Current Robotics Reports (2021) 2:85–94
https://doi.org/10.1007/s43154-021-00043-8

GROUP ROBOTICS (M GINI AND F AMIGONI, SECTION EDITORS)

Roadmaps for Robot Motion Planning with Groups of Robots

Rahul Shome[1]

Accepted: 14 January 2021 / Published online: 24 February 2021

Abstract

Purpose of Review Autonomous robotic systems require the core capability of planning motions and actions. Centralized motion planning exhibits significant challenges when applied to multi-robot problems. Reasoning about groups of robots typically cause an exponential increase in the size of the search space that an algorithm has to explore. Moreover, each robot by itself might be an articulated mechanism with a large number of controllable joints, or degrees of freedom which can pose its own difficulties in planning. Roadmaps have been a popular graph-based method of representing the connectivity of valid motions in such large search spaces including specialized variants for multi-robot motion planning.

Recent Findings This article primarily covers recent algorithmic advances that are based on roadmaps for motion planning, with specific optimizations necessary for the multi-robot domain. The structure of the multi-robot problem domain leads to efficient graphical decomposition of the problem on roadmaps. These algorithms provide some desired theoretical properties of being guaranteed to find a solution, as well as optimality of the discovered solution. Extensions to richer planning applications are also discussed.

Summary The design of efficient multi-robot planning algorithms like the roadmap-based ones discussed in this article provides the cornerstone for the deployment of large-scale multi-robot teams to solve real-world problems.

Keywords Multi-robot motion planning · Roadmaps · Motion planning · dRRT

Introduction

Robots form the key embodiment of artificial intelligence. Capable robots rely on efficient algorithms dictating how they should move and operate. Beyond controlling a single robot, a key question becomes how we can make effective use of robotic teams. Having multi-robot systems should allow roboticists to solve more problems faster. Industrial automation provides application domains where a large number of robots are already in use in warehouses and fulfillment centers. The development of aerial robot technology has yielded impressive demonstrations involving large groups of such drones. The push towards autonomous cars is poised to introduce large-scale fleets into road networks. Despite the enormous possibilities that multi-robot teams possess, they simultaneously exhibit significant computational challenges. Such a motivating illustration is shown in Fig. 1.

Each robot in a group can have its own objective. Individual robots will have their own optimal solution path to reach their goal. When such a robot exists in multi-robot teams, it can happen that the robots obstruct each other and have to coordinate in order to ensure every robot reaches its goal. Consider a situation where two robots find their individual desired optimal motion crossing the same narrow corridor from opposite sides as shown in Fig. 2.

Decoupled approaches [1–7] reason about the robots separately. These approaches solve subproblems typically corresponding to each robot and combine these solutions. Guarantees in discrete domains assume inherent decoupling [6]. Methods focusing on dynamical systems [4] and control-based methods [8] can more readily scale because of the efficiency of decentralized reasoning, but typically trade off theoretical guarantees and quality assurances. Multi-robot path planning can also be cast as a problem of

This article belongs to the Topical Collection on *Group Robotics*

✉ Rahul Shome
rahul.shome@rice.edu

[1] Department of Computer Science, Rice University, Houston, TX, USA

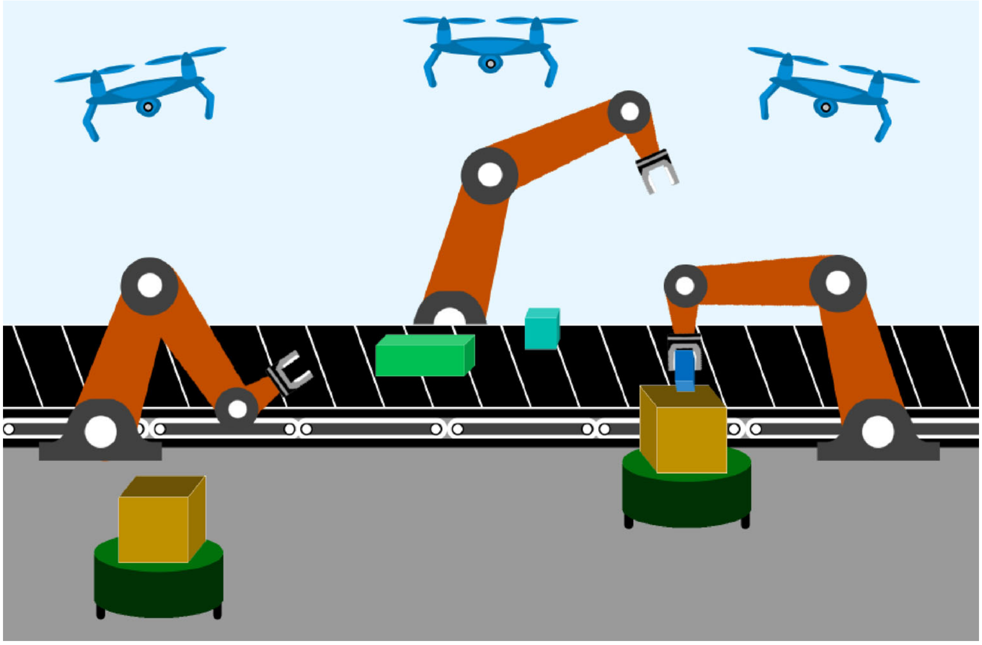

Fig. 1 Teams of robots can be very useful if they can work together. The challenge lies when they have to closely coordinate. The image shows teams of aerial robots, robotic arms, and mobile robots

learning policies for each robot [7]. Nonetheless, decoupled techniques exhibit inherent limitations in general problem domains due to lack of centralized information.

As an illustration, consider a greedy team in Fig. 2 that is liable to get stuck in the middle of the corridor. A decoupled approach can make one of the robots assume priority and move before the other robot can move. Such a sequential execution increases the time taken and can weaken the benefits of using multiple robots. Another option lets one of the robots assume priority and makes the other robot avoid the prioritized robot, either by changing its speed along the same path or resorting to alternate paths. While being relatively straightforward strategies, which are faster, they are liable to cause situations where a solution might not be found or the discovered solution has poor quality. This motivates centralized approaches where all the robots in the team work together to plan to reach the objectives of the team *as a whole*, i.e., centralized strategies [9–13]. A toy problem instance is shown in Fig. 2.

