Decentralized Multi-Agent Coverage Path Planning with Greedy Entropy Maximization

Kale Champagnie, Farshad Arvin, and Junyan Hu

***Abstract*—In this paper, we present GEM, a novel approach to online coverage path planning in which a swarm of homogeneous agents act to maximize the entropy of pheromone deposited within their environment. We show that entropy maximization (EM) coincides with many conventional goals in offline coverage path planning, while also generalizing to online settings. We first propose the concept of uniformity, which is a generalised metric that allows offline and online CPP approaches to be viewed through a unified lens. We then evaluate our approach by measuring the rate at which entropy is maximized within a variety of static and dynamic environments. Our experimental results demonstrate that GEM achieves state-of-the-art performance in online coverage, competitive with offline methods, despite requiring no direct communication among agents.**

***Index Terms*—Coverage path planning, multi-agent systems, swarm intelligence, autonomous robots, artificial pheromone.**

I. Introduction

Coverage Path Planning (CPP) refers to the task of directing one or more mobile agents such that they collectively explore an area of interest (AOI), while also avoiding obstacles [1]. Generally, CPP involves two interrelated processes — viewpoint generation and path generation. Viewpoint generation involves specifying a set of regions within the target area (viewpoints), such that their union encompasses the whole area. CPP has been applied to numerous fields, including search and rescue operations [2]–[5], household robotics [6], connected and autonomous vehicle platooning [7], agricultural monitoring [6], exploration of hazardous environments (e.g. due to unsafe levels of radiation) [8], robot and bee colony interactions [9], aerial surveillance [10], and broader applications in mathematics related to efficient coverage of discrete and continuous state-spaces.

Offline Coverage Path Planning is a traditional form of coverage path planning which generates viewpoints and path plans prior to executing those plans within the environment [11], [12]. Offline CPP approaches have maintained persistent interest in the literature due a number of valuable properties; for instance, their ability to produce near-optimal solutions in static environments [13]. A primary disadvantage of offline approaches is their reliance on rather unrealistic assumptions about the environment agents will be embedded in. However, many scenarios where CPP could be applied, do not necessarily fit this model. A related problem arising from offline approaches is their reliance on complete a priori knowledge about the environment. However, many realistic scenarios may involve environments which are only partially observable or indeed completely unmapped. Besides, many offline approaches are also centralized, meaning agents receive plans from a single coordination unit, which may have limited network reachability [14]. To overcome these limitations, online CPP methods have been developed in conjunction with conventional approaches to address many of the issues outlined above [15]. Their defining feature is the construction of plans at execution time (either partially or entirely), often in a dynamic fashion that adapts to unanticipated environmental changes [16]. Some of the most successful approaches have utilized or replicated concepts from biological systems such as stigmergic communication (i.e. pheromonal signaling) and other decentralized swarm-like mechanics to achieve real-time coverage without an explicit planning algorithm [1]. However, there are still significant areas for improvement within contemporary approaches. For instance, whereas offline planning algorithms have a clear objective (cover every viewpoint in an efficient manner), there has been less consensus among researchers about which objective goals online methods should strive for.

In this work, we focus on developing a novel CPP approach that takes inspiration from the collective behaviour of ants to explore arbitrary, highly dynamic environments in an efficient, distributed manner. Unlike many existing approaches, our method is entirely decentralized and requires no direct communication between agents, instead relying on indirect pheromonal signalling. We additionally propose a generalised framework for evaluating the performance of CPP solutions based on the notion that *successful coverage*, in most cases reduces to *uniform exploration* of the environment, such that each viewpoint receives a similar number of visits from the team of agents.

II. Uniformity Design

The primary aim of coverage path planning is to efficiently visit every viewpoint within a target area. In static environments, we may consider coverage to be complete once every viewpoint has been visited at least once by an agent. We may then measure the efficiency of the solution with respect to how quickly it completes and in terms of other metrics such as the overlap (redundancy) between generated paths. However, in dynamic environments, it may not be sufficient to visit each viewpoint only once. Additionally, online CPP methods should be capable of adapting to environmental changes which create opportunities to explore previously unreachable regions of the AOI.

