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

Prioritized Robotic Exploration with Deadlines: A Comparison of Greedy, Orienteering, and Profitable Tour Approaches

Sayantan Datta and Srinivas Akella

Abstract—**This paper addresses the problem of robotic exploration of unknown indoor environments with deadlines. Indoor exploration using mobile robots has typically focused on exploring the entire environment without considering deadlines. The objective of the prioritized exploration in this paper is to rapidly compute the geometric layout of an initially unknown environment by exploring key regions of the environment and returning to the home location within a deadline. This prioritized exploration is useful for time-critical and dangerous environments where rapid robot exploration can provide vital information for subsequent operations. For example, firefighters, for whom time is of the essence, can utilize the map generated by this robotic exploration to navigate a building on fire. In our previous work, we showed that a priority-based greedy algorithm can outperform a cost-based greedy algorithm for exploration under deadlines. This paper models the prioritized exploration problem as an Orienteering Problem (OP) and a Profitable Tour Problem (PTP) in an attempt to generate exploration strategies that can explore a greater percentage of the environment in a given amount of time. The paper presents simulation results on multiple graph-based and Gazebo environments. We found that in many cases the priority-based greedy algorithm performs on par or better than the OP and PTP-based algorithms. We analyze the potential reasons for this counterintuitive result.**

I. Introduction

A significant research area in robotics and computer vision is 3D data acquisition and reconstruction to create a map of an initially unknown environment. A challenging problem is to select the locations in the partially observed map that the robot must visit to explore and map the environment further. The speed and extent of exploration by the robot is constrained by its limited knowledge of the environment, and constraints on the available time and energy of the robot. The prioritized exploration problem's objective is to identify the layout of the environment within a deadline [1]. For example, such a deadline may be imposed by excessive radiation in an environment, where the robot has to explore quickly to avoid long term damage to its hardware, or by a need to return safely despite a rapidly depleting battery.

A commonly used technique is to iteratively compute a target position for the robot to visit, in the partially known map. The map is updated as the robot moves to the target position. To explore the environment quickly and efficiently, the robot should ideally visit each target location at most once. In practice, with limited knowledge of the environment, the robot should minimize revisiting locations that it has already visited. This connects our problem to the Hamiltonian Path Problem and node routing problems such as the Traveling Salesperson Problem (TSP). These well known problems have been used to model coverage problems where each vertex needs to be visited only once [2], [3].

The prioritized exploration problem is a variant of the exploration problem where the robot must explore an initially unknown environment to compute its layout (i.e., connectivity) maximally while returning to the home location within a deadline. It attempts to do so by creating a path that visits high priority regions while ensuring the robot returns to the home location within the deadline. The prioritized exploration problem can be modeled as the problem of visiting a sequence of the most rewarding vertices within the limited exploration time. Vertices with higher rewards provide better views of the unknown environment. This paper considers the following question: Would using a Orienteering Problem (OP) formulation or a Profitable Tour Problem (PTP) formulation to explore the environment improve performance?

The OP and PTP are NP-complete problems [4] and require time exponential in the number of vertices to solve optimally. In this paper, when needed we reduce the number of candidate vertices to a feasible size so the prioritized exploration problem can be solved rapidly.

The paper discusses the relevant literature in Section II, and describes adaptation of the Orienteering Problem and Profitable Tour Problem for the prioritized exploration problem in Section III. Results of simulation experiments are presented in Section IV, followed by a discussion of the behavior of the exploration algorithms in Section V.

II. Related Work

The problem of robotic exploration has been studied from multiple perspectives. The work described in this paper builds on previous work on robot exploration, mapping techniques, and path planning.

Cost-based frontier exploration techniques that have been used for single-robot exploration [5] and multi-robot exploration [6] have laid the groundwork for robotic exploration. Next-best-view [7] approaches such as receding horizon next-best-view [8] have also been used for 3D mapping of environments. Robotic exploration has been modeled as different problems such as target searching, mapping, and coverage in unknown or partially observed environments [9], [10], [11], [12], [3].

Another approach for robot exploration is Active SLAM, a variant of the Simultaneous Localization and Mapping problem (SLAM). Active SLAM is the task of actively planning robot paths while simultaneously building a map and

The authors are with the Department of Computer Science, University of North Carolina at Charlotte, NC 28223, USA. This work was partially supported by NSF Award IIP-1919233.

sdatta3@uncc.edu, sakella@uncc.edu

localizing within it [13], [12], [14]. Such techniques generate a consistent map that is essential for 3D inspection tasks, sometimes at the cost of slower exploration.