A roadmap is a graph of valid motions of a robotic system. This article focuses specifically on centralized motion planning techniques for multi-robot problems using roadmaps. The article first sets up the formal definition of the problem. A key focus is laid on the efficient, general techniques that leverage the structure of the multi-robot problem in what are called *tensor roadmaps*, and the algorithms designed therewith [14••, 15••]. Other simplified models [16••] of the problem, including extensions to other

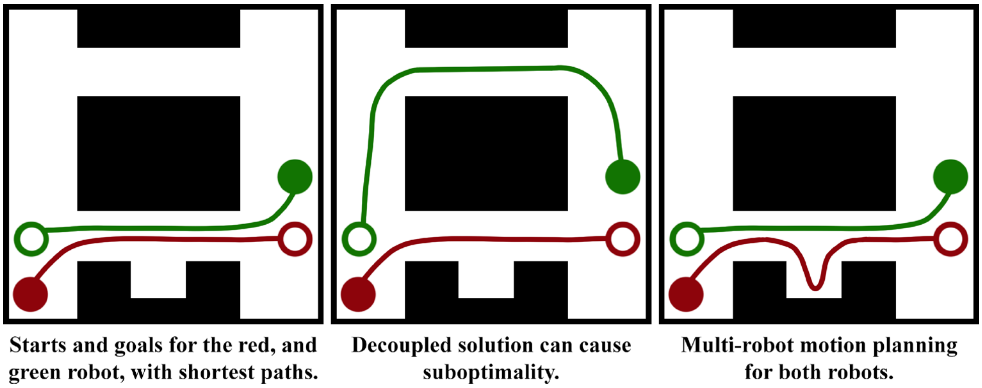
Starts and goals for the red, and green robot, with shortest paths.
Decoupled solution can cause suboptimality.
Multi-robot motion planning for both robots.

Fig. 2 *(Left)* A toy problem is demonstrated in the first figure. A red and green disk robots have to reach their goals (denoted by the empty circles). The black regions are obstacles. The individual shortest paths show being greedy causes a deadlock in the middle of the lower corridor. *(Middle)* A decoupled solution can make the green robot avoid the red path. This causes suboptimal behavior, which is worse in terms of the multi-robot solution. *(Right)* The shortest solution for both the robots involves the robots coordinating with each other within the lower corridor using coordinated multi-robot motion planning applications, are briefly discussed. The article describes the core concepts behind state-of-the-art applications of roadmaps to multi-robot motion planning.

Fundamentals of Multi-robot Motion Planning

A motion planning problem involves a *workspace* $\mathcal{W} \subset SE(3)$ within which resides the physical geometries of the robotic system r. The degrees of freedom of the robotic systems denote all the controllable aspects of the robot—for instance *X,Y* coordinates of a freely moving robot on a plane; *X*, *Y*, *Z*, *roll*, *pitch*, *yaw* of an aerial robot; or the set of joints of a robotic arm. Controlling the values of any of these degrees of freedom changes how the physical geometries of the robot exist in the workspace. The number of degrees of freedom d creates a d-dimensional space [17]. This is called a *configuration space* $\mathcal{Q} \subset \mathbb{R}^d$, which is typically assumed to be Euclidean. A point within the configuration space, referred to as a configuration $x \in \mathcal{Q}$, fully specifies the values of every controllable degree of freedom of the robot. Some of these points might describe the robot's geometries *colliding* with obstacles or unsafe regions in the workspace. Such configurations are invalid, and their collection represents the invalid subset of the configuration space $\mathcal{Q}_{\text{obs}} \subset \mathcal{Q}$. The valid subset is denoted by $\mathcal{Q}_{\text{free}} = \mathcal{Q} \setminus \mathcal{Q}_{\text{obs}}$ and represents all the *collision-free* configurations of the robot.

The problem starts with the robotic system at an initial configuration $x_{\text{init}} \in \mathcal{Q}_{\text{free}}$. The objective of the robot is denoted by a goal configuration $x_{\text{final}} \in \mathcal{Q}_{\text{free}}$.

A geometric path consists of a sequence of configurations parameterized by time. This is typically denoted by a mapping $\pi : [0, T] \rightarrow \mathcal{Q}$ where π is the path and maps configurations to a time range 0 to T. $\pi(t)$ represents the configuration along π at time t.

A *feasible motion planning solution* is one that starts at the initial configuration, ends at the goal configuration, and passes solely through valid configurations along the way, i.e.,

$$\begin{aligned}\pi_{\text{feasible}}(0) &= x_{\text{init}}, \pi_{\text{feasible}}(T)\\ &= x_{\text{final}}, \pi_{\text{feasible}}(t) \in \mathcal{Q}_{\text{free}} \forall t \in [0, T].\end{aligned}$$

Definition 1 (Feasible Motion Planning) An instance of a feasible motion planning problem is denoted by a tuple $(\mathcal{Q}_{\text{free}}, x_{\text{init}}, x_{\text{final}})$. A solution to the problem is denoted by a collision-free valid path π_{feasible} connecting x_{init} to x_{final}.

In multi-robot motion planning, there are R robots. If each of them has d degrees of freedom,[1] the group of robots can be thought of as a single robotic system with Rd degrees of freedom, i.e., describing an Rd-dimensional *multi-robot configuration space* $\mathcal{Q}_{\text{free}} \subset \mathcal{Q}^1_{\text{free}} \times \cdots \times \mathcal{Q}^R_{\text{free}} \subset \mathbb{R}^{Rd}$. Note that as the number of robots increases, the volume of the configuration space increases exponentially. Each superscripted index $[1, 2 \cdots R]$ denotes the valid configuration space for the corresponding robot. The same superscript convention will be followed for other variables as well. A multi-robot configuration x correspondingly is composed of the individual robot states $x = (x^1, \cdots x^R)$. It should be noted that a valid multi-robot configuration not only avoids collisions with obstacle geometries but also other robots along their respective motions. This makes the problem significantly more challenging that reasoning about each robot and environment obstacles separately. Each multi-robot path also contains the path followed by each constituent robot along it, i.e., $\pi(t) = (\pi^1(t), \cdots \pi^R(t))$. Denote this decomposition by $\pi = (\pi^1, \cdots \pi^R)$. This is shown in Fig. 3.

Definition 2 (Feasible Multi-robot Motion Planning) A multi-robot motion planning problem for R robots is denoted by a tuple:

$$\left(\mathcal{Q}_{\text{free}}, (x^1_{\text{init}}, \cdots x^R_{\text{init}}), (x^1_{\text{final}}, \cdots x^R_{\text{final}})\right).$$

The feasible solution to the problem can be denoted by:

$$\pi_{\text{feasible}} = (\pi^1_{\text{feasible}}, \cdots \pi^R_{\text{feasible}}).$$

A cost can be defined for the solution paths. This is a positive real number assigned to each path. Different applications can require specific cost functions like path length, time of execution, and energy expended. Typical multi-robot metrics over a multi-robot path represents some combination of the individual path metrics. This can be the sum of path lengths or some weighted linear combination. The cost can even be described as the Euclidean path length in the multi-robot configuration space. The most common motivating metrics are the maximum time duration of one of the robot paths (*makespan*). A geometric analog of the same is the maximum of the path lengths.

