# MAPF-Based Multi-Robot Autonomous Exploration with Task Assignment in Unknown Space*

* Multi Robot Planning and Coordination Project Final Report


Xiaoyang Zhan
*Department of Mechanical Engineering*
*Carnegie Mellon University*
Pittsburgh, PA, US
xzhan2@andrew.cmu.edu

Akshay Raman
*Department of Mechanical Engineering*
*Carnegie Mellon University*
Pittsburgh, PA, US
akshayra@andrew.cmu.edu

Christopher Suzuki
*Department of Mechanical Engineering*
*Carnegie Mellon University*
Pittsburgh, PA, US
csuzuki2@andrew.cmu.edu



***Index Terms*—Multi-Robot Exploration, Multi-agent Path Finding**


## I. Introduction

In the multi-robot autonomous exploration and mapping problem, robots need to plan paths to explore and map an entire unknown environment given localization and sensor information (such as LIDAR, cameras, etc.). Multi-robot autonomous exploration is widely applied in various scenarios, such as inspection, and search and rescue operations. Previous work in this domain typically follows a global-local level pattern. At the global level, a task planner allocates different regions or low-level heuristics (such as viewpoints) to different robots. Meanwhile, at the local level, each robot executes exploration methods to complete the assigned tasks.

Most existing methods only include regions near the explored boundary in the global task allocation problem and use TSP-based or CVRP-based solvers to address it. However, we propose using a MAPF-based method to solve the global task allocation problem. In the MAPF problem, we consider a set of robots navigating to different goals, aiming to generate collision-free paths for each robot. This problem naturally shares similarities with the multi-robot exploration problem: given each robot's sensor range, more collisions within this range result in a lower average mapping rate, thus reducing the exploration algorithm's efficiency. Therefore, we propose applying MAPF methods in the exploration planning pipeline to minimize the overlap of sensor coverage during the exploration stage, thereby maximizing average exploration efficiency.

The proposed method consists of global and local planners. At the global level, the environment is divided into grid-based subspaces, each assigned to an agent, with all the targets of one robot linked by a path. The target assignment and path searching are tightly coupled as a TAPF problem. We treat robots with a given sensor range as amplified "virtual robots" and apply MAPF algorithms to plan paths and assign targets simultaneously, ensuring they avoid collisions en route to their targets. At the local level, we use existing algorithms to execute exploration within a subspace and handle path finding when transitioning from one subspace to another. The map data is then combined using a centralized system to stitch together a full depiction of the environment.

In summary, by utilizing the MAPF algorithm, our proposed method designs a novel global task assignment and path planning strategy for the multi-robot exploration pipeline. This global planning strategy is then tightly combined with our local exploration planner to build a comprehensive solution. The novelty and contributions are highlighted as follows:

- We propose a MAPF-based global task planner for the global level exploration, distinctly different from existing multi-robot exploration algorithms.
- We bridge the MAPF algorithm and 3D exploration with dense map information. Directly applying the MAPF algorithm with a 3D occupancy map would generate a vast search space, but we tackled this by embedding the MAPF module at the global level as guidance for the local planner.
- We design a local planner that couples with the global planner. It dynamically communicates with the global planner to provide heuristics that guide the global planner to achieve better performance.

## II. Related work

In robotics autonomous exploration domain, a common strategy is to divide the space into subspace(or sub-grids) and assign high level exploration task to the robots, as illustrated in [1] and [2]. Our methods also build upon this strategy. In addition, [2] applies VRP solver to solve global task assignment. Although we probably won't apply same VRP solver, this strategy could potentially be a good benchmark. Other strategies include [3] which utilize the assignment of subspaces as a TSP problem to assign exploration regions to robots. This work mainly focuses on minimizing travel distance for each robot in accomplishing the exploration task. In [4] they utilize submap descriptors to see if similar portions of the map have been explored and robot repulsion heuristic

to focus on complete exploration. Another method to split up the exploration into sub graph is explored by this paper where the agents are split into separate assigned targets using Multi-target Potential Field Exploration method

## III. Methodology

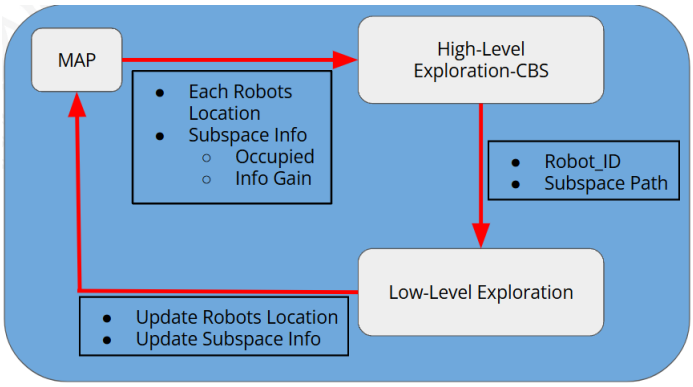



Fig. 1: System overview of integrating high level and low level planning

The overall system incorporate two stages of planning to solve this exploration problem. A global map is discretized into set sized sub-spaces that represent the overall shape of the environment. This map keeps all of the exploration information gain of each subspace that is updated by each agent and their location. The map information is passed to the high level which generates a path for each agent to explore a target sub-region. At each sub-region an agent starts its low level planner. The integration of the two planners are explained in depth in Section III-E.

