arXiv:2503.03509v2 [cs.RO] 11 Sep 2025

Sampling-Based Multi-Modal Multi-Robot Multi-Goal Path Planning

Valentin N. Hartmann*, Tirza Heinle*, Yijiang Huang, Stelian Coros

***Abstract*— In many robotics applications, multiple robots are working in a shared workspace to complete a set of tasks as fast as possible. Such settings can be treated as multi-modal multi-robot multi-goal path planning problems, where each robot has to reach *a set* of goals. Existing approaches to this type of problem solve this using prioritization or assume synchronous task completion, and are thus neither optimal nor complete. We formalize this problem as a single centralized path planning problem and present planners that are probabilistically complete and asymptotically optimal. The planners plan in the composite space of all robots and are modifications of standard sampling-based planners with the required changes to work in our multi-modal, multi-robot, multi-goal setting. We validate the planners on a diverse range of problems including scenarios with various robots, planning horizons, and collaborative tasks such as handovers, and compare the planners against a suboptimal prioritized planner.**

Videos and code for the planners and the benchmark is available at `https://vhartmann.com/mrmg-planning/`.

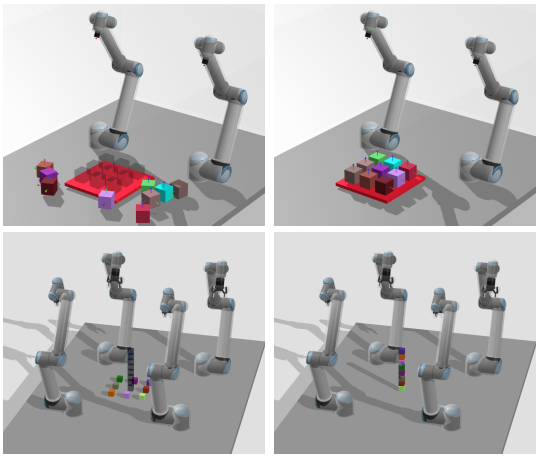


Fig. 1: Examples of problems with their initial state (left) and goal state (right): (top) two robots cooperating to reorient boxes, (bottom) four robots stack boxes on top of each other.

I. Introduction

As adoption of robots increases and simple tasks become more and more automated, it will be increasingly important to deploy solutions that are not only able to work longer and cheaper but are also competitive in throughput with humans. In order to achieve this, in many cases, multiple robots need to be used and effectively coordinated in the same workspace: Enabling motion planning for multiple tasks with multiple robots is crucial in order to maximize the usefulness of robots in industrial settings. While workcells with multiple robots exist, the robots typically act independently from each other in order to simplify the programming and avoid dealing with robot-robot interactions.

Most work in continuous multi-robot planning is focusing on single-goal settings where all robots start moving at the same time and reach their respective goals simultaneously [1]–[3]. Conversely, in most real use cases, multiple robots need to do multiple tasks in sequence, e.g., welding multiple points, or picking and placing multiple things after another in order to sort objects. Even when only considering a single pick and place task per robot, each robot needs to reach two goals: The pick, and the place location. Since the robots act in the same environment in these scenarios and thus possibly block each other from doing their tasks, we can not formulate the problem at hand as a *sequence of path planning problems*, but need to solve the multi-robot multi-goal planning problem if we want to find an optimal solution.

Multi-modal planning can be seen as multi-goal planning problem: Multi-modal path planning [4]–[7] finds paths through sequences of *modes*, i.e., through variations of a continuous configuration space that occur, e.g., by grasping an object, or moving an object in the workspace. This means that in each *mode* a different set of constraints is active. Most work on multi-modal path planning for robots only considers single-robot settings.

On the other hand, multi-agent path finding (MAPF) [8], [9] deals with high numbers of agents, but typically considers grid-like 2d environments with a homogeneous group of disk robots, as found, e.g., in warehouses. MAPF was extended to consider multiple goals in the multi-agent pickup and delivery (MAPD) setting [10]. These approaches do not transfer easily to changing environments and continuous configuration spaces.

Our contributions in this paper are

- a formalization of multi-modal, multi-robot, multi-goal path planning in continuous spaces,
- adaptions of probabilistically complete and almost-surely asymptotically optimal planners to this setting.

We compare the planners to a prioritized planner on a multi-modal, multi-robot, multi-goal motion planning benchmark containing 21 different base-scenarios with up to 74 sub-goals, that can be further adjusted in difficulty by changing the number of robots and tasks, which we open-source.

II. Related Work

Most multi-robot planning work is focusing either on the setting where all robots share the same work and configuration space, or on solving single goal problems. It is commonly assumed that the environment stays completely static during planning, therefore excluding manipulation planning

All authors are with the Computational Robotics Lab (CRL), ETH Zürich.
* indicates equal contribution.

TABLE I: Comparison of path planning approaches.

Approach	Examples	Multi-Robot	Multi-Goal	Multi-Modal	Continuous	Complete	Opt.
MAPF	[8], [9]	✓				✓	✓
MAPD	[10], [11]	✓	✓			✓	✓
Multi-Robot Planning	[12]–[15]	✓			✓	✓	✓
Multi-Modal Planning	[4], [5]		✓	✓	✓	✓	✓
Prioritized planning	[16]–[18]	✓	✓	✓	✓		
Ours		✓	✓	✓	✓	✓	✓

settings. We group the different research areas and their respective focus in Table I. In the following, we discuss single-goal multi-robot planning and MAPF more in-depth.

A. Continuous Multi-Robot Planning

A simple approach to multi-robot planning is planning in the composite space of all robots [14], and applying standard methods such as RRT(*) [19] or PRM(*) [20] in this higher dimensional space. This approach is not scalable to many robots or high degrees of freedom.