Definition 3 (Optimal Multi-robot Motion Planning) An optimal motion planning problem is denoted by a tuple

$$\left(\mathcal{Q}_{\text{free}}, (x^1_{\text{init}}, \cdots x^R_{\text{init}}), (x^1_{\text{final}}, \cdots x^R_{\text{final}}), \text{cost}\right).$$

The optimal path π_{opt} is a feasible motion planning solution that minimizes the cost function.

Definition 4 (Roadmap) A roadmap is a graph with n vertices, where each vertex in the vertex set is a valid

[1] In general, the robots can be non-uniform and have different degrees of freedom.

configuration. The edge-set defines each edge as an ordered pair of vertices (x_u, x_v). In a valid roadmap, each edge is also associated with a path $\pi_{u\to v}$ that connects x_u to x_v. Such a connection can be shown as $\overrightarrow{x_u, x_v}$, which is typically a short path connecting through the configuration space.[2] A valid roadmap requires all such edge paths to be valid.

$$\begin{aligned} &\mathcal{G}(\mathcal{V}, \mathcal{E}) \\ \mathcal{V} &= \{x \in \mathcal{Q}_{\text{free}}\},\ |\mathcal{V}| = n \\ \mathcal{E} &= \{e = (x_u, x_v),\ x_u, x_v \in \mathcal{V},\ \pi_{u\to v} \\ &= \overrightarrow{x_u, x_v},\ \pi_{u\to v}(t) \in \mathcal{Q}_{\text{free}} \forall t\} \end{aligned}$$

Later sections of the article will detail different ways such roadmaps can be constructed. The choices that affect the construction include how the vertices are chosen. Not all possible valid edges between the vertices are included in the edge-set, and different strategies carefully choose the subset of such edges to include in the roadmap. However, something common to all these methods is that a motion planning solution can be recovered from a valid roadmap.

A path on the roadmap $\pi_{\mathcal{G}}$ connecting two vertices x_1 and x_L is an ordered sequence of L vertices, pairwise connected by valid edges (x_i, x_{i+1}), and corresponding edge-paths $\pi_{i\to i+1}$.

$$\pi_{\mathcal{G}} = (x_1, x_2 \cdots x_L).$$

Let the $\oplus$ denote the concatenation of paths. Then a path π connecting x_1 and x_L is $\pi = \pi_{1\to 2} \oplus \pi_{2\to 3} \cdots \oplus \pi_{L-1\to L}$. So a path can be reconstructed from a roadmap path. This operation is denoted by the shorthand $\pi = \oplus \pi_{\mathcal{G}}$.

Definition 5 (Multi-robot Motion Planning on a Roadmap) Given a multi-robot motion planning problem:

$$\left(\mathcal{Q}_{\text{free}}, (x^1_{\text{init}}, \cdots x^R_{\text{init}}), (x^1_{\text{final}}, \cdots x^R_{\text{final}}), \text{cost}\right)$$

and a valid roadmap $\mathcal{G}$ that includes the initial and goal configurations as vertices, a feasible solution on the roadmap is a sequence of edges connecting the x_{init} to x_{final} as follows:

$$\begin{aligned} \pi_{\mathcal{G}:\text{feasible}} &= (x_1, x_2 \cdots x_L),\ x_1 = x_{\text{init}}, \\ x_L &= x_{\text{final}},\ (x_i, x_{i+1}) \in \mathcal{E}\ \forall i \in [1, L-1]. \end{aligned}$$

The feasible solution path $\pi_{\text{feasible}} = \oplus \pi_{\mathcal{G}:\text{feasible}}$. The optimal roadmap solution $\pi_{\mathcal{G}:\text{opt}}$ corresponds to the feasible roadmap solution that minimizes the cost function.

It should be noted that a concatenation of edge paths along $\pi_{\mathcal{G}:\text{opt}}$ (represented as $\oplus \pi_{\mathcal{G}:\text{opt}}$) is restricted by definition to connections across vertices in the roadmap. This means that the true optimal cost in $\mathcal{Q}_{\text{free}}$ needs not be the same as the optimal roadmap solution which only involves a set of vertices in $\mathcal{Q}_{\text{free}}$, i.e., $\text{cost}(\pi_{\text{opt}}) \leq \text{cost}(\oplus \pi_{\mathcal{G}:\text{opt}})$.

A roadmap constructed in the multi-robot configuration space will include multi-robot configurations, and roadmap paths will correspondingly express multi-robot solutions (Fig. 3). The objective of multi-robot motion planning on roadmaps is to construct $\mathcal{G}$ using (a) appropriately chosen vertices, and (b) choosing a subset of valid edges, and then (c) efficiently recover solution paths[3] traversing roadmap vertices with a cost sufficiently close to the optimal cost.

Roadmap-Based Multi-robot Motion Planners

This sections highlights some key roadmap-based approaches that have been designed for use in multi-robot motion planning. The underlying structure of a roadmap, and the principle of motion planning over it adhere to the descriptions outlined in Definitions 4 and 5. There are however different ways in which both the construction and representation of the roadmap, and efficient strategies for finding motions over it have been presented to specifically address the centralized multi-robot problem.

Planning Directly on Probabilistic Roadmaps in the Multi-robot Configuration Space

Robots with high degrees of freedom like many-jointed robotic arms can have configuration spaces with complicated topologies, which is not easy to describe exactly. Sampling [18, 19] provides a way to cover such spaces effectively. Probabilistic roadmap methods (PRM) [18] describe a sampling-based motion planning algorithm to deal with high-dimensional planning problems. The *sampling* aspect

[2] Typically these are straight-line interpolations, but in general, the edge only needs a guarantee that the path connects the two vertices x_u and x_v. This is denoted by *steerability* of the system, i.e., the ability to steer to a specific configuration x_v from x_u, and is not a valid assumption for robots with non-trivial dynamics.

[3] Note that the construction phase and the solution recovery phase are distinct. This means that for scenarios where the environment is known beforehand, a roadmap can be constructed and reused for all motion planning problems in the same (or a similar) environment. Conversely, a roadmap optimistically constructed ignoring some or all obstacles can also be reused to recover the solution on the subgraph that is collision-free.

