D surface of a field involves the following steps:

- constructing a 2D polygon combining the field and obstacles polygons
- generating a point grid, named *2D grid*, inside the field bounding box
- eliminating the points of the 2D grid that are not inside the 2D polygon
- interpolating new points on the field and obstacle borders
- combining the 2D grid and the interpolated points and triangulating them

The result of this process is a homogeneous triangulated surface, with the exception of possibly being non-homogeneous near the borders of the field and its obstacles. A visual representation of this process on a small synthetic field is illustrated on Fig. 3.3.

As illustrated on Fig. 3.3a, for a field with only one obstacle, a 2D polygon of the field is first generated excluding the obstacle (yellow points) from the field polygon (green points). Afterwards, a 2D grid with a spacing of ℓ_g generated inside the field bounding box (red points on Fig. 3.3b). Then, only points contained within the 2D polygon are preserved. This step is illustrated in Fig. 3.3c. Following that, a series of counterclockwise points and a series of clockwise points with a maximum spacing of ℓ_g are interpolated respectively on the field and obstacle borders (pink points on Fig. 3.3d). Finally, as illustrated in Fig. 3.3e, all remaining points are triangulated to generate a 2D surface of the field.

3.2.2 3D Surface Construction

After constructing a high resolution 2D surface of the field, the elevation grid and an interpolation approach can be used to estimate the altitude of every single point of the 2D surface. The result of this process is a high resolution 3D surface. To generate the 3D surface three well-known interpolation approaches are examined.

The first approach is *Bilinear Interpolation (BI)* [74]. In computer vision and image processing, bilinear interpolation, also known as bilinear filtering or bilinear texture mapping, is one of the fundamental interpolation methods. Assuming that the elevation of points P_{11}, P_{12}, P_{21} and P_{22}, illustrated on Fig.3.4, is already known. The goal is to calculate the elevation at point P. It can be done as follows:

$$f(P) = a_{00} + a_{10}x + a_{01}y + a_{11}xy \tag{3.2}$$

where $f(P)$ is the elevation at P. The coefficients a_{00}, a_{10}, a_{01} and a_{11} are found by solving the following linear system :

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_1 & y_2 & x_1y_2 \\ 1 & x_2 & y_1 & x_2y_1 \\ 1 & x_2 & y_2 & x_2y_2 \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{11} \end{bmatrix} = \begin{bmatrix} f(P_{11}) \\ f(P_{12}) \\ f(P_{21}) \\ f(P_{22}) \end{bmatrix} \tag{3.3}$$

where $f(P_{11})$, $f(P_{12})$, $f(P_{21})$ and $f(P_{22})$ are respectively the elevation at points P_{11}, P_{12}, P_{21} and P_{22}. Therefore, this linear system can be solved as follows :

$$\begin{bmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{11} \end{bmatrix} = \frac{1}{(x_2 - x_1)(y_2 - y_1)} \begin{bmatrix} x_2y_2 & -x_2y_1 & -x_1y_2 & x_1y_1 \\ -y_2 & y_1 & y_2 & -y_1 \\ -x_2 & x_2 & x_1 & -x_1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} f(P_{11}) \\ f(P_{12}) \\ f(P_{21}) \\ f(P_{22}) \end{bmatrix} \tag{3.4}$$

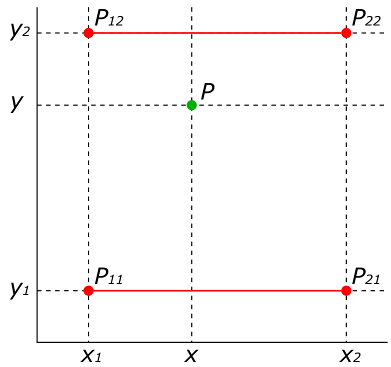


FIGURE 3.4: Bilinear interpolation

The second interpolation approach is known as *Inverse Distance Weighting (IDW)* (first proposed by Shepard [140]). In this approach, the elevation of a point is calculated with a weighted average of surrounding points that their elevation is already known. The weights are computed as inverse of the distance to each neighborhood point. Therefore, the elevation of point P is calculated according to its N nearest neighbor points $\{P_i | P_i \in \mathbb{R}^2, i \in \mathbb{N}, 1 \leq i \leq N\}$ as follows :

$$f(P) = \begin{cases} \frac{\sum_{i=1}^{N} w_i(P_i) f(P_i)}{\sum_{i=1}^{N} w_i(P_i)}, & \text{if } d(P, P_i) \neq 0 \text{ for all } i \\ f(P_i), & \text{if } d(P, P_i) = 0 \text{ for some } i \end{cases} \tag{3.5}$$

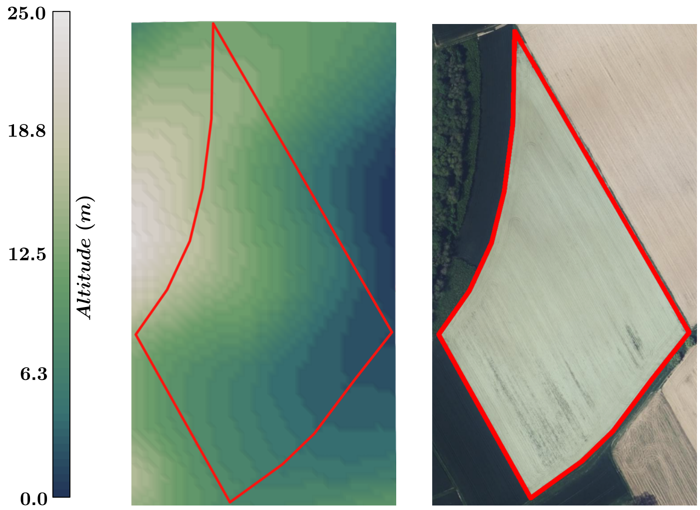
where

$$w(P_i) = \frac{1}{d(P, P_i)^p} \tag{3.6}$$

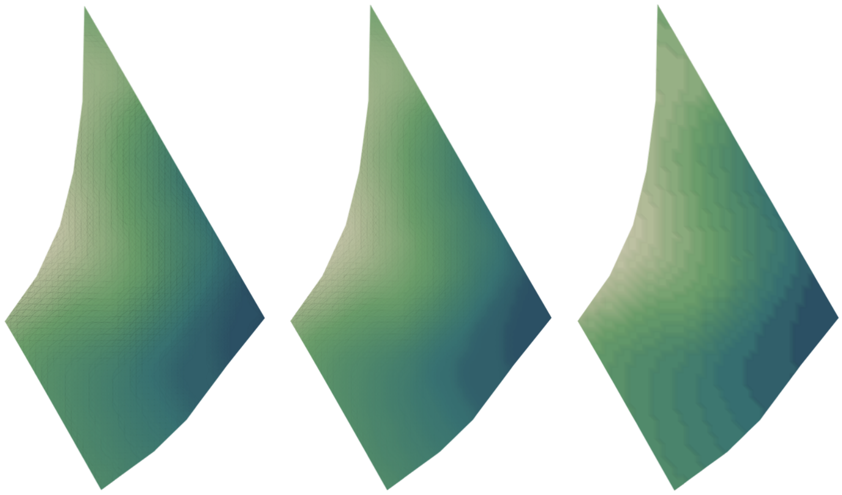
where d denotes the euclidean distance and p is a positive real number, called the power parameter.

The third interpolation approach, which is based on ray casting first proposed by Roth [125], is referred to as *Ray Casting Interpolation (RCI)*. To perform RCI, the elevation grid is first triangulated to obtain an elevation surface. After that, for each point P on the field surface a vertical line is constructed at P. The elevation of P is then determined finding the intersection of the line and the elevation surface.

After using one of the interpolation methods to create a 3D surface of a field, the same normalization from Equation (3.1), which was applied to the elevation grid, is also applied to all $P_i(x_i, y_i, z_i)$ points of the resulting 3D surface. This normalization helps to standardize the elevation data, making it easier to visualize the final result.



(A) Elevation grid (B) Satellite image



(C) BI (D) IDW (E) RCI

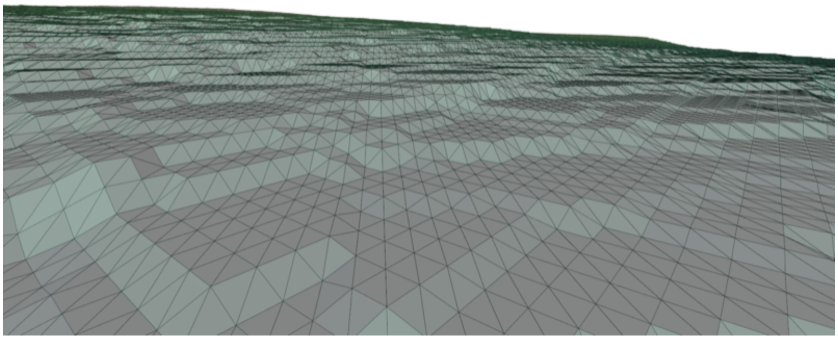
FIGURE 3.5: The visual result of three different interpolation methods for 3D surface construction for a field of 4.22ha

Fig. 3.5 illustrates the result of the interpolation methods that have been discussed on a real field of 4.22ha computed based on 2D data. For better visibility, the elevation grid in this figure is triangulated and shown as a 3D surface. Fig. 3.6 provides a close-up of the results. To obtain these results, ℓ_e and ℓ_g were set to 5 and 0.25 meters respectively. The number of nearest points for BI and IDW were set to $N = 20$ points and the power parameter of IDW was set to $p = 2$. Table 3.1 summarizes the numerical result for each method. Note that each approach used the same elevation grid and 2D surface that were generated only once.

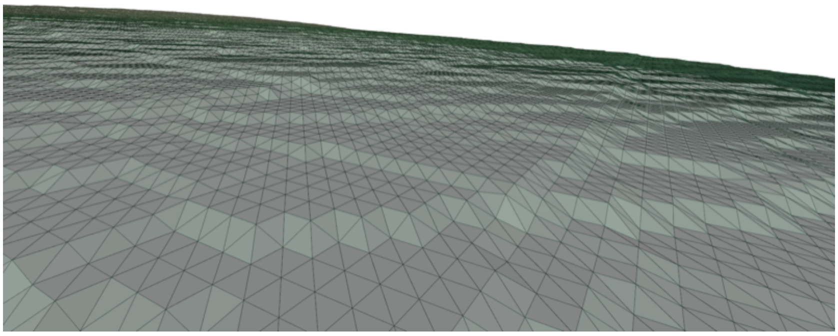
Table 3.1 presents the area results obtained from the 3D surface generated by each interpolation method, as compared to the surface computed based on the 2D data. The results show that the area computed based on the 3D surface is 2.28%, 1.16%

and 1.03% higher than the surface computed based on the 2D data using BI, IDW, and RCI, respectively.

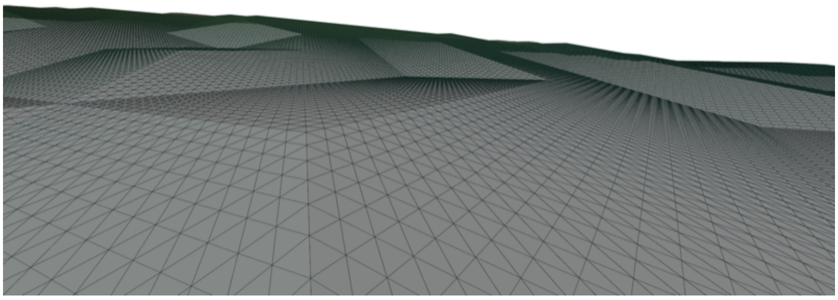
Due to the absence of a ground truth, it is difficult to determine the accuracy of these methods and compare them. It may require to use a drone equipped with airborne LiDAR technology to construct an accurate grand truth dataset to study the efficiency of different 3D reconstruction method for an agricultural field. However a visual comparison of the outcomes reveals that IDW creates a smoother surface that appears to be more realistic.



(A) BI



(B) IDW



(C) RCI

FIGURE 3.6: A close-up of a 3D surface generated using three different interpolation methods

Method	Generation time (s)			3D surface			
	Elevation grid	2D surface	3D surface	Points	Triangles	Variation in elevation (m)	area (m^2)
BI	1855.02	19.03	0.95	676023	1348210	16.0	43110.0
IDW	1855.02	19.03	1.05	676023	1348210	15.7	42639.1
RCI	1855.02	19.03	0.63	676023	1348210	16.2	42584.4

TABLE 3.1: Numerical results of three different interpolation method for a field of $42150.5m^2$ (computed based on 2D data)

3.3 A Dataset of Real Fields

To the best of our knowledge, there is no existing dataset in literature that contains all the necessary information to evaluate and validate path planning approaches on both 2D and 3D surfaces of agricultural fields. To address the lack of such a dataset, we created a dataset of thirty agricultural fields located in France. These fields were selected manually with the aim of obtaining a diverse range of shapes and sizes (from 1.83 to 13.21 hectares), including simple shapes where no field decomposition is necessary and more complex shapes requiring field decomposition. This ensures that the dataset represents a broad range of real-world scenarios for evaluating and validating path planning approaches. The dataset, containing all the necessary information for each of the thirty agricultural fields, is publicly available on Zenodo [115].

(A) France

(B) Poland

(C) Brazil

(D) USA - Texas

FIGURE 3.7: Shape of fields in different continents/countries

The shape of agricultural fields is a product of a complex interplay of factors, including historical, geographic, and topographic factors, as well as cultural and economic practices. For instance, fields in countries with a more recent history of land ownership and partitioning may be more likely to have a simple, square or rectangular shapes, while fields in countries with a more complex history of ownership and partitioning may have more complex shapes. Similarly, geography and topography can also play a role in the shape of fields, with fields in flat, open areas likely having simpler shapes than fields in mountainous, hilly regions and close to forest. For instance, agricultural fields in the US tend to have a rectangular and simple shape, whereas fields in Western European countries tend to have a more irregular and complex shape.

Fig.3.7 illustrates the shape of agricultural fields in different continents and countries, including France, Poland, the US (Texas), and Brazil, with almost the same scale. The scale of the image provides a comparative representation of the size and shape of fields in each location.

The size of agricultural holdings varies significantly across countries and regions. In France, the average size of agricultural holdings was almost 39 hectares per holding in 2016, according to Eurostat [39]. In Poland, the average size of agricultural holdings was smaller, at almost 9 hectares per holding in 2016 [39]. In contrast, farms in the US are much larger than those in France or Poland, with an average size of 180 hectares per farm in 2021 [2]. The large scale of agriculture in the US contributes to the rectangular and uniform shape of fields, which are optimized for modern farming practices and machinery. In Brazil, the size of agricultural holdings can vary widely depending on the region and type of farming, ranging from small family farms to large industrial agribusinesses. This can result in a diverse range of field shapes and sizes across the country.

The reason for choosing fields located in France to create a dataset is due to the variety of field shapes in the country, which is partly due to its history and geography. Additionally, the French government provides high-precision elevation data of the French territory publicly and for free, making it an ideal location to obtain the necessary information to create the dataset.

For each field, this dataset gathers the following information into separate files:

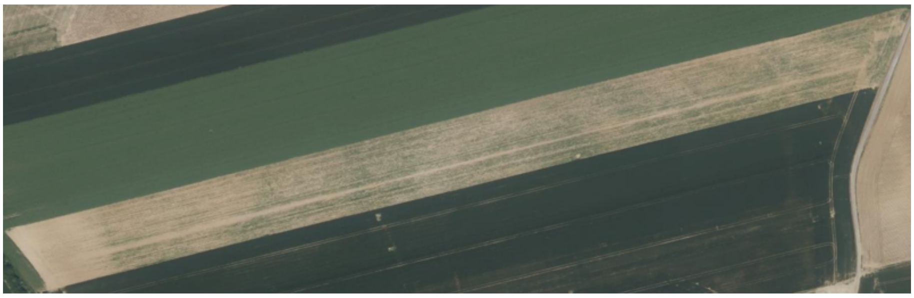
- an aerial image (PNG)
- a 2D polygon (XML)
- a 2D triangulated surface (PLY), where $\ell_g = 0.25m$
- an elevation grid (PLY), where $\ell_e = 5m$
- a 3D triangulated surface (PLY), with IDW parameters $N = 20$ and $p = 2$
- a set of 2D line segments representing the access segments (XML)

where the *Polygonal File Format (PLY)* is a file format used to store 3D computer graphics data, including 3D models and 3D scans [146]. It is ASCII or binary file format that consists of a header and a body. The header contains information about the format, such as the number of vertices, faces, and the properties of each vertex and face, such as color, texture, and normal information. The body of the file contains the actual data of the 3D object. Both the 2D and 3D surfaces of a field were represented as a 3D model, but the only difference between them is that in the 2D surface, all points have a z-value of zero.

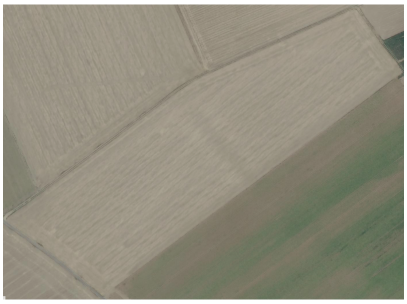
Figs. 3.8, 3.9 and 3.10 provide an aerial image of each field. The shape of these fields are illustrated in Figs. 3.11, 3.12 and 3.13 where green line segments represents access segments. The 3D surface of each field is illustrated in Figs. 3.14 and 3.15. Table 3.3 gives an access to annotated data on Géoportail [45] platform. For a field, it provides also a point inside the field. For each field, the calculated area using both 2D and 3D surface of the field is provided in Table 3.2. As summarized in this table, for all cases, the area of the 2D surface was less than the area of the 3D surface. This is normal, because none of these fields are completely flat. Therefore, measuring the area based on the 2D surface always leads to an underestimation.

(A) field #1

(B) field #2

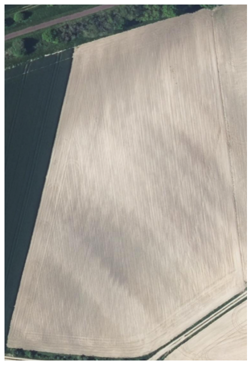


(C) field #3



(D) field #4

(E) field #5

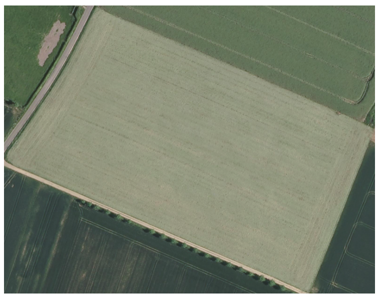


(F) field #6

(G) field #7

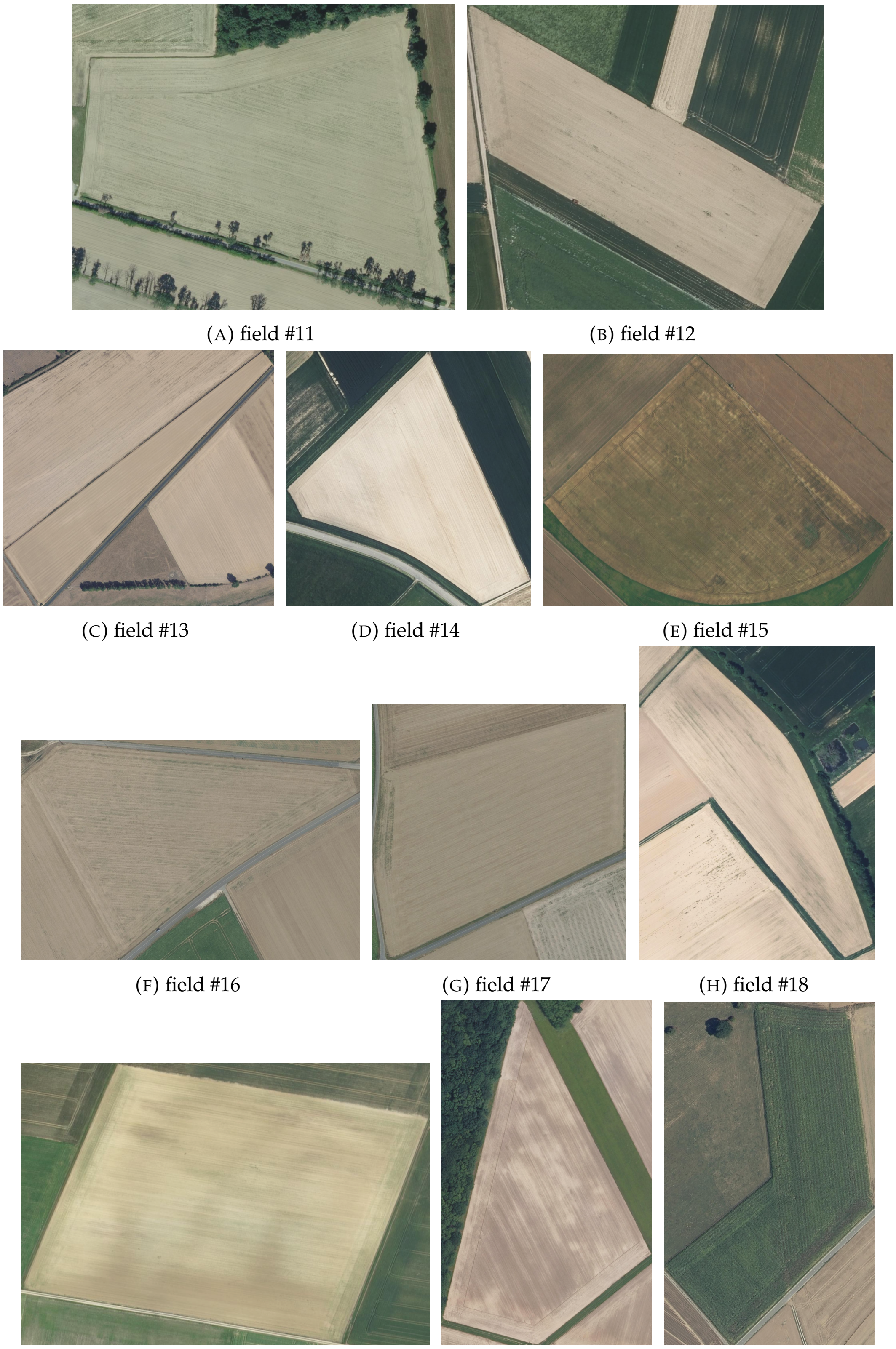
(H) field #8

(I) field #9



(J) field #10

FIGURE 3.8: Dataset: aerial images. Part 1/3



(A) field #11

(B) field #12

(C) field #13

(D) field #14

(E) field #15

(F) field #16

(G) field #17

(H) field #18

(I) field #19

(J) field #20

(K) field #21

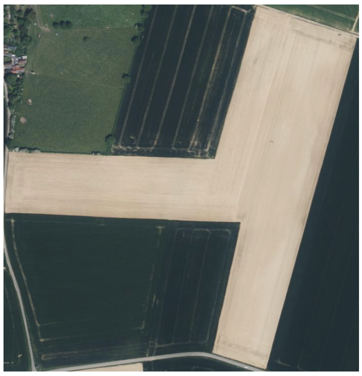
FIGURE 3.9: Dataset: aerial images. Part 2/3

(A) field #22

(B) field #23

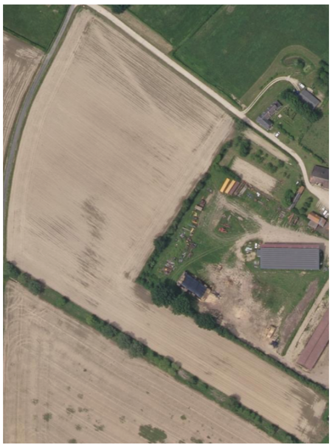
(C) field #24

(D) field #25

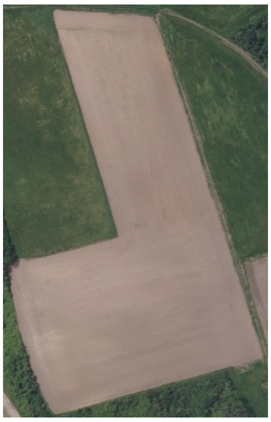


(E) field #26

(F) field #27



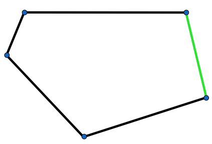
(G) field #28



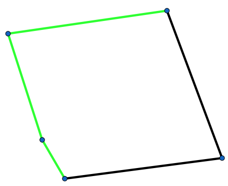
(H) field #29

(I) field #30

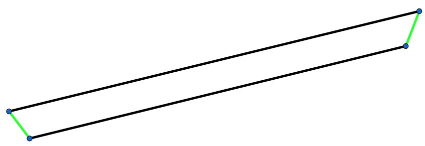
FIGURE 3.10: Dataset: aerial images. Part 3/3



(A) field #1

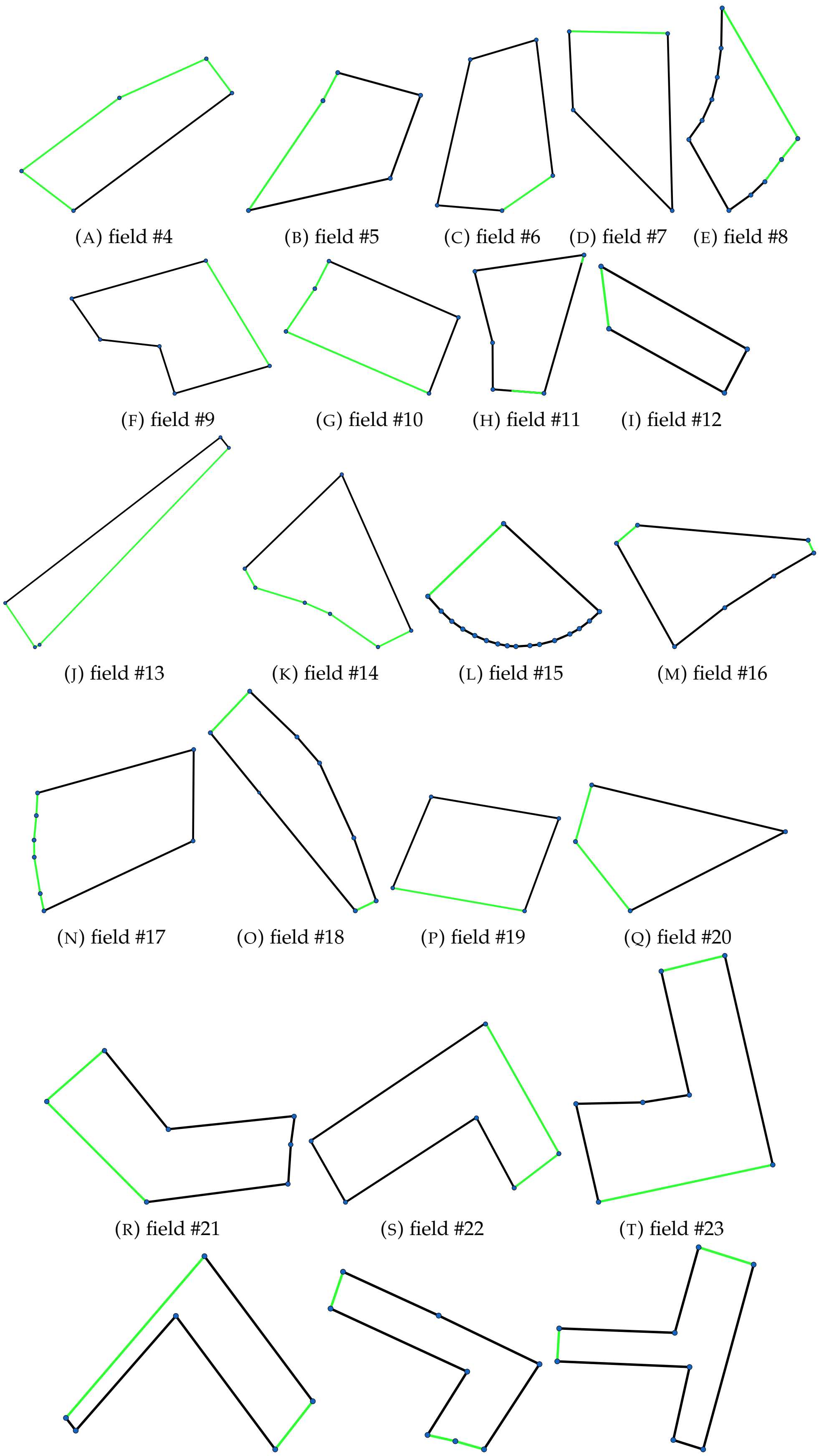


(B) field #2



(C) field #3

FIGURE 3.11: Dataset: 2D polygons and access segments. Part 1/3



(A) field #4 (B) field #5 (C) field #6 (D) field #7 (E) field #8

(F) field #9 (G) field #10 (H) field #11 (I) field #12

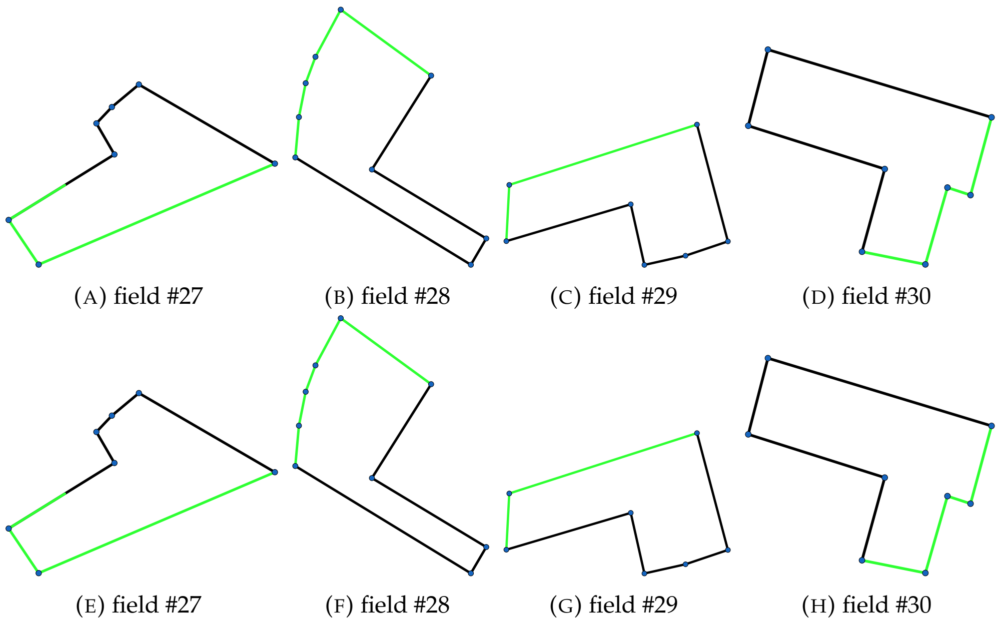
(J) field #13 (K) field #14 (L) field #15 (M) field #16

(N) field #17 (O) field #18 (P) field #19 (Q) field #20

(R) field #21 (S) field #22 (T) field #23

(U) field #24 (V) field #25 (W) field #26

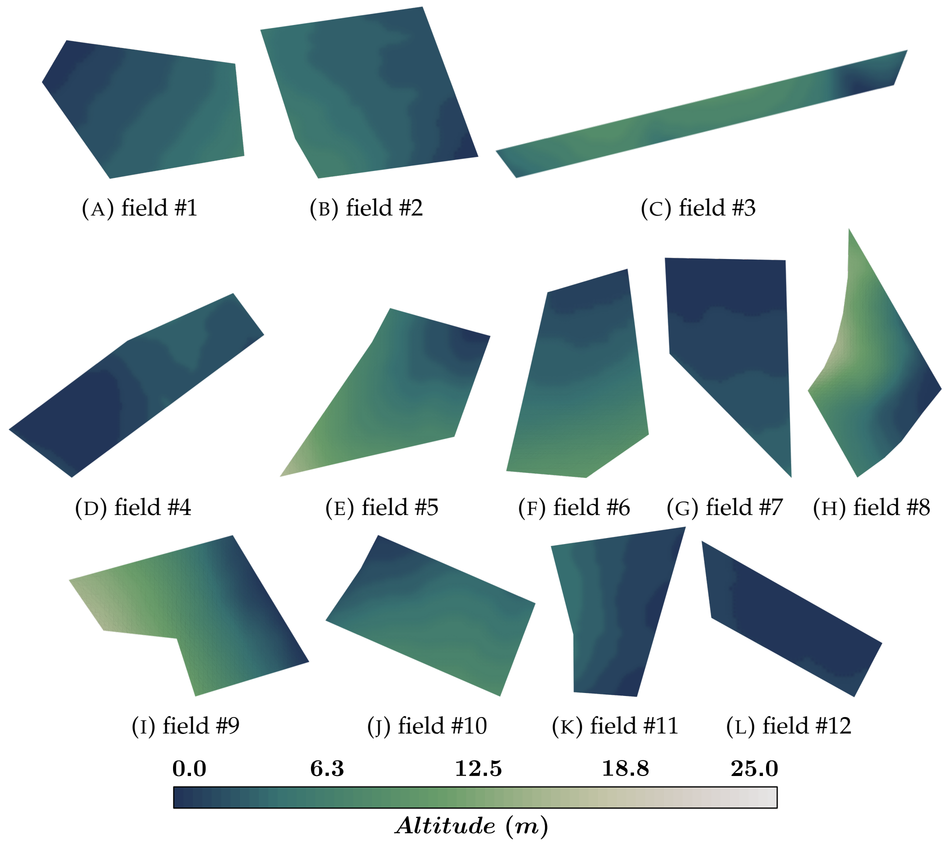
FIGURE 3.12: Dataset: 2D polygons and access segments. Part 2/3



(A) field #27 (B) field #28 (C) field #29 (D) field #30

(E) field #27 (F) field #28 (G) field #29 (H) field #30

FIGURE 3.13: Dataset: 2D polygons and access segments. Part 3/3



(A) field #1 (B) field #2 (C) field #3

(D) field #4 (E) field #5 (F) field #6 (G) field #7 (H) field #8

(I) field #9 (J) field #10 (K) field #11 (L) field #12

FIGURE 3.14: Dataset: 3D surfaces. Part 1/2

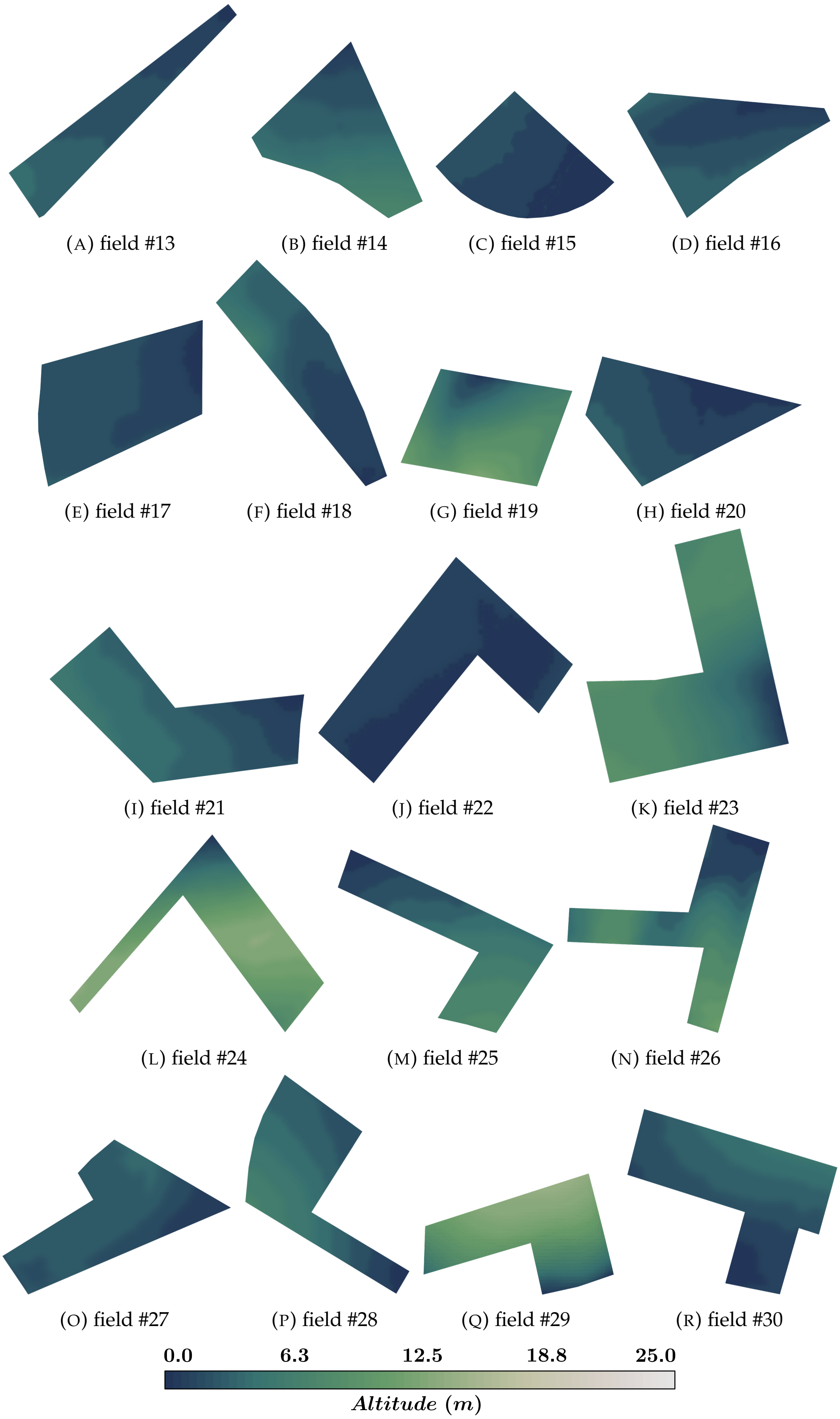


FIGURE 3.15: Dataset: 3D surfaces. Part 2/2

Field	#1	#2	#3	#4	#5
2D area (m^2)	34038.2	50855.9	72346.3	82293.2	43937.7
3D area (m^2)	34076.5	50915.7	72582.8	82351.5	44102.6

Field	#6	#7	#8	#9	#10
2D area (m^2)	22440.8	45813.8	42150.5	25502.7	38734.5
3D area (m^2)	22525.0	45850.7	42639.1	25824.3	38823.5

Field	#11	#12	#13	#14	#15
2D area (m^2)	78497.4	18861.5	58107.7	19502.8	132047.0
3D area (m^2)	78567.6	18883.1	58131.9	19541.5	132149.0

Field	#16	#17	#18	#19	#20
2D area (m^2)	35911.8	34391.0	18263.8	76803.0	43412.9
3D area (m^2)	35958.7	34413.9	18300.8	77056.3	43460.9

Field	#21	#22	#23	#24	#25
2D area (m^2)	30886.2	22148.3	44999.8	61831.5	62527.0
3D area (m^2)	30921.5	22185.6	45103.3	62088.5	62597.0

Field	#26	#27	#28	#29	#30
2D area (m^2)	104066.0	37968.8	35802.8	26899.1	41456.1
3D area (m^2)	104267.0	37999.7	35848.8	27297.0	41505.8

TABLE 3.2: Dataset: calculated area based on 2D and 3D data

Field	#1	#2	#3	#4	#5
Link	bit.ly/3FYtuKu	bit.ly/3WGAyRI	bit.ly/3zX1vqJ	bit.ly/3DJL0PI	bit.ly/3htb8H3
Lon / Lat	7.435° / 48.7732°	7.474° / 48.7825°	2.9205° / 49.8115°	1.6713° / 47.9864°	3.3216° / 50.6623°

Field	#6	#7	#8	#9	#10
Link	bit.ly/3WGTfER	bit.ly/3DP8vqG	bit.ly/3NLmQJf	bit.ly/3EeTvUo	bit.ly/3UOyTrv
Lon / Lat	7.4311° / 48.8245°	2.4845° / 50.3106°	7.5924° / 48.831°	7.4641° / 48.8146°	1.3491° / 48.012°

Field	#11	#12	#13	#14	#15
Link	bit.ly/3zW7v30	bit.ly/3UMC6I3	bit.ly/3TjkOkA	bit.ly/3UAmdo0	bit.ly/3GpjdXZ
Lon / Lat	3.4701° / 46.652°	7.5742° / 48.8071°	3.578° / 46.7016°	7.4269° / 48.8194°	3.5611° / 46.6875°

Field	#16	#17	#18	#19	#20
Link	bit.ly/3tcGhRN	bit.ly/3zW26sE	bit.ly/3Trsqlq	bit.ly/3DWHgKJ	bit.ly/3NN8pnT
Lon / Lat	2.5127° / 48.2645°	2.6443° / 48.2546°	7.9196° / 48.9513°	2.1269° / 46.8124°	1.5874° / 47.1346°

Field	#21	#22	#23	#24	#25
Link	bit.ly/3DShkA3	bit.ly/3zZK1dg	bit.ly/3TmwcMC	bit.ly/3E3l8OK	bit.ly/3E0Raeq
Lon / Lat	0.6254° / 49.191°	2.7067° / 50.3336°	7.4416° / 48.7223°	3.1021° / 48.2449°	1.6183° / 49.9655°

Field	#26	#27	#28	#29	#30
Link	bit.ly/3tvN0Xg	bit.ly/3A0tZ2D	bit.ly/3fTlQGl	bit.ly/3hBeLL2	bit.ly/3Edm2cN
Lon / Lat	3.5476° / 50.1441°	3.6644° / 48.0046°	1.7086° / 47.2054°	1.6893° / 47.1421°	3.1018° / 48.5853°

TABLE 3.3: Dataset: links to annotated data of each field in Géoportail [45] platform as well as the coordinate of a point inside the field

3.4 Conclusion

This chapter has looked at the process of terrain modeling and its role in creating a digital representation of a physical land surface in either 2D or 3D format. It explained that 2D terrain modeling creates a flat projection of the actual terrain onto a plane, while 3D terrain modeling creates a representation that takes into account the actual topography and elevation variations. Despite the fact that 3D terrain modeling offers a more accurate representation of the terrain surface, it has been limited in its use due to computational complexity. In this chapter, a comprehensive overview of the data required for generating both 2D and 3D models of a field was also provided, along with a dataset of thirty fields located in France.