There are various approaches to tackle this limitation. Discrete RRT (dRRT(*) [13], [21]), builds a roadmap for each robot, and combines them to implicitly form a tensor graph. Instead of planning naively on the implicit graph, an RRT-like strategy is proposed to explore this implicit graph. Fast-dRRT [17] applies this approach sequentially on the subproblems to solve multi-goal problems. M* [1] plans in the separate spaces when possible, and plans jointly only where required, i.e., when conflicts between the single-robot plans arise. Similarly, conflict based search (CBS) [22] plans for each robot individually, and introduces constraints to solve conflicts between single agent plans. CBS originates form the MAPF-setting, but can also be applied to continuous multi-robot planning problems [12], [23], [24].

Prioritized planners as proposed in, e.g., [15], [16], [18], [25] achieve more scalable solvers by not requiring completeness and optimality. Similarly, [26] proposes a scalable approach that is based on operator splitting and sampling, and while the approach is complete, it is not optimal.

The prioritized planners can be used to plan for multi-goal problems by incrementally generating a path per agent, and considering it as fixed for the later planning problems as done in [16], [18]. To the best of the authors' knowledge, there are no multi-modal multi-robot multi-goal planners that are complete and optimal in the general setting.

A slightly different approach is taken in [27], where plans for the separate robots are planned, and then - if conflicts between the paths arise - fixed by introducing pauses.

Compared to the discussed works, we are interested in optimally solving multi-robot multi-goal planning problems without any restricting assumptions. Further, we are interested in manipulation, i.e., problems where the environment changes through the actions of the robot.

B. Multi-Agent Path Finding

MAPF [9] generally refers to planning in grid-like 2D environments with disk-robots, as found in warehouses. A common approach to solve this problem is conflict based [22] or priority based search (PBS) [28]. There are many works that improve upon the initial approach, such as improved CBS [29], or bounded suboptimal CBS [30], which scale to 1000s of agents.

MAPD refers to the setting where we do not only intend to go from point A to point B, but traveling between multiple points of interest, as part of a single planning problem [10], [31]. While MAPD can be framed as lifelong MAPF problem, i.e., once an agent has finished its task of the MAPF problem, a new goal is assigned to the robot, these solvers are suboptimal [32]. Multi-label A* [11] (MLA*) extends the classic A* algorithm to a sequence of multiple goals, and is able to optimally solve MAPD problems. Both MAPF and MAPD planners typically assume that the environment remains unchanged over the planning duration.

III. Multi-Robot Multi-Goal Path Planning

Informally, our objective is to find a collision free path for each robot that passes through a sequence of goals that minimizes a cost (e.g., the latest completion time of all robots). Goals might involve multiple robots: A handover of an object between two robots implies a constraint on two robots. A goal could additionally imply a *mode transition*, i.e., a robot grasping something and thus changing the environment for the remaining planning problem.

A. Preliminaries

Before formalizing the problem, we introduce the components that are required: The tasks, the concept of a mode, and the state space that we plan in. We follow the work from Thomason et al. [4], and generalize it to multiple robots. Compared to [4], task sequencing and allocation happens implicitly in the planning.

1) Task: A task S consists of the robots that are assigned to the task, and a set of constraints g[1] that need to be fulfilled to consider the task done. A task can have post-conditions that can alter the scene-graph of the environment, i.e., which objects are linked to each other. As example, consider the task 'robot r_1 grasps object o_1': Here, the goal constraints are that the robot is grasping o_1, and the post-condition is that o_1 is linked to the end-effector of the robot.

We use $s \in \mathcal{S}_R = \mathcal{S}_{r_1} \times ... \times \mathcal{S}_{r_n}$ to denote the *task assignment* of all robots, where $\mathcal{S}_{r_i}$ is the set of tasks that are assigned to robot r_i.

2) Modes: The constraints of a task can be fulfilled by a set of poses, where the chosen pose might affect the environment that we plan in. Consider again the grasping-task: The robot is able to fulfill a grasping constraint by grasping from any side, which changes the collision geometry that we need to consider in the rest of the planning problem and it

[1]Typically, the constraint g is a single goal pose or a goal region, but can also be a more complex constraint such as a grasp constraint.

might influence how the constraints for follow-up tasks can be fulfilled. We use *mode* $m \in \mathcal{M} = \mathcal{S} \times \mathcal{Q}_o$ to refer to the combination of the discrete task assignment s (which implies a scene-graph), and the poses of all movable objects $q_o \in \mathcal{Q}_o$, defined by the relative transformation to their parent-frames.

3) Task Order Specification: A task sequence is induced by an oracle $\mathcal{O}(m) : \mathcal{M} \to \mathcal{S}^*$ which maps the current assignment and state of the environment to the list of possible task assignments $\mathcal{S}^*$. In practice, we describe the ordering of tasks as (partially connected) dependency graph $G = (S, E)$, with the tasks $S \in \mathcal{S}_{r_i}$, and the directed edges between tasks $E \in (\mathcal{S}_{r_i} \times \mathcal{S}_{r_j})$. We require the terminal task to involve all robots. Some examples of how a task sequence can be specified are illustrated in Fig. 2. A (partial) task sequence implies all possible transitions between task assignments (Fig. 3), and thus all logically valid task assignment sequences. We use $\mathcal{T}$ to denote the set of all valid task assignment sequences that bring us from the start to the goal and obey the constraints imposed by the oracle.

4) State Space: We describe a path in the composite space of all robots, all objects, and which tasks are currently active per robot:

$$\mathcal{Q} = \underbrace{\mathcal{Q}_{r_1} \times \cdots \times \mathcal{Q}_{r_N}}_{\mathcal{Q}_R} \times \mathcal{Q}_o \times \underbrace{\mathcal{S}_{r_1} \times \cdots \times \mathcal{S}_{r_N}}_{\mathcal{S}_R}.$$

Here, $\mathcal{Q}_{r_i}$ is the configuration space of robot r_i, and correspondingly, $\mathcal{Q}_R$ is the composite configuration space of all robots; $\mathcal{Q}_o$ is the composite configuration space of all objects. We use $\mathcal{Q}_{\text{free}} \subseteq \mathcal{Q}$ to denote the part of the configuration space that is collision-free.