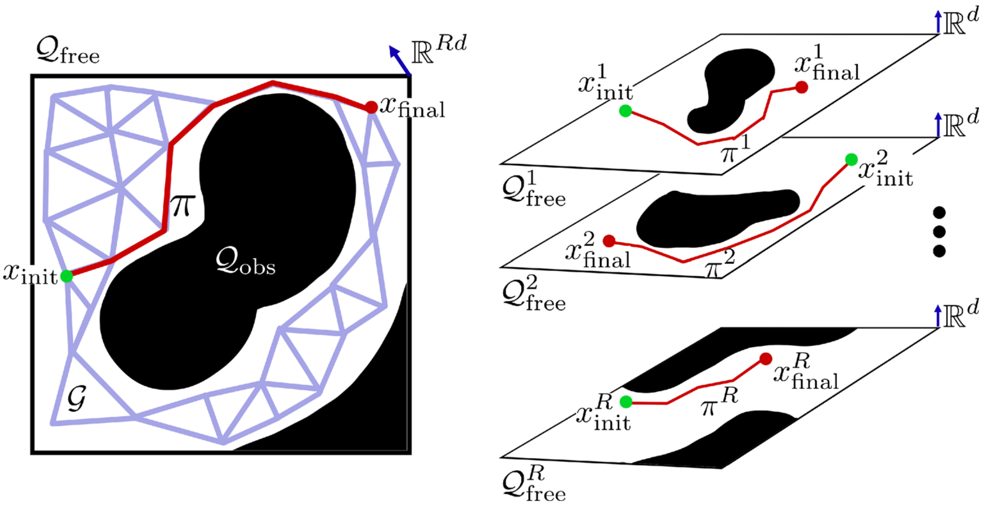

Fig. 3 The image on the left shows a roadmap $\mathcal{G}$ in the valid multi-robot configuration space $\mathcal{Q}_{\text{free}}$ solving the motion planning problem to connect a multi-robot start configuration x_{init} to the goal x_{final} through a valid motion planning solution path π. The right half of the image visualizes the corresponding start and goal pairs for each of the R robots, in their respective d-dimensional $\mathcal{Q}^i_{\text{free}}$, and the component of the solution π^i providing the valid motions for the individual robots

arises from the choice of vertices in $\mathcal{G}$, which are sampled uniformly at random[4] from $\mathcal{Q}_{\text{free}}$.

The edge-set $\mathcal{E}$ in the graph constructed by PRM contain *local connections*, i.e., connections defined by the valid subset of some local neighborhood. Typical connection strategies include (a) connecting each vertex to all other vertices lying within a radial neighborhood defined by a connection radius and (b) connecting each vertex to k nearest neighbors.

As the number of samples n, i.e., the size of $\mathcal{G}$ increases so does the coverage of the space. It has been shown that a PRM is guaranteed to find a solution if one exists asymptotically in n, a property called *probabilistic completeness*. More recent results [20, 21] have additionally shown that careful choices of the connection radius and k as functions of n can guarantee that $\mathcal{G}$ will asymptotically contain roadmap paths that converge in cost to the optimal solution cost. This property is called *asymptotic optimality*. This variant was called PRM*.

A key detail about both these asymptotic properties is that they are closely related to the effectiveness of the random sampling in covering $\mathcal{Q}_{\text{free}}$. The larger $\mathcal{Q}_{\text{free}}$ is, the more samples would be required achieve good coverage of the space.

The formulation of roadmaps is general enough to directly apply to the multi-robot configuration space [22]. This involves sampling all the available Rd degrees of freedom, i.e., randomly situate all the robots, and then connect neighborhoods of such configurations by simultaneously moving all the robots along an edge.

Once a roadmap has been constructed (like the one illustrated in Fig. 3), a solution exists if x_{init} and x_{final} are in the same connected component of $\mathcal{G}$. The number of vertices necessary for $\mathcal{G}$ to have a solution and achieve a good enough solution cost depends upon the volume of $\mathcal{Q}_{\text{free}}$. Say a single robot roadmap displays connectivity and acceptable solutions for $\hat{n}$ samples. To preserve similar properties for each of the robots in the Rd-dimensional space, $\mathcal{O}(\hat{n}^R)$ samples would be necessary in the roadmap constructed in the multi-robot configuration space. Once such a roadmap is constructed, typically heuristic search A* [23] can be used to find the optimal roadmap path.

The roadmaps can quickly become prohibitively expensive to *compute, store, or search* as R increases. Planning directly over multi-robot configuration space roadmaps is only feasible for simple problems with relatively small configurations spaces, for a small number of robots. This shortcoming motivates specialized techniques to address the curse of dimensionality in the multi-robot problem domain.

Tensor Roadmaps

In the typical multi-robot problems discussed thus far each robot possesses some notion of independence, its own objective, its own degrees of freedom, and its own valid motions and connected configuration space. This *independence* is beneficial in problem instances where each robot is an independent kinematic chain (i.e., changing any of the degrees of freedom of one robot does not update the geometries of any of the other robots). In such scenarios the multi-robot roadmap has an intrinsic structure [13, 24, 25]. Since each of the vertices can be represented by a set of individual robot configurations, and the edges represent coordinated motions for each of the robots, intuitively the multi-robot roadmap can be thought of as a combination of single robot roadmap slices.

[4] Deterministic sampling strategies have also been proposed for PRM methods. The motivation remains the same that the chosen sampling strategy has to effectively cover the space as the number of samples increases. Uniform sampling is the classical technique, though for simpler domains grids, or other informed samples preserve the same arguments.

Definition 6 (Tensor roadmap) Given a roadmap $\mathcal{G}^i$ with n^i samples each constructed for robot r^i, in its corresponding valid configuration space $\mathcal{Q}^i_{\text{free}}$ for each $i \in [1, 2, \cdots R]$, a valid tensor roadmap $\mathcal{G}$ in the multi-robot configuration space is defined as follows:

$$\begin{aligned}
\mathcal{G}(\mathcal{V}, \mathcal{E}) &= \mathcal{G}^1 \times \mathcal{G}^2 \cdots \times \mathcal{G}^R \\
\mathcal{V} &= \{x = (x^1, \cdots x^R),\ x^i \in \mathcal{G}^i.\mathcal{V}, \\
&\quad x \in \mathcal{Q}_{\text{free}}\},\ |\mathcal{V}| \leq \prod_{i=1}^{R} n^i \\
\mathcal{E} &= \{e = (x_u, x_v), \quad x_u, x_v \in \mathcal{V}, \\
&\quad (x^i_u, x^i_v) \in \mathcal{G}^i.\mathcal{E} \quad \forall i \in [1, \cdots R], \\
&\quad \pi_{u \to v} = (\pi^1_{u \to v}, \cdots, \pi^R_{u \to v}), \\
&\quad \pi_{u \to v}(t) \in \mathcal{Q}_{\text{free}} \quad \forall t\}
\end{aligned}$$

This means that the *tensor roadmap* contains all combinations of vertices from $\mathcal{G}^i$, and each such tensor vertex is connected to edges which are again all possible combinations of edges that exist in each $\mathcal{G}^i$. For both the vertices and edges, they need to be collision free given the multi-robot configuration space, which accounts for all the robot geometries and their interactions as well. It is straightforward to see however that the number of vertices in such a roadmap is at most the product of all the individual roadmap sizes. The nature of these edge-neighborhoods is shown in Fig. 4. This attributes the way in which the tensor roadmap can grow in size with R. This adheres to the trend observed in the volume of $\mathcal{Q}_{\text{free}}$ as well.

