Introduction of a Skip-Row pattern in complete coverage path planning for agricultural fields

1st Danial Pour Arab
ICube, CNRS (UMR 7357)
University of Strasbourg
Strasbourg, France
pourarab@etu.unistra.fr

2nd Matthias Spisser
InnoLab
T&S - Technology and Strategy
Strasbourg, France
m.spisser@technologyandstrategy.com

3rd Caroline Essert
ICube, CNRS (UMR 7357)
University of Strasbourg
Strasbourg, France
essert@unistra.fr

***Abstract*—Over the past two decades, an evolutionary effort has been established in the agricultural sector to develop efficient autonomous systems that can carry out common in-field operations including harvesting, mowing, and spraying. Increasing production while decreasing costs and environmental damages is one of the main objectives for these autonomous systems. Due to the nature of these tasks, complete coverage path planning techniques are crucial to determining the best path that covers the entire field while accounting for terrain characteristics, operational needs, and robot properties.**

In this study, we propose a novel complete coverage path planning approach to define the ideal path for a wheeled robot across an agricultural field. To identify all feasible solutions satisfying a set of predefined constraints, a method based on tree exploration is first proposed that examines skip-row patterns. Second, the most optimal solution is selected by a selection method. Maximizing the covered area while minimizing overlaps, non-working path length, number of turns containing reverse moves, and overall travel time are the objectives of the selection method.

We showed on 6 real-world fields geometries that the row skip approach offered benefits in terms of reduction of the required headland size, and often helped decreasing the number of necessary reverse moves and the overlaps, while increasing the covered area.

***Index Terms*—Complete Coverage Path Planning, Precision Agriculture, Autonomous Agriculture, Vehicle Routing Problem, Wheeled robots, Path Planning, Route Planning**

I. Introduction

Automated systems and robots are increasingly used in every sector of industry since they boost productivity while cutting expenses. In comparison to humans, they are able to do a wider range of specialized and tedious activities with a remarkable precision and accuracy. Agriculture is no exception to this.

More food and agricultural products are required due to a growing population [1], yet agriculture is also known as a remarkable source of the air and environmental pollution [2]. Autonomous systems capable of optimizing the cost and efficiency of in-field operations is indispensable and demands an urgent attention.

This work was funded by research grant CIFRE 2019/1084 from Technology & Strategy Group (T&S) and the French National Association for Research and Technology (ANRT).

One of the main challenges to perform an autonomous operation such as harvesting, tillage, seeding, and pulverization is to find a path that optimally covers the entire field. This challenge is commonly known as *Complete Coverage Path Planning* (CCPP). To generate an optimal and feasible path, it is essential to consider the geometry and characteristics of the field, the robot, and the machinery i.e. implement (*Imp* for short) connected to the robot.

Due to the complexity of this problem, in the majority of cases, the CCPP is referred to in the literature as two distinct and sequential tasks: *Coverage Path Planning* (CPP) and *Agricultural Routing Problem* (AVRP). CPP consists in generating a set of parallel trajectories while AVRP consists in finding an optimal sequence of these trajectories for a single or multiple robots.

Generating parallel trajectories based on a reference direction or the longest edge of the field is a prevalent CPP method chosen by [3]–[9]. Furthermore, improving on this method, generating parallel trajectories to a curved reference line was also proposed [10], [11]. Another improvement was to generate straight trajectories parallel to the boundary that minimized the number of trajectories [12]–[14].

After determining the parallel lines, the generated paths often simply links them sequentially by half-turns. To determine an optimal sequence, some authors applied a genetic algorithm [15]. Zhou et al. [9] applied the ant colony optimization. Jeon et al. [5] examined sequential and gathering patterns. In the gathering pattern, the distance between two consecutive trajectories is around half of the width of the field.

All of these methods, however, perform CCP and AVRP separately as two distinct problems. In a previous work, we proposed a one-step CCPP algorithm [16] to cover convex and concave fields entirely, including the headlands, for field operations in which the implement is in contact with the ground when it is engaged. We demonstrated that a one-step CCPP has the potential to find some interesting solutions that are not possible to find with a two-step CCPP. This approach provides optimal solutions with a sequential pattern of adjacent tracks and headland tracks.