The task of robot exploration can be viewed as a variant of the Traveling Salesperson Problem (TSP) as the robot must compute a Hamiltonian path to visit the possible frontier locations to explore the map. Similarly, the Team Orienteering Problem and Orienteering Problem with Neighborhoods, variants of the prize-collecting TSP with budget constraints, have been used to address the multi-robot exploration problem [15], [16].

Prior work has focused on exploration with goals such as finding a specific target, or consistent mapping at the cost of slower exploration. Most approaches consider the exploration to be complete when either the entire environment is mapped or the target is found in the unknown location. The prioritized exploration problem that we previously addressed [1] considers partial exploration of the environment, within a changing deadline. A similar problem is addressed by using a Multi-Robot Team Orienteering formulation in [15], where the environment is explored using heterogeneous robots with different energy limits. We are not aware of a single-robot exploration approach that addresses the prioritized exploration problem using the Orienteering Problem or Profitable Tour Problem formulations.

The Orienteering Problem (OP) [17], [18], [19] belongs to the class of Traveling Salesperson Problems with profits, where it is not necessary to visit all vertices. The input to the OP is a complete graph, a time budget, a start and an end vertex. Each graph vertex has an associated positive prize p_i. The goal of OP is to determine a path from the start to end vertex within the time budget, while selecting vertices to visit to maximize the total collected prize. The travel time between any two vertices is assumed to be non-zero. OP is suited to address problems where the time budget is limited compared to the time required to visit all vertices. When the time budget exceeds the time to visit all the vertices, the path generated by solving the OP formulation might not provide the shortest path to visit the vertices.

The Profitable Tour Problem also belongs to the class of TSPs with profits. To address the cost of the path, the Profitable Tour Problem (PTP) [20], [21] seeks to maximize the net profit, the total prize collected along the path minus the path cost. This formulation provides the shortest path when the time budget exceeds the time required to visit all the vertices.

The next section discusses modeling the prioritized robot exploration problem using the OP and PTP formulations.

III. Problem Formulation

A single robot explores an unknown environment, senses and maps the area, and models the map as an exploration graph, $G_e = (V_e, E_e)$. V_e is the set of explored vertices and each vertex $v_i \in V_e$ represents a physical location in the exploration environment. Each vertex v_i has a non-negative prize p_i that represents its priority. The vertices are labeled based on the building structure they are located in, such as Corridor, Large Room, or Small Room. The set of vertices V_e is divided into two subsets: V_d, the set of discovered vertices in unvisited frontier regions, and V_{vis}, the set of vertices visited by the robot. E_e is the set of edges, where each edge $e_{ij} \in E_e$ connects two vertices v_i and v_j. An edge represents a collision-free path between the two vertices. The estimated time to traverse the path along an edge is c_{ij}.

The exploration algorithm selects a target vertex $v_t \in V_d$ for the robot to visit. The exploration map is updated based on sensor data from the robot enroute to v_t. Once the robot reaches v_t, a new exploration graph G_e is computed based on the updated map. Once vertex v_t is visited, V_d is updated as $V_d = V_d \setminus v_t$, and V_{vis} is updated as $V_{vis} \cup v_t$. The robot should return to the home vertex h, from where it originally began exploration, by the deadline t_r. The value of t_r is reduced to reflect the current time remaining, as the robot explores the environment.

A. Orienteering Problem Formulation

The objective of the OP is to maximize the prize collected while traversing a path on the exploration graph. A path here is the sequence of vertices the robot takes from the start to the end vertex. If the start and end vertices are the same, the path is called a tour. The prize p_i at each vertex is based on the priority of the vertex. For our formulation, we have set the corridor vertices to have the highest prize, followed by large rooms, and finally small rooms. Once the robot visits a target vertex, a new path is computed accommodating the newly discovered vertices from the previously unexplored region. The objective function is shown in Equation 1, where s is the start vertex. The binary variable y_i is 1 if vertex i is visited and 0 otherwise. The exploration graph is represented as a weighted adjacency matrix M. As the formulation requires a fully-connected graph, an all-pairs shortest path algorithm is used to create a fully connected distance matrix M_c. As the robot needs to visit only the discovered vertices, we create distance matrix M_{cd}, from M_c, to include only s, h and the discovered vertices. The number of vertices in M_{cd} is N.