In the high level, different from traditional task assignment methods such as TSP and VRP, we apply MAPF-based search: robots with a given sensor range (which is the valid mapping range) are treated as a amplified "virtual robots", and our goal is to plan paths for robots such that they avoid "collide" with each other in global amplified level. The motivation is, the more collision in sensor range, the less average mapping rate the robots have, thus the lower efficiency the exploration algorithm can achieve. However, those collision constraints are soft constraints, meaning they could collide with each other when necessary. Therefore, we want the planed path to has less collision in terms of sensor range but not necessarily zero. This framework is expected to give us higher flexibility compared with traditional target assignment methods, such as the flexibility to handle unexpected delays during exploration because of unknown obstacles/occlusions in environments.

In the low level, we apply existing single robot exploration algorithm to finish each tasks for robots. This algorithm also has the capability to handle robot dynamics and achieve safe navigation with occupancy-map based local planner and trajectory optimizer.

### A. High-Level Planner Method

The high-level planner in this study utilizes a Conflict-Based Search (CBS) method tailored for multi-target resolution, referred to as a multi-target CBS approach. The implementation of the multi-target CBS algorithm is demonstrated on a 2D holonomic simulation to establish the proof of concept, the details of which are discussed further in the results section. The primary function of the high-level planner is to determine the optimal subregions (targets) each agent should traverse, thus maximizing the environment's exploration. By employing CBS, the planner optimizes the avoidance of 'soft collisions' (Lidar collisions) between agents, enhancing the overall exploration throughput through maximized information gain for each visited subregion. This is done by maximizing information gain for each subregion visited. Initially, the simulation assumes each grid cell as a subregion where the Lidar's range surpasses the size of the subregion, thereby thoroughly capturing data and marking the cell as 'visited'. However, realistically, subregions are often larger than the Lidar range, with potential obstacles obstructing a complete view. In such cases, a subregion's visit does not guarantee its full exploration on the first attempt. Consequently, if residual information gain remains, the high-level planner might schedule a replan for the subregion at a later time. The exploration process initiates with

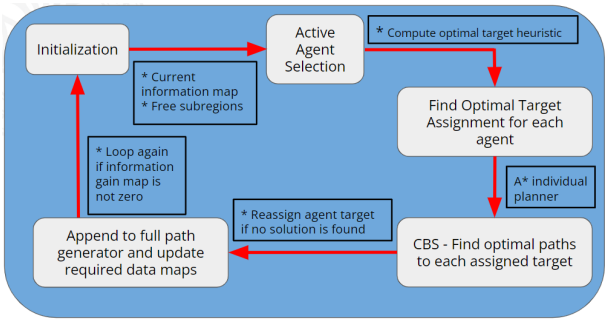



Fig. 2: The high level architecture divided into five submodules. The architecture highlights that the first two submodules intialize the map and are called each time within explore environment. Upon initialization, CBS find solution is run where the optimal target is selected and the path is pruned to avoid redundant Lidar soft collisions. The final path is concatenated to the full path the loop resets until the full environment is explored.

the activation of a central explorer function, detailed in the pseudo code provided in Algorithm 1. This function continues until either the entire 'explorable' environment is covered or a timeout occurs due to unresolved exploration areas. The 'explore environment' function manages this process and is divided into five critical subsections, as illustrated in Figure 2:

- **Initialization:** The environment is initialized using a symbolic map representation ('.' for free space and '@' for obstacles). Initially, only the map's boundaries are known, marking the interior as unexplored. Should an

obstacle be discovered during exploration, the high-level planner reinitializes the map to incorporate this new data. At the beginning of the exploration, the information map will contain an array of 1 for all the subregions as the area is completely unexplored. If the highlevel planner is recalled during exploration, the latest information gain map is sent into the explore environment as the starting input.

- **Active Agent Selector:** This submodule selects agents based on the remaining unexplored subregions. If a subregion is over 85% explored, it is marked as explored; otherwise, it's considered unexplored. Agents are allocated such that if there are more subregions than agents, all agents are utilized. Conversely, if there are more agents than subregions, the 'allocate agent' function determines the optimal distribution, preventing energy waste by avoiding the assignment of multiple agents to the same subregion.
- **Optimal Target Assignment:** For each active agent, a Dijkstra-based heuristic map is computed to facilitate A* in determining optimal paths to the selected targets. This process, detailed in subsequent sections, involves calculating and selecting the path that offers the highest information gain.
- **CBS and Target Reassignment:** A 'soft' CBS approach is implemented to mitigate Lidar range overlaps in subregions, prompting agents to select less direct but more diverse paths to maximize area coverage and minimize 'soft' collisions. If a chosen target does not minimize potential conflicts, the planner reassigns new targets to the agents.
- **Full Path and Map Update:** After target selection, paths are constructed and appended to the overall exploration path, and maps are updated accordingly. This loop continues until all subregions are explored or updated data necessitates a new cycle from a different starting point.