However, in some cases, it may be interesting to consider skipping every other track: with some vehicles that have a large turning radius, this method may reduce the size required for

headlands or avoid reversing. A human-operated vehicle will prefer to avoid reversing and will find it difficult to skip rows. But for an automated system or robot, this is a viable option that has the potential to reduce the headlands area, which are generally under-performing parts of the land due to the soil compaction.

Based on our previous works [16], in this paper we propose an extension of our one-step CCPP approach. In Section II, we first recall the main principles of our method. Then, we explain how the algorithm takes into account row skipping in the sequence of trajectories. In Section V, we detail our experiment and results, before discussing and concluding them.

II. Tree-based intelligent search

Our approach, explained in more detail in [16], is performed in three primary steps: 1) preprocessing 2) exploration 3) Selection of optimal solutions. The preprocessing step determines headlands, trajectories within headlands, as well as turning spaces needed for traveling from one headland to another. Considering the accessible edges of the field also known as *access segments*, entry points are also determined. The exploration algorithm finds every potential solution, while respecting some predefined constraints, and stores them in a solution space. A solution is a path i.e. a sequence of trajectories that begins at an entrance, covers the field entirely, and ends on one of the access segments. Finally, in the last step, the cost of each solution is computed and the one that has the lowest cost is selected.

In the extended algorithm, the first (preprocessing) step remains identical to our previous method. The exploration step is modified to generate solutions with a skip-row pattern instead of a sequential pattern. In a skip-row pattern two consecutive working trajectories in the main part of the field are not adjacent. As an exception, they may be adjacent when they are located close to a border of the field. This is useful to cover previously skipped tracks. In order to take into account and minimize the number of turns that include reverse moves as a soft constraint, the last step (selection of optimal solutions) is also slightly modified.

Headlands, that are used for performing half-turns, are computed adjacent to the borders of the field. Performing half-turns causes unworked areas in headlands as well as gaps before and after each half-turn due to the distance needed for lowering/raising the implement properly. Therefore, p trajectories (inner trajectories) are considered inside a headland for covering unworked areas caused by half-turns. One trajectory outside and adjacent to each headland is also considered to cover the unworked areas caused by gaps. These trajectories are known as gap covering trajectories.

During the preprocessing step, turning spaces are also computed around each corner of the field borders to ensure feasible turns from one headland to another. For further detail on the preprocessing step, we refer the readers to our previous approach [16].

III. Exploration algorithm

The inputs of the exploration algorithm are the result of the preprocessing, the entry points, a set of hard constraints, γ_{on}, γ_{off}, ℓ_t, ℓ_o and a *coverage threshold*, where γ_{on} and γ_{off} represent, respectively, the minimum turning radius of the robot while its implement is on (lowered into the soil and activated) and off (raised from the ground). ℓ_t represents the distance needed for lowering/raising the implement properly. ℓ_o represents the offset between the robot and its implement.

Starting from an entrance, the exploration algorithm progressively constructs and explores a tree where each sequence of connected nodes represent a possible sequence of trajectories. A sequence of trajectories that satisfies the constraints, has a coverage rate greater or equal to the coverage threshold Δ_{cov} and ends on an access segment, is stored as a solution in the solution space.

Let us recall the hard constraints, defined as follows:

- the robot and its implement must remain inside the field
- crossing a previously worked area with a new trajectory, during which the implement is off, is forbidden
- overlaps in the main part of the field, i.e. outside the headlands and the gap covering trajectories is forbidden
- the total overlap area caused by all trajectories of a sequence must not exceeds a *global overlap threshold* Δ_{global} that is a percentage of the field area
- to avoid unwanted local loops, the overlap of a trajectory of a sequence with its ancestors in the same sequence must not exceed a percentage of the trajectory's worked area (Δ_{local}). to let the robot find an access segment and exit the field, once the coverage rate exceeds Δ_{cov}, this constraint is neglected
- the robot must travel at least Δ_{min_dist} while its implement is on to authorize another trajectory for which the implement is not on. The main reason is to avoid lowering the implement to the ground and raising it for short distances

After initializing a tree with an entry point as its root, the construction and the exploration of the tree is performed for each unvisited leaf node as a depth first exploration.