The robot's exploration is constrained by the deadline imposed (Equation 2). The binary variable x_{ij} is 1 when the path contains the edge e_{ij} and 0 otherwise. To ensure that each of the discovered vertices is visited at most once, we have a set of inbound and outbound constraints for all the vertices (Equations 3.1 — 3.3). Equations 3.1 enforce a single inbound edge to h and a single outbound edge from s to make the robot return to h at the end of path and start from s at the start. Equations 3.2 ensure that there are no edges inbound to s and there are no outbound edges from h if $s \neq h$. For every vertex in the path that is not s or h, Equations 3.3 ensure that it has one inbound edge and one outbound edge. A set of subtour elimination constraints (Equation 4), based on the Miller-Tucker-Zemlin (MTZ) constraints [22], prevents independent subtours and ensures that all vertices in the solution are in a single path. This introduces a variable $u_i \in \mathbb{R}$ for each vertex i to compute the sequence of vertices in the generated solution.

$$\text{Maximize} \sum_{\substack{i=1\\ i\neq s,h}}^{N} p_i\, y_i \tag{1}$$

$$\sum_{\substack{i=1\\ i\neq h}}^{N} \sum_{\substack{j=1\\ j\neq i,s}}^{N_t} c_{ij}\, x_{ij} \;\leq\; t_r \tag{2}$$

$$\sum_{\substack{i=1\\ i\neq h}}^{N} x_{ih} = 1, \qquad \sum_{\substack{i=1\\ i\neq s}}^{N} x_{si} = 1 \tag{3.1}$$

$$\sum_{\substack{i=1\\ i\neq s}}^{N} x_{is} = 0, \qquad \sum_{\substack{i=1\\ i\neq h}}^{N} x_{hi} = 0 \qquad s\neq h \tag{3.2}$$

$$\sum_{\substack{i=1\\ i\neq j}}^{N} x_{ij} = y_j, \qquad \sum_{\substack{i=1\\ i\neq j}}^{N} x_{ji} = y_j \quad j\in\{1,\ldots,N\}, j\neq\{s,h\} \tag{3.3}$$

$$u_i - u_j + 1 \leq (N-1)(1-x_{ij}) \qquad \forall i,j = 2,\ldots,N; i\neq j \tag{4}$$

$$2\leq u_i\leq N, \quad u_i\in\mathbb{R} \qquad \forall i = 2,\ldots,N$$

$$y_i\in\{0,1\}, \quad x_{ij}\in\{0,1\} \qquad \forall i,j = 1,\ldots,N; i\neq j$$

B. *Profitable Tour Formulation*

The Profitable Tour Problem (PTP) [20], [21] is a variant of OP. Its objective is to maximize the difference between the total collected prize and the cost incurred. For our single-robot indoor exploration problem, the PTP objective function is the sum of prizes over all the visited vertices minus the total cost (Equation 5), where m is a multiplier that scales the vertex prizes. The PTP formulation has the same constraints as the OP, including the deadlines constraint.

$$\text{Maximize} \quad m\sum_{\substack{i=1\\ i\neq s,h}}^{N_t} p_i\, y_i - \sum_{\substack{i=1\\ i\neq h}}^{N_t} \sum_{\substack{j=1\\ j\neq i,s}}^{N_t} c_{ij}\, x_{ij} \tag{5}$$

For both the OP and PTP formulations, the first vertex in the path generated from the optimization solution is the target vertex, the vertex to be visited next by the robot. This first vertex plays an important role during exploration because it determines the set of new discovered vertices, including any high priority vertices. The graph G_e is updated with the new vertices after visiting the first vertex and a new path is recomputed. Visiting a high priority vertex (i.e., with a large prize) as the first vertex is likely to enable the exploration algorithm to scan a larger portion of the environment before the path is recomputed. Both the OP and PTP objective functions do not require the first vertex to be a high priority vertex. Consequently, the paths generated by the OP and PTP formulations are not guaranteed to have a high priority vertex as the first vertex. (This is in contrast to the prioritized greedy algorithm, where the first vertex is always the highest priority vertex [1].) When a high priority vertex is visited later in the path, multiple computed paths may exist with the same total prize along the path. In such instances, we swap vertices in the computed path when feasible without increasing the cost, so the robot can first visit the high priority vertex.