This work was supported by EU H2020-FET-OPEN RoboRoyale project [grant number 964492].

K. Champagnie is with the Department of Computer Science, University College London, London, UK. (e-mail: kale.champagnie.20@ucl.ac.uk)

F. Arvin and J. Hu are with the Department of Computer Science, Durham University, Durham, UK. (e-mail: {farshad.arvin, junyan.hu}@durham.ac.uk)

In this section, we introduce a generalized coverage path planning metric and objective function, which allows us to view both offline and online CPP approaches through the same lens. The metric in question is referred to as *uniformity*, which aims to capture the notion of spatially and temporally balanced visitation to each viewpoint within the AOI. It quantifies the extent to which agents effectively distributed their shared resources into covering the AOI as rapidly as possible and maintaining said coverage after perturbations to the environment (e.g. the removal of obstacles, or indeed agents themselves).

A. Uniformity

We refer to this metric as uniformity as it essentially quantifies how evenly spread visits to viewpoints are over the AOI. In this section, we provide a precise definition of uniformity and show that maximizing uniformity implies optimizing for the offline and online metrics. We first define a measure for the uniformity of an environment at time t. We then, use this definition to define the uniformity of a coverage path plan, as the cumulative uniformity achieved throughout the duration of the plan.

We define the state of an environment at time t as a 3-tuple $S_t = (V_t, E_t, A_t)$ where:

- V_t is the set of viewpoints at time t.
- E_t is the set of edges (i.e. accessible paths) between viewpoints at time t.
- $A_t \subseteq V_t$ is the subset of viewpoints occupied by agents at time t.

To quantify the uniformity of such an environment, we equip each viewpoint with a pheromone level which decays over time. We may also call this the temperature or energy of a viewpoint. Whenever an agent visits the viewpoint, the pheromone level is replenished and increased by a constant amount. Otherwise, the pheromone level is depleted and reduced by a constant factor (i.e. exponential decay). The rate at which pheromone decays is an important parameter which essentially reflects how many frequently visits must be made to a particular viewpoint.

Let $v \in V$ be a viewpoint. Then its pheromone level at time t is given by

$$L_t(v) = \alpha L_{t-1}(v) + k 1_{A_t}(v), \quad L_0(v) = 0 \tag{1}$$

where:

- α is the decay rate coefficient — for instance $\alpha = 0.98$ causes the pheromone to decay by 2% on every iteration.
- k is the amount of pheromone deposited by agents when visiting the viewpoint, typically $k = 1$.
- $1_{A_i}(v)$ is an indicator function, returning one if $v \in A_i$ or zero otherwise.

Using this recurrence relation, we can obtain a pheromone distribution for the environment at time t. This is a probability distribution encoding the relative pheromone levels of each viewpoint. We may compute it by simply normalizing over all viewpoints in V:

$$p_t(v) = \frac{L_t(v)}{\sum_{u \in V} L_t(u)}$$

Now, we would like to quantify the uniformity of this distribution. A semantically reasonable choice is to compute the Shannon entropy of a hypothetical random variable distributed according to p_t. That is,

$$\mathrm{H}[p_t] = \mathrm{H}[X], \quad X \sim p_t$$

According to information theory, $\mathrm{H}[X]$ may be interpreted as minimum number of bits required to communicate a particular outcome in X. However, for our purposes we needn't make use of this interpretation. Instead, we can simply see $\mathrm{H}[X]$ as measuring how evenly spread pheromones are among the set of viewpoints. If most of the pheromone is concentrated in only a few viewpoints, then the entropy will be low. As we will show, this occurs precisely when conventional CPP objectives such as low overlap fail to be achieved.