As illustrated in Fig. 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 and/or with a turning space defines the next nodes that will be generated and might be added to the tree after being validated by the hard constraints.

In case ray r_p intersects with a turning space, trajectories for traveling from one headland to another are generated as detailed in our previous approach [16].

In case ray r_p intersects with an inner border of a headland, first a sequence of three trajectories are generated to reach the corresponding inner border. As illustrated in Fig. 1, this sequence contains a gap trajectory of length ℓ_t from N_p to N_{c1} for lowering the implement, a working trajectory from N_{c1} to N_{c2} and an inverse gap trajectory of length $\ell_t + \ell_o$ from N_{c2} to N_{c3} for raising the implement. Afterwards, four

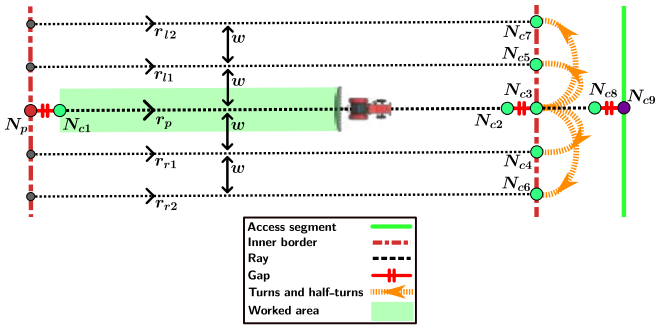
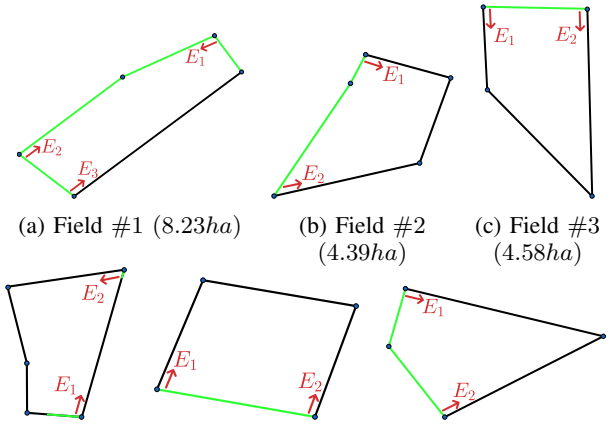


Fig. 1: Trajectory and tree representation of back-and-forth moves. The purple point represents an exit node. For readability purposes, only inner border of two headlands are represented



(a) Field #1 ($8.23ha$) (b) Field #2 ($4.39ha$) (c) Field #3 ($4.58ha$)

(d) Field #4 ($7.85ha$) (e) Field #5 ($7.68ha$) (f) Field #6 ($4.34ha$)

Fig. 2: Green segments are accessible edges of the field. Entrances are represented by red arrows

side rays $r_(r1)$, $r_(r2)$, $r_(l1)$, and $r_(l2)$ are defined. They 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 they are added to the tree after being validated.

When the intersection of either N_{c6} or N_{c7} with an inner border exist, it means that taking an adjacent track is needed to fill all previous skips.

The node generation scheme is repeated for all unvisited leaves of the tree. If an access segment is nearby and the coverage rate is already satisfied, a possible sequence of trajectories to exiting the field is also generated. The sequence of nodes N_p, N_{c1}, N_{c2}, N_{c3} and N_{c9} in Fig. 1 illustrate an example of this case.

