Supplementary Material

TABLE S-I
COMPARISON OF COVERAGE PATH PLANNING ALGORITHMS

References	Year	Online	Multi-robot	Bounded	Unbounded
[14]	2014	×	✓	✓	×
[22]	2014	✓	×	✓	×
[23]	2014	✓	×	✓	×
[24]	2015	✓	✓	✓	×
[25]	2016	✓	×	✓	×
[26]	2017	✓	✓	✓	×
[27]	2017	✓	✓	✓	×
[28]	2017	✓	×	✓	×
[29]	2018	✓	✓	✓	×
[30]	2018	✓	×	✓	×
[15]	2019	×	✓	✓	×
[16]	2019	×	✓	✓	×
[17]	2019	×	✓	✓	×
[33]	2019	✓	×	✓	×
[34]	2020	✓	✓	✓	×
[18]	2020	×	✓	✓	×
[31]	2020	✓	×	✓	×
[32]	2020	✓	✓	✓	×
[19]	2022	×	✓	✓	×
[20]	2022	×	✓	✓	×
[21]	2022	×	✓	✓	×
[13]	2022	✓	✓	✓	×
Ours	2023	✓	✓	✓	✓

TABLE S-II
NOTATIONS

Symbol	Definition
$\hat{\mathbb{O}}_i$	Set of uncovered target points
$\hat{\mathbb{O}}_i'$	Set of additional uncovered target points
$\bar{\mathbb{O}}_i$	Set of covered target points
$\mathbb{O}_i$	Set of target points occupied by obstacles
N	Upper capacity limit of $\hat{\mathbb{O}}_i$
n	Number of robots
Υ	Set of robots
Φ_i	Position of the ith herd
Ψ_i^k	kth dynamic predator of robot i
r_i	Position of the ith robot
$\mathring{o}_i$	Start target point
$\hat{o}_i$	Destination target point
$\check{o}_i$	Previous target point
$\bar{o}_i$	Next target point
o_i'	Temporary target point
C_i	Set of candidate target points
o_j	jth candidate target point
o_{j^*}	Candidate target point with maximal reward
$R(o_j)$	Total reward of o_j
$R^H(o_j)$	Herd attraction reward of o_j
$R^D(o_j)$	Dynamic predator avoidance reward of o_j
$R^S(o_j)$	Smoothness reward of o_j
$R^B(o_j)$	Boundary reward of o_j
$W(o_j)$	missing reward of o_j
Ω	Reward function parameters
λ	Weighting factor for missing reward
ω^D	Weighting factor for dynamic predator avoidance reward
ω^S	Weighting factor for smoothness reward
ω^B	Weighting factor for boundary reward
$H(o_j)$	Distance from o_j to Φ_i
$D(o_j, \Psi_i^k)$	Distance from o_j to Ψ_i^k
$S(\hat{o}_i, \Psi_i^k)$	Inverted sigmoid function
κ	Slope
a	Distance from $\hat{o}_i$ to Ψ_i^k
b	Effective range
$\Theta(o_j)$	Deflection angle $\angle\check{o}_i\hat{o}_i o_j$
$B(o_j)$	Number of uncovered and obstacle-free neighbors of o_j
η	Neighborhood radius
$\hat{t}$	Maximum time

A. Computational Complexity

Based on the communication, all the target points found have the same index among robots and their coverage status is stored using binary vectors. Thus a robot can be updated quickly by sharing the index of target points whose status has changed since the last communication, and its complexity is $\mathcal{O}(\log n)$. A k-d tree and a B+ tree are used to store $\hat{\mathbb{O}}_i$, where the k-d tree is used to find candidate target points in the neighborhood of $\hat{o}_i$, and the B+ tree is used to find candidate target points between $\hat{o}_i$ and the virtual herd Φ_i. Since the upper capacity limit of $\hat{\mathbb{O}}_i$ is N, the complexity is $\mathcal{O}(\log N)$. The time complexity of inserting or deleting a point in the k-d tree and the B+ tree is also $\mathcal{O}(\log N)$. When a robot is at a dead end, its time complexity to get out of it depends on the point-to-point planner. In addition, the complexity of scanning the environment is not considered in this analysis. Generally, the number of robots $n \ll N$, so the overall time complexity of DH-CPP is $\mathcal{O}(\log N)$.