Hence, the uniformity of the environment at time t, may be concisely written as

$$U(S_t) = \mathrm{H}[p_t] = -\sum_{v \in V_t} p_t(v) \log_2 p_t(v). \tag{2}$$

For completeness, we also define the uniformity of an entire coverage path plan A, as the cumulative uniformity achieved at each time step.

$$U(A) = \sum_{t \leq |A|} U(S_t). \tag{3}$$

B. Generality

Throughout this section, we have claimed that uniformity generalizes both offline and online coverage metrics. To support the variety of our claim, we now devise such a proof using the formal definitions of uniformity and conventional CPP metrics. We show that in each case, maximizing uniformity entails optimizing for the metric in question.

1) Increasing Uniformity Eventually Implies Increasing Coverage Completeness: We show that if $U(S_{t'}) - U(S_t) > c$ then $\mathrm{CC}_{t'} > \mathrm{CC}_t$.

In other words, uniformity cannot be increased beyond a certain amount without also increasing coverage completeness.

Suppose that at time t, we have $|C_t| = k$. Then only k viewpoints have ever been visited. A viewpoint u which has never been visited has no pheromone since $L_0(u) = 0$ and there exists no time $t' \leq t$ such that $1_{A_{t'}}(v) = 1$.

Then, all pheromones must be entirely concentrated on the k visited viewpoints. The minimal uniformity occurs when all pheromone is concentrated on a single viewpoint u, such that

$$p_t(v) = \begin{cases} 1 & \text{if } v = u \\ 0 & \text{otherwise} \end{cases}. \tag{4}$$

Then entropy of this distribution is exactly zero and there is no uncertainty about where the pheromone is located.

Now suppose that at time $t' > t$, we again have $|C_{t'}| \leq k$. The maximal uniformity occurs when $p_t(v) = 1/k$ for all $v \in V_t$. In this case the entropy is exactly $\log_2 k$. Hence, the maximum increase in uniformity between t and t' is

$$\max U(S_{t'}) - \min U(S_t) = \log_2 k - 0 = \log_2 k. \quad (5)$$

Hence, if $U(S_{t'}) - U(S_t) > \log_2 k$, the assumption that $|C_{t'}| \leq k$ is false, and $|C_{t'}| > |C_t|$. Therefore, $U(S_{t'}) - U(S_t) > c$ implies $\text{CC}_{t'} > \text{CC}_t$ where $c = \log_2 \text{CC}_t$.

2) Maximizing Uniformity Implies Maximizing Online Coverage Completeness: We show that if uniformity is maximal, online coverage completeness must be maximal.

Let l be the lifespan such that if $C_t(v) \iff \text{VP}_t(v) < l$. We can always choose a decay rate coefficient α such that $p_t(v) = 0$ if and only if $\text{VP}_t(v) \geq l$. In other words, we can choose a decay rate which assigns probabilities greater than 0 to viewpoints only if they were visited within the lifespan. Essentially we are encoding l in α.

Maximum uniformity is achieved when $p_t(v) = 1/|V_t|$ for all $v \in V_t$. Then for all v we have that $p_t(v) > 0$ which implies $\text{VP}_t(v) < l \implies C_t(v)$. Hence, when uniformity is maximized under α, every viewpoint is covered under the lifespan l, thus online coverage completeness is also maximized.

C. Derivative Metrics

Now that we have justified the notion of uniformity on intuitive and formal grounds, we introduce several derivative metrics which may expressed in terms of uniformity.

1) Perplexity (PPL): Uniformity is measured logarithmically in bits (or nats if we choose to use the natural logarithm). However, it is more natural for us to work with linear scales when comparing and evaluating coverage algorithms. The most straightforward method for linearising entropy is to work with perplexity, which simply inverts the logarithm

$$\text{PPL}(S_t) = 2^{U(S_t)}. \quad (6)$$

Throughout the remainder of this paper, we will generally work with PPL rather than U directly when presenting results.