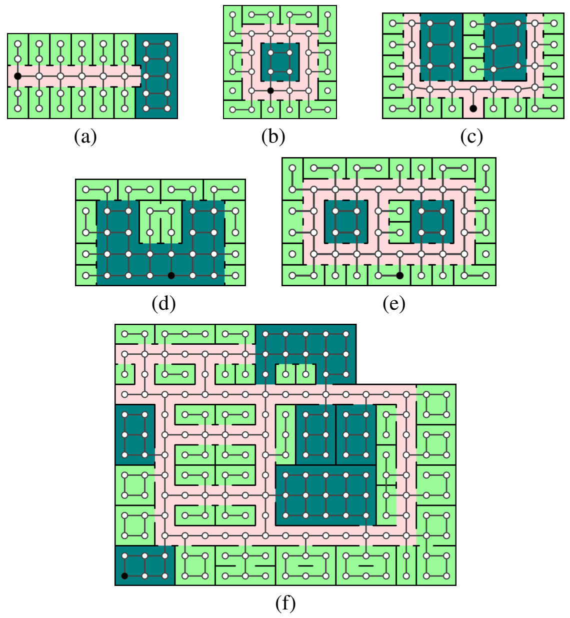


Fig. 1: Graph environments for testing exploration algorithms with small rooms (light green), large rooms (dark green), and corridors (pink). The first row consists of elementary environments: (a) Straight Corridor, (b) Looped Corridor, and (c) Branched Corridor. The second row consists of two environments with different connectivity: (d) Large Home, which has no corridors, and (e) Research Lab, which has two looped corridors. The third row illustrates the (f) Office environment, which is the largest. The black vertex shows the starting vertex for exploration. All environments are drawn to scale.

C. *Implementation in Graph Environments*

Graph environments are simulated environments represented as graphs, which are initially unknown. As the robot visits a vertex, adjacent vertices along the path are discovered. Example graph environments are shown in Figure 1. Each vertex has a pre-defined label: Small Room, Large Room, or Corridor, and an associated prize. In an office building, as a corridor connects other parts of the environment, the Corridor vertices have the highest prizes, followed by the Large Room vertices, and then the Small Room vertices. If the number of discovered vertices exceeds 15, they are clustered using the k-medoids algorithm into 10 clusters to ensure fast optimization solve times. The prize of a vertex cluster is the average prize of each vertex in the cluster.

D. *Implementation in Gazebo Environments*

Gazebo environments are 3D environments to simulate real-world environments. Here, the robot runs a SLAM algorithm to create an occupancy grid map. This map is skeletonized to a skeleton image and converted to a skeleton graph. The skeleton graph is used as the exploration graph G_e. This skeleton graph has discovered vertices generated along the frontier of the explored region. The home vertex is the vertex closest to the original start position of the exploration, and the start vertex is the vertex closest to the current robot position. The vertices are labeled as Corridors, Large Room, and Small Room based on the geometry of the obstacles (e.g., walls)

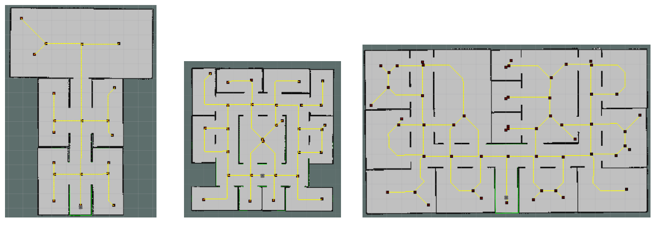


Fig. 2: Three Gazebo environments from [1]: (a) Straight Corridor (b) Looped Corridor (c) Branched Corridor. The figures show occupancy grid maps of the environments, where yellow lines represent edges and black points represent vertices. The environments are drawn to scale.

visible from the emulated Lidar scans. The three types of vertices have prizes similar to the graph environments. For each vertex, if multiple rays of the emulated Lidar scan overlap the exploration frontier, the vertex is marked as a discovered vertex. The discovered vertices are clustered if required, as described in Section III-C. The robot moves along the edges of the skeleton graph to reach the target vertex, to ensure a collision-free path.

IV. Experiments

We ran simulation experiments on a set of graph environments and a set of 3D environments in Gazebo. The graph environments are used to test the performance of the algorithms without the mapping, localization, and robot navigation components affecting the evaluation. The Gazebo environments enable repeatable robot exploration experiments in realistic environments. The experiments were run on an Intel Core i7 9800X with 64 GB of RAM, using the Python (v3.6.8) wrapper of Gurobi Optimizer (v9.5.1).

Graph Environments: These are a set of six simulated environments represented as graphs, as shown in Figure 1. Three are elementary corridor layouts from [1] and three environments are a combination of the elementary layouts. The Large Home (Figure 1(d)) has no corridors and consists of a large room with small rooms connected to it. The Research Lab (Figure 1(e)) has two looped corridors, and Office (Figure 1(f)), the largest environment, has multiple corridors with multiple intersections.

Gazebo Environments: A set of three environments, Straight Corridor, Looped Corridor, and Branched Corridor from [1] has been used to simulate real-world exploration (Figure 2). Using the Gazebo [23] simulator, these environments are explored using a simulated TurtleBot3 robot equipped with a single scan 360° Lidar with a range of 6 meters, simulating the Slamtec RPLidar A1M8 sensor. The GMapping SLAM algorithm [24] is used to create an occupancy grid map from the sensor data. This occupancy grid map is converted to the skeleton graph as mentioned in Section III-D.

