Scalable Asymptotically-Optimal Multi-Robot Motion Planning

Andrew Dobson Kiril Solovey Rahul Shome Dan Halperin Kostas E. Bekris

Abstract—Finding asymptotically-optimal paths in multi-robot motion planning problems could be achieved, in principle, using sampling-based planners in the composite configuration space of all of the robots in the space. The dimensionality of this space increases with the number of robots, rendering this approach impractical. This work focuses on a scalable sampling-based planner for coupled multi-robot problems that provides asymptotic optimality. It extends the dRRT approach, which proposed building roadmaps for each robot and searching an implicit roadmap in the composite configuration space. This work presents a new method, dRRT*, and develops theory for scalable convergence to optimal paths in multi-robot problems. Simulated experiments indicate dRRT* converges to high-quality paths while scaling to higher numbers of robots where the naïve approach fails. Furthermore, dRRT* is applicable to high-dimensional problems, such as planning for robot manipulators.

I. INTRODUCTION AND PRIOR WORK

This work proposes a method for coupled multi-robot motion planning with formal quality guarantees. While many planners exist for such challenges [22], [4], [3], this remains a difficult problem, especially for multiple high-dimensional robots, as in Fig. 1. Most applications require robots to achieve simultaneous motion in a shared workspace, computing paths quickly to adapt to sensing input. Preprocessing given knowledge of the static scene can produce high-quality paths online very quickly, and compact sampling-based roadmaps perform well under these conditions [8], [9]. These methods also converge to optimal solutions given sufficient density [7], i.e., at least \( O(n \log(n)) \) edges are needed for a roadmap with \( n \) vertices, while near-optimal solutions are achieved after finite computation time [2], [6].

Naïvely constructing a sampling-based roadmap or tree in the robots’ composite configuration spaces achieves asymptotic optimality for multi-robot problems, but does not scale due to memory requirements depending exponentially on the problem’s dimension. The alternative is decoupled planning, where paths for robots are computed and then coordinated [10]. These methods typically lack completeness and optimality guarantees, but hybrid approaches can achieve optimal decoupling to retain guarantees [21]. Some work deals with complex kinodynamic constraints [12], while others focus on coordinated manipulation [15]. Control-based methods scale to hundreds of robots, but generally lack global guarantees [20], [19].

The dRRT approach [16] is a scalable sampling-based approach which is probabilistically complete. It searches an implicit tensor product roadmap in the composite configuration space [18] by constructing one roadmap per robot. This work proposes dRRT*, which is an efficient asymptotically optimal variant of this method. Simulations show the method practically generates high-quality paths of equivalent quality to naïve approaches while scaling to complex, high-dimensional problems.

II. PROBLEM SETUP AND NOTATION

Consider a shared workspace with \( R \geq 2 \) robots, and each operating in configuration space \( \mathcal{C}_i \) for \( 1 \leq i \leq R \). Let \( \mathcal{C}_i^f \subset \mathcal{C}_i \) be each robot’s free space, where it is free from collision with the static scene, and \( \mathcal{C}_i = \mathcal{C}_i \setminus \mathcal{C}_i^f \) is the forbidden space for robot \( i \).

The composite configuration space \( \mathcal{C} = \prod_{i=1}^{R} \mathcal{C}_i \) is the Cartesian product of each robot’s \( \mathcal{C} \)-space. A composite configuration \( Q = (q_1, \ldots, q_R) \in \mathcal{C} \) is an \( R \)-tuple of robot configurations. For two distinct robots \( i, j \), denote by \( I_i^j(q_j) \subset \mathcal{C}_i \) the set of configurations where \( i \) and \( j \) collide. Then, the composite free space \( \mathcal{C}_i^f \subset \mathcal{C} \) consists of configurations \( Q = (q_1, \ldots, q_R) \)
subject to:

1. $q_i \in C_i$ for every $1 \leq i \leq R$;
2. $q_i \notin I_1(q_j), q_j \notin I_1(q_i)$ for every $1 \leq i < j \leq R$.

Each free configuration $Q \in C$ requires robots to not collide with obstacles, and each pair to not collide with each other. The composite forbidden space is defined as $C^o = C \setminus C^f$.

Given $S, T \in C^f$, where $S = (s_1, \ldots, s_R), T = (t_1, \ldots, t_R)$, a trajectory $\Sigma : [0, 1] \to C^f$ is a continuous curve in $C^f$, such that $\Sigma(0) = S, \Sigma(1) = T$, where the $R$ robots move simultaneously. $\Sigma$ is an $R$-tuple $(\sigma_1, \ldots, \sigma_R)$ of single-robot paths such that $\sigma_i : [0, 1] \to C_i$.