Clearly, not all degrees of freedom are actuated. We assume that we can only plan for the robots' degrees of freedom directly, and all others need to be influenced indirectly.

B. Problem formulation

Bringing everything together, a multi-modal, multi-robot, multi-goal path planning problem is given by the tuple $(R, \mathcal{Q}_R, q_{\text{start}}, m_{\text{start}}, \mathcal{O})$, where R is the set of robots, $\mathcal{Q}_R$ is the configuration space of the robots, q_{start} is the initial state, m_{start} is the start mode, and $\mathcal{O}$ is the oracle giving us the possible next task assignments from the current mode. In the following, we will use q^{r_i} for the pose of robot r_i.

We want to find a collision free path $\pi(t) : \mathbb{R} \to \mathcal{Q}_R$ and the task assignment sequence $s(t) : \mathbb{R} \to \mathcal{S}_R$, that minimizes the cost function $c(\cdot)$. The path π maps time to the composite robot state $\mathcal{Q}_R$, and the mode sequence $s(t)$ maps time to the task assignment $\mathcal{S}_R$, which together imply the scene-graph at a time, and thus the poses of all objects.

Cost functions: We are often interested in finding the minimum makespan plan. However, purely minimizing makespan can lead to optimal paths that bring undesired side-effects, e.g., containing unnecessary movement. Thus, we consider cost functions of the family

$$c(q_1, q_2) = (1-w) \max_r ||q_1^r - q_2^r||_2 + w \sum\nolimits_r ||q_1^r - q_2^r||_2. \quad (1)$$

where if w is small, we get a minimum makespan optimization problem (which is 'regularized' by the path length), and if w is 1 the total cost is the sum of all robot path lengths.

IV. Planners

We now introduce the necessary adaptions to four sampling-based planners (RRT*, PRM* [33], AIT* and EIT* [34]). We refer the reader to the original works for a detailed explanation of how the algorithms work. In all these planners, the space we plan in is the previously defined one. However, instead of directly sampling the (unactuated) object poses, we sample $x = (q, m)$, where $q \in \mathcal{Q}_R$ is a pose containing all robots' poses and m is a mode that determines the task assignment and all object poses. Generally, the planners then proceed as their standard version would, with an adjusted distance function that accounts for the connectivity between the modes. Before discussing any specific adjustments required in the planners in Section IV-A, we discuss the distance function used in the planners, and the concept of the informed set [35] as well as mode sampling.

1) Distance functions: Similar to [4], we assume that any node in a different mode is inaccessible except through the transition nodes:

$$d(x_1, x_2) = \begin{cases} d_{\text{mode}}(q_1, q_2) & \text{if } m_1 = m_2, \\ 0 & \text{if } q_1 = q_2 \text{ and } m_2 \in \text{succ}(m_1) \\ \infty & \text{otherwise,} \end{cases}$$

where we use $\text{succ}(m)$ to indicate the set of successor modes of mode m. For the in-mode distance any metric can be chosen; the work in [36] discusses distance metrics in multi-robot problems in more depth. For our planners, we choose the maximum of the per-robot L_2 distance, as this performed best in our experiments.

2) Mode sampling: As previously introduced, we sample configurations (q, m) from the combined configuration and mode space. In all planners, we store the list of modes M that we reached so far, and expand it whenever we reach a new mode (i.e., whenever we reach a configuration that fulfills the constraints g of a task). Instead of uniformly sampling from the set M, we maintain the most recently added mode F_{-1} and the set of modes F that we did not yet expand (the frontier). We introduce a mode sampling strategy specifically tailored to our multi-goal multi-modal problem:

$$m \sim \begin{cases} \mathcal{U}(F) & \text{with probability } p(1-\epsilon) \\ \mathcal{U}(F_{-1}) & \text{with probability } p\epsilon \\ \mathcal{U}(M \setminus F) & \text{with probability } 1-p \end{cases}$$

where $p \in [0, 1]$ and $\epsilon \in [0, 1]$ are weighting parameters, and $\mathcal{U}(\cdot)$ denotes a uniform distribution over a finite set. If $p = 1$ and $\epsilon = 1$, we reach a fully *greedy* version, resulting in the most direct progression possible towards the terminal mode.

We refer to the *frontier sampling strategy* if $p > 0.5$ holds. This biases the planner towards the unexpanded modes (i.e., frontiers) compared to the fully greedy approach, while still

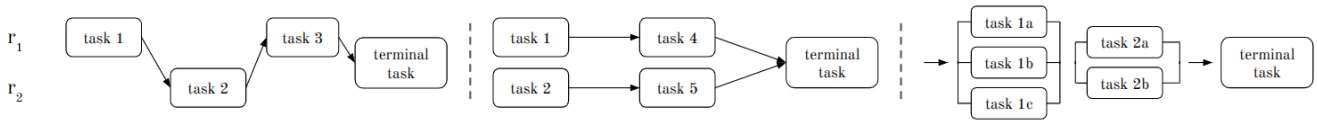


Fig. 2: Examples for the different ways in which a task ordering can be specified: **(left)** As a sequence of tasks **(middle)** as a partially ordered sequence and **(right)** as a set of unassigned and unordered tasks (i.e., the order of task groups 1 and 2 is exchangeable).

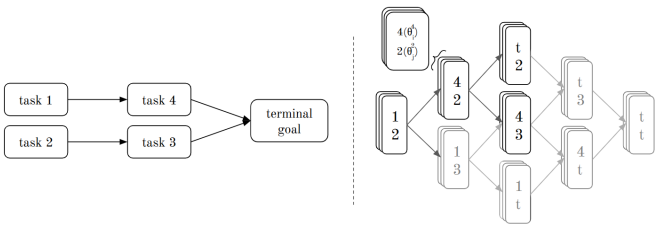


Fig. 3: Example of a mode-graph that is implied by the ordering on the left. Modes with full opacity show an example of how a set of modes can be incrementally reached. Each mode can be reached with different parameterizations θ_i^k (e.g., different grasps), as indicated for mode (4, 2).

maintaining a greedy progression toward the terminal state[2]. For $p = 0.5$ and $\epsilon = 0$, M is sampled uniformly.