A. Metrics

The performance of the exploration algorithms is evaluated using the percentage of the environment explored. For the graph environments, we compare the performance of the (1) Priority-based greedy exploration (P-greedy) algorithm from [1], (2) Cost-based greedy (C-greedy) algorithm motivated by [5], (3) OP-based exploration algorithm (OPE), and (4) PTP-based exploration algorithm (PTPE). The P-greedy algorithm visits the highest priority vertex that is feasible given the deadline. It prioritizes in the order of corridors, large rooms, and small rooms. For the Gazebo environments, we compare the P-greedy, OPE, and PTPE algorithms.

For the graph environments, explored regions of the map are represented by the set of explored vertices V_e, where $V_e = (V_d \cup V_{vis})$. The performance metric for the graph environments is the percentage of explored vertices in the complete environment. The performance metric for the Gazebo environments is the percentage of the total area explored. In the Gazebo environments, exploration time is measured in seconds and includes computation time and the time taken for robots to move in the environment. In all experiments, the robots are informed of a deadline at the start of exploration.

B. Results

We compare the performance of the two algorithms OPE and PTPE to the priority-based greedy exploration algorithm (P-greedy) [1]. For the OPE and PTPE, the prizes of the vertices in corridors, large rooms, and small rooms are set to 10000, 100, and 1, respectively.

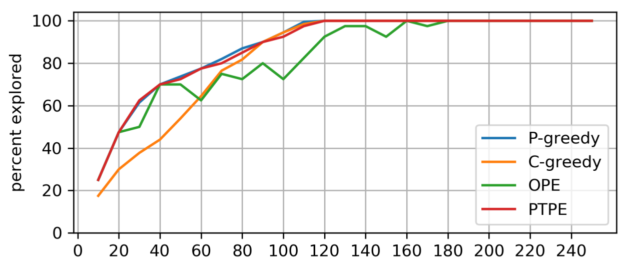
The algorithms (P-greedy, C-greedy, OPE, and PTPE with multiplier $m = 10$) have been tested on the six graph environments shown in Figure 1. The results are illustrated in Figure 3. They compare the percentage of the environment explored by the different exploration algorithms within a deadline.

To keep Gurobi's solve time under one second for the large graph environments, we clustered the vertices into 10 clusters when the number of vertices exceeded 15. Clustering the vertices in the exploration environment reduces performance, as shown in Figure 4. For PTPE and OPE, the difference in the percentage of explored vertices with and without clustering is significant when the number of discovered vertices is greater than 2–3 times the number of clusters.

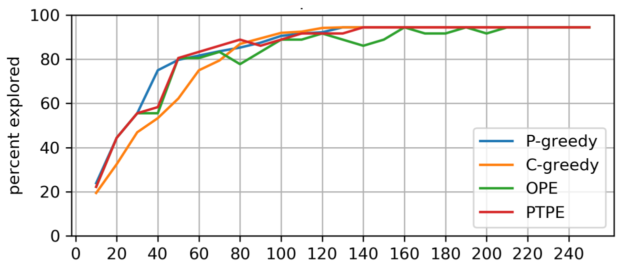
The performances of the P-greedy, OPE, and PTPE algorithms on the three Gazebo environments of Figure 2 are compared in Table I. For most deadline instances, the performances of all three algorithms are very close. For a few instances with performance differences, the causes can be identified. For example, the significant difference between P-greedy and the optimization based algorithms for the 1500 s deadline on the Straight Corridor environment stems from the skeleton graph having multiple vertices along the direction from the robot's position to the frontier. See Section V-B for details.

V. Discussion

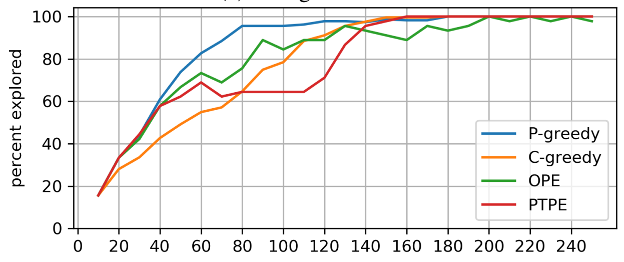
The OPE and PTPE algorithms create paths to visit the set of vertices that maximize their objectives while ensuring the robot satisfies the deadline. In both formulations, there is no guarantee that the first vertex in the path will be a