Power of Implicit Structure The construction of the tensor roadmap defined above depends on the samples and edge-connectivity of each individual $\mathcal{G}^i$. More importantly, given each $\mathcal{G}^i$ the structure of $\mathcal{G}$ can be implicitly derived. This means that if each of the R individual roadmaps has $\mathcal{O}(\hat{n})$ samples, it is sufficient to store a R roadmaps separately with a total of $\mathcal{O}(R\hat{n})$ samples, instead of the tensor roadmap which has $\mathcal{O}(\hat{n}^R)$ samples.

What is left now is to define computationally efficient ways to reconstruct the structure of $\mathcal{G}$ on the fly and recover solutions from it.

*M**

An efficient search strategy to implicitly explore the tensor roadmap was proposed in the M^* algorithm [15••, 26, 27]. This was an improvement over the traditional optimal heuristic search algorithm A^*, by specifically exploiting the structure of the tensor roadmap.

A^* maintains a priority queue and proceeds by systematically selecting vertices and *expanding* their entire neighborhoods. This expansion step involves adding to the queue all of the roadmap neighbors connected by out-edges. It is possible to implicitly derive these neighborhoods in the tensor roadmap from the out-edges in the individual robot roadmaps. Such an expansion step however still needs to maintain all the expanded neighborhoods and a queue that grows drastically due to the exponentially larger neighborhoods of the tensor roadmap.

The key insight in M^* is that, not all the neighbors of a vertex in the tensor roadmap are useful.

Each robot in a motion planning problem has its own goal. In the absence of other robots, each robot has an optimal policy (shortest roadmap path) to reach the goal. If the motion planning problem does not involve any of the robots ever interacting along their shortest paths, then

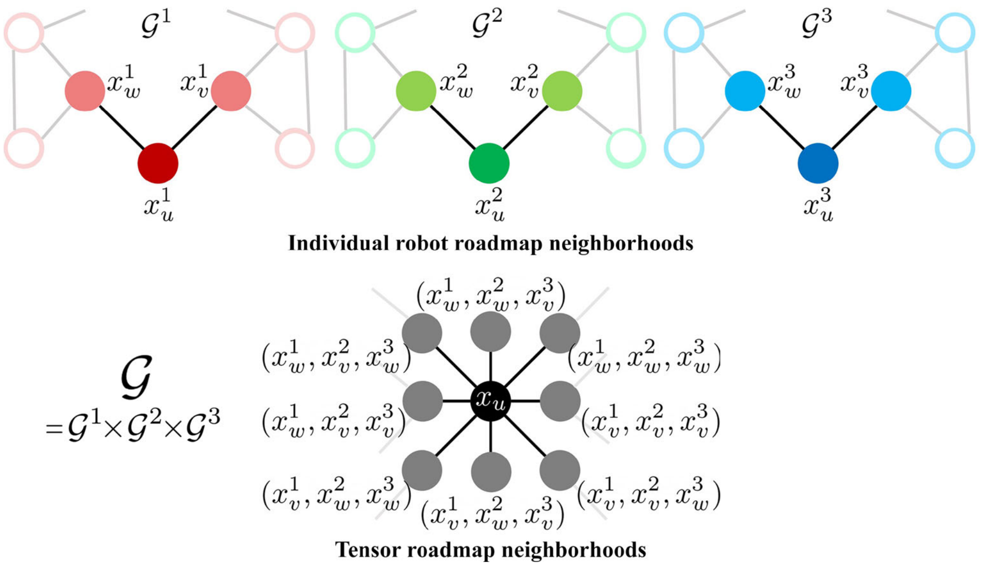

Fig. 4 *Top:* Neighborhoods of vertices x^1_u, x^2_u, and x^3_u in individual robot roadmaps $\mathcal{G}^1$, $\mathcal{G}^2$, and $\mathcal{G}^3$ are shown. The two neighbors each are denoted by the subscripts v and w. *Bottom:* The tensor roadmap $\mathcal{G}$ in the multi-robot configuration space shows the neighborhood of vertex $x_u = (x^1_u, x^2_u, x^3_u)$ composed of a combination of the individual roadmap's out-edges

the optimal solution[5] will consist of each robot moving along their respective shortest paths. This property implies that reasoning about groups of robots depends on whether they are expected to interact. More concretely, M^* only considers subsets of the tensor roadmap neighborhoods which are expected to involve some $\hat{R}$ number of robot-robot interactions. When $\hat{R} = 0$, the robot can follow its individual shortest path greedily, since no other robot stands in its way.

During the search, if the algorithm finds that the selected vertex has a neighborhood that only involves a subset of robots that can interact (i.e., some combination of $\mathcal{G}^i$ out-edges involve overlaps of robot geometries), then it *only expands the neighborhood composed from the interacting robot out-edges*. The algorithm keeps track of the limited neighborhood of every explored vertex, based on the robot interactions at the vertex level, and the edge-level (i.e., two robots can collide at a vertex, or two robots might collide while traversing a pair of individual robot edges concurrently). An illustration of such different interaction sets is shown in Fig. 5.

M^* ensures that it prunes any unnecessary expansions by considering interacting robot subsets, effectively reducing the dimensionality of the search space in local neighborhoods. M^* preserves the expansions necessary to converge to any optimal solution A^* might discover. This makes M^* much more scalable than performing A^* on the tensor roadmap though M^* can exhibit a worst-case complexity that is the same as A^*.

Both algorithms build an *optimal search tree* rooted at the start vertex, and growing outwards towards the goal, with successive expansions dictating the parent-child structure. It should be noted that the heuristic search-based strategy requires that every useful expansion be included in the queue immediately. Even though M^* limits this to the inter acting robot subset, there is any overhead for determining interactions for the large set of edges. For constrained problems with high interactions, not enough of the neighborhoods might be pruned, and this can still lead to scalability issues.