This methodical approach ensures a systematic and efficient exploration strategy, leveraging advanced pathfinding and multi-agent coordination techniques to tackle the complex challenges of multi-agent exploration.

### *B. Target Selection*

Target selection in our multi-agent exploration system employs an iterative algorithm that evaluates all valid points within a specified radius to identify the one that maximizes information gain. The algorithm uses an A* pathfinding approach, incorporating a penalization for revisiting previously explored cells to encourage exploration of new areas. The target with the highest information gain, determined through this method, becomes the focus for an agent's current iteration. The target selection process begins by initializing with the current information gain map, as depicted in Figure 3, which outlines the four main submodules involved in selecting the best target. At the start, a set of random points within the initial search radius is generated, excluding points that have already been explored. This radius is deliberately kept small initially to prioritize exploration near the starting area before expanding to more distant subregions. The algorithm filters out some of these points based on the momentum characteristics of each agent, calculated to maintain a consistent travel direction. This strategy prevents abrupt changes in direction and ensures that contiguous subregions are explored thoroughly before altering course. This is crucial for avoiding situations where a remote subregion remains unexplored until the later stages, which would be inefficient. Upon selecting all the valid target points, the best target is computed by looping through each of these points and finding the maximum heurisitic gain over each target. This calculation integrates three informed heuristics to derive the total heuristic value for each target. The final target and path is the value with the maximum heuristic gain for all the sampled points. If no suitable target is found within the initial radius, the radius is incrementally enlarged, and the selection process repeats.

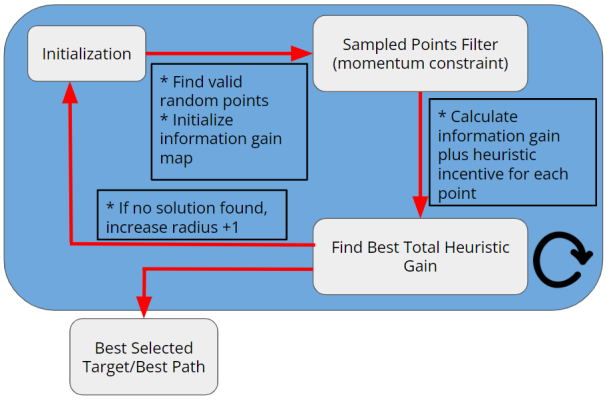
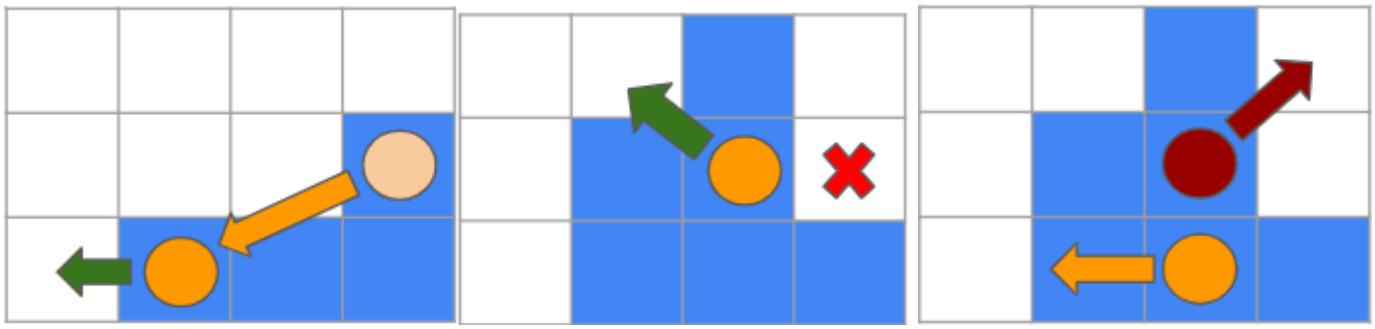



Fig. 3: The select target architecture loops until a solution is found. The radius will keep increasing till size of the map where it will locate a cell that is not fully explored.

**Algorithm 1** Environment Exploration and Target Selection

**function** EXPLOREENVIRONMENT
  **if** $initialPaths$ = None **then**
    **raise** Exception
  **end if**
  Append $initialPaths$ to $finalPaths$, update map
  **while** not ISFULLYEXPLORED **do**
    $activeAgents \leftarrow$ SELECTACTIVEAGENTS
    $newPaths \leftarrow$ FINDSOLUTION($activeAgents$)
    **if** $newPaths$ = None **then**
      **break**
    **end if**
    Append $newPaths$ to $finalPaths$, update map
  **end while**
  **return** $finalPaths$
**end function**=0



(a) Momentum Heuristic (b) Exploration Density Heuristic (c) Repulsion Potential Heuristic

Fig. 4: The three main heuristics used to guide agents to the most optimal target selection. The momentum heuristic which calculates the next sampled point to be to left of the green arrow after the momentum calculation shown in the orange arrow. The exploration density heuristic shows the incentive for the orange agent to move towards the green arrow as there is a higher amount of unexplored sub regions to the left. Repulsion potential heuristic guides the two agents away from each other.