The objective is to find a trajectory which minimizes a given cost function $c(\cdot)$. The analysis assumes the cost is the sum of robot path lengths i.e., $c(\Sigma) = \sum_{i=1}^R |\sigma_i|$, where $|\sigma_i|$ denotes the standard arc length of $\sigma_i$. The arguments also work for $\max_{i=1..R} |\sigma_i|$. Section IV shows sufficient conditions for dRRT° to asymptotically converge to returning optimal trajectories over the cost function $c$.

III. METHODS FOR COMPOSITE SPACE PLANNING

For a fixed $n \in \mathbb{N}$, define for every robot $i$ the PRM roadmap $G_i = (V_i, E_i)$, constructed over $C_i$, such that $|V_i| = n$ with connection radius $r(n)$. Then, $G = (\bar{V}, \bar{E}) = G_1 \times \cdots \times G_R$ is the tensor product roadmap in space $C$ (for an illustration, see Figure 2). Formally, $\bar{V} = \{(v_1, v_2, \ldots, v_R), \forall i, v_i \in V_i\}$ is the Cartesian product of the nodes from each roadmap $G_i$. For two vertices $V = (v_1, \ldots, v_m) \in \bar{V}, V' = (v'_1, \ldots, v'_m) \in \bar{V}$ the edge set $\bar{E}$ contains edge $(V, V')$ if for every $i$ it is that $v_i = v'_i$ or $(v_i, v'_i) \in E_i$.

As shown in Algorithm 1, dRRT° grows a tree $T$ over $G$, rooted at the start configuration $S$ and initializes path $\pi_{best}$ (line 1). The method stores the node added each iteration $V$ (Line 2), as part of an informed process to guide the expansion of $T$ towards the goal. The method iteratively expands $T$ given a time budget (Line 3), as detailed by Algorithm 2, storing the newly added node $V$ (Line 4). After expansion, the method traces the path which connects the source $S$ with the target $T$ (Line 5). If such a path is found, it is stored in $\pi_{best}$ if it improves upon the cost of the previous solution (Lines 6, 7). Finally, the best path found $\pi_{best}$ is returned (Line 8).

**Algorithm 1:** dRRT°($G, S, T$)

1. $\pi_{best} \leftarrow \emptyset$, $\text{init}(S)$;
2. $V \leftarrow S$;
3. while time.elapsed() $<$ time.limit do
4. $\quad V \leftarrow \text{Expand_dRRT}^*(G, T, V, T)$;
5. $\quad \pi \leftarrow \text{TracePath}(T, S, T)$;
6. $\quad$ if $\pi \neq \emptyset$ and cost$(\pi) < \text{cost}(\pi_{best})$ then
7. $\quad \pi_{best} \leftarrow \pi$;
8. return $\pi_{best}$

The expansion step is given in Alg. 2. The default initial step of the method is given in Lines 1–4, i.e., when no $V_{last}$ is passed (Line 1), which corresponds to an exploration step similar to RRT: a random sample $Q_{rand}$ is generated in $C$ (Line 2), its nearest neighbor $V_{near}$ in $T$ is found (Line 3) and the oracle function $O_d(\cdot, \cdot)$ returns the implicit graph node $V_{new}$ that is a neighbor of $V_{near}$ on the implicit graph in the direction of $Q_{rand}$ (Line 4). If a $V_{last}$, however, is provided (Line 5)—which happens when the last iteration managed to generate a node closer to the goal relative to its parent—then the $V_{new}$ is greedily generated so as to be a neighbor of $V_{last}$ in the direction of the goal $T$ (Line 6).

In either case, the method next finds neighbors $N$, which are adjacent to $V_{new}$ in $G$ and have also been added to $T$ (Line 7). Among $N$, the best node $V_{best}$ is chosen, for which the local path $L(V_{best}, V_{new})$ is collision-free and that the path total cost to $V_{new}$ is minimized (Line 8). If no such parent can be found (Line 9), the expansion fails and no node is returned (Line 10). Then, if $V_{new}$ is not in $T$, it is added (Lines 11-13). Otherwise, if it exists, the tree is rewired so as to contain edge $(V_{best}, V_{new})$, and the cost of the $V_{new}$'s sub-tree (if any) is updated (Lines 14, 15). Then, for all nodes in $N$ (Line 16), the method tests $T$ should be rewired through $V_{new}$ to reach this neighbor. Given that $L(V_{new}, v)$ is collision-free and is of lower cost than the existing path to $v$ (Line 17), the tree is rewired to make $V_{new}$ be the parent of $v$ (line 18).

