Integrating Multi-Robot Search Path Planning to Nav2

Manuel Agraz Vallejo **Roger Wienaah** **Michael Anku**
Collaborative Robotics and Intelligent Systems
Oregon State University
Corvallis, OR 97333
{agrazvam, wienaahr, ankum}@oregonstate.edu

1 Executive Summary

Search and rescue (SAR) missions can be hazardous to humans, especially when they involve entering buildings on fire or areas where debris can easily fall. In scenarios where the targets to be rescued are moving, such as when searching for a lost person or locating someone in a fire, the threat to human lives increases the longer the search takes. Multi-robot teams have shown success in simulation [4] and in hardware trials [7] in these types of missions, thus reducing the risk to human operatives. To coordinate these teams, multi-robot search planning methods [3, 2] have been used to find paths that locate the target in the least amount of time through information sharing. However, these multi-robot planning methods are not readily available when searching for packages in the most commonly used robotic middleware for navigation, the ROS2 Nav2 stack [5]. This complicates its integration with most available robotic platforms that rely on ROS2 [6]. In this work, we integrate the multi-robot efficient search path planning (MESPP) [3] algorithm into the ROS 2 Nav2 stack as an extension to the current global planner. To test this, we created a set of 3 test worlds simulating an indoor setting where a target will wander around the map. In this setting, we run experiments with 2 robots searching for the moving target. As our baseline, we implemented a coordinated exhaustive search of the map. Our results show that integrating MESPP with ROS2 Nav2 speeds up the average capture time of the moving target, while also reducing the amount of map coverage when compared to our baseline.

2 Initial Timeline and Revisions

Our initial timeline remained the same throughout the project. As shown in Fig. 1, we reached all of our goals, except for the plugin creation for ROS2 Nav2. Creating a plugin proved to be more challenging than initially anticipated, and thus, it was left for future work. The main bottleneck was integrating the algorithm with our ROS2 architecture.

Phases	Week	Goals	Responsible	Score
Phase 1	3	Project Proposal	Everyone	-
	4	Stage task creation with multiple robots and a moving target.	Manuel	5/5
	5	Test the NavStack baseline on each robot, and implement average capture time and area coverage percentage metric logging.	Manuel and Michael	5/5
	6	Implement multi-robot search path planning algorithm as a ROS node.	Roger	5/5
Phase 2	7	Integrate our approach with ROS2 NavStack as a plugin.	Manuel and Roger	4/5
	8	Run a total of 6 experiments, 1 on each of the 3 environments for baseline and ours.	Michael	5/5
	9	Project presentation.	Everyone	5
	10	Write and submit report.	Everyone	-
Stretch Goals		Run experiments using 3 and 4 robots, with 1 moving target. And 2,3 and 4 robots with 2 moving targets	Everyone	10

Figure 1: The project timeline split by phases, weeks, and goals. Every milestone has a score and a team member assigned. According to our rubric, we obtained a score of 29/30 points.

3 Method

The MESPP algorithm [3] represents the environment as a graph (see Fig. 2), where each node is a possible robot location, and each edge is a valid motion path, meaning movement without collision with obstacles. The goal is to coordinate multiple robots so they can search for a moving target efficiently across this graph. The algorithm keeps a belief distribution over all graph nodes, representing the probability that the target is at each location. At every step, this belief is updated using the target's motion model and the robots' sensing results. As robots visit and search certain nodes, the probability in those areas is reduced, and the planner focuses attention on the most likely unexplored regions. This shared belief state induces implicit coordination where each robot plans its path while taking into account the effect the other robot's action had on the belief state. This helps reduce duplicated search effort and improves team efficiency. In our implementation, the MESPP planner generated graph-based search waypoints, and these waypoints were sent to ROS2 Nav2 for execution. Nav2 handled local path planning, obstacle avoidance, and robot motion, while MESPP handled high-level search coordination. This allowed us to extend NavStack from single-robot navigation into a coordinated multi-robot search system. The implemented MESSP algorithm is provided below:

Algorithm 1 MESPP Algorithm

Require: Graph $G = (V, E)$, planning horizon h, team size m, mission deadline τ
Initialize target belief vector $b(0)$
for $t = 0$ to τ **do**
  for each robot $i = 1 \dots m$ **do**
    Fix the planned paths of all other robots $j \neq i$
      (Robots with $j > i$ assume their initial/current positions)
    Solve MILP over horizon h for robot i to maximize expected capture probability
    Update the planned path π^i for robot i
  end for
  Execute the first step of the joint paths for all robots
  Obtain sensor observations from all robots
  Update belief $b(t + 1)$ using target motion model M and observations
  if Target is captured (Same Vertex or Within Range) **then**
    break
  end if
end for