An alternative way to search the implicit tensor roadmap was introduced in the sampling-based framework of *discrete Rapidly-exploring Random Tree* (dRRT) [12, 14••, 28].

dRRT*

As the number of robot interactions increases in a part of the configuration space, the size of the neighborhood that might need to be explored can get quickly out of hand. Sampling within these neighborhoods provides an easy way to mitigate this problem. As long as the sampling procedure guarantees that the desired edge in the tensor neighborhood is expanded with enough attempts, asymptotic properties can be argued, while preserving practical performance for the search.

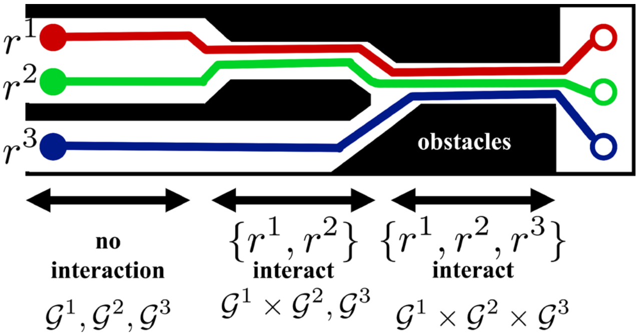
r^1
r^2
r^3
obstacles
no interaction
$\{r^1, r^2\}$ interact
$\{r^1, r^2, r^3\}$ interact
$\mathcal{G}^1, \mathcal{G}^2, \mathcal{G}^3$
$\mathcal{G}^1 \times \mathcal{G}^2, \mathcal{G}^3$
$\mathcal{G}^1 \times \mathcal{G}^2 \times \mathcal{G}^3$

Fig. 5 A toy problem with three disk robots (r^1,r^2,r^3) that have start on the left and have to reach the right of the environment. The three sections indicate regions where different degrees of robot-robot interaction might occur. The middle section involves robots $\{r^1,r^2\}$ needing to pass through a corridor. The third section requires all three robots $\{r^1,r^2,r^3\}$ to pass through the last corridor. M^* can focus on the limited subset of robots whose combined neighborhood needs to be explored depending upon their interactions. The bottom row indicates the tensor neighborhoods involved out of $\mathcal{G}^1$, $\mathcal{G}^2$, and $\mathcal{G}^3$

The original idea was proposed [12] as an extension to the RRT algorithm. As opposed to roadmap-based methods, RRT builds a tree structure rooted at the start configuration. The *discrete* nature of dRRT is to restrict the expansion of the tree to be confined to the tensor roadmap, which is a discrete approximation of $\mathcal{Q}_{\text{free}}$. This structure is shown in Fig. 6. From the theoretical properties of RRT, it was shown that given enough iterations for growing the tree, a feasible solution on the tensor roadmap would be discovered. It was initially shown that when the underlying roadmaps were probabilistically complete, as the size of each $\mathcal{G}^i$ increases, and the amount of iterations given to the tree expansion increases, the algorithm is guaranteed to find a solution.

Optimality Building on this idea, a new algorithm dRRT* [14••, 28] was proposed to provide asymptotic optimality. It had been known that roadmaps could be constructed with specific neighborhood connections [20, 21] to assure such a property for motion planning problems. It turns out when each individual roadmap $\mathcal{G}^i$ is constructed to be asymptotically optimal, the tensor roadmap is also asymptotically optimal, i.e., the tensor roadmap will eventually contain a roadmap path with a solution cost that converges to the optimal solution cost.[6] The asymptotic guarantee of the search

[5] Though the optimality is stated for shortest path lengths, the optimality at this step describing paths for individual robots which can be combined into a multi-robot optimal path.

[6] Optimality was shown for different cost functions: Euclidean arc length in $\mathcal{Q}_{\text{free}}$, and any linear combination of individual robot shortest paths including *makespan* distance.

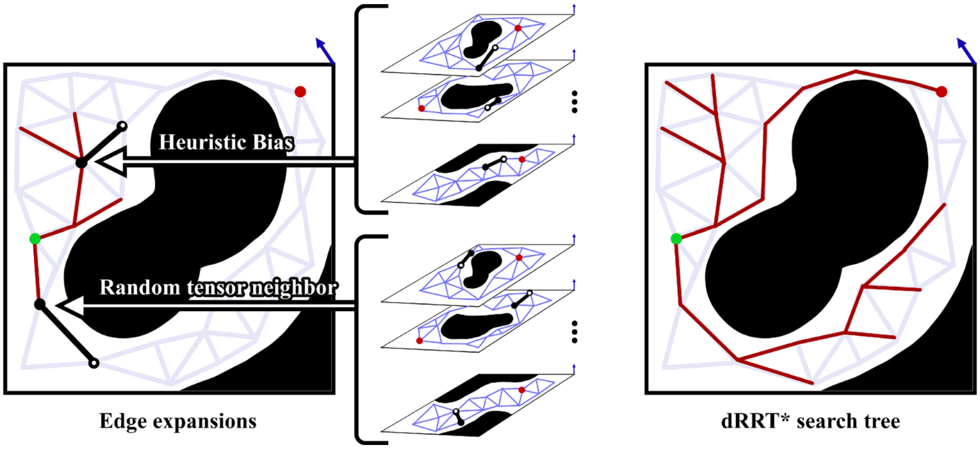

Fig. 6 The image on the left shows two edge expansions of dRRT* in $\mathcal{Q}_{\text{free}}$: a heuristic biased one in the top middle and a random tensor neighbor in the bottom middle. The heuristic bias edge makes progress towards the red goal vertex in the individual robot roadmaps. The image on the right shows what the search tree might look like once the goal is reached. Note that edges are rewired within tensor neighborhoods as well

tree's optimal nature relies on ensuring that every neighborhood edge could be selected uniformly at random.

The tree expansions include a fraction of iterations where both a tree node, and a neighboring edge are selected at random. A neighboring tensor edge can be deduced by randomly choosing from each $\mathcal{G}^i$ neighborhood, requiring R sampling operations instead of having to reconstruct the entire exponential tensor neighborhood. Once an edge is expanded, it must be collision checked to ensure no robot-robot interactions occur. A *rewiring* step is also performed for every new expansion to keep the local roadmap neighborhood of a newly added tree node optimally connected. This allows more aggressive convergence to good-quality solutions.

The practical performance of the algorithm heavily relies on the informed heuristic guidance. Each vertex is assigned a heuristic estimate[7] for the motion planning problem. When there is no interaction, the heuristic is assumed to be the best each robot can do given its own $\mathcal{G}^i$. Goal biasing in a tree-based planner attempts to greedily shoot the tree towards the goal. Therefore, goal biasing in the tree expansion takes the form of choosing the next edge on each robot's optimal path. As an additional algorithmic change to promote progress towards the goal, every time all the robots make progress towards the goal, the goal biasing is repeatedly applied, until it gets stuck with robot-robot collisions. These two exploration and exploitation strategies are shown in Fig. 6.