3) Locally informed sampling: Once we found a solution to our planning problem, the informed set [35] can be used to restrict the points that we need to consider for planning to the points that can improve the current solution. The informed set in our problem is

$$\mathcal{Q}_{\text{inf}} = \{q \,|\, c_{\text{lb}}(q_{\text{start}}, q) + c_{\text{lb}}(q, Q_{\text{goal}}) \leq c_{\text{best}}\},$$

with Q_{goal} denoting the set of possible goal poses.

In our setting, the optimal paths are typically long compared to the size of the state space. Therefore, the informed set is typically larger than the valid configuration space (similar as in [37]), and thus not helpful.

Given that we do know that a solution needs to pass through some regions in space (the goals that we need to reach), the standard approach of taking the cost Eq. (1) as c_{lb} is not approximating the true path cost well. It is not easy to construct a good lower bound for the path-cost, since the regions that we need to pass through are *subspaces of the full composite space* (i.e., even if we know that robot r_i needs to be at pose q_{g_1} first, robot r_j might be unconstrained, thus leading to a volume that we need to pass through instead of a single point). Additionally, compared to the Euclidean norm as cost, our cost function does not easily allow for direct sampling from the informed set.

To alleviate these issues, we propose *locally informed sampling*: instead of using the start and the goal in Section IV-.3, we randomly sample two points q_i, q_j on the previously found path, and do informed sampling with these points:

$$\mathcal{Q}_{\text{lis}} = \{q \,|\, c(q_i, q) + c(q, q_j) \leq c^{\pi}(q_i, q_j)\},$$

where $c^{\pi}(\cdot)$ is the cost to reach a point on the path along the best found path. We extended this to the sampling of modes by sampling from the set of modes that are in-between the modes that q_i, q_j are in respectively.

[2]A fully greedy strategy is not probabilistically complete in general.

A. Considerations for Multi-Modal, Multi-Goal Planners

We now discuss changes that deviate from the standard approach of the planners that we adapt.

1) Composite RRT:* Compared to standard RRT* [19], [33], we maintain a set of previously reached modes M (i.e., a mode-graph similar to Fig. 3), which is initialized with the start mode. In order to sample configurations in the planner, we uniformly sample poses within the limits of the robots, and sample modes from this set. If we previously found a path, we use locally informed sampling to sample the mode and the pose. Further, we bias the random sampling in the RRT to the goal in a mode, instead of the terminal goal. Other than this, the standard versions of randomly sampling states and connecting to them is used.

We also introduce a bidirectional variant (BiRRT*), which samples goals in each mode (which are used as starts in the next mode), and then runs bidirectional RRT* in each mode.

2) Composite PRM:* The PRM* planner also follows the standard version closely. The main change is that the roadmap is constructed by first sampling a batch of transition nodes (i.e., nodes that fulfill the constraints of a task), which enlarges the set of reached modes and then sampling a batch of nodes in the reached modes, until a goal mode is reached. Then, a standard A* with an edge queue is used for the search over the roadmap.

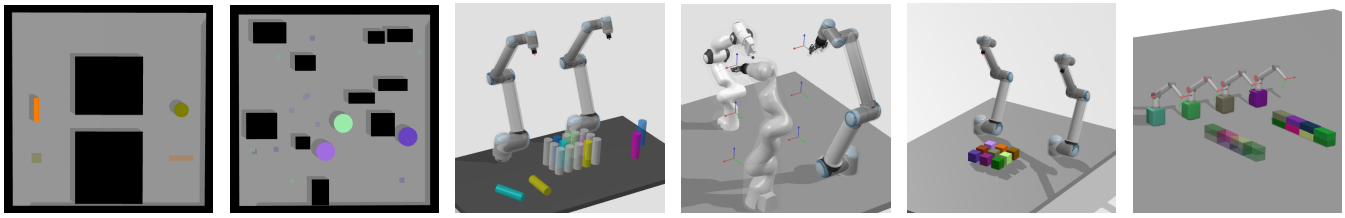
Heuristics: Since A* needs a heuristic in order to be effective, and the euclidean distance is not informative in our setting, we compute a heuristic for the A* search by running a reverse search on the transition nodes to compute a lower bound on the cost to reach a mode. The heuristic is then the sum of the cost to reach a specific transition and the cost to reach the goal from this transition mode:

$$h(x = (q, m)) = \min_i \left[c(q, q_i^{m,\text{trans}}) + c_{\text{rev}}(q_i^{m,\text{trans}})\right] \quad (2)$$

where $q^{m,\text{trans}}$ are the poses belonging to the transition nodes in a mode.

3) Composite IT-planners: AIT* and EIT* [34], [38], [39] use adaptive problem-specific heuristics to focus the search on promising regions. These heuristics are computed and refined through an *asymmetric bidirectional search* to obtain more informative exploration and improved solution quality.

Search: To explore the graph, AIT* and EIT* perform their respective *asymmetric bidirectional search* strategies, as described in [34]. Although each planner utilizes distinct heuristics, both follow an identical approach to handle transitions: When the forward search reaches a transition node, it



(a) 2D hallway. (b) 2D random. (c) Bottle insertion. (d) Multi waypoints. (e) Rearrangement. (f) Mobile assembly.

