2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

GESCE: Graph-based Ergodic Search in Cluttered Environments

Burhanuddin Shirose[1], Adam Johnson[2], Bhaskar Vundurthy[3], Howie Choset[3], and Matthew Travers[3]

***Abstract*— In this paper, we present a novel motion planning algorithm that inherits the strengths of both optimization and search-based planners. Optimization-based planners use the gradient of an objective function to generate a desired path, whereas search-based planners operate on a graph capturing the salient topology of a robot's free space. A class of optimization-based planners leverages prior information, modeled as a probability distribution of target locations in an environment, to guide path generation. We embrace one specific measure, referred to as *ergodicity*, which encourages a robot to spend a proportion of its time, weighted by the distribution, where it is likely to find targets of interest. Methods that minimize ergodicity were not designed to handle obstacles in the environment, and augmented approaches that add "soft" constraints for obstacles to the cost function may still yield a path that collides with an obstacle. In this work, we present a hybrid approach that first generates a graph of the environment's free space, followed by searching the graph with ergodicity as a heuristic. Our approach not only restricts the search to the free space, thereby avoiding obstacles by design, but also generates trajectories with low ergodicity values. Extensive testing on 125 test scenarios with varying degrees of clutter, information distribution, and robot start locations illustrate the efficacy of our algorithm.**

***Index Terms*— Ergodic search, cluttered environments, graph-based search, path planning**

I. Introduction

Robotic systems have long been used in place of human operators for tasks such as data collection or search and rescue in disaster relief scenarios [1]. These tasks can often require the system to operate inside of complex environments such as warehouses or reactors. Naive approaches to perform a search in such spaces are slow and wasteful for robotic systems with practical limitations such as battery life [2]. Instead, the system should leverage prior information about the nature of the mission in order to maximize its effort in areas deemed the most informative [3] [4].

One such form of search which leverages prior information about the target locations is *ergodic search*. In [5], Matthew and Mezic introduce a metric, referred to as *ergodicity*, which quantifies the proportion of time a robot spends in a region with respect to the underlying information density. By optimizing the ergodicity measure, a planner can generate paths which maximize the time spent in regions which are most likely to contain the desired targets [6] [7] [8].

Although originally designed for obstacle-free environments, ergodic search can be modified to be leveraged in highly cluttered or complex environments such as interior of a building. Prior works have demonstrated obstacle avoidance through use of potential functions or by incurring additional penalties for traversing within a certain radius of an obstacle [9] [10]. These methods, however, provide no guarantee of yielding a feasible trajectory as the cost incurred only encourages, not enforces, a collision-free path.

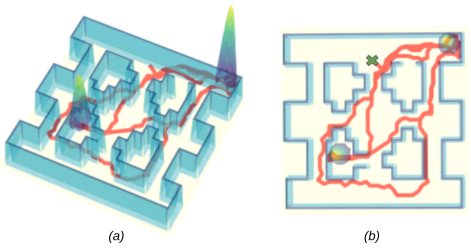


Fig. 1: Map of an indoor environment highlighting two information-dense locations illustrated as peaks. This work presents a novel ergodic search algorithm (GESCE) that generates feasible trajectories that navigate high-clutter environments while ensuring visitations to information-dense locations. (a) A 3D view of the building interior, highlighting the information peaks. (b) A top-down representation of GESCE-generated trajectory for a vehicle with Dubins car kinematics.

In this work, we introduce GESCE, a novel ergodic search algorithm capable of generating feasible trajectories in the presence of varying degrees of clutter. In our approach, we construct a graph over the environment such that all vertices and edges reside in unoccupied space. We then conduct a search over this graph utilizing the ergodic metric as a heuristic at each branching step of the search. We demonstrate that our methodology outperforms current state-of-the-art trajectory optimization techniques, scoring an order of magnitude lower in ergodicity values in low-clutter environments. Furthermore, we emulate high-clutter environments by leveraging multi-agent path finding (MAPF) benchmarking maps [11] and show that our algorithm consistently yields feasible trajectories with ergodicity values in the order of 10^{-3}, where other algorithms fail to produce any feasible trajectory at all.

The remainder of the paper is structured as follows: Section II provides a brief literature review on how obstacles are handled in prior works. Section III constructs the necessary background of ergodic coverage and outlines the vehicle dynamics for our simulations. Section IV dissects the components of our approach, and Section V showcases a comparison of results between our algorithm with prior works. Section VI concludes the paper.