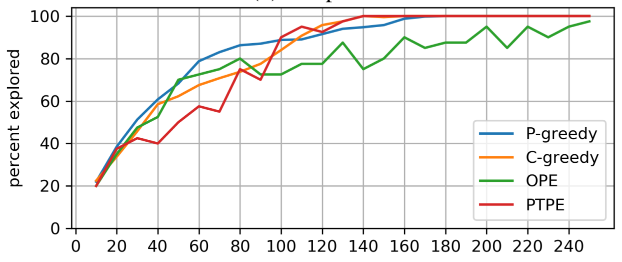
(a) Straight Corridor



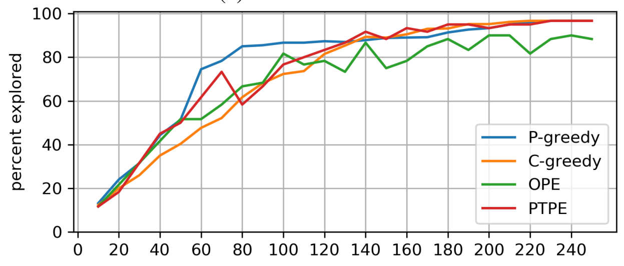
(b) Looped Corridor



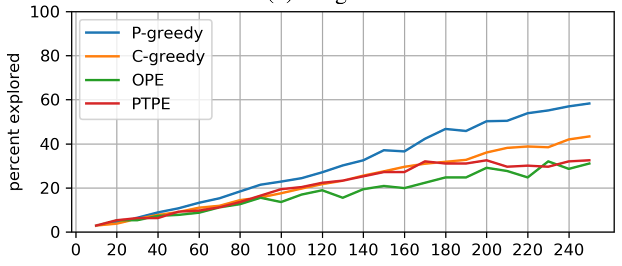
(c) Branched Corridor



(d) Large Home

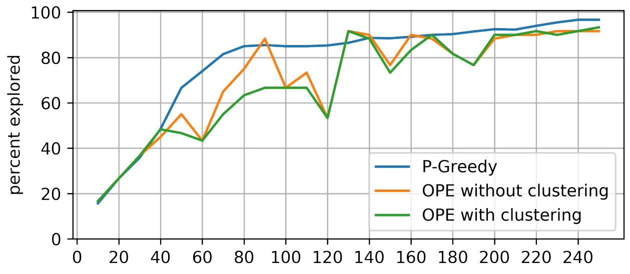


(e) Research Lab

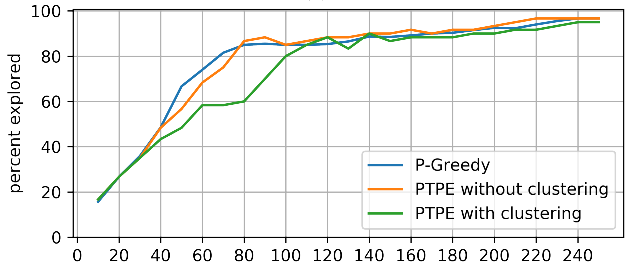


(f) Office

Fig. 3: Plots comparing the performance of the P-greedy, C-greedy, OPE, and PTPE prioritized exploration algorithms for the graph environments of Figure 1. The vertical axis shows the average percentage of explored vertices over five independent runs for each deadline. The horizontal axis shows the deadlines. Note that the Office environment requires a larger time to explore due to its larger size.



(a)



(b)

Fig. 4: Impact of clustering on prioritized exploration of the Research Lab graph environment. The horizontal axis shows the deadlines. The plots show performance of the (a) OPE and (b) PTPE exploration algorithms without clustering (orange) and with clustering (green).

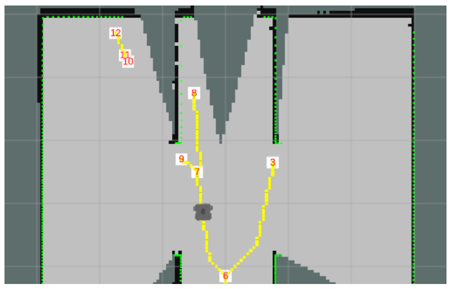


Fig. 5: Robot exploring the Straight Corridor Gazebo environment, moving along an edge of the skeleton graph. Skeleton graphs in Gazebo environments generate multiple vertices along the direction from the robot's current position to the exploration frontier. Here the vertices (7, 9, 8) are along the corridor. The skeleton graph shown here will be updated using the updated map after the robot reaches the target vertex. The frontier is the boundary between the light gray and dark gray regions of the map.

high priority vertex. As the path is recomputed after the robot visits a discovered vertex, the new path may again have the same limitation. The P-greedy algorithm, in contrast, chooses the highest priority vertex that is closest to the robot's current position as the target vertex, provided the robot can explore the vertex and return home within the deadline. Given

TABLE I: Exploration results for the Gazebo environments of Figure 2. P-greedy is the prioritized-greedy algorithm, OPE is the OP-based exploration algorithm, and the PTPE is the PTP-based exploration algorithm with multiplier $m = 10$.