The algorithm is designed to quickly make progress towards the goal in parts of the multi-robot configuration space where the robots do not need to interact. Whenever coordination would be required, the random sampling is designed to explore around the blockages, beyond which greedy progress again takes over until the goal is attained.

dRRT* has been shown to be very robust to planning for a large number of high-degree of freedom systems like robotic arms (which are typically have $d = 7$), where each robot can have its unique roadmap and configuration space. Due to the generality of the sampling-based formulation, it is amenable to be applied to a wide variety of such combinations of robotic systems.

Note to Practitioners The number of samples, say $\hat{n}$, in each $\mathcal{G}^i$ is effectively a parameter. Though the theoretical properties dictate making the roadmaps as large as possible, in practice this can get prohibitvely expensive to search. The design of dRRT* is tuned to overcome the exponent in the tensor product's size $\hat{n}^R$, i.e., overcome the scaling issues with larger number of robots R. It is advised to maintain the most compact individual roadmaps that is sufficient to solve problems in an environment, i.e., keeping $\hat{n}$ small. It should be noted that even if $\hat{n}$ is relatively small, say 1000, the tensor roadmap being implicitly searched is going to be $\hat{n}^R$, which for 5 robots for instance will have 10^{15} vertices.

Multi-agent Path Finding on a Graph

The multi-agent path finding (MAPF) problem on a graph is a special case of the general roadmap formulation described in the previous sections. If every robot r^i is identical, and the configuration space of each robot is also identical, then it is possible to only consider a single instance $\mathcal{G}^i$ for all the robots. For instance, consider a factory floor setting with planar robots. The configuration space of each robot consists of the reachable passageways in the factory. Such settings can be discretized effectively into a single graph. The conflict resolution in this case is simplified since collision checking can be avoided in favor of simply ensuring *vertices and edges on the graph are never occupied*

[7] Each robot's individual shortest path to the goal proves a very reliable heuristic in the multi-robot configuration space. An efficient way to precompute this would be to maintain an all-pairs shortest path data structure tied to each $\mathcal{G}^i$. This is feasible when each individual roadmap is reasonably sized.

by more than one robot at a time. The typical complexities of motion planning here reduce to an optimization problem on a graph.

This problem was reduced [16••] to a *multi-commodity network flow* by thinking of each robot as a commodity, edges as channels with capacity 1 at any time step. The consideration of *discrete time-steps* as a dimension for coordinating the robots leads to a *time-expanded* graph, where copies of vertices are created for every time-step, with a specific edge-structure connecting time step t to time step $t + 1$. This is shown in the *gadget* in Fig. 7.

The network flow problem is formulated as a mixed integer linear program to optimize the solution in terms of edge weights on the graph. This line of work has extensively applied this method large multi-agent problem instances.

Pebble-Graphs Extensive work has been done on an abstraction of this problem [16••, 29–32] which thinks of robots as pebbles on commonly shared graphs. Several algorithmic and complexity results have been demonstrated through the years in this domain.

Road Networks A related problem is multi-agent path finding over road networks [33–35]. Here again the graph is shared by all the agents, but as opposed to the classical MAPF problem, roads display loads and congestion. Routing is seen as an optimization problem, with large-scale pickup-and delivery, or even multi-modal variants [35].

Geometric Reasoning A model of representing multi-robot teams with shared workspaces reasons about each robot as a disk [36, 37] or ball [38, 39] and applies computational geometric reasoning to argue stronger properties for systems that can be simplified to fit these models.

Extension to Task Planning

Insights from multi-robot motion planning have been applied to different applications of task planning. The most straightforward application involves concurrently performing sequences of actions using multiple robots [25], where each action is itself a motion plan. Such problems involve task and motion planning, and motions realizing multi-robot actions require all the typical considerations of multi-robot motion planning. The multiplicity of available actions [40] also draws unique parallels to multi-robot planning. Specifically when the actions involve picking and placing multiple objects, it describes an object rearrangement problem. The objects themselves can be thought of as pebbles [41]. dRRT*-eque techniques [42] have been applied to problems where multiple robots need to coordinate their actions. The task planning domain itself can have graphical structures [43] that can be leveraged in tandem with the tensor decomposition. In a multi-robot pick-and-place domain, task planning reduces to an MAPF problem on specially constructed graph [44].

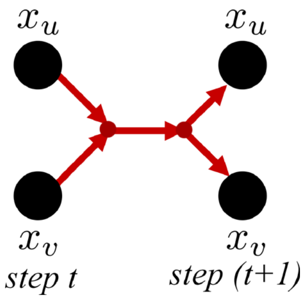

Fig. 7 The gadget for constructing the time-expanded graph. This replaces every undirected graph edge (x_u, x_v). Traversing a graph edge between steps t and $t+1$ makes a robot cross the corresponding gadget

Conclusions

Roadmap-based centralized, coordinated, multi-robot motion planning provides theoretically sound, and practical ways to find high-quality multi-robot motions. It is still a challenge to provide real-time performance similar to decoupled, optimization-based techniques, but there is value in roadmap-based, complete, optimal methods. These can provide the *valid and good solutions* that can be reused in experience-based, or learning-based frameworks. Currently, roadmap-based approaches provide the tools that are general enough to scale to many high degree-of-freedom robots, while being versatile and efficient enough to solve problems involving large numbers of simpler robotic systems. A focus on pushing the performance of coordinated motion planning would be key in properly seizing the benefits of large-scale deployment of robotic teams in real-world scenarios in the very near future.

Declarations

Conflict of Interest The author declares no competing interest.

Human and Animal Rights and Informed Consent This article does not contain any studies with human or animal subjects performed by the author.

References

Papers of particular interest, published recently, have been highlighted as:
• Of importance
•• Of major importance

1. Erdmann M, Lozano-Perez T. On multiple moving objects. Algorithmica. 1987;2(1-4):477.
2. Ghrist R, O'Kane JM, LaValle SM. Computing pareto optimal coordinations on roadmaps. IJRR. 2005;24(11):997–1010.