IV. Selecting the most optimal solution

Assuming that e entrances are proposed in the preprocessing step, the exploration algorithm is performed for each of the entrances. The results of all explorations are stored in a single solution space. Afterwards, coverage rate S_{cov}, overlap rate S_{ovl}, non-working distance S_{nwd}, operation time S_{otm} and number of turns having reverse moves S_{rvs} are computed for each solution and normalized by (1).

$$S = \frac{S - S_{min}}{S_{max} - S_{min}} \tag{1}$$

where S_{min} and S_{max} represent the minimum and maximum value of the corresponding metric over all solutions of the solution space. For a solution i.e. a sequence of trajectories, the coverage and overlap rates are respectively computed as the sum of worked and overlap area caused by each trajectory. The operation time for each solution is computed by (2).

$$S_{otm} = \frac{L_{on}}{V_{on}} + \frac{L_{off}}{V_{off}} + \frac{L_{gap}}{V_{gap}} \tag{2}$$

where L_{on}, L_{on} and L_{gap} are respectively the length of all trajectories during which the implement is on, of and in transition (from on to off or vice versa). 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 distance for the corresponding solution is then computed as $L_{off} + L_{gap}$.

Considering $\mathbf{C} = (1 - S_{cov}\ S_{ovl}\ S_{nwd}\ S_{otm}\ S_{rvs})$ 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 given as input for the corresponding soft constraint, the final cost of each solution is computed by (3).

$$f = \frac{\mathbf{C}\mathbf{W}^\intercal}{W_{cov} + W_{ovl} + W_{otm} + W_{nwd}} \tag{3}$$

Finally the solution that has the lowest cost is represented as the most optimal solution found by our approach.

V. Results

To compare the presented approach (Skip-Row pattern CCPP, SR-CCPP for short) against our previous CCPP approach (simple CCPP), six real-world fields were selected. The area of these fields varies from 4.34 to 8.23 hectares (see Fig. 2). Both approaches were implemented and run on an Intel Xeon(R) W-2135 CPU @ 3.70GHz × 12 with 32GB RAM.

To compare the final cost of the most optimal result found by each approach, the solution spaces acquired by their exploration algorithms were combined and the selection method was applied on the combined solution space.

To compare SR-CCPP and simple CCPP, we considered two different value for γ_{off}; $2m$ and $3m$, while other parameters remain the same. The number of trajectories within a headland was set to two ($p = 2$). Other parameters are given in Table

Parameter	Value	Parameter	Value
w	$3m$	Δ_{global}	5%
γ_{on}	$15m$	Δ_{local}	95%
V_{on}	$3.5m/s$	Δ_{min_dist}	$4m$
V_{gap}	$2.5m/s$	W_{cov}	0.65
V_{off}	$1.5m/s$	W_{ovl}	0.15
ℓ_t	$2m$	W_{nwd}	0.05
ℓ_o	$2m$	W_{otm}	0.05
Δ_{cov}	96%	W_{rvs}	0.10

TABLE I: Inputs and parameters

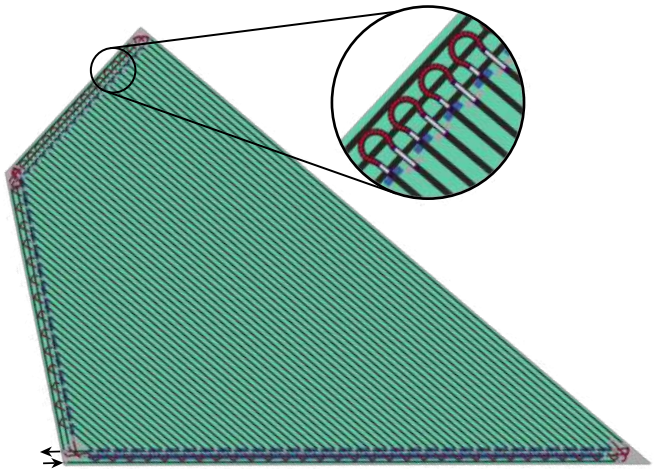
Field	Approach	time (s)	Coverage	Overlap	Rvs	L_{nw} (m)	f
#1	SR-CCPP	160.59	98.93%	1.23%	2	1184, 17	**0.110**
	CCPP	9.73	98.80%	2.27%	4	3005.92	0.281
#2	SR-CCPP	131.47	98.80%	2.05%	0	1485.36	**0.118**
	CCPP	10.01	98.21%	3.58%	2	1433.84	0.336
#3	SR-CCPP	282.28	98.96%	1.39%	0	1565.47	**0.118**
	CCPP	18.65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