2) Relative Perplexity (RPPL): A further transformation we may apply to U, is to compute perplexity, relative to the maximum possible perplexity achievable within an environment. Maximum perplexity is achieved when $p_t(v) = 1/|V_t|$ for all $v \in V_t$ and has a value of $2^{|V_t|}$. Hence, relative perplexity (RP) is given by

$$\text{RPPL}(S_t) = \frac{2^{U(S_t)}}{2^{|V_t|}} = 2^{U(S_t) - |V_t|}. \quad (7)$$

When $\text{RPPL} = 1$, we have ideal coverage. Values less than 50% are generally considered unsatisfactory. Note that the RPPL achieved by an algorithm is dependant on the decay rate coefficient α. Higher values of α (lower decay rate) make achieving high RPPL easier, whereas lower values represent resource-intensive coverage problems where each viewpoint requires more frequent attention.

III. Greedy Entropy Maximization Design for Multi-Agent Coverage

In this section, we present a concrete uniformity-based coverage path planning algorithm which takes inspiration from the collective behavior of ants. Our approach is referred to as GEM (Greedy Entropy Maximization) and aims to maximize uniformity (entropy) using a swarm of homogeneous agents that make decisions greedily (i.e. only using local information). GEM makes almost no a priori assumptions about the environment (apart from the assumption that it's discrete). GEM can also work with an arbitrary number of agents and yields a time complexity which is invariant to map size. In addition, agents require no direct communication as the algorithm instead relies on implicit/indirect stigmergic communication via pheromones.

A. Problem Formulation

The primary goal of GEM is to maximize the uniformity of the environment $U(S_t)$, and more generally the cumulative uniformity obtained over the entire course of a run, denoted $U(A)$. Let $\{(S_0, R_0), \ldots, (S_t, R_t), \ldots\}$ be a Markov-decision process where $S_t = (V_t, E_t, A_t)$ is the state of the environment at time t and R_t is the immediate reward obtained from transitioning into state S_t from S_{t-1}. The agent (which in our case refers to the entire swarm of mobile agents), emits a corresponding sequence of actions $\{a_0, \ldots, a_t, \ldots\}$ which cause the environment to emit a new state and reward according to the conditional distribution $p\left(R_{t+1}, S_{t+1} | S_t, a_t\right)$.

1) Actions: The swarm emits a sequence of actions $\{a_0, \ldots, a_t, \ldots\}$ describing the movements of each agent at every time step. For simplicity, we can let $a_t = A_t$, such that each action just lists the viewpoints each agent should occupy. Some actions are invalid in particular states; namely actions which would cause multiple agents to occupy the same viewpoint at the same time, or actions which place agents in viewpoints occupied by obstacles. For the sake of generality, we avoid defining precisely how obstacles are represented, but define an inaccessible viewpoint as one which has no inbound edges. We say $v \in V_t$ is inaccessible if

$$(\forall u \in V_t)((u, v) \notin E_t). \quad (8)$$

In contrast, an accessible viewpoint is one with at least one inbound edge:

$$(\exists u \in V_t)((u, v) \in E_t). \quad (9)$$

In GEM, agents may only move between adjacent viewpoint such that

$$(\forall v \in A_t)(\exists u \in O_{t-1})((u, v) \in E_{t-1}), \quad (10)$$

i.e., every viewpoint in A_t must be reachable from a viewpoint in A_{t-1} (the prior locations of each agent), along an edge in E_{t-1}.

2) States: The state of the environment is essentially a graph with vertices representing viewpoints and edges representing accessibility between those viewpoints. For instance, in grid-based CPP, each grid cell may be assigned a particular viewpoint, with edges between call geometrically adjacent cells and itself (reflexivity).

We generally assume that V_t remains fixed such that $\forall t(V_t = V_0)$. In most environment models, there is no need to change V_t as all environmental perturbations can be represented through alterations to the set of edges E_t. For instance, again considering grid-based CPP, walls may be represented simply by the lack of an edge between two adjacent cells.

3) Rewards: Rewards are signals transmitted to an agent in addition to states, which signal whether a previous action was positive or negative. In the case of GEM, a natural choice for this reward signal is just the uniformity of the current state, i.e.,

$$R_t = U(S_t) \tag{11}$$

We could also use the change in uniformity between consecutive states; in RL solutions, this may be preferable as it provides a bounded reward signal with negative rewards helping to negatively reinforce actions that reduce coverage. However, for GEM, which is not based on RL, we can uniformity directly.