### C. High-Level Planner Heuristics

The high-level planner utilizes three primary heuristics to optimize the exploration speed and quality, influencing both path formation and target selection. These heuristics are detailed below and their effectiveness is further analyzed through an ablation study in the results section. Figure 4 illustrates visually how each heuristic behaves.

- **Momentum Heuristic:** This heuristic filters potential target points based on their alignment with the agent's current momentum, defined by the movement vectors of past actions. The momentum field map, updated after each agent's move, dictates viable target points by ensuring they fall within the current movement direction. If no points meet this criterion, previously discarded points are reconsidered.
- **Exploration Density:** Contrasting with the momentum heuristic, this heuristic drives agents towards larger, unexplored areas irrespective of their current trajectory. It identifies and prioritizes extensive connected unexplored regions, thereby maximizing potential information gain from new explorations.
- **Replusion Potential:** To prevent redundant coverage and potential collisions, the repulsion potential heuristic employs a formula to calculate a repulsion factor that discourages agents from converging on the same area. The formula for this factor is:

$$V = \sum_{\substack{(x_a,y_a)\in\text{agents} \\ (x_a,y_a)\neq(x_0,y_0)}} A \exp\left(-\frac{(x-x_a)^2+(y-y_a)^2}{2\sigma^2}\right) \tag{1}$$

Where,

  - $V(x,y)$: Total repulsion potential at point $(x,y)$.
  - The sum is taken over all agents $(x_a,y_a)$ except for the agent located at $(x_0,y_0)$.
  - $A$: Amplitude of the repulsion.
  - $\sigma$: Spread of the repulsion effect.
  - $(x-x_a)^2+(y-y_a)^2$: Squared distance between the point $(x,y)$ and each agent $(x_a,y_a)$.

To ensure no single heuristic dominates the exploration strategy, a dynamic weighting system is implemented. Initially, the momentum heuristic is prioritized to leverage the predominantly unexplored environment. As exploration progresses and unexplored regions diminish, the weight shifts towards the exploration density heuristic. The weight of the repulsion heuristic remains constant throughout the process to consistently mitigate collision risks.

### D. Low-Level Planner Method

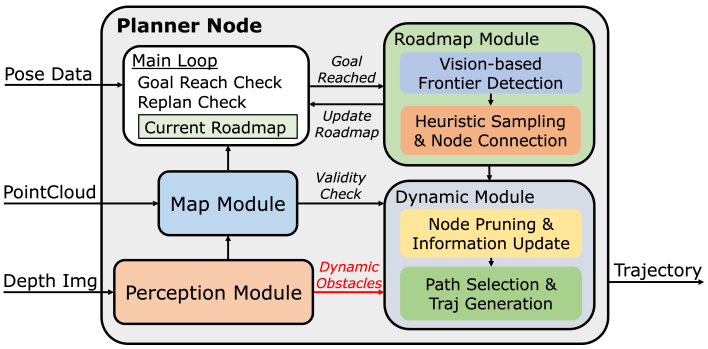



Fig. 5: Planner overview. The system input comprises the pose, pointcloud, and depth image data. At the beginning of each planning iteration, the roadmap module incrementally constructs the roadmap. Consequently, the dynamic module will update node information and generate trajectories for exploration and collision avoidance based on map and perception data.

The low-level planner is an implementation from a novel single agent exploration method called a Heuristic-based Incremental Roadmap Probabilistic Exploration (HIRE) planner [5]. This method is originally for UAVs navigating in dynamic 3D environments. However, for our implementation we use ground robots with Lidars to explore 3D space using 2D

roadmaps that consists of two main modules as shown in Fig 5: the roadmap module and the dynamic module. Input data include pose, point cloud, and depth image information. The roadmap module utilizes a vision-based detector to identify heuristic frontier regions, facilitating efficient sampling and probabilistic roadmap construction. The dynamic module updates node information and prunes invalid nodes based on map validity. The planner selects the best view path to generate collision-free trajectories for exploration. Heuristic incremental roadmap construction ensures even distribution of nodes within free space and growth towards unexplored regions. Node pruning and information update handle dynamic environments, ensuring collision-free nodes with updated information. Path selection and trajectory generation prioritize nodes based on information gain, selecting the best exploration path and generating trajectories using vision-aided B-spline trajectory. Replanning is fast due to collision-free roadmap nodes, enabling efficient collision avoidance. The information gain information will also be shared to the high level planner to make better informed decision in where to explore next for each agent.