Fig. 4: The initial states of a selection of problems that are available in the benchmark. The poses that have to be reached by the robots or objects are drawn with lower opacity or indicated with a marker.

only expands to valid successor modes, enforcing forward-directed connectivity; the reverse search applies the same logic in opposite direction (i.e., expanding towards previous modes). Both planners need an admissible a priori cost-to-come (i.e., from the start to the state x) heuristic - we use a modified version of Eq. (2) with the lower bound cost to reach a mode computed from the start node instead of the goal (i.e., replacing c_{rev} with the cost computed by running a forward search only over the transition nodes of the previous modes). This heuristic is a trade-off between heuristic accuracy and computational efficiency.

The reverse search in EIT* incorporates heuristics approximating the computational effort required to find a valid path. We define the computational effort as the collision checking effort. Following this definition, the effort-to-come is computed analogously to the cost-to-come heuristic in Eq. (2), where cost terms are substituted by effort.

B. Postprocessing

When a path is found in any of the planners, we run a postprocessing step to rapidly improve the path cost, which decreases the volume of the informed set. Our approach is a minor modification of partial shortcutting [40]: The main idea is that we do not shortcut all dimensions of the path at once, but select a subset of dimensions from the composite space, and try to shortcut only this subset, while keeping the rest of the plan the same. Instead of uniformly sampling from all possible subsets, we notice that in practice it is more efficient to choose a robot, and then shortcut all dimensions of this robot at once. This enables improving the unconstrained parts of the path (i.e., the parts that do not influence a mode-transition), while still keeping the simplicity of the shortcutting approach.

C. Proof sketches

All planners uniformly sample all transitions, and sample uniformly within the modes, therefore eventually discovering all possible modes. Within the modes, both planners work as their respective base version, and are therefore probabilistically complete, and will eventually connect to the transitions. If a path exists, the discovered transitions will lead to the terminal mode, which contains the goal pose. Thus, both planners will find a path if one exists, and are therefore probabilistically complete.

The proof sketch for almost-sure asymptotic optimality follows a similar logic: The search that we run in the PRM and EIT/AIT-planners is optimal, and therefore, as the number of uniform-random samples approaches infinity, we converge to the optimal paths within the modes, and to the optimal mode-transitions. Similarly, the rewiring-step in the RRT-planners is the same as in RRT*, and thus inherits the optimality guarantees from RRT* even over multiple modes.

V. Experiments

The main contribution of this work is the formulation of the multi-modal, multi-robot, multi-goal planning problem and the adaption of standard sampling-based planners to the formulation. We implement this formulation in a variety of base scenarios in an open-source benchmark featuring diverse sets of robots and tasks. The 21 base scenarios range from simple settings where the optimal solution path is known to help validate properties of the planner, to more complex scenarios with up to 5 robots (with a total of 30 degrees of freedom), and up to 74 goals. For many of the problems, there are versions with a complete task ordering and assignment, and others that are either unordered, partially ordered, or where tasks are unassigned. We show a selection of scenarios in Fig. 4.

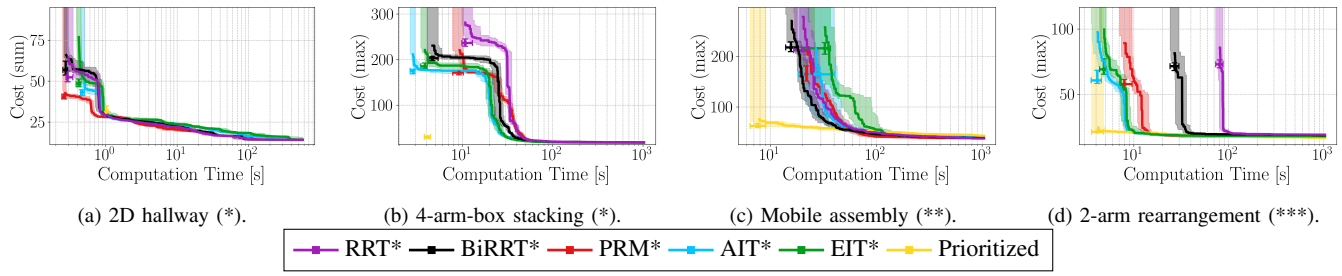
We provide proof-of-concept implementations of the planners in Python, while the computationally expensive parts, i.e., collision checking and forward kinematics use a more performant backend, which can be easily replaced, allowing for, e.g., GPU parallelization if the backend supports it. Currently, the benchmark supports implementations of environments in Pinocchio [41] and RAI[3].

A. Experiments

We present a selection of experiments to showcase the different types of scenarios and task sequence specifications and compare the optimal planners to a prioritized planner [42] on them[4]. We report results for four representative scenarios: (a) A 2D scenario with two robots that is similar to the classic wall-gap, where the robots have to switch positions and go back to their start poses (6D conf. space, 3 subgoals, Fig. 4a), (b) a scenario with 4 robot arms where 8 boxes have to be stacked (24D conf. space, 17 subgoals, Fig. 1), (c) a scenario involving 4 mobile manipulators rearranging a wall (24D conf. space, 17 subgoals, Fig. 4f),

[3] https://github.com/MarcToussaint/robotic

[4] Synchronized planners and standard MAPF planners are not applicable to these problems, and are thus not part of the benchmark.



(a) 2D hallway (*). (b) 4-arm-box stacking (*). (c) Mobile assembly (**). (d) 2-arm rearrangement (***).

Fig. 5: Evolution of median cost over time along with the 95% non-parametric confidence intervals over 50 runs. We also show the median initial solution time and cost using the square with error bars. (*) indicates a fully given task sequence, (**) indicates a partial task ordering, (***) indicates unassigned and unordered tasks. Note that the prioritized planner only acts as anytime planner and improves its cost in the (**) and (***) settings, where multiple different sequences can be generated and planned for.

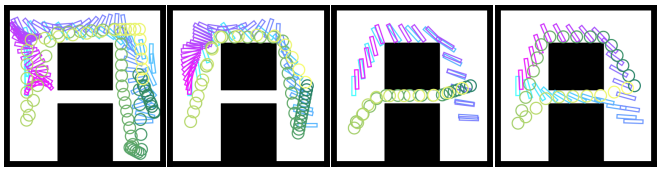


Fig. 6: The path at different iterations in the process, from the initial to the optimized solution. Color progression indicates time from start to end (rectangle , disk) .