Deadline (in secs.)	Algorithm	Straight Corridor				Looped Corridor				Branched Corridor			
		Corridor	Large Room	Small Room	Total	Corridor	Large Room	Small Room	Total	Corridor	Large Room	Small Room	Total
1500	P-greedy	99.3%	43.7%	97.0%	72.3%	99.9%	99.7	98.4	99.1%	76.3%	63.2%	57.3%	64.0%
	OPE	99.9%	63.9%	98.4%	82.5%	100%	99.7%	99.1%	99.6%	76.6%	77.7%	50.3%	65.3%
	PTPE	99.8%	80.0%	97.9%	89.8%	100%	99.7%	94.6%	97.2%	75.0%	60.7%	57.1%	62.8%
1000	P-greedy	99.6%	16.3%	96.4%	59.2%	100.0%	99.9%	97.2%	98.0%	66.8%	54.7%	49.2%	55.5%
	OPE	99.4%	24.2%	97.4%	63.4%	100%	99.7%	90.0%	94.9%	66.1%	57.0%	36.9%	50.5%
	PTPE	99.8%	28.5%	97.1%	65.3%	100%	99.5%	96.2%	98.0%	71.5%	49.9%	47.6%	59.8%
500	P-greedy	79.6%	0.0%	78.5%	41.8%	80.3%	97.9%	51.9%	68.3%	50.2%	24.2%	23.9%	30.9%
	OPE	80.3%	0.0%	52.6%	31.9%	87.4%	98.8%	50.6%	69.5%	55.1%	32.8%	22.4%	34.1%
	PTPE	79.3%	0.0%	51.7%	31.4%	79.9%	97.7%	49.3%	66.0%	55.1%	28.6%	27.9%	35.3%

this fundamental difference between the P-greedy algorithm and the non-greedy PTPE and OPE algorithms, this section discusses our observations from executing these exploration algorithms on different types of environments.

A. Exploration in the Graph Environments

The P-greedy algorithm has the best performance for most deadline instances, as shown in Figure 3. PTPE is second in performance in the Straight Corridor, Looped Corridor and Research Lab environments. For a few deadline instances, PTPE outperforms P-greedy on the Looped Corridor and Research Lab environments. In the Branched Corridor environment, P-greedy significantly outperforms the other algorithms. Both P-greedy and PTPE initially explore one of the two available corridors. Once the corridor is visited, P-greedy chooses to visit the other corridor, while PTPE explores the room vertices adjacent to the explored corridor instead of visiting the other corridor, hurting its performance. In the Large Home environment, OPE performs better than PTPE for smaller deadline instances.

B. Exploration in the Gazebo Environments

The Gazebo environments use skeleton graphs. A skeleton graph updated after the robot reaches a target vertex may not have a vertex at the current robot position. As a result, the robot spends a few seconds orienting and moving to the new skeleton graph, impacting the exploration time. Furthermore, a shortest path along the skeleton graph might not be the shortest path between two points on the map. As shown in Figure 5, the skeleton graph may generate multiple vertices between the robot's current position and the frontier of the explored map. The P-greedy algorithm visits the highest priority vertex that is closest to the robot, while OPE and PTPE may visit a different target vertex, farther away from the robot's current position. This results in frequent skeleton graph updates for the P-greedy algorithm, and each time, the robot spends a few seconds repositioning itself in the new skeleton graph, thereby affecting the P-greedy exploration performance.

C. Limitations of Online Methods

The exploration problem can be categorized as an online problem [25] as the robot sequentially receives information from the initially unknown environment. As the exploration environment is partially observed by the robot, optimal solutions for the partially known environments might not be optimal over the entire environment. We notice that each exploration algorithm performs differently for a given environment. As the structure of corridors and rooms determine the performance of the algorithm, we can consider the environment as an oblivious adversary [26]. However, if an adversary is aware of the exploration algorithm's priorities for visiting different building structures, it can devise an environment in which the algorithm would perform poorly. Although no particular exploration algorithm is a clear winner in all types of environments, we observe that the priority-based greedy algorithm performs competitively in almost all environments.