### *E. Integration*

As shown in Fig 1, the exploration cycle begins with the map information gain and agent sub-space start positions being initialized. Information gain is calculated by the amount of unknown voxels divided by the total number of voxels that reside within the subspace. We do not incorporate free and occupied voxels as they are explored space. This information is sent to our high level exploration CBS which will generate exploration subspace paths for each agent, while minimizing sensor range collision to reduce redundant exploration. Once an agent arrives at a subspace it will start its low level exploration using a novel heuristic-based incremental roadmap exploration method. Once an agent is done exploring it will send its subspace location and exploration information gain update to the map where it will initiate a re-plan from the high level CBS as new information has been added to the map. This will repeat until the map is fully explored. Our high level planner was implemented using Python and low level planner in C++ which communicate through our ROS framework in a Isaac simulation enviornment.

## IV. Results

### *A. Simulation environments*

We build two simulator platform for testing global task planner and local planner(as well as overall algorithm). The global level simulator is a simple 2D grid-based simulator environment, as shown in Fig. 6. Every cell in this simulator represent a global subspace for robot to explore. We assume each robot spend same time to explore every global sub-space., currently. Therefore, the traversing time from one cell to its neighbor for each robot is the same. Different robots are shown as circles with different colors. We highlight the history path of each robot with corresponding color, which represent a global sub-space being explored.

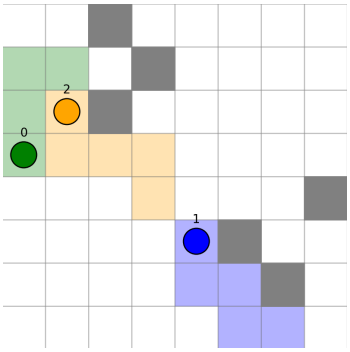



Fig. 6: An example 2d MAPF simulation environment: This environment contains 3 robots, they are represented as green, blue and yellow circles. The history trajectory of each robots, meaning regions explored by the robots, are also filled with the corresponding colors. The gray cells are inaccessible regions.

To test the local planner and overall algorithm, we build a realistic simulator platform with Isaac Sim, shown in Fig. 7. This environment simulate 3D real-world space with robot kino-dynamic models, as well as various sensors such as Lidar, depth camera, IMU, etc. It supports software interface with ROS communication. We also build a few different testing scenes. This platform enable us to fully simulate real-world multi-robot exploration scenarios.

Fig. 7: An example of 3D simulator. The left side is the scene, where 3 husky wheeled robot equipped with Lidar as well as IMU. The right side shows the RVIZ view of sensor data: The red arrws shows the odometry of the three robots, and point cloud in 3 different color shows the real-time lidar point cloud data from each of the robots.

To test the overall integrated system and the autonomous exporation results, comming with 3D mapping, we build 2 different environments: office and warehouse, where warehouse has higher obstacle density than the office scenario, as Fig. 8 shows.

### *B. High-Level Planner Ablation*

To evaluate the effectiveness of the high-level planner, we conducted an extensive two-part comparison against a baseline naive planner. This naive planner utilizes a simplistic target selection function that does not incorporate Conflict-Based Search (CBS) and is solely guided by the information gain per cell. The ablation study was carried out within a 2D Multi-Agent Path Finding (MAPF) simulation environment to ensure rapid model switching and to focus primarily on

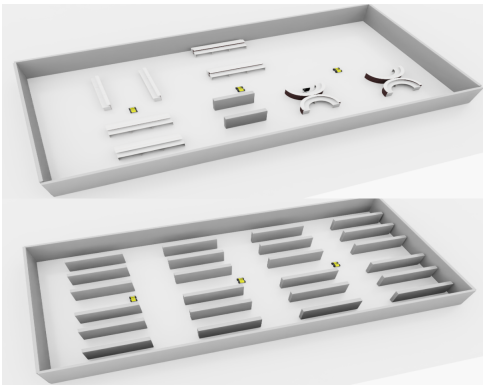


Fig. 8: 2 testing environments: office and warehouse. They both has size of 20x40m, containing 3 robots of 1.0x0.8m and sparse lidar of various ranges

the high-level target selection functions, distinct from the comprehensive exploration strategies discussed in the Overall System Integration Results section.

To test individual and overall heuristic effectiveness, different heuristic levels were used to create an ablation of exploration effectiveness. Three different high level heuristic scenarios were compared against a baseline naive exploration model. The high level naive model comprised of the explore environment function but the select target function was modified to randomly pick points and the best target was the point with most information gain. This reduced the model to look at only short term information gain without cbs. The three heuristic-enhanced models tested were:

- **Momentum-Only Model:** This model filtered points based on momentum but selected targets based purely on information gain.
- **Exploration Density-Only Model:** This model focused solely on the exploration density heuristic.
- **Full Heuristic Model:** This comprehensive model integrated all three discussed heuristics (momentum, exploration density, and repulsion potential).