[1]Burhanuddin Shirose is from the Department of Mechanical Engineering, Carnegie Mellon University, USA. [2] Adam Johnson is from the Department of Electronics and Computer Engineering, Carnegie Mellon University, USA. [3] Bhaskar Vundurthy, Howie Choset and Matthew Travers are from The Robotics Institute, Carnegie Mellon University, USA. {bshirose, adamjohn, pvundurt, choset, mtravers}@andrew.cmu.edu

II. Literature Review

Traditional methods for ergodic search use gradient-based optimization to generate a set of controls that minimize the ergodicity of the corresponding path. In order to handle obstacles in the environment, these prior works either include obstacles as constraints in their optimization formulation or discard trajectories that intersect with obstacles mid-optimization. One of the earliest works that deals with obstacles in the context of ergodic search is [10], where the authors utilize potential field-based obstacle avoidance techniques. They create repulsive fields around an idealized circular obstacle to guide trajectories around it, achieving high ergodic coverage without collisions. However, this method relies on modeling obstacles as circles, making it challenging to represent complex geometries such as the internal architecture of a building.

In [9], the authors introduce additional costs for obstacle intersections during the optimization and formulate their problem as a stochastic trajectory optimization. They sample the control space to generate initial trajectories and penalize those which intersect obstacles to refine the search. While this work has the prospect of handling cluttered environments such as the interior of a building, it struggles to overcome situations where all sampled trajectories within the computation horizon may intersect with the obstacles, such as a tight corridor.

An alternative effort for achieving obstacle avoidance in ergodic search which favors hard constraints instead of soft constraints is presented in [12]. The authors propose the use of control barrier functions (CBFs) generated around obstacles to serve as additional constraints for trajectory optimization. The functions are weighted by a term which controls the proximity a robot is allowed to traverse near an obstacle. Since the barrier functions serve as constraints for the optimizer, they ensure the feasibility of the solution in simple environments. However, the use of CBFs makes the algorithm prone to failure in narrow corridors and difficult to escape local minima.

In this work, we begin by constructing a graph that represents the accessible free space for the robot. Subsequently, we execute a search algorithm on this graph aimed at producing a path that minimizes its ergodicity value. By virtue of the graph being on unobstructed areas, the resultant trajectories are inherently obstacle-free. Further, any local minima introduced by the environment is naturally handled as the search expands from the start location to scan the entire graph.

III. Background

In this section, we provide a background of the ergodic metric and the formulation of vehicle dynamics used in our simulations.

A. Ergodic Metric

In our work, we utilize the formulation of ergodicity detailed in [9]. Here, the authors present that the time-average characteristics of a robot's path, denoted by $\gamma : (0, t] \to X$, evaluate the proportion of time spent at a specific point $\boldsymbol{x} \in X$, where $X \subset \mathbb{R}^d$ represents a domain of dimensionality d. The time-average attributes of a trajectory at a point $\boldsymbol{x}$ are defined as follows:

$$\Gamma(\boldsymbol{x}) = \frac{1}{t}\int_0^t \delta(\boldsymbol{x} - \gamma(\tau))d\tau$$

where δ is the Dirac delta function.

Let $\xi(\boldsymbol{x})$ be the given probability distribution function defined over the domain. The ergodicity of a robot's trajectory with respect to $\xi(\boldsymbol{x})$ is given by

$$\Phi(t) = \sum_{k=0}^{m} \lambda_k \left|\Gamma_k(t) - \xi_k\right|^2$$

where the coefficient λ_k, defined as $\lambda_k = \frac{1}{(1+|k|)^s}$, assigns greater importance to lower frequency components with $s = \frac{d+1}{2}$. The Fourier coefficients of the distributions $\Gamma(\boldsymbol{x})$ and $\xi(\boldsymbol{x})$ are denoted as $\Gamma_k(t)$ and ξ_k respectively.

$$\Gamma_k(t) = \langle \Gamma, f_k \rangle = \frac{1}{t}\int_0^t f_k(\gamma(\tau))d\tau$$
$$\xi_k = \langle \xi, f_k \rangle = \int_X \xi(\mathbf{x}) f_k(\mathbf{x})d\mathbf{x}$$

where $f_k(\boldsymbol{x}) = \frac{1}{h_k}\prod_{i=1}^{m} \cos\left(\frac{k_i \pi}{L_i} x_i\right)$ are the Fourier basis functions, h_k is a normalizing factor, $m \in \mathbb{Z}$ is the number of basis functions, and $\langle \cdot, \cdot \rangle$ is the inner product with respect to the Lebesgue measure.