The dataset generated in this chapter can provide valuable insights into the evaluation of future path planning approaches. However at the moment of writing this manuscript, it contains fields with no obstacles inside them. As a future perspective, our aim is to expand the variety of fields by adding more fields, and making it more comprehensive and convenient.

CHAPTER 4

Problem Statements and Challenges

In this chapter, we will focus on discussing the challenges and constraints associated with CCPP for various types of agricultural operations. These operations include fertilizing, harvesting, mowing, planting, seeding, spraying, and tillage. We will analyze some general metrics and constraints that are applicable to most of these operations, as well as any additional factors that need to be taken into account, such as the minimum working distance, turning limits, the implement transition state, robot capacity, and predefined trajectories or driving direction.

To address these challenges, we will propose a first approach, which is an exhaustive CCPP approach. This approach involves a complete search of all possible paths in the field, taking into account the various constraints and limitations. We will evaluate the performance and limitations of this approach through simulations. The results of this evaluation will provide a foundation for the development of more advanced and efficient CCPP approaches in future chapters.

The remainder of this chapter is organized as follows: the challenges and constraints encountered in CCPP for various agricultural operations will first be covered in Section 4.1. An exhaustive approach for CCPP will then be proposed in Section 4.2. Finally, Section 4.3 concludes this chapter.

4.1 Agricultural Operations and CCPP

Agricultural operations such as fertilizing, harvesting, mowing, planting, seeding, spraying, and tillage are critical processes for ensuring high-quality crops. To achieve complete coverage of the field during these operations, a CCPP approach must be used to compute a path for the machinery. This is crucial to guarantee uniform crop growth and maximize yield.

However, the implementation of CCPP depends on the specific operation being performed and can vary based on the machinery used. As a result, several factors and constraints must be considered to ensure that the CCPP is performed optimally and effectively. These factors and constraints may differ between the different types of operations.

In the rest of this section, the focus will be on providing definitions and examples to highlight some differences between various types of agricultural operations. After that, various metrics and constraints will be discussed that are necessary to ensure optimal and effective implementation of the CCPP approach for these operations.

4.1.1 Definitions

Depending on the operation and required machinery to perform a CCPP, several factor and constraints must be taken into account. To begin, let's define a few terms and notations. We define w as the *working width* which corresponds to the implement width attached to the robot. The implement can be either *on* or *off*. Accordingly, *worked area* is the surface that is covered while the implement is on. Conversely, the surface that is never covered while the implement is on is referred to as *unworked area*.

For some operations such as tillage like operations, the implement is in contact with the ground while it is on. In such operations, to perform a half-turn or a tight turn the implement must be turned off and elevated to avoid damages. To perform such turns, the minimum turning radius of the robot γ_{off} must also be respected. When the implement is on and it is in contact with the ground, it might be possible to perform slight turns with a greater minimum turning radius γ_{on}. Consequently, the corresponding surface will be considered as worked area.

During such operations, the robot cannot quickly switch the implement state from on to off or vice versa. It needs to be carried out gradually while the robot advances straight forward. Therefore, switching the implement from on to off and vice versa is referred to as *transition state*. The straight trajectories during the transition state, that is required before and after each tight turn, have a constant length of ℓ_t. During these trajectories, that referred to as *transition trajectories*, the surface remains unworked. These unworked areas are referred to as *gaps*.

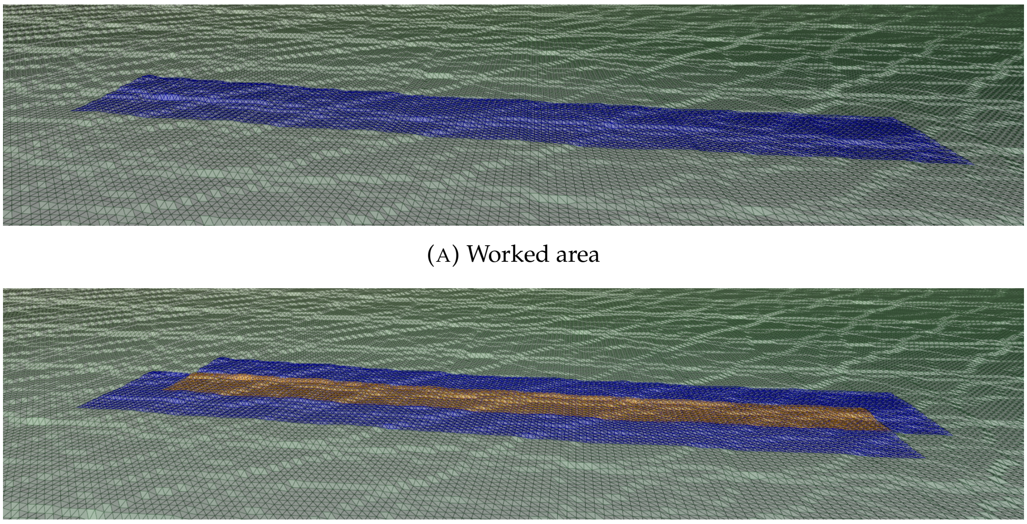
For some other operation in which the implement is not in contact with ground or crops, the implement state can be changed instantly from on to of or vice versa and the robot can perform tight turns event with an activated implement. Consequently, except worked and unworked areas other definition are redundant for such operations. For some other operations the trajectories or the driving direction may be predefined based on the direction of already grown crops. For instance for spraying, the trajectories can be predefined based on the previous seeding operation using a tramline farming system [11]. In such operations the problem can be solve applying a AVRP approach.

As result, considered definitions, metrics and constraints for an operation may differ from those for another operation. The remainder of this section discusses all metrics, constraints, and crucial factors needed for various field operations.

4.1.2 Metrics

Worked area: the first important metric for all operations in is the worked area. This metric is calculated only when the implement is turned on. In case of a 2D and flat field the worked area for a straight trajectory can be calculated as the product of the product of w and the trajectory length. For a slightly curved trajectory, however, it must be first sampled to a set of points. Having the direction at each simple, the two lateral ends of the implement are then computed and a trapezoid is created for each pair of consecutive samples. In this case the worked area is simply the sum of areas of all trapezoids.

In case of a 3D and inclined field, the worked surface for a straight trajectory is first acquired as a 2D rectangle, and for a curved trajectory as a set of 2D trapezoids. Once one or a set of 2D polygons is acquired, a triangulated surface of the working trajectory is constructed in the same way that we constructed a 2D field surface in Section 3.2. Afterwards, the elevation of each point of the generated surface(s) is calculated using RCI approach. Finally the worked area is calculated as the sum of areas of all triangles. Fig. 4.1a illustrates the computed worked area on a 3D surface for a trajectory of length $30m$ while w was set to $4m$. For this trajectory the worked area on a 2D surface was $120.0m^2$ while the worked area on a 3D surface was $121.5m^2$.



(A) Worked area

(B) Overlap area

FIGURE 4.1: Illustration of computed worked and overlap areas on a 3D surface, with worked area represented in blue and overlap in orange

Overlap area: This metric, that calculates the overlap area of two trajectories, is only calculated for two trajectories along which the implement is on. A 2D polygon representing the worked surface is first generated for each trajectory. Afterwards, a 2D polygon referred to as the intersection surface is created by calculating their intersection. In case of a 2D and flat field the overlap area is simply the area of the intersection surface. In case of a 3D and inclined field, the overlap area is determined in the same way as the working area following resampling, triangulation and elevation interpolation of the intersection surface.

Inside: This Boolean metric verifies whether the robot and its implement are completely inside the field. An accurate computation of this metric requires the size of the robot and its implement as well as their position. However, most of the time the implement width is was greater that the robot width. Therefore, it can be computed knowing the position of two lateral ends of the implement. This metric can be calculated using only 2D data because if a point is placed on a polygon's 3D surface, their vertical projections on a 2D plane will likewise be superimposed.

Damage: Not all operations require this Boolean metric and it might need to be customized for different operations. For instance for seeding, it verifies whether the robot is crossing a previously worked surfaces while its implement is turned off. This metric in this case is helpful to avoid unnecessary damages to previously worked surfaces with the robot wheels, and without working it again. For harvesting, however, it quite opposite. It verifies whether the robot is crossing an area that is not worked yet while its implement is turned off. This metric, in this case is used for detecting damages to unharvested crops. These two types of damages are distinguished by the terms *work damage* and *crop damage* respectively.

The position of the robot's wheels and/or the implement must be known in order to calculate the damage. To achieve this, first a 2D polygon of worked area of each prior traveled trajectory with an activated tool is constructed. Depending on the operation type, this metric is then calculated deducing whether the robot's wheels and/or the implement are located inside at least one polygon. Similar to the previous metric, this one can likewise be calculated using only 2D data.

the worked area, overlap area, and inside metrics are relevant to all mentioned operations, while the damage metric only applies to specific operations where it is necessary to avoid damages to crops or previously worked surfaces. Table 4.1 gives an overview of the damage metric required for different type of operations.

Operation type	Fertilizing	Harvesting	Mowing	Planting	Seeding	Spraying	Tillage
Damage	NA	crop damage	NA	work damage	work damage	NA	work damage

TABLE 4.1: Damage constraint applicable to various operation types

4.1.3 Constraints and Important Factors

The following constraints can be defined in accordance with the metrics described in the previous section:

- maximizing the worked area
- minimizing the overlaps
- preventing damages to previously worked areas or to unharvested crops
- the robot and its implement must remain inside the field during the operation

where according to Table 4.1, only for spraying, fertilizing and mowing operations, preventing damages is not applicable.

To provide a precise and reliable CCPP approach, in addition to these constraints, additional important factors must be taken into account. These factors are defined as follows:

The *Minimum Working Distance constraint (MWD)*: This constraint is mainly relevant for operations in which the implement is in contact with the ground while it is turned on. According to this constraint, the robot must travel at least Δ_{mwd} when its implement is turned on, to authorize to turn it off. The primary reason of this constraint is because lowering the implement into the ground and raising it only for a short distance is can be both inconvenient and expensive.

Turning limit: This constraint is also relevant only for operations in which the implement is in contact with the ground while it is turned on. This constraint indicates that while performing half-turns or tight turns, the implement must be lifted and turned off in order to prevent damage to the implement or the robot. Depending on the implement, slight turns while it is turned on might be authorized.

Implement Transition State (ITS): For some operations, changing the state of the implement from on to off or vise versa can not be done abruptly. To turn the implement on or off, a transition state must be performed on a straight transition trajectory.

Robot capacity: The ability of the robot to transport agricultural materials must be taken into account for some operations. For instance, seeding requires a seed storage onboard.

Predefined Driving Direction (PDD): For some operations, the trajectories or driving direction may be predetermined depending on the previous operation carried out on the field.

Operation type	MWD	Turning limit	ITS	Robot capacity	PDD
Fertilizing				✓	✓
Harvesting		✓		✓	
Mowing		✓			
Planting	✓	✓	✓	✓	
Seeding	✓	✓	✓	✓	
Spraying				✓	✓
Tillage	✓	✓	✓		

TABLE 4.2: Relevant constraints and factors for various operation types

Table 4.2 summarizes the constraints and factors that must be taken into account for different types of operations. However, mentioned metrics, constraints and factors may varies depending also on the kind of plant being cultivated on the field as well as the machinery. For instance, harvesting may perform in multiple steps and require different machinery for forage crops comparing to corn crops.

4.2 Exhaustive Approach for CCPP

This section describes a preliminary attempt towards developing an exhaustive CCPP approach for a simplified tillage-like operation while considering some of constraints that were detailed in the previous section.

In general, in terms of processing time and resources, an exhaustive method could be quite expensive. Therefore, it would not be a feasible approach for widespread usage. However, such an approach could be applied only on a limited number of fields with a variety of shapes to find the optimal solution. These solutions can be then used as references for evaluation further lightweight and robust methods.

4.2.1 Simplified Tillage-Like Operation

To develop the exhaustive approach, a simplified tillage-like operation was considered with the following assumptions:

- the field contains no obstacles
- the robot can enter and exit the field at any point along its boundaries
- the robot can only move straight forward when the implement is turned on
- the robot can instantly turn on or off the implement

- the robot has an infinite energy and agricultural material capacity
- there is no distance between the robot and its implement

4.2.2 Tree Construction and Exploration

The exhaustive approach consists of constructing a tree, where each node represents a pair of location and direction for the robot. The root of the tree represents the initial location and direction of the robot. The node generation and exploration process starts from the root using a depth-first exploration approach.

As illustrated in Fig. 4.2, the node generation and exploration start from the root as a depth first exploration. For each unvisited node three children are generated: one forward node representing the forward move of the robot while its implement is turned on, and two turning nodes for turning to right and left while the implement is off. Therefore, a turn or half-turn can be constructed as a sequence of turning nodes.

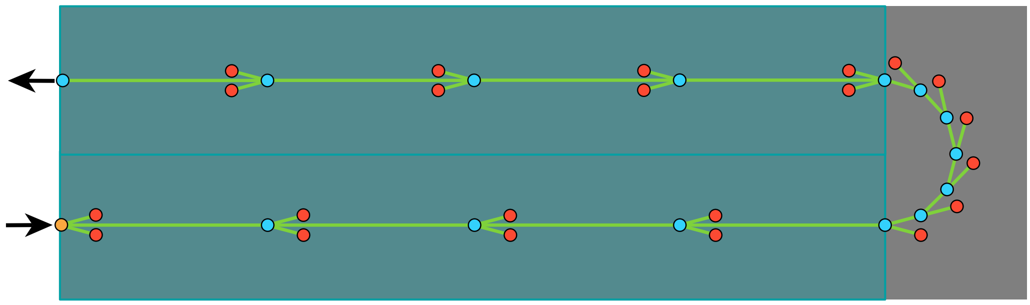


FIGURE 4.2: Construction and exploration of the tree. Orange circle represents the root. Visited nodes are represented as blue circles while unvisited nodes are red

As illustrated in Fig. 4.3, the spacing and the direction of turning nodes are chosen such that it includes a change of direction of 90° and 180°.

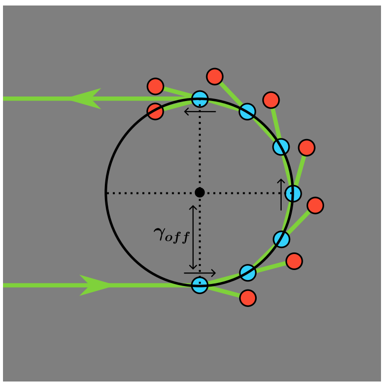


FIGURE 4.3: Turn sampling. Visited nodes are represented as blue circles while unvisited nodes are red

A node generated by the node generation and exploration process is added to the tree if and only if the robot's position and the two lateral ends of the implement are found inside the field.

A path *i.e.*, a sequence of trajectories is represented by a branch of the tree from the root to a leaf node. A path that covers at least Δ_{cov} percent of the field surface is considered as a solution. All found solutions are then added to a solution space and the one that results the best field coverage is selected.

4.2.3 Computational Time Estimation

The processing time of the exhaustive approach was initially estimated using a synthetic field of $194.0m^2$. This field is illustrated in Fig. 4.4 while the red arrow indicate the initial location and direction of the robot.

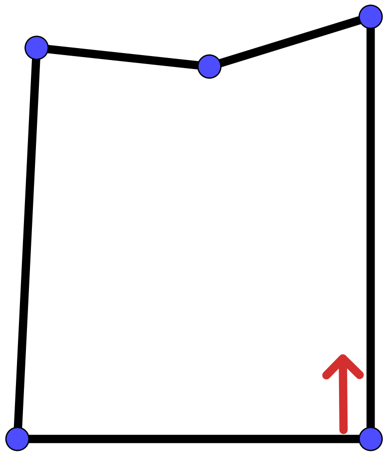


FIGURE 4.4: The shape of a synthetic field used for estimating the computational time of the proposed approach

The length of a forward branch and a turning branch was set to 300.0 and $46.9cm$ respectively while turning branches were generate with a deviation of $9°$ to the left and right. w, γ_{off}, Δ_{mwd} and Δ_{cov} are respectively set to $3m$, $1.5m$, $9m$ and 70%.

Two heuristics are also applied to stop the node generation for branches that cause:

- overlap of more than 5% of the field area
- consecutive turning branches that their cumulative length exceeds $10m$

To cover at least 70% of the field illustrated in Fig.4.4, we estimated that at least a sequence of 80 trajectories is needed. This implies that a tree with a height of at least 80 must be constructed and explored.

The number of nodes of a tree where each intermediate node have exactly three children can be computed as follows:

$$V = \frac{3^d - 1}{2} \tag{4.1}$$

where V is the number of nodes and d indicate the height of the tree (the height of the root is considered as one). However, in this approach some nodes might not be validated by imposed heuristics or constraints which would prevent them from being added to the tree. Therefore the number of nodes can not be estimated accurately by Equation 4.1.

To determine the number of nodes accurately, first, the precise number of nodes is calculated for low heights where it is feasible to construct the tree and counting its nodes in a reasonable amount of time. Therefore, the accurate number of nodes is acquired limiting the height of the tree to $10, 15, ..., 65$. Afterwards, for a known number of nodes V, in Equation 4.1, the height of the tree d is replaced by an unknown variable u that can be calculated by Equation 4.2. Such that, a linear regression between the height of the tree and u may provide a better approximation of the number of nodes for greater heights where manually counting the number of validated nodes can takes too much time. Table 4.3 summarizes the number of node and the value of u for different heights and Fig. 4.5 illustrates their linear relationship.

$$u = 1 + \log_3 (V \times 2) \tag{4.2}$$

Height	10	15	20	25	30	35	40	45	50	55	60	65
V	$5.1e2$	$1.0e4$	$2.1e5$	$9.9e5$	$2.4e6$	$9.3e6$	$5.1e7$	$2.1e8$	$6.2e8$	$2.1e9$	$4.3e9$	$5.5e9$
u	6.1	9.0	11.8	13.2	14.0	15.2	16.8	18.1	19.1	20.2	20.8	21.1

TABLE 4.3: The exact number of nodes for different height of the tree

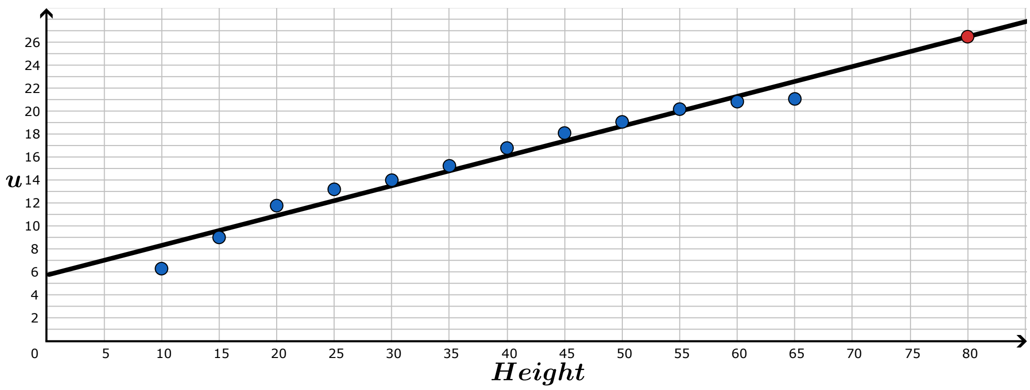


FIGURE 4.5: The linear regression between tree heights and calculated u

According to the linear model, for a tree construction and exploration with a height of 80, the estimated value of u is 26.5. Replacing d by u in Equation 4.1, the number of validated nodes is then calculated as 2.2×10^{12}.

We also determined that visiting 4×10^6 nodes *i.e.*, generating one forward and two turning nodes as children of the unvisited node and validating them, takes almost one minute by one CPU and five second by twelve CPUs. Therefore, covering at least 70% of the synthetic field takes almost 382 and 32 days respectively by one and twelve CPUs.

4.3 Conclusion

In this chapter, the focus was on the examination of the difficulties and limitations encountered in the implementation of CCPP for various agricultural operations, including planting, harvesting, fertilizing, tillage, mowing, seeding, and spraying. The study analyzed common constraints and metrics that apply to most of these operations, as well as specific considerations including minimum working distance, turning limits, implement transition state, robot capacity, and predefined trajectories or driving direction.

A first approach, the exhaustive CCPP approach, was proposed to tackle these challenges. This approach involved a comprehensive search of all possible paths in the field, taking into account the various constraints and limitations. The performance and limitations of this approach were evaluated through simulations.

The simulation results showed that even for small size fields and a simplified operation with several heuristics while considering only three degree of freedom for the robot, it took a huge amount of time to find a solution. As a result, it might be impossible to implement an exhaustive CCPP approach for a field operation using the available computational resources and programming paradigm.

Although the proposed exhaustive approach was found to be impractical on real-world fields due to required computational time and resources, it served as a foundation another approach that utilize intelligent tree construction and generation. This approach is detailed in the following chapter.

CHAPTER 5

Intelligent Tree-based Search

In this chapter a novel CCPP approach is proposed to define the ideal motions of mobile robots across an agricultural field. This approach is inspired from the exhaustive approach presented in chapter 4 Section 4.2. However, the problem is reformulated to find feasible solutions for big size fields in a reasonable amount of time. Similar to the exhaustive approach, this novel CCPP includes a tree construction and exploration to find all potential solutions that satisfy a set of hard constraints. Afterwards, the best solutions are identified through a process called *Similarity check and selection of optimal solutions*. The optimization objectives are to maximize worked area and to minimize overlaps, non-working traveled distance and operation time.

The remainder of this chapter is organized as follows: The target operations as well as the objectives and main contributions of the proposed approach are presented in Section 5.1. Section 5.2 provides a detailed description of this approach. In Section 5.3, The results of an experimental study on real fields is provided and a comparison to ground truth is conducted. The importance and efficiency of the proposed approach are also highlighted. Finally, Section 5.4 brings this chapter to a conclusion.

5.1 Objectives, Motivations and Contributions

As described previously in Chapter 4 Section 4.1.3, different constraints and factor may apply to different kind of operations. For instance, the trajectories for fertilizing and spraying are predefined based on the prior operations or the direction of planting/seeding. For these two kind of operations only a AVRP approach is sufficient to determine the optimal sequence of trajectories while for other kind of operations, both the trajectories and their sequence must be determined. Therefore, to cover a wide range of operations, we decided to propose a CCPP approach for tillage like operations in which the implement is in contact with the ground when it is turned on. In this way, the proposed approach could be adapted to find solutions for harvesting, mowing, planting and seeding with a slight adjustment of the parameters and/or constraints.

As described in chapter 2, all approaches proposed in the literature, performed CCPP as two distinct tasks: CMC and AVRP. That appears to be an effective strategy to solve this difficult and complex problem. However, this strategy could have certain disadvantages and might miss some interesting solutions. We believe that generating a coverage path using a one-step approach might lead to some interesting solutions that can not be found by a two-step approach. The novelty of the approach we provide is to generate the parallel tracks and the turns in a single process, with the objective of allowing more possible alternatives.

Moreover, a variety of simplifications and assumptions have been applied in the literature to overcome the complexity of the problem. Most of these simplification facilitate the identification of a feasible solution for a variety of fields in a reasonable amount of time, even if it may not be the most optimal. Applying simplifications might accompany the risk of oversimplification, which might lead to unattainable solutions in practice. For instance, presuming that a field can be accessible from all of its boundaries, might result in a solution that the farmer couldn't use in practice. Since being accessible from all boundaries is not the case for the majority of fields, crossing some edges may damage the robot or even the neighboring field.

A basic strategy to solve this problem was to generate an inner offset for a field polygon as headlands. Covering the headlands was then performed manually at the end of operation. Only Edwards et al. [37], Jeon et al. [65], Nilsson and Zhou [96], and Nørremark, Nilsson, and Sørensen [99] took into consideration the automatic coverage of the headlands. However Edwards et al. [37] and Jeon et al. [65] considered all field boundaries as accessible. Consequently, their approaches might generate half-turns in headlands that cross the field edges. Conversely, another simplification might be to only consider one or two distinct points as potential entry or exit locations. This would severely limit the range of potential solutions. This strategy was utilized by Nilsson and Zhou [96], who only took into account one point for both entering and exiting the field. Only Nørremark, Nilsson, and Sørensen [99] considered both headland coverage and access segments on the field boundaries.

For operations in which the implement is in contact with the ground while it is turned on, two other simplifications can be mentioned. The first is supposing that tight turns can be made with the implement on. This might cause damages to the implement or reduce its lifespan. The second simplification is to consider that lowering and raising the implement could be done instantly which is not true and it would lead to an overestimation of the worked area.

Therefore, another motivation of this approach is to provide feasible coverage path for fields of various shapes and size, while preventing oversimplifications as much as possible. To achieve this goal, we propose an integrated system taking into account the following considerations and contributions:

- automated identification of entry and exit point on the access segments
- generation of tracks and turns in one step while optimizing worked area, overlaps, damage, and operation time
- various dividing lines to split complex fields to several simple shape sub-fields
- providing several optimal paths with a variety of properties
- intelligent coverage of the headlands
- geometry of the vehicle and the implement are taken into account:
  - offset between the vehicle and the implement
  - minimum turning radius of the vehicle when the implement is turned off

  - minimum turning radius of the vehicle when the implement is turned on

- distance needed for lowering/raising the implement is taken into account
- reverse moves are allowed for performing turns and half-turns
- curved edges are taken into account

5.2 Methodology

Our approach include the following three phases:

- preprocessing
- tree-based intelligent search: tree construction and exploration
- similarity check and selection of optimal solutions

Preparing the field is the goal of the preprocessing phase. Its inputs are the field polygon, one or several dividing lines to split the field polygon if needed, the access segments, the working width, and the minimum turning radius of the robot. The output of this step is a set of entrances, the headlands and a set of *turning spaces*. Turning spaces are essential for perform a turn from one headland to another or taking a trajectory inside of a headland.

The second phase, also referred to as *exploration algorithm*, serves to discover almost any possible solutions and store them in a solution space. A solution is a *coverage path i.e.*, a sequence of trajectories that begins at an entrance, covers the field and the headlands at best, and ends on one of access segments.

If many entrances and/or dividing lines are given, these two phases would be performed numerous times. The number of explorations is equal to the number of entries for a simple field where no decomposition is necessary and the preprocessing is performed only once.

For a complex field, that d different dividing lines are given and e entrances are found, it requires to perform d reprocessing and $d * e$ explorations. The approach conducts also one additional prepossessing and e exploration while considering nod dividing line. Since the optimal solution for some concave fields might be found without splitting the field. For instance, detecting three possible entrances and proposing three distinct dividing lines requires the preprocessing phase to be run four times (one time with no dividing line plus the number of dividing lines). Consequently, it requires twelve iterations of the exploration algorithm (the product of four different preprocessing results and three entrances). As a result, the twelve solution spaces acquired by the exploration algorithm are merged into a single solution space.

The cost of each solution is then calculated during the similarity check and selection of optimal solutions. Then, families of solutions are constructed by grouping similar solutions together using a similarity function. Finally, for each family, just the least costly solution is preserved. In the following sections, after providing a few definitions, each of these phases is detailed.

5.2.1 Definitions

To describe the proposed approach, some notions and notations previously defined in Chapter 4 Section 4.1 are required. In this section, we provide a brief reminder of them. These notions can be summarized as follows:

- w: the working width *i.e.*, the width of the implement
- **implement sates**: *on*, *off* and *transition state*
- **worked area**: the surface that is covered when the implement is on
- **unworked area**: the surface that is not worked
- γ_{off}: the minimum turning radius of the robot while its implement is off
- γ_{on}: the minimum turning radius of the robot while its implement is on
- **transition trajectory**: a straight trajectory required during the transition state
- ℓ_t: the minimum length of a transition trajectory
- **gaps**: the unworked area due to transition trajectories
- ℓ_o: the distance between the robot and its implement

5.2.2 Preprocessing: Defining Headlands and Turning Spaces

Before launching the exploration algorithm, a preprocessing phase is required to determine entrances, generate headlands and turning spaces. The following sections provide details on each of these preparatory process.

Entrances

An Entrance include the location where the robot can enter the field as well as the direction of the robot at this location. Given the field polygon and the access segments, our approach determines the entrances on each corner of an access segment where its distance from the adjacent boundary of the field polygon is $w/2$. Its direction is the same as the direction of the adjacent boundary. Once all potential entrances have been identified, an expert must confirm them or reject those that appear redundant.

Figs. 5.5 and 5.6 represent the selected entrances for the proposed dataset in Chapter 3 Section 3.3.

Headlands

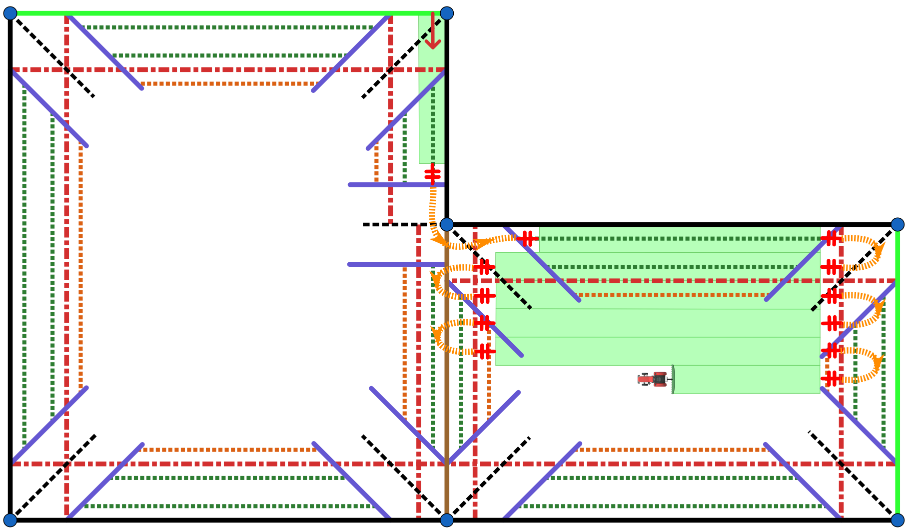
A headland is a space adjacent to the field boundary where half-turns can be performed. When half-turns are performed in a headland, this space is remains unworked. Each half-turn also causes two gaps before and after it along the trajectories. Headlands are generally worked once the main part of the field has been completely worked or the headland is no longer needed for performing half-turns. Our approach begins by identifying these headlands and envisioning how they could be covered.

For each field boundary, a headland area is constructed containing the following information:

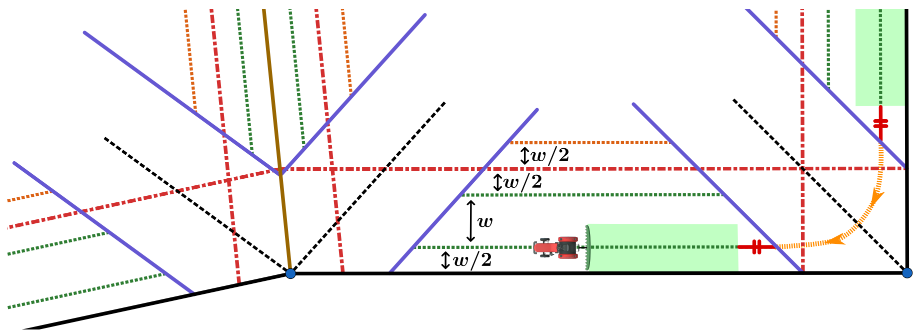
- one *outer border* that lines up with the field boundary

- one *inner border* generated parallel to the outer border inside the field at a distance of $p * w$, where p is given as an input, or indicated from w and γ_{off}
- p *inner trajectories* generated between the outer and inner borders and parallel to them, at a distance of $w/2$ from the borders and w from each other
- one *gap-covering trajectory* generated inside the field at a distance of $w/2$ from the inner border and parallel to it

Fig. 5.1 illustrates these elements with $p = 2$. Another type of headland is defined if a dividing line is given. It is completely contained within the field polygon, has two inner borders, p inner trajectories, and two gap-covering trajectories centered around the dividing line, and has no outer border. Fig. 5.1 also illustrates this kind of headland where the dividing line is depicted in brown.



(A) Preprocessing result for a field



(B) A close-up on a preprocessing result

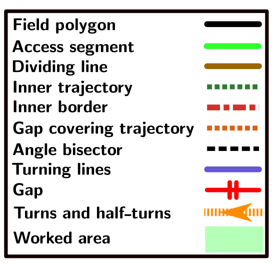
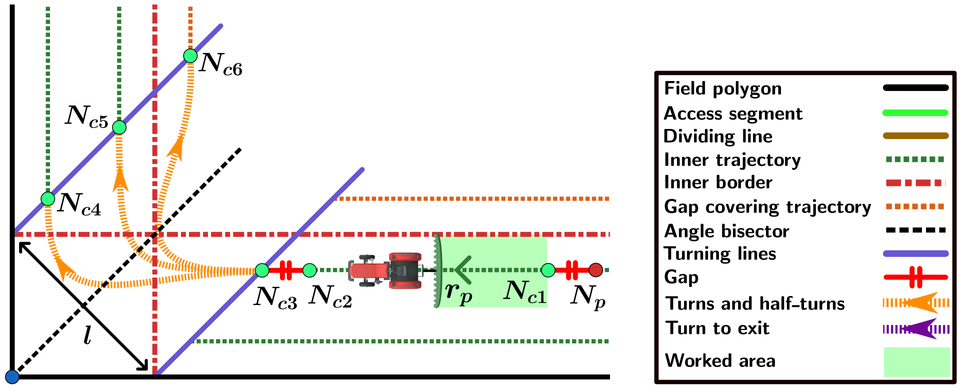


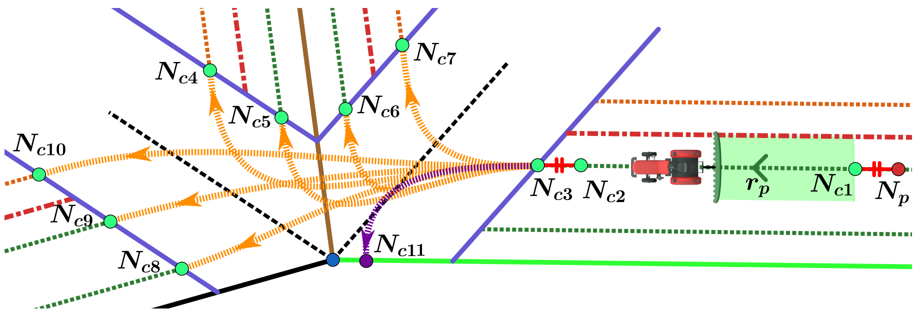
FIGURE 5.1: Preprocessing result for $p = 2$. The field polygon is represented by blue points and black and green solid segments

The gap-covering trajectories are utilized to cover the unworked areas created by gaps, whereas the inner trajectories are mostly beneficial for covering the unworked areas caused by half-turns.

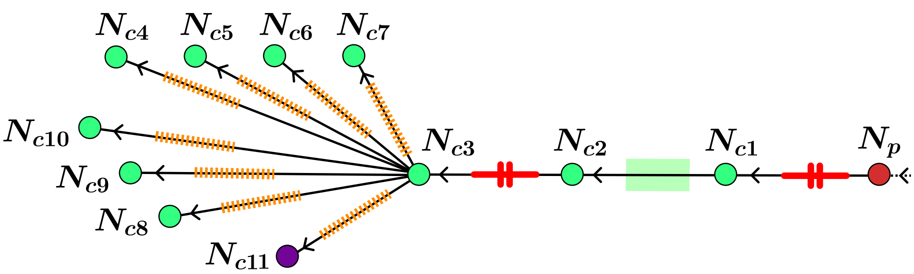
Turning Spaces



(A)



(C)



(D)

FIGURE 5.2: Trajectory (*a*, *c*) and tree (*b*) representations of turns inside the turning spaces. Sub-figure (*b*) is the tree representation of trajectories of (*c*)

A turning space is constructed at the angle between two adjacent headlands. A turning space ensure a safe and feasible turn to travel from one headland to another. Therefor a turning space can be used for:

- Traveling between inner trajectories of adjacent headlands
- traveling from one of inner trajectories of a headland to the gap-covering trajectory of the another one

- traveling from the gap-covering trajectory of a headland to one of inner trajectories of the another one
- Traveling between the gap-covering trajectories of adjacent headlands
- switching between sub-fields if a dividing line is provided

As shown in Figs. 5.2a and 5.2c, the angle bisector is first calculated (black dashed line) for two adjacent headlands. Two *turning lines* (blue solid lines) are then calculated parallel to the angle bisector with a spacing of $l/2$ where l is calculated as follows:

$$l = \max\Big(\sqrt{2}(p * w)\ ,\ 2(\ell_o + \gamma_{off})\Big) \tag{5.1}$$

The intersections of turning lines with inner trajectories or gap-covering trajectories are utilized to determine start and end point of turning trajectories for traveling from one headland to another. As a result, regardless of angle between two adjacent headland, turning spaces can guarantee a sufficient space for traveling from one headland to another.

It is important to note that not all of the headlands and turning spaces produced during the preprocessing phase will be utilized. The exploration algorithm will determine which ones must be used. Section 5.2.4 provides more details about how turning spaces and headlands are used.

5.2.3 Trajectory Types and Metrics

It is crucial to present certain practical concepts and metrics to make the proposed approach easier to comprehend. Hence, this section describes several types of trajectories and all possible and valid sequences that can be formed by these trajectories. Additionally, metrics included in our approach are described.

To generate continuous and smooth turns, Dubins trajectories (first proposed by Dubins [35]) and Reeds–Shepp curves (fist proposed by Reeds and Shepp [120]) have been employed.

Taking as input a minimum turning radius, the departure and destination locations, and the orientation of the robot at these locations, both approaches calculate the shortest curve from the departure to the destination location. To mention their difference, a trajectory generated by Dubins method contains only forward moves while the Reeds-Shepp method considers also reverse moves.

A turn containing a reverse move requires the robot to stop, reversing its move, accelerate, stop, do the reversing again and attain its target speed. As a result, in terms of trajectory length, a turn produced by the Reeds-Shepp method is optimal. However, in term of travel time, the Dubins method may provide a better turn. Furthermore, during a slight turn respecting γ_{on}, the Dubins method can be employed to generate slight turns, whereas the implement must remain off if a reverse move is required. As a result, our approach prioritizes turns without any reversal moves. A turn with reverse moves is also examined, if performing a turn with no reverse moves is not possible.

Taking into account these two types of turns as well as three various states of the implement (on, off, and transition), six different kinds of trajectories can be defined as follows:

- ***STRAIGHT_ON***: straight trajectory (implement on)
- ***DUBINS_ON***: turn computed via Dubins method (implement on)
- ***DUBINS_OFF***: turn computed via Dubins method (implement off)
- ***REEDS_OFF***: turn computed via Reeds-Shepp method (implement off)
- ***GAP_OFF_ON***: transition trajectory of length ℓ_t (implement in transition from off to on)
- ***GAP_ON_OFF***: transition trajectory of length ℓ_t (implement in transition from on to off)

When constructing a sequence of these six kinds of trajectories a few rules must be respected. As the first rule, a *GAP_OFF_ON* trajectory must always be followed by a *STRAIGHT_ON* trajectory, where both trajectories have the same direction. The second rule is that a *STRAIGHT_ON* trajectory only can be followed by a *DUBINS_ON* or a *GAP_ON_OFF* trajectory. Consequently the third rule indicates that a *GAP_ON_OFF* trajectory can be either used for exiting the field or it must be followed by a *DUBINS_OFF* or a *REEDS_OFF* trajectory. According to the fourth rule, a *DUBINS_OFF* or a *REEDS_OFF* trajectory can be either used for exiting the field or it must be followed by a *GAP_OFF_ON* trajectory. These rules are referred to as *trajectory sequence rules*. As a remark, a trajectory used for exiting the field must end up on an access segment.

A path is a sequence of k trajectories, including six types of trajectories. A trajectory Λ_i with a length of ℓ_i, where $\{i \in \mathbb{N} | i \leq k\}$, is defined as $\Big((P_i, \vartheta_i), (P_i', \vartheta_i'), \Gamma_i\Big)$. Here, (P_i, ϑ_i) represents its starting point and direction, (P_i', ϑ_i') represents its destination point and direction, and Γ_i denotes its trajectory type. Assuming that the robot and its implement are moving in the same directions, P_i and P_i' denote the location of the implement. The location of the robot on the path can be calculated using the distance between the implement and the robot.

It is important to note that a *DUBINS_ON* trajectory must only be used to travel from one headland to another while respecting γ_{on}. The angle between these two headland must be between $\pi - \alpha$ and $\pi + \alpha$ where α is calculated as follows:

$$\alpha = \arcsin\left(\frac{l}{2 * \gamma_{on}}\right) * 2 \tag{5.2}$$

To evaluate the suitability of a trajectory being added to a sequence of trajectories, the four metrics previously defined in Chapter 4 Section 4.1.2, are employed. These metric are:

- **worked area**: only the 2D computation of this metric is considered
- **overlap area**: only the 2D computation of this metric is considered
- **work damage**
- **inside**

The worked area is only calculable for *STRAIGHT_ON* and *DUBINS_ON* trajectories, and the working area for all other types of trajectories is zero. The overlap area can be calculated for two trajectories of types *STRAIGHT_ON*, two *DUBINS_ON*, or one *STRAIGHT_ON* and one *DUBINS_ON*. Otherwise it is equal to zero.