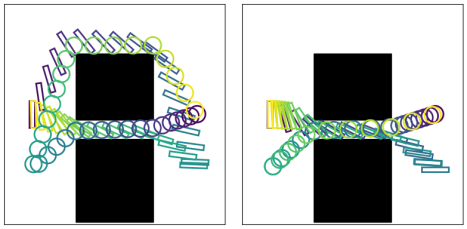


Fig. 7: The optimal paths when using a max **(left)** or sum **(right)** cost function in the 2D hallway scenario, where the robots have to reach a goal on the other side and return. Color indicates time from purple (start) to yellow (end).

and (d) a scenario with 2 robot arms that have to collaborate with handovers to rearrange and reorient 9 boxes (12D conf. space, 28 subgoals, Fig. 4e). In these scenarios, the task sequence is fully determined for scenario (1) and (2), *partially given* for scenario (3) and *not assigned or ordered* for scenario (4). To deal with partially ordered problems in the prioritized planner, we generate valid random sequences, and plan for those until the time runs out.

We use the cost function with $w = 0.01$, i.e., the min-makespan problem with a small path-length regularization in scenarios (b)-(4d), and $w = 1$ in scenario (a). We report the median solution costs over 50 runs with different random seeds. We use a frontier mode sampling strategy until a path is found, and then switch to uniform mode sampling in all planners if not stated otherwise.

The experiments were run with Python 3.10 on a Ryzen 7 5800X (8-core, 4'491 MHz) and 32 GB RAM.

B. Results

Figure 5 shows the resulting cost evolution plots. Figure 6 shows the evolution of a path for the hallway problem.

Comparing the optimal planners introduced in this work and the prioritized planner shows that the suboptimal planner finds reasonable solutions (between 20% and 70% over the optimal solution, but much better than the initial solutions) quickly. As expected, the optimal planners converge to solutions with lower cost, but do so at a later time. Surprisingly, the planners that plan in composite space find initial solutions faster than the suboptimal planner in some settings. We explain this mainly with the fact that treating the previously planned paths as fixed sometimes constrains the planning space such that the planning problem becomes considerably harder compared to the setting where all agents can be moved around freely.

Convergence in complex problems: In scenarios (c) and (d) the planners do not reliably converge to the same solution at the end of their runtime but remain 5%-10% above the lowest found solution, showing that some of the found solutions are not optimal. This is because the problem space is extremely large due to the discrete choice of task assignment and ordering in addition to the high dimensionality of the continuous space. Thus, while the planners are theoretically optimal, in practice, we find that the planners tend to get stuck in a local optimum and do not switch the discrete assignment and ordering often. Note that in scenario (c), the median final cost of the prioritized planner is lower than the median final cost of some of the composite space planners. This is due to the better exploration of different sequences.

Influence of the cost function: We illustrate the difference of the sum-cost and the max-cost on the hallway example in Fig. 7: Both robots go through the wall gap twice when using the sum-cost, and do not make use of the passage at the top, compared to the (regularized) max-cost, where they make use of the passage at the top.

C. Ablations

We present ablations to explain the design decisions that deviate from the standard versions of the planners. In the following, we do not show the results for all planners as the effects are similar for most, and would crowd the figures.

1) Locally informed sampling: Only using standard informed sampling in the problems we consider here is insufficient, as the informed set is large compared to the configuration space, and thus effectively equivalent to uniform

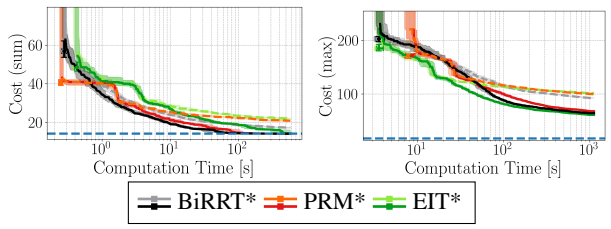


Fig. 8: Local (solid) and global (dashed) informed sampling on the 2D hallway **(left)** and the 4-arm box-stacking **(right)** scenarios. The dashed blue line is the best cost of the non-ablated planners.

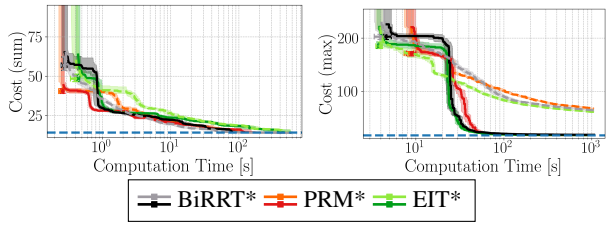


Fig. 9: Planners with (solid) and without (dashed) shortcutting on the 2D hallway **(left)** and the 4-arm box-stacking **(right)** scenarios. All planners are using locally informed sampling. The dashed blue line is the best cost of the non-ablated planners.

sampling. The planners do not use shortcutting in order to better highlight the difference between the ablations. The planners using locally informed sampling outperform the planners that do use standard informed sampling (Fig. 8).

2) Shortcutting in the planner: Shortcutting a path after it was found leads to faster convergence than using only informed sampling (Fig. 9) while not affecting the optimality or completeness properties of the planners. This effect is more pronounced in high-dimensional scenarios. If we only do shortcutting, but not locally informed sampling, the planner does not converge to the same solution.

3) Mode sampling strategy: Employing a *frontier sampling strategy* during the initial search significantly accelerates the discovery of a first solution compared to uniform sampling (Fig. 10) in high dimensional, long horizon scenarios. While not shown here, a greedy sampling strategy gives a further significant speedup for the initial solution but does sacrifice completeness if not done carefully.