B. Analysis of Running Time

In bounded environments, the movable area is fixed and a robot is more affected by obstacles and other robots. Thus a robot is more likely to fall into a dead-end, which is an important factor affecting $\bar{T}_r$. Compared with the point-to-point planner used by the other three algorithms, Pac-AUV repeatedly covers more target points to get out of a dead-end with lower computational complexity than its three peers. As a result, it has the smallest running time in unbounded environments. DH-CPP's running time is similar to that of DPPCPP and BoB, and larger than Pac-AUV's. However, DH-CPP can be applied to environments with dynamic obstacles while Pac-AUV and BoB cannot. In environments with dynamic obstacles, it enables a robot to find and cover the missing areas faster than the compared algorithm. Considering that a real robot's moving time is much longer than the algorithm's running time, DH-CPP, which requires fewer time steps to cover all target points, has better performance than all of its compared peers.

In unbounded environments, the increased size of the movable area makes robots less likely to fall into a dead-end. Pac-AUV and BoB plan coverage paths through boustrophedon motion, which has a smaller running time than DH-CPP and DPPCPP that require the calculation of a reward function for each candidate target point. Pac-AUV needs to assign missions to robots, while the other three algorithms do not. In a bounded environment, it only needs to perform mission allocation once at the beginning of the task. But in an unbounded environment, it needs to reallocate missions when the movable area expands, thus increasing its running time significantly. Therefore, BoB has the smallest runtime in unbounded environments. DH-CPP's running time is similar to that of DPPCPP and Pac-AUV, and larger than BoB's. However, among the four algorithms, DH-CPP is the only one that meets the requirements of the CPP problem in an unbounded environment. The other three algorithms all have a large number of uncovered target points near robots' starting positions and have missing points within their coverage range.

TABLE S-III
SIMULATION RESULTS FOR BOUNDED ENVIRONMENTS WITH STATIONARY OBSTACLES

Scenario		DH-CPP	DPPCPP	Pac-AUV	BoB
1	N_t	301	**300**	309	315
	$\bar{T}_r$	0.17	0.12	**0.03**	0.11
2	N_t	284	**275**	288	303
	$\bar{T}_r$	0.13	0.1	**0.01**	0.15
3	N_t	**272**	275	349	320
	$\bar{T}_r$	0.12	0.12	**0.02**	0.35
4	N_t	**254**	**254**	337	287
	$\bar{T}_r$	0.11	0.09	**0.01**	0.08

TABLE S-IV
SIMULATION RESULTS FOR BOUNDED ENVIRONMENTS WITH DYNAMIC OBSTACLES

Scenario		DH-CPP	DPPCPP
5	N_t	**301**	**301**
	$\bar{T}_r$	0.13	**0.09**
6	N_t	**312**	317
	$\bar{T}_r$	0.14	**0.12**
7	N_t	**315**	326
	$\bar{T}_r$	**0.13**	0.18

TABLE S-V
SIMULATION RESULTS FOR UNBOUNDED ENVIRONMENTS WITH STATIONARY OBSTACLES

Scenario		DH-CPP	DPPCPP	Pac-AUV	BoB
8	D_a	**12.51**	20.11	18.86	12.89
	N_m	**0**	**0**	**0**	**0**
	N_c	**900**	**900**	889	890
	$\bar{T}_r$	0.16	0.1	0.17	**0.02**
9	D_a	**12.77**	18.15	17.66	13.61
	N_m	**0**	**0**	**0**	**0**
	N_c	**900**	**900**	879	883
	$\bar{T}_r$	0.14	0.09	0.13	**0.01**
10	D_a	**11.9**	20.19	17.49	14.91
	N_m	**0**	**0**	**0**	25
	N_c	894	**900**	880	885
	$\bar{T}_r$	0.17	0.07	0.12	**0.01**
11	D_a	**10.2**	20.67	16.96	13.76
	N_m	**0**	**0**	7	**0**
	N_c	894	**897**	861	860
	$\bar{T}_r$	0.13	0.08	0.14	**0.02**

TABLE S-VI
SIMULATION RESULTS FOR UNBOUNDED ENVIRONMENTS WITH DYNAMIC OBSTACLES

Scenario		DH-CPP	DPPCPP
12	D_a	**11.42**	17.5
	N_m	**0**	**0**
	N_c	**900**	**900**
	$\bar{T}_r$	0.13	**0.08**
13	D_a	**11.59**	18.75
	N_m	**0**	4
	N_c	884	**900**
	$\bar{T}_r$	0.13	**0.08**
14	D_a	**11.58**	18.27
	N_m	**0**	4
	N_c	884	**900**
	$\bar{T}_r$	0.11	**0.08**

C. Comparison Experiments with Different Numbers of Robots

In this section, DH-CPP's performance in dual environments with 2-5 robots is evaluated. Four scenarios of bounded and unbounded environments with stationary and dynamic obstacles are considered, i.e., Scenarios 4, 7, 11 and 14.