The work damage verifies whether the robot is crossing a trajectory that was previously worked (types *STRAIGHT_ON* or *DUBINS_ON*), with a new trajectory while its implement is not turned on (types *DUBINS_OFF*, *REEDS_OFF*, *GAP_OFF_ON* or *GAP_ON_OFF*). This prevents unnecessary damages to previously worked surfaces with the robot wheels without working it again.

5.2.4 Intelligent Tree-based Search: Construction and Exploration

The general idea of the intelligent search is to construct a tree of nodes representing potential sequences of trajectories (locations, directions, and trajectory types) satisfying the hard constraints. It takes as input the result of the preprocessing, the access segments, a set of hard constraints, γ_{on}, γ_{off}, ℓ_t, ℓ_o and the *coverage threshold*. The concept of a node, the hard constraints, and the procedure for creating the tree are all explained in the following sections.

Nodes

Every node in the tree represents a potential candidate for the trajectory's next step. It includes a flag designating one of the six different trajectory types as well as the associated destination and direction. A set of parent and child nodes N_p and N_c are represented respectively by $(P_p, \vartheta_p, \Gamma_p)$ and $(P_c, \vartheta_c, \Gamma_c)$. Consequently, a trajectory Λ_c from N_p to N_c is represented as $\Big((P_p, \vartheta_p), (P_c, \vartheta_c), \Gamma_c\Big)$.

The root of the tree is a specific node represents the entrance location and direction, as well as a default trajectory type with the implement off. A leaf node of the tree must contain an exit point located on an access segment, a direction, as well as a trajectory type for which the implement is not turned on. As a result, a solution is a path represented by a branch of the tree *i.e.*, a sequence of trajectories from the root to a leaf node.

Hard Constraints

Each trajectory of a solution must satisfy the hard constraints. In this section five hard constraints are defined where some of the are previously discussed in Chapter 4 Section 4.1.3. Most of them are linked to one of the previously mentioned metrics. Therefore, a node $N_c = (P_c, \vartheta_c, \Gamma_c)$ can be added to the tree as a child of $N_p = (P_p, \vartheta_p, \Gamma_p)$ only if the candidate trajectory Λ_c $\Big((P_p, \vartheta_p), (P_c, \vartheta_c), \Gamma_c\Big)$ satisfies all the hard constraints defined as follows:

Inside constraint: the inside metric of Λ_c must be true which means the robot and its implement must remain inside the field during this trajectory.

Damage constraint: the work damage of Λ_c over trajectories constructed for all ancestor nodes of N_p must remain false.

Limited overlap constraint: this constraint disallows overlaps in the center of the field *i.e.*, outside the headlands and the gap-covering trajectories.

Global overlap constraint: The global overlap constraint imposes a threshold for the overall overlap area within authorized zones. When adding N_c to the tree, the overlap area of Λ_c with all its ancestors is computed, and added to a cumulative sum. In other words, this cumulative sum represents the total overlap area caused by all trajectories from root to N_c. If the cumulative overlap exceeds Δ_{global} the node N_c is discarded. Δ_{global} is the *global overlap threshold i.e.*, a percentage of the field area. If a dividing line is provided, this metric is applied on each sub-field independently.

Local loop constraint: at a more local level, to prevent undesirable local loops, when adding N_c to the tree, the overlap area between Λ_c and its ancestors is computed. If the overlap area exceeds Δ_{local} the node N_c is discarded. Δ_{local} is a percentage of the worked area of Λ_c named *local loop threshold*. In order to provide the robot a trajectory towards an access segment to exit the field, an exception is made to the local loop constraint when the objective in terms of coverage rate is satisfied.

Switch constraint: in case a dividing line is provided, this constraint implies that a switch between sub-fields is permitted if one of the following cases is met:

- the worked area since the last switch is more than Δ_{switch} *i.e.*, *switch threshold*
- for the current sub-field, the objective coverage rate is already met

MWD constraint: this constraint makes sure that the total length of successive trajectories *STRAIGHT_ON* or *DUBINS_ON* is higher than Δ_{mwd} (*i.e.*, *MWD threshold*). Therefore, after a *STRAIGHT_ON* or a *DUBINS_ON* trajectory, only another *STRAIGHT_ON* or *DUBINS_ON* trajectory is allowed if MWD is not met yet.

Tree Construction and Exploration

The tree construction and exploration can be detailed at two different levels:

- initialization
- node generation and exploration
  - cycles of traversals and half-turns
  - headland switch
  - exiting the field

The initialization outlines the process used to construct the first trajectories. The process of creating and adding back and forth trajectories to the tree is described as the node generation and exploration. It describes also how and in which condition a turning space is used to generate turns that guide the robot to an inner or gap-covering trajectory of a headland. Finally, it describes possible ways to generate a final trajectory towards an access segment to exit the field.

Initialization: the tree is first initialized by inserting an entrance $N_0 = (P_0, \vartheta_0, \Gamma_0)$ as the tree's root. The generation of the first branches of the tree, that illustrated in Fig. 5.3, is describes as follows:

- going straight and start working immediately: in this case, the first trajectory will be of type *GAP_OFF_ON* to point N_1. Afterwards, the next trajectory will have to be of type *STRAIGHT_ON*.
- crossing the headland to point N_2 with implement off. afterwards, the next trajectory will have to be of type *GAP_OFF_ON* then *STRAIGHT_ON*.

- turning immediately in the headland to points N_3 or N_4. afterwards, the next trajectory will have to be of type *GAP_OFF_ON* then *STRAIGHT_ON*.

These nodes are added to the tree as children of N_0 after being confirmed by the hard constraints. N_0 is the marked as visited. Further exploration is then conducted on unvisited nodes of the tree. Since gap-covering trajectories are mainly utilized to cover the unworked areas caused by transition trajectories, they are not employed for the initialization. Consequently they are not illustrated in Fig. 5.3.

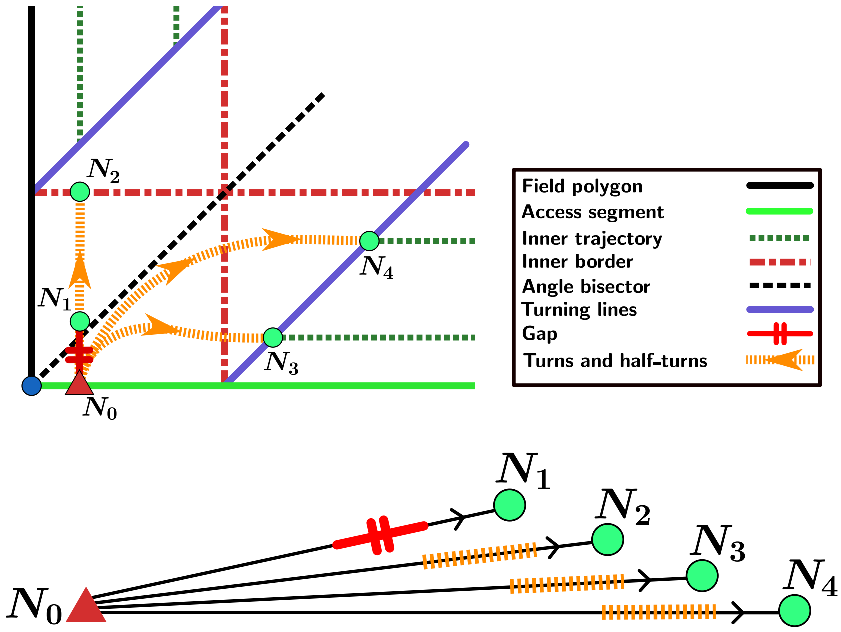


FIGURE 5.3: Trajectory and tree representation of the initialization. Orange dashed lines represent trajectories from N_0 toward its children

Node generation and exploration: a depth first exploration is carried out after initializing the tree. In general, new nodes are created for each unvisited node while respecting the trajectory sequence rules. After being validated by hard constraints, these new nodes are then added to tree as children of the corresponding node. Afterwards, one of these children is chosen for further node generation and exploration. If all generated nodes for an unvisited node violate at least one hard constraint, the unvisited node is removed from the tree and the exploration continues on its siblings.

For each unvisited node $N_p = (P_p, \vartheta_p, \Gamma_p)$, a ray r_p that starts from P_p with the direction ϑ_p is first generated. Next, the intersection of r_p with all headlands and turning spaces is calculated. Therefore, three following cases are possible:

- r_p intersects with the inner border of a headland that leads to generation of a cycle of traversal and half-turn
- r_p intersects with a turning space that leads to a headland switch
- r_p intersects with and access segment. In this case exiting the field can be planned

If the field polygon is divided into sub-polygons by a dividing line, only the inner borders and turning spaces that are inside the same sub-polygon as P_p are taken into account. This helps to reduce the number of unnecessary transitions between sub-polygons.

Cycle of traversal and half-turn: a cycle of traversal and half-turn refers to a specific sequence of trajectories. The sequence includes four different types of trajectories: *GAP_OFF_ON*, *STRAIGHT_ON*, *GAP_ON_OFF* and *DUBINS_OFF*. These movements can be executed in the main part of the field or within a headland along the boundary of the field.

In Fig. 5.4, ray r_p hits the inner border of the vertical headland on the right, denoted as destination inner border, at the position of N_{c3}. The path that goes from N_p to N_{c3}, via two intermediate nodes (N_{c1} and N_c2), are direct trajectories of types *GAP_OFF_ON*, *STRAIGHT_ON* and *GAP_ON_OFF* respectively. The possible turns from N_{c3} are then calculated to complete the cycle. To realize it, the location of N_{c4} and N_{c5} are first determined by finding the intersection of rays r_r and r_l with the destination inner border. These two rays are parallel to r_p with a spacing of w. The direction at N_{c4} and N_{c5} is the opposite direction of ϑ_p.

In the specific scenario when we are near to a neighboring headland and the trajectory is oblique to it, point N_{c6} is also computed as the intersection of rays r_r or r_l with the inner border of the neighbor headland. This gives more flexibility to the algorithm to continue the back and forth moves on the inner border of the neighbor headland when it arrives at a corner of the field.

Please note that to complete a cycle if a turn of type *DUBINS_OFF* does not satisfy the hard constraints, a *REEDS_OFF* turn is also examined. In other word, if a turn containing only forward moves is not possible then a turn containing both froward and reverse moves is also examined.

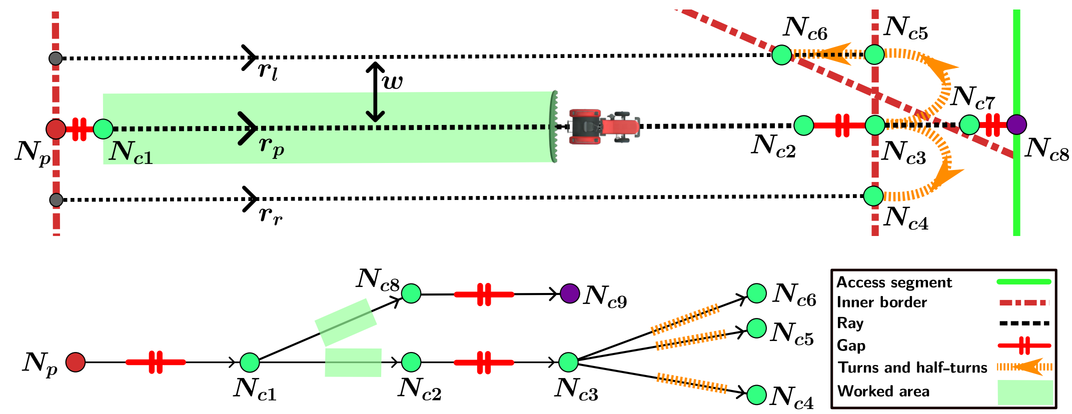


FIGURE 5.4: Trajectory and tree representation of a cycle of traversal and half-turn. N_p is the selected leaf for which the node generation is conducted. The purple point represents an exit node. For readability purposes, turning spaces, inner and gap-covering trajectories of headlands are not represented in this figure

headland switch: the process of traveling from one headland to an adjacent headland via a turning space is referred to as headland switch. It is used to cover the headlands that remained unworked. It is also useful for going from one sub-field to another, if the field is divided to sub-fields. Different possible headland switches are illustrated in Fig. 5.2.

As shown in Figs. 5.2a and 5.2c, N_{c3} located at the intersection of r_p and a turning space referred to as the selected turning space. Fig. 5.2c illustrates a complex case of headland switch where different ways to travel from the right headland to its two neighbor headlands are shown. Sub-figure *c* represents a complex case in which all possible ways to switch from the right headland to its two neighbor headlands are shown. Such complex cases only occur around a dividing line.

A headland switch contains also a sequence of a *GAP_OFF_ON*, *STRAIGHT_ON*, *GAP_ON_OFF* and *DUBINS_OFF* trajectories. The first three trajectories (trajectories from N_p to N_{c3})), are used to arrive at the selected turning space. Afterwards, all possible turns from N_{c3} to other headlands at destination nodes (N_{c4}, ..., N_{c10}) are computed. The location of destination nodes is calculated as the intersection of the inner and gap-covering trajectories of the target headlands with the corresponding turning spaces. The direction of a destination node matches the direction of the target inner or gap-covering trajectory. The trajectory type of these turns is first set to *DUBINS_OFF*. If they do not satisfy the hard constraints, their trajectory type is modified to *REEDS_OFF* and a turn generation with the Reeds-Shepp method is also examined. The number of destination nodes depends on the number of inner and gap-covering trajectories of the target headland/headlands.

Exiting the field: the possibility of exiting the field during a cycle of traversal and half-turn is examined, if r_p intersects with an access segment. In this case the exit path is a sequence of *GAP_OFF_ON*, *STRAIGHT_ON* and *GAP_ON_OFF* trajectories. An exit path is illustrated in Fig. 5.4 by the sequence of nodes $N_p \rightarrow N_{c1} \rightarrow N_{c7} \rightarrow N_{c8}$.

Another possible way to exit the field is during a headland switch. This case occur when a turning space gives an access to an access segment. This case is represented in Fig. 5.2c with the sequence of nodes $N_p \rightarrow N_{c1} \rightarrow N_{c2} \rightarrow N_{c3} \rightarrow N_{c11}$.

In these two cases, a leaf node is generated to determine the end of a coverage path. After computing the total worked area of a path, it is stored as a solution if the total worked area is greater or equal to a predefined coverage threshold Δ_{cov}. The output of the exploration algorithm is a set of valid solutions that is referred to as the *solution space*.

5.2.5 Similarity Check and Selection of optimal Solutions

The resulted solution space generally contains a large number of solutions. It is difficult for a user to verify all these solutions one by one to find the ideal solutions. To provide only a set of most pertinent solutions to the user, we first propose a cost function based on four metrics and then describe how we classify the solutions into families based on a similarity criterion and choose the best solution of each family.

The proposed cost function built as a weighted average sum of four following metrics:

- S_{cov}: coverage rate
- S_{ovl}: overlap rate
- S_{nwd}: non-working traveled distance
- S_{otm}: operation time

The coverage and overlap rates are calculated for a solution *i.e.*, a path, as the total worked and overlap areas of the path. The operation time for each path is calculated as follows:

$$S_{otm} = \frac{L_{on}}{V_{on}} + \frac{L_{off}}{V_{off}} + \frac{L_{gap}}{V_{gap}} \tag{5.3}$$

where L_{on}, L_{on} and L_{gap} are respectively the cumulative length of all trajectories during which the implement is on, of and in transition. Accordingly V_{on}, V_{off} and V_{gap} are the average speed of the robot when its implement is in on, off and in transition. Non-working traveled distance for the a path is then can be calculated as $L_{off} + L_{gap}$.

After being calculated, each metric $S \in \{S_{cov}, S_{ovl}, S_{nwd}, S_{otm}\}$ is normalized using the following equation :

$$S = \frac{S - S_{min}}{S_{max} - S_{min}} \tag{5.4}$$

where S_{min} and S_{max} represent the minimum and maximum value of the corresponding metric across all solutions in the solution space.

Following the computation and normalization of all metrics for each solution, a set of soft constraint ($\boldsymbol{C}$) and their weight ($\boldsymbol{W}$) can be defined as follows:

$$\boldsymbol{C} = (1 - S_{cov}\ S_{ovl}\ S_{nwd}\ S_{otm}) \tag{5.5}$$

$$\boldsymbol{W} = (W_{cov}\ W_{ovl}\ W_{nwd}\ W_{otm}) \tag{5.6}$$

where W_{cov}, W_{ovl}, W_{otm} and W_{nwd} are weights given as input for the corresponding soft constraint. Therefore, the final cost of each solution can be computed as follows:

$$f = \frac{\boldsymbol{C}\boldsymbol{W}^{\intercal}}{W_{cov} + W_{ovl} + W_{otm} + W_{nwd}} \tag{5.7}$$

The produced solution space could include a number of solutions that are quite similar to one another and differ just by one or two turns. To eliminate the similarity while preserving a variety of propositions, the solution space is first divided into families of solutions based on a similarity criterion. From each family the solution that has the lowest cost is proposed to the user.

The similarity criteria is based on the *general direction* of the solutions, which corresponds to the primary direction of the back and forth trajectories *i.e.*, the one that is utilized the most in the main part of the field. The general direction is calculated for each sub-polygon if the field polygon is split into sub-polygons. As a result, if two solutions have the same general direction(s), the are regarded as similar and added to the same family. Finally, the best solution of each family is proposed to the user and the *most optimal solution* is highlighted. The maximum number of families is determined by the number of potential general directions, which is equal to the number of field boundaries.

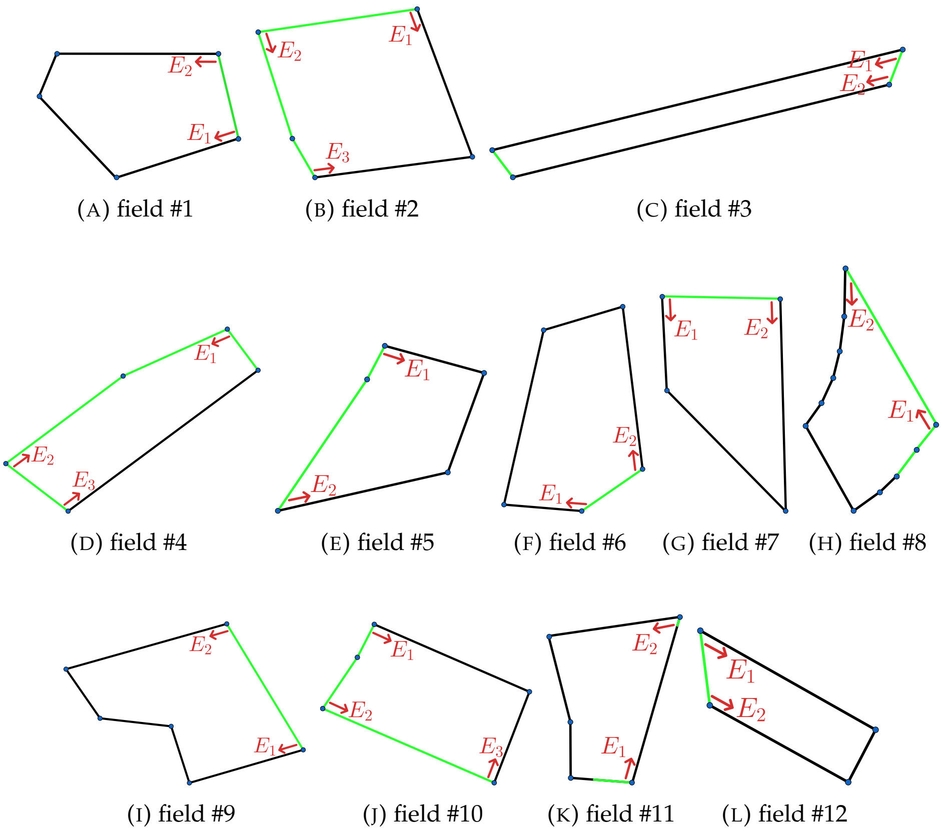
5.3 Results and Discussion

5.3.1 Experiment

To implement this approach, a program written in C++ was developed that includes a GUI for setting input parameters and visualizing the output solutions. Through the use of an OpenMP [104] implementation, all processes can operate in parallel to speed up calculations. An Intel Xeon(R) W-2135 CPU @ 3.70GHz × 12 with 32GB RAM were used to run the program.

The dataset proposed in Chapter 3 Section 3.3, were used to evaluate the presented approach. Twenty of these fields (Fields #1 - #20), referred to as simple fields, have been used with no field decomposition. Ten other fields (Fields #21 - #30), referred to as complex fields, for which at least two different dividing lines were provided to try different field decompositions.

Fig.5.5 and Fig.5.6 respectively show simple fields and complex fields. The field polygons are represented by black and green line segments, with the green ones indicating access segments. The entrances are depicted by red arrows, and in Figure 5.6, dividing lines are shown in brown. It is important to note that during each exploration, only one of the dividing lines (if provided) and one of entrances will be used. Therefore, multiple explorations may be performed for a given field, depending on the number of provided dividing lines and entrances, with all possible combinations of them being considered.

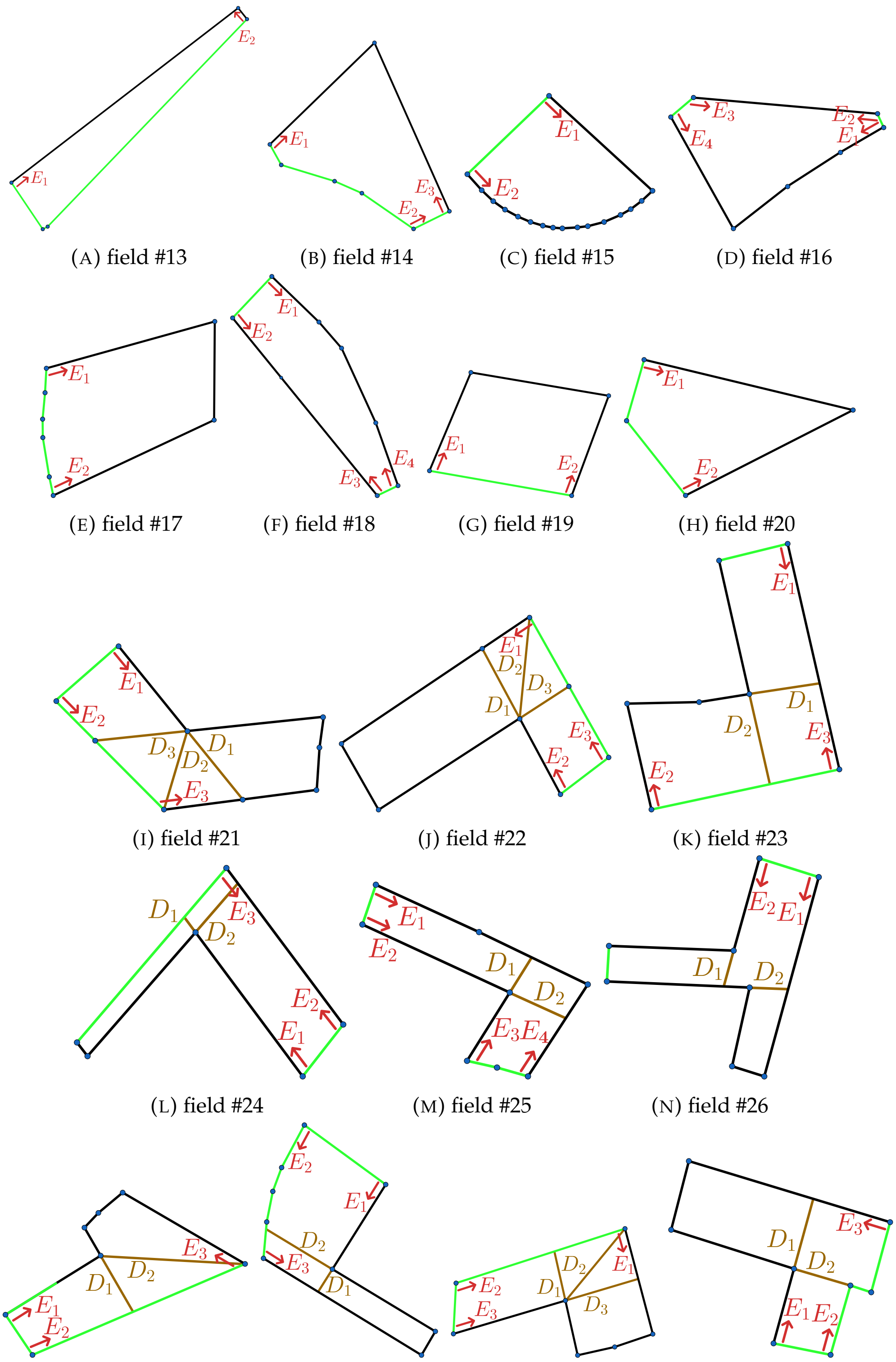


(A) field #1 (B) field #2 (C) field #3

(D) field #4 (E) field #5 (F) field #6 (G) field #7 (H) field #8

(I) field #9 (J) field #10 (K) field #11 (L) field #12

FIGURE 5.5: Simple fields: 2D polygons, access segments and entrances. Part 1/2



(A) field #13 (B) field #14 (C) field #15 (D) field #16

(E) field #17 (F) field #18 (G) field #19 (H) field #20

(I) field #21 (J) field #22 (K) field #23

(L) field #24 (M) field #25 (N) field #26

(O) field #27 (P) field #28 (Q) field #29 (R) field #30

FIGURE 5.6: Complex fields: 2D polygons, access segments, entrances and dividing lines. Part 2/2

The approach was applied on each field with the variety of given dividing lines (if provided) and entrances. Our approach identified several families of solutions for each field and the best solution of each family is selected. One of them that had the lowest cost was represented as the most optimal solution and others were represented as alternatives to the user. All selected paths were compared to the ground truth, by visually comparing them to the reference satellite image of the field, where the tracks are visible. The solution that had the most visual similarity to the reference satellite image is referred to as the *most similar solution*.

To determine the most similar solution, the general direction of back and forth moves in the main part of the field as well as how a field was divided into sub-fields were verified. We considered a farmer as an expert who is completely familiar with his/her own field. Consequently, the path that was chosen by the farmer can be used as a reference.

From satellite images, it was simple to identify the headlands as well as the general direction of back and forth moves. However, determining the number of trajectories within a headland as well as the parameters of the vehicle and its implement was almost impossible. Therefor, these parameters were guessed at best for each field, and their averages were then applied to all fields. Table 5.1 summarizes the parameters used in our approach.

Parameter	Description	Value	Parameter	Description	Value
w	working width	$3m$	Δ_{cov}	coverage threshold	97%
γ_{on}	minimum turning radius - implement on	$15m$	Δ_{global}	global overlap threshold	5%
γ_{off}	minimum turning radius - implement off	$1.5m$	Δ_{local}	local loop threshold	95%
V_{on}	average speed - implement on	$3.5m/s$	Δ_{switch}	switch threshold	93%
V_{gap}	average speed - implement transition	$2.5m/s$	Δ_{mwd}	minimum working distance threshold	$8m$
V_{off}	average speed - implement off	$1.5m/s$	W_{cov}	weight of S_{cov}	0.6
ℓ_t	transition trajectory length	$2m$	W_{ovl}	weight of S_{ovl}	0.1
ℓ_o	robot-implement offset	$2m$	W_{nwd}	weight of S_{nwd}	0.2
p	number of inner trajectories of headlands	2	W_{otm}	weight of S_{otm}	0.1

TABLE 5.1: The input and parameters of the proposed approach

In the reminder of this chapter an analytical result over the entire dataset is provided. A subset of the dataset is then chosen to illustrate the results and highlight some intriguing features of our approach without weighting down this chapter.

5.3.2 Analysis of The Results

Area (ha)		Coverage		Overlap		Single exploration time (s)		Selection time (s)	
Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
4.87	2.82	98.69%	0.62%	3.00%	1.39%	64.70	81.21	1.71	2.71

(A) Simple fields

Area (ha)		Coverage		Overlap		Single exploration time (s)		Selection time (s)	
Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
4.69	2.41	98.23%	0.58%	3.09%	1.17%	617.20	610.02	10.60	15.73

(B) Complex fields

TABLE 5.2: Numerical results of the evaluation

Table 5.2 summarizes the average results obtained from the evaluation. The results indicate that our approach achieved a coverage rate of 98.69% with an average overlap of 3.00% for simple fields, while for complex fields the average coverage rate was 98.23% with an average overlap of 3.09%. The success of our approach can be attributed to its ability to handle curved edges and cover headlands efficiently.

In 85% of the simple fields, the solution that was most similar to the satellite images was also identified as the most optimal solution. However, in the remaining 15% of cases, although the most similar solution was found by our approach, it did not meet the predefined criteria for being considered the most optimal solution.

In the case of complex fields, the most optimal solutions were identical to the most similar solutions in 70% of the cases. In 10% of the cases, the most similar solution was found but not considered the most optimal based on the predefined criteria. However, in the remaining 20% of the cases, no similar solution was found. We found two potential explanations. First, as mentioned earlier, we guessed the parameters at best. It is possible that the values we used for the thresholds and vehicle parameters do not match those used by the farmer. Modifying these parameters, such as lowering the coverage threshold Δ_{cov} and/or raising the global overlap threshold Δ_{global}, could result in more solutions and potentially increase the likelihood of finding the most similar solution. Another reason could be that the farmer might have used certain constraints or preferences that were not taken into account in our approach. For example, the farmer may have used visual clues to help guide their work, which could have influenced the trajectory choices, but we did not consider this as a relevant factor for a robot.

To clarify, the number of explorations conducted for a field is determined by the selected entrances and provided dividing lines. Therefore, multiple explorations were carried out for each field. As a result, the single exploration time shown in Table 5.2 represents the average and standard deviation of all explorations performed respectively on simple and complex fields.

To provide an overview of the results, here we show the results of a subset of fields from the dataset, including four simple (#6, #7, #8 and #9) and two complex fields (#22 and #24).

Field	Exploration			Similarity check and selection of optimal solutions							
	total	Solutions	time (s)	time (s)	$CW^{\intercal}$				f	Coverage	Overlap
#6	2	9459	147.30	0.24	0.000	0.081	0.038	0.049	0.168	98.43%	4.42%
#7	2	2511	22.32	0.07	0.000	0.036	0.022	0.044	0.101	98.58%	1.80%
#8	2	1258	42.16	0.06	0.000	0.048	0.012	0.023	0.083	98.35%	2.66%
#9	2	186	75.4849	0.01	0.000	0.099	0.073	0.085	0.256	97.77%	4.87%
#22	12	82	599.12	0.00	0.000	0.048	0.010	0.014	0.072	97.64%	3.61%
#24	9	280782	2848.74	12.32	0.056	0.036	0.011	0.017	0.121	98.91%	2.01%

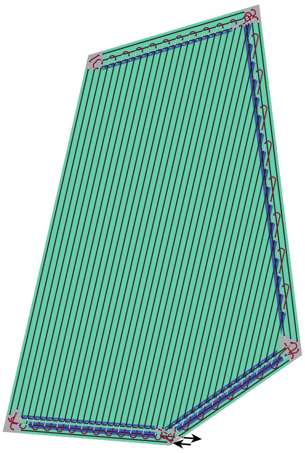
(A) Most optimal solution

Field	Exploration			Similarity check and selection of optimal solutions							
	total	Solutions	time (s)	time (s)	$CW^{\intercal}$				f	Coverage	Overlap
#6	2	9459	147.30	0.24	0.083	0.050	0.059	0.045	0.237	98.23%	3.48%
#22	12	82	599.12	0.00	0.398	0.038	0.013	0.012	0.461	97.22%	3.35%

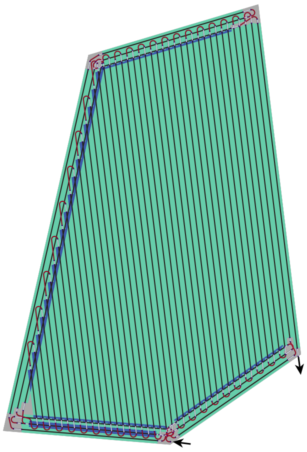
(B) Most similar solution

TABLE 5.3: Numerical results of six fields from the dataset. The most optimal and most similar solutions are the same for Fields #7, #8, #9 and #24

The obtained results on the six fields are illustrated in Figs. 5.7 and 5.8, and the numerical results are summarized in Table 5.3. The figures demonstrate that our approach was effective in covering unworked areas caused by half-turns and gaps, and was able to intelligently select the headlands to perform half-turns. In addition, Fig. 5.7f illustrates our approach's ability to handle curved trajectories within headlands.

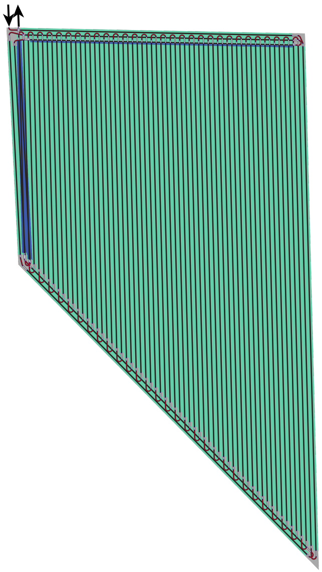


(A) #6 - most optimal solution



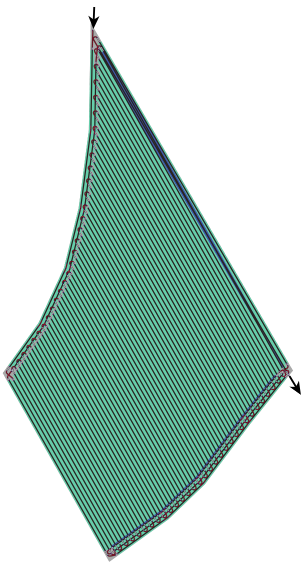
(B) #6 - most similar solution

(C) #6 - satellite image



(D) #7 - most optimal & similar solution

(E) #7 - satellite image



(F) #8 - most optimal & similar solution

(G) #8 - satellite image

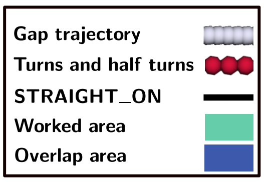
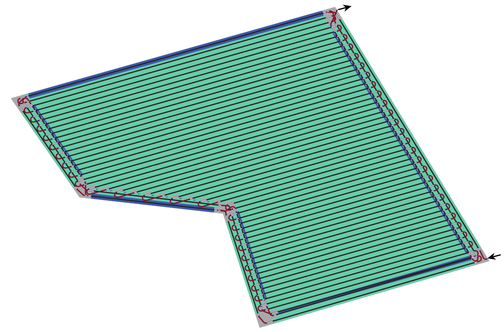
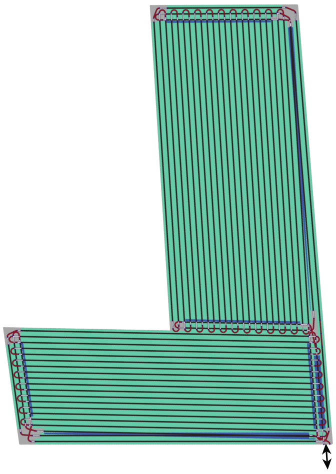


FIGURE 5.7: Obtained results on six fields from the dataset and their reference satellite image. The black arrows indicate where the robot enters and exits. Part 1/2

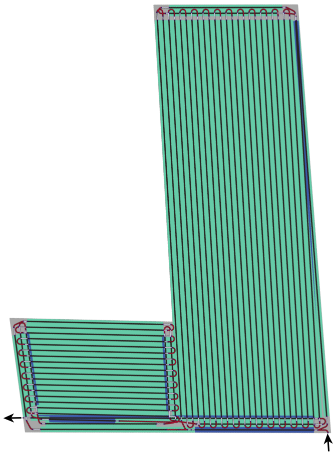


(A) #9 - most optimal & similar solution

(B) #9 - satellite image

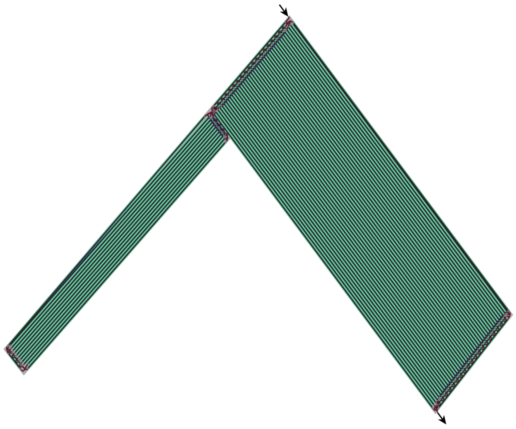


(C) #22 - most optimal solution



(D) #22 - most similar solution

(E) #22 - satellite image



(F) #24 - most optimal solution

(G) #24 - satellite image

FIGURE 5.8: Obtained results on six fields from the dataset and their reference satellite image. The black arrows indicate where the robot enters and exits. Part 2/2

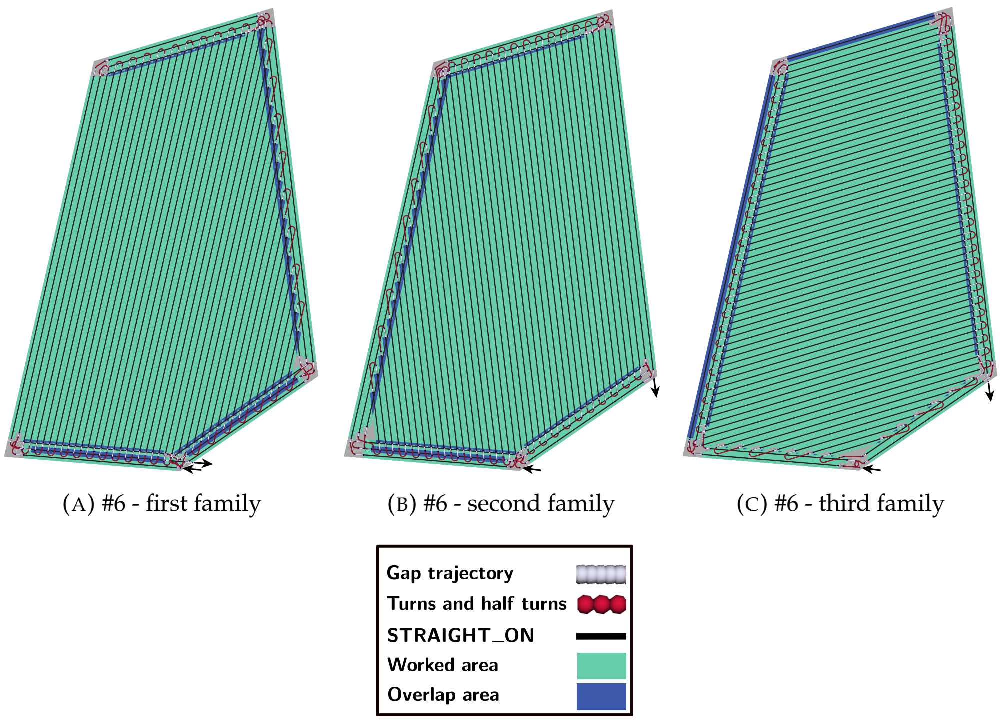
Our approach for complex fields not only found the most optimal solution, but also determined the optimal dividing line for field decomposition. Figs. 5.8f, 5.8c, and 5.8d represent the results obtained for the complex fields.

Generating the solution space for Fields #22 and #24 took almost 47 and 10 minutes, respectively. This is because for these fields, nine and twelve successive explorations had to be performed to account for all the combinations of entries and dividing lines. To speed up the process for complex fields, a smart polygon decomposition that considers the agricultural use case, such as the geometry of the field and the inclination, could be computed in a preliminary step. This approach would avoid unnecessary explorations and accelerate the process.

The presented approach's one-step strategy was able to produce solutions that cover multiple sub-fields in a single path, as shown in Figs. 5.8f, 5.8g, and 5.8c. The path covers the main part and some parts of the headlands of the first sub-field, then proceeds to cover the second sub-field completely, and finally returns to cover the remaining headlands of the first sub-field and exit the field. This kind of solution is not possible with classic approaches that use two sequential steps (CMC and AVRP), highlighting the interest of our approach's one-step strategy. The following sections explore other noteworthy features of the approach.

Interest of Grouping Solutions into Families

Fig. 5.9 depicts the most optimal solution from three distinct solution families for Field #6, while Table 5.4 provides a summary of the numerical outcomes for each solution. It is apparent that the differences between these solutions are rather small according to the numerical results. However, as seen in Fig.5.9, the solutions appear distinct in terms of their general direction. While two solutions share a similar entry and exit, the first solution uses a different exit. This highlights the interest of clustering the solutions into separate families to present a variety of good solutions to the farmer.