Then cumulative reward received by the agent is simply given by

$$R = \sum_{t<t_{\max}} R_t \tag{12}$$

The agent's goal is then to select a set of actions A such that

$$R = \max_{A \in \mathcal{A}} R \tag{13}$$

where $\mathcal{A}$ is the set of all possible action sequences the agent could take.

This completes our decision-theoretic formulation of online CPP. We now discuss how GEM provides an effective solution.

B. Greedy Entropy Maximization Algorithm

GEM is a completely decentralized online CPP algorithm in which a swarm of homogeneous agents act to maximize the entropy of pheromone deposited within the environment. Each agent is assumed to have minimal sensing such that it can only detect the state of viewpoints immediately adjacent to the viewpoint it's currently stationed on. Moreover, we make no assumptions about whether agents have memory nor whether they are capable of communicating with each other remotely.

In GEM, each agent within the swarm must have access to the pheromone level $L_t(v)$ deposited on every viewpoint adjacent to its own position. A decision heuristic function $h(\cdot)$ is then applied to determine which adjacent viewpoint the agent should move to. Formally,

$$A_t^i = \text{argmin}_{v \in \text{adj}(A_{t-1}^i)} h(L_t(v)) \tag{14}$$

where $\text{adj}(A_{t-1}^i)$ is the set of viewpoints adjacent to A_{t-1}^i.

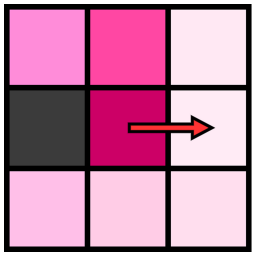


Fig. 1. Actions taken by a GEM agent following the least pheromone heuristic with pheromone levels depicted by colour intensity.

The simplest heuristic, and the one we use our implementation is just the identity function $h(x) = x$. In this case, each agent simply moves to whichever viewpoint has the least pheromone deposited on it via

$$A_t^i = \text{argmin}_{v \in \text{adj}(A_{t-1}^i)} L_t(v). \tag{15}$$

Figure 1 shows a hypothetical scenario in a grid-based CPP problem with an agent stationed at the centre cell. The intensity of each cell indicates the relative amount of pheromone deposited on it. Under the identity heuristic, the agent chooses to move to the adjacent cell with least pheromone, indicated by the arrow. An obstacle is shown on the left in grey.

Once an agent moves to a new viewpoint, it deposits fresh pheromone to it, which increases the pheromone level. Pheromone levels decay over time according to

$$L_t(v) = \alpha L_{t-1}(v) + k1_{A_t}(v), , \quad L_0(v) = 0 \tag{16}$$

where α is the decay rate coefficient and k is the amount of new pheromone deposited.

Intuitively, we might expect this heuristic to perform moderately since it actively peruses viewpoints that haven't been visited recently. However, quite surprisingly, we find it performs exceptionally competing with offline CPP algorithms such as DARP.

The GEM algorithm is very simple and as such requires very little code to implement. We provide pseudo-code for the algorithm below.

Algorithm 1 GEM

```
for t ∈ {0, ..., t_max} do
    for A_t^i ∈ A_t do
        A_{t+1}^i = argmin_{v ∈ adj(A_{t-1}^i)} L_t(v)
    end for
    for v ∈ V_t do
        L_{t+1}(v) = αL_t(v) + k1_{A_t}(v)
    end for
end for
```

At every time-step, each agent moves to whichever viewpoint has the least pheromone deposited on it. Every viewpoint's pheromone level is then updated according to exponential decay and whether an agent is current stationed on it.

C. Complexity Analysis

A primary advantage of GEM over several existing methods is that it's computational complexity is scales linearly or constantly with respect to important parameters such as the number of agents, or number of viewpoints within the environment.