First, the performance of the proposed DH-CPP in bounded environments, i.e., Scenarios 4 and 7, with 2-5 robots is evaluated. The simulation results are presented in Table S-VII. $N_r \in \{2.3, 4, 5\}$ is the number of robots. In a bounded environment with stationary obstacles (Scenario 4), as N_r increases, N_t of the four algorithms decreases. DH-CPP is more affected by obstacles and other robots than DPPCPP, so it ranks the 2nd and DPPCPP attains the best performance. But the extra time steps DH-CPP requires compared to DPPCPP are only 2.2% fewer. In all four experiments with different N_r, N_t of Pac-AUV and BoB is much higher than that of DH-CPP and DPPCPP, indicating that Pac-AUV and BoB require more time steps to cover all target points. In addition, Pac-AUV needs to assign missions to robots, while the other three algorithms do not. Under the interference of obstacles, the robot may not be able to reach its assigned target point. Such a case happens in Scenario 4 with four robots, resulting in its failure to cover all target points. In a bounded environment with stationary obstacles and different numbers of robots, the proposed DH-CPP attains a competitive performance.

In a bounded environment with dynamic obstacles (Scenario 7), Pac-AUV and BoB plan coverage paths through boustrophedon motion and cannot handle dynamic obstacles. Their experimental results are marked as '\' in Table S-VII. DPPCPP has the difficulty of dealing with suddenly moving dynamic obstacles within the coverage range. In

contrast, by employing the missing reward and considering target points between the destination target point and the virtual herd, DH-CPP enables the robot to find the missing area in time and move to cover it autonomously, and thus its N_t is smaller. As N_r increases, N_t of DH-CPP and DPPCPP decreases. In Scenario 7 with five robots, robots have covered all obstacle-free target points when obstacles 3-5 start to move. At this time, there is no significant difference in the speed of the two algorithms to reach the missing area. Hence, they get a close N_t. In a bounded environment with dynamic obstacles and different N_r, the proposed DH-CPP enables robots to find and cover the missing areas faster than the compared algorithm and attains the best performance.

Next, DH-CPP's performance in unbounded environments, i.e., Scenarios 11 and 14, with 2-5 robots, is evaluated. The simulation results are presented in Table S-VIII. In an unbounded environment with stationary obstacles (Scenario 11), the proposed DH-CPP attains the best performance. DPPCPP and BoB's D_a are much higher than that of DH-CPP. Pac-AUV's D_a is much higher than DH-CPP's in experiments with 2 and 3 robots. This indicates that DPPCPP, Pac-AUV and BoB have a large number of uncovered target points near robots' starting positions. In experiments with 4 and 5 robots, DH-CPP and Pac-AUV get a close D_a. However, this is due to a large number of missing points within Pac-AUV's coverage range. In all four experiments, the number of missing points (N_m) of the proposed DH-CPP is 0. This demonstrates that it has the ability to plan a coverage path without missing points. In contrast, both Pac-AUV and BoB have missing points. In addition, as N_r increases, the number of covered target points (N_c) of the four algorithms increases. But the interference of other robots leads to more repeated coverage. DH-CPP covers from the center of robots' starting positions. Hence, this interference has the least impact on it among all compared algorithms. Therefore, we can observe that it obtains the largest N_c in experiments with 4 and 5 robots. In an unbounded environment with stationary obstacles and different numbers of robots, among the four algorithms, DH-CPP is the only one that well solves the CPP problem in an unbounded environment. Its three peers all have a large number of uncovered target points near robots' starting positions. It is less affected by other robots than its peers and performs better in unbounded environments with more robots.