(A) #6 - first family (B) #6 - second family (C) #6 - third family

FIGURE 5.9: Most optimal solution from different families for Field #6. The black arrows indicate where the robot enters and exits

Field	Figure	CW^{T}				f	Coverage	Overlap
#6	5.9a	0.000	0.081	0.038	0.049	0.168	98.43%	4.42%
#6	5.9b	0.083	0.050	0.059	0.045	0.237	98.23%	3.48%
#6	5.9c	0.312	0.099	0.181	0.094	0.685	97.68%	4.95%

TABLE 5.4: Numerical results for different families for Field #6

Benefit of Multiple Entrances

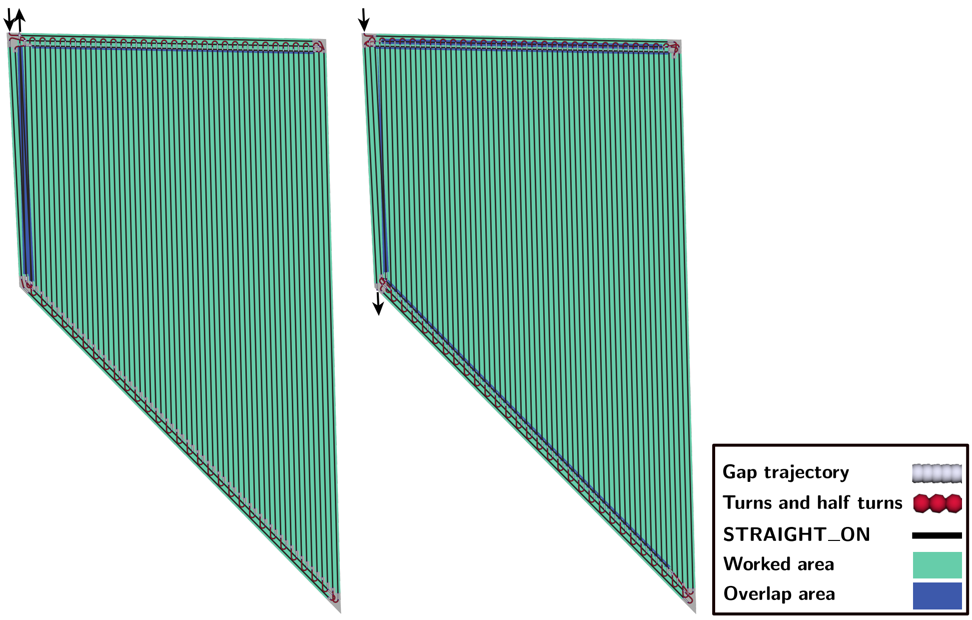
Table 5.5 summarizes the numerical result for each entrance of Field #9 separately. The most optimal solution was through E_1. Considering E_2 as the entrance, our approach was also capable to find some acceptable results. However they were not as good as the one found while considering E_1. That means, taking into account more entrances enhances the probability of finding a better result. However it also increases the exploration time. In this specific case, taking E_1 into account in addition to E_2 increased the coverage rate by 0.71% and nearly doubled the exploration time.

Field	Entrance	Exploration		Similarity check and selection of optimal solutions						
		Solutions	time (s)	$CW^{\intercal}$				f	Coverage	Overlap
#9	E_1	168	39.79	0.000	0.099	0.073	0.085	0.256	97.77%	4.87%
#9	E_2	18	35.69	0.558	0.022	0.000	0.000	0.580	97.06%	3.09%

TABLE 5.5: Numerical results on Field #9 for each of its entrances

Impact of Field Accessibility

The optimal solutions for two scenarios on Field #7 are shown in Fig.5.10: the first scenario assumes that the field is only accessible from its upper edge, which reflects the real-world situation, while the second scenario assumes that the field is accessible from all its edges, which is an oversimplification. Table 5.6 presents the numerical results for each scenario. Although we obtained a better solution according to the defined criteria with the second scenario, in reality, this solution would not be feasible as the robot would enter the neighbor's field from the bottom. This highlights that an inaccurate description of the accessibility of a field may result in an unfeasible solution that can damage the robot or the neighbor's field.



(A) #7 - one accessible edge (B) #7 - all edges are accessible

FIGURE 5.10: Most optimal solutions for Field #7 while considering it is only accessible from its upper edge (a) or from all its edges (b). The black arrows indicate where the robot enters and exits

Field	accessible edges	Exploration		Similarity check and selection of optimal solutions		
		Solutions	time (s)	time (s)	Coverage	Overlap
#7	1 edge	2511	22.32	0.07	98.58%	1.80%
#7	all edges	10137	35.4729	0.30	98.61%	0.88%

TABLE 5.6: Comparison of results over Field #7 while considering it is accessible only by its upper edge (real world scenario) and all its edges

Forward and Reverse Half-Turns

We conducted an additional test on Field #7 to showcase our approach's capacity to perform half-turns other than U-turns. In this test, we kept all the parameters identical to the previous tests, except for γ_{off}, which was set to 2 meters. As shown in Fig.5.11, our approach selected to execute half-turns utilizing reverse movements on the left side of the field, while no reverse moves were necessary on the right side. Table 5.7 presents the numerical outcomes for this experiment.

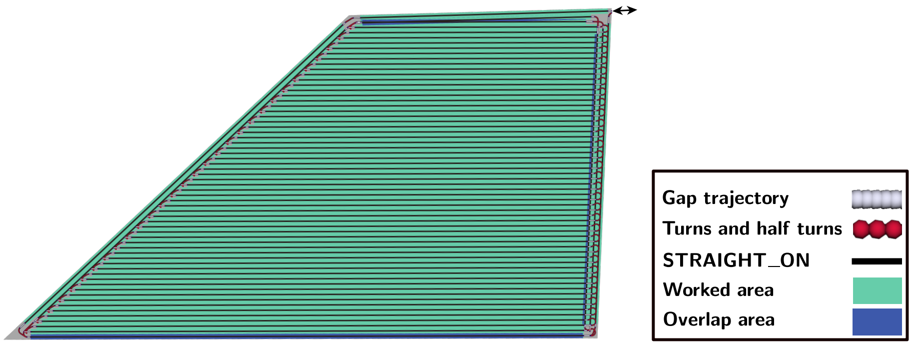


FIGURE 5.11: Most optimal solutions for Field #7 when γ_{off} was set to $2m$. The black arrows indicate where the robot enters and exits

Field	Exploration			Similarity check and selection of optimal solutions							
	total	Solutions	time (s)	time (s)	CW^T				f	Coverage	Overlap
#7	2	135	7.70	0.02	0.002	0.062	0.059	0.065	0.187	98.46%	2.96%

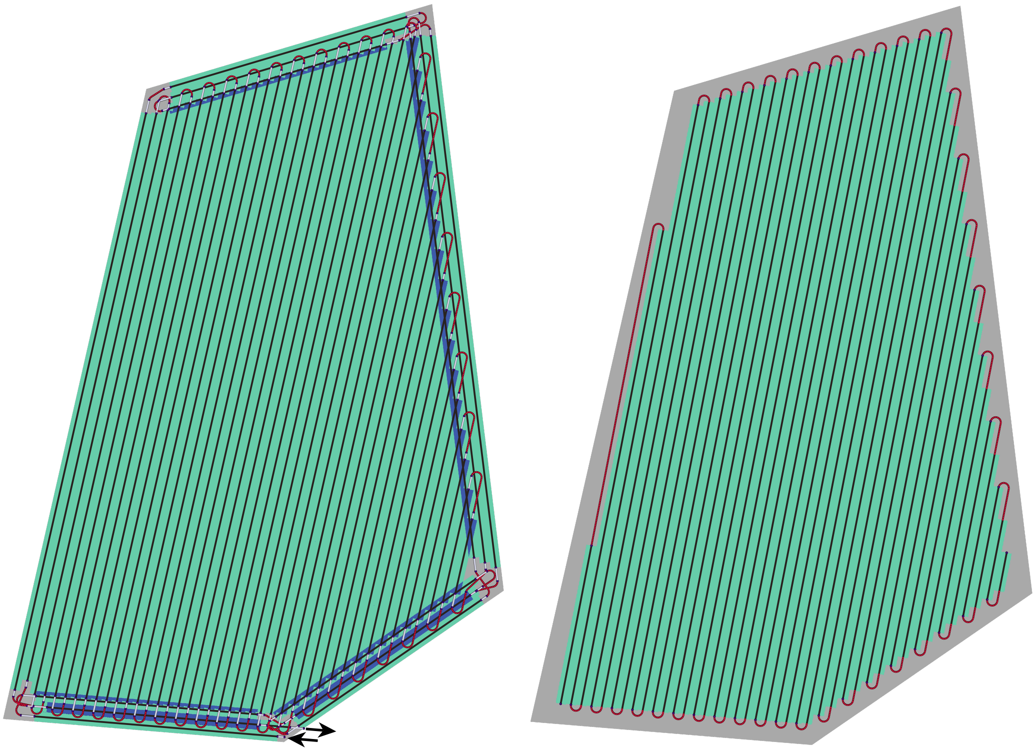
TABLE 5.7: Numerical results of Field #7 while γ_{off} was set to $2m$

Discussion on Comparative Results

Due to the significant variations in the nature of different approaches, the variety of constraints considered or not, and the absence of a standardized dataset, it is challenging, if not impossible, to make a direct comparison between our approach and existing methods in the literature. To highlight the difficulty of comparing our approach to previous studies in the literature, we attempted to compare it with an open-source algorithm named *Fields2Cover (F2C)*, which was proposed by Mier, Valente, and Bruin [87]. F2C takes as input the field polygon, the width of headlands ($p * w$), the minimum turning radius of the robot (γ_{off}), and one objective function chosen from three options: minimizing the total path length, minimizing the number of half-turns in headlands, and maximizing the coverage rate. The objective function chosen for this comparison was to minimize the number of half-turns. It should be noted that our algorithm simultaneously addresses these three constraints under different terminology. It should also be mentioned that F2C does not include headlands in its coverage computation.

We compared the performance of our approach and F2C on six fields (#6, #7, #8, #9, #22, #24) by analyzing their computation times, number of half-turns, coverage, and overlap. Fig.5.12 provides a visual comparison of the results obtained by the two approaches for Field #6. Our method achieved a higher coverage rate than F2C by an average of 12.42%. However, F2C had lower overlap area than our approach, with an average overlap of 0% compared to 3.23% for our method. The computational time of F2C was significantly lower than our approach, averaging 1.52*s* for F2C and 624.6*s* for ours. F2C also had slightly fewer half-turns than our approach, with an average of 1.5 less half-turns.

However, it should be noted that the comparison of the two approaches must be viewed in the context of headland coverage. As shown in Figs. 5.12, our method's overlaps and extra turns are primarily due to headland coverage, while the absence of headland coverage in F2C leads to fewer turns, no overlaps, and ultimately less computational time. Moreover, our method naturally provides better coverage as it takes into account headlands.



(A) #6 - Our approach (B) #6 - F2C

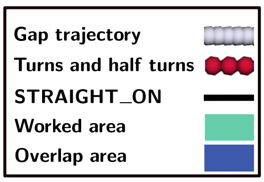


FIGURE 5.12: Result obtained on field #6 from the dataset using our approach and F2C

The general direction of tracks in our approach is aligned with one of the borders of the field, which is not the case with F2C. This may partly explain the lower coverage achieved by F2C. Furthermore, the approach presented by F2C will still necessitate coverage of an extra circular headland, ultimately leading to increased overlaps. Our method already considers the headland coverage that is not fully circular, resulting in less required work.

It is worth noting that F2C did not consider transition states of the implement in its computation. As shown in Fig.5.12b, the right trajectory is too short and cannot be achieved practically due to the minimal trajectory length required to activate and deactivate the implement. This will ultimately result in a loss of coverage. These examples highlight the challenge of comparing various approaches.

To address the limitations of comparing our approach to existing literature, we opted to visually compare our results to satellite images as ground truth. Nonetheless, this choice comes with some limitations, mainly due to variations in machinery and the different preferences and constraints applicable to a human operator versus a robot. Despite these challenges, we were able to show in this study that our approach can provide optimal coverage paths, comparable to current practices, suitable for various configurations and use cases, fully parameterizable, and achievable in a reasonable amount of time.

5.4 Conclusion

In this chapter, we proposed a new CCPP approach that uses tree exploration to generate an optimal path starting from an entrance location, covering the field and headlands, and ending at an accessible edge of the field. To achieve this, it first conducted one or more explorations while considering multiple entrances and dividing lines (if provided). The result of this exploration was a solution space that contained all possible solutions satisfying a set of hard constraints. Finally, a similarity check and selection of optimal solutions was applied to extract a variety of most optimal paths without redundancies, by minimizing a weighted average cost function of the soft constraints. The main goal of this approach was to maximize worked area while minimizing overlaps, non-working traveled distance, and operation time.

This study revealed that exploring multiple entrances and/or dividing lines can increase the likelihood of finding better solutions. Additionally, it highlighted the importance of considering the actual accessibility of the field when generating optimal paths.

Currently, our approach is able to generate paths where working trajectories are sequential. It would be intriguing to explore other coverage patterns, such as skipping adjacent rows. Another limitation of our approach is its reliance on dividing lines as input to simplify complex field polygons into more manageable sub-polygons. This process can be computationally expensive, as multiple dividing lines must be tested to identify the one that leads to a better solution.

As we strive for an even more efficient and versatile CCPP approach, in the next chapter, we will introduce two extensions to the presented approach. The first extension will incorporate row-skip patterns into the path generation process. The second extension will focus on automating the field decomposition step and adapting it to various field characteristics using a deep learning-based method.

In pursuit of a more efficient and versatile CCPP approach, the next chapter will introduce two extensions to the approach presented in this chapter. The first extension aims at incorporating row-skip patterns into the path generation process, enhancing its adaptability and performance. The second extension will concentrate on automating the field decomposition step, tailoring it to various field characteristics through the application of a deep learning-based method.

CHAPTER 6

Extensions: Row-Skip Pattern and DL-based Field Decomposition

In the previous chapter, we presented a CCPP approach for autonomous agricultural robots that generates an optimal path to cover a field and its headlands while minimizing overlaps, non-working path length, and operation time. While the original CCPP approach, hereafter referred to as *O-CCPP*, demonstrates promising results for various field shapes, its performance can be further enhanced to be more efficient.

As a reminder, the O-CCPP method, detailed in Chapter 5, comprises three main steps: preprocessing, exploration, and selection of optimal solutions. In the preprocessing step, headlands, trajectories within headlands, and turning spaces were determined. Additionally, entry points were determined by considering the access segments. The exploration algorithm found every potential solution that adhered to predefined constraints and stored them in a solution space. Each solution represented a path, or sequence of trajectories, that began at an entrance, covered the field completely, and ended at an access segment. Finally, in the last step, the cost of each solution was calculated, and the ones with the lowest cost were selected.

In this chapter, we introduce two extensions to the O-CCPP approach to increase its versatility and efficiency. The first extension involves incorporating row-skip patterns into the path generation process, expanding its applicability and enhancing its performance. The second extension aims at automating the field decomposition step and adapting it to various field characteristics using a Deep Learning-based method.

In the remainder of this chapter, we will provide a detailed description of these extensions, exploring how they improve upon the O-CCPP approach and contribute to a more robust and adaptive path planning process for agricultural robots.

6.1 Row-Skip Coverage Pattern

Skipping rows creates a pattern that can decrease the size of headlands, which are often less-productive areas due to soil compaction. This technique has the potential to increase the efficiency of field coverage. However, just as a driver operating a vehicle would prefer to avoid reversing, he or she would also prefer to avoid skipping rows because of the lack of visual cues to take the right tracks. On the contrary, for an automated system or robot equipped with a guidance system, row-skipping is a perfectly viable option.

Several studies have investigated other coverage path patterns beyond sequential patterns. For instance, Jeon et al. [65] examined sequential and gathering patterns to connect the parallel tracks in headlands. In the gathering pattern, the distance between two successive tracks is approximately half the field's width. They integrated in their approach a headland and boundary corner turning methods for efficiently covering headland and corners. However, they considered that the robot can perform rough turns even when its implement is in contact with the ground. This is an oversimplification that might damage the machinery. It might also lead to an overestimation of the coverage rate.

Another approach was proposed by Mier, Valente, and Bruin [87], which can generate different route patterns such as sequential, row-skip, and spiral pattern. The spiral pattern is a variation of the row-skip pattern that is capable of skipping several tracks instead of only one track. However, their approach does not account for headland coverage.

In this section, we introduce a row-skip pattern approach that is able to generate a coverage path with a row-skip pattern. Our approach aims at maximizing the covered area while minimizing overlaps, non-working path length, number of turns containing reverse moves, and overall travel time. It covers the headlands automatically while considering the geometry of both the robot and its implement.

6.1.1 Methodology

The Row-Skip CCPP also consists of three main parts: preprocessing, exploration, and selection of optimal solutions. The preprocessing part is exactly the same as the O-CCPP approach, where the headlands containing p inner trajectories and one gap-covering trajectory, and turning space are generated based on the working width and the robot parameters including γ_{off}, ℓ_o. It also determines entry points while considering access segments of the field. For more detail about this process we refer readers to Chapter 5, Section 5.2.2.

In this variant, the exploration step has been modified to generate solutions with a row-skip pattern instead of a sequential pattern. Two consecutive working trajectories in the main part of the field are not adjacent, except when they are located close to a border of the field, allowing to go back and cover the previously skipped tracks. Finally, the selection of the optimal solution is also slightly modified to take into account the number of turns containing reverse moves, in addition to the other soft constraints: coverage rate, overlap rate, non-working traveled distance, and operation time.

Exploration Algorithm

The exploration algorithm takes as input the output of the preprocessing step, a set of hard constraints, and several input parameters including γ_{on}, γ_{off}, ℓ_t and ℓ_o. All input parameters are summarized in Table 6.1.

The exploration algorithm builds a tree of potential trajectory sequences, considering three scenarios: generation of a traversal cycle and half-turn, headland switch, and field exit. Most scenarios remain unchanged from the O-CCPP approach. For

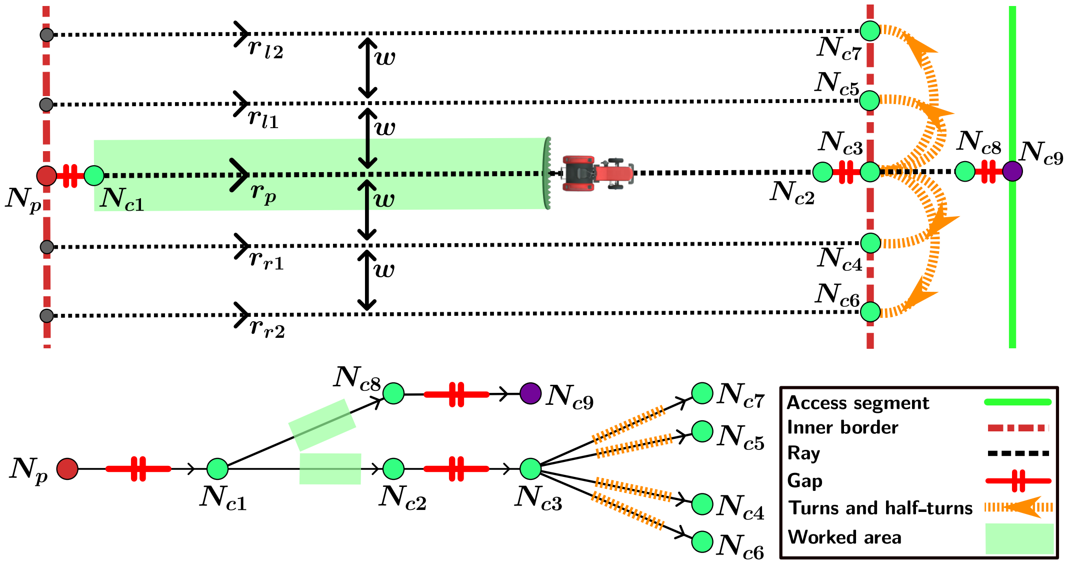


FIGURE 6.1: Trajectory and tree representation of back-and-forth moves. The purple point represents an exit node. For readability purposes, only inner border of two headlands are represented

details, refer to Chapter 5, Section 5.2.4. This chapter highlights modifications to the traversal cycle and half-turn generation.

As illustrated in Fig.6.1, for each leaf node N_p a ray r_p is constructed based on its location and the direction of the robot at this location. The intersection of ray r_p with an inner border of a headland leads to the generation of a cycle of traversal and half-turn.

If the ray r_p intersects with an inner border of a headland, a sequence of three trajectories is generated to reach the corresponding inner border. As shown in Fig.6.1, this sequence includes a gap trajectory from N_p to N_{c1} to lower the implement, a working trajectory from N_{c1} to N_{c2}, and an another gap trajectory from N_{c2} to N_{c3} to raise the implement. Afterwards, Four side rays r_{r1}, r_{r2}, r_{l1}, and r_{l2} are then defined, which are parallel to r_p, with two on each side of r_p, and are located at a respective distance of w and $2w$ from r_p. The closest rays r_{r1} and r_{l1} correspond to the adjacent tracks, while rays r_{r2} and r_{l2} correspond to a track-skipping move. If the intersections of the most distant rays r_{r2} and r_{l2} with an inner border both exist, two corresponding nodes N_{c6} and N_{c7} are generated. Otherwise, nodes N_{c4} and N_{c5} are also generated at the intersection with r_{r1} and r_{l1}. Finally, a turn from N_{c3} to each of these new nodes is generated and added to the tree after being validated by hard constraints.

If there is no intersection between either r_{r2} or r_{l2} and an inner border, it indicates that the robot is close to the field edges. In this scenario, the robot must take an adjacent track to cover all of the previously skipped rows.

This process, along with other scenarios of the exploration *i.e.*, headland switch and exiting the field, is repeated for all unvisited nodes. Any sequences of trajectories that achieve a certain level of coverage rate are stored in the solution space.

Selection of the Optimal Solution

After storing all possible solutions in a solution space, similar to the O-CCPP, coverage rate S_{cov}, overlap rate S_{ovl}, non-working distance S_{nwd} and operation time S_{otm}

are computed for each solution. However, a new metric, the number of turns with reverse moves S_{rvs}, is also considered in addition to the previous metrics. Afterwards all these metrics are normalized by following equation:

$$S = \frac{S - S_{min}}{S_{max} - S_{min}} \tag{6.1}$$

where S_{min} and S_{max} represent the minimum and maximum value of the corresponding metric over all solutions of the solution space.

The soft constraints are integrated by defining $\mathbf{C} = (1 - S_{cov}\ S_{ovl}\ S_{nwd}\ S_{otm}\ S_{rvs})$ and $\mathbf{W} = (W_{cov}\ W_{ovl}\ W_{nwd}\ W_{otm}\ W_{rvs})$, where W_{cov}, W_{ovl}, W_{otm}, W_{nwd}, and W_{rvs} are weights specified as input. The final cost of each solution is obtained by applying Equation (6.2).

$$f = \frac{\mathbf{C}\mathbf{W}^{\intercal}}{W_{cov} + W_{ovl} + W_{otm} + W_{nwd} + W_{rvs}} \tag{6.2}$$

The final solution chosen by our approach is the one with the lowest cost, which is determined by a combination of different soft constraints and their corresponding weights. By adjusting the weights assigned to each soft constraint, the algorithm can be customized to prioritize certain aspects of the path planning process, such as reducing overlap or minimizing non-working distance. This allows users to tailor the algorithm to their specific needs and requirements.

6.1.2 Results

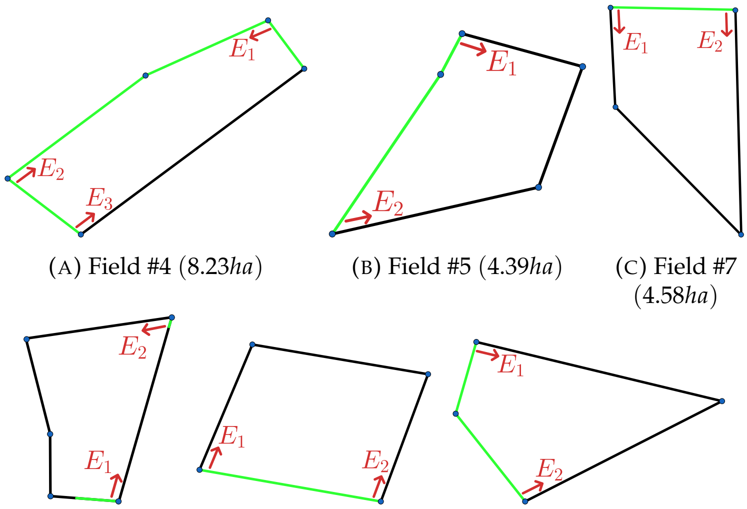
To evaluate the performance of the proposed Row-Skip CCPP algorithm (*RS-CCPP*) in comparison to the O-CCPP, six fields were selected from the dataset introduced in Chapter 3 (#4, #5, #7, #11, #19, and #20). These fields vary in size from 4.34 to 8.23 hectares, as shown in Fig.6.2. The implementation and execution of both approaches were conducted on an Intel Xeon(R) W-2135 CPU @ 3.70GHz × 12 with 32GB RAM.

To obtain a fair value for the cost function, the solution spaces acquired by the exploration algorithms of both approaches were combined and the selection method was applied on the combined solution space. This allowed for a direct comparison of the final cost of the most optimal result found by each approach.

Parameter	Description	Value	Parameter	Description	Value
w	working width	$3m$	Δ_{global}	global overlap threshold	5%
γ_{on}	minimum turning radius - implement on	$15m$	Δ_{local}	local loop threshold	95%
V_{on}	average speed - implement on	$3.5m/s$	Δ_{mwd}	minimum working distance threshold	$4m$
V_{gap}	average speed - implement transition	$2.5m/s$	W_{cov}	weight of S_{cov}	0.65
V_{off}	average speed - implement off	$1.5m/s$	W_{ovl}	weight of S_{ovl}	0.15
ℓ_t	transition trajectory length	$2m$	W_{nwd}	weight of S_{nwd}	0.05
ℓ_o	robot-implement offset	$2m$	W_{otm}	weight of S_{otm}	0.05
Δ_{cov}	coverage threshold	96%	W_{rvs}	weight of S_{rvs}	0.10

TABLE 6.1: The input and parameters of the proposed approach

To compare RS-CCPP and O-CCPP, we investigated the effect of two distinct minimum turning radius values for the robot when its implement is off (γ_{off}): $2m$ and $3m$, while keeping other parameters constant. The number of trajectories within a headland was set to two ($p = 2$), and other parameters are provided in Table 6.1. To obtain a comprehensive evaluation, we computed the final cost of the most optimal result found by each approach. The results of these comparisons are presented in Tables 6.3 and 6.4. Fig.6.5 illustrates the most optimal solution for Field #20.



(A) Field #4 (8.23ha) (B) Field #5 (4.39ha) (C) Field #7 (4.58ha)

(D) Field #11 (7.85ha) (E) Field #19 (7.68ha) (F) Field #20 (4.34ha)

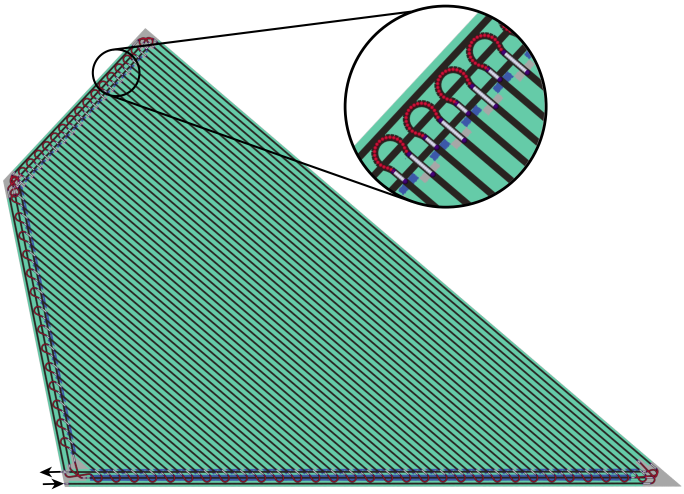
FIGURE 6.2: Access segments are in green. Red arrows are entrances

Field	Approach	time (s)	Coverage	Overlap	Rvs	L_{nw} (m)	f
#4	RS-CCPP	160.59	98.93%	1.23%	2	1184, 17	**0.110**
	CCPP	9.73	98.80%	2.27%	4	3005.92	0.281
#5	RS-CCPP	131.47	98.80%	2.05%	0	1485.36	**0.118**
	CCPP	10.01	98.21%	3.58%	2	1433.84	0.336
#7	RS-CCPP	282.28	98.96%	1.39%	0	1565.47	**0.118**
	CCPP	18.65	98.86%	2.15%	0	1303.98	0.136
#11	RS-CCPP	27.58	97.52%	2.46%	0	1990.87	0.363
	CCPP	7.63	98.23%	2.11%	1	1957.83	**0.168**
#19	RS-CCPP	222.69	99.00%	0.31%	0	1249.25	**0.074**
	CCPP	10.99	99.05%	2.38%	2	1402.57	0.180
#20	RS-CCPP	335.91	98.71%	1.20%	0	1314.63	**0.105**
	CCPP	31.23	98.81%	3.15%	0	1194.58	0.135

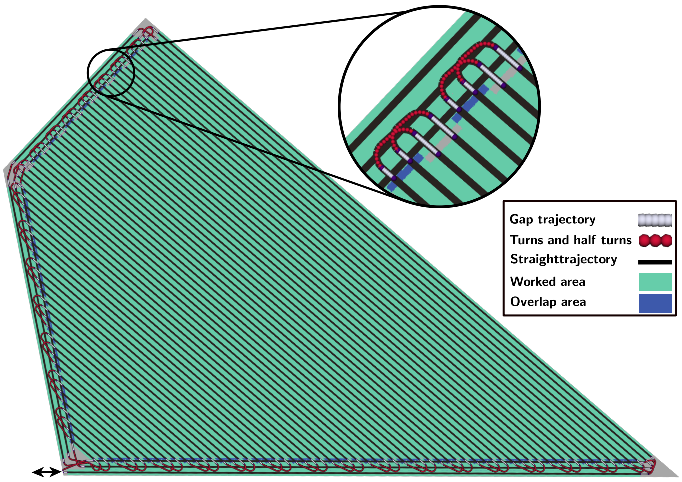
FIGURE 6.3: Numerical results $\gamma_{off} = 2m$: $time$ is exploration time, Rvs counts turns with reverse move, L_{nw} is non-working distance

Field	Approach	time (s)	Coverage	Overlap	Rvs	L_{nw} (m)	f
#4	RS-CCPP	58.08	98.61%	1.68%	10	2916.47	**0.193**
	CCPP	14.14	98.42%	0.23%	57	882.89	0.199
#5	RS-CCPP	35.92	98.65%	2.01%	48	1415.98	**0.201**
	CCPP	8.70	98.75%	3.31%	71	1080.41	0.224
#7	RS-CCPP	38.48	98.73%	3.39%	34	1206.01	**0.193**
	CCPP	9.01	98.66%	4.35%	74	1117.74	0.312
#11	RS-CCPP	132.44	99.08%	2.28%	10	2013.52	**0.137**
	CCPP	23.96	97.48%	4.24%	103	1485.31	0.585
#19	RS-CCPP	98.97	98.99%	0.31%	4	1243.58	**0.088**
	CCPP	10.54	99.20%	2.09%	80	1093.59	0.188
#20	RS-CCPP	51.86	98.56%	1.20%	48	1222.81	**0.140**
	CCPP	13.73	98.76%	3.12%	48	969.11	0.220

FIGURE 6.4: Numerical results $\gamma_{off} = 3m$: $time$ is exploration time, Rvs counts turns with reverse move, L_{nw} is non-working distance



(A) Simple CCPP



(B) RS-CCPP

FIGURE 6.5: Most optimal results obtained for Field #6 while $\gamma_{off} = 2m$. The black arrows indicate where the robot enters and exits

6.1.3 Discussion

Let us remind that in our approach, turns and half-turns are achieved by first attempting to use only forward moves, using the method presented by Dubins [35]. If this approach is not feasible, the algorithm considers turns that require reverse moves. Such turns are generated using the method proposed by Reeds and Shepp [120].

The results obtained by RS-CCPP and O-CCPP with $\gamma_{off} = 2m$ are summarized in Table 6.3. The table shows that, in general, RS-CCPP outperformed O-CCPP for five out of six fields. Specifically, RS-CCPP achieved a better coverage rate than O-CCPP for Fields #4, #5, and #7. Moreover, RS-CCPP resulted in considerably less overlap for Fields #4, #5, #7, #19, and #20. However, in terms of non-working traveled distance O-CCPP was better for four fields.

When γ_{off} was increased to three meters, RS-CCPP outperformed O-CCPP for all fields, as shown in Table 6.4. By skipping rows, RS-CCPP generated fewer turns that involved reverse moves, and reduced overlap rates by nearly 1% in average. O-CCPP resulted in less non-working traveled distance compared to RS-CCPP, which can be attributed to the fact that O-CCPP generates more turns that include reverse moves.

The results depicted in Fig.6.5 reveal that when rows are skipped, the turns require less space in the headlands. This is due to the fact that a tighter turn tends to have a bulb shape that takes up more space, while a wider turn may be flatter. This highlights the ability of the row-skip pattern to decrease the size required for headlands. Typically, headlands are less productive regions of a field, and reducing their size has the potential to increase the productivity of the field.

Depending on various factors such as the characteristics of the robot, the shape and accessibility of the field, headland width, implement size, and optimization goals, one method may outperform the other. For example, if the primary objective is to minimize non-working traveled distance, O-CCPP might be more effective. Therefore, incorporating both approaches or developing a more advanced CCPP that can combine both methods could enhance the effectiveness of the final method in finding optimal solutions.

In conclusion, while the RS-CCPP offers a promising alternative to the traditional O-CCPP, it is important to note that the current approach does not account for field decomposition. To further enhance the capabilities of coverage path planning, in the next chapter we will introduce a novel method that effectively incorporates both sequential and row-skip patterns, handles field decomposition, and even allows for the generation of different patterns for different sub-polygons of the field. This advanced approach aims at providing a more versatile and adaptable solution for optimizing field coverage and addressing a wider range of challenges in agricultural applications.

6.2 DL-based Adaptive Field Decomposition

One of the strengths of the O-CCPP approach lies in its capability to evaluate multiple field decomposition strategies, given a set of input dividing lines. However, this feature also presents a drawback, as examining various field decomposition possibilities and selecting the one that results in an optimal solution can be highly time-consuming.

In this section, we present an extension to the O-CCPP that aims at address this limitation. We propose the development of a deep learning-based method for agricultural field decomposition, which seeks to enhance the efficiency of the O-CCPP approach. By automating the decomposition step and tailoring it to various field characteristics, our proposed solution avoids multiple runs of O-CCPP while maintaining its adaptability and effectiveness.

In the following sections, we will initially present a concise overview of existing field decomposition methods employed in CCPP approaches found in the literature, along with a discussion on the application of DL-based methods in agriculture. Subsequently, we will elaborate on the design and implementation of our proposed Deep Learning-based solution, presenting it as an extension to the O-CCPP approach, aiming at enhance its efficiency and adaptability in complex agricultural scenarios.

While our proposed approach may not provide an immediate definitive result, it offers a valuable perspective for future research, paving the way for advancements in this domain and enabling further exploration into effective solutions

6.2.1 Overview of Field Decomposition in Agriculture

Field decomposition is a crucial step for addressing fields with complex shapes. In certain situations, it is also helpful for managing obstacles within the field. It consists of dividing complex shapes into several sub-fields with much simpler shapes, making it easier to plan coverage paths in agricultural settings.

Most field decomposition methods incorporated in CCPP approaches are shape-based decomposition methods, which focus solely on the field's shape during the decomposition process. Oksanen et al. [100] and Oksanen and Visala [101] integrated a split-and-merge strategy based on a trapezoid decomposition method. Due to its distinctive shape, a trapezoid serves as an excellent candidate, as its two parallel opposite sides can correspond to the driving direction, while the remaining sides can function as headlands. Following the division of the field into trapezoids, the subsequent step involves combining them as much as possible.

With the aim of finding and evaluating all planar subdivisions for optimal path planning, [68] proposed a method that constructed a topological undirected graph from the field's planar subdivision representation, added diagonals, and drew rays from vertices. This new graph was used to perform depth-first searches, identifying all possible dividing lines that separated the field into two sub-regions. The process ensured the discovery of all dividing lines for optimal field decomposition.

Incorporating inclinations into field decomposition, Jin and Tang [67] introduced an approach for obstacle-free fields that divides terrain into slope and flat zones using field boundaries and slope contour lines. Dogru and Marques [31, 33] proposed a method tailored for rectangular fields with obstacles. Initially, the terrain's gradient was calculated, and a threshold was applied, generating a 2D map delineating boundaries between regions with distinct slopes. This map was subsequently merged with the provided 2D obstacle map. Finally, the combined map was partitioned into approximately convex polygons.

An effective field decomposition method integrated into CCPP approaches must consider both the shape and inclinations of a field while minimizing partitioning. Excessive partitioning could lead to an increased number of headlands for half-turns, consequently resulting in less productive, compacted soil near the field edges and dividing lines. It is important to note that some fields, even with complex and inclined shapes, may not necessitate decomposition. Therefore, an efficient decomposition method must first determine whether a field requires decomposition or not and, if so, ensure minimal partitioning.

A CCPP approach is a multi-objective task, and altering the focus on each objective can significantly impact the overall result. It is more efficient and ideal to let the CCPP approach decide which partitioning would yield a better outcome based on the specific objectives of the operation being performed on the field. The O-CCPP approach, described in the previous chapter, can examine multiple dividing lines and select the one leading to a better outcome, satisfying a unique set of objectives. However, it was demonstrated that employing such an approach can be extremely time-consuming.

In an effort to address this challenge, we aim at develop a deep learning-based approach that offers effective field partitioning and optimal driving directions for each sub-polygon within the field. Prior to exploring the details of this concept, let us provide a brief overview of the role of Deep Learning in the domain of agriculture.

6.2.2 Overview of Deep Learning-based Approaches in Agriculture

Deep learning (DL) has significantly impacted various aspects of agriculture, enabling researchers and practitioners to address complex challenges in this domain. The applications of DL-based approaches in agriculture can be broadly categorized according to their primary goals, including classification, detection, segmentation, time series forecasting, and reinforcement learning.

In the realm of classification and detection, DL techniques have been employed for identifying and classifying crop diseases, pests [79, 82, 133], weeds [118, 119, 162], and automating fruit harvesting [103].

Segmentation techniques have been applied to land cover classification [85, 105, 144], facilitating the efficient categorization of various land cover types, including crops, forests, urban areas, and water bodies. Segmentation techniques have also been applied for orchard trees segmentation using aerial images Anagnostis et al. [6] and Sun et al. [143].

Time series forecasting models, such as recurrent neural networks and long short-term memory networks, have been employed to predict crop yields (Elavarasan and Vincent [38], Sharma, Rai, and Krishnan [138], and Sun et al. [142]). By analyzing factors like weather, soil conditions, and crop growth stages, these models offer accurate forecasts, assisting farmers in making informed decisions about crop management.

Reinforcement learning has been implemented for CCPP in kiwifruit picking robots [158] and autonomous navigation in paddy fields [1]. These approaches enable agricultural robots to navigate and perform tasks more efficiently, addressing labor shortages and enhancing productivity in the sector.

Although deep learning has been applied across various aspects of agriculture using a diverse range of models, the exploration of generative models in this domain remains limited. Furthermore, only a few works have specifically focused on CCPP for wheeled robots. To provide a deeper understanding of the current state of research in CCPP for wheeled robots using DL, we will discuss some of the initial efforts in this area.

One initial effort was DeepWay [86], an innovative DL-based approach for global automatic path generation from satellite images of row-based crops. It leveraged a lightweight DL model that predicted way-points on rows between trees from an occupancy grid map derived from satellite images. DeepWay performed well on real-world remote sensing-derived maps. Despite its promising results, DeepWay had limitations, such as sensitivity to inter-row distance and dependency on external segmentation for occupancy grid map generation. Nevertheless, DeepWay demonstrated the potential of combining DL and path planning for wheeled robots in agricultural fields, paving the way for future research and development in this area.

In another study, a DL-based CCPP for a kiwifruit picking robot was proposed by Wang et al. [158]. The researchers developed a deep reinforcement learning-based method, employing a region partitioning algorithm to convert the CCPP problem into a traveling salesman problem. The proposed approach resulted in faster convergence and significantly enhanced coverage path efficiency, with a reduction in path length and overall navigation time compared to existing approaches in the literature. However, the method's applicability might have been limited to specific environments or robotic systems, its performance relied on the quality of collected environmental data, and it might not have been efficient for larger or more complex environments due to increased computational resources and time requirements.

Lastly, a framework proposed by [77] consisted of three layers that handled environment mapping, path generation, CCPP, and dynamic obstacle avoidance. These layers worked sequentially, using the results of the previous layer as a reference to decrease computational effort and improve efficiency. The CCPP approach was a two-step process that first determined the search direction and then generated the path using a DL-based method. The search direction aimed to minimize the total length of back-and-forth coverage trajectories, while the DL-based path generation used a fully convolutional deep neural network to estimate intersection points and predict global CCPP trajectories. Advantages of this approach included efficient path planning, adaptability and scalability, and low spatial complexity. However, it also had limitations, such as a focus on single-objective optimization, limited start and end points, lack of consideration for headland coverage, and the possibility of not generating good coverage paths when the minimum turning radius of the robot is greater than half of the distance between parallel tracks.

DL has seen considerable progress in agriculture, but few studies focus on CCPP for wheeled robots. While DL-based CCPP excels in other domain, such as ship hull inspection robots [91], tetromino-based cleaning robots [75], vacuum cleaning robots [98], and collaborative unmanned aerial vehicles for monitoring [34]. This presents an opportunity for further research and development, which can potentially enhance the efficiency and effectiveness of agricultural operations involving wheeled robots. By understanding and building upon the initial efforts in this area, such as those mentioned in this section, researchers and developers can explore new ways to leverage DL for more efficient and sustainable agricultural practices.

6.2.3 Conceptualizing a DL-based Approach for Field Decomposition

The integration of DL techniques in agriculture presents an opportunity to develop novel solutions for field decomposition and CCPP. However, to develop such an approach, a suitable dataset is required. To the best of our knowledge, no dedicated dataset exists for this purpose. Therefore, the first step towards creating a DL-based approach for field decomposition in agriculture is to develop a comprehensive dataset that accurately represents the diversity of complex field shapes encountered in real-world agricultural scenarios. This dataset can serve as a benchmark for evaluating and comparing various DL-based field decomposition methods, promoting further research and development in this area.