This comprehensive model integrated all three discussed heuristics (momentum, exploration density, and repulsion potential).

Figure 9 presents box plots for the performance of the four configurations on both maps, with Figure 9a showcasing results for the 8x9 map and Figure 9b for the 4x9 map. The full heuristic model consistently outperformed the naive approach on both maps. Notably, the exploration density-only model generally surpassed the naive model but showed variability, occasionally underperforming due to its tendency to redirect agents across the map prematurely, leading to inefficient backtracking to previously unexplored regions. The momentum heuristic showed the most consistent results with the least spread as depicted by the box plot.

Figure 10 illustrates the maximum exploration times, measured in path lengths, across varying numbers of agents from 2 to 8 for both the Naive Planner and the Full Heuristic Planner. The naive planner exhibited a sharp decrease in exploration time with the increase from 2 to 4 agents, starting at 59 path length units and decreasing to 35 units, but showed diminishing returns thereafter, stabilizing at about 26 units by 8 agents. This pattern suggests significant initial efficiency gains with more agents, albeit with diminishing benefits as agent numbers increase. Conversely, the Full Heuristic Planner started with a higher initial path length of 61 units at 2 agents, indicating a slower onset compared to the Naive Planner. However, it displayed a more consistent improvement rate, achieving an optimal exploration time of 23 units by 8 agents. This gradual yet steady improvement underscores the Full Heuristic Planner's superior performance, particularly with higher agent counts, reflecting its enhanced ability to leverage additional agents effectively.

### *C. Low-Level Planner Results*

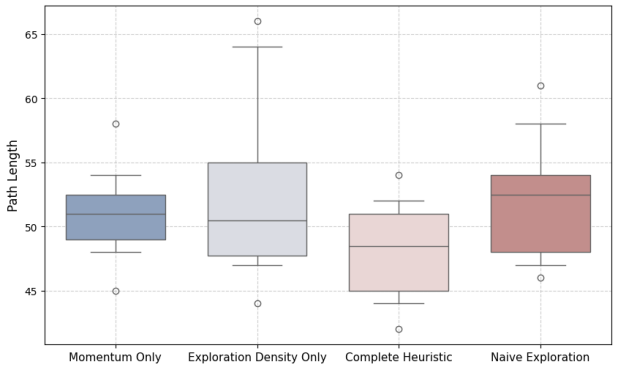
The results from the low level planner shows the robot is able to successfully explore a subspace within a given area. Once the search in the subspace is complete a command is sent to explore a new subspace. Figure 12 shows the UAV simulation finish exploring a given subspace and transitioning to a new subspace. In Fig 11 shows our single agent low level planner compared against other state of the art single agent exploration methods showing our method is more efficient than other existing methods [1] [6].

### *D. Overall System Integration Results*

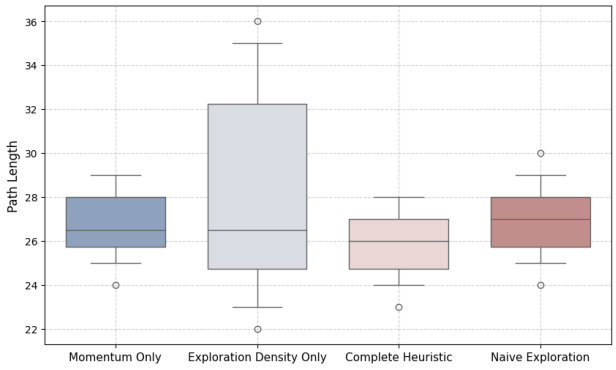
Combining both high-level and low-level planner, we build a complete multi-robot autonomous exploration system. This section will first describe the integration test result in one demo environment, then talk about ablation study and the lesson we took from the experiments.

We use a demo environment to verify our algorithm. Fig. 13 shows the whole process. In (a), The map is partially complete and the global planner drives each robot to different regions. Then, following the global subspace assignment, the local planner generate path to drive the robot transit from current subspace to destination subspace. Note that robot is not force to go to the center of next destination subspace. Instead, it searches over its own PRM graph and select points near the center of destination subspace. When there is no much information gain, the local planner also has freedom to choose goal by itself to clean local unknown regions. In (b), the exploration process has finished and the full map was built.

We also make ablation study over various parameter settings and simulation scenarios. We create 2 simulator environments with different obstacle density: office and warehouse. Both environments has a size of 20m x 40m, with 3 ground robot of size 1.0m x 0.8m. Each robot is carrying a sparse lidar with various sensing range. For each of the environments, we compared different results with sensing range of 5m and 10m, and subspace grid size of 5m and 10m. Fig. 14 shows the testing result in office environments. According to the graph, comparing the cyan and green curve, the cyan curve with




(a) 8 by 9 Path Length Ablation Study




(b) 4 by 9 Path Length Ablation Study

Fig. 9: The path length is compared across various configuration models over 2 maps. Path length is used as a metric for overall exploration time as it signifies the maximum length an agent has to travel before exploration is complete.

