2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Collision-free Coverage Path Planning for the Variable-speed Curvature-constrained Robot

Lin Li[1,3], Dianxi Shi[2], Songchang Jin[2], Yixuan Sun[1], Xing Zhou[4], Shaowu Yang[1], and Hengzhu Liu[1]

***Abstract*— Dubins coverage has been extensively researched to address the coverage path planning (CPP) problem of a known environment for the curvature-constrained robot. However, its fixed-speed assumption prevents the robot from accelerating to reduce the time and limits its flexibility to avoid obstacles. Therefore, this paper presents a collision-free CPP approach (CFC) for the obstacle-constrained environment, which enhances time efficiency by constructing the variable-speed Dubins paths and ensures robot safety by building a risk potential surface for representing the possibility of collision. Furthermore, CFC models the CPP problem as an asymmetric traveling salesman problem (ATSP) and utilizes a graph pruning strategy to reduce the computational cost. Comparison tests with other Dubins coverage methods demonstrate that CFC provides shorter coverage times and better runtimes than the other Dubins coverage methods while preventing collision risk between the robot and obstacles. Physical experiments in a laboratory setting demonstrate the applicability of CFC to the physical robot.**

I. INTRODUCTION

Typical robotic applications [1]–[5] require appropriate coverage path planning (CPP). CPP aims to determine a path to cover the interested area while minimizing time and avoiding obstacles [6]. Roughly, the complexity of the CPP problem is determined by factors such as the kinematic constraints of robots and the complexity of the mission environment [7]. Other literature and our previous work [8] have shown that, in general, a robot with non-holonomic constraints is more challenging to plan than a robot with holonomic constraints [9], [10], and a strictly obstacle-constrained environment is more involved in planning than an open and obstacle-free environment [11].

Real-world CPP applications typically involve an obstacle-constrained environment and non-holonomic robots. The typical kinematic constraints of a robot are curvature and speed [12]. Along this line, Dubins [13] addressed the shortest path planning problem for a robot moving at a fixed speed and bounded curvature in an open area. Given that the Dubins-like paths have the advantages of easy implementation and high efficacy [14], several Dubins coverage methods [3], [6], [10], [15]–[18] utilizing Dubins paths have been extensively researched. Unfortunately, the Dubins coverage problem has been proved to be NP-hard [7].

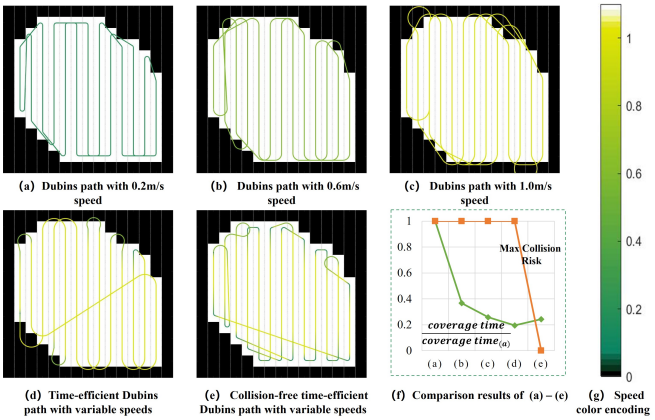

Fig. 1. Comparison of Dubins coverage path with fixed/variable speeds. All paths are color-mapped based on the speed information. The path is considered safe if its maximum collision risk is zero.

Although Dubins coverage methods perform well, they suffer from two issues. The first one is that Dubins coverage methods assume that the robot moves at a fixed speed, limiting its versatility and flexibility given that it can travel at variable speeds within mechanical capability—this property can sometimes lessen time cost [9], [12]. Fig. 1 demonstrates an example of conventional Dubins coverage versus variable-speed Dubins coverage. Currently, most path planning methods for the variable-speed curvature-constrained vehicle focus on point-to-point path planning [19]–[21]. However, the time-efficient CPP problem for variable-speed curvature-constrained robots has not been solved yet.

The second issue is the safety of the robot in an obstacle-constrained environment. Some Dubins coverage approaches assume that the mission environment has no obstacles or obstacles that can be traversed. However, impenetrable obstacles may exist within or at the boundary of the mission environment in practical applications such as field monitoring [22], battlefield loss assessment [23], and automation in agriculture [24]. As a result, adapting these methods to obstacle-constrained environments may yield unfeasible results. Other coverage methods [25]–[27] protect the robot from collision disaster by placing a buffer field called headland around obstacles to execute the turning maneuver. However, since turns are considered expensive and non-working, a wider headland may introduce a low coverage ratio, while a narrower headland increases the possibility of collision.