Based on the assumption that farmers possess a unique expertise and understanding of the distinctive characteristics of their fields, we constructed a dataset of aerial images that indicate how farmers decompose their fields into sub-polygons (if required) and which driving direction they used for each sub-polygon. Our objective was to use this dataset to train a DL-based model capable of generalizing field decomposition and determining the driving direction for a field or its sub-polygons using the field boundaries and elevation data as input.

The proposed concept involves several steps, including data collection, annotation, data preprocessing, model design, test and validation. In the remainder of this section, we will describe each step in detail to provide a comprehensive understanding of the proposed approach.

Data Collection

The process of data collection is of paramount importance when developing a DL-based approach for field decomposition. To achieve this, various types of data are necessary, such as satellite or aerial images, field boundaries, and elevation data. In order to obtain these data, information from 900 fields located throughout France was collected from the extensive public data collection maintained by the French National Institute of Geographic and Forestry Information (IGN [59]). Their public data collection contains various databases containing several types of geographic, topographic, and image data covering the entire French territory.

Aerial images: To obtain high-quality aerial images of the fields, the BD ORTHO® database was relied upon, which provides an extensive collection of aerial images of the French territory at a resolution of $20cm$ [57]. The BD ORTHO® dataset is produced with both RGB and infrared channels, making it an excellent source for capturing detailed information about the fields.

Field boundaries: To extract the contours of the fields, we used the RPG dataset, which contains polygonal shapes of the fields located in France [60]. The contours have been extracted automatically for each selected field.

Elevation data: To obtain elevation data, the REST API provided by IGN [58] is utilized. This API offers a service for determining the altitude at one or more specified points.

Annotation

To facilitate the annotation process of the collected data, an interactive annotation tool was developed as a plugin for QGIS [7]. QGIS is a free and open-source GIS software that allows users to create, edit, visualize, analyze, and publish geospatial information. By developing the annotation tool as a plugin for QGIS, the flexibility and functionality of QGIS were exploited to provide an efficient and user-friendly interface for annotating the collected data.

The developed annotation tool allows to iterate through the fields within a specific area. For each field, the corresponding boundary polygon and aerial image can be loaded and visualized. Based on visual cues, the user can split the field polygon into sub-polygons by drawing line segments. Finally, the driving directions for each sub-polygon can be annotated as line segments as well.

The outcome of the annotation process for a field is a comprehensive set of data, including high-resolution aerial images of the field in GeoTIFF file format [123], one or multiple polygons representing the sub-divisions of the field in Shapefile format [107], and one or several driving directions annotated for each sub-polygon in Shapefile format.

Let us note that performing elevation queries for all pixel is a time-consuming process that can significantly impair the usability of the annotation process. To avoid slowing down the interactive annotation, elevation data for each field is added during the preprocessing step instead.

Data Preprocessing

Data preprocessing is a critical step in developing a DL-based approach for field decomposition, as it involves transforming raw data into a format that can be efficiently used by the DL model. In this section, we will discuss the various preprocessing steps employed to prepare the collected data for training and validation of the proposed model.

Field Boundaries Processing: The Shapefile containing the field boundaries obtained from the RPG dataset is used during the preprocessing stage to extract individual field polygons and crop the corresponding aerial images. Following this process, the Shapefile is converted to a single-channel raster file where the value of each pixel on the boundary represents its direction„ computed and discredited as an angle between 1 and 180 degrees with a precision of 1 degree. The values of all other pixels are set to zero. This method elegantly captures the necessary information about the field boundaries in a format that can be effectively utilized by the DL model during training and inference.

It is important to note that a line segment or boundary can form two different angles with the x-axis, one falling in the quadrants with positive y-values, and another in the quadrants with negative y-values. In order to maintain consistency and facilitate the learning process of the DL model, only the angles that fall on the positive y side are selected. In the special case of a horizontal line segment, which can form both 0 and 180 degrees angles with the x-axis, the 180° angle is preferred. This approach ensures a unified representation of the angles, enabling the model to learn more effectively from the input data.

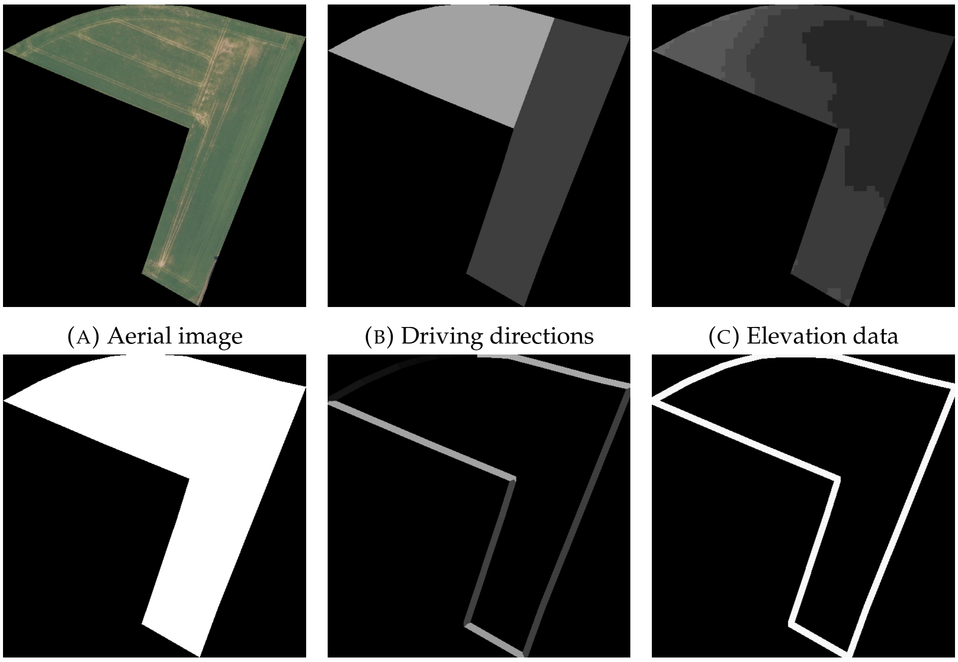
Driving Directions Processing: In a similar way, the driving directions contained in the second Shapefile are discretized as angles between 1 and 180 degrees. Subsequently, a single-channel raster image is created in which all pixels inside the field polygon are associated with their corresponding angle, while all pixels outside the field are set to zero. This approach effectively captures the driving direction information within the field, providing valuable input for the DL model during training and inference.

Image Cropping and Resizing: The high-resolution aerial images obtained from the BD ORTHO® dataset may have varying dimensions, making it necessary to crop and resize the images to a uniform size that is compatible with the input requirements of the DL model. To achieve this, the images are first cropped based on the field boundaries, ensuring that the cropped images only contain the area of interest (*i.e.*, the field). Subsequently, the cropped images are resized to a fixed resolution, preserving the aspect ratio, to prevent any distortion in the image content. This ensures that all input images have the same dimensions, facilitating efficient training and validation of the DL model.

Elevation Data Retrieval: As mentioned earlier, elevation data for each field is obtained during the preprocessing step. For each field, the elevation at each pixel of the aerial image is queried using the REST API provided by IGN. This process generates a single-channel raster file containing the elevation data, where the value of each pixel represents the elevation at the location of the pixel.

Binary Mask Generation: For each raster image generated in the previous steps, a corresponding binary mask is also created. They allow to identify the relevant areas and enable the DL model to focus on the meaningful information within the field boundaries while ignoring the background or irrelevant regions. Binary masks further enhance the model's ability to learn and generalize from the input data effectively.

Fig.6.6 provides a visual representation of the generated images and their corresponding masks for a single field. It is important to note that a single mask can be employed for aerial images, driving directions, and elevation data, and this mask is referred to as the field mask.



(A) Aerial image (B) Driving directions (C) Elevation data

(D) Field mask (E) Boundary directions (F) Boundary mask

FIGURE 6.6: Visual representation of generated images and corresponding masks

The aerial images serve primarily as a reference for annotating the driving directions during the annotation process. In the preprocessing stage, these images are used to query the elevation data at the specific locations of their pixels. The aerial images themselves are not directly incorporated into the DL model.

Data Augmentation: To enhance the robustness and generalization capabilities of the proposed model, data augmentation techniques are employed during the preprocessing stage. Four different augmentation functions are applied to each field: a 90° rotation, a 180° rotation, a horizontal flip, and a vertical flip. For each of these augmentations, the value of pixels representing a direction (in both the driving direction and boundary images) is modified accordingly to maintain consistency. Data augmentation generates additional training samples by applying various transformations to the original images, which increases the diversity and size of the training dataset. In this case, the data augmentation process results in 3600 more samples, ultimately helping the model to better learn the underlying patterns and relationships in the data.

Data Normalization: Normalization of the input data ensures that all input features have the same scale, thereby improving the convergence and stability of the model during training. In this case, the relevant input features, including the field boundaries, driving directions, and elevation data, are normalized to a range of $[0, 1]$. Through this normalization process, the DL model is enabled to effectively learn and generalize from the input data across different fields.

Data Splitting: The final step in the data preprocessing stage involves splitting the collected dataset into training, validation, and test sets. This ensures that the model is trained on a diverse range of samples, validated on a separate set of samples to fine-tune the model hyperparameters, and finally tested on a completely unseen set of samples to evaluate the model's performance. A common approach for data splitting, such as the $70 - 15 - 15$ rule, is employed, where 70% of the dataset is used for training, 15% for validation, and the remaining 15% for testing.

Model Design and Implementation

The objective of the model is to process the generated images and masks in order to determine driving directions and enable field decomposition. The model's inputs consist of a two-channel image ($2 \times 512 \times 512$) containing elevation data in the first channel and boundary directions in the second channel, and a two-channel mask ($2 \times 512 \times 512$) with the field mask in the first channel and the boundary mask in the second channel. The model's output is a driving direction image representing the field. Thus, the driving direction image generated during the preprocessing stage serves as the labels for the model, providing a reference for learning and evaluation. Field decomposition can be derived from the driving direction image.

Implementation: The implementation of the model follows a PyTorch training pipeline [108]. In the initial stage, the PyTorch data loader is employed to load and combine channels from various images and masks generated during the preprocessing step. The model is then constructed using the PyTorch library and its associated tools. Finally, training is carried out by adhering to the train loop paradigm provided by PyTorch.

Architecture: In our model, we chose a U-Net-like architecture [124], closely following the implementation used by Liu et al. [80], which incorporates the partial convolution technique proposed by Liu et al. [81]. The implemented architecture makes use of partial convolution layers in place of traditional convolution layers. These partial convolutions help handling missing data by only considering valid input values during convolutions, which is useful to consider only data inside the field.

The architecture is designed as an encoder-decoder network, where the encoder captures high-level features through a series of convolutional and downsampling layers, while the decoder reconstructs the output image through a series of upsampling and convolutional layers.

Our model comprises several encoder blocks with partial convolution layers, each followed by Batch Normalization and Leaky ReLU activation functions. The encoder section successively increases the number of channels while reducing the spatial dimensions of the input image. In the decoder section, the feature maps are upsampled and concatenated with corresponding encoder feature maps, followed by partial convolution layers, Batch Normalization, and Leaky ReLU activation functions. The output layer of the network uses a traditional convolution layer followed by a Sigmoid activation function to generate the final output image.

The overall architecture is designed to handle input images with dimensions of 512×512 pixels and two channels. It contains a total of $32,856,107$ trainable parameters, resulting in a forward/backward pass size of approximately $717,784$ MB. A detailed diagram of the network architecture can be found in Fig.6.7, illustrating the structure and connectivity of the encoder and decoder blocks.

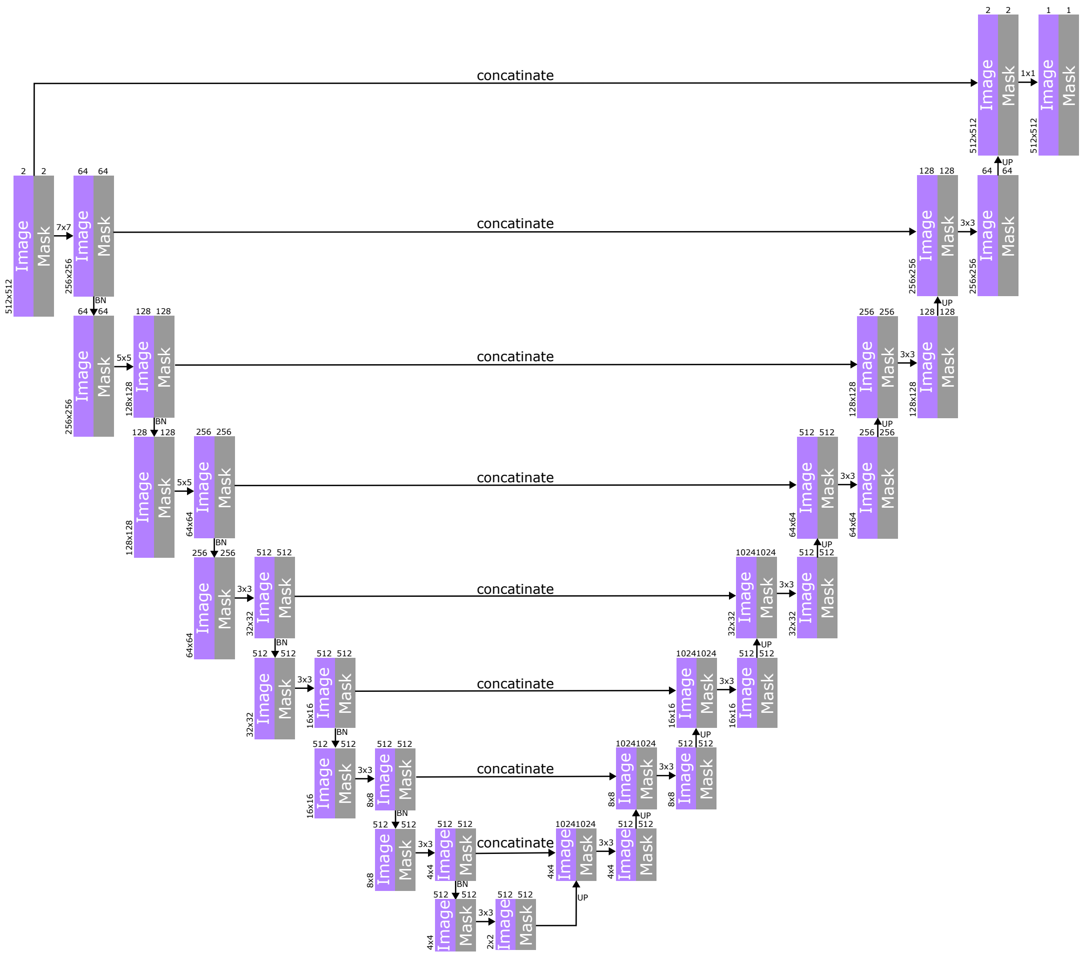


FIGURE 6.7: U-Net-like architecture with partial convolutions and skip connections

Loss function: We proposed a custom loss function for our model to effectively learn the driving direction while preserving the field boundaries. This custom loss function combines two separate loss components, each responsible for calculating the loss inside and outside the field using the provided field mask. The overall loss function L is the sum of these two components, defined as:

$$L = L_{\text{inside}} + L_{\text{outside}} \tag{6.3}$$

L_{inside}: This component calculates the loss within the field. It employs the Smooth L1 loss with $\beta = 0.05$ as its underlying loss function. The Smooth L1 loss is robust to outliers and helps in learning the driving direction within the field. Mathematically, the Smooth L1 loss is defined as:

$$\rho_\beta(x) = \begin{cases} \frac{1}{2}x^2/\beta & \text{if } |x| < \beta \\ |x| - \frac{1}{2}\beta & \text{otherwise} \end{cases} \tag{6.4}$$

For each pixel (i, j) inside the field, the Smooth L1 loss is calculated between the predicted pixel value $\hat{y}_{ij}$ and the ground truth pixel value y_{ij}. The sum of these losses is divided by the number of pixels inside the field to obtain the average loss L_{inside}.

L_{outside}: This component calculates the loss outside the field. It utilizes the L1 loss as its underlying loss function. The L1 loss encourages the model to generate output values close to zero outside the field, ensuring that the generated driving direction image is confined within the field boundaries. For each pixel (i, j) outside the field, the L1 loss is calculated between the predicted pixel value $\hat{y}_{ij}$ and the ground truth pixel value y_{ij}. The sum of these losses is divided by the number of pixels outside the field to obtain the average loss L_{outside}.

By combining these two loss components, the proposed custom loss function helps the model to learn the driving direction effectively while preserving the field boundaries.

6.2.4 Results and Discussion

Despite extensive experimentation and fine-tuning, the model's performance on the test data was unsatisfactory. Several attempts were made to improve the model's generalization capabilities, including parameter adjustments, layer freezing, the utilization of various optimizers such as Adam, RMSprop, and SGD, as well as exploring different learning rates and learning rate schedulers. However, these efforts did not yield the desired outcome.

Upon investigating the dataset, it became apparent that the fundamental assumption regarding farmers' expertise might not be accurate. It was observed that farmers often choose a driving direction that aligns with one of the field boundaries, preferably the longest one. This alignment provides a visual reference for driving the tractor parallel to the boundary and reduces the number of half-turns required, resulting in fewer errors during the transition to the next track. Additionally, this approach is less physically demanding for the farmers.

As a consequence, it is crucial to acknowledge that, in addition to the diverse constraints that may be considered by individual farmers based on their personal preferences, manual tractor driving also presents challenges associated with the physical demands and driving skills of the farmers. It is worth noting that, in the context of an autonomous robot, factors such as physically demanding tasks and driving skill become irrelevant.

Moreover, the unique characteristics of each field and the varying preferences of farmers create substantial variability, which complicates the process of identifying a common thread among their choices. This heterogeneity within the dataset has hindered the model's capacity to generalize effectively and accurately predict driving directions across various fields.

In conclusion, the model's inability to deliver satisfactory results on the test data might be attributed to the complexities and inconsistencies in the dataset, as well as the variety in farmers' preferences, operation, and machinery requirements. Future research may benefit from reevaluating these assumptions and exploring alternative strategies to account for the diverse preferences and decision-making processes of farmers.

In light of these hypotheses, a potential solution to improve the model's performance could involve generating a dataset using our CCPP approach detailed in the previous chapter. This approach would create coverage paths for a variety of fields while considering multiple dividing lines, in consultation with an expert, and allowing the CCPP approach to determine the optimal dividing line based on the optimization criteria. The dataset could be generated for fields with various shapes and incorporate diverse weights for soft constraints.

However, as demonstrated in the previous chapter, the proposed CCPP can be extremely slow when evaluating multiple dividing lines. Therefore, it would be beneficial to first develop a robust approach that can efficiently handle multiple dividing lines. Once such an approach is established, a comprehensive dataset can be generated for the deep learning model.By incorporating this dataset, the deep learning model can become even more robust and benefit from the enhancements provided by the refined CCPP approach. In turn, this synergy between the CCPP and deep learning model would lead to improved model performance, more accurate driving direction predictions, and ultimately, a faster and more efficient CCPP approach.

Consequently our immediate focus will now pivot towards developing a sophisticated CCPP approach capable of managing multiple dividing lines and accounting for a wide range of constraints, including those related to field slopes while achieving a faster computation time. By establishing a comprehensive and efficient CCPP approach, our intention is to lay a solid foundation for future improvements in the proposed DL model. In the upcoming chapter, we will elaborate on this advanced CCPP approach, which has been designed to tackle the complexities and inconsistencies within the dataset, as we strive to create a more effective and comprehensive solution.

6.3 Conclusion

In this chapter, we presented two extensions to the O-CCPP approach. The first extension incorporated a row-skip pattern into the path generation process, which improved the efficiency and versatility of the CCPP. The proposed RS-CCPP method demonstrated promising results when compared to the original O-CCPP approach. However, it is crucial to recognize that the choice between RS-CCPP and O-CCPP depends on various factors, such as the robot's characteristics, field shape, and optimization goals. As a result, incorporating both approaches or developing a more advanced CCPP that can employ both patterns would be beneficial in finding optimal solutions.

The second extension aimed to automate the field decomposition step using a deep learning-based method. However, the model's performance on the test data was unsatisfactory due to complexities and inconsistencies within the dataset. Future research may benefit from generating a dataset using an advanced CCPP approach that is capable of efficiently handling multiple dividing lines and accounting for a wide range of constraints, including field slopes. By establishing a comprehensive and efficient CCPP approach, the foundation for future improvements in the proposed DL model can be laid.

In the next chapter, we will introduce a more advanced CCPP approach that combines the strengths of the sequential and row-skip patterns, addresses field decomposition, and allows for the generation of different patterns for different subpolygons of the field. This advanced approach aims at providing a more versatile and adaptive solution for optimizing field coverage and addressing a wider range of challenges in agricultural applications.

CHAPTER 7

Advanced 3D Hybrid Path Planning with Multiple Objectives

In this chapter, we present a novel hybrid approach for CCPP that takes into consideration the inclination of the field. Unlike previous approaches where the general direction of parallel trajectories were limited to the direction of the field edges, this method is capable of exploring all possible driving directions, from 0 to 180 degrees, with a predefined step size. For each driving direction, the method generates parallel trajectories within the field and searches for the best entrance to reach and travel them. It then covers the headlands and searches for an exit trajectory towards the nearest accessible part of the field. Additionally, for each driving direction, the method examines two different coverage patterns: sequential and row-skip.

In cases where a dividing line is provided, it decomposes the field polygon into sub-polygons and explores all possible combinations of driving directions and coverage patterns for each sub-polygon. Once a sub-polygon is fully covered, it searches for the nearest uncovered sub-polygon and repeats the same process. Once all sub-polygons have been covered, it searches for the nearest exit.

Upon generating all possible solutions, this method selects the optimal solution that maximizes coverage rate, minimizes overlap rate, non-working traveled distance, and operation time directly. Furthermore, the method optimizes soil erosion and energy consumption indirectly by considering the slopes of the trajectories.

This new approach offers several advantages over previous CCPP methods, including faster processing times, flexibility in choosing coverage patterns, and consideration of slopes for finding the optimal solution. In this chapter, we provide a detailed description of the methodology and evaluate its performance through simulations and comparisons with other CCPP approaches.

7.1 Motivation and Concept

In Chapters 5 and 6, we presented our approach for generating coverage paths for autonomous agricultural robots. Our original CCPP approach utilized a tree-based construction and exploration algorithm, while the extension presented in Chapter 6 included the ability to generate row-skip patterns for more efficient coverage. Our approach was effective in finding optimal solutions for both simple and complex fields, including those with headlands. However, it had some limitations, including generating solutions with trajectories aligned only with one of the field edges, generating row-skip pattern only for simple fields, and being relatively slow compared to the open-source algorithm, F2C [87], with an average computational time of 624.6*s* for our approach and 1.52*s* for F2C. In contrast, while F2C was significantly faster in finding solutions, it was limited in its ability to cover headlands, address complex field shapes, account for the transition state of the robot, determine proper entry and exit points, and perform multi-objective optimization.

To leverage the strengths of our previous approach, which effectively covers headlands, addresses complex field shapes, considers field accessibility, and accounts for implement transition state, as well as the benefits of the F2C method, which efficiently generates parallel trajectories for a variety of driving directions, we propose a novel approach that combines the two. This novel approach overcomes the limitations of both methods and considers working trajectory inclinations, which have a direct impact on soil erosion and energy consumption of the robot.

7.2 Methodology

Similar to our previous approach, this novel method also begins with a preprocessing step that involves decomposing the field into sub-polygons if a dividing line is provided. It also constructs headlands and their corresponding trajectories, as well as turning spaces. Afterwards, we utilized the F2C approach to generate parallel trajectories (*i.e.*, tracks) in the main part of the field (*i.e.*, excluding the headlands), with a specified driving direction in the range 0 to 180 degrees. Fig. 7.1 illustrates the process for a single driving direction.

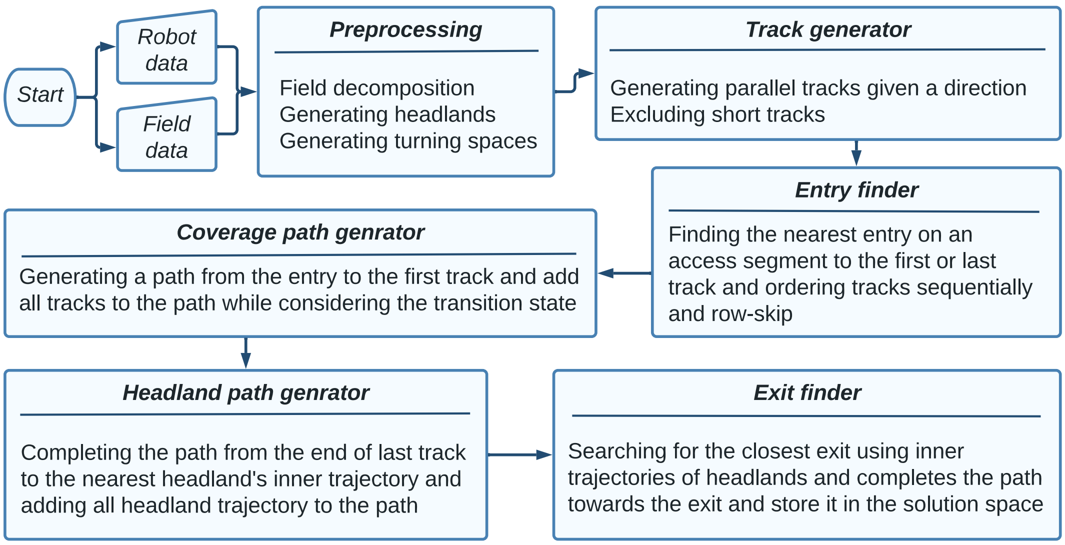


FIGURE 7.1: Diagram of the proposed approach

For a set of tracks, the method first excludes any short track that does not meet the minimum working distance constraint. Next, the method examines two possible cases: 1) ordering tracks sequentially, and 2) ordering them in a row-skip pattern. The closest entry on an access segment is then determined for each case, and a path is constructed from the entry towards the start of the ordered tracks. All ordered tracks are added to the path including a transition state at the start and end point. The path is then completed by covering all headlands inner trajectories, and gap-covering trajectories. The method then searches for the closest exit using inner trajectories of headlands and completes the path towards the exit point.

For a driving direction, two solutions (*i.e.*, paths), one with sequential and another with row-skip pattern, are added to the solution space. This process is repeated for several driving directions from 0 to 180 degrees with a fixed step size ℓ_s.

In the case of complex fields, where a set of dividing lines is provided to decompose the field into sub-polygons, the method starts with each sub-polygon that has at least one access segment. Starting from one sub-polygon, the method generates parallel tracks, orders them in sequential and row-skip patterns, finds the nearest entry, constructs a path from the entry towards the start of the ordered tracks, adds all ordered tracks to the path while considering the transition state, and completes the path by covering all inner trajectories and gap-covering trajectories. The method then performs a search to find the nearest uncovered sub-polygon using inner trajectories, and iterates to cover the remaining sub-polygons. Once all sub-polygons are covered, the method performs a search to find an exit and complete the path.

Therefore, the solution space contains solutions that are a combination of all possible patterns, driving directions, as well as the order of visiting sub-polygons. For instance, if there are two sub-polygons, each with at least one access segment, and $\ell_s = 1$ degree, the solution space will contain $259,200$ solutions. Equation (7.1) can be used to compute the number of solutions, where $N_{polygon}$ denotes the number of sub-polygons, $N_{accessible}$ represents the number of sub-polygons with at least one access segment, and $N_{pattern}$ is the number of patterns used to generate parallel tracks.

$$N_{solution} = \left(\frac{180 \times N_{pattern}}{\ell_s}\right)^{N_{polygon}} \times N_{accessible} \tag{7.1}$$

In the remainder of this section, we provide detailed explanations of each process involved in the proposed method, including the specific algorithms and techniques used to accomplish each step.

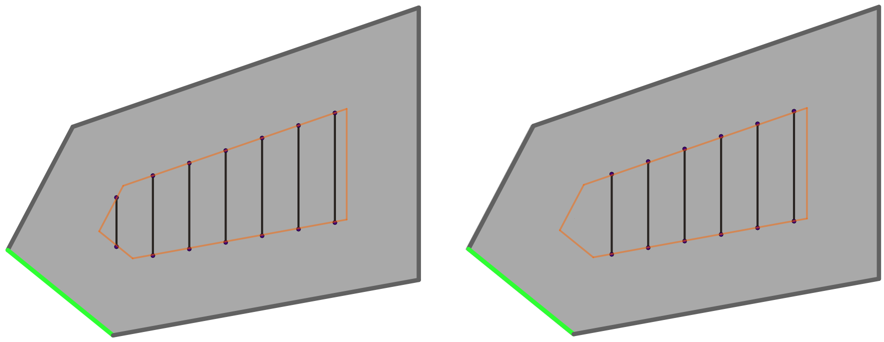
7.2.1 Preprocessing

The preprocessing step in this method is similar to the original approach, with only one difference which is the number of gap-covering trajectories. In this version, the number of gap-covering trajectories is user-defined rather than being fixed at one.

As a reminder, the preprocessing involves generating headlands containing p inner trajectories and g gap-covering trajectories, as well as turning spaces based on the working width and robot parameters such as γ_{off} and ℓ_o. Additionally, if a dividing line is provided, the field is decomposed into sub-polygons. For a more detailed description of this process, please refer to Chapter 5, Section 5.2.2.

In contrast to the original approach, where adding more inner trajectories in headlands or gap-covering trajectories increased the number of branches in the tree and resulted in longer computational times (as shown in Chapter 5, Fig.5.2), the proposed approach allows for a variable number of inner and gap-covering trajectories, depending on the implement width and the length of transition trajectories.

In the original approach, entry points were determined during the preprocessing stage, requiring an operator to decide which ones to keep and which ones to reject.



(A) Parallel tracks for driving direction $\frac{\pi}{2}$ (B) After excluding short tracks

FIGURE 7.2: Parallel track generation. Headland borders and access segment are depicted respectively in orange and green

However, increasing the number of entry points also resulted in longer computational times due to the additional explorations required. In the new approach, the determination of entry points is more flexible, as they are not predetermined during the preprocessing stage. Instead, a specific entry point is determined for a given driving direction after generating the parallel tracks, by finding the closest point that provides access to the parallel tracks.

7.2.2 Generating and Covering Parallel Tracks

The next process generates parallel tracks within the main part of the field, using the F2C approach, with a predetermined driving direction and working width. In our approach, all tracks that do not meet the minimum working distance constraint are excluded. This constraint ensures that the robot travels a minimum distance, Δ_{mwd}, with its implement turned on, before it can be turned off (see Chapter 4, Section 4.1.3). Fig.7.2 provides an illustration of this process for a driving direction of $\frac{\pi}{2}$.

Following the generation of parallel tracks, the shortest path for the robot is determined to enter the field and begin traveling along these tracks. For each coverage pattern (sequential and row-skip), there are four possible cases depending on which end of the first or last tracks is closest to an access segment. Figs. 7.3 and 7.4 depict all of these cases, regardless of the presence of an access segment.

Considering the presence of an access segment, case 7.3b is selected for the sequential pattern, while case 7.4b is selected for the row-skip pattern. To achieve this, the access segment is first democratized into a set of points with a spacing of ℓ_a, and a direction perpendicular to the access segment towards the interior of the field is associated with each point. The closest point to either end of the first and last track is then selected, and a Dubins trajectory is generated to reach it [35]. Therefore, the selected point on the access segment specifies which end of which track should be used for reaching the parallel tracks and in which order they must be visited. Fig.7.5 presents the path that begins at the access segment and travels all parallel tracks for both patterns. In addition, a transition trajectory is included at both ends of each track for turning the implement on and off.

7.2.3 Covering Headlands

Following the addition of all parallel tracks to the path, the headland coverage process begins by searching for the nearest headlands to the end point of the generated

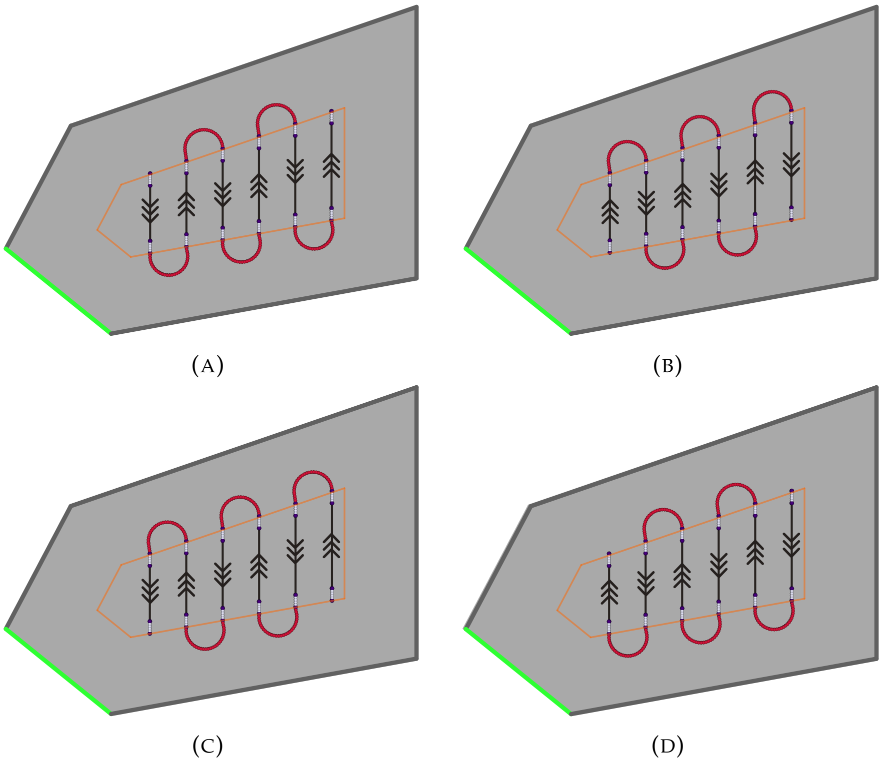


FIGURE 7.3: Variations in selecting a track and its ending point for starting, depicted with turns and transition trajectories respectively in red and white, for sequential pattern.

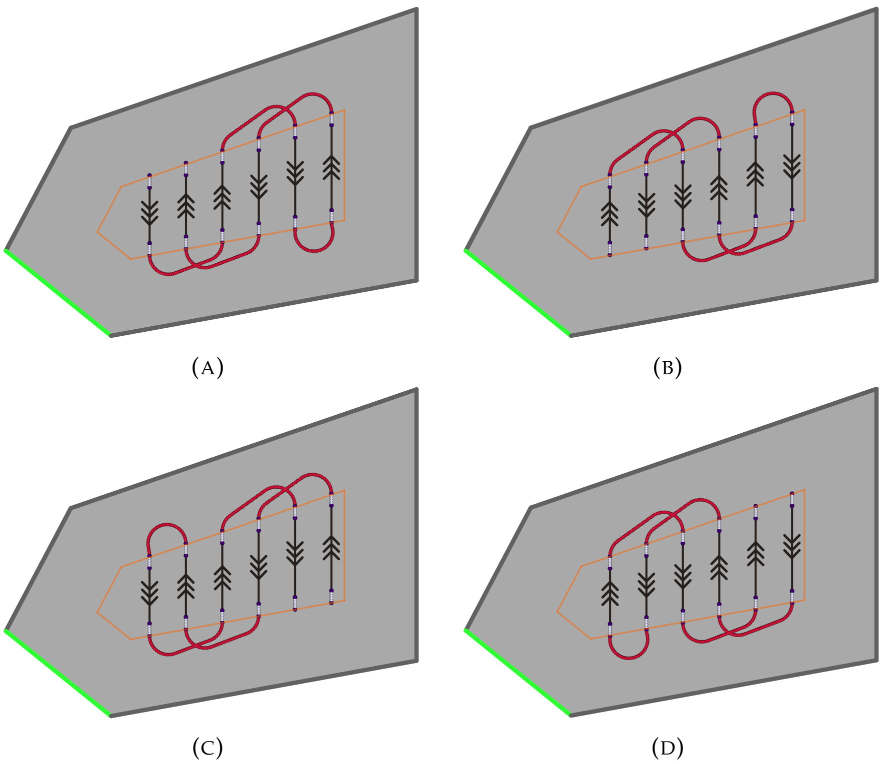
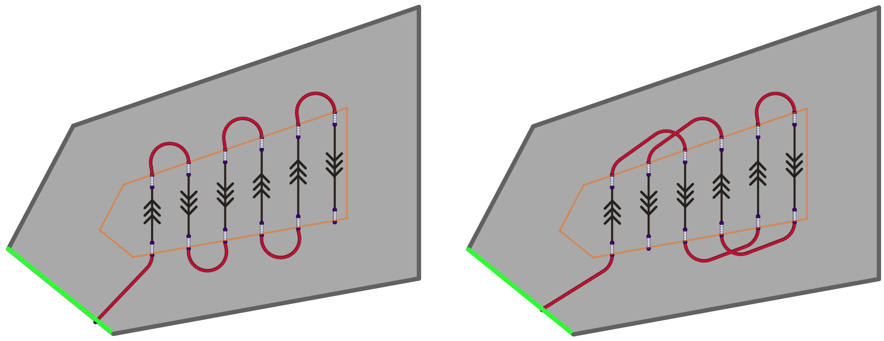


FIGURE 7.4: Variations in selecting a track and its ending point for starting, depicted with turns and transition trajectories respectively in red and white, for row-skip pattern.



(A) Sequential pattern

(B) Row-skip pattern

FIGURE 7.5: The path starting from an access segment and covering all tracks for the selected cases. Turns and transition trajectories are depicted respectively in red and white.

path. In this process, only the end points of the first-level inner trajectory of each headland is considered *i.e.*, the one closest to the field edge. Therefore, the first-level inner trajectory with the shortest Dubins trajectory from the end of the last track to one of its end points is chosen to complete the path. Once the first-level inner trajectories of all headlands are covered, the method proceeds to cover the second-level inner trajectories and all remaining inner and gap-covering trajectories.

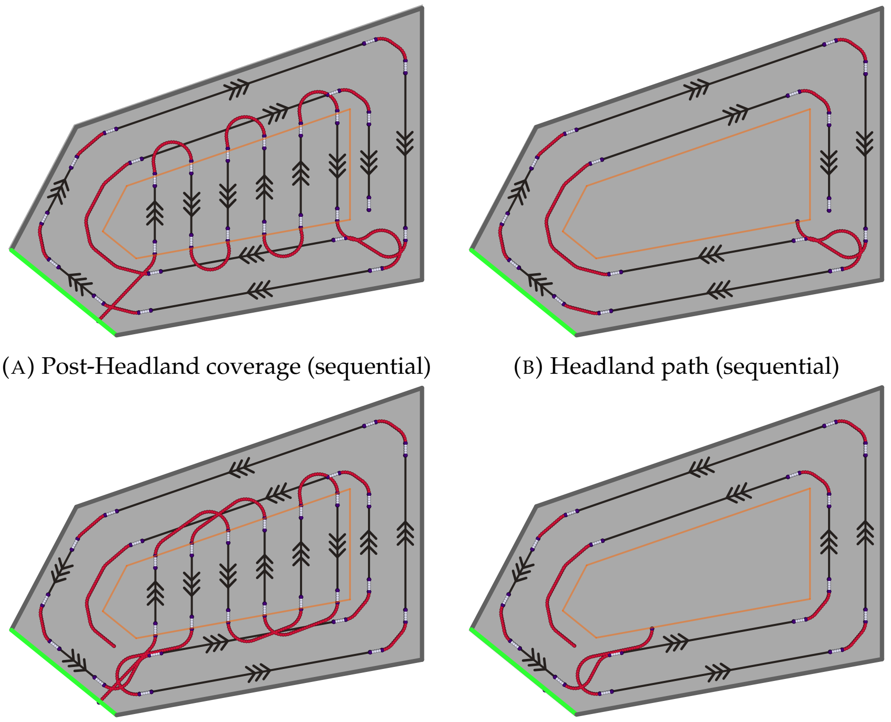
The process for covering headlands, for both sequential and row-skip patterns, is illustrated in Fig.7.6, excluding gap-covering trajectories for clarity. Figs.7.6a and 7.6c illustrate the path from the access segment to the end of the last covered inner trajectory. Figs.7.6b and 7.6d, on the other hand, illustrate only a portion of the path from the end of the last track to the end of the last covered inner trajectory. It is important to note that in some cases, such as the example shown in the figure, some of inner and gap-covering trajectories may not be covered due to the minimum working distance constraint.

The circular coverage pattern for headlands can either be clockwise or counterclockwise, depending on which end point of a first-level inner trajectory is closest to the end of the last track. Utilizing this circular pattern for covering headlands is mainly for reducing computational time as well as for further improvement, allowing for the coverage of all field corners.

7.2.4 Exiting the Field

Following the completion of headland coverage process, the method searches for the nearest access segment to exit the field. This search process, referred to as *circular search*, starts by traversing the first-level inner trajectories of the headlands in both clockwise and counterclockwise directions until the nearest access segment is found. The result is two potential paths around the field, with the shorter one being selected as the exit path. Fig.7.7 shows the exit search results for both sequential and row-skip patterns.