B. Car Kinematics

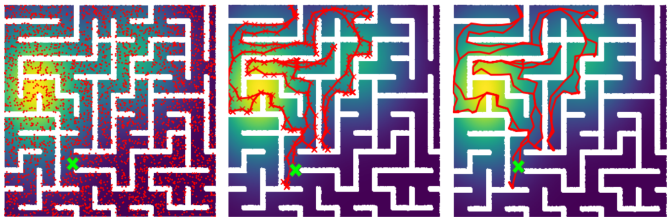
In this work, we consider our robot to have the kinematics of a Dubins car. The Dubins car model is a simplified representation of a wheeled robot's motion. The robot's configuration is given by its position (x, y) and heading angle (θ). Its motion can be defined by specifying the sequence and parameters of elementary movements (straight lines and circular arcs). The robot's dynamics can thus be written as:

$$\dot{x} = v\cos\theta, \quad \dot{y} = v\sin\theta, \quad \dot{\theta} = \omega$$

where v is velocity, and ω is turning rate. Dubins car model uses a constant value for v and controls the vehicle with $\omega_{\text{min}} < \omega < \omega_{\text{max}}$.

IV. GESCE: Graph-based Ergodic Search in Cluttered Environments

In this section, we present our proposed solution for generating an ergodic trajectory given an obstacle map of the environment and the target probability distribution. Our solution's methodology features three distinct components: (1) Graph generation, (2) Ergodic search over the graph, and (3) Trajectory generation given the graph search's output, as outlined in **Algorithm 1**. A detailed discussion about each component is covered in its respective subsection.



(a) (b) (c)

Fig. 2: Pictorial representation of different steps in GESCE from **Algorithm 1**. (a) Firstly, the free space is sampled uniformly to generate nodes (shown in red). (b) Next, a node sequence is obtained from an ergodic metric-based graph search that results in the path with the lowest ergodicity value. (c) Lastly, this node sequence is used to generate a kinematically feasible trajectory. The green **X** is the robot's starting location.

Algorithm 1 GESCE algorithm

1: **Input:**Information and obstacle map, robot start location
2: **Output:** A kinematcally feasible ergodic path
3: Generate a random geometric graph G in the free space of the environment
4: Search over the graph using the ergodic metric as the heuristic using **Algorithm 2**
5: Post-process to generate a path which respects the robot-dynamics using **Algorithm 3**

A. Graph generation

This work uses a discrete approximation of the underlying space in the style of probabilistic road maps (PRMs) in order identify object-free paths. First, consider a Eucledean space, $\mathcal{E}$, describing the position of a robotic system. The first step in our algorithm is to partition the space into two subsets, $\mathcal{E}\_obs$ and $\mathcal{E}\_free$, representing the obstacles and the free space respectively. We approximately represent the Euclidean space with a random geometric graph (RGG) populated with N samples from $\mathcal{E}\_free \cap \mathcal{E}$. Connectivity of the RGG is of the r-disc type, where r is a user-specified hyperparameter. Any given two vertices are connected with an edge if they are visible in the presence of obstacles and the distance between them is less than r.

The result from this process is a low-resolution graph representative of the environment, in which neither the vertices nor any point along an edge intersects with an obstacle. Thus, a search conducted on this graph, where the robot is able to move along the edges, guarantees an obstacle-free path for the robot without any other constraints based on clutter.

B. Graph-Based Ergodic Search

The next step is to leverage the generated graph to compute a path that minimizes the ergodicity (Φ), also referred to as the ergodic path $\mathcal{P}(\boldsymbol{v})$, to every reachable node in the graph from the source node s. To this end, **Algorithm 2** begins with the immediate neighbors of the source node s (Lines $3-5$) and expands outward. The algorithm iterates through the neighbors of every subsequent node and uses the ergodicity of the path from the source node as a heuristic to determine an ergodic path from the source node to a given node (Lines $6-15$).