(a) $\gamma_{off} = 2m$

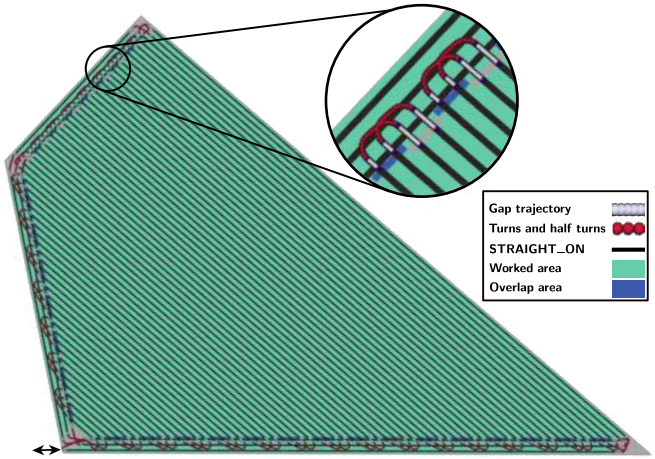
Field	Approach	time (s)	Coverage	Overlap	Rvs	L_{nw} (m)	f
#1	SR-CCPP	58.08	98.61%	1.68%	10	2916.47	**0.193**
	CCPP	14.14	98.42%	0.23%	57	882.89	0.199
#2	SR-CCPP	35.92	98.65%	2.01%	48	1415.98	**0.201**
	CCPP	8.70	98.75%	3.31%	71	1080.41	0.224
#3	SR-CCPP	38.48	98.73%	3.39%	34	1206.01	**0.193**
	CCPP	9.01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

(b) $\gamma_{off} = 3m$

Fig. 3: Numerical results. The exploration time is referred to as *time* for short. Rvs represents the number of turns that include reverse moves and L_{nw} represents the non-working traveled distance



(a) Simple CCPP



(b) SR-CCPP

Fig. 4: Most optimal results obtained for Field #6 while $\gamma_{off} = 2m$. The black arrows indicate where the robot enters and exits

I. Table 3 summarizes numerical results of these comparisons and Fig. 4 illustrate the most optimal results for Field #6.

VI. Discussion

Let us recall that for performing turns and half-turns, our approach first tries to employ only forward moves by applying the method proposed by [17]. If it is not possible, turns that include reverse moves are also considered. This second type of turns are generated by the method proposed by [18].

Table 3a summarizes the results obtained by the presented approach (SR-CCPP) and our previous approach (simple CCPP) while $\gamma_{off} = 2m$. According to these results, in general SR-CCPP performed better than simple CCPP for five fields out of six. For Fields #1, #2 and #3, SR-CCPP achieved a better coverage rate. For Fields #1, #2, #3, #5 and #6, SR-CCPP did considerably less overlaps. However, in terms of non-working traveled distance simple CCPP was better for four fields.

Increasing γ_{off} to three meters, SR-CCPP performed better for all cases. As summarized in Table 3b, skipping feature of SR-CCPP made it able to use less turns including reverse moves for all fields. It also decreased the overlap rate by almost 1% or even more, for the last four fields. However, skipping tracks also caused more non-working traveled distance.

As can be noticed in Fig. 4, when skipping rows, turns use less space in the headlands. A tight turn will tend to have a bulbous shape that occupies more space, whereas a wider turn may be flatter. This highlights the ability of the row skip pattern to decrease the size required for headlands. Given that a headland is generally a less productive area of the field, this can potentially increase the productivity of the field.