4 Experiments

The main task in each experiment trial was deploying two robots to find a moving target across three different indoor worlds. The experiments consist of 2 capturing robots and 1 moving target robot. All three robots used the ROS2 Nav2 stack for global and local planning, obstacle avoidance, and low-level control. We ran a total of 6 experiments: 3 worlds (see Fig. 3) and two methods on each, a baseline, and the MESPP algorithm. The baseline method consists of a coordinated exhaustive search; in this method, the agents keep a shared graph visitation list that they reference to avoid visiting nodes the other searcher already visited. The simulation environment used is the Stage robot simulator [1], the visualization of the searchers and the target can be seen in Fig. 4.

Each experiment trial is timed. The timer ends when one robot captures the target under two conditions: path blocking or proximity. The path blocking condition refers to the robots occupying the only neighboring node adjacent to the target's current node. The proximity condition is a simple Euclidean distance check if the target is within 0.5m. At the end of each trial, we report 2 metrics: the capture time and the number of nodes visited. Finally, for each world and algorithm combination, we ran a total of 5 trials.

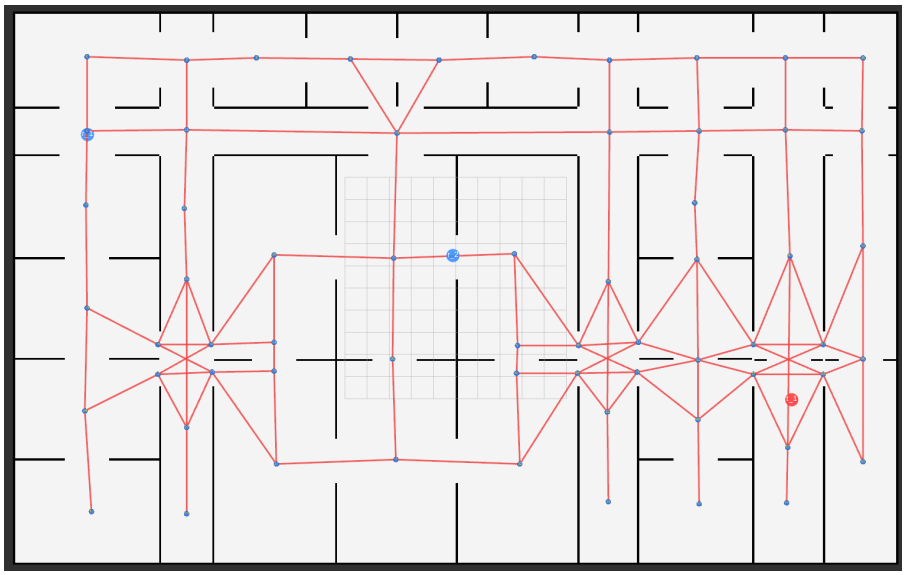
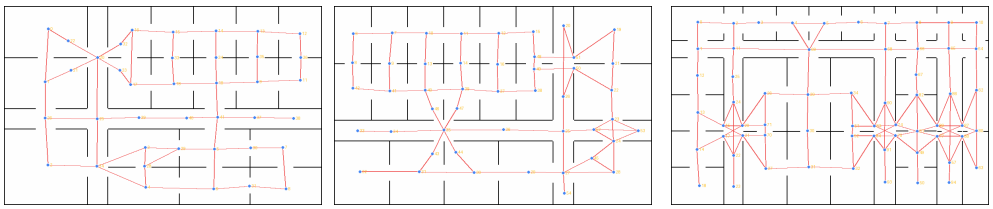


Figure 2: The search robots are represented in blue, and the target in red. The search robots have to navigate through the graph to find the moving target. The fact that the target is constantly moving can prolong search if the robots do not coordinate efficiently.