Algorithm 2 Graph-based Ergodic Search

1: **Input:** Graph $G(V, E)$, source node s, information map
2: **Output:** An ergodic path $\mathcal{P}(\boldsymbol{v})$ from s to all nodes
3: Initialize path to every node $\mathcal{P}(\boldsymbol{v}) \leftarrow \{s\}\ \forall\ \boldsymbol{v} \in V$
4: Initialize a priority queue Q using $\Phi(\mathcal{P}(\boldsymbol{v}))$ as keys
5: Add s to Q
6: **while** Q is not empty **do**
7: Pop the node $\boldsymbol{u}$ from Q
8: **for** each neighbor $\boldsymbol{u'}$ of $\boldsymbol{u}$ **do**
9: Generate a path $\mathcal{A}(\boldsymbol{u'})$ by adding $\boldsymbol{u'}$ to $\mathcal{P}(\boldsymbol{u})$
10: **if** $\Phi(\mathcal{A}(\boldsymbol{u'})) < \Phi(\mathcal{P}(\boldsymbol{u'}))$ **then**
11: $\mathcal{P}(\boldsymbol{u'}) = \mathcal{A}(\boldsymbol{u'})$
12: Add $\boldsymbol{u'}$ to Q
13: **end if**
14: **end for**
15: **end while**

It is worth noting that picking a path that passes through a high information region is naturally achieved by using ergodicity as a heuristic for graph search, as shown in Fig. 2(b). Consequently, the algorithm not only generates a path that minimizes ergodicity from a given start location but also generates the most ergodic path between the source node and every other reachable location in the environment. We refer to this as the *point to point* ergodic path and present details on the performance of our approach and comparisons with prior work in Section V-D.

In our search algorithm, we intentionally refrain from maintaining a closed list of expanded nodes, a feature commonly employed in Dijkstra-style graph searches. Absence of a closed list allows the search to visit each node multiple times thus enabling further exploration of the map. This deliberate omission is attributed to the nature of the ergodic metric, which is a highly non-convex function. As a result, the metric does not serve as an admissible or consistent heuristic. Consequently, global optimality of the results obtained from our search cannot be guaranteed.

Our graph search sequentially traverses nodes with the lowest ergodicity values. Through this process, it explores nodes while preserving their visitation order, thereby generating ergodic paths to each node. Upon completion of the search, the absence of nodes in the open list signifies that further visitation from any node would increase the ergodicity of the visited node.

An important distinction between our framework and previous approaches lies in the absence of predetermined path length constraints. While prior works such as [9][12] yield the generation of a fixed length path, our approach grants the solver autonomy in determining trajectory length based on the ergodicity value.

C. Trajectory Generation

In order to bridge the gap between the graph-based ergodic search and feasible trajectories for the robot, we employ a

dedicated trajectory generator module. This module accepts the path generated by the graph search and converts it to a sequence of kinodynamically feasible trajectories while checking for collisions with obstacles.

The trajectory generator module (**Algorithm 3**) iterates through the nodes in the increasing order of their ergodicities to compute a feasible ergodic trajectory (Lines $3-4$). To this end, the nodes in the ergodic path $\mathcal{P}(\boldsymbol{v})$ obtained from **Algorithm 2** are used to determine the Dubins trajectory (Lines $6-7$). As long as these trajectories do not intersect with any obstacles, they are appended to each other to obtain an ergodic trajectory from the source node to the node under consideration (Lines $8-14$). If there are any intersections, the node under consideration is discarded, and the algorithm switches to the node with the next lowest ergodicity value until all nodes are explored (Lines $8-10$).

Algorithm 3 Trajectory Generation Algorithm

1: **Input:** An ergodic path $\mathcal{P}(\boldsymbol{v})$ from s to all nodes, robot's kinodynamic (Dubins car) constraints, obstacle map
2: **Output:** Kinodynamically feasible ergodic trajectory $\mathcal{T}$
3: Create a new list $\boldsymbol{V}'$ with nodes ordered by increasing ergodicity $\Phi(\mathcal{P}(\boldsymbol{v}))$
4: **for** node $\boldsymbol{v}' \in \boldsymbol{V}'$ **do**
5: Initialize an empty trajectory $\mathcal{T}$
6: **for** each node pair $(\boldsymbol{v}'_{i-1}, \boldsymbol{v}'_i)$ in the path $\mathcal{P}(\boldsymbol{v}')$ **do**
7: Compute Dubins trajectory $\mathcal{D}$ from $\boldsymbol{v}'_{i-1}$ to $\boldsymbol{v}'_i$
8: **if** $\mathcal{D}$ intersects with obstacles **then**
9: Reset the trajectory $\mathcal{T}$
10: ***Break***
11: **else**
12: Append $\mathcal{D}$ to $\mathcal{T}$
13: **end if**
14: **end for**
15: **if** $\mathcal{T}$ is not empty **then**
16: **return** Ergodic trajectory $\mathcal{T}$
17: **end if**
18: **end for**
19: **return** 'No feasible trajectory exists'

V. Results

In this section, we begin by defining and quantifying the notion of clutter. Subsequently, we present the results of our approach compared to existing methods within the field, particularly focusing on low-clutter regions. Lastly, we illustrate our results on benchmark Multi-Agent Pathfinding (MAPF) maps that represent high-clutter environments.