3. LaValle SM, Hutchinson S. Optimal motion planning for multiple robots having independent goals. IEEE TRA. 1998;14(6):912–925.
4. Peng J, Akella S. Coordinating multiple robots with kinodynamic constraints along specified paths. IJRR. 2005;24(4):295–310.
5. van den Berg J, Overmars M. Prioritized motion planning for multiple robots. In: IROS; 2005. p. 430–435.
6. van den Berg J, Snoeyink J, Lin M, Manocha D. Centralized path planning for multiple robots: optimal decoupling into sequential plans. In: RSS; 2009.
7. Khan A, Zhang C, Li S, Wu J, Schlotfeldt B, Tang S, Ribeiro A, Bastani O, Kumar V. Learning safe unlabeled multi-robot planning with motion constraints. In: IEEE/RSJ IROS; 2019. p. 7558–7565.
8. Tang S, Kumar V. A complete algorithm for generating safe trajectories for multi-robot teams. In: ISRR; 2015.
9. Kloder S, Hutchinson S. Path planning for permutation-invariant multi-robot formations. In: ICRA; 2005. p. 650–665.
10. O'Donnell PA, Lozano-Pérez T. Deadlock-free and collision-free coordination of two robot manipulators. In: IEEE ICRA; 1989. p. 484–489.
11. Salzman O, Hemmer M, Halperin D. On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages. IEEE TASE. 2015;12(2):529–538.
12. Solovey K, Salzman O, Halperin D. Finding a needle in an exponential haystack: discrete RRT for exploration of implicit roadmaps in multi-robot motion planning. IJRR. 2016;35(5):501–513.
13. Svestka P, Overmars M. Coordinated path planning for multiple robots. RAS. 1998;23:125–152.
14.•• Shome R, Solovey K, Dobson A, Halperin D, Bekris KE. dRRT: scalable and informed asymptotically-optimal multi-robot motion planning. AURO. 2020;44(3):443–467. **dRRT is an asymptotically optimal sampling-based motion planner for coordinated motion planning for groups of robots. A search tree explores an implicitly constructed tensor roadmap in the combined configuration space**.
15.•• Wagner G, Choset H. Subdimensional expansion for multirobot path planning. Artif Intell J. 2015;219:1–24. **A search-based approach to discover solutions over a Cartesian product of individual robot roadmaps in regions of the search space where subsets of robots are expected to interact**.
16.•• Yu J, LaValle SM. Optimal multirobot path planning on graphs: complete algorithms and effective heuristics. IEEE T-RO. 2016;32(5):1163–1177. **Multi-robot path finding for identical robots on a shared roadmap can be mapped to a mutli-commodity network flow problem that can be efficiently solved using mixed integer linear programming**.
17. Lozano-Perez T. Spatial planning: a configuration space approach. In: Autonomous robot vehicles. Springer; 1990. p. 259–271.
18. Kavraki LE, Svestka P, Latombe J, Overmars M. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans Robot Autom. 1996;12(4):566–580.
19. LaValle SM, Kuffner JJ. Randomized kinodynamic planning. IJRR. 2001;20:378–400.
20. Karaman S, Frazzoli E. Sampling-based algorithms for optimal motion planning. IJRR. 2011;30(7):846–894.
21. Janson L, Schmerling A, Clark A, Pavone M. Fast marching tree: a fast marching sampling-based method for optimal motion planning in many dimensions. IJRR. 2015;34(7):883–921.
22. Sȧnchez-Ante G, Latombe J. Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In: IEEE ICRA; 2002. p. 2112–2119.
23. Hart PE, Nilsson NJ, Raphael B. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cybern. 1968;4(2):100–107.
24. Gravot F, Alami R. A method for handling multiple roadmaps and its use for complex manipulation planning. In: IEEE ICRA; 2003. p. 2914–2919.
25. Gharbi M, Cortés J, Siméon T. Roadmap composition for multi-arm systems path planning. In: IEEE/RSJ IROS; 2009. p. 2471–2476.
26. Wagner G, Choset H. M*: a complete multirobot path planning algorithm with performance bounds. In: IROS. IEEE; 2011. p. 3260–3267.
27. Wagner G, Kang M, Choset H. Probabilistic path planning for multiple robots with subdimensional expansion. In: IEEE ICRA; 2012. p. 2886–2892.
28. Dobson A, Solovey K, Shome R, Halperin D, Bekris K. Scalable asymptotically-optimal multi-robot motion planning. In: IEEE MRS; 2017. p. 120–127.
29. Kornhauser DM, Miller GL, Spirakis PG. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. Master's thesis, M. I. T., Dept of Electrical Engineering and Computer Science. 1984.
30. Luna R, Bekris K. An efficient and complete approach for cooperative path-finding. In: Conference on artificial intelligence, San Francisco, California, USA; 2011. p. 1804–1805.
31. Yu J. Constant factor time optimal multi-robot routing on high-dimensional grids. In: RSS; 2018.
32. Chinta R, Han SD, Yu J. Coordinating the motion of labeled discs with optimality guarantees under extreme density. In: WAFR. Springer; 2018. p. 817–834.
33. Ma H, Li J, Kumar TS, Koenig S. Lifelong multi-agent path finding for online pickup and delivery tasks. In: AAMAS; 2017. p. 837–845.
34. Ho F, Salta A, Geraldes R, Goncalves A, Cavazza M, Prendinger H. Multi-agent path finding for uav traffic management. In: AAMAS; 2019. p. 131–139.
35. Choudhury S, Solovey K, Kochenderfer MJ, Pavone M. Efficient large-scale multi-drone delivery using transit networks. In: ICRA. IEEE; 2020. p. 4543–4550.
36. Adler A, de Berg M, Halperin D, Solovey K. Efficient multi-robot motion planning for unlabeled discs in simple polygons. IEEE TASE. 2015;12(4):1–17.
37. Solovey K, Yu J, Zamir O, Halperin D. Motion planning for unlabeled discs with optimality guarantees. In: RSS; 2015.
38. Turpin M, Michael N, Kumar V. Concurrent assignment and planning of trajectories for large teams of interchangeable robots. In: IEEE ICRA; 2013. p. 842–848.
39. Solomon I, Halperin D. Motion planning for multiple unit-ball robots in Rd. In: WAFR. Springer; 2018. p. 799–816.
40. Cohen JB, Phillips M, Likhachev M. Planning single-arm manipulations with n-Arm robots. In: RSS; 2014.
41. Krontiris A, Shome R, Dobson A, Kimmel A, Bekris K. Rearranging similar objects with a manipulator using pebble graphs. In: IEEE humanoids; 2014. p. 1081–1087.
42. Shome R, Bekris K. Anytime multi-arm task and motion planning for pick-and-place of individual objects via handoffs. In: IEEE MRS. IEEE; 2019. p. 37–43.
43. Shome R, Solovey K, Yu J, Bekris K, Halperin D. Fast and high-quality dual-arm rearrangement in synchronous, monotone tabletop setups. In: WAFR; 2018. p. 778–795.
44. Shome R, Bekris K. Synchronized multi-arm rearrangement guided by mode graphs with capacity constraints. In: WAFR; 2020.