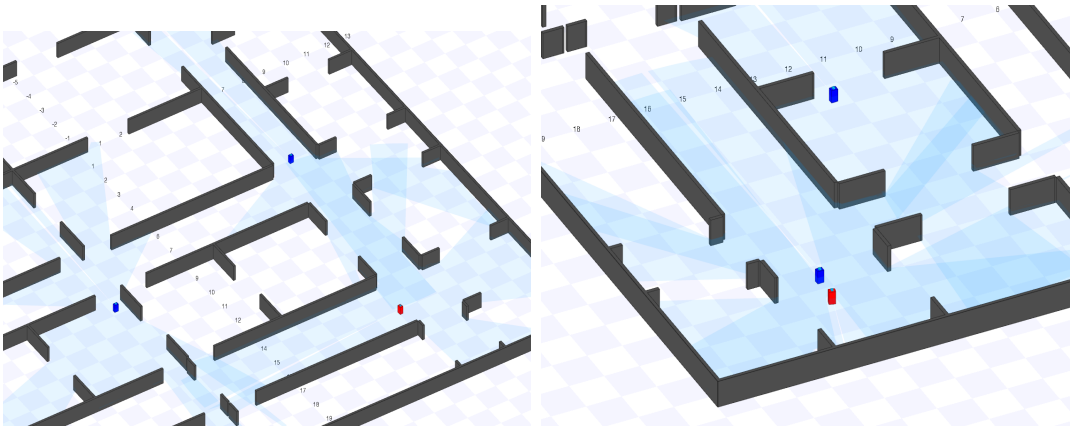
(a) World #1 and its graph. (b) World #2 and its graph. (c) World #3 and its graph.

Figure 3: The 3 worlds used for experiments, the blue dots represent valid pose nodes and the red lines represent the edges between the nodes. We used a custom graph editor for each world to discretize the occupancy grid map into a graph structure, where nodes represent feasible robot positions and edges represent valid transitions between them for search planning.

5 Results

The capture time and node explored percentage can be seen in Fig. 5. We see that across all worlds, the MESPP approach achieves lower average node coverage and capture times. This occurs due to the shared belief state coordination guiding the planning of agents towards nodes that have a higher probability of containing the target. For example, in world #2, the exhaustive search baseline explored above 90% of the map's nodes on most trials, while our MESPP baseline had an average of 30% coverage. This difference is also reflected in the capture times of world #2, where the average capture time of the baseline is 180s, compared to the average 79s of our MESPP implementation, achieving a 56% faster capture time.

As a summary of all worlds, Fig. 6 shows the boxplot including 5 trials from each world, totalling 15 datapoints per boxplot. This shows an overlap between the minimum capture time of the baseline and the median of our MESPP method; these capture time values come mostly from the results in world #3, which is where there is the most overlap between capture times. We see a similar overlap on the percentage of nodes explored; however, this overlap is much smaller, as it is only between the minimum baseline value and the maximum MESPP value. Overall, across all worlds, we see that the average capture time of the baseline is 140s, almost twice the average time when compared to the 72s of the MESPP approach.



(a) The searcher robots searching in world #3.

(b) The target was captured due to proximity.

Figure 4: The image shows the Stage simulation showing the 2 blue searcher robots and the red moving target robot. Fig. 4b shows a capture instance due to proximity; blocking wasn't possible given that the target robot still had two neighboring rooms free of searchers.

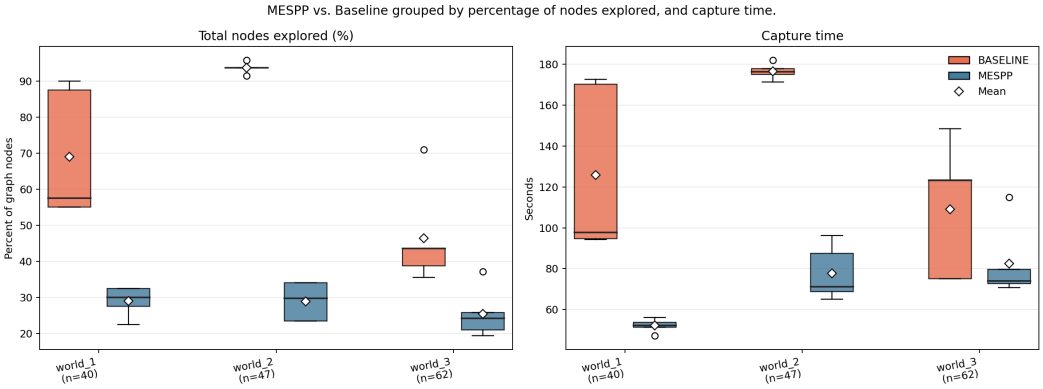


Figure 5: Percentage of nodes explored and capture time data per world.