As discussed, agents are entirely independent of each other and do not rely on each other to make decisions. As such, we first consider the time and space complexity of GEM, when running as a concurrent algorithm, i.e. where agents run in parallel. In this case, at every time step, each agent simply needs to observe the pheromone present on every viewpoint adjacent to its own. This operation is $O(n)$ in the number of adjacent viewpoints. In many cases, such as in grid-based CPP problems, this factor is constant, so the operation takes $O(1)$ time. Identifying the viewpoint with least pheromone is consequently also $O(n)$ or $O(1)$.

Similarly, pheromone updates can be performed in a concurrent fashion as $L_t(v)$ depends only on $L_{t-1}(v)$. In settings, where pheromones have a physical implementation, this operation make take $O(1)$ time. In other environments, such as simulations, the operation nonetheless be performed efficiently using various concurrency techniques, such as GPU programming.

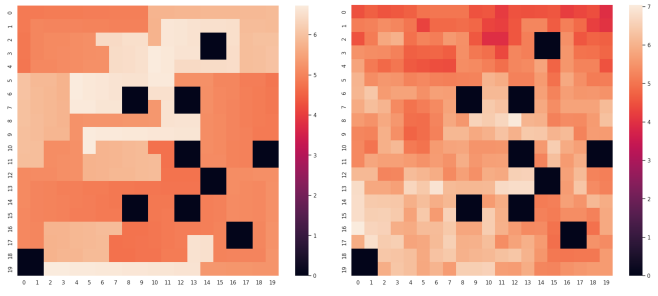
Overall, in physical environments, the algorithm can have complexities as low $O(1)$ in both the number of agents, and the number of viewpoints. In simulated environments, the algorithm's time complexity is $O(n)$ in the number of agents and $O(n)$ in the number of viewpoints. However, using concurrency techniques, the concrete time required to update pheromone levels for each viewpoint may be reduced considerably.

IV. Experiments

A. Experimental Setup

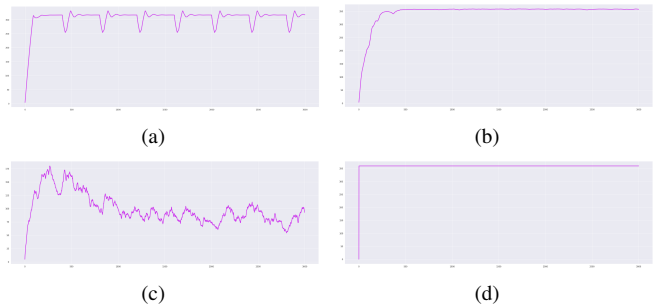
Our experiments aim to ascertain how well GEM performs under a range of environmental conditions, including static and dynamic environments. In our case, we use a variant of DARP known as Replannable DARP (R-DARP) [13]. R-DARP essentially recomputes the DARP-optimal solution whenever the environment changes. R-DARP achieves very strong performance in dynamic environments, but is often infeasible to implement in practice because it requires complete knowledge of the environment state at all times. If GEM can compete with or surpass R-DARP, then we may deem GEM as highly effective.

In our experiments, we simulate random obstacle environments, i.e., static and dynamic environments where each viewpoint is occupied with probability p. In dynamic cases, the environment is re-generated every so often (e.g. every 10 time-steps). In static cases, it's generated only once. This category of environment allows us to gauge how well GEM performs in arbitrary AOIs.



(a) (b)

Fig. 2. (a) R-DARP pheromone distribution after 3000 iterations. RPPL = 331/360 = 91% and (b) GEM pheromone distribution after 3000 iterations. RPPL = 358/360 = 99%



(a) (b) (c) (d)

Fig. 3. (a) R-DARP PPL over time; $\max = 331$, $\mu = 306$; (b) GEM PPL over time; $\max = 358$, $\mu = 343$; (c) Random Walker PPL over time; $\max = 180$, $\mu = 99$; (d) Optimal PPL over time; $\max = 360$, $\mu = 360$.