In an unbounded environment with dynamic obstacles (Scenario 14), Pac-AUV and BoB cannot handle dynamic obstacles. DPPCPP's D_a is much higher than DH-CPP's D_a in all four experiments, indicating that it does not well solve the CPP problem in an unbounded environment. DPPCPP has missing points in experiments with 3, 4 and 5 robots. It has the difficulty of dealing with suddenly moving dynamic obstacles within the coverage range. In contrast, DH-CPP enables the robot to quickly find and cover the missing area, and $N_m = 0$ in all experiments. As N_r increases, N_c increases. The interference of other robots has less impact on DH-CPP than DPPCPP, so DH-CPP obtains a higher N_c than DPPCPP in the experiment with 5 robots. In an unbounded environment with dynamic obstacles and different N_r, the proposed DH-CPP enables the robot to find the missing area in time and move to cover it autonomously and attains the best performance among compared algorithms.

TABLE S-VII
SIMULATION RESULTS VS. N_r IN BOUNDED ENVIRONMENTS

Scenario	N_r		DH-CPP	DPPCPP	Pac-AUV	BoB
4	2	N_t	378	**370**	412	399
		$\bar{T}_r$	0.16	0.13	**0.02**	0.06
	3	N_t	**254**	**254**	337	287
		$\bar{T}_r$	0.11	0.09	**0.01**	0.08
	4	N_t	190	**186**	\	232
		$\bar{T}_r$	0.09	**0.07**	\	0.09
	5	N_t	153	**150**	321	172
		$\bar{T}_r$	0.1	0.07	**0.01**	0.08
7	2	N_t	**462**	477	\	\
		$\bar{T}_r$	0.17	**0.15**	\	\
	3	N_t	**315**	326	\	\
		$\bar{T}_r$	**0.13**	0.18	\	\
	4	N_t	**235**	247	\	\
		$\bar{T}_r$	0.12	**0.1**	\	\
	5	N_t	**196**	197	\	\
		$\bar{T}_r$	0.13	**0.12**	\	\

D. Experiments with Different Environmental Parameter Settings

DH-CPP's performance in unbounded environments with an initial movable area size of 25×25, 40×40 and 55×55, and boundary expansion rate of 1-3 units is evaluated. Two scenarios with stationary and dynamic obstacles are considered, i.e., Scenarios 11 and 14. The simulation results of different initial movable area sizes and different boundary expansion rates are presented in Tables S-IX and S-X, respectively. Pac-AUV and BoB cannot handle dynamic obstacles. S_m is the initial movable area size, and its value is 25×25, 40×40 and 55×55. R_e is the boundary expansion rate, and its value is 1-3 units.

DH-CPP can be applied to an unbounded environment without artificially setting boundaries. The artificially set boundaries only restrict its movement. Therefore, in Scenario 11 with S_m=25×25, DH-CPP's movement is impeded by the boundary, resulting in covering fewer target points than it does at a size of 40×40 and 50×50. But in other experiments with different S_m and R_e, DH-CPP's coverage path does not reach artificially set boundaries, so its performance remains stable and is the best among the four algorithms. Its three peers have a strong dependence on the boundary and can be applied to unbounded environments by artificially setting boundaries. Therefore, they are more badly affected by the initial movable area size and boundary expansion rate than DH-CPP. As S_m and R_e increase, DPPCPP, Pac-AUV and BoB seldomly pass through the area where obstacles are located in Scenarios with

large S_m and R_e. This reduces their number of missing points and repeated coverage of target points, but results in a significant increase in D_a, indicating that more target points near robots' starting positions are not covered. In all four experiments, N_m of DH-CPP is 0 while its peers all have missing points. When the initial movable area size is small, DPPCPP and Pac-AUV, which cover from the boundary, are the most affected among the four algorithms. Their movement is severely limited, resulting in less N_c in Scenarios with S_m=25$\times$25. In experiments with different environmental parameter settings, the proposed DH-CPP is the most stable and the best among the four algorithms.