This paper proposes a collision-free CPP approach (CFC) for the variable-speed curvature-constrained robot. CFC di-

*This work was supported by the National Natural Science Foundation of China (Grant No.91948303)

[1] College of Computer, National University of Defense Technology, Changsha, Hunan 410003, China.
[2] Artificial Intelligence Research Center (AIRC), Defense Innovation Institute, Beijing 100166, China.
[3] Tianjin Artificial Intelligence Innovation Center (TAIIC), Tianjin 300457, China.
[4] College of Intelligence Science and Technology, National University of Defense Technology, Changsha, Hunan 410003, China.
Correspondence: dxshi@nudt.edu.cn, jsc04@tsinghua.org.cn

vides the area into cells and generates the coverage path in two steps. First, a path planning based on risk pruning (P2RP) algorithm is proposed to produce a collision-free time-efficient path between cells. It characterizes the time-efficient path with variable speeds to reduce the time cost and constructs a risk potential surface representing the possibility of collision to avoid the obstacles. Next, a sequence generation (SG) algorithm is presented to determine the optimal sequence of all cells. The SG algorithm simplifies the computation by modeling CPP as an ATSP rather than the conventional generalized TSP (GTSP) and reducing its search space through graph pruning. Compared with the traditional Dubins coverage methods, CFC allows the robot to speed up in open areas to reduce coverage time and slow down near obstacles to decrease collision risks. CFC was validated in both simulations and real-world experiments. In summary, the contributions of this paper are as follows:

- We present a P2RP algorithm, which improves time efficiency by constructing variable-speed Dubins paths and ensures robot safety by build a risk potential representing the possibility of collision.
- We reduce the computational cost by modeling the CPP problem as an ATSP and a graph pruning strategy.
- Extensive validations. CFC's performance is validated by: i) simulations of different-sized scenes, which show fewer coverage times and better computation times than Dubins coverage methods while preventing the collision risks with obstacles, ii) real-world laboratory experiments.

II. PROBLEM STATEMENT

This paper assumes that the prior knowledge about the mission environment $\mathscr{A} \subset \mathbb{R}^2$ exists and has been modeled as a binary map. The cell with values of 0 and 1 on the map represents the obstacle cell or the allowed cell, respectively. The robot assigned to the coverage task occupies a limited area, like a rectangle with a certain width. It is equipped with a task sensor to perform the specific coverage task (e.g., lawn mowing, floor cleaning) with a rectangle region of radius r_2. Moreover, a minimum safe distance r_1 around the robot is reserved to prevent collisions introduced by the inertia or the robot's body.

At any given time t, the motion of the robot can be described with the following tuple:

$$\begin{cases} \dot{x}(t) = s(t)\cos\theta(t) \\ \dot{y}(t) = s(t)\sin\theta(t) \\ \dot{\theta}(t) = u(t) \end{cases}, \tag{1}$$

where $\{x,y\} \in \mathbb{R}^2, \theta \in [0, 2\pi)$; $u(t)$ and $s(t)$ are the robot's turning rate and speed at time t, respectively. The robot can travel at variable speeds, s.t., $s \in [s_{min}, s_{max}]$, where s_{min} and s_{max} represent the minimum and the maximum speed, respectively. s_{min} and s_{max} are determined by several factors, such as the peak power of the motor and soft controller [28], [29]. The acceleration a of robot is limited by the minimum and maximum values, s.t., $a \in [a_{min}, a_{max}]$. The turning rate u is bounded and symmetric, s.t., $u \in [-u_{max}, +u_{max}]$, where u_{max} and the $+/-$ represent the maximum turning rate and the left/right turn, respectively. The curvature $\kappa = \frac{u}{s}$, which is the inverse of the turning radius, depends on the speed and turning rate, s.t.,$|\kappa| \in [0, \frac{u_{max}}{v_{min}}]$. $\kappa = 0$ indicates that the robot travels at a straight line, while the maximum curvature represents the robot turning at the maximum speed and minimum turning rate. The objective of the CPP is to produce a path that covers all non-obstacle areas of $\mathscr{A}$ under the sensor perception of the task sensor while avoiding obstacles and minimizing the coverage time.

III. COLLISION-FREE COVERAGE PATH PLANNING APPROACH (CFC)

This paper proposes an offline CFC to address the CPP problem in obstacle-constrained environments. As depicted in Fig. 2, CFC consists of an area decomposition module, the P2RP module, and the SG module. The area decomposition module receives the binary map of the mission environment and divides it into cells, while the P2RP module generates the collision-free time-efficient path between cells. Next, the SG module models the CPP problem into 0-1 integer linear programming (0-1 ILP) and reduces its complexity by a graph pruning strategy. The safe-and-speedy coverage path is obtained by determining the optimal sequence of all cells.

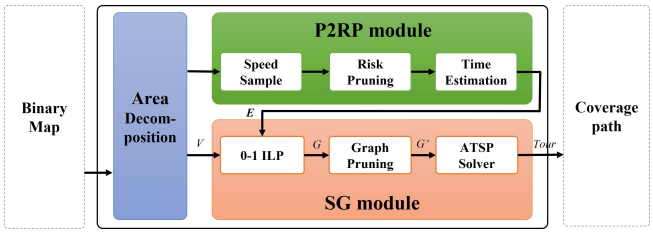

Fig. 2. The framework of CFC.

A. Area Decomposition

The area decomposition module utilizes the boustrophedon cellular decomposition (BCD) method [30] to divide the mission environment into multiple cells. All cells constitute a cell set $T = \{\tau_\alpha \in \mathbb{R}^2, \alpha = 1, ..., |T|\}$. Every cell in T has a fixed width (i.e., $2r_2$), and its height depends on the boundary of obstacles and the mission environment. It is assumed that each cell is long enough for the robot to accelerate from s_{min} to s_{max} or slow from s_{max} to s_{min}. As shown in Fig. 3, each cell consists of the upper and lower endpoints (i.e., v_u and

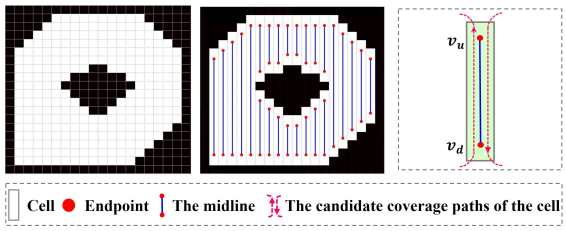

Fig. 3. [Left] The binary map. [Middle] Area decomposition. [Rigth] The endpoints of each cell.

v_d) and a line segment connecting two endpoints. The robot can cover the cell from either endpoint to another one.