B. Experimental Results

1) Qualitative Analysis: Random obstacle environments are the simplest form of environment commonly used by authors to evaluate CPP algorithms. In such environments, the accessibility of each viewpoint has a Bernoulli distribution, typically uniform across the whole AOI. The probability p that a viewpoint is occupied may be referred to as the obstacle density of the environment. In our experiments, we investigate how well GEM performs in environments with densities between 0.1 and 0.9. Note that when measuring relative perplexity (RPPL), the presence of more obstacles is automatically taken into account. RPPL is measured in terms of accessible viewpoints only.

A 20x20 random obstacle environment with $p = 0.98$ is shown below Fig. 2(a) and 2(b) after 3000 iterations of R-DARP and GEM respectively. While each algorithm produces significantly different paths, we find that they both achieve strong RPPL scores of 91% and 99% respectively. R-DARP produces pheromone trails which are highly uniform between contiguous viewpoints. This can be attributed to the fact that each viewpoint is only ever visited by a single agent (in static environments). However, this property also leads to some degree of self-overlap, which results in non-ideal uniformity. In contrast, GEM produces pheromone which is less uniform between contiguous viewpoints, but more evenly distributed overall, leading to a superior performance. The qualitative difference between these two distributions can be examined

TABLE I
n VS. μ WITH $k = 1000$, $p = 0.01$, $A = 20^2$, $\beta = 0.01$

	$n = 1$	$n = 5$	$n = 10$
GEM	**0.45**	**0.85**	**0.93**
R-DARP	0.41	0.73	0.88
RandomWalker	0.16	0.47	0.55

TABLE II
p VS. μ WITH $k = 1000$, $n = 5$, $A = 20^2$, $\beta = 0.01$

	$p = 0.0$	$p = 0.25$	$p = 0.5$
GEM	**0.94**	0.72	**0.20**
R-DARP	0.88	**0.74**	0.14
RandomWalker	0.16	0.47	0.07

further by plotting the PPL score achieved for each iteration between $t = 0$ and $t = 3000$, illustrated in Fig. 3(a) and Fig. 3(b), respectively. For comparison, we also render the same plots for a random walker and the theoretically optimal CPP algorithm, as shown in Fig. 3(c) and Fig. 3(d).

2) Quantitative Analysis: In this analysis, we are primarily interested in how uniformity-based metrics (i.e., PPL and RPPL) vary with respect to CPP parameters including:

- n — the number of agents used to conduct CPP
- p — the obstacle density of the environment
- β — the pheromone decay rate
- A — the number of viewpoints in the AOI (i.e. its area)

For a particular assignment of these parameters, we measure the following response variables:

- $m = \max R$ — the maximum RPPL achieved within k iterations.
- $\mu = \mathrm{E}[R]$ — the mean RPPL achieved within k iterations.
- $\sigma = \sqrt{\mathrm{V}[R]}$ — the standard deviation of RPPL achieved within k iterations.

We randomly generate 1000 CPP problems on a 20x20 AOI with a fixed obstacle density $p = 0.1$ and constant decay rate of $\beta = 0.01$. We vary the number of agents and observe how this affects the expected RPPL achieved within $k = 1000$ iterations, for GEM, R-DARP and RandomWalker. The results are shown in Table I.

To investigate the relationship of obstacle density and μ, we generate 1000 random CPP problems on a 20x20 AOI with a fixed number of agents $n = 5$ and a decay rate of $\beta = 0.01$. We vary the obstacle density between 0% and 50% and the results are illustrated in Table II.

As it can bee seen from all these results, GEM consistently performs effectively in arbitrary environments, often able to achieve a mean RPPL of above 90% when given a suitable number of agents for the target decay rate. We find that GEM performs competitively with R-DARP in terms of how many agents are required to reach a certain level of coverage. It generally surpasses R-DARP in terms of the stability of coverage throughout the duration of a run, leading to higher mean RPPL.

V. Conclusions

In this work, we proposed a decentralized coverage path planning strategy based on GEM. Furthermore, our experiments demonstrated the effectiveness of using stigmergy-based approaches for coverage path planning, and the potential of GEM as a practical solution for real-world applications. For future works, fault-tolerant control techniques [17] will be integrated with the proposed method to further improve the coverage performance.