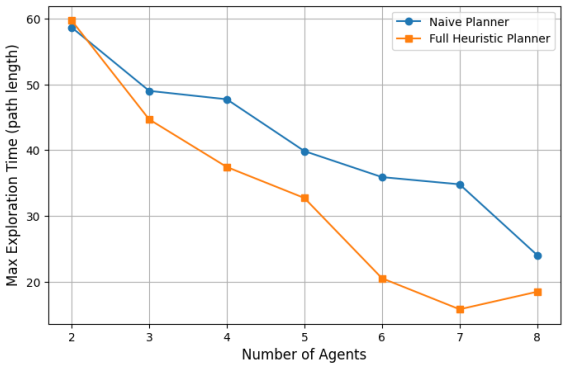



Fig. 10: Exploration time vs Number of Agents deployed on a 8 by 9 map. Initially, the full heuristic has a worse solution quality but as the number of agents increase, the exploration time significantly drops for the full heuristic planner. The max path length is used as the metric for estimating total exploration time.

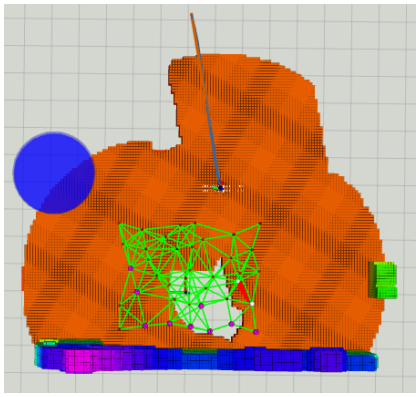


Fig. 12: Rviz output of a UAV exploration simulation environment: This Rviz shows a drone, which has fully explored a subspace. Represented in green lines, red and pink spheres is the roadmap of the explored area. The blue line is the trajectory the UAV is following to the new subspace. The orange is the explored floor of the environment that was seen by the drones camera.

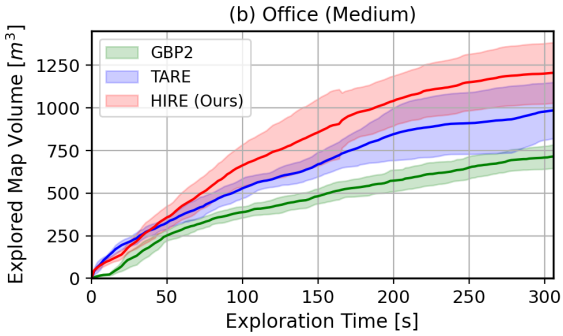



Fig. 11: Comparison of the average exploration rates in simulation environment Office (Medium). The shaded areas show the one-standard-deviation ranges of the exploration rates. Our planner achieves the highest exploration rate in the environment.

large subspace grid size(10 m) has better performance since it reaches higher exploration rate at each time stamp. The similar results also happen when we compare the red and blue curve: Red curve with larger subspace grid size(10 in this case) has better exploration efficiency. This result makes sense since if the subspace size is too small(close or even smaller than sensor range), the robot can explore most of the region at the first glance, so the local planner will not really be helpful. Instead, the local planner will actually be hurtful because local path will only give small reward, and it would be better to follow path from global planner to explore further subspace. However, we do observed that the improvement from blue curve to red curve is no as large as that from green to cyan. Our explanation is that for the cyan curve, its subspace grid size is much larger than its sensor range, so that will give a lot performance improvement. However, for the red curve, although the ratio of subspace size over sensor range is larger than blue, its own

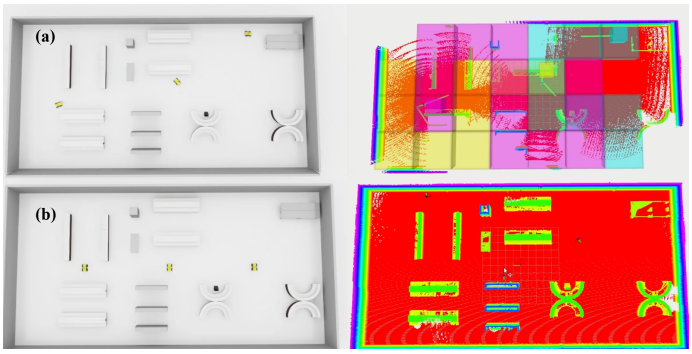



Fig. 13: Exploration process: The pose of robot are shown in little frames, the local path of robots are shown in green straight line and the global subspace assignment are shown in transparent cubes of 3 color corresponding to each robot. In addition, the little cubs in 3 color shows the current global subspace assignment for the very next step for each robot. Fig(a) is a video clip when the robot is still in the process of exploration and mapping. Fig(b) shows the final mapping result after finishing the exploration

subspace size is same as sensor range, so the performance should still not be ideal. Lastly, the performance of both red and blue curves are better than green and cyan curve, which is completely reasonable because the sensor range of red and blue curve (10m in this case) are twice than the other two. The sensor range is the most important factor that impact the exploration efficiency. The Fig. 15 shows the similar results in warehouse environment. Similarly, the scenarios with larger sensor range are always performing better, the larger subspace size also improves the performance for the comparison of red and blue curve when sensor range is 10m. Overall, our experiments observations are consistent with our expection of the algorithm.