B. P2RP algorithm

The curvature-constrained collision-free path planning problem in an obstacle-constrained environment has been proved to be NP-hard [31]. Thus, this paper proposes a heuristic P2RP algorithm to solve the above-mentioned problem. The P2RP algorithm simultaneously considers the travel time and obstacle clearance. It obtains the collision-free time-efficient path between cells by the following three steps:

1) Speed sample: Traditionally, the Dubins path assumes that the robot moves forward at a fixed speed. The fixed speed, however, prevents the robot from using its acceleration capabilities to decrease the time and restricts its flexibility in bypassing obstacles. Inspired by the fact that changing the speed or turning radius of the Dubins paths can reduce the travel time [21], the P2RP algorithm constructs the variable-speed Dubins paths to improve the time efficiency.

The variable-speed Dubins path extends the original Dubins path [13] and consists of three path segments, which can be represented by the sequence *CSC* (*LSR*, *RSR*, *RSL*, and *LSR*) or *CCC* (*RLR* and *LRL*) [32], where C presents the curve segments denoted by *R/L* and S refers to the straight segment. The P2RP algorithm uniformly samples the robot speed to obtain a set of speed $S = \{s_1, ..., s_K\}$. The pairs of speed in S can be combined into a speed pair set $SP = \{(s_\alpha, s_\gamma), s_\alpha, s_\gamma \in S, \alpha, \gamma = 1, ..., K\}$. Each (s_α, s_γ) in SP corresponds to the speed of the first and third C segments in the variable-speed Dubins path. Since the C segment assumes a certain curvature $\kappa = \frac{u}{v}$ (i.e., a certain speed and turning rate), the robot can only accelerate or decelerate on the S segment [21]. Due to this, the C segments in the *CSC* sequence can have different speeds, while the *CCC* sequence has to remain a fixed speed since it has no S segment. Given a start pose (x, y, θ), the next pose $L_\delta/R_\delta/S_\delta$ of the robot in the *L/R/S* segments of the variable-speed Dubins path are as follows [19]:

$$L_\delta(x,y,\theta) = (x - r_\delta \sin\theta + r_\delta \sin(\theta+\delta), \\ y + r_\delta \cos\theta - r_\delta \cos(\theta+\delta), \theta+\delta) \tag{2}$$

$$R_\delta(x,y,\theta) = (x + r_\delta \sin\theta - r_\delta \sin(\theta-\delta), \\ y - r_\delta \cos\theta + r_\delta \cos(\theta-\delta), \theta-\delta) \tag{3}$$

$$S_\delta(x,y,\theta) = (x + \delta \sin\theta, y + \delta \cos\theta, \theta) \tag{4}$$

where $\delta \in [0, 2\pi)$ in Eq. (2) and Eq. (3) represents the radian offset from the start pose, whereas the $\delta \in R^+$ in Eq. (4) presents the moving distance of the robot from the start pose; r_δ represents the turning radius of the C path segment, s.t., $r_\delta = \frac{s}{u}$.

2) Risk Pruning: In order to ensure the robot's safety, the variable-speed Dubins paths should maintain a safe distance from obstacles. The obstacle clearance is defined as the minimum distance between a path and obstacles. Several path planning methods addressed the obstacle clearance problem by measuring the collision risk based on the robot's state

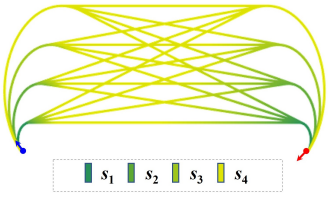

Fig. 4. An example of the variable-speed Dubins paths with the type of *RLR*. The robot moves at a fixed speed in the circle segments, but the variable speed in the straight segment. The paths are encoded by the speed information.

[12], [19], [33]. Although these methods perform well, they suffer from a high computational cost since the clearance calculation must be performed for every state configuration and its intermediate steps. Alternatively, we propose heuristically establishing a potential surface around obstacles to represent collision risk between robots and obstacles. Consider a point $\tau \in \mathscr{A}$, its risk potential is calculated as follows:

$$\mathscr{R}(\tau) = \begin{cases} 1 - \frac{d}{r_1} & d \le r_1 \\ 0 & d > r_1 \end{cases}, \tag{5}$$

where d represents the distance between point τ and the nearest obstacles; r_1 refers to the minimum safe distance of the robot. Eq. (5) indicates that the robot in τ is considered safe if d is wider than r_1, i.e., $\mathscr{R}(\tau) = 0$. As $d \leftarrow 0$, the risk values approaches 1. Fig. 5(a) shows an example of the risk potential surface.

After building the risk potential surface, the P2RP algorithm determines whether a variable-speed Dubins path is safe by looking up the risk potential of each point in the path. If there is at least a point τ, s.t., $\mathscr{R}(\tau) > 0$, the path is considered with the risk of collision with obstacles; otherwise, the path is collision-free. Fig. 5(c) demonstrate an example of the risk pruning strategy.