In general, depending on the features of the robot, the

field shape and accessibility, the headlands width and the goals of optimization, one method might outperform another. For instance, considering the minimization of non-working traveled distance as the primary goal of the optimization by modifying the weight of constraints, simple CCPP may perform better. As a result, allowing the decision-making algorithm to try each approach, or to combine them into a more sophisticated CCPP, could improve the effectiveness of the final method for finding a good solution for a variety of machinery, operations and field shapes.

VII. Conclusion

In this paper, a one-step CCPP method that can provide a coverage path with a skip-row pattern is proposed. The presented approach maximizes 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 consider the geometry of both the robot and its implement. We compared the proposed approach with our previous approach and we showed that considering both sequential and skip-row pattern increases the possibility of finding the optimum solution. It also increases the ability of a CCPP approach to find a proper solution for a variety of parameters and field shapes.

References

[1] A. Marques, I. S. Martins, T. Kastner, *et al.*, "Increasing impacts of land use on biodiversity and carbon sequestration driven by population and economic growth," *Nature ecology & evolution*, vol. 3, no. 4, pp. 628–637, 2019.

[2] I. M. Meftaul, K. Venkateswarlu, R. Dharmarajan, P. Annamalai, and M. Megharaj, "Pesticides in the urban environment: A potential threat that knocks at the door," *Science of The Total Environment*, vol. 711, p. 134 612, 2020.

[3] C. Cariou, Z. Gobor, B. Seiferth, and M. Berducat, "Mobile robot trajectory planning under kinematic and dynamic constraints for partial and full field coverage," *Journal of Field Robotics*, vol. 34, no. 7, pp. 1297–1312, 2017.

[4] I. 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.

[5] C.-W. Jeon, H.-J. Kim, C. Yun, X. Han, and J. H. Kim, "Design and validation testing of a complete paddy field-coverage path planner for a fully autonomous tillage tractor," *Biosystems Engineering*, vol. 208, pp. 79–97, 2021.

[6] J. Jin and L. Tang, "Optimal coverage path planning for arable farming on 2d surfaces," *Transactions of the ASABE*, vol. 53, no. 1, pp. 283–295, 2010.

[7] T. Oksanen and A. Visala, "Coverage path planning algorithms for agricultural field machines," *Journal of field robotics*, vol. 26, no. 8, pp. 651–668, 2009.

[8] R. S. Nilsson and K. Zhou, "Method and bench-marking framework for coverage path planning in arable farming," *Biosystems Engineering*, vol. 198, pp. 248–265, 2020.

[9] K. Zhou and D. Bochtis, "Route planning for capacitated agricultural machines based on ant colony algorithms.," in *HAICTA*, 2015, pp. 163–173.

[10] M. G. Plessen, "Freeform path fitting for the minimisation of the number of transitions between headland path and interior lanes within agricultural fields," *arXiv preprint arXiv:1910.12034*, 2019.

[11] K. Zhou, A. L. Jensen, D. Bochtis, M. Nørremark, D. Kateris, and C. G. Sørensen, "Metric map generation for autonomous field operations," *Agronomy*, vol. 10, no. 1, p. 83, 2020.

[12] G. T. Edwards, J. Hinge, N. Skou-Nielsen, A. Villa-Henriksen, C. A. G. Sørensen, and O. Green, "Route planning evaluation of a prototype optimised infield route planner for neutral material flow agricultural operations," *Biosystems Engineering*, vol. 153, pp. 149–157, 2017.

[13] Y. Cao, Y. Han, J. Chen, X. Liu, Z. Zhang, and K. 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.

[14] Y. Cao, Y. Han, J. Chen, X. Liu, Z. Zhang, and K. 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.

[15] I. A. Hameed, "Intelligent coverage path planning for agricultural robots and autonomous machines on three-dimensional terrain," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 3-4, pp. 965–983, 2014.

[16] D. Pour Arab, M. Spisser, and C. Essert, "Complete coverage path planning for wheeled agricultural robots," 2022. DOI: 10.22541/au.166869706.64844882/v1.

[17] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[18] J. Reeds and L. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pacific journal of mathematics*, vol. 145, no. 2, pp. 367–393, 1990.