VI. Discussion & Limitations

Due to planning in the composite space and treating the problem as a single big planning problem, the planners do not scale well with the number of robots (due to the increase in state-space dimensions) and they do not scale well with the number of goals, as can be seen in the slower convergence in the high dimensional problems. However, we believe that formulating the problem in composite space helps understand how choices such as prioritization affect the continuous space, and this deeper understanding can enable the design of better planners.

We introduce a problem formulation and planners that can deal with all types of task specification. However, we believe that there is a better way to approach the problem than a brute-force search, which the sampling-based planners

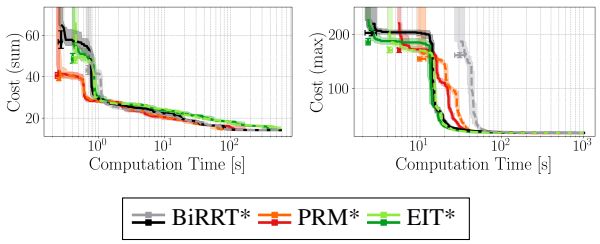


Fig. 10: Frontier (solid) and uniform (dashed) initial mode sampling strategies on the 2D hallway **(left)** and 4-arm box-stacking **(right)** scenarios.

effectively do here. Despite these limitations, we find that the planning in composite space is likely good enough for many complex industrial settings, where the number of used robots does not go much beyond four robots. Furthermore, with our initial mode sampling strategy defined in Section IV-.2, we were able to rapidly reduce the decision space by selecting a feasible mode sequence early in the search, allowing us to find an initial solution quickly.

VII. Conclusion

We presented a formalization of the multi-modal, multi-robot, multi-goal motion planning problem, and open-source the benchmark we developed, containing diverse problems reaching from simple environments with short goal sequences, to relatively long horizon problems requiring coordination of multiple robots. We adapt multiple standard sampling-based planners that are probabilistically complete and asymptotically optimal to this type of problem. We also benchmarked a prioritized planner against these optimal planners and show that the solutions found by such a suboptimal planner can be competitive in some settings.

There are some clear improvements possible, such as exploiting the multi-robot structure of the planning problems by adapting, e.g., dRRT to our problems. Further, the nature of the mode families [43], or the multi-query-like structure of the problem [44] could be used. We also believe that receding horizon-planners are an interesting avenue of research in order to transition this purely offline planning approach to real-world execution.

There are many extensions to the chosen problem formulation: The formulation and planners we propose supports extension of the transition logic to full multi-robot task and motion planning by changing the task-assignment-oracle. In the future, we also want to support constrained multi-robot planning, or kinodynamic planning.

Acknowledgments

The authors thank Miguel Zamora for insightful discussions and comments on both text and illustrations.

References

[1] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artif. intell.*, vol. 219, pp. 1–24, 2015.

[2] R. Shome and L. E. Kavraki, "Asymptotically optimal kinodynamic planning using bundles of edges," in *Proc. of the IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2021, pp. 9988–9994.

[3] S. Lin, A. Liu, J. Wang, and X. Kong, "A review of path-planning approaches for multiple mobile robots," *Machines*, vol. 10, no. 9, p. 773, 2022.

[4] W. Thomason, M. P. Strub, and J. D. Gammell, "Task and Motion Informed Trees (TMIT*): Almost-surely asymptotically optimal integrated task and motion planning," *IEEE Robot. and Automat. Lett. (R-AL)*, vol. 7, no. 4, pp. 11 370–11 377, 2022.

[5] K. Hauser, "Task planning with continuous actions and nondeterministic motion planning queries," in *Proc. of AAAI Workshop on Bridging the Gap between Task and Motion Planning*, 2010.

[6] P. Englert, I. M. R. Fernández, R. K. Ramachandran, and G. S. Sukhatme, "Sampling-Based Motion Planning on Sequenced Manifolds," in *Proc. of Robotics: Science and Systems (R:SS)*, 2021.

[7] P. S. Schmitt, W. Neubauer, W. Feiten, K. M. Wurm, G. V. Wichert, and W. Burgard, "Optimal, sampling-based manipulation planning," in *Proc. of the IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2017, pp. 3426–3432.

[8] J. Yu and S. M. LaValle, "Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics," *IEEE Trans. Robot.*, vol. 32, no. 5, pp. 1163–1177, 2016.

[9] R. Stern, "Multi-agent path finding–an overview," *Artificial Intelligence: 5th RAAI Summer School, Tutorial Lectures*, pp. 96–115, 2019.

[10] H. Ma, J. Li, T. S. Kumar, and S. Koenig, "Lifelong multi-agent path finding for online pickup and delivery tasks," in *Int. Conf. on Autonomous Agents and MultiAgent Syst. (AAMAS)*, ser. AAMAS '17, 2017, p. 837–845.

[11] F. Grenouilleau, W.-J. Van Hoeve, and J. N. Hooker, "A multi-label A* algorithm for multi-agent pathfinding," in *Int. Conf. on Autom. Plan. and Sched.*, vol. 29, 2019, pp. 181–185.

[12] A. Moldagalieva, J. Ortiz-Haro, M. Toussaint, and W. Hönig, "db-cbs: Discontinuity-bounded conflict-based search for multi-robot kinodynamic motion planning," in *Proc. of the IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2024, pp. 14 569–14 575.

[13] R. Shome, K. Solovey, A. Dobson, D. Halperin, and K. E. Bekris, "dRRT*: Scalable and informed asymptotically-optimal multi-robot motion planning," *Autonomous Robots*, vol. 44, no. 3-4, pp. 443–467, 2020.

[14] G. Sanchez and J.-C. Latombe, "Using a PRM planner to compare centralized and decoupled planning for multi-robot systems," in *Proc. of the IEEE Int. Conf. Robot. Automat. (ICRA)*, vol. 2, 2002, pp. 2112–2119 vol.2.