Finally, if in this iteration the heuristic value of $V_{new}$ is lower than its parent node $V_{best}$ (line 19), the method
Algorithm 2: Expand dRRT∗(G, T, Vlast, T)

1 if Vlast == NULL then
2 \(Q^{\text{rand}} \leftarrow \text{Random Sample}()\);
3 \(V_{\text{near}} \leftarrow \text{Nearest Neighbor}(T, Q^{\text{rand}})\);
4 \(V_{\text{new}} \leftarrow \bigcup_d(V_{\text{near}}, Q^{\text{rand}})\);
5 else
6 \(V_{\text{new}} \leftarrow \bigcup_d(V_{\text{last}}, T)\);
7 \(N \leftarrow \text{Adjacent}(V_{\text{new}}, G) \cap \forall T\);
8 \(V_{\text{best}} \leftarrow \arg \min_{\forall V_{\text{new}} \in N \text{s.t.} L(V_{\text{new}}, G) \subseteq \mathbb{C}^d} c(V_{\text{new}}) + c(L(V, V_{\text{new}}))\);
9 if Vbest == NULL then
10 return NULL;
11 if Vnew \(\notin T\) then
12 T.Add_Vertex(Vnew);
13 T.Add_Edge(V_best, Vnew);
14 else
15 T.Rewire(V_best, Vnew);
16 for \(v \in N\) do
17 if \(c(V_{\text{new}}) + c(L(V_{\text{new}}, v)) \leq c(v)\) and \(L(V_{\text{new}}, v) \subseteq \mathbb{C}^d\) then
18 T.Rewire(V_{new}, v);
19 if \(h(V_{\text{new}}) < h(V_{\text{best}})\) then
20 return V_{new};
21 else
22 return NULL;

returns \(V_{\text{new}}\) (Line 20), causing the next iteration to greedily expand \(V_{\text{new}}\). Otherwise, \(\text{NULL}\) is returned so as to do an exploration step. Note that the approach is implemented with helpful branch-and-bound pruning after an initial solution is found, though this is not reflected in the algorithms.

\(V_{\text{new}}\) is determined via an oracle function. Using this oracle function and a simple rewiring scheme is sufficient for showing asymptotic optimality for dRRT∗ (see Section IV). The oracle function \(\bigcup_d\) for a two-robot case is illustrated in Figure 3. First, let \(\rho(Q, Q')\) be the ray coming from configuration \(Q\) terminating at \(Q'\). Then, denote \(\angle_v Q(Q', Q'')\) as the minimum angle between \(\rho(Q, Q')\) and \(\rho(Q, Q'')\). When \(Q^{\text{rand}}\) is drawn in \(\mathbb{C}^d\), its nearest neighbor \(V_{\text{near}}\) in \(T\) is found. Then, project the points \(Q^{\text{rand}}\) and \(V_{\text{near}}\) into each robot space \(\mathbb{C}_i\), i.e., ignore the configurations of other robots.

The method separately searches the single-robot roadmaps to discover \(V_{\text{new}}\). Denote \(V_{\text{near}} = (v_1, \ldots, v_R)\), \(Q^{\text{rand}} = (q_1, \ldots, q_R)\). For every robot \(i\), let \(N_i \subseteq \mathbb{C}_i\) be the neighborhood of \(v_i \in \mathbb{V}_i\), and identify \(v_i' = \arg \min_{v \in N_i} \angle_{v_i} Q_{i}^{\text{rand}}, v\).

The oracle function returns node \(V_{\text{new}} = (v_1', \ldots, v_R')\).

As in the standard RRT as well as in dRRT, the dRRT∗ approach has a Voronoi-bias property \cite{11}. It is, however, slightly more involved to observe as shown in Figure 4. To generate an edge \((V_{\text{old}}, V_{\text{new}})\), random sample \(Q^{\text{rand}}\) must be drawn within the Voronoi cell of \(V_{\text{new}}\), denoted Vor\((V)\) (A) and in the general direction of \(V_{\text{old}}\), denoted Vor\((V)\) (B). The intersection of these two volumes \(\text{Vol}(V) = \text{Vor}(V) \cap \text{Vor}(V')\) is the volume to be sampled generate \(V_{\text{new}}\) via \(V_{\text{near}}\).

IV. ANALYSIS

In this section, the theoretical properties of dRRT∗ are examined, beginning with a study of the asymptotic convergence of the implicit roadmap \(\hat{G}\) to containing a path in \(\mathbb{C}^d\) whose cost converges to the optimum. Then, it is shown dRRT∗ eventually discovers the shortest path in \(\hat{G}\), and that the combination of these two facts proves the asymptotic optimality of dRRT∗.

For simplicity, the analysis is restricted to the setting of robots operating in Euclidean space, i.e. \(\mathbb{C}_i\) is a \(d\)-dimensional Euclidean hypercube \([0, 1]^d\) for fixed \(d \geq 2\).

Additionally, the analysis is restricted to the specific cost function of total distance, i.e., \(|\Sigma| := \sum_{i=1}^R |\sigma_i|\).