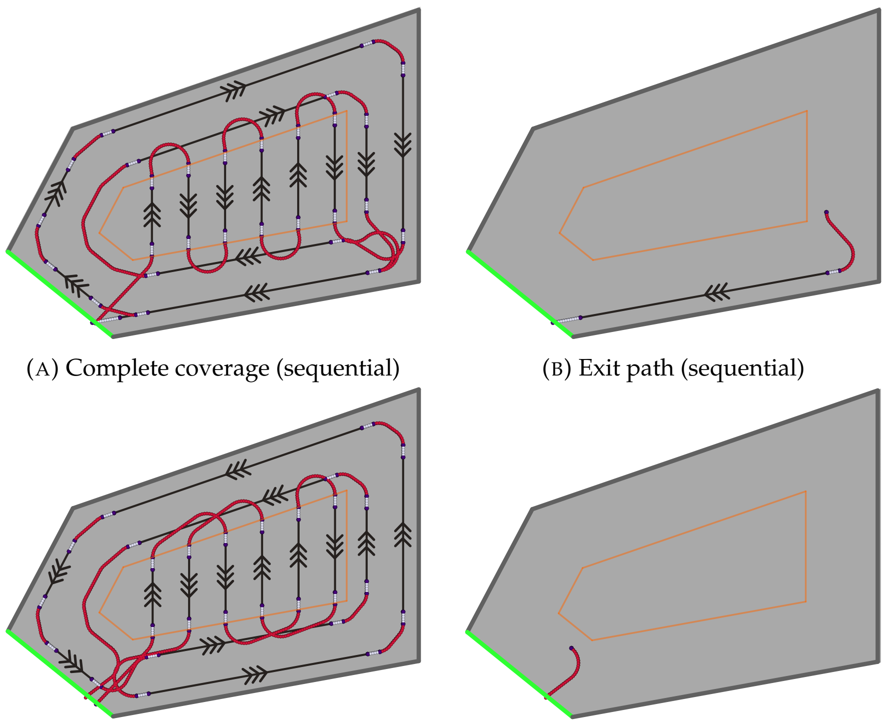
As depicted in Fig. 7.7b, if the last inner trajectory taken to reach the access segment allows for continuing in the same direction towards the exit, a transition is made just before the access segment to turn off the implement and exit the field without changing direction. Otherwise, the access segment is discretized into a set of points with a spacing of ℓ_a, with each point being associated with a direction perpendicular



(A) Post-Headland coverage (sequential) (B) Headland path (sequential)

(C) Post-Headland coverage (row-skip) (D) Headland path (row-skip)

FIGURE 7.6: Illustration of the headland coverage process for both sequential and row-skip patterns.



(A) Complete coverage (sequential) (B) Exit path (sequential)

(C) Complete coverage (row-skip) (D) Exit path (row-skip)

FIGURE 7.7: Illustration of the complete coverage path and exit path for both sequential and row-skip patterns.

to the access segment towards the exterior of the field. A Dubins trajectory is then computed from the end of the path towards the closest point to it to complete the exit path. Fig.7.7d shows an example of this second case.

This process may result in overlaps in headlands, depending on the proximity of an access segment. In such cases, the first-level trajectory of some headlands may need to be traversed again to reach the nearest access segment. For example, as shown in Fig. 7.7b, the first-level trajectory of the lower headland is covered again to reach an access segment located on the right side of the field.

At the end of this process, the method generates two complete coverage paths, one with a sequential pattern and the other with a row-skip pattern, for each driving direction. These paths are added to the solution space for further evaluation and comparison.

7.2.5 Complex Fields with Sub-Polygons

For complex fields divided into sub-polygons using dividing lines, the proposed method explores all possible combinations of driving directions and coverage patterns for each sub-polygon. This involves generating all possible driving directions for each sub-polygon, which can result in a large number of combinations. For instance, if there are two sub-polygons and $\ell_s = 1$ degree, the method would consider all possible combinations of driving directions between 0 and 180 degrees, resulting in 180^2 potential combinations.

For each combination of driving directions, the method considers all possible orders of visiting the sub-polygons, as well as both sequential and row-skip coverage patterns for each sub-polygon. For instance, if there are two sub-polygons and both have at least one access segment for the robot to enter the field, the method would generate all possible combinations of visiting them in either order (*i.e.*, sub-polygon one followed by sub-polygon two, or sub-polygon two followed by sub-polygon one), and for each order, consider both sequential and row-skip coverage patterns for each sub-polygon. The number of solutions for covering all sub-polygons can be determined by Equation (7.1).

Given a combination of driving directions, the method begins with each sub-polygon that has at least one access segment and performs all previously detailed process for entering the first sub-polygon, generating and covering all its tracks with two different pattern, sequential and row-skip and covering its headlands. This results in two coverage paths for the first sub-polygon, one with sequential pattern and another with row-skip pattern. Afterwards, for each of these paths a circular search is performed similar to the search for finding the nearest access segment, but instead for finding the nearest uncovered sub-polygon. Therefore, the path is completed to reach the nearest uncovered sub-polygon.

After reaching the nearest uncovered sub-polygon, its tracks are generated and the short ones are excluded, following the process detailed in Section 7.2.2. Then, for each coverage pattern (sequential and row-skip) for the uncovered sub-polygon, there are four possible cases depending on which end of the first or last tracks is closest to the previous sub-polygon. After covering its headlands, the whole process is repeated for reaching the next uncovered sub-polygon and covering it, until all sub-polygons are fully covered.

Afterwards, if the last covered sub-polygon has an access segment, a circular search is performed to find the shortest exit path, following the process detailed in Section 7.2.4. Otherwise, a circular search is performed to find the nearest sub-polygon that has at least one access segment. After reaching it, another circular search is performed to find the shortest exit path.

After obtaining all possible solutions for a combination of driving directions, the method adds them to the solution space. Then, it proceeds to generate solutions for another combination of driving directions, repeating the process until all combinations have been considered.

7.2.6 Selection of the Optimal Solution

After generating all possible solutions and storing them in a solution space, the coverage rate S_{cov}, overlap rate S_{ovl}, non-working distance S_{nwd}, and operation time S_{otm} are computed for each solution, similar to the original CCPP. In addition, a new metric, the slope of working trajectories S_{slp}, is also computed and taken into consideration.

The computation of the slope metric for working trajectories of a solution involves discretizing each trajectory into a series of points with a spacing of ℓ_{slp}. The slope of each segment between two consecutive points is then calculated as the angle of the segment to the x-y plane by following equation:

$$SLP = \arctan(\frac{rise}{run}) \times \frac{180.0}{\pi} \tag{7.2}$$

where *rise* represents the difference in height between the two points, and *run* represents the distance between the two points along the x and y axis. The length of each segment is then cumulatively summed into the following slope categories:

- ***Flat*** **(ℓ_{s0})**: if $0° \leq SLP < 2°$ or $-2° < SLP \leq 0°$
- ***Slightly sloped*** **(ℓ_{s1})**: if $2° \leq SLP < 5°$ or $-5° < SLP \leq -2°$
- ***Sloped*** (ℓ_{s2}): if $5° \leq SLP < 8°$ or $-8° < SLP \leq -5°$
- ***Steep*** **(ℓ_{s3})**: if $8° \leq SLP < 12°$ or $-12° < SLP \leq -8°$
- ***Very steep*** **(ℓ_{s4})**: if $12° \leq SLP < 16°$ or $-16° < SLP \leq -12°$
- ***Extremely steep*** **(ℓ_{s5})**: if $SLP \geq 16°$ or $SLP \leq -16°$

where for a solution, ℓ_{s0}, ℓ_{s1}, ℓ_{s2}, ℓ_{s3}, ℓ_{s4}, and ℓ_{s5} represent the cumulative length of all segments that fall into each corresponding slope category. Afterwards, each of these values are normalized using the following Equation:

$$S = \frac{S - S_{min}}{S_{max} - S_{min}} \tag{7.3}$$

where S_{min} and S_{max} represent the minimum and maximum value of the corresponding slope category or metric over all solutions of the solution space.

For a solution, $\mathbf{L_s} = (\ell_{s0}\ \ell_{s1}\ \ell_{s2}\ \ell_{s3}\ \ell_{s4}\ \ell_{s5})$ and $\mathbf{W_s} = (W_{s0}\ W_{s1}\ W_{s2}\ W_{s3}\ W_{s4}\ W_{s5})$ are defined, where W_{s0}, W_{s1}, W_{s2}, W_{s3}, W_{s4}, and W_{s5} are weights of each slope category given as input. For a solution, S_{slp} is computed as follows:

$$S_{slp} = \frac{\mathbf{L_s W_s}^\intercal}{W_{s0} + W_{s1} + W_{s2} + W_{s3} + W_{s4} + W_{s5}} \tag{7.4}$$

For the computation of other metrics, we refer readers to Chapter 5, Section 5.2.5. After computing all metrics, they are normalized using Equation (7.3).

Based on these metrics, a set of soft constraints or cost functions, along with their corresponding weight given as input are defined as $\mathbf{C} = (1 - S_{cov}\ S_{ovl}\ S_{nwd}\ S_{otm}\ S_{slp})$ and $\mathbf{W} = (W_{cov}\ W_{ovl}\ W_{nwd}\ W_{otm}\ W_{slp})$, respectively. The final cost of each solution is computed by applying the following equation:

$$f = \frac{\mathbf{CW}^\intercal}{W_{cov} + W_{ovl} + W_{otm} + W_{nwd} + W_{slp}} \tag{7.5}$$

The solution with the lowest cost is selected by the approach, which combines various soft constraints and their corresponding weights. By adjusting the weights assigned to each soft constraint, as well as to each slope category, the algorithm can be customized to prioritize specific aspects of the path planning process, such as minimizing overlap or reducing non-working traveled distance. This allows users to tailor the algorithm to their unique needs and requirements.

It is worth mentioning that the slope categories were developed in consultation with an expert. However, it is important to note that they can be adjusted or modified to meet the specific needs and requirements of a particular operation.

7.3 Results and Discussion

7.3.1 Experimental Setup

Similar to the original approach, we developed a program in C++ with a GUI to set input parameters and visualize the resulting solutions. To speed up computations, we used an OpenMP [104] implementation that allows for parallel processing. The program was executed on the same hardware setup, an Intel Xeon(R) W-2135 CPU @ 3.70GHz × 12 with 32GB RAM.

To evaluate the effectiveness of the presented approach, we used the same dataset as the original approach, as introduced in Chapter 3 Section 3.3. Twenty fields of the dataset are considered as simple (Fields #1 - #20) with no field decomposition and ten other fields are considered as complex (Fields #21 - #30), where at least two dividing lines are provided for exploring different field decompositions. The dividing lines are visualized in Fig. 5.6.

We maintained the same parameters for the robot as in the original approach. However, some of the parameters used in the original approach, such as coverage, global overlap, local loop, and switch thresholds, were not relevant to the presented approach. Instead, we introduced new parameters, such as the driving direction step size (ℓ_s), the spacing of access segment discretization (ℓ_a) and the spacing of working trajectory discretization for slope computation (ℓ_{slp}). These parameters were respectively set to $3°$, $0.5m$, and $0.5m$. The values other parameters used for evaluating the effectiveness of our new approach are listed in Table 7.1.

Parameter	Description	Value	Parameter	Description	Value
w	working width	$3m$	W_{cov}	weight of S_{cov}	0.25
γ_{on}	minimum turning radius - implement on	$15m$	W_{ovl}	weight of S_{ovl}	0.10
γ_{off}	minimum turning radius - implement off	$1.5m$	W_{nwd}	weight of S_{nwd}	0.05
V_{on}	average speed - implement on	$3.5m/s$	W_{otm}	weight of S_{otm}	0.05
V_{gap}	average speed - implement transition	$2.5m/s$	W_{slp}	weight of S_{slp}	0.55
V_{off}	average speed - implement off	$1.5m/s$	W_{s0}	weight of ℓ_{s0}	0.00
ℓ_t	transition trajectory length	$2m$	W_{s1}	weight of ℓ_{s1}	0.10
ℓ_o	robot-implement offset	$2m$	W_{s2}	weight of ℓ_{s2}	0.15
Δ_{mwd}	minimum working distance threshold	$8m$	W_{s3}	weight of ℓ_{s3}	0.20
p	number of inner trajectories	2	W_{s4}	weight of ℓ_{s4}	0.25
g	number of gap-covering trajectories	1	W_{s5}	weight of ℓ_{s5}	0.30

TABLE 7.1: The input and parameters of the proposed approach

With the aim of providing comprehensive data for fellow researchers and showcasing the adaptability of our approach to a variety of robotic systems, at the time of writing this thesis, we have published the results (*i.e.* a way-point) under a different set of parameters on Zenodo [115], alongside the dataset used for evaluating our method. Moving forward, we intend to contribute additional results based on an increasingly diverse set of parameters and configurations.

In the following sections, we present an analytical and comparative evaluation of the presented approach against the original approach, covering the entire dataset. In addition, we provide an illustrative result to highlight some of the interesting features of the presented approach. However, we include a comprehensive visualization of the results for the entire dataset in the appendix to avoid overwhelming the main body of this chapter.

7.3.2 Analysis of the Results

As a reminder, the original approach clusters solutions into families based on the general direction of working trajectories. To ensure a fair comparison between the original approach (*O-CCPP*) and the presented approach (*H-CCPP*) on a field, we added the best solution from each family in O-CCPP that met the criteria described in Chapter 5, Section 5.3.1 to the solution space constructed by H-CCPP. The selection method of H-CCPP was then applied to the solution space to compute all costs and detect the best solution. This allowed for a comparison of all computed costs, including the slope cost, to be made between the two approaches under the same criteria. Although it would be more relevant to add all the solutions of O-CCPP to the solution space of H-CCPP, the large number of solutions obtained by the original approach in most cases makes it highly time-consuming to compute the slope cost.

Table 7.2 presents the average results obtained by O-CCPP and H-CCPP. The results for simple fields indicate that O-CCPP achieved a slightly higher coverage rate, while H-CCPP achieved a slightly lower overlap rate. For complex fields, O-CCPP performed slightly better in term of coverage and overlap rates. However, in terms of computational time, H-CCPP significantly faster than O-CCPP for both simple and complex fields, despite covering more driving directions and attempting two different coverage patterns (sequential and row-skip), as opposed to O-CCPP, which only employed sequential coverage pattern.

The radar charts depicted in Fig. 7.8 provide an overview of the performance of H-CCPP and O-CCPP using the criteria defined in Section 7.2.6. The charts indicate the number of fields in which each approach achieved a better score than the other

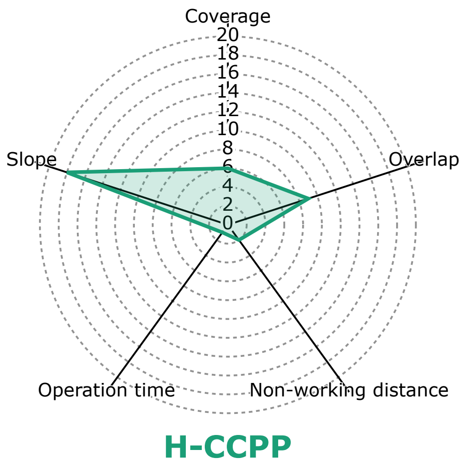
Approach	Area (ha)		Coverage		Overlap		Computational time (s)	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD
H-CCPP	4.87	2.82	98.30%	0.82%	2.56%	1.15%	3.12	1.86
O-CCPP	4.87	2.82	98.65%	0.63%	2.63%	1.35%	170.48	222.45

(A) Simple fields

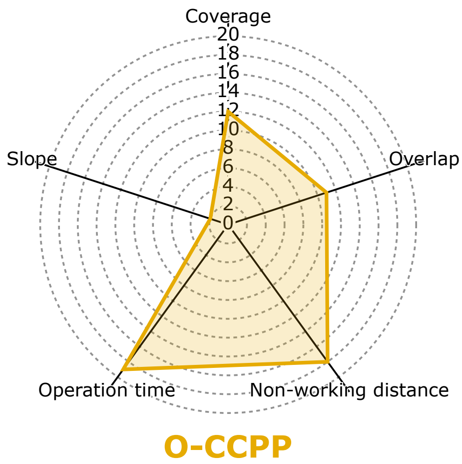
Approach	Area (ha)		Coverage		Overlap		Computational time (s)	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD
H-CCPP	4.69	2.41	97.48%	0.71%	3.56%	1.15%	1502.78	765.74
O-CCPP	4.69	2.41	98.04%	0.72%	2.55%	1.16%	5382.03	5077.03

(B) Complex fields

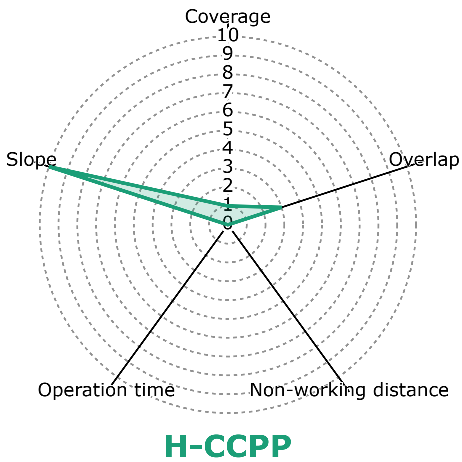
TABLE 7.2: Numerical results of the evaluation



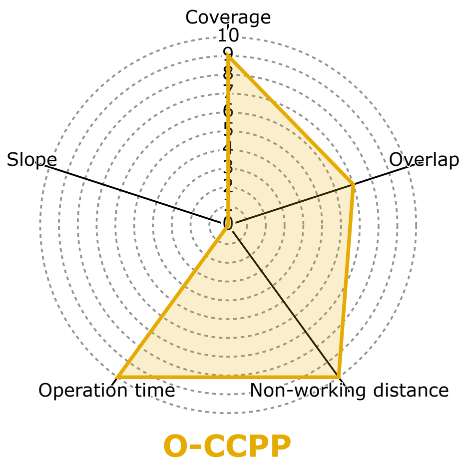
(A) Simple fields



(B) Simple fields



(C) Complex fields



(D) Complex fields

FIGURE 7.8: Radar chart comparing the performance of H-CCPP and O-CCPP on simple fields, showing the number of fields for which each approach achieved a better score for each soft constraint. Higher values for each criterion indicate better performance

approach for each criterion. The higher the value for each criterion, the better the performance of the corresponding approach in terms of minimizing the cost while considering its weight.

The radar charts reveal that H-CCPP outperformed O-CCPP in terms of slope cost for 18 out of 20 simple fields and for all complex fields. On the other hand, O-CCPP performed better in terms of worked area, overlaps, non-working traveled distance, and operation time, achieving better scores for 12, 11, 18, and 19 of the 20 simple fields, respectively, and for 9, 7, 10, and 10 of the 10 complex fields, respectively. This suggests that while H-CCPP is more effective in terms of slope cost, O-CCPP performs better in terms of other criteria such as worked area and overlaps, as well as non-working distance and operation time.

Despite O-CCPP's better performance on certain individual criteria, averaging all costs and computing the final cost using Equation (7.5) revealed that H-CCPP outperformed O-CCPP for 60% of the simple fields. For complex fields, each approach was efficient for 50% of cases. These results suggest that, on average, H-CCPP achieves better overall performance, even though O-CCPP may perform better on certain specific criteria.

In Fig.7.9, the results of applying H-CCPP and O-CCPP on Field #21 are presented, where the field is only accessible by its two left side edges. To provide a more detailed view of the outcomes, close-up 3D views are presented in Fig.7.10, which enable us to better observe the inclination and slope of trajectories.

The solution selected by H-CCPP has a row-skip pattern and did not include any field decomposition, as it was able to propose trajectories perpendicular to the slope of the field. This solution might be efficient in reducing energy consumption and preventing soil runoff caused by irrigation or rain. However, it resulted in more half-turns and consequently longer non-working traveled distances. Meanwhile, the selected solution by O-CCPP include a field decomposition that result in shorter non-working traveled distance.

The reason why O-CCPP did not select a solution similar to H-CCPP for Field #21 can be attributed to its limited capability to perform a row-skip pattern. This may lead to more overlaps as the robot attempts to reach an access segment to exit the field, thereby decreasing the overall efficiency of the solution.

It is important to note that changing the weight of the soft constraints can significantly affect the outcome of both approaches. For example, giving more weight to the cost of non-working traveled distance may lead H-CCPP to select a solution that include a field decomposition as well. However, for O-CCPP, it is not possible to improve its score in terms of slope cost. This is because the number of driving directions *i.e.*, the general direction, of results obtained by O-CCPP is limited to the directions of the field's edges. Moreover, since we have already included the best solution of all families *i.e.*, solutions with all possible driving directions for O-CCPP, in the solution space and have given a significant weight to slope cost, the solutions found by O-CCPP were not efficient in terms of slope cost. Hence, it could not be improved beyond the performance achieved in this evaluation.

In order to illustrate the versatility of H-CCPP in satisfying various criteria, we present the results of the selection method for Field #22 under different selection settings in the next section.

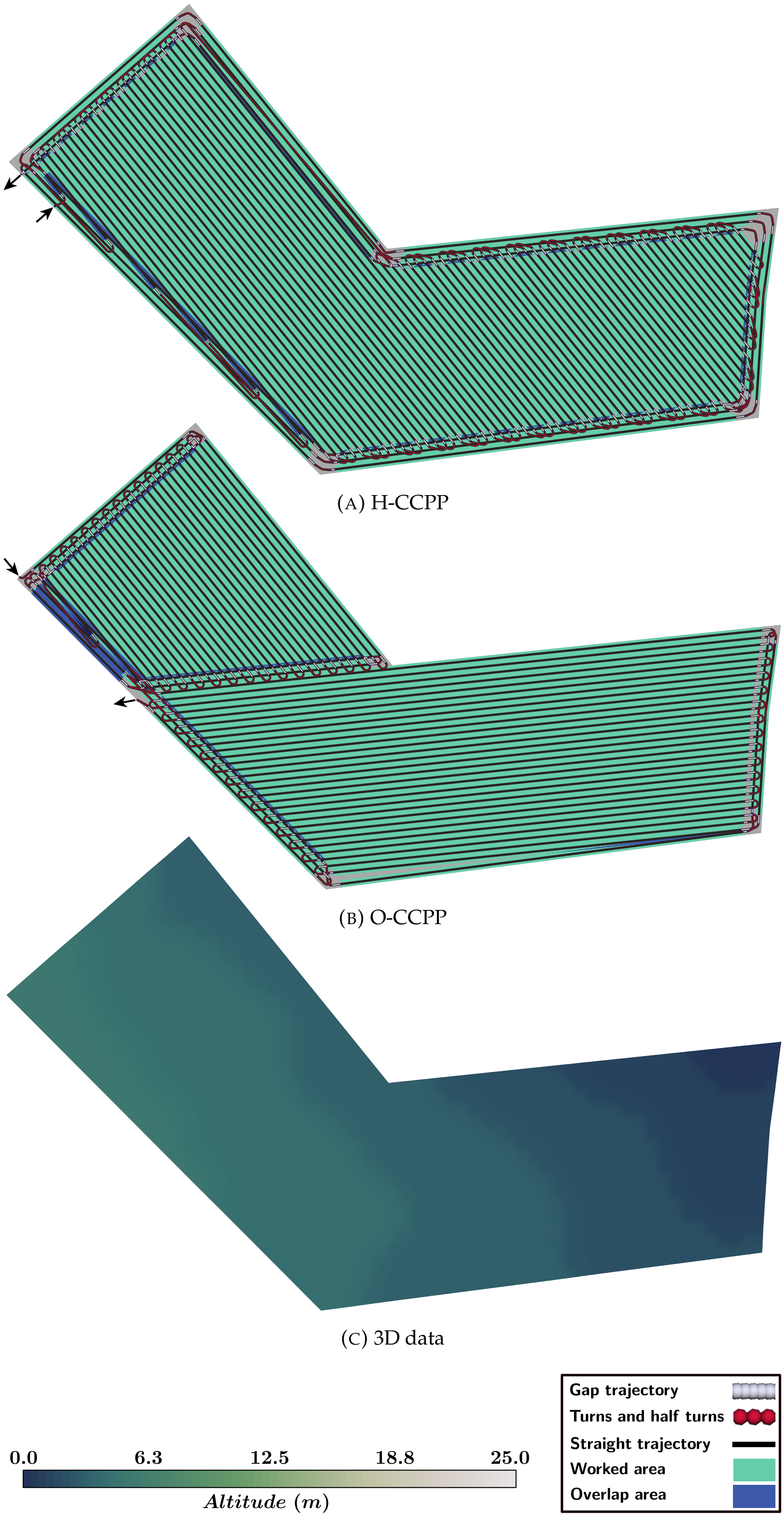
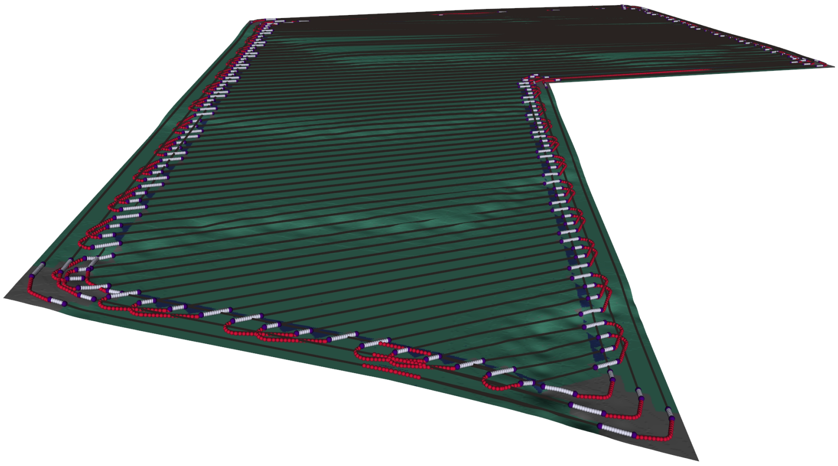
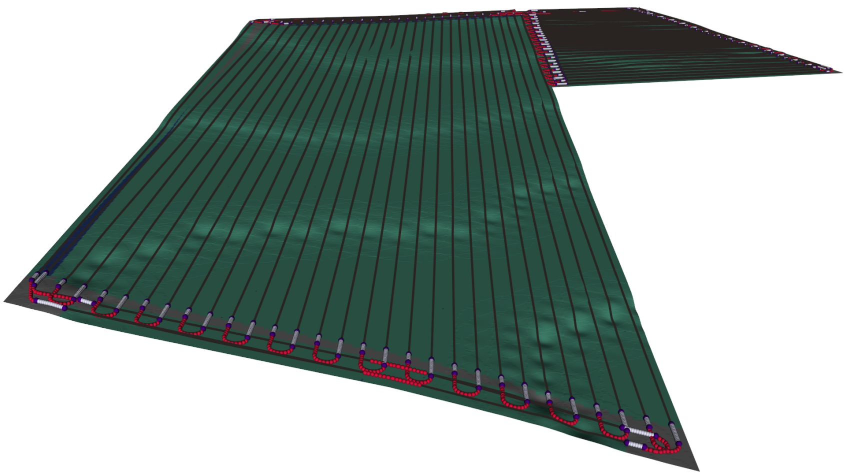


FIGURE 7.9: Illustration of the results obtained for Field #21 using H-CCPP and C-CCPP



(A) H-CCPP



(B) O-CCPP

FIGURE 7.10: Close-up views of the results obtained for Field #21 using H-CCPP and C-CCPP

7.3.3 H-CCPP under Different Optimization Settings

This section aims at illustrating the versatility of the proposed approach in generating solutions can satisfy various criteria. We conducted several tests on Field #22, emphasizing the weight of a particular soft constraint as the setting of the selection method. Table 7.3 presents the different settings of the selection method, while keeping the other parameters unchanged as given in Table 7.1. To enable a clear visual distinction between the solutions found under each selection setting, we adjusted the weight of one soft constraint to one and set the others to zero in each selection setting, as detailed in Table 7.3.

In the conducted tests on Field #22, the previously generated solution space for this field was utilized. Hence, the part of the approach responsible for constructing the solution space and computing all costs was executed only once, while only the final cost (computed by Equation 7.5) was competed three times, once per selection setting. Consequently, the computation time remained consistent for all tests.

Parameter	Description	Value
W_{cov}	weight of S_{cov}	1
W_{ovl}	weight of S_{ovl}	0
W_{nwd}	weight of S_{nwd}	0
W_{otm}	weight of S_{otm}	0
W_{slp}	weight of S_{slp}	0

(A) Selection Setting #1

Parameter	Description	Value
W_{cov}	weight of S_{cov}	0
W_{ovl}	weight of S_{ovl}	1
W_{nwd}	weight of S_{nwd}	0
W_{otm}	weight of S_{otm}	0
W_{slp}	weight of S_{slp}	0

(B) Selection Setting #2

Parameter	Description	Value
W_{cov}	weight of S_{cov}	0
W_{ovl}	weight of S_{ovl}	0
W_{nwd}	weight of S_{nwd}	1
W_{otm}	weight of S_{otm}	0
W_{slp}	weight of S_{slp}	0

(C) Selection Setting #3

TABLE 7.3: Different settings for the selection method of H-CCPP

Fig.7.11 illustrates the results obtained for each selection setting. The solution selected for Setting #1 (top image), where all weight was placed on the coverage cost, includes no field decomposition. This is primarily because decomposing the field into sub-polygons would result in more corners remaining uncovered and, thus, a lower coverage rate compared to solutions without field decomposition. However, it should be noted that improving this approach to make it able to cover corners during headland coverage by a reverse move might change this result.

The solution obtained for Setting #2 (middle image), where the weight was solely put on the overlap cost, is noteworthy. The selected solution entails a field decomposition, where a sequential pattern was used for one sub-polygon and a row-skip pattern for another. This is primarily due to the use of a row-skip pattern for the first sub-polygon, enabling minimal overlap when reaching the second sub-polygon.

In Setting #3, where all the weight was put on the non-working traveled distance cost, the selected solution (bottom image) included a field decomposition where the driving direction of each sub-polygon corresponded to its longest edge. This is not surprising, as selecting the longest edge as the driving direction would result in a lower number of half-turns, and consequently less non-working traveled distance.

Certainly, these tests demonstrated the flexibility and versatility of the proposed approach in generating solutions that can meet different combinations of constraints and criteria. By adjusting the weight of each soft constraint, the selection method was able to find different solutions that prioritize certain criteria over others. This ability to adapt and find a good compromise between conflicting criteria is particularly valuable in real-world applications, where multiple factors need to be considered and optimized simultaneously.

7.4 Conclusion

In this chapter, we presented a novel approach, H-CCPP, for generating efficient coverage paths for agricultural robots. We evaluated the effectiveness of the new approach by comparing it with the original approach, O-CCPP, using the same dataset introduced in Chapter 3, Section 3.3.

The evaluation revealed that H-CCPP is significantly faster than O-CCPP. For simple fields, the computational time is reduced on average from 170.48s to 3.12s, a reduction of approximately 98.17%. For complex fields, the computational time is reduced on average from 5382.03s to 1502.78s, a reduction of approximately 72.08%. This substantial reduction in computation time offers notable advantages such as faster decision-making, more efficient resource utilization, and the ability to adapt to evolving field conditions.

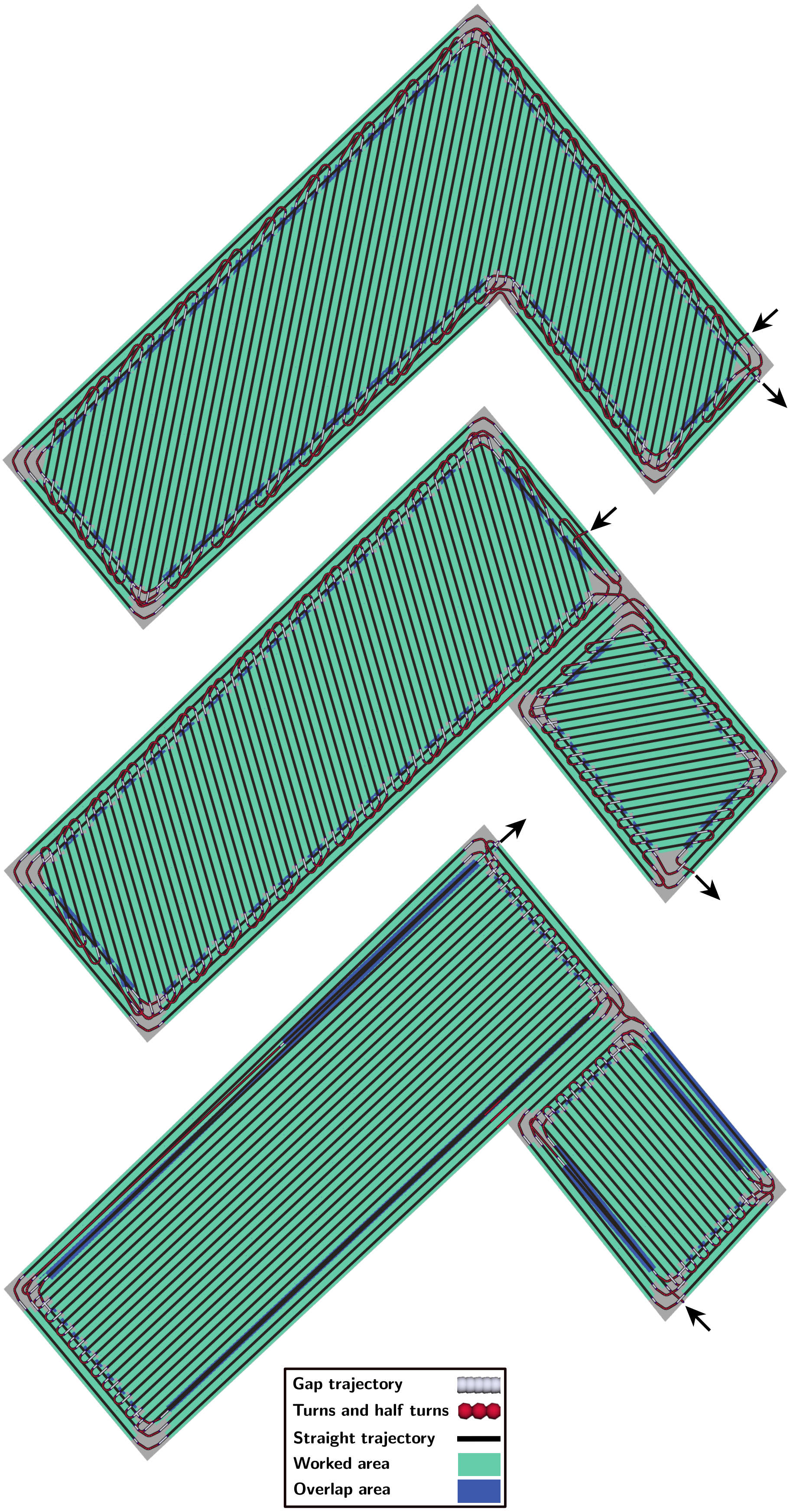


FIGURE 7.11: Illustration of the results obtained for Field #22 under different selection settings: Settings #1 to #3 (top to bottom)

Although H-CCPP achieved tremendously better performance in terms of minimizing the slope cost, O-CCPP performed slightly better in terms of worked area, overlaps, non-working traveled distance, and operation time. However, when computing the final cost, which considers all criteria, H-CCPP outperformed O-CCPP for 60% of simple fields and 50% of complex fields.

Additionally, H-CCPP provided several quantitative advantages over O-CCPP, enhancing its effectiveness in generating coverage paths. These advantages include examining various driving directions for parallel track generation inside the field, rather than relying solely on trajectories aligned with one of the field edges. H-CCPP also considered row-skip patterns for both simple and complex fields, examining all possible coverage pattern combinations for all its sub-fields. Furthermore, H-CCPP automatically determined both entry and exit points, as opposed to only determining the exit point based on a given entry point.

In addition to these advantages, we illustrated the versatility of H-CCPP in generating solutions that satisfy different combinations of constraints and criteria by conducting several tests on Field #22. By adjusting the weight of each soft constraint, the selection method was able to find different solutions that prioritize certain criteria over others.

Overall, the results showed that H-CCPP is an effective approach for generating coverage paths for agricultural robots, especially in terms of slope cost, and its flexibility and adaptability make it valuable in real-world applications where multiple factors need to be considered and optimized simultaneously. However, one potential area of improvement for H-CCPP is to make it able to cover corners during headland coverage by incorporating reverse moves. This enhancement may improve its performance in terms of coverage rate.

CHAPTER 8

Conclusion & Perspectives

The objective of this chapter is to synthesize and reflect upon the research conducted throughout this thesis, highlighting the major findings, contributions, and implications. To achieve this goal, we will first summarize the major findings and contributions that have arisen from our investigation, highlighting the innovative progress made in the development of efficient and comprehensive *Complete Coverage Path Planning (CCPP)* approaches for agricultural wheeled robots.

Furthermore, we will explore the practical implications of our findings, emphasizing the potential influence on agricultural practices. Additionally, we will acknowledge the limitations of our study and outline potential future research directions, offering suggestions on how the CCPP approaches developed in this thesis could be further refined, optimized, and expanded to address a wider range of scenarios and applications in the agriculture domain.

8.1 Major Findings and Contributions

Throughout this thesis, we have made numerous significant contributions that advance the state of the art in CCPP approaches, a key requirement for enabling efficient autonomous systems in agriculture. These noteworthy contributions and findings include:

- A systematic review of the challenges and proposed solutions for CCPP in wheeled agricultural robots, providing valuable insights into the current state of the art.
- The creation of a dataset containing 2D and 3D models of 30 fields located in France, which serves as a valuable resource for future research and development of path planning approaches.
- Development of an efficient CCPP approach that generates optimal coverage paths for autonomous agricultural robots, minimizing overlaps, non-working path length, and overall travel time.
- Exploration of a deep learning-based approach for field decomposition in agricultural CCPP, highlighting the challenges and complexities of the required data.

- Development of an advanced 3D hybrid path planning approach with multiple objectives, capable of considering trajectory inclinations and various other factors to optimize coverage rate, overlap rate, non-working traveled distance, and operation time.

By reflecting on the journey undertaken in this thesis and the knowledge gained through our research, we aim at providing a comprehensive and balanced overview of the work accomplished, as well as chart a path forward for continued exploration and advancement in the field of autonomous agricultural robotics.

8.2 Practical Implications and Applications

In this thesis, when studying the theory of complete coverage path planning and proposing algorithms, we always had in mind the practical implications and potential applications of our research findings. In this section, we highlight them for each major contribution and explore the opportunities they present for the agricultural industry.

8.2.1 Systematic Review and Open Dataset

The implications of the systematic review and the open dataset of 2D and 3D field models are closely intertwined, as they will both facilitate the development and validation of algorithms for the scientific community.

The systematic review of challenges and proposed solutions for CCPP for wheeled agricultural robots has practical implications for both researchers and the agricultural industry. For researchers, it serves as a comprehensive reference that highlights knowledge gaps and helps prevent duplication of effort, promoting collaboration in the field. For the agricultural industry, the review provides valuable insights into the most effective CCPP techniques and technologies, enabling stakeholders to make informed decisions about investing in and adopting autonomous agricultural robotics for various tasks.

Moreover, the dataset featuring 2D and 3D models of 30 diverse fields located in France is an invaluable resource that empowers researchers and technology developers in agricultural robotics to evaluate and validate path planning approaches across a broad spectrum of real-world scenarios and agricultural settings. The inclusion of both 2D and 3D terrain models not only fosters a deeper understanding of each modeling approach's strengths and weaknesses but also promotes their integration into other techniques, inspiring further research and development. Consequently, this may lead to the emergence of more precise and efficient algorithms for a range of applications, such as path planning, soil erosion analysis, and energy consumption estimation.

8.2.2 Advanced 3D Hybrid CCPP Approach

The advanced hybrid CCPP approach presented in this thesis offers several practical implications and applications. This novel method combines the strengths of previous CCPP approaches, efficiently addressing complex field shapes and headland coverage, while considering the robot characteristics such as width, the transition state of its implement, and the offset between the robot and its implement. Additionally, the approach considers field accessibility for automatically identifying entry and exit points. It examines two distinct coverage patterns (sequential and row-skip) and explores various driving directions from 0 to 180 degrees for the field or its sub-fields. Besides optimizing coverage rate, overlap rate, non-working traveled distance, and operation time, the approach also accounts for slope costs, which indirectly optimize soil erosion and energy consumption.

The hybrid CCPP approach is highly adaptable and flexible, enabling it to satisfy various combinations of constraints and criteria. By adjusting the weights of soft constraints, the selection method can prioritize specific criteria over others, providing tailored solutions for diverse agricultural scenarios.

With its faster processing times and improved performance, the hybrid CCPP approach offers enhanced efficiency for agricultural robots in real-world applications. This adaptability enables simultaneous consideration and optimization of multiple factors, making it a valuable tool for modern agriculture.

8.3 Limitations and Perspectives

Although the approaches that we have proposed in this manuscript have many advantages and interesting potential impact, they also have some limitations that may be the subject of future research. We present here a number of perspectives in an ordered manner, taking into account their relative difficulty and the time required for realization, allowing for prioritizing future research efforts in an efficient and effective manner.

8.3.1 Open Dataset

The dataset featuring 2D and 3D models of 30 diverse fields located in France has some limitations that could be addressed in future works. These limitations include the absence of fields with obstacles within them, the lack of deformable soil models that consider variations in soil quality, the limited variety of field sizes, shapes, and inclinations, and the narrow focus on fields in France.

To improve the dataset and address these limitations, future works could focus on several aspects. First, new fields that incorporate obstacles could be added, which would allow for a more comprehensive evaluation of the CCPP approaches in scenarios that closely resemble real-world agricultural settings, where various obstacles such as trees, rocks, or infrastructure may be present. Second, integrating deformable soil models that account for variations in soil quality over the fields would provide a more realistic representation of the agricultural landscape.