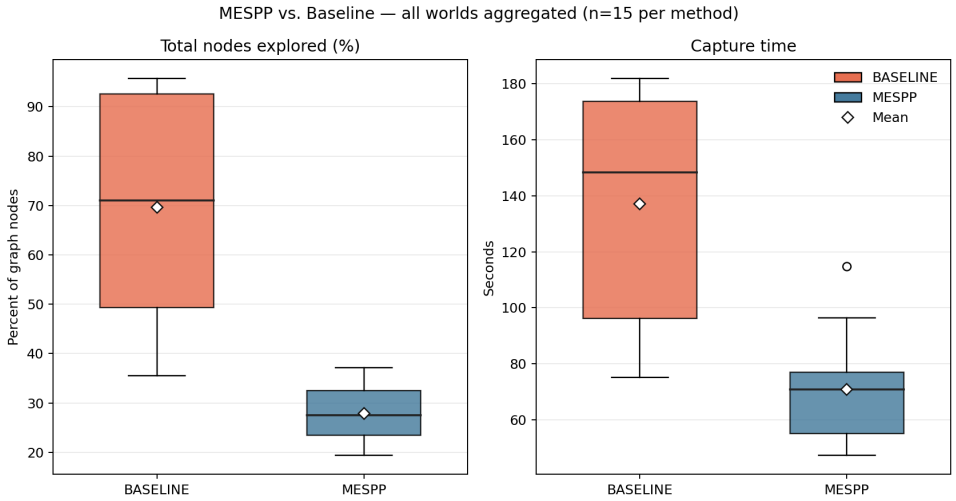


Figure 6: The boxplots show the aggregated data across worlds for both capture time and percentage of nodes explored.

6 Discussion

This project showed that extending ROS2 Nav2 to multi-robot systems requires more than just adding more robots. It requires seamless coordination. The baseline exhaustive search led to redundant exploration, while the MESPP-based approach enabled coordination, reducing overlap and improving search efficiency. This resulted in faster target capture and fewer nodes explored. A key learning was the importance of probabilistic planning. By maintaining a belief of where the target is likely to be, the robots were able to focus on high-probability regions instead of searching randomly. This is what provided the main difference between the exhaustive search and the MESPP. We also encountered practical challenges. Integrating a high-level planner with Nav2 required additional effort since Nav2 is primarily designed for single-robot navigation. Additionally, increasing graph density improved path quality but also increased computational cost, highlighting a trade-off between performance and efficiency. Another important takeaway was the benefit of implicit coordination, which provided good performance while remaining computationally feasible, compared to more expensive centralized approaches. On a whole, this project demonstrated that combining high-level planning algorithms with existing navigation frameworks can significantly improve multi-robot performance, especially in search applications.

7 Future Work

Due to time constraints, some planned improvements could not be fully implemented and are proposed as future work. These include:

Integrating MESPP as a Nav2 plugin: Converting the MESPP implementation into a fully functional Nav2 plugin and embedding it within the behaviour tree would allow seamless integration with existing NavStack workflows. This would improve modularity and make the system easier to reuse and extend.

Scaling to larger maps and more robots: Future experiments should consider larger and more complex environments, as well as increasing the number of robots. This would help evaluate the scalability of the approach and its performance under more realistic, large-scale scenarios.

Adversarial target modelling: Instead of assuming a non-adversarial target, future work could involve targets that actively evade detection. This would introduce more realistic dynamics and require more advanced planning and prediction strategies.

Increasing environment diversity: Testing the system across a wider range of environments (e.g., cluttered spaces, dynamic obstacles, or different layouts) would improve robustness and provide better insight into real-world applicability.

References

[1] Brian P. Gerkey, Richard T. Vaughan, and Andrew Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th International Conference on Advanced Robotics (ICAR'03)*, pages 317–323, Coimbra, Portugal, June/July 2003.

[2] Geoffrey Hollinger, Athanasios Kehagias, and Sanjiv Singh. Efficient, guaranteed search with multi-agent teams. In *Proceedings of Robotics: Science and Systems (RSS '09)*, July 2009.

[3] Geoffrey Hollinger, Sanjiv Singh, Joseph Djugash, and Athanasios Kehagias. Efficient multi-robot search for a moving target. *International Journal of Robotics Research*, 28(2):201 – 219, February 2009.

[4] Miguel Juliá, Arturo Gil, and Oscar Reinoso. Searching dynamic agents with a team of mobile robots. *Sensors*, 12(7):8815–8831, 2012. ISSN 1424-8220. doi: 10.3390/s120708815. URL `https://www.mdpi.com/1424-8220/12/7/8815`.

[5] Steven Macenski, Francisco Martin, Ruffin White, and Jonatan Ginés Clavero. The marathon 2: A navigation system. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[6] Steven Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66):eabm6074, 2022. doi: 10.1126/scirobotics.abm6074. URL https://www.science.org/doi/abs/10.1126/scirobotics.abm6074.

[7] Francesco Riccio, Emanuele Borzi, Guglielmo Gemignani, and Daniele Nardi. Multi-robot search for a moving target: Integrating world modeling, task assignment and context. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page 1879–1886. IEEE Press, 2016. doi: 10.1109/IROS.2016.7759298. URL https://doi.org/10.1109/IROS.2016.7759298.