A. Quantifying Clutter in Environments

Given our method is proposed as an ergodic search on cluttered environments, it is important to establish a metric that can quantify "clutter". While many alternatives to describing clutter exist, we present a metric that captures the distance of the farthest visible points from all the discretized locations in the environment. Specifically, we formulate our metric by uniformly discretizing a unit side-length map into 100*100 points and then casting diametric lines centered at every point to calculate the furthest distance to an obstacle. We then average the maximum distance of each point and take its inverse to quantify an environment's clutter.

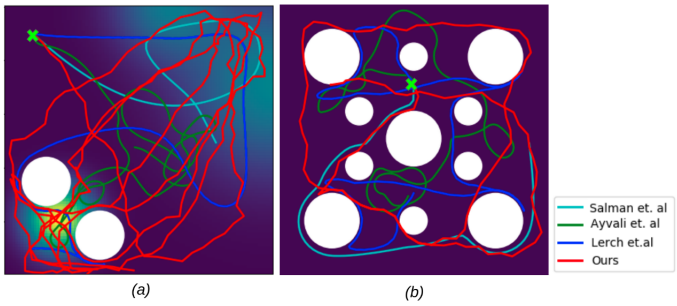


Fig. 3: Comparison of ergodic trajectories generated using GESCE (our approach) with prior works. (a) Information map featuring an arbitrarily generated information distribution with two circular obstacles in white. (b) Information map with a uniform distribution and higher clutter. All trajectories assume single integrator dynamics for the robots.

In this metric, a low value indicates low clutter, while a high value indicates a high amount of clutter. Therefore, by our metric, an unbounded obstacle-free map has a clutter value of 0, while a fully occupied map will have a clutter value of ∞. A highly cluttered environment now signifies that the robot's motion is highly restricted, indicating complex obstacle geometries. Table II illustrates the values of our proposed metric on the maps we use from prior works as well as those from the MAPF benchmark dataset [11].

B. GESCE performance on low-clutter maps (< 3.0)

In our study, we compare our results against [9] [10] [12] using two distinct maps. The first map, shown in Fig. 3 (a), is a map very similar to the one used in [9]. It contains two isolated obstacles with three information peaks distributed throughout. The second map, shown in Fig. 3 (b), is a map very similar to the ones on which [12] [10] have presented their results. It exhibits a uniform distribution with multiple obstacles. It should be noted that both maps have a low clutter value (< 3.0), which is indicative of large amounts of free space in the environment.

For our simulations, we construct the graph with $N = 5000$ sampled points and a connection radius r as 0.05, normalized such that the map sides are of unit length. The metrics we use to compare the results are the ergodicity values and success rate. The success rate signifies the percentage of runs a feasible output trajectory is obtained for each algorithm. As illustrated in Table I, we outperform [10] [9] significantly on the first map (Fig. 3 (a)), achieving an ergodicity value that is lower by a factor of 10. Compared to [12], we have a more moderate performance improvement, achieving an ergodicity value that is 33% lower. In order to demonstrate the repeatability of our results, the ergodicity values in Table I are an average of 5 trials with varying robot starting locations.

With an increase in the clutter from 1.72 to 2.49 for the second map (Fig. 3 (b)), our algorithm outperforms all of

	Map 1 Fig. 3 (a) (Single Integrator dynamics)		Map 2 Fig. 3 (b) (Single Integrator dynamics)		Map 2 (Not shown) (Dubins Car dynamics)	
	Ergodicity ($\times 10^{-2}$)	Success Rate	Ergodicity ($\times 10^{-2}$)	Success Rate	Ergodicity ($\times 10^{-2}$)	Success Rate
Salman et. al [10]	19.86	80%	14.55	100%	N/A	N/A
Ayvali et. al [9]	19.41	80%	8.27	80%	7.2	20%
Lerch et. al [12]	1.828	100%	2.786	100 %	N/A	0%
GESCE	**1.258**	**100%**	**0.29**	**100%**	**0.32**	**100%**