Discussions on lifting these restrictions is provided in Section VI.

3For simplicity, it is assumed that all the robots have the same number of degrees of freedom \(d\).
A. Optimal Convergence of $\hat{G}$

For each robot, an asymptotically optimal PRM* roadmap $G_i$ is constructed having $n$ samples and using a connection radius $r(n)$ necessary for asymptotic convergence to the optimum [7]. By the nature of sampling-based algorithms, each graph cannot converge to the true optimum with finite computation, as such a solution may have clearance of exactly 0. Instead, this work focuses on the notion of a robust optimum ⁴, showing that the tensor product roadmap $\hat{G}$ converges to this value.

**Definition 1:** A trajectory $\Sigma : [0, 1] \to C_i$ is robust if there exists a fixed $\delta > 0$ such that for every $\tau \in [0, 1], X \in C^\infty$ it holds that $\|\Sigma(\tau) - X\|_2 \geq \delta$, where $\|\cdot\|_2$ denotes the standard Euclidean distance.

**Definition 2:** A value $c > 0$ which denotes a path cost is robust if for every fixed $\epsilon > 0$, there exists a robust path $\Sigma$ such that $|\Sigma| \leq (1 + \epsilon)c$. The robust optimum $c^*$, is the infimum over all such values.

For any fixed $n \in \mathbb{N}^+$, and a specific instance of $\hat{G}$ constructed from $R$ roadmaps, having $n$ samples each, denote by $\Sigma(n)$ the shortest path from $S$ to $T$ over $G$.

**Definition 3:** $\hat{G}$ is asymptotically optimal (AO) if for every fixed $\epsilon > 0$ it holds that $|\Sigma(n)| \leq (1 + \epsilon)c^*$ a.a.s., where the probability is over all the instantiations of $\hat{G}$ with $n$ samples for each PRM.

Using this definition, the following theorem is proven. Recall that $d$ denotes the dimension of a single-robot configuration space.

**Theorem 1:** $\hat{G}$ is AO when

$$r(n) \geq r^*(n) = (1 + \eta)2 \left( \frac{1}{\sqrt{d}} \left( \log n \right) \right)^{\frac{1}{2}},$$

where $\eta$ is any constant larger than 0.

**Remark.** Note that $r^*(n)$ was developed in [6, Theorem 4.1], and guarantees AO of PRM* for a single robot. The proof technique described in that work will be one of the ingredients used to prove Theorem 1. ⁶

By the definition of $c^*$, for any given $\epsilon > 0$ there exists a robust trajectory $\Sigma : [0, 1] \to C^\infty$, and a fixed $\delta > 0$, such that the cost of $\Sigma$ is at most $(1 + 1/2 - \epsilon)c^*$ and for every $X \in C^\infty, \tau \in [0, 1]$ it holds that $\|\Sigma(\tau) - X\|_2 \geq \delta$.

Next, it is shown that $\hat{G}$ contains a trajectory $\Sigma(n)$ such that:

$$|\Sigma(n)| \leq (1 + o(1)) \cdot |\Sigma|,$$

a.a.s.. This immediately implies that $|\Sigma(n)| \leq (1 + \epsilon)c^*$, which will finish the proof of Theorem 1.

Thus, it remains to show that there exists a trajectory on $\hat{G}$ which satisfies Equation 1 a.a.s.. As a first step, it will be shown that the robustness of $\Sigma = (\sigma_1, \ldots, \sigma_R)$ in the composite space implies robustness in the single-robot setting, i.e., robustness along $\sigma_i$.

For $\tau \in [0, 1]$ define the forbidden space parameterized by $\tau$ as

$$C^n_i(\tau) = \bigcup_{j=1, j \neq i}^R I^j_i(\sigma_j(\tau)).$$

**Claim 1:** For every robot $i$, $\tau \in [0, 1]$, and $q_i \in C^n_i(\tau), \|\sigma_i(\tau) - q_i\|_2 \geq \delta$.

**Proof:** Fix a robot $i$, and fix some $\tau \in [0, 1]$ and a configuration $q_i \in C^n_i(\tau)$. Next, define the following composite configuration $Q = (\sigma^1(\tau), \ldots, q_i, \ldots, \sigma^R(\tau))$.

Note that it differs from $\Sigma(\tau)$ only in the $i$-th robot’s configuration. By the robustness of $\Sigma$ it follows that

$$\delta \leq \|\Sigma(\tau) - Q\|_2$$

$$= \left(\|\sigma_i(\tau) - q_i\|_2^2 + \sum_{j=1, j \neq i}^R \|\sigma_j(\tau) - \sigma_j(\tau)\|_2^2\right)^{\frac{1}{2}}$$

$$\leq \|\sigma_i(\tau) - q_i\|_2.$$

The result of claim 1 is that the paths $\