[15] A. Orthey, S. Akbar, and M. Toussaint, "Multilevel motion planning: A fiber bundle formulation," *Int. J. of Robot. Research*, vol. 43, no. 1, pp. 3–33, 2024.

[16] V. N. Hartmann, A. Orthey, D. Driess, O. S. Oguz, and M. Toussaint, "Long-horizon multi-robot rearrangement planning for construction assembly," *IEEE Trans. Robot.*, 2022.

[17] A. Solano, A. Sieverling, R. Gieselmann, and A. Orthey, "Fast-dRRT*: Efficient Multi-Robot Motion Planning for Automated Industrial Manufacturing," *arXiv preprint arXiv:2309.10665*, 2023.

[18] J. Chen, J. Li, Y. Huang, C. Garrett, D. Sun, C. Fan, A. Hofmann, C. Mueller, S. Koenig, and B. C. Williams, "Cooperative task and motion planning for multi-arm assembly systems," *arXiv preprint arXiv:2203.02475*, 2022.

[19] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," *Algorithmic and computational robotics*, pp. 303–307, 2001.

[20] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Proc. of the IEEE Int. Conf. Robot. Automat. (ICRA)*, vol. 12, no. 4, pp. 566–580, 1996.

[21] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning," *Int. J. of Robot. Research*, vol. 35, no. 5, pp. 501–513, 2016.

[22] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artif. intell.*, vol. 219, pp. 40–66, 2015.

[23] I. Solis, J. Motes, R. Sandström, and N. M. Amato, "Representation-optimal multi-robot motion planning using conflict-based search," *IEEE Robot. and Automat. Lett. (R-AL)*, vol. 6, no. 3, pp. 4608–4615, 2021.

[24] J. Kottinger, S. Almagor, and M. Lahijanian, "Conflict-Based Search for Multi-Robot Motion Planning with Kinodynamic Constraints," in *Proc. of the IEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2022, pp. 13 494–13 499.

[25] J. P. Van Den Berg and M. H. Overmars, "Prioritized motion planning for multiple robots," in *Proc. of the IEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*. IEEE, 2005, pp. 430–435.

[26] K. Okumura and X. Défago, "Quick multi-robot motion planning by combining sampling and search," *arXiv:2203.00315*, 2022.

[27] G. Han, J. Park, and C. Nam, "Stop-N-Go: Search-based Conflict Resolution for Motion Planning of Multiple Robotic Manipulators," *arXiv preprint arXiv:2410.07606*, 2024.

[28] H. Ma, D. Harabor, P. J. Stuckey, J. Li, and S. Koenig, "Searching with consistent prioritization for multi-agent path finding," in *Proc. of the AAAI Conf. on artif. intell.*, vol. 33, no. 01, 2019, pp. 7643–7650.

[29] E. Boyarski, A. Felner, R. Stern, G. Sharon, O. Betzalel, D. Tolpin, and E. Shimony, "ICBS: The improved conflict-based search algorithm for multi-agent pathfinding," in *Int. Symp. on Combinatorial Search*, vol. 6, no. 1, 2015, pp. 223–225.

[30] J. Li, W. Ruml, and S. Koenig, "EECBS: A bounded-suboptimal search for multi-agent path finding," in *Proc. of the AAAI Conf. on artif. intell.*, vol. 35, no. 14, 2021, pp. 12 353–12 362.

[31] O. Salzman and R. Stern, "Research challenges and opportunities in multi-agent path finding and multi-agent pickup and delivery problems," 2020, pp. 1711–1715.

[32] G. Mouratidis, B. Nebel, and S. Koenig, "Fools Rush in Where Angels Fear to Tread in Multi-Goal CBS," in *Int. Symp. on Combinatorial Search*, vol. 17, 2024, pp. 243–251.

[33] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. of Robot. Research*, vol. 30, no. 7, pp. 846–894, 2011.

[34] M. P. Strub and J. D. Gammell, "Adaptively Informed Trees (AIT*) and Effort Informed Trees (EIT*): Asymmetric bidirectional sampling-based path planning," *Int. J. of Robot. Research*, vol. 41, no. 4, pp. 390–417, Apr. 2022.

[35] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *Proc. of the IEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 14–18 Sept. 2014, pp. 2997–3004.

[36] A. Atias, K. Solovey, O. Salzman, and D. Halperin, "Effective metrics for multi-robot motion-planning," *Int. J. of Robot. Research*, vol. 37, no. 13-14, pp. 1741–1759, 2018.

[37] M. Faroni, N. Pedrocchi, and M. Beschi, "Accelerating sampling-based optimal path planning via adaptive informed sampling," *Autonomous Robots*, vol. 48, no. 2, pp. 1–12, 2024.

[38] M. P. Strub and J. D. Gammell, "Adaptively Informed Trees (AIT*): Fast asymptotically optimal path planning through adaptive heuristics," in *Proc. of the IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2020, pp. 3191–3198.

[39] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *Proc. of the IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2015, pp. 3067–3074.

[40] R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *Int. J. of Robot. Research*, vol. 26, no. 8, pp. 845–863, 2007.

[41] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The Pinocchio C++ library – A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *IEEE Int. Symp. on Syst. Integrations (SII)*, 2019.

[42] V. N. Hartmann and M. Toussaint, "Towards computing low-makespan solutions for multi-arm multi-task planning problems," in *Int. Conf. on Autom. Plan. and Sched.: Planning and Robotics Workshop (RobPlan)*, 2023. [Online]. Available: https://arxiv.org/abs/2305.17527

[43] Z. Kingston, C. Chamzas, and L. E. Kavraki, "Using experience to improve constrained planning on foliations for multi-modal problems," in *Proc. of the IEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2021, pp. 6922–6927.

[44] V. N. Hartmann, M. P. Strub, M. Toussaint, and J. D. Gammell, "Effort informed roadmaps (EIRM*): Efficient asymptotically optimal multiquery planning by actively reusing validation effort," in *Int. Symp. of Robot. Research*, 2022.