TABLE S-VIII
SIMULATION RESULTS VS. N_r IN UNBOUNDED ENVIRONMENTS

Scenario	N_r		DH-CPP	DPPCPP	Pac-AUV	BoB
11	2	D_a	**10.87**	21.24	18.89	13.13
		N_m	**0**	**0**	3	5
		N_c	593	**600**	569	591
		$\bar{T}_r$	0.15	0.04	0.17	**0.01**
	3	D_a	**10.2**	20.67	16.96	13.76
		N_m	**0**	**0**	7	**0**
		N_c	894	**897**	861	860
		$\bar{T}_r$	0.13	0.08	0.14	**0.02**
	4	D_a	15.13	17.33	**15.09**	17.59
		N_m	**0**	**0**	126	**0**
		N_c	**1189**	1183	1160	1152
		$\bar{T}_r$	0.15	0.1	0.1	**0.02**
	5	D_a	15.64	17.25	**15.01**	18.87
		N_m	**0**	**0**	157	**0**
		N_c	**1473**	1463	1363	1415
		$\bar{T}_r$	0.17	0.12	0.09	**0.02**
14	2	D_a	**10.85**	20.55	\	\
		N_m	**0**	**0**	\	\
		N_c	593	**600**	\	\
		$\bar{T}_r$	0.1	**0.05**	\	\
	3	D_a	**11.58**	18.27	\	\
		N_m	**0**	4	\	\
		N_c	884	**900**	\	\
		$\bar{T}_r$	0.11	**0.08**	\	\
	4	D_a	**13.33**	19.04	\	\
		N_m	**0**	4	\	\
		N_c	1174	**1189**	\	\
		$\bar{T}_r$	0.14	**0.09**	\	\
	5	D_a	**14.37**	16.56	\	\
		N_m	**0**	4	\	\
		N_c	**1463**	1455	\	\
		$\bar{T}_r$	0.16	**0.14**	\	\

TABLE S-IX
SIMULATION RESULTS OF DIFFERENT INITIAL MOVABLE AREA SIZES

Scenario	S_m		DH-CPP	DPPCPP	Pac-AUV	BoB
11	25×25	D_a	9.61	10.41	**9.23**	10.79
		N_m	**0**	**0**	11	3
		N_c	**874**	843	737	868
		$\bar{T}_r$	0.12	0.11	0.06	**0.01**
	40×40	D_a	**10.2**	20.67	16.96	13.76
		N_m	**0**	**0**	7	**0**
		N_c	894	**897**	861	860
		$\bar{T}_r$	0.13	0.08	0.14	**0.02**
	55×55	D_a	**10.2**	27.91	24.81	16.57
		N_m	**0**	**0**	**0**	**0**
		N_c	894	**900**	897	878
		$\bar{T}_r$	0.13	0.07	0.24	**0.01**
14	25×25	D_a	12.19	**10.88**	\	\
		N_m	**0**	4	\	\
		N_c	**884**	844	\	\
		$\bar{T}_r$	0.11	**0.09**	\	\
	40×40	D_a	**11.58**	18.27	\	\
		N_m	**0**	4	\	\
		N_c	884	**900**	\	\
		$\bar{T}_r$	0.11	**0.08**	\	\
	55×55	D_a	**11.58**	29.75	\	\
		N_m	**0**	**0**	\	\
		N_c	884	**900**	\	\
		$\bar{T}_r$	0.11	**0.07**	\	\

TABLE S-X
SIMULATION RESULTS OF DIFFERENT BOUNDARY EXPANSION RATES

Scenario	R_e		DH-CPP	DPPCPP	Pac-AUV	BoB
11	1	D_a	**10.2**	20.67	16.96	13.76
		N_m	**0**	**0**	7	**0**
		N_c	894	**897**	861	860
		$\bar{T}_r$	0.13	0.08	0.14	**0.02**
	2	D_a	**10.2**	22.42	19.73	14.06
		N_m	**0**	**0**	16	**0**
		N_c	894	**900**	884	862
		$\bar{T}_r$	0.13	0.09	0.17	**0.01**
	3	D_a	**10.2**	25.08	21.61	15.8
		N_m	**0**	**0**	**0**	**0**
		N_c	894	**900**	883	885
		$\bar{T}_r$	0.14	0.09	0.21	**0.01**
14	1	D_a	**11.58**	18.27	\	\
		N_m	**0**	4	\	\
		N_c	884	**900**	\	\
		$\bar{T}_r$	0.11	**0.08**	\	\
	2	D_a	**11.58**	22.88	\	\
		N_m	**0**	4	\	\
		N_c	884	**900**	\	\
		$\bar{T}_r$	0.11	**0.08**	\	\
	3	D_a	**11.58**	25.61	\	\
		N_m	**0**	**0**	\	\
		N_c	884	**900**	\	\
		$\bar{T}_r$	0.11	**0.09**	\	\