3) Time estimation: After eliminating paths with collision risks, the P2RP algorithm evaluates the time cost of the remained paths and outputs the one with the least time. Since the variable-speed Dubins path consists of three path segments, its total time cost $\mathscr{M}$ is the sum of its three segments,

$$\mathscr{M} = \frac{\alpha}{s_\alpha} + \frac{\beta}{s_\beta} + \frac{\gamma}{s_\gamma}, \tag{6}$$

where s_α, s_β, and s_γ represent the robot's speed in the first to third segment of the variable-speed Dubins path; α, β, γ represent the corresponding path lengths. As the only variable in Eq. (6), s_β is determined by the type of the variable-speed Dubins path. Since the Dubins paths with the

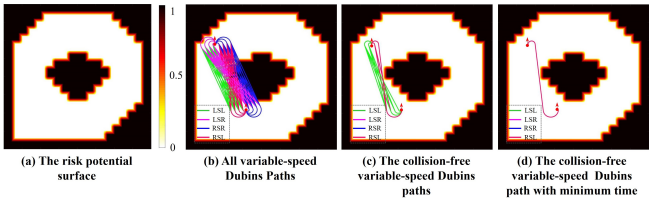

Fig. 5. The produce of the P2RP algorithm

CCC sequence do not have a S segment to change speed, its s_β is equal to s_α and s_γ. The s_β of Dubins paths with the *CSC* sequence depends on the length of the S path segment. We utilized the time-optimized speed profile in [21] which consider both speed and forward acceleration limits, to calculate its time cost. The collision-free path with minimum time cost will be the output. Fig. 5 demonstrates an example of the P2RP algorithm.

Algorithm 1 shows the pseudo-code of the P2RP algorithm. The P2RP algorithm initializes the minimum time cost (i.e., *min_time*) and the speed pair *bs* of the output path (Line 1) and builds the risk potential surface (Line 2). Next, it samples the speed to obtain the speed pair set *SP* (Line 3-4). For each speed pair in *SP* (Line 5), the P2RP algorithm simulates the variable-speed Dubins path (Line 6) and calculates its maximum risk potential and the time cost (Line 6-7). *min_time* and *bs* will be updated if the path is collision-free and with fewer time cost (Line 9-13). After all paths are evaluated, the *min_time* and *bs* are returned.

Algorithm 1 P2RP algorithm

Input: $start, dest, \mathscr{A}, [s_{min}, s_{max}], [a_{min}, a_{max}]$.
Output: min_time, bs.

1: Initialize: $min\_time \leftarrow Inf, bs \leftarrow \emptyset$;
2: $Pot \leftarrow$ Construct_Risk_Potential($\mathscr{A}$);
3: $S \leftarrow$ Speed_sample(s_{min}, s_{max});
4: $SP \leftarrow \{(s_\alpha, s_\gamma), s_\alpha, s_\gamma \in S\}$;
5: **for** each $(s_\alpha, s_\gamma) \in SP$ **do**
6: $Dpath \leftarrow$ variable-speed_Dubins_path(s_α, s_γ);
7: $max\_risk \leftarrow$ risk_pruning($Dpath, Pot$);
8: $time\_cost \leftarrow$ time_estimation($Dpath, a_{min}, a_{max}$) ;
9: **if** $max\_risk = 0$ and $time\_cost < min\_time$ **then**
10: $min\_time \leftarrow time\_cost$;
11: $bs \leftarrow (s_\alpha, s_\gamma)$;
12: **end if**
13: **end for**
14: return min_time, bs;

Computational complexity: The computational complexity of the P2RP algorithm is $O(K^2)$.

C. SG algorithm

Since the Dubins coverage problem has been proved to be NP-hard [7], this section develops an SG algorithm to provide an approximate solution. In order to decrease the computational cost, the SG algorithm models the CPP problem as an 0-1 ILP and reduces its search space using a heuristic graph pruning strategy, thereby providing a safe-and-speedy coverage path.

1) 0-1 ILP: Offline coverage methods usually divided the mission environment into cells and presented all cells on a graph. With this graph representation, the Dubins coverage problem can be modeled as a GTSP, which produces the coverage path by finding a minimum cost cycle that includes precisely one node from each cluster [7] [10] [16]. Although these methods perform well, they had to convert the GTSP into the ATSP by the Noon and Bean transformation [34] or doubling the vertices [35], and then utilize the ATSP solvers to obtain the optimal solution. This paper represents cells as a connected graph $G = \{V, E\}$, with the vertex representing the endpoint of cells and the edge between vertices representing the collision-free time-efficient path produced by the P2RP algorithm. Unlike the Dubins coverage methods, the SG algorithm models the CPP problem as a 0-1 ILP. The 0-1 ILP model is essentially an ATSP, thus preventing the complex computation for transforming the GTSP to the ATSP.