TABLE I: Comparison of ergodicity values from our work (using GESCE) against [10][9][12]

Obstacle Map	**Clutter**
Unbounded Map with No obstacles	0
Bounded Map with No obstacles	1.46
Map1 Fig. 3 (a)	1.72
Map2 Fig. 3 (b)	2.49
Maze-32-32-4 Fig. 4 (f)	6.04
Maze128-128-10 Fig. 4 (g)	6.42
Boston_0_256 Fig. 4 (h)	10.79
Berlin_1_256 Fig. 4 (i)	11.40
Paris_1_256 Fig. 4 (j)	14.07

TABLE II: Measure of clutter on maps used in Section V

the prior works by an order of magnitude with a 100% success rate. The results shown in Fig. 3 (b) deal with single integrator dynamics for the vehicles. An additional experiment with Dubins car dynamics is documented in the last two columns of Table I, where **Algorithm 3** now employs Dubins car constraints to compute the ergodic trajectory. While our method achieves a 100% success rate with an ergodicity of 0.32×10^{-2}, prior works either fail to generate feasible trajectories or provide output trajectories with high ergodicity values.

C. Results against high-clutter MAPF benchmark maps

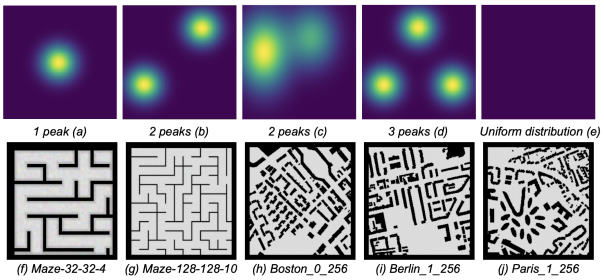


Fig. 4: (a-e) Display the generated information maps used for testing. (f-j) Present the obstacle maps selected from the MAPF benchmarking set [11], which were used to generate the results shown in Table III.

Obstacle Map	**Ergodicity** ($\times 10^{-3}$) **Information Map (Fig. 4)**				
	(a)	**(b)**	**(c)**	**(d)**	**(e)**
Maze-32-32-4	7.71	16.09	6.07	5.44	3.01
Maze128-128-10	0.47	4.48	3.18	4.94	2.45
Boston_0_256	0.64	8.90	5.05	6.80	4.55
Berlin_1_256	1.05	8.67	2.19	2.24	3.05
Paris_1_256	0.76	9.98	3.57	5.15	4.96

TABLE III: Averaged results from 5 runs.

We present the outcomes of our simulation on maps Maze-32-32-4, Maze-128-128-10, Boston_0_256, Berlin_1_256 and Paris_1_256, taken from the MAPF benchmarking dataset [11]. These maps have been selected for their high clutter value (> 6), signifying the robot's movement is constrained at many locations within the environment. For each obstacle map, we examine five distinct information peaks: one central information peak, two diagonally positioned equivalent information peaks, two peaks with differing strengths, 3 peaks presented in a triangular pattern, and a uniform distribution. All information peaks and obstacle maps are illustrated in Fig. 4.

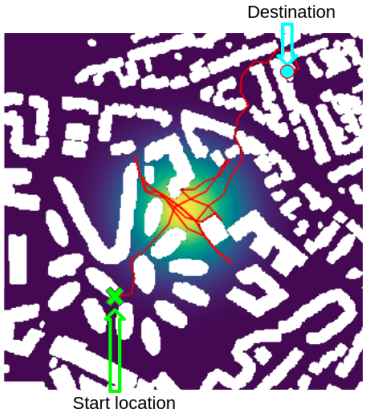


Fig. 5: Output of GESCE when used for point to point travel while conducting ergodic search

We conducted tests on the five aforementioned obstacle maps and information maps, performing five simulations for each scenario with randomly chosen starting locations. The results from these 125 scenarios are then averaged, with the corresponding values depicted in table III. Resulting trajectories for five representative tests are shown in Fig. 6, and results for all 125 runs can be found on GESCE Results (https://bshirose.github.io/projects/GESCE/). As evidenced in Fig. 6, the agent allocates its time in regions proportionally to the region's information density. Furthermore, it naturally avoids obstacles and effectively navigates mazes to reach peaks of information.