VI. Conclusion

This paper considers the prioritized exploration problem, whose objective is to compute the geometric layout of an initially unknown environment by rapidly exploring it and returning to the home location within a deadline. We formulated it as an Orienteering Problem and as a Profitable Tour Problem. In the graph environments, we found that a priority-based greedy exploration algorithm performs on par or better than the optimization based algorithms in most instances. In the Gazebo environments, all three exploration algorithms perform very similarly. While no algorithm emerged a clear winner across all exploration environments, the priority-based greedy algorithm performed quite competitively despite its low computational overhead.

We plan to investigate two directions in our future work. The first is further testing of the prioritized exploration algorithms in Gazebo and real-world environments to better understand their performance. This includes tests over a large set of deadlines and environments, and using a separate path planner to compute efficient collision-free trajectories for faster robot exploration. The second direction is to obtain the values of the vertex priorities based on the semantic information they provide about the building structure.

References

[1] S. Datta and S. Akella, "Prioritized indoor exploration with a dynamic deadline," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 3108–3114.

[2] H. Choset, "Coverage for robotics–a survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1, pp. 113–126, 2001.

[3] 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.

[4] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. Springer, 1972, pp. 85–103.

[5] B. Yamauchi, "A frontier-based approach for autonomous exploration." in *Computational Intelligence in Robotics and Automation*, vol. 97, 1997, pp. 146–151.

[6] ——, "Frontier-based exploration using multiple robots," in *International Conference on Autonomous Agents*, vol. 98, 1998, pp. 47–53.

[7] C. Connolly, "The determination of next best views," in *IEEE International Conference on Robotics and Automation*, vol. 2, 1985, pp. 432–435.

[8] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3D exploration," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 1462–1468.

[9] Y. Cai, S. X. Yang, and X. Xu, "A combined hierarchical reinforcement learning based approach for multi-robot cooperative target searching in complex unknown environments," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, 2013, pp. 52–59.

[10] C. Robin and S. Lacroix, "Multi-robot target detection and tracking: taxonomy and survey," *Autonomous Robots*, vol. 40, no. 4, pp. 729–760, 2016.

[11] W.-C. Lee and H.-L. Choi, "Complex semantic-spatial relation aided indoor target-directed exploration," *IEEE Access*, vol. 9, pp. 167 039–167 053, 2021.

[12] D. G. Vutetakis and J. Xiao, "An autonomous loop-closure approach for simultaneous exploration and coverage of unknown infrastructure using MAVs," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 2988–2994.

[13] I. Lluvia, E. Lazkano, and A. Ansuategi, "Active mapping and robot exploration: A survey," *Sensors*, vol. 21, no. 7, 2445, 2021.

[14] D. Pittol, M. Mantelli, R. Maffei, M. Kolberg, and E. Prestes, "Loop-aware exploration graph: A concise representation of environments for exploration and active loop-closure," *Robotics and Autonomous Systems*, vol. 155, p. 104179, 2022.

[15] T. Sakamoto, S. Bonardi, and T. Kubota, "A routing framework for heterogeneous multi-robot teams in exploration tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6662–6669, 2020.

[16] G. Best and G. A. Hollinger, "Decentralised self-organising maps for the online orienteering problem with neighbourhoods," in *International Symposium on Multi-Robot and Multi-Agent Systems*. IEEE, 2019, pp. 139–141.

[17] I.-M. Chao, B. L. Golden, and E. A. Wasil, "A fast and effective heuristic for the orienteering problem," *European Journal of Operational Research*, vol. 88, no. 3, pp. 475–489, 1996.

[18] P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.

[19] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Research Logistics*, vol. 34, no. 3, pp. 307–318, 1987.

[20] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[21] D. Feillet, P. Dejax, and M. Gendreau, "Traveling salesman problems with profits," *Transportation Science*, vol. 39, no. 2, pp. 188–205, 2005.

[22] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *Journal of the ACM*, vol. 7, no. 4, pp. 326–329, 1960.

[23] C. E. Aguero, N. Koenig, I. Chen, H. Boyer, S. Peters, J. Hsu, B. Gerkey, S. Paepcke, J. L. Rivero, J. Manzo, E. Krotkov, and G. Pratt, "Inside the Virtual Robotics Challenge: Simulating Real-Time Robotic Disaster Response," *IEEE Transactions on Automation Science and Engineering*, vol. 12, pp. 494–506, April 2015.

[24] 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.

[25] R. M. Karp, "On-line algorithms versus off-line algorithms: How much is it worth to know the future?" in *IFIP Congress*, vol. 1, 1992, pp. 416–429.

[26] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*. Cambridge University Press, 2005.