The 0-1 ILP model is presented as Eq. (7)-(8):

$$min(\sum(|e_{i,j}|x_{i,j})), \tag{7}$$

s.t.,

$$\begin{aligned} \sum_{i=1}^{2|T|}\sum_{j=1}^{2|T|} x_{i,j} &= 2|T|, \\ \sum_{j=1}^{2|T|} x_{i,j} = \sum_{j=1}^{2|T|} x_{j,i} &= 1, i = 1, ..., 2|T|, \\ x_{2k+1,2k+2} + x_{2k+2,2k+1} &= 1, k = 1, ..., |T|, \end{aligned} \tag{8}$$

where the binary variable $x_{i,j} = 1$ indicates that the edge $e_{i,j}$ between the vertex v_i and v_j has been selected into the optimal solution, otherwise $x_{i,j} = 0$; $|e_{i,j}|$ represents the weight of $e_{i,j}$. The objective of the 0-1 ILP is to find a traversal sequence of all vertices with a minimum cost. Its constraints indicate that the robot should travel all $2|T|$ vertices of G, enter and leave each vertex for exactly once, and visit the vertices (i.e., the endpoints) that belong to the same cell sequentially.

2) Graph Pruning: The performance of the ATSP depends on the complexity of the graph G since more edges and vertices mean a bigger search space. To combat the complexity, the authors in [7] presented a specified directions decomposition (SDC) algorithm. Although the SDC algorithm improves the runtime, its optimality is sacrificed to some extent since it deletes half of the vertices and its incident edges. This paper presents a heuristic graph pruning strategy to reduce the search space by eliminating some edges. The graph pruning strategy is derived from the observation that the edges with the largest weight are less likely to appear in the optimal plan than those with less weight. It contains two heuristics methods:

- Edge weight. Denote the set of edges entering the vertex v_α as E_α. In E_α, the top $w|E_\alpha|$ weighted edges will be removed, where w and $|E_\alpha|$ represent a predefined percentage and number of edges, respectively. The edges that leave v_α are trimmed similarly.
- Vertex type. Graph G contains two types of vertices, those at the top and those at the bottom of the cell. Suppose that the set of vertices, which connect to v_α and has the same type of vertex as v_α, is V'_α. Compute the mean u and standard deviation δ of E'_α (E'_α represents the edge set between v_α and V'_α). The edges whose weight exceeds $u + \delta * w$ in E'_α will be deleted, where w is a predefined constant.

Fig. 6 shows an example of the SG algorithm.

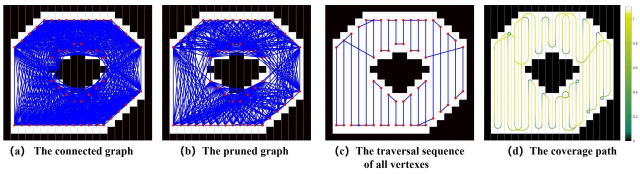

Fig. 6. The procedure of the SG algorithm.

Algorithm 2 shows the pseudo-code of the SG algorithm. The SG algorithm first represents the cells into a graph G (Line 1) and calculates its cost matrix C through the P2RP algorithm (Line 2). Next, the graph pruning strategy eliminates edges by setting their weights as a large positive number (Line 3). The SG algorithm uses the state-of-art ATSP solver [36] to obtain the optimal traversal sequence *Tour* (Line 4) and finally produce a collision-free time-efficient coverage path in the form of a waypoint sequence (Lines 5).

Algorithm 2 SG algorithm

Input: $T, [s_{min}, s_{max}], [a_{min}, a_{max}]$.
Output: *path*.

1: $G = (V, E) \leftarrow$ Generate_Graph(V);
2: $C \leftarrow$ Calculate_Cost_Matrix($G, [s_{min}, s_{max}], [a_{min}, a_{max}]$) ; //According to the P2RP Algorithm
3: $C' \leftarrow$ Graph_Prune(C) ;
4: $Tour \leftarrow$ SolveATSP(C') ; // According to Eq. (7)-(8)
5: $path \leftarrow$ Generate_path($Tour, [s_{min}, s_{max}], [a_{min}, a_{max}]$);
6: Return *path*;

Computational complexity: The complexity of the SG algorithm is $O(|T|^3)$.

IV. Experiment

The performance of CFC was validated by simulations as well as real experiments.

A. Simulation Experiments

Setup: The first validation was performed on a PC with an Intel(R) Core(TM) CPU 3.2GHz RAM 32GB. Two scenarios populated with multiple obstacles were drawn with dimensions of 20 × 20 m and 30 × 30 m and divided into multiple cells of 1 m. A curvature-constrained robot was simulated with kinetic constraints, such as the speed [0.2, 1] m/s, the acceleration [0.5, 1.5] m^2/s, the maximum turning rate 1 rad/s, the minimum turning radius 0.2 m, and the minimum safe distance (i.e., r_1) 0.25 m. Furthermore, the robot was equipped with a task sensor with a radius (r_2) of 0.5 m. The speed set S of the robot is set as $\{0.2, 0.36, 0.52, 0.68, 0.84, 1\}$ m/s. In addition, the headlands around obstacles were also reserved for executing the turning maneuvers. Since the headland width affects the performance of coverage methods, seven widths $[1, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0] * r_2$ were tested.

We compared CFC with two Dubins coverage methods: MinNWT [16] and Semi-BCD [7]. MinNWT [16] produced the safe path by eliminating the edges between cells separated by obstacles. In practical applications containing impenetrable obstacles, Semi-BCD may produce a coverage path overlapping with obstacles since it assumes that obstacles in the area can be passed through. Both Dubins coverage algorithms assumed that the robot moves at a fixed speed, i.e., its speed has to be one of the speeds in S. In order to obtain a fair comparison, we compared CFC with the two Dubins coverage methods at different speeds with different headland widths. For the same headland width, the results of two Dubins coverage methods with different speeds are represented by a line connecting six points (each point corresponds to the results of the Dubins coverage method at a sampling speed in S). In contrast, the results of CFC are displayed with a dashed blue line across all speeds.