D. Point to point

In [9], the authors introduce a way to compute an ergodic trajectory given a start and end location. It is worth noting that GESCE's approach not only retains this feature but also extends it to highly cluttered environments. We show GESCE's output when used for point to point traversal in Fig. 5, which demonstrates our algorithm's ability to navigate from the robot's starting location, denoted by a green X, to a specified destination, denoted by a blue circle, while

Fig. 6: Results of GESCE on MAPF benchmarking maps taken from [11]. Green X represents the robot's start location.

spending a significant proportion of time in the information dense region centered in the map.

VI. Conclusions and Future Work

In this work, we have introduced a novel graph-based approach to perform ergodic search in cluttered environments. We have demonstrated that by searching a graph which represents free space in the environment, we can ensure the generation of collision-free trajectories with low ergodicity values. Our method outperforms the current state-of-the-art trajectory optimization techniques in low-clutter environments and yields feasible ergodic trajectories in high-clutter environments where previous methods fail. While our approach encodes the free space into the graph, the deployment on the robots relies heavily on the conversion of ergodic paths to kinodynamically feasible trajectories via **Algorithm 3**. Future work in this context could focus on developing generic local planners that include ergodicity in the trajectory generation modules. Furthermore, given our algorithm is tailored to act as an ergodic trajectory generator for complex environments, it fulfills the role of a foundational low-level ergodic trajectory generator for extension to multi-agent systems[13] [14].

VII. Acknowledgement

We thank Charles Noren and Prasanna Sriganesh for their continuous support and invaluable feedback.

References

[1] Y. Liu and G. Nejat, "Robotic urban search and rescue: A survey from the control perspective," *Journal of Intelligent and Robotic Systems*, vol. 72, Nov. 2013.

[2] J. L. Casper, M. Micire, and R. R. Murphy, "Issues in intelligent robots for search and rescue," in *Unmanned Ground Vehicle Technology II*, G. R. Gerhart, R. W. Gunderson, and C. M. Shoemaker, Eds., International Society for Optics and Photonics, vol. 4024, SPIE, 2000, pp. 292–302.

[3] M. Sun, A. Gaggar, P. Trautman, and T. Murphey, "Fast ergodic search with kernel functions," Mar. 2024.

[4] A. Macwan, G. Nejat, and B. Benhabib, "Target-motion prediction for robotic search and rescue in wilderness environments," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 5, pp. 1287–1298, 2011.

[5] G. Mathew and I. Mezić, "Metrics for ergodicity and design of ergodic dynamics for multi-agent systems," *Physica D: Nonlinear Phenomena*, vol. 240, no. 4, pp. 432–442, 2011, ISSN: 0167-2789.

[6] I. Abraham and T. Murphey, "Decentralized ergodic control: Distribution-driven sensing and exploration for multi-agent systems," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, Jun. 2018.

[7] L. M. Miller and T. D. Murphey, "Trajectory optimization for continuous ergodic exploration," *2013 American Control Conference*, pp. 4196–4201, 2013.

[8] Z. Ren, A. K. Srinivasan, B. Vundurthy, I. Abraham, and H. Choset, "A pareto-optimal local optimization framework for multiobjective ergodic search," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3452–3463, 2023.

[9] E. Ayvali, H. Salman, and H. Choset, "Ergodic coverage in constrained environments using stochastic trajectory optimization," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5204–5210, 2017.

[10] H. Salman, E. Ayvali, and H. Choset, "Multi-agent ergodic coverage with obstacle avoidance," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 27, no. 1, pp. 242–249, Jun. 2017.

[11] R. Stern, N. R. Sturtevant, A. Felner, *et al.*, "Multi-agent pathfinding: Definitions, variants, and benchmarks," *Symposium on Combinatorial Search (SoCS)*, pp. 151–158, 2019.

[12] C. Lerch, D. Dong, and I. Abraham, "Safety-critical ergodic exploration in cluttered environments via control barrier functions," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 10 205–10 211.

[13] A. K. Srinivasan, G. Gutow, Z. Ren, I. Abraham, B. Vundurthy, and H. Choset, "Multi-agent multi-objective ergodic search using branch and bound," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 844–849.

[14] Z. Ren, A. K. Srinivasan, B. Vundurthy, I. Abraham, and H. Choset, "A pareto-optimal local optimization framework for multiobjective ergodic search," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3452–3463, 2023.