References

[1] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[2] F. Rekabi-Bana, J. Hu, T. Krajník, and F. Arvin, "Unified robust path planning and optimal trajectory generation for efficient 3D area coverage of quadrotor UAVs," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[3] A. Nedjati, A. Nedjati, B. Vizvári, and J. Arkat, "Complete coverage path planning for a multi-uav response system in post-earthquake assessment," *Robotics*, vol. 5, no. 4, p. 26, 12 2016.

[4] B. Ai, M. Jia, H. Xu, J. Xu, Z. Wen, B. Li, and D. Zhang, "Coverage path planning for maritime search and rescue using reinforcement learning," *Ocean Engineering*, vol. 241, p. 110098, 2021.

[5] K. Wu, J. Hu, Z. Ding, and F. Arvin, "Finite-time fault-tolerant formation control for distributed multi-vehicle networks with bearing measurements," *IEEE Transactions on Automation Science and Engineering*, 2023.

[6] D. Noh, W. Lee, H.-R. Kim, I.-S. Cho, I.-B. Shim, and S. Baek, "Adaptive coverage path planning policy for a cleaning robot with deep reinforcement learning," in *2022 IEEE International Conference on Consumer Electronics (ICCE)*, 2022, pp. 1–6.

[7] S. Xie, J. Hu, Z. Ding, and F. Arvin, "Cooperative adaptive cruise control for connected autonomous vehicles using spring damping energy model," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 3, pp. 2974–2987, 2022.

[8] C. Denniston, T. R. Krogstad, S. Kemna, and G. S. Sukhatme, "Planning safe paths through hazardous environments," 2018.

[9] F. Rekabi-Bana, M. Stefanec, J. Ulrich *et al.*, "Mechatronic design for multi robots-insect swarms interactions," in *2023 IEEE International Conference on Mechatronics*, 2023, pp. 1–6.

[10] M. Torres, D. A. Pelta, J. L. Verdegay, and J. C. Torres, "Coverage path planning with unmanned aerial vehicles for 3D terrain reconstruction," *Expert Systems with Applications*, vol. 55, pp. 441–451, 2016.

[11] P. Abad-Manterola, I. A. D. Nesnas, and J. Burdick, "Motion planning on steep terrain for the tethered axel rover," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 4188–4195.

[12] O. Salzman and D. Halperin, "Optimal motion planning for a tethered robot: Efficient preprocessing for fast shortest paths queries," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 4161–4166.

[13] A. C. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos, "DARP: Divide areas algorithm for optimal multi-robot coverage path planning," *J. Intell. Robotics Syst.*, vol. 86, no. 3–4, p. 663–680, jun 2017.

[14] S. D. Han and J. Yu, "Ddm: Fast near-optimal multi-robot path planning using diversified-path and optimal sub-problem solution database heuristics," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1350–1357, 2020.

[15] Y. Hu, J. Geng, C. Wang, J. Keller, and S. Scherer, "Off-policy evaluation with online adaptation for robot exploration in challenging environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3780–3787, 2023.

[16] M. Hassan and D. Liu, "Ppcpp: A predator–prey-based approach to adaptive coverage path planning," *IEEE Transactions on Robotics*, vol. 36, no. 1, pp. 284–301, 2020.

[17] K. Wu, J. Hu, Z. Li, Z. Ding, and F. Arvin, "Distributed collision-free bearing coordination of multi-UAV systems with actuator faults and time delays," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

Durham University
Durham Research Online

Citation on deposit:

Champagnie, K., Arvin, F., & Hu, J. (in press). Decentralized Multi-Agent Coverage Path Planning with Greedy Entropy Maximization. In 2024 IEEE International Conference on Industrial Technology (ICIT) (1-6)

For final citation and metadata, visit Durham Research Online URL:
https://durham-repository.worktribe.com/output/2379771

Copyright statement: This accepted manuscript is licensed under the Creative Commons Attribution 4.0 licence.
https://creativecommons.org/licenses/by/4.0/