Results: Fig. 7 and Fig. 8 show the comparison results of CFC, MinNWT [16] and Semi-BCD [7]. Four metrics are used for the performance evaluation as follows: *i) coverage time, ii) maximum collision risk, iii) coverage rate, and iv) computation time*. First, in Fig. 7(a) and Fig. 8(a) we show the coverage times of CFC, MinNWT and Semi-BCD. It is observed that CFC always provides the shortest or near shortest coverage times. Furthermore, Fig. 7(b) and Fig. 8(b) show that the maximum collision risk of CFC remains zeros, i.e., its coverage paths always be safe. In contrast, the maximum collision risk of MinNWT [16] and Semi-BCD [7] depends on the headland width and the robot speed. Their coverage paths become safe only if the headland width is greater or equal to $3 \times r_2$. In addition, as shown in Fig. 7(c) and Fig. 8(c), the coverage ratios of the three coverage methods are all sensitive to the width of the headland. The wider the headland, the lower the coverage ratio. With a headland less than or equal to $2 \times r_2$, CFC can provide a greater than or equal to 95% coverage ratio. Lastly, as shown in Fig. 7(d) and Fig. 8(d), CFC always takes less time except

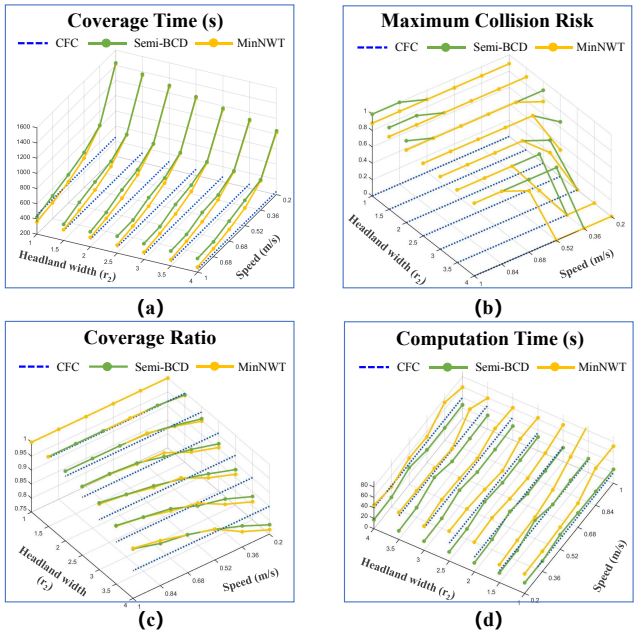

Fig. 7. The comparison of CFC with MinNWT [16] and Semi-BCD [7] in 20 m × 20 m scene.

Semi-BCD [7] in the 20 m × 20 m scene. However, Semi-BCD [7] reduced the runtime by sacrificing some optimality of the coverage time.

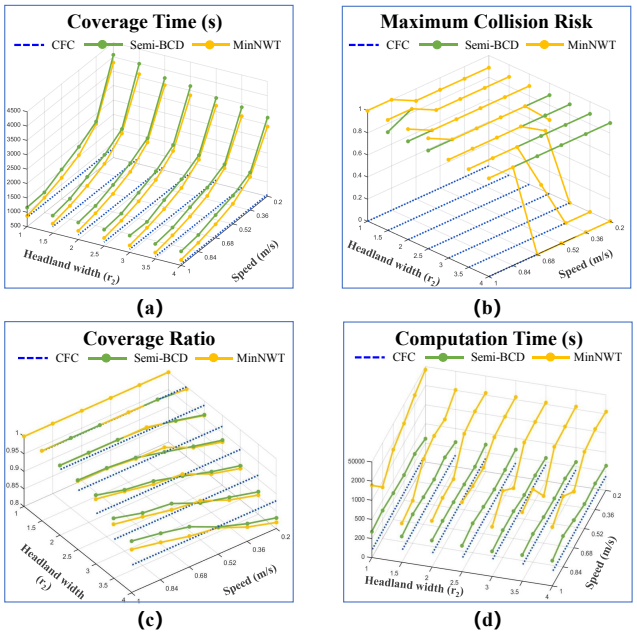

Fig. 8. The comparison of CFC with MinNWT [16] and Semi-BCD [7] in 30 m × 30 m scene.

B. Real Experiments

Real experiments in a laboratory setting further validated the performance of CFC. The laboratory area had a size of 9.5 m × 10.5 m and was populated with multiple obstacles. The prior knowledge (i.e., the grid map) of the area was obtained by the ROS SLAM package [37], and we utilized the BCD method to divide it into cells with a width of 0.72 m. An Ackerman robot with dimensions of 0.3 m × 0.24 m × 0.17 m was used for real experiments. Its kinetic constraints included the speed of $[0.3, 0.5]$ m/s [1], the acceleration of $[0.5, 1.0]$ m^2/s, the minimum turning radius of 0.3 m, a maximum turning rate of 1.0 rad/s, and a minimum safe distance 0.35 m. The speed set S was set as $\{0.3, 0.4, 0.5\}$ m/s. In addition, a positioning package [38] and the pure pursuit algorithm [29] were integrated with the robot for real-time positioning and path following.