Moreover, the diversity of field sizes, shapes, and inclinations could be increased, providing a more robust foundation for developing and testing algorithms that can adapt to various constraints and conditions. The dataset could also be expanded to include fields from other countries using various data sources, possibly creating an open database with contributions from the wider research community. This international collaboration would not only increase the variety and number of fields but also promote a more comprehensive understanding of agricultural challenges and solutions.

In addition, fields could be assigned tags based on their properties, such as the presence of obstacles, complexity, or other features. This would enable researchers to extract specific subsets of fields for targeted testing on particular categories. Expanding the dataset to include fields with different crop types, such as vineyards or orchards, would also broaden the range of applications and provide more opportunities for innovation.

Ultimately, an improved and more comprehensive dataset would contribute to the progress of sustainable agriculture and the optimization of diverse agricultural practices. It would provide a basis for more realistic simulations that will drastically reduce the cost of on-field experiments, which could be extremely expensive. It would also facilitate advancements in other applications such as soil erosion analysis, energy consumption estimation, and other techniques in precision agriculture, leading to more robust and efficient solutions for autonomous agricultural robots.

8.3.2 CCPP Algorithms

Creating an autonomous system in agriculture requires the seamless integration of multiple specialized and interconnected subsystems, including path planning, path following, positioning systems, perception systems, and safety features. The result of our CCPP approach is a coverage path, which consists of a set of way-points containing information such as location, direction, and implement state. The planned path can be executed with the assistance of other subsystems.

Although our research has made significant progress in advancing the field of CCPP for agricultural wheeled robots, some limitations and areas for improvement still exist. In this section, we will discuss these limitations and propose potential directions for future research to enhance the capabilities of CCPP approaches, highlighting the subsystems that could be targeted for the suggested improvements.

Experimental Validation

The first limitation of our research is the lack of real-world experiments conducted on a field in conjunction with other subsystems. Despite the progress made by our research team in developing these subsystems, we could not perform field experiments due to the unavailability of a suitable testing environment.

Conducting real-world experiments would have allowed us to thoroughly evaluate the performance of our CCPP approaches in practical agricultural scenarios and identify any unforeseen challenges or limitations that may arise when integrated with other systems. For example, the reliability and accuracy of the location, or possible irregularities in the ground, may impact the precision with which the planned trajectory is actually followed. It may then be necessary to consider adjusting the planning in real time. Moreover, such experiments would have facilitated the fine-tuning and optimization of our algorithms, resulting in more accurate and reliable CCPP solutions.

Future research could focus on collaborations with agricultural institutions or organizations capable of investing in and acquiring suitable fields for testing purposes. By conducting experiments in real-world environments, researchers can better understand the complexities of implementing autonomous agricultural robots and further refine and optimize the CCPP approaches, ultimately leading to more effective and reliable autonomous operations.

Uncovered Field Corners

A limitation in our CCPP approach is the potential for uncovered corners in the field. While the current approach effectively covers headlands, it does not adequately address corner coverage when traveling from an inner trajectory of one headland to another.

The circular coverage pattern for headlands, implemented in our latest approach, presents an opportunity to address the issue of uncovered corners. This circular pattern is primarily utilized for reducing computational time, but can also be adapted to ensure comprehensive coverage of all field corners by considering the reverse move requirement for reaching and positioning the implement on the edge of the field before working an inner trajectory.

To tackle the issue of uncovered corners, future research could focus on refining the CCPP approach by incorporating strategies that ensure corner coverage during headland traversal. This may involve optimizing turning points and path transitions between headlands and their inner trajectories.

By enhancing the coverage of field corners and headlands, the overall efficiency and effectiveness of the CCPP approach can be improved, leading to more optimal solutions in terms of coverage rate.

Compatibility with Boom Section Control

In recent years, the advent of map-based automatic boom section control has seen widespread adoption by various manufacturers for agricultural sprayers. Leveraging global positioning system technology, this innovative solution tracks material application from previous sprayer passes. By dynamically accumulating and processing this data, the system independently controls boom sections, yielding material savings through the deactivation of sections in previously sprayed areas [83]. While this technology is primarily considered during execution, integrating it into the CCPP approach presents the notable benefit of delivering more accurate coverage and overlap rates estimation.

To enhance the compatibility of our CCPP approaches with boom section control technology, future research could focus on developing algorithms that efficiently incorporate section control information during the path planning process. By considering the real-time coverage status of the field and the activation or deactivation of implement sections, the final result of our approaches could be post processed to yield even more accurate coverage and overlap rates estimation.

2D Projection of 3D Surface and Trajectories

Another limitation of our research lies in the handling of 3D field surfaces and trajectories. Although our latest approach incorporates the 3D model of the field for computing the slope of working trajectories, we have not employed the 3D model for adjusting the spacing between parallel trajectories.

Integrating the 3D model for spacing adjustment can lead to more accurate and optimized trajectory planning, as it takes into account the variations in elevation and slope, which can significantly impact the efficiency and coverage of the agricultural operation. This approach can contribute to reducing missed areas or overlaps, ultimately leading to more efficient resource usage and better overall performance. On the other hand, fully integrating the 3D model for trajectory planning and spacing adjustment introduces additional computational complexity, which may result in increased computation times.

Future work could focus on improving CCPP approaches that efficiently incorporate 3D field models for trajectory planning and spacing adjustment, taking into consideration the trade-offs between computational complexity and solution efficiency. By addressing these challenges, the resulting CCPP solutions can offer enhanced performance and resource efficiency in various agricultural applications with complex terrain.

Obstacle Avoidance

Obstacle avoidance, particularly when dealing with static obstacles such as trees, rocks, or infrastructure is another limitation of our CCPP approaches. In many cases, proper field decomposition can effectively manage these static obstacles, allowing our approach to handle them without significant issues. However, in some situations, it may be more optimal to address these obstacles using innovative strategies during the path planning stage of the CCPP approach.

Dynamic obstacles, such as moving labors, or animals, on the other hand, require detection by the robot's perception system and management through the autonomous system's safety features. To handle such obstacles, several strategies may be useful, such as implementing a waiting strategy, informing the operator to resolve the issue, or even slightly deviating the trajectory when feasible, while maintaining the overall efficiency of the coverage path.

Incorporating obstacle avoidance techniques directly into the CCPP approach or other subsystems, such as perception and safety features, can lead to more efficient and adaptable solutions. This allows for better accommodation of both static and dynamic obstacles, potentially reducing the need for extensive field decomposition. Taking static obstacles into account during path planning enables the approach to dynamically adapt to the unique layout and constraints of the field, resulting in more optimal coverage paths. By refining the integration of these techniques, the overall adaptability and efficiency of autonomous agricultural robots can be further improved in various real-world scenarios and complex field environments.

Robustness and Computational Complexity

The robustness and computational complexity of the advanced CCPP approach, especially when applied to complex fields that require decomposition, can be mentioned as a current limitation. This issue could become more pronounced when real-time path recompilation is required to address evolving field conditions due to factors such as unpredictable weather events, soil moisture changes, or the presence of unexpected obstacles.

While the approach demonstrates high efficiency for simple fields without decomposition needs, taking just a few seconds, the computation time can increase substantially for complex fields that require decomposition, potentially taking several minutes. The complexity of the approach is directly and exponentially dependent on the number of sub-fields, leading to an increased computation time. This time may further extend if the approach is provided with multiple sets of dividing lines, as it must evaluate each set to identify the one that leads to the optimal solution.

This increase in complexity is primarily due to the exploration of all possible combinations of driving directions, coverage patterns for each sub-field, as well as their visiting order. The approach generates all potential driving directions for each sub-field and, for each combination of driving directions, considers all possible orders of visiting the sub-fields, as well as both sequential and skip-row coverage patterns for each sub-fields. This results in a substantial number of combinations to evaluate. As the number of sub-fields grows, the computational complexity of the algorithm increases exponentially, potentially affecting the overall efficiency of the CCPP solution.

To address this limitation, we propose a two-step solution as a perspective for enhancing the robustness of the proposed approach while maintaining the same quality of the final solution. The first step involves finding the optimal driving direction for each sub-field separately, instead of attempting all possible combinations. By isolating the driving direction optimization, we can significantly reduce the complexity of the problem. Once the optimal driving direction for each sub-field is determined, all possible combinations of coverage pattern in every sub-field, as well as the order of visiting them can be examined. This will reduce the number of possible combinations drastically.

However, before proceeding further, the efficiency of the resulting approach must first be determined through a comparison with our current approach. Once its efficiency is proven, the second step would involve generating a comprehensive dataset using the improved approach. This dataset would examine a set of dividing lines provided by an expert and identify which set of dividing lines leads to the optimal solution under the proposed optimization objectives for a large number of fields. This information can be used to create an annotated dataset indicating the optimal dividing lines, the optimal driving direction for each sub-field, and the optimization objectives.

By utilizing the resulting dataset, the deep learning-based approach proposed in this thesis can be refined. Once trained on the new dataset, the model can propose a set of dividing lines for optimally decomposing the field and determining the optimal driving direction for each sub-field while considering the optimization objectives. The information provided by the deep learning model can further improve the computation time of the CCPP approach developed in the first step. Moreover, it could lay the foundation for a fully AI-based CCPP approach, offering a more elegant and efficient solution to the challenges of agricultural path planning.

In conclusion, enhancing the robustness and computational efficiency of our CCPP approach not only addresses the current limitations but also allows for the inclusion of additional factors such as soil erosion and energy consumption directly using more complex models. Furthermore, it would enable the incorporation of new constraints such as soil compaction through the use of soil-wheel interaction models and controlled traffic farming practices that track and provide information about previously executed paths on the field [55, 145].

Robot capacity

One more limitation to consider is the robot's capacity, both in terms of agricultural materials and fuel or energy levels. Efficiently managing these resources is crucial to ensure the robot can effectively cover the field without interruptions or delays.

One potential solution could be achieved by incorporating a post-processing method into our latest approach. This approach focuses on first covering the main part of the field before addressing the headlands. This strategy offers an advantage, as the circular headlands surrounding the field can be used for recharging or reloading the robot. By planning a path within the headlands towards a stationary service station or even incorporating a secondary mobile service unit, the primary robot can be recharged or resupplied as needed.

Once the main part of the field is covered, a similar strategy can be employed using the remaining uncovered headlands. Additionally, if the robot is near an access segment and its current energy or material level is insufficient to complete the job, a mobile service unit can be deployed to recharge or resupply the robot, ensuring it can finish its task without disruptions.

By incorporating these strategies into the CCPP approach, the system can better address capacity constraints, ultimately leading to more efficient and reliable autonomous agricultural robots.

Multi-Robot Systems

Our research primarily focuses on CCPP approaches for single-robot systems. However, many real-world scenarios involve deploying multiple robots that cooperatively complete tasks, necessitating coordination and collaboration among them. Incorporating multi-robot systems into CCPP approaches can result in more efficient and effective solutions, as robots can cover different areas of the field simultaneously, reducing operation time and mitigating the impact of individual robot failures.

A potential solution for integrating multi-robot systems is to divide the coverage path generated by our latest approach into several shorter segments, preferably at points where turns or half-turns occur in headlands and turning spaces. These shorter segments can then be allocated to multiple robots. During this process, it is essential to consider the robots' capacity to ensure that each segment can be covered by a fully charged robot without requiring recharging or reloading.

Future work could investigate the development of CCPP approaches that incorporate multi-robot systems, addressing challenges such as inter-robot communication, task allocation, and collision avoidance. The emergence of 5G connectivity could play a crucial role in enabling real-time communication and coordination among multiple robots, significantly enhancing their collaborative capabilities. Implementing these systems may necessitate improvements in perception and safety features. Furthermore, a management system, possibly leveraging IoT and 5G technology, could be essential for allocating, controlling, and re-planning tasks, or even for alerting the operator if a robot is unable to complete its assignment on time or experiences a malfunction. This would promote more efficient resource utilization and enhance overall performance in large-scale agricultural operations.

8.3.3 Fully Deep Learning-Based Approaches

At the time of writing this thesis, there is no deep learning-based approach to agricultural CCPP. Our exploration towards a deep learning-based approach for field decomposition in agricultural CCPP represented one of the first steps in this direction, highlighting the challenges and complexities associated with acquiring and annotating training data. This exploration emphasizes the need for a comprehensive dataset that accurately captures diverse field shapes from real-world scenarios.

As a long-term perspective, future research could focus on leveraging deep learning not only for field decomposition but also for learning and reproducing coverage paths. The first step toward such an approach would be to create a comprehensive dataset that includes various field shapes, annotated field decompositions, and the coverage paths along with the objectives that lead to those paths.

It is important to consider that identifying the objectives that lead to the coverage path can be a challenging task. Currently, most of the coverage paths are generated by farmers and are executed by manually or semi-manually operated tractors. As a result, the factors influencing the coverage paths may not only depend on primary goals such as coverage rate, overlaps, soil erosion, soil compaction, energy consumption, and environmental pollution, but also on farmers' preferences, which can vary from one individual to another. Furthermore, the physical demand and driving skills of the farmers can play a significant role in shaping the coverage paths.

However, when it comes to fully autonomous robots, factors that depend on individual farmers become irrelevant. Autonomous robots do not experience fatigue while performing difficult maneuvers, nor do they rely on visual cues to approximately locate the next track to follow. Instead, these robots can consistently execute optimal coverage paths, taking into account the primary objectives and constraints.

Given these considerations, creating a comprehensive dataset could be achieved by extracting data from various fields needed by latest CCPP approaches, with the help of an expert to determine the best or the most pertinent set of dividing lines for field decompositions, to generate full coverage paths under various optimization criterion. These coverage paths, as well as their objectives could serve as a good reference for deep learning models.

Potential models that could be explored for training include generative models such as variational autoencoders [73] and generative adversarial networks [46], which can learn the underlying structure of the data and generate new, plausible coverage paths. Additionally, reinforcement learning approaches [160] could be employed to learn optimal policies for navigating complex fields while considering multiple objectives.

In conclusion, the integration of deep learning approaches in agricultural CCPP holds great promise for the future of autonomous farming. By building upon the foundation laid in this thesis and advancing the development of comprehensive dataset, researchers can explore a wide range of deep learning models to create innovative solutions for complex agricultural scenarios. These solutions will not only improve the efficiency and sustainability of agricultural practices, but also contribute to meeting the growing global food demand while minimizing environmental impacts.

With the groundwork laid by the research presented in this thesis, it is our hope to inspire further exploration and innovation in the realm of autonomous agriculture, ultimately fostering a brighter and more sustainable future for agriculture. By embracing interdisciplinary collaboration and incorporating cutting-edge technology, researchers can drive significant progress in autonomous agricultural robotics, revolutionizing the way we approach agricultural challenges and ushering in a new era of smart farming.

List of publications

Articles published or accepted for publication:

- Danial Pour Arab, Matthias Spisser, and Caroline Essert. “Complete Coverage Path Planning for wheeled agricultural robots”. In: *Journal of Field Robotics* (Nov. 2022). Accepted, to be published, preprint available in the Wiley Open Research collection on Authorea. DOI: 10.22541/au.166869706.64844882/v1
- Danial Pour Arab, Matthias Spisser, and Caroline Essert. “Introduction of a Skip-Row pattern in complete coverage path planning for agricultural fields”. In: *International Conference on Automation, Robotics and Applications (ICARA)*. Abu Dhabi, United Arab Emirates: IEEE, Jan. 2023

Published open source datasets:

- Danial Pour Arab, Matthias Spisser, and Caroline Essert. *Agricultural Fields 2D and 3D Models Dataset*. Version 1.0.0. This work is funded by T&S - Technology and Strategy Strasbourg and ANRT (Association Nationale de la Recherche et de la Technologie). Apr. 2023. DOI: 10.5281/zenodo.7805321

Articles under revision:

- Danial Pour Arab et al. “A survey on Complete Coverage Path Planning for wheeled agricultural robots”. In: *Journal of Field Robotics* (). Submitted to the Special Issue on Agricultural Robots for Ag 4.0: Building intelligent data-collectors and autonomous workers for smart farms. Currently at 1st revision stage.

Articles being written:

- Danial Pour Arab, Matthias Spisser, and Caroline Essert. “Advanced 3D Hybrid Path Planning with Multiple Objectives for Complete Coverage of Agricultural Fields”. In: (). Writing stage.

Bibliography

[1] Shyam P Adhikari, Gookhwan Kim, and Hyongsuk Kim. "Deep neural network-based system for autonomous navigation in paddy field". In: *IEEE Access* 8 (2020), pp. 71272–71278.

[2] United States Department of Agriculture. *Farms and Land in Farms 2021*. https://www.nass.usda.gov.

[3] Uzair Ahmad, Abozar Nasirahmadi, Oliver Hensel, and Stefano Marino. "Technology and data fusion methods to enhance site-specific crop monitoring". In: *Agronomy* 12.3 (2022), p. 555.

[4] Bernard Ijesunor Akhigbe, Kamran Munir, Olugbenga Akinade, Lukman Akanbi, and Lukumon O Oyedele. "IoT technologies for livestock management: a review of present status, opportunities, and future trends". In: *Big data and cognitive computing* 5.1 (2021), p. 10.

[5] Randa Almadhoun, Tarek Taha, Lakmal Seneviratne, and Yahya Zweiri. "A survey on multi-robot coverage path planning for model reconstruction and mapping". In: *SN Applied Sciences* 1.8 (2019), p. 847.

[6] Athanasios Anagnostis, Aristotelis C Tagarakis, Dimitrios Kateris, Vasileios Moysiadis, Claus Grøn Sørensen, Simon Pearson, and Dionysis Bochtis. "Orchard mapping with deep learning semantic segmentation". In: *Sensors* 21.11 (2021), p. 3813.

[7] QGIS Association. *QGIS Geographic Information System*. http://www.qgis.org.

[8] Héctor Azpúrua, Gustavo M Freitas, Douglas G Macharet, and Mario FM Campos. "Multi-robot coverage path planning using hexagonal segmentation for geophysical surveys". In: *Robotica* 36.8 (2018), pp. 1144–1166.

[9] B Benet, R Lenain, and V Rousseau. "Development of a sensor fusion method for crop row tracking operations". In: *Advances in Animal Biosciences* 8.2 (2017), pp. 583–589.

[10] Chris Bluett, Jeff N Tullberg, John E McPhee, and Diogenes L Antille. "Soil and Tillage Research: Why still focus on soil compaction?" In: *Soil and Tillage Research* 194 (2019), pp. 1–2.

[11] DD Bochtis, CG Sørensen, Patrizia Busato, IA Hameed, E Rodias, O Green, and G Papadakis. "Tramline establishment in controlled traffic farming based on operational machinery cost". In: *Biosystems Engineering* 107.3 (2010), pp. 221–231.

[12] DD Bochtis, CG Sørensen, and SG Vougioukas. "Path planning for in-field navigation-aiding of service units". In: *Computers and Electronics in Agriculture* 74.1 (2010), pp. 80–90.

[13] DD Bochtis and Claus G Sørensen. "The vehicle routing problem in field logistics part I". In: *Biosystems engineering* 104.4 (2009), pp. 447–457.

[14] DD Bochtis and Claus G Sørensen. "The vehicle routing problem in field logistics: Part II". In: *Biosystems engineering* 105.2 (2010), pp. 180–188.

[15] Dionysis D Bochtis, Claus G Sørensen, Patrizia Busato, and Remigio Berruto. "Benefits from optimal route planning based on B-patterns". In: *Biosystems Engineering* 115.4 (2013), pp. 389–395.

[16] Andrea Botta, Paride Cavallone, Lorenzo Baglieri, Giovanni Colucci, Luigi Tagliavini, and Giuseppe Quaglia. "A review of robots, perception, and tasks in precision agriculture". In: *Applied Mechanics* 3.3 (2022), pp. 830–854.

[17] Mernout Burger, Marco Huiskamp, and Tamás Keviczky. "Complete field coverage as a multi-vehicle routing problem". In: *IFAC Proceedings Volumes* 46.18 (2013), pp. 97–102.

[18] Taua M Cabreira, Carmelo Di Franco, Paulo R Ferreira, and Giorgio C Buttazzo. "Energy-aware spiral coverage path planning for uav photogrammetric applications". In: *IEEE Robotics and automation letters* 3.4 (2018), pp. 3662–3668.

[19] Yi Cao, Yu Han, Jian Chen, Xuzan Liu, Zichao Zhang, and Kai Zhang. "A Tractor Formation Coverage Path Planning Method Based on Rotating Calipers and Probabilistic Roadmaps Algorithm". In: *2019 IEEE International Conference on Unmanned Systems and Artificial Intelligence (ICUSAI)*. IEEE. 2019, pp. 125–130.

[20] Yi Cao, Yu Han, Jian Chen, Xuzan Liu, Zichao Zhang, and Kai Zhang. "Optimal Coverage Path Planning Algorithm of the Tractor-formation Based on Probabilistic Roadmaps". In: *2019 IEEE International Conference on Unmanned Systems and Artificial Intelligence (ICUSAI)*. IEEE. 2019, pp. 27–32.

[21] Christophe Cariou, Zoltan Gobor, Benjamin Seiferth, and Michel Berducat. "Mobile robot trajectory planning under kinematic and dynamic constraints for partial and full field coverage". In: *Journal of Field Robotics* 34.7 (2017), pp. 1297–1312.

[22] Christophe Cariou, Jean Laneurit, Jean-Christophe Roux, and Roland Lenain. "Multi-robots trajectory planning for farm field coverage". In: *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE. 2020, pp. 351–356.

[23] Christophe Cariou, Jean-Christophe Roux, and Roland Lenain. "Laser beam deflection of a 2D LiDAR for canopy detection on an autonomous spraying robot". In: *2021 7th International Conference on Automation, Robotics and Applications (ICARA)*. IEEE. 2021, pp. 80–84.

[24] The Intergovernmental Panel on Climate Change. *Climate Change 2021: The Physical Science Basis*. https://www.ipcc.ch/report/ar6/wg1/. 2021.

[25] *Combined Powers*. https://combined-powers.com.

[26] Jesus Conesa-Muñoz, Jose M Bengochea-Guevara, Dionisio Andujar, and Angela Ribeiro. "Efficient distribution of a fleet of heterogeneous vehicles in agriculture: a practical approach to multi-path planning". In: *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*. IEEE. 2015, pp. 56–61.

[27] Jesus Conesa-Muñoz, José María Bengochea-Guevara, Dionisio Andujar, and Angela Ribeiro. "Route planning for agricultural tasks: A general approach for fleets of autonomous vehicles in site-specific herbicide applications". In: *Computers and Electronics in Agriculture* 127 (2016), pp. 204–220.

[28] Jesus Conesa-Muñoz, Gonzalo Pajares, and Angela Ribeiro. "Mix-opt: A new route operator for optimal coverage path planning for a fleet in an agricultural environment". In: *Expert Systems with Applications* 54 (2016), pp. 364–378.

[29] José Correa, Johannes A Postma, Michelle Watt, and Tobias Wojciechowski. "Soil compaction and the architectural plasticity of root systems". In: *Journal of experimental botany* 70.21 (2019), pp. 6019–6034.

[30] Edsger W Dijkstra. "A note on two problems in connexion with graphs". In: *Edsger Wybe Dijkstra: His Life, Work, and Legacy*. 2022, pp. 287–290.

[31] Sedat Dogru and Lino Marques. "Energy efficient coverage path planning for autonomous mobile robots on 3D terrain". In: *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*. IEEE. 2015, pp. 118–123.

[32] Sedat Dogru and Lino Marques. "Improved Coverage Path Planning Using a Virtual Sensor Footprint: a Case Study on Demining". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 4410–4415.

[33] Sedat Dogru and Lino Marques. "Towards fully autonomous energy efficient coverage path planning for autonomous mobile robots on 3D terrain". In: *2015 European Conference on Mobile Robots (ECMR)*. IEEE. 2015, pp. 1–6.

[34] Zanliang Dong and Chunhui Liu. "Collaborative coverage path planning of UAV cluster based on deep reinforcement learning". In: *2021 IEEE 3rd International Conference on Frontiers Technology of Information and Computer (ICFTIC)*. IEEE. 2021, pp. 201–207.

[35] Lester E Dubins. "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents". In: *American Journal of mathematics* 79.3 (1957), pp. 497–516.

[36] Yael Edan, Shufeng Han, and Naoshi Kondo. "Automation in agriculture". In: *Springer handbook of automation*. Springer, 2009, pp. 1095–1128.

[37] Gareth TC Edwards, Jørgen Hinge, Nick Skou-Nielsen, Andrés Villa-Henriksen, Claus Aage Grøn Sørensen, and Ole Green. "Route planning evaluation of a prototype optimised infield route planner for neutral material flow agricultural operations". In: *Biosystems Engineering* 153 (2017), pp. 149–157.

[38] Dhivya Elavarasan and PM Durairaj Vincent. "Crop yield prediction using deep reinforcement learning model for sustainable agrarian applications". In: *IEEE access* 8 (2020), pp. 86886–86901.

[39] Eurostat. *Agriculture statistics - family farming in the EU*. https://ec.europa.eu/eurostat.

[40] John T Evans IV, Santosh K Pitla, Joe D Luck, and Michael Kocher. "Row crop grain harvester path optimization in headland patterns". In: *Computers and Electronics in Agriculture* 171 (2020), p. 105295.

[41] Jaime Pulido Fentanes, Iain Gould, Tom Duckett, Simon Pearson, and Grzegorz Cielniak. "3-D soil compaction mapping through kriging-based exploration with a mobile robot". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3066–3072.

[42] Food and Agricultural Organization of the United Nations. *The State of Food and Agriculture 2021*. https://www.fao.org/documents/card/en/c/cb4476en. 2021.

[43] Enric Galceran and Marc Carreras. "A survey on coverage path planning for robotics". In: *Robotics and Autonomous systems* 61.12 (2013), pp. 1258–1276.

[44] François Gauthier-Clerc, Ashley Hill, Jean Laneurit, Roland Lenain, and Eric Lucet. "Online velocity fluctuation of off-road wheeled mobile robots: A

reinforcement learning approach". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 2421–2427.

[45] Géoportail. *The national portal for territorial knowledge implemented by IGN*. https://www.geoportail.gouv.fr/.

[46] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[47] IA Hameed, DD Bochtis, CG Sørensen, and M Nøremark. "Automated generation of guidance lines for operational field planning". In: *Biosystems engineering* 107.4 (2010), pp. 294–306.

[48] Ibrahim A Hameed. "Coverage path planning software for autonomous robotic lawn mower using Dubins' curve". In: *2017 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE. 2017, pp. 517–522.

[49] Ibrahim A Hameed. "Intelligent coverage path planning for agricultural robots and autonomous machines on three-dimensional terrain". In: *Journal of Intelligent & Robotic Systems* 74.3-4 (2014), pp. 965–983.

[50] Ibrahim A Hameed, DD Bochtis, and CG Sorensen. "Driving angle and track sequence optimization for operational path planning using genetic algorithms". In: *Applied Engineering in Agriculture* 27.6 (2011), pp. 1077–1086.

[51] Ibrahim A Hameed, Anders la Cour-Harbo, and Ottar L Osen. "Side-to-side 3D coverage path planning approach for agricultural robots to minimize skip/overlap areas between swaths". In: *Robotics and Autonomous Systems* 76 (2016), pp. 36–45.

[52] Peter E Hart, Nils J Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths". In: *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107.

[53] Mahdi Hassan, Dikai Liu, and Xiang Chen. "Squircular-CPP: A Smooth Coverage Path Planning Algorithm based on Squircular Fitting and Spiral Path". In: *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE. 2020, pp. 1075–1081.

[54] H Ben Hassen, A Elaoud, and K Masmoudi. "Modeling of agricultural soil compaction using discrete Bayesian networks". In: *International Journal of Environmental Science and Technology* (2020), pp. 1–12.

[55] P Heritier, D Miclet, E Piron, M Chanet, R Lenain, et al. "Tools and methods to develop and validate soil-wheel interaction model and knowledge". In: *1st AXEMA-EurAgEng Conference" Intensive and environmentally friendly agriculture: an opportunity for innovation in machinery and systems", Paris, France, 25 February 2017*. European Society of Agricultural Engineers. 2017, pp. 68–77.

[56] Ashley Hill, Jean Laneurit, Roland Lenain, and Eric Lucet. "Online Gain Tuning Using Neural Networks: A Comparative Study". In: *AgriEngineering* 4.4 (2022), pp. 1200–1211.

[57] IGN. *L'image géographique du territoire national, la France vue du ciel*. https://geoservices.ign.fr/bdortho.

[58] IGN. *REST API of IGN for determining the elevation of one or more points*. https://geoservices.ign.fr/documentation/services/api-et-services-ogc/calcul-altimetrique-rest.

[59] IGN. *The French National Institute of Geographic and Forestry Information*. https://geoservices.ign.fr.

[60] IGN. *Une base de données géographiques servant de référence à l'instruction des aides de la politique agricole commune*. https://geoservices.ign.fr/rpg.

[61] Mohammad Samunul Islam and Golap Kanti Dey. “Precision agriculture: renewable energy based smart crop field monitoring and management system using WSN via IoT”. In: *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)*. IEEE. 2019, pp. 1–6.

[62] Martin Andreas Falk Jensen, Dionysis Bochtis, Claus Grøn Sørensen, Morten Rufus Blas, and Kasper Lundberg Lykkegaard. “In-field and inter-field path planning for agricultural transport units”. In: *Computers & Industrial Engineering* 63.4 (2012), pp. 1054–1061.

[63] Martin F Jensen, Dionysis Bochtis, and Claus G Sørensen. “Coverage planning for capacitated field operations, part II: Optimisation”. In: *Biosystems Engineering* 139 (2015), pp. 149–164.

[64] Martin F Jensen, Michael Nørremark, Patrizia Busato, Claus G Sørensen, and Dionysis Bochtis. “Coverage planning for capacitated field operations, Part I: Task decomposition”. In: *Biosystems Engineering* 139 (2015), pp. 136–148.

[65] Chan-Woo Jeon, Hak-Jin Kim, Changho Yun, Xiongzhe Han, and Jung Hun Kim. “Design and validation testing of a complete paddy field-coverage path planner for a fully autonomous tillage tractor”. In: *Biosystems Engineering* 208 (2021), pp. 79–97.

[66] Yuxiong Ji, Lantao Ni, Cong Zhao, Cailin Lei, Yuchuan Du, and Wenshuo Wang. “TriPField: A 3D potential field model and its applications to local path planning of autonomous vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* (2023).

[67] Jian Jin and Lie Tang. “Coverage path planning on three-dimensional terrain for arable farming”. In: *Journal of field robotics* 28.3 (2011), pp. 424–440.

[68] Jian Jin and Lie Tang. “Optimal coverage path planning for arable farming on 2D surfaces”. In: *Transactions of the ASABE* 53.1 (2010), pp. 283–295.

[69] *John Deere*. https://www.deere.fr.

[70] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. “Probabilistic roadmaps for path planning in high-dimensional configuration spaces”. In: *IEEE transactions on Robotics and Automation* 12.4 (1996), pp. 566–580.

[71] Amin Khajepour, Majid Sheikhmohammady, and Ehsan Nikbakhsh. “Field path planning using capacitated arc routing problem”. In: *Computers and Electronics in Agriculture* 173 (2020), p. 105401.

[72] Amna Khan, Iram Noreen, and Zulfiqar Habib. “On Complete Coverage Path Planning Algorithms for Non-holonomic Mobile Robots: Survey and Challenges.” In: *J. Inf. Sci. Eng.* 33.1 (2017), pp. 101–121.

[73] Diederik P Kingma, Max Welling, et al. “An introduction to variational autoencoders”. In: *Foundations and Trends® in Machine Learning* 12.4 (2019), pp. 307–392.

[74] Earl J Kirkland. “Bilinear interpolation”. In: *Advanced Computing in Electron Microscopy*. Springer, 2010, pp. 261–263.

[75] Anirudh Krishna Lakshmanan, Rajesh Elara Mohan, Balakrishnan Ramalingam, Anh Vu Le, Prabahar Veerajagadeshwar, Kamlesh Tiwari, and Muhammad Ilyas. “Complete coverage path planning using reinforcement learning for tetromino based cleaning and maintenance robot”. In: *Automation in Construction* 112 (2020), p. 103078.

[76] Steven M LaValle and James J Kuffner. “Rapidly-exploring random trees: Progress and prospects: Steven m. lavalle, iowa state university, a james j. kuffner, jr., university of tokyo, tokyo, japan”. In: *Algorithmic and computational robotics* (2001), pp. 303–307.

[77] Tingjun Lei, Chaomin Luo, Gene Eu Jan, and Zhuming Bi. "Deep learning-based complete coverage path planning with re-joint and obstacle fusion paradigm". In: *Frontiers in Robotics and AI* 9 (2022).

[78] Roland Lenain, Benoit Thuilot, Christophe Cariou, and Philippe Martinet. "Model predictive control for vehicle guidance in presence of sliding: application to farm vehicles path tracking". In: *Proceedings of the 2005 IEEE international conference on robotics and automation*. IEEE. 2005, pp. 885–890.

[79] Dengshan Li, Rujing Wang, Chengjun Xie, Liu Liu, Jie Zhang, Rui Li, Fangyuan Wang, Man Zhou, and Wancai Liu. "A recognition method for rice plant diseases and pests video detection based on deep convolutional neural network". In: *Sensors* 20.3 (2020), p. 578.

[80] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. "Image inpainting for irregular holes using partial convolutions". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 85–100.

[81] Guilin Liu, Kevin J Shih, Ting-Chun Wang, Fitsum A Reda, Karan Sapra, Zhiding Yu, Andrew Tao, and Bryan Catanzaro. "Partial convolution based padding". In: *arXiv preprint arXiv:1811.11718* (2018).

[82] Jun Liu and Xuewei Wang. "Tomato diseases and pests detection based on improved Yolo V3 convolutional neural network". In: *Frontiers in plant science* 11 (2020), p. 898.

[83] JD Luck, SK Pitla, SA Shearer, TG Mueller, CR Dillon, JP Fulton, and SF Higgins. "Potential for pesticide and nutrient savings via map-based automatic boom section control of spray nozzles". In: *Computers and Electronics in Agriculture* 70.1 (2010), pp. 19–26.

[84] Sandeep Manjanna, Alberto Quattrini Li, Ryan N Smith, Ioannis Rekleitis, and Gregory Dudek. "Heterogeneous multi-robot system for exploration and strategic water sampling". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1–8.

[85] Vittorio Mazzia, Aleem Khaliq, and Marcello Chiaberge. "Improvement in land cover and crop classification based on temporal features learning from Sentinel-2 data using recurrent-convolutional neural network (R-CNN)". In: *Applied Sciences* 10.1 (2019), p. 238.

[86] Vittorio Mazzia, Francesco Salvetti, Diego Aghi, and Marcello Chiaberge. "DeepWay: A Deep Learning waypoint estimator for global path generation". In: *Computers and Electronics in Agriculture* 184 (2021), p. 106091.

[87] Gonzalo Mier, João Valente, and Sytze de Bruin. "Fields2Cover: An open-source coverage path planning library for unmanned agricultural vehicles". In: *arXiv preprint arXiv:2210.07838* (2022).

[88] Umakanta Mishra, Abu Hashan Md Mashud, Ming-Lang Tseng, and Jei-Zheng Wu. "Optimizing a sustainable supply chain inventory model for controllable deterioration and emission rates in a greenhouse farm". In: *Mathematics* 9.5 (2021), p. 495.

[89] Elsayed Said Mohamed, AA Belal, Sameh Kotb Abd-Elmabod, Mohammed A El-Shirbeny, A Gad, and Mohamed B Zahran. "Smart farming for improving agricultural management". In: *The Egyptian Journal of Remote Sensing and Space Science* 24.3 (2021), pp. 971–981.

[90] Vasileios Moysiadis, Panagiotis Sarigiannidis, Vasileios Vitsas, and Adel Khelifi. "Smart farming in Europe". In: *Computer science review* 39 (2021), p. 100345.

[91] MA Viraj J Muthugala, SM Bhagya P Samarakoon, and Mohan Rajesh Elara. “Toward energy-efficient online Complete Coverage Path Planning of a ship hull maintenance robot based on Glasius Bio-inspired Neural Network”. In: *Expert Systems with Applications* 187 (2022), p. 115940.

[92] *Naïo*. https://www.naio-technologies.com.

[93] The United Nations. *The 2022 Revision of World Population Prospects*. https://population.un.org/wpp. 2022.

[94] Hassan Nehme, Clément Aubry, Thomas Solatges, Xavier Savatier, Romain Rossi, and Rémi Boutteau. “Lidar-based structure tracking for agricultural robots: Application to autonomous navigation in vineyards”. In: *Journal of Intelligent & Robotic Systems* 103 (2021), pp. 1–16.

[95] *Nexus Robotic*. https://nexusrobotics.ca.

[96] René S Nilsson and Kun Zhou. “Method and bench-marking framework for coverage path planning in arable farming”. In: *Biosystems Engineering* 198 (2020), pp. 248–265.

[97] David Nistér, Oleg Naroditsky, and James Bergen. “Visual odometry for ground vehicle applications”. In: *Journal of Field Robotics* 23.1 (2006), pp. 3–20.

[98] DongKi Noh, WooJu Lee, Hyoung-Rock Kim, Il-Soo Cho, In-Bo Shim, and SeungMin Baek. “Adaptive coverage path planning policy for a cleaning robot with deep reinforcement learning”. In: *2022 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE. 2022, pp. 1–6.

[99] Michael Nørremark, René Søndergaard Nilsson, and Claus Aage Grøn Sørensen. “In-field route planning optimisation and performance indicators of grain harvest operations”. In: *Agronomy* 12.5 (2022), p. 1151.

[100] Timo Oksanen et al. *Path planning algorithms for agricultural field machines*. Helsinki University of Technology, 2007.

[101] Timo Oksanen and Arto Visala. “Coverage path planning algorithms for agricultural field machines”. In: *Journal of field robotics* 26.8 (2009), pp. 651–668.

[102] Margaret A Oliver and Richard Webster. “Kriging: a method of interpolation for geographical information systems”. In: *International Journal of Geographical Information System* 4.3 (1990), pp. 313–332.

[103] Yuki Onishi, Takeshi Yoshida, Hiroki Kurita, Takanori Fukao, Hiromu Arihara, and Ayako Iwai. “An automated fruit harvesting robot by using deep learning”. In: *Robomech Journal* 6.1 (2019), pp. 1–8.

[104] OpenMP. *The OpenMP API supports multi-platform shared-memory parallel programming in C/C++ and Fortran*. https://www.openmp.org/.

[105] J Padmapriya and T Sasilatha. “Deep learning based multi-labelled soil classification and empirical estimation toward sustainable agriculture”. In: *Engineering Applications of Artificial Intelligence* 119 (2023), p. 105690.

[106] Yue Pan, Yu Han, Lin Wang, Jian Chen, Hao Meng, Guangqi Wang, Zichao Zhang, and Shubo Wang. “3D Reconstruction of Ground Crops Based on Airborne LiDAR Technology”. In: *IFAC-PapersOnLine* 52.24 (2019), pp. 35–40.

[107] White PaperdJuly. “ESRI shapefile technical description”. In: *Comput. Stat* 16 (1998), pp. 370–371.

[108] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. “Automatic differentiation in pytorch”. In: (2017).

[109] Francis J Pierce and Peter Nowak. “Aspects of precision agriculture”. In: *Advances in agronomy* 67 (1999), pp. 1–85.

[110] Mogens Graf Plessen. "Freeform Path Fitting for the Minimisation of the Number of Transitions between Headland Path and Interior Lanes within Agricultural Fields". In: *arXiv preprint arXiv:1910.12034* (2019).

[111] Mogens Graf Plessen. "Optimal in-field routing for full and partial field coverage with arbitrary non-convex fields and multiple obstacle areas". In: *Biosystems Engineering* 186 (2019), pp. 234–245.

[112] Mogens M Graf Plessen. "Partial field coverage based on two path planning patterns". In: *Biosystems engineering* 171 (2018), pp. 16–29.

[113] Philip Polack, Florent Altché, Brigitte d'Andréa-Novel, and Arnaud de La Fortelle. "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" In: *2017 IEEE intelligent vehicles symposium (IV)*. IEEE. 2017, pp. 812–818.

[114] Ciro Potena, Daniele Nardi, and Alberto Pretto. "Fast and accurate crop and weed identification with summarized train sets for precision agriculture". In: *Intelligent Autonomous Systems 14: Proceedings of the 14th International Conference IAS-14 14*. Springer. 2017, pp. 105–121.

[115] Danial Pour Arab, Matthias Spisser, and Caroline Essert. *Agricultural Fields 2D and 3D Models Dataset*. Version 1.0.0. This work is funded by T&S - Technology and Strategy Strasbourg and ANRT (Association Nationale de la Recherche et de la Technologie). Apr. 2023. DOI: 10.5281/zenodo.7805321.

[116] Danial Pour Arab, Matthias Spisser, and Caroline Essert. "Complete Coverage Path Planning for wheeled agricultural robots". In: *Journal of Field Robotics* (Nov. 2022). Accepted, to be published, preprint available in the Wiley Open Research collection on Authorea. DOI: 10.22541/au.166869706.64844882/v1.

[117] Danial Pour Arab, Matthias Spisser, and Caroline Essert. "Introduction of a Skip-Row pattern in complete coverage path planning for agricultural fields". In: *International Conference on Automation, Robotics and Applications (ICARA)*. Abu Dhabi, United Arab Emirates: IEEE, Jan. 2023.

[118] Rekha Raja, Thuy T Nguyen, David C Slaughter, and Steven A Fennimore. "Real-time weed-crop classification and localisation technique for robotic weed control in lettuce". In: *Biosystems Engineering* 192 (2020), pp. 257–274.

[119] Reenul Reedha, Eric Dericquebourg, Raphael Canals, and Adel Hafiane. "Transformer neural network for weed and crop classification of high resolution UAV images". In: *Remote Sensing* 14.3 (2022), p. 592.