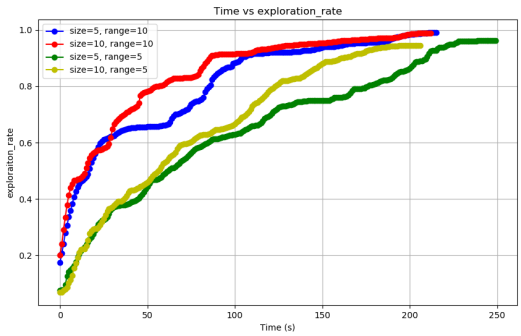



Fig. 14: Ablation study in office environment: The horizontal axis is time and the vertical axis is exploration rate, which is the percentage of voxels being explored up to current time stamp. The red curve is exploration result with lidar range of 10m and using subspace grid size of 5m, the blue curve is 10 m range and 10m grid size, and the same idea for the reset two.

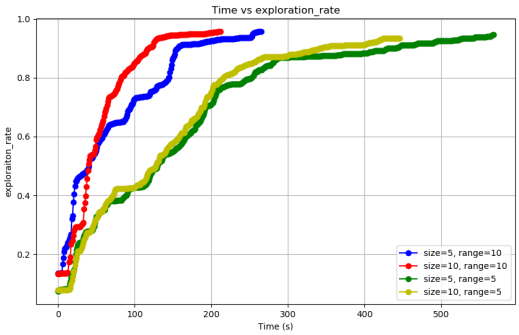



Fig. 15: Ablation study in warehouse environment: The horizontal axis is time and the vertical axis is exploration rate, which is the percentage of voxels being explored up to current time stamp. The red curve is exploration result with lidar range of 10m and using subspace grid size of 5m, the blue curve is 10 m range and 10m grid size, and the same idea for the reset two.

## V. Conclusion

In this project, we demonstrate a robust multi-robot autonomous exploration system, utilizing an integrated approach of high-level and low-level planning methods based on Multi-Agent Path Finding (MAPF) framework. The implementation of conflict-based search algorithms for high-level task assignment and target selection significantly enhanced the system's ability to efficiently manage the exploration space, reduce sensor overlap, and maximize overall exploration throughput. Additionally, using a MAPF based central framework allows the possibility to use heuristic algorithms to better guide agents during the exploration process while decentralized local decision making.

The system's effectiveness was validated through extensive simulation in diverse environments, showcasing its adaptability and superiority over traditional naive planners. The high-level planner's incorporation of momentum, exploration density, and repulsion potential heuristics provided a nuanced mechanism for dynamic target reassignment and path planning, ensuring optimal exploration with minimal redundancy. The low-level planning and overall integration was equally effective, demonstrating the ability to navigate and map complex 3D environments accurately. It employed advanced pathfinding algorithms and probabilistic roadmap strategies to maintain high efficiency and fidelity in exploration tasks.

## VI. Contribution Statement

- Xiaoyang Zhan: Initial implementation of global target selection; Implementation of simulator platform; final integration of high-level/low-level planner and overall integration experiments.
- Akshay Raman: Highlevel level planner: global target selection and cbs path resolving, heuristic functions and active agent selector, testing and validity of high level planning performance

- Christopher Suzuki: Low level planner: single agent local exploration; ros wrap high level planner and create initial communication and integration between high and low level planners in ros.

## REFERENCES


[1] C. Cao, H. Zhu, H. Choset, and J. Zhang, "Tare: A hierarchical framework for efficiently exploring complex 3d environments," in *Proceedings of Robotics: Science and Systems (RSS '21)*, July 2021.

[2] B. Zhou, H. Xu, and S. Shen, "Racer: Rapid collaborative exploration with a decentralized multi-uav system," 2022.

[3] A. Gautam, V. S. Shekhawat, and S. Mohan, "A graph partitioning approach for fast exploration with multi-robot coordination," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 459–465.

[4] Y. Zhang, Y. Zhang, and X. Li, "Multi-robot exploration system in unknown environment based on submap," in *2021 4th International Conference on Robotics, Control and Automation Engineering (RCAE)*, 2021, pp. 256–260.

[5] Z. Xu, C. Suzuki, X. Zhan, and K. Shimada, "Heuristic-based incremental probabilistic roadmap for efficient uav exploration in dynamic environments," 2023.

[6] M. Kulkarni, M. Dharmadhikari, M. Tranzatto, S. Zimmermann, V. Reijgwart, P. De Petris, H. Nguyen, N. Khedekar, C. Papachristos, L. Ott, R. Siegwart, M. Hutter, and K. Alexis, "Autonomous teamed exploration of subterranean environments using legged and aerial robots," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 3306–3313.