Before performing real-world experiments, we conducted preliminary tests to ensure that the robot can follow the trajectory generated by CFC. Preliminary tests have shown that if the robot turns very close to an obstacle, the positioning package [38] may provide inaccurate positions. Inaccurate positions probably result in the robot turning in circles constantly, leading to a catastrophic outcome. In order to avoid this catastrophic outcome, we set the width of the headland to 0.90 m.

Following the preliminary tests, CFC was utilized to produce a safe-and-speedy coverage path in the form of a waypoint sequence. Fig. 9 shows the laboratory area, the Ackerman robot, and the coverage path encoded by the speed information. In addition, a laptop computer, which communicates with the Ackerman robot through a wireless connection, was used to receive and display the robot's poses. Fig. 10 shows the snapshots of the real-world experiment, where the robot successfully followed the coverage path generated by CFC.

[1]The speed is set as $[0.3, 0.5]$ m/s since the path following algorithm [29] performs well only when the robot speed is less than 0.6 m/s in the preliminary tests.

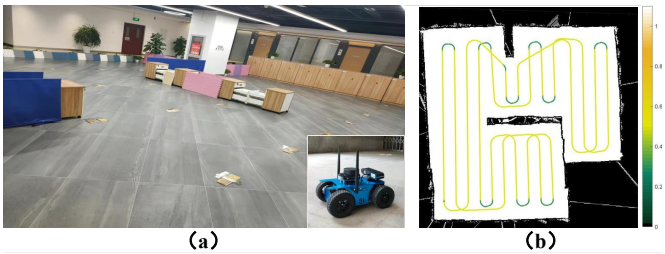

Fig. 9. The laboratory area, the Ackermann robot, and the coverage path produced by CFC.

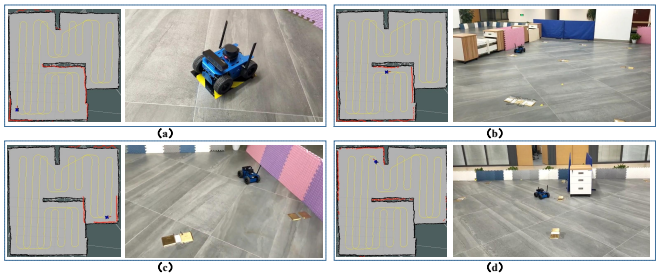

Fig. 10. Snapshots of the real experiment. In every subfigure, the left and right parts represent the robot pose displayed by the Rviz and the snap shot of the robot in the lab.

V. Conclusion

This paper presents an offline CPP approach called CFC for the variable-speed curvature-constrained robot. CFC generates the coverage path by a P2RP algorithm and an SG algorithm. The P2RP algorithm reduces the time cost by selecting appropriate speeds for the robot and ensures obstacle clearance by constructing a risk potential surface, thereby providing a collision-free time-efficient path between cells. The SG algorithm decreases the computational cost by modeling the CPP problem into ATSP and a graph pruning strategy. Compared with other Dubins coverage methods, CFC generates a time-efficient coverage path while ensuring robot safety. CFC is validated via simulations and real-world experiments. Experimental results show that CFC completes coverage in shorter coverage times and computation times while avoiding obstacles and is applicable for the real robot.

One limitation of our method is that it can only address the CPP problem of a single robot in known environments. However, there have been several practical applications that are beyond the capabilities of a single robot or involve unknown environments. Thus, future works include extending the multi-robot system and the online path planning to unknown environments.

REFERENCES

[1] T. Duckett, S. Pearson, S. Blackmore, B. Grieve, W.-H. Chen, G. Cielniak, J. Cleaversmith, J. Dai, S. Davis, C. Fox, *et al.*, "Agricultural robotics: the future of robotic agriculture," *arXiv preprint arXiv:1806.06762*, 2018.

[2] Y. Wu, B. Zhang, X. Yi, and Y. Tang, "Communication-motion planning for wireless relay-assisted multi-robot system," *IEEE Wireless Communications Letters*, vol. 5, no. 6, pp. 568–571, 2016.

[3] N. Karapetyan, A. Braude, J. Moulton, J. A. Burstein, S. White, J. M. O'Kane, and I. Rekleitis, "Riverine coverage with an autonomous surface vehicle over known environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3098–3104.

[4] X. Kan, H. Teng, and K. Karydis, "Online exploration and coverage planning in unknown obstacle-cluttered environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5969–5976, 2020.

[5] P. Zhang, H. Wang, B. Ding, and S. Shang, "Cloud-based framework for scalable and real-time multi-robot slam," in *2018 IEEE International Conference on Web Services (ICWS)*. IEEE, 2018, pp. 147–154.

[6] C. S. Tan, R. Mohd-Mokhtar, and M. R. Arshad, "A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms," *IEEE Access*, 2021.

[7] J. S. Lewis, W. Edwards, K. Benson, I. Rekleitis, and J. M. O'Kane, "Semi-boustrophedon coverage with a dubins vehicle," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 5630–5637.

[8] X. Zhou, H. Wang, and B. Ding, "How many robots are enough: a multi-objective genetic algorithm for the single-objective time-limited complete coverage problem," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2380–2387.