[120] James Reeds and Lawrence Shepp. "Optimal paths for a car that goes both forwards and backwards". In: *Pacific journal of mathematics* 145.2 (1990), pp. 367–393.

[121] Kenneth G Renard. *Predicting soil erosion by water: a guide to conservation planning with the Revised Universal Soil Loss Equation (RUSLE)*. United States Government Printing, 1997.

[122] GF Ricci, J Jeong, AM De Girolamo, and Francesco Gentile. "Effectiveness and feasibility of different management practices to reduce soil erosion in an agricultural watershed". In: *Land Use Policy* 90 (2020), p. 104306.

[123] Niles Ritter, Mike Ruth, Brett Borup Grissom, George Galang, John Haller, Gary Stephenson, Steve Covington, Tim Nagy, Jamie Moyers, Jim Stickley, et al. "Geotiff format specification geotiff revision 1.0". In: *SPOT Image Corp* 1 (2000).

[124] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *Medical Image Computing*

and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. Springer. 2015, pp. 234–241.

[125] Scott D Roth. "Ray casting for modeling solids". In: *Computer graphics and image processing* 18.2 (1982), pp. 109–144.

[126] EJ Sadler, RiG Evans, KC Stone, and CR Camp. "Opportunities for conservation with precision irrigation". In: *Journal of soil and water conservation* 60.6 (2005), pp. 371–378.

[127] Luís Santos, Filipe Santos, Jorge Mendes, Pedro Costa, José Lima, Ricardo Reis, and Pranjali Shinde. "Path planning aware of robot's center of mass for steep slope vineyards". In: *Robotica* 38.4 (2020), pp. 684–698.

[128] Luís Santos, Filipe N Santos, Sandro Magalhães, Pedro Costa, and Ricardo Reis. "Path planning approach with the extraction of topological maps from occupancy grid maps in steep slope vineyards". In: *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2019, pp. 1–7.

[129] Luís C Santos, Filipe N Santos, EJ Solteiro Pires, António Valente, Pedro Costa, and Sandro Magalhães. "Path Planning for ground robots in agriculture: a short review". In: *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2020, pp. 61–66.

[130] Luís Carlos Santos, André Silva Aguiar, Filipe Neves Santos, António Valente, and Marcelo Petry. "Occupancy Grid and Topological Maps Extraction from Satellite Images for Path Planning in Agricultural Robots". In: *Robotics* 9.4 (2020), p. 77.

[131] Lus Santos, Nuno Ferraz, Filipe Neves dos Santos, Jorge Mendes, Raul Morais, Pedro Costa, and Ricardo Reis. "Path planning aware of soil compaction for steep slope vineyards". In: *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2018, pp. 250–255.

[132] W Schroeder, K Martin, and B Lorensen. "The visualization toolkit, 4th edn. Kitware". In: *New York* (2006).

[133] Michael Gomez Selvaraj, Alejandro Vergara, Henry Ruiz, Nancy Safari, Sivalingam Elayabalan, Walter Ocimati, and Guy Blomme. "AI-powered banana diseases and pest detection". In: *Plant Methods* 15 (2019), pp. 1–11.

[134] Hasan Seyyedhasani and Joseph S Dvorak. "Dynamic rerouting of a fleet of vehicles in agricultural operations through a Dynamic Multiple Depot Vehicle Routing Problem representation". In: *biosystems engineering* 171 (2018), pp. 63–77.

[135] Hasan Seyyedhasani and Joseph S Dvorak. "Reducing field work time using fleet routing optimization". In: *Biosystems Engineering* 169 (2018), pp. 1–10.

[136] Hasan Seyyedhasani and Joseph S Dvorak. "Using the Vehicle Routing Problem to reduce field completion times with multiple machines". In: *Computers and Electronics in Agriculture* 134 (2017), pp. 142–150.

[137] Hasan Seyyedhasani, Joseph S Dvorak, and Eric Roemmele. "Routing algorithm selection for field coverage planning based on field shape and fleet size". In: *Computers and electronics in agriculture* 156 (2019), pp. 523–529.

[138] Sagarika Sharma, Sujit Rai, and Narayanan C Krishnan. "Wheat crop yield prediction using deep LSTM model". In: *arXiv preprint arXiv:2011.01498* (2020).

[139] Mengwei Shen, Suzhen Wang, Shuang Wang, and Yan Su. "Simulation Study on Coverage Path Planning of Autonomous Tasks in Hilly Farmland Based

on Energy Consumption Model". In: *Mathematical Problems in Engineering* 2020 (2020).

[140] Donald Shepard. "A two-dimensional interpolation function for irregularly-spaced data". In: *Proceedings of the 1968 23rd ACM national conference*. 1968, pp. 517–524.

[141] Iddo Shnaps and Elon Rimon. "Online coverage of planar environments by a battery powered autonomous mobile robot". In: *IEEE Transactions on Automation Science and Engineering* 13.2 (2016), pp. 425–436.

[142] Jie Sun, Liping Di, Ziheng Sun, Yonglin Shen, and Zulong Lai. "County-level soybean yield prediction using deep CNN-LSTM model". In: *Sensors* 19.20 (2019), p. 4363.

[143] Qilin Sun, Ruirui Zhang, Liping Chen, Linhuan Zhang, Hongming Zhang, and Chunjiang Zhao. "Semantic segmentation and path planning for orchards based on UAV images". In: *Computers and Electronics in Agriculture* 200 (2022), p. 107222.

[144] Ziheng Sun, Liping Di, and Hui Fang. "Using long short-term memory recurrent neural network in land cover classification on Landsat and Cropland data layer time series". In: *International journal of remote sensing* 40.2 (2019), pp. 593–614.

[145] Tseganesh Wubale Tamirat, Søren Marcus Pedersen, Robert John Farquharson, Sytze de Bruin, Patrick Dermot Forristal, Claus Grøn Sørensen, David Nuyttens, Hans Henrik Pedersen, and Maria Nygård Thomsen. "Controlled traffic farming and field traffic management: Perceptions of farmers groups from Northern and Western European countries". In: *Soil and Tillage Research* 217 (2022), p. 105288.

[146] Greg Turk. "The ply polygon file format". In: *Recuperado de* (1994).

[147] Amalia Utamima, Torsten Reiners, and Amir H Ansaripoor. "Evolutionary estimation of distribution algorithm for agricultural routing planning in field logistics". In: *Procedia Computer Science* 161 (2019), pp. 560–567.

[148] Amalia Utamima, Torsten Reiners, and Amir H Ansaripoor. "Optimisation of agricultural routing planning in field logistics with Evolutionary Hybrid Neighbourhood Search". In: *Biosystems Engineering* 184 (2019), pp. 166–180.

[149] Mahdi Vahdanjoo, Christian Toft Madsen, and Claus Grøn Sørensen. "Novel Route Planning System for Machinery Selection. Case: Slurry Application". In: *AgriEngineering* 2.3 (2020), pp. 408–429.

[150] Mahdi Vahdanjoo and Claus G Sorensen. "Novel route planning method to improve the operational efficiency of capacitated operations. Case: Application of organic fertilizer". In: *AgriEngineering* 3.3 (2021), pp. 458–477.

[151] Mahdi Vahdanjoo, Kun Zhou, and Claus Aage Grøn Sørensen. "Route Planning for Agricultural Machines with Multiple Depots: Manure Application Case Study". In: *Agronomy* 10.10 (2020), p. 1608.

[152] CA Van Doren, RS Stauffer, and EH Kidder. "Effect of Contour Farming on Soil Loss and Runoff 1". In: *Soil Science Society of America Journal* 15.C (1951), pp. 413–417.

[153] Thomas Van Klompenburg, Ayalew Kassahun, and Cagatay Catal. "Crop yield prediction using machine learning: A systematic literature review". In: *Computers and Electronics in Agriculture* 177 (2020), p. 105709.

[154] Juan P Vasconez, George A Kantor, and Fernando A Auat Cheein. "Human–robot interaction in agriculture: A survey and current challenges". In: *Biosystems engineering* 179 (2019), pp. 35–48.

[155] David Vieira, Rodolfo Orjuela, Matthias Spisser, and Michel Basset. "Longitudinal vehicle control based on off-road tire model for soft soil applications". In: *IFAC-PapersOnLine* 54.10 (2021), pp. 304–309.

[156] David Vieira, Rodolfo Orjuela, Matthias Spisser, and Michel Basset. "Positioning and Attitude determination for Precision Agriculture Robots based on IMU and Two RTK GPSs Sensor Fusion". In: *IFAC-PapersOnLine* 55.32 (2022), pp. 60–65.

[157] Andrés Villa-Henriksen, Gareth TC Edwards, Liisa A Pesonen, Ole Green, and Claus Aage Grøn Sørensen. "Internet of Things in arable farming: Implementation, applications, challenges and potential". In: *Biosystems engineering* 191 (2020), pp. 60–84.

[158] Yinchu Wang, Zhi He, Dandan Cao, Li Ma, Kai Li, Liangsheng Jia, and Yongjie Cui. "Coverage path planning for kiwifruit picking robots based on deep reinforcement learning". In: *Computers and Electronics in Agriculture* 205 (2023), p. 107593.

[159] Thomas Weingartshofer, Bernhard Bischof, Martin Meiringer, Christian Hartl-Nesic, and Andreas Kugi. "Optimization-based path planning framework for industrial manufacturing processes with complex continuous paths". In: *Robotics and Computer-Integrated Manufacturing* 82 (2023), p. 102516.

[160] Marco A Wiering and Martijn Van Otterlo. "Reinforcement learning". In: *Adaptation, learning, and optimization* 12.3 (2012), p. 729.

[161] Chenming Wu, Chengkai Dai, Xiaoxi Gong, Yong-Jin Liu, Jun Wang, Xianfeng David Gu, and Charlie CL Wang. "Energy-efficient coverage path planning for general terrain surfaces". In: *IEEE Robotics and Automation Letters* 4.3 (2019), pp. 2584–2591.

[162] Xiaolong Wu, Stéphanie Aravecchia, Philipp Lottes, Cyrill Stachniss, and Cédric Pradalier. "Robotic weed control using automated weed and crop classification". In: *Journal of Field Robotics* 37.2 (2020), pp. 322–340.

[163] Ya Xiong, Yuanyue Ge, Lars Grimstad, and Pål J From. "An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation". In: *Journal of Field Robotics* 37.2 (2020), pp. 202–224.

[164] H Murat Yilmaz. "The effect of interpolation methods in surface definition: an experimental study". In: *Earth Surface Processes and Landforms: The Journal of the British Geomorphological Research Group* 32.9 (2007), pp. 1346–1361.

[165] Alexander Zelinsky, Ray A Jarvis, JC Byrne, Shinichi Yuta, et al. "Planning paths of complete coverage of an unstructured environment by a mobile robot". In: *Proceedings of international conference on advanced robotics*. Vol. 13. 1993, pp. 533–538.

[166] Haojie Zhang, Yudong Zhang, and Tiantian Yang. "A survey of energy-efficient motion planning for wheeled mobile robots". In: *Industrial Robot: the international journal of robotics research and application* (2020).

[167] Haisen Zhao, Fanglin Gu, Qi-Xing Huang, Jorge Garcia, Yong Chen, Changhe Tu, Bedrich Benes, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. "Connected fermat spirals for layered fabrication". In: *ACM Transactions on Graphics (TOG)* 35.4 (2016), pp. 1–10.

[168] Kun Zhou and Dionysis Bochtis. "Route Planning For Capacitated Agricultural Machines Based On Ant Colony Algorithms." In: *HAICTA*. 2015, pp. 163–173.

[169] Kun Zhou, Allan Leck Jensen, Dionysis Bochtis, Michael Nørremark, Dimitrios Kateris, and Claus Grøn Sørensen. "Metric Map Generation for Autonomous Field Operations". In: *Agronomy* 10.1 (2020), p. 83.

[170] Guoyu Zuo, Peng Zhang, and Junfei Qiao. “Path planning algorithm based on sub-region for agricultural robot”. In: *2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR 2010)*. Vol. 2. IEEE. 2010, pp. 197–200.

Université de Strasbourg

Danial POUR ARAB

msii ÉCOLE DOCTORALE

Calcul et optimisation de trajectoires pour véhicules autonomes soumis aux contraintes d'un environnement agricole

"Trajectory optimisation for autonomous vehicles under agricultural environment constraints"

Résumé

Dans cette thèse, nous développons une approche générique et efficace pour la planification de couverture complète de trajectoires (CCPP) pour les robots agricoles autonomes. Nous visons à concevoir des méthodes de CCPP qui tiennent compte des exigences spécifiques de chaque opération et outil, ainsi que des caractéristiques 3D du champ, tout en optimisant plusieurs objectifs tels que la couverture, les chevauchements, la distance non travaillée, la consommation d'énergie et l'érosion des sols. Les contributions incluent une revue systématique des algorithmes de CCPP, la modélisation du terrain, la création d'un ensemble de données et le développement d'approches efficaces pour la planification des trajectoires.

Mots-clés : Planification de Trajectoire pour une Couverture Complète, Robots Agricoles Autonomes, Modélisation du Terrain, Recherche Intelligente Basée sur les Arborescences, Apprentissage Profond, Décomposition des Parcelles, Planification de Trajectoire en 3D.

Résumé en anglais

This thesis focuses on developing a generic and efficient Complete Coverage Path Planning (CCPP) approach for autonomous agricultural robots. The research addresses the complexities of various agricultural operations, considering the specific requirements of each operation and implement, along with the 3D characteristics of the field. The thesis contributions include a systematic review of CCPP algorithms, a detailed discussion of 2D and 3D terrain modeling with a dataset of 30 fields in France, development of an efficient tree-based intelligent search algorithm for CCPP, exploration of deep learning-based field decomposition, and the creation of an advanced 3D hybrid path planning approach with multiple objectives. The ultimate goal is to contribute to the advancement of CCPP approaches for autonomous agricultural robots and provide a foundation for further research in this area.

Keywords: Complete Coverage Path Planning, Autonomous Agricultural Robots, Terrain Modeling, Tree-based Intelligent Search, Deep Learning, Field Decomposition, 3D Hybrid Path Planning.

APPENDIX A

Résumé en Français (Version Longue)

A.1 Introduction

Les systèmes agro-alimentaires ont été mis à rude épreuve par la pandémie de COVID-19, le changement climatique et la croissance démographique, entraînant une insécurité alimentaire et une malnutrition accrues à l'échelle mondiale. Selon le Groupe d'experts intergouvernemental sur l'évolution du climat, le secteur agricole devrait être gravement touché par le réchauffement climatique d'ici 2040. De plus, la population mondiale devrait atteindre 10,9 milliards d'ici 2100, nécessitant une augmentation de la production alimentaire. Pour relever ces défis, des méthodes agricoles efficaces, durables et résilientes sont nécessaires. Les systèmes autonomes, tels que les robots autonomes à roues, offrent des perspectives d'amélioration de l'efficacité et de réduction des coûts en agriculture. Cependant, des défis importants subsistent dans le développement et le déploiement de ces systèmes autonomes pour améliorer l'agriculture de précision et l'agriculture intelligente.

L'agriculture de précision utilise la technologie de l'information, les outils d'analyse de données et les capteurs pour optimiser la production agricole grâce à la prise de décision basée sur les données. Son objectif est d'augmenter les rendements, de minimiser les impacts environnementaux et d'améliorer la durabilité de l'agriculture. L'agriculture intelligente, une évolution de l'agriculture de précision, emploie des technologies avancées telles que la robotique, l'automatisation et l'Internet des objets (IoT) pour optimiser davantage les opérations agricoles. Elle exploite les données de diverses sources pour la prise de décision en temps réel et la gestion précise des activités agricoles.

Les principales applications de ces pratiques comprennent l'équipement agricole autonome, la surveillance et la gestion des cultures, la gestion du bétail, l'irrigation de précision, la prédiction des rendements et l'optimisation de la chaîne d'approvisionnement.

Développer un système autonome pour l'agriculture implique d'intégrer plusieurs sous-systèmes, tels que la planification de trajectoire, le suivi de trajectoire, les systèmes de positionnement, les systèmes de perception et les fonctionnalités de sécurité.

La planification de trajectoire consiste à créer un itinéraire réalisable et optimal d'un point à un autre tout en tenant compte des contraintes environnementales et du véhicule. Elle peut également impliquer la génération d'un chemin assurant une couverture complète du champ. Le suivi de trajectoire, en revanche, consiste à adhérer avec précision au chemin prévu à l'aide de diverses techniques de contrôle et des dernières avancées en matière de contrôle de direction [78, 155, 44, 56].

Un positionnement précis est essentiel, et les Systèmes Globaux de Navigation par Satellite (GNSS) fournissent des informations de position. Cependant, d'autres systèmes de positionnement tels que l'odométrie visuelle, les unités de mesure inertielle ou la localisation basée sur LiDAR peuvent être nécessaires dans certains scénarios où les performances du GNSS sont dégradées [97, 156, 94].

Les systèmes de perception permettent de comprendre l'environnement en utilisant plusieurs capteurs et algorithmes pour la détection, la segmentation et le suivi des objets [23, 114, 16]. Une perception robuste repose sur des méthodes de fusion de capteurs qui intègrent les données de plusieurs capteurs [3, 9].

Les fonctionnalités de sécurité, y compris l'évitement d'obstacles, la détection de collisions et les mécanismes d'arrêt d'urgence, sont essentielles pour protéger les travailleurs humains et les autres véhicules à proximité du système autonome. Ces fonctionnalités sont généralement réalisées grâce à des algorithmes avancés utilisant les données des capteurs [154].

Les systèmes agricoles autonomes peuvent être adaptés à partir de machines agricoles traditionnelles en les équipant de capteurs, d'actionneurs et de contrôleurs pour des opérations manuelles et autonomes. Cette adaptation offre des avantages tels que l'utilisation de l'équipement et de l'infrastructure existants et la possibilité de passer d'un mode à l'autre selon les besoins. En alternative, des robots autonomes personnalisés peuvent être spécialement conçus pour des tâches et des environnements particuliers. Ces robots, plus petits et plus agiles, sont excellents pour naviguer sur des terrains difficiles. Dans cette étude, tous les véhicules autonomes sont désignés sous le terme de "robots autonomes", avec un accent mis sur ceux conçus pour des tâches spécifiques.

Le Laboratoire d'Innovation de Technology & Strategy à Strasbourg a développé un prototype de robot équipé de divers capteurs, dont un LiDAR, une caméra RGB-D et un GNSS. Le robot intègre des algorithmes sophistiqués pour le suivi de trajectoire, la navigation, la détection d'obstacles et la sécurité. Sa conception permet une navigation autonome sur un terrain déformable. Cette thèse se concentre principalement sur le sous-système de planification de trajectoire du robot, en développant spécifiquement une approche de Planification de Trajectoire de Couverture Complète (PTCC) pour un robot autonome équipé d'un outil spécifique à une tâche. L'objectif est de créer un algorithme de planification adapté à une utilisation sur le terrain réel, en conjonction avec d'autres sous-systèmes développés et améliorés par l'équipe, pour démontrer l'efficacité et la praticité de l'approche proposée.

Comme mentionné précédemment, développer un système autonome en agriculture est une tâche complexe qui implique d'intégrer plusieurs sous-systèmes spécialisés et interconnectés. L'intégration réussie de ces sous-systèmes est cruciale pour le fonctionnement efficace et efficient du système sur le terrain. De plus, les exigences de chaque sous-système peuvent varier en fonction de l'opération agricole spécifique effectuée. De plus, différentes pratiques agricoles, telles que l'agriculture arable et l'agriculture en verger, peuvent avoir des exigences distinctes pour leurs opérations respectives.

L'agriculture en verger consiste à cultiver des fruits et des noix, tels que les pommes, les poires, les cerises et les amandes, dans des arbres plantés de manière rapprochée. La disposition dense des arbres présente des défis pour la navigation et l'exécution de tâches à l'aide de machines agricoles traditionnelles. Les systèmes autonomes offrent une solution pour naviguer entre les arbres et effectuer des tâches telles que la taille, la pulvérisation et la cueillette des fruits. Ces systèmes peuvent optimiser les calendriers d'irrigation et de fertilisation et prévoir les épidémies de maladies, réduisant ainsi les coûts de main-d'œuvre et améliorant l'efficacité. La localisation prédéfinie des arbres en agriculture en verger facilite l'utilisation de systèmes autonomes, car les trajectoires sont déjà définies. Cependant, cette étude se concentre principalement sur les opérations effectuées en agriculture arable, où la génération d'un chemin de couverture complète est nécessaire, et la direction des trajectoires parallèles n'est pas prédéfinie par de telles contraintes.

L'agriculture arable implique la culture à grande échelle de cultures dans les champs. Elle comprend des tâches répétitives qui peuvent bénéficier considérablement des systèmes autonomes. Les tâches comprennent le labour et le travail du sol, l'ensemencement, la plantation, la fertilisation, l'irrigation, la pulvérisation, la récolte et la tonte. Les systèmes autonomes peuvent effectuer ces tâches efficacement et avec précision, minimisant le risque d'erreur humaine et le besoin de main-d'œuvre. Par exemple, les semoirs autonomes peuvent garantir une croissance optimale des plantes en plantant des graines à des intervalles et des profondeurs précis, tandis que les systèmes de pulvérisation autonomes peuvent appliquer avec précision les pesticides et les herbicides, minimisant les risques environnementaux.

En agriculture, la PTCC joue un rôle crucial pour naviguer efficacement un robot sur l'ensemble du champ tout en minimisant les chevauchements. Elle implique la création d'un chemin réalisable et optimal qui prend en compte les contraintes du robot et les caractéristiques de l'outil, tout en optimisant un ensemble d'objectifs.

Cependant, développer une approche PTCC générique pour l'agriculture est difficile en raison des diverses opérations, outils et contraintes impliqués. Par exemple, l'ensemencement nécessite une mise en œuvre précise à une profondeur spécifique, nécessitant un mouvement droit tout en ajustant la hauteur de l'outil. En revanche, la pulvérisation ne nécessite pas de contact avec le sol, mais le robot doit éviter d'endommager les cultures germées.

De plus, la PTCC devient complexe lorsqu'on prend en compte l'inclinaison et les pentes du champ, qui peuvent affecter les performances du robot et la qualité des résultats. Construire un modèle de champ en 3D ajoute une autre couche de complexité mais est crucial pour optimiser l'efficacité et l'efficience du chemin, en particulier en termes de consommation d'énergie et d'érosion du sol. Par conséquent, les approches PTCC doivent prendre en compte les exigences spécifiques de chaque opération et outil, ainsi que les caractéristiques 2D et 3D du champ, pour garantir des performances optimales.

A.1.1 Motivations

La principale motivation de cette thèse est de développer une approche PTCC générique et efficace pour les robots agricoles autonomes qui peut répondre aux complexités de diverses opérations agricoles. Plus précisément, l'objectif est de concevoir des méthodes PTCC qui peuvent prendre en compte les exigences spécifiques de chaque opération et outil, tout en intégrant les caractéristiques 3D du champ et en optimisant plusieurs objectifs tels que la zone de couverture, la réduction des chevauchements, la distance non travaillée, la consommation d'énergie et l'érosion du sol. L'approche proposée devrait être applicable à une large gamme d'applications agricoles et de formes de champs, garantissant des performances optimales. En fin de compte, l'objectif est de contribuer à l'avancement des approches PTCC pour les robots agricoles autonomes, fournissant une base pour des recherches et un développement ultérieurs dans ce domaine.

A.1.2 Contributions

Cette thèse introduit plusieurs contributions significatives au domaine de la robotique agricole:

Une revue systématique des algorithmes existants pour résoudre les problèmes de PTCC dans les robots agricoles à roues. La revue analyse 48 articles et discute des facteurs clés qui impactent l'efficacité des approches PTCC, y compris la modélisation du terrain et des contraintes, et les techniques de planification de trajectoire.

Des discussions détaillées et un ensemble de données sur la génération de modèles de champs 2D et 3D, en utilisant des données provenant de 30 champs en France. L'ensemble de données, disponible sur Zenodo [115], offre des perspectives précieuses pour les chercheurs en robotique agricole, aidant au développement de solutions précises et à l'évaluation des futures approches de planification de trajectoire.

Développement d'une approche PTCC efficace pour créer des chemins optimaux pour les robots agricoles autonomes. L'approche vise à obtenir une couverture de champ de haute précision tout en minimisant les chevauchements, la longueur du chemin non travaillé et le temps de déplacement global. Elle utilise un algorithme de recherche intelligent basé sur un arbre et prend en compte des facteurs importants tels que la géométrie du robot et de l'outil.

Investigation sur une approche basée sur l'apprentissage profond pour la décomposition de champ dans la PTCC agricole. Cette recherche met en évidence les complexités de l'ensemble de données et souligne la nécessité de reconsidérer certaines hypothèses sur les préférences des agriculteurs, les opérations et les exigences des machines. Bien que les résultats puissent ne pas répondre aux attentes initiales, cette étude souligne l'importance d'une approche PTCC robuste capable de gérer efficacement plusieurs lignes de division et diverses contraintes.

Développement d'une approche avancée de planification de trajectoire hybride 3D à objectifs multiples pour une couverture complète. Cette approche combine les forces de la méthode précédemment proposée et d'un algorithme open-source, en tenant compte des inclinaisons de la trajectoire de travail qui impactent l'érosion du sol et la consommation d'énergie, parmi d'autres objectifs. L'évaluation par rapport à l'algorithme proposé à l'origine démontre l'efficacité et l'efficience de cette approche hybride.

A.2 Modélisation du Terrain

La modélisation du terrain permet de créer des représentations numériques 2D ou 3D d'une surface terrestre physique, essentielle pour la planification de trajectoires, l'analyse de l'érosion des sols et l'estimation de la consommation d'énergie. La qualité du modèle influence considérablement la précision de ces applications. La modélisation du terrain en 2D offre une projection plate du terrain, tandis que la modélisation en 3D tient compte de la topographie et des variations d'altitude. Bien que les modèles 3D offrent une représentation du terrain plus précise, leur utilisation dans la modélisation du terrain agricole est limitée en raison de la complexité computationnelle associée.

L'acquisition de données précises est essentielle pour créer des modèles de champ fiables. Dans notre cas d'utilisation, les données nécessaires pour construire des modèles de champ 3D et 2D comprennent des points dans le sens antihoraire représentant les frontières du champ, un ensemble de segments de ligne pour les segments d'accès, des données d'altitude et des points optionnels dans le sens horaire pour représenter les obstacles statiques. Les segments d'accès sont des lignes 2D sur les frontières du champ indiquant d'où le robot peut entrer et sortir des points du champ.

Les frontières du champ, les obstacles et les segments d'accès sont acquis à l'aide de l'outil d'annotation Geoportail [45]. Il offre une interface graphique pour dessiner des points, des lignes et des polygones sur des images satellites, et les résultats peuvent être exportés sous forme de fichier Keyhole Markup Language (KML), qui comprend toutes les coordonnées.

Les données d'altitude sont acquises à l'aide des services de calcul d'altitude de l'IGN [58]. Une grille d'altitude est générée dans la boîte englobante du champ, puis l'API REST est utilisée pour déterminer l'altitude de chaque point. Cependant, ce processus peut être long et peut générer du bruit dans les données pour les grilles haute résolution.

Une solution pratique à ce problème est de créer une grille d'altitude de résolution inférieure. Cette grille d'altitude de faible résolution peut ensuite être utilisée pour construire une surface 3D haute résolution du champ à travers des méthodes d'interpolation d'altitude.

A.2.1 Construction de Surface 2D

La création d'une surface 2D d'un champ est un processus en plusieurs étapes :

- Un polygone 2D est créé qui combine le champ et les polygones d'obstacle
- Une grille 2D est générée dans la boîte englobante du champ
- Les points trouvés en dehors du polygone 2D sont supprimés
- De nouveaux points sont interpolés le long des frontières du champ et des obstacles
- Tous les points restants sont combinés et triangulés

Le résultat est une surface triangulée de manière homogène, sauf près des frontières du champ et des obstacles où elle peut être non homogène. Dans le cas d'un champ avec un obstacle, le polygone du champ est généré en premier, en excluant l'obstacle.

Ensuite, une grille 2D est générée dans la boîte englobante du champ. Seuls les points à l'intérieur du polygone 2D sont conservés. Ensuite, des points sont interpolés sur les frontières du champ et des obstacles. Enfin, tous les points restants sont triangulés pour créer une surface de champ 2D.

A.2.2 Construction de Surface 3D

Après avoir construit une surface 2D haute résolution du champ, la grille d'altitude et une approche d'interpolation peuvent être utilisées pour estimer l'altitude de chaque point de la surface 2D. Le résultat de ce processus est une surface 3D haute résolution. L'une des méthodes couramment utilisées à cette fin est le *Inverse Distance Weighting (IDW)*.

Dans cette approche, l'altitude d'un point est calculée avec une moyenne pondérée des points environnants dont l'altitude est déjà connue. Les poids sont calculés comme l'inverse de la distance à chaque point voisin. Par conséquent, l'altitude du point P est calculée en fonction de ses N points voisins les plus proches $\{P_i | P_i \in \mathbb{R}^2, i \in \mathbb{N}, 1 \leq i \leq N\}$ comme suit :

$$f(P) = \begin{cases} \frac{\sum_{i=1}^{N} w_i(P_i) f(P_i)}{\sum_{i=1}^{N} w_i(P_i)}, & \text{si } d(P, P_i) \neq 0 \text{ pour tout } i \\ f(P_i), & \text{si } d(P, P_i) = 0 \text{ pour certains } i \end{cases} \tag{A.1}$$

où

$$w(P_i) = \frac{1}{d(P, P_i)^p} \tag{A.2}$$

où d désigne la distance euclidienne et p est un nombre réel positif, appelé paramètre de puissance.

En l'absence d'une vérité terrain, il est difficile de déterminer la précision de cette construction de surface 3D. Il peut être nécessaire d'utiliser un drone équipé de la technologie LiDAR aéroportée pour construire un ensemble de données de vérité terrain précis pour étudier l'efficacité des différentes méthodes de reconstruction 3D pour un champ agricole.

A.2.3 Un ensemble de données de champs réels

Il existe un écart reconnu dans la littérature existante pour un ensemble de données englobant des données complètes pour évaluer les stratégies de planification de trajectoire sur les surfaces de champ agricole 2D et 3D. Pour combler ce vide, nous avons créé un ensemble de données publiquement disponible sur Zenodo composé de trente champs agricoles diversifiés en France. Cet ensemble de données, allant de $1,83$ à $13,21$ hectares et présentant des formes variées, offre un large éventail de scénarios réels pour valider les approches de planification de trajectoire.

Pour chaque champ, cet ensemble de données rassemble les informations suivantes dans des fichiers séparés :

- une image aérienne (PNG)
- un polygone 2D (XML)

- une surface triangulée 2D (PLY), avec un espacement de 0,25 mètres
- une grille d'altitude (PLY), avec un espacement de 5 mètres
- une surface triangulée 3D (PLY), avec des paramètres IDW $N = 20$ et $p = 2$
- un ensemble de segments de ligne 2D représentant les segments d'accès (XML)
- des ensembles de lignes de division pour 10 champs pour les décomposer

où le *Polygonal File Format (PLY)* est un format de fichier pour stocker des données graphiques 3D, y compris des modèles et des scans. Notamment, les surfaces de champ 2D et 3D peuvent être représentées comme des modèles 3D dans ce format, la seule différence étant que tous les points d'une surface 2D ont une valeur z de zéro.

A.3 Méthodes

A.3.1 Recherche basée sur un arbre intelligent

L'une de nos approches est un PTCC basé sur un arbre pour trouver un ou plusieurs chemins de couverture. Elle comporte trois phases clés : le prétraitement, la recherche intelligente basée sur un arbre (ou algorithme d'exploration) et la vérification de similarité pour sélectionner les solutions optimales.

Dans la phase de prétraitement, nous préparons le champ en utilisant des entrées comme le polygone du champ, les lignes de division, les segments d'accès, la largeur de travail et le rayon de braquage minimum du robot. Cette phase produit des sorties comme un ensemble d'entrées, des bordures et des espaces de rotation nécessaires pour la navigation entre les bordures.

L'algorithme d'exploration, la deuxième phase, vise à découvrir toutes les solutions possibles et à les stocker dans un espace de solution. Chaque solution est un chemin de couverture - une séquence de trajectoires qui commence à une entrée, couvre de manière optimale le champ et les bordures, et se termine sur l'un des segments d'accès.

Dans des scénarios avec plusieurs entrées et/ou lignes de division, ces phases sont répétées plusieurs fois. Pour les champs complexes, s'il y a d lignes de division différentes et e entrées, cela nécessite d exécutions de prétraitement et $d * e$ explorations. Nous considérons également des cas sans lignes de division, car la solution optimale pour certains champs peut ne pas nécessiter de division du champ.

La phase finale consiste à calculer le coût de chaque solution qui est un coût moyen pondéré de couverture, de chevauchement, de distance parcourue non travaillée et de coûts de temps d'opération, à regrouper des solutions similaires en familles à l'aide d'une fonction de similarité, et à ne conserver que la solution la moins coûteuse de chaque famille.

L'arbre est construit à l'aide de nœuds qui représentent des séquences potentielles de trajectoires, qui doivent satisfaire à certaines contraintes strictes.

Nœuds : Chaque nœud de l'arbre représente une étape potentielle suivante dans la trajectoire. Il contient des informations sur le type de trajectoire, la destination et la direction. L'arbre commence par un nœud racine représentant l'entrée et se termine par un nœud feuille représentant un point de sortie.

Contraintes strictes : Chaque trajectoire doit respecter un ensemble de contraintes strictes. Ces contraintes garantissent que le robot reste à l'intérieur du champ, évite de causer des dommages, limite les chevauchements et respecte des longueurs et modalités de trajectoire spécifiques. Certaines de ces contraintes comprennent :

- **Contrainte intérieure :** Le robot doit rester à l'intérieur du champ.
- **Contrainte de dommage :** Le robot ne doit pas endommager les trajectoires existantes.
- **Contrainte de chevauchement limité :** Les chevauchements au centre du champ ne sont pas autorisés.
- **Contrainte de chevauchement global :** Il y a un seuil pour la superficie totale de chevauchement.
- **Contrainte de boucle locale :** Empêche les boucles locales indésirables.
- **Contrainte de commutation :** Permet de passer d'un sous-champ à l'autre dans certaines conditions.
- **Contrainte MWD :** Garantit une longueur de trajectoire minimale.

Construction et exploration de l'arbre : L'arbre est initialisé avec un nœud d'entrée. De là, les nœuds sont générés et explorés en utilisant une approche en profondeur. L'exploration prend en compte divers scénarios comme les cycles de traversées et les demi-tours, les commutations de bordure et la sortie du champ.

Vérification de similarité et sélection des solutions optimales

Après avoir construit l'arbre et généré des solutions potentielles, l'étape suivante consiste à évaluer et à sélectionner les solutions les plus optimales.

Fonction de coût : Une fonction de coût est introduite pour évaluer chaque solution. Cette fonction prend en compte quatre métriques : le taux de couverture, le taux de chevauchement, la distance parcourue non travaillée et le temps d'opération. Chaque métrique est normalisée pour garantir une évaluation cohérente.

Classification des solutions : Pour faciliter la sélection d'une solution idéale par les utilisateurs, les solutions générées sont regroupées en familles en fonction de leur similarité. La similarité est déterminée par la direction générale des solutions. De chaque famille, la solution ayant le coût le plus bas est proposée à l'utilisateur, garantissant une variété d'options sans redondance écrasante.

Développement du programme et ensemble de données

Un programme sophistiqué a été développé en C++ pour déterminer les trajectoires optimales pour les robots opérant dans les champs agricoles. Ce programme, équipé d'une interface graphique, permet aux utilisateurs de définir des paramètres d'entrée et de visualiser les trajectoires résultantes. En exploitant les capacités de traitement parallèle de la bibliothèque OpenMP, le programme atteint des temps de calcul plus rapides. L'efficacité du programme a été testée sur un ensemble de données présenté dans un chapitre précédent, qui comprend à la fois des champs simples et complexes. Ces champs ont été visualisés à l'aide de figures, avec différentes couleurs et marqueurs indiquant diverses caractéristiques comme les entrées, les lignes de division et les polygones de champ.

Évaluation et Résultats

L'approche a été rigoureusement appliquée à chaque champ de l'ensemble de données, en tenant compte des différentes lignes de division et entrées fournies. Les trajectoires générées ont ensuite été comparées à de véritables images satellite pour évaluer leur précision et leur praticité. Les résultats ont été prometteurs : pour les champs simples, le taux de couverture était impressionnant à $98,69\%$, et pour les champs complexes plus difficiles, il était de $98,23\%$. Fait intéressant, dans de nombreux cas, la solution la plus optimale (en termes de couverture et de chevauchement) ressemblait également étroitement aux trajectoires réelles visibles sur les images satellite.

Familles de solutions et accessibilité du champ : L'un des aspects innovants de l'approche était sa capacité à regrouper les solutions en "familles". Cela signifie que pour un champ donné, plusieurs chemins optimaux étaient présentés à l'utilisateur, offrant une variété de choix qui pourraient convenir à différents besoins ou préférences. L'importance de l'accessibilité du champ a également été soulignée. Si l'accessibilité d'un champ est décrite de manière inexacte, la solution générée pourrait être impraticable ou même nuisible, car le robot pourrait involontairement s'aventurer dans un champ voisin.

Comparaison avec d'autres méthodes : Une comparaison directe a été faite avec un autre algorithme, Fields2Cover (F2C). Bien que F2C ait été plus rapide sur le plan computationnel, la nouvelle approche a affiché un taux de couverture plus élevé. Cependant, cette comparaison a mis en évidence les défis inhérents à la comparaison directe de différentes méthodologies, en particulier lorsqu'elles prennent en compte différentes contraintes et objectifs. Par exemple, les trajectoires de F2C ignoraient les contraintes pratiques comme la longueur minimale de trajectoire requise pour activer et désactiver les outils agricoles.

A.3.2 Modèle de couverture avec saut de rangée

Notre prochaine approche est basée sur le concept d'un modèle de couverture "Row-Skip", où certaines rangées d'un champ sont intentionnellement sautées lors de la couverture initiale et couvertes plus tard. Ce modèle peut réduire la taille des bordures, qui sont souvent moins productives en raison de la compaction du sol. Bien que les opérateurs humains puissent trouver le saut de rangée difficile en raison du manque de repères visuels, les systèmes automatisés ou les robots avec des systèmes de guidage peuvent efficacement mettre en œuvre cette stratégie. Plusieurs études ont exploré divers modèles de couverture, mais beaucoup simplifient trop certains aspects ou ne tiennent pas compte de défis spécifiques comme la couverture des bordures. La section introduit une nouvelle approche de modèle de saut de rangée qui vise à maximiser la couverture du champ tout en tenant compte de diverses contraintes.

Le Row-Skip PTCC se compose également de trois parties principales : le prétraitement, l'exploration et la sélection des solutions optimales. Le prétraitement est similaire à l'approche O-PTCC, en se concentrant sur la génération des bordures et la détermination des points d'entrée. L'étape d'exploration, cependant, est modifiée pour générer des solutions avec un modèle de saut de rangée. Le processus de sélection est également ajusté pour tenir compte de nouvelles métriques, telles que le nombre de virages avec des mouvements inverses.

L'algorithme d'exploration construit un arbre de séquences de trajectoires potentielles, en tenant compte de scénarios tels que les cycles de traversée, les commutations de bordure et les sorties de champ. Une grande partie de l'exploration consiste à déterminer quand sauter des rangées et quand couvrir les rangées précédemment sautées, en particulier lorsque le robot approche des bords du champ. L'algorithme utilise des rayons pour déterminer le chemin du robot et les intersections potentielles avec les bordures.

Après avoir exploré les solutions possibles, l'algorithme calcule des métriques telles que le taux de couverture, le taux de chevauchement, la distance non travaillée, le temps d'opération et le nombre de virages avec des mouvements inverses pour chaque solution. Ces métriques sont ensuite normalisées et pondérées pour déterminer le coût final de chaque solution. La solution ayant le coût le plus bas est sélectionnée comme solution optimale.

Résultats

Le Row-Skip PTCC a été comparé à l'approche précédente (O-PTCC) en utilisant six champs d'un ensemble de données. Les résultats ont montré que le Row-Skip PTCC surperformait généralement l'O-PTCC, en particulier lors de la prise en compte de métriques telles que le taux de couverture et le taux de chevauchement. Cependant, l'O-PTCC avait parfois une distance non travaillée plus faible.

Les résultats indiquent que le Row-Skip PTCC pourrait être une stratégie viable pour maximiser la couverture du champ tout en minimisant le chevauchement. Cependant, des recherches supplémentaires sont nécessaires pour déterminer les meilleures stratégies de saut de rangée pour différents types de champs et de cultures.

A.4 Conclusion

La planification de trajectoire pour les robots agricoles est un défi complexe qui nécessite une prise en compte minutieuse des contraintes du champ, des capacités du robot et des besoins de l'opération agricole. Cette recherche a introduit deux approches pour résoudre le problème de la planification de trajectoire : l'O-PTCC et le Row-Skip PTCC. Les deux approches ont montré des résultats prometteurs lorsqu'elles sont testées sur des ensembles de données réels, bien que chacune ait ses propres avantages et inconvénients.

À l'avenir, il serait bénéfique d'explorer d'autres modèles de couverture, d'intégrer des données en temps réel pour une planification adaptative et de tester ces méthodes sur un plus grand ensemble de données. De plus, l'exploration de l'intégration de la technologie de détection pour éviter les obstacles dynamiques pourrait également améliorer la robustesse et la fiabilité des solutions de planification de trajectoire pour les robots agricoles.