[9] K. Kučerová, P. Váňa, and J. Faigl, "Variable-speed traveling salesman problem for vehicles with curvature constrained trajectories," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 4714–4719.

[10] N. Karapetyan, J. Moulton, J. S. Lewis, A. Q. Li, J. M. O'Kane, and I. Rekleitis, "Multi-robot dubins coverage with autonomous surface vehicles," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2373–2379.

[11] B. Sakcak and S. M. LaValle, "Complete path planning that simultaneously optimizes length and clearance," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 10 100–10 106.

[12] J. Song, S. Gupta, and T. A. Wettergren, "T*: Time-optimal risk-aware motion planning for curvature-constrained vehicles," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 33–40, 2018.

[13] 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.

[14] X.-N. Bui, J.-D. Boissonnat, P. Soueres, and J.-P. Laumond, "Shortest path synthesis for dubins non-holonomic robot," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE, 1994, pp. 2–7.

[15] A. Xu, C. Viriyasuthee, and I. Rekleitis, "Optimal complete terrain coverage using an unmanned aerial vehicle," in *2011 IEEE International conference on robotics and automation*. IEEE, 2011, pp. 2513–2519.

[16] X. Yu, T. A. Roppel, and J. Y. Hung, "An optimization approach for planning robotic field coverage," in *IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2015, pp. 004 032–004 039.

[17] X. Yu, "Optimization approaches for a dubins vehicle in coverage planning problem and traveling salesman problems," Ph.D. dissertation, 2015.

[18] J. Conesa-Muñoz, G. Pajares, and A. Ribeiro, "Mix-opt: A new route operator for optimal coverage path planning for a fleet in an agricultural environment," *Expert Systems with Applications*, vol. 54, pp. 364–378, 2016.

[19] J. P. Wilson, K. Mittal, and S. Gupta, "Novel motion models for time-optimal risk-aware motion planning for variable-speed auvs," in *OCEANS 2019 MTS/IEEE SEATTLE*. IEEE, 2019, pp. 1–5.

[20] A. Wolek, E. M. Cliff, and C. A. Woolsey, "Time-optimal path planning for a kinematic car with variable speed," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 10, pp. 2374–2390, 2016.

[21] K. Kučerová, P. Váň, and J. Faigl, "On finding time-efficient trajectories for fixed-wing aircraft using dubins paths with multiple radii," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020, pp. 829–831.

[22] X. Kan, H. Teng, and K. Karydis, "Online exploration and coverage planning in unknown obstacle-cluttered environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5969–5976, 2020.

[23] X. Yang, L. M. Alvarez, and T. Bruggemann, "A 3d collision avoidance strategy for uavs in a non-cooperative environment," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1, pp. 315–327, 2013.

[24] X. Zhang, C. Fan, Z. Cao, J. Fang, and Y. Jia, "Novel obstacle-avoiding path planning for crop protection uav using optimized dubins curve," *International Journal of Agricultural and Biological Engineering*, vol. 13, no. 4, pp. 172–177, 2020.

[25] J. Versleijen and S. de Bruin, "Path planning on agricultural fields with obstacles," *Geo-Information Science and Remote Sensing Thesis Report GIRS-2019–2021*, 2019.

[26] D. S. Paraforos, R. Hübner, and H. W. Griepentrog, "Automatic determination of headland turning from auto-steering position data for minimising the infield non-working time," *Computers and electronics in agriculture*, vol. 152, pp. 393–400, 2018.

[27] M. G. Plessen, "Optimal in-field routing for full and partial field coverage with arbitrary non-convex fields and multiple obstacle areas," *Biosystems engineering*, vol. 186, pp. 234–245, 2019.

[28] D. E. Rivera, M. Morari, and S. Skogestad, "Internal model control: Pid controller design," *Industrial & engineering chemistry process design and development*, vol. 25, no. 1, pp. 252–265, 1986.

[29] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, Tech. Rep., 1992.

[30] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and service robotics*. Springer, 1998, pp. 203–209.

[31] P. K. Agarwal, T. Biedl, S. Lazard, S. Robbins, S. Suri, and S. Whitesides, "Curvature-constrained shortest paths in a convex polygon," *SIAM Journal on Computing*, vol. 31, no. 6, pp. 1814–1851, 2002.

[32] A. M. Shkel and V. Lumelsky, "Classification of the dubins set," *Robotics and Autonomous Systems*, vol. 34, no. 4, pp. 179–202, 2001.

[33] J. D. Hernández, M. Moll, E. Vidal, M. Carreras, and L. E. Kavraki, "Planning feasible and safe paths online for autonomous underwater vehicles in unknown environments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1313–1320.

[34] C. E. Noon and J. C. Bean, "An efficient transformation of the generalized traveling salesman problem," *INFOR: Information Systems and Operational Research*, vol. 31, no. 1, pp. 39–44, 1993.

[35] V. Dimitrijević and Z. Šarić, "An efficient transformation of the generalized traveling salesman problem into the traveling salesman problem on digraphs," *Information Sciences*, vol. 102, no. 1-4, pp. 105–110, 1997.

[36] S. Narayanan, "Travelling salesman problem," *MATLAB Central File Exchange*, 2019.

[37] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[38] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C)*, vol. 2. IEEE, 1999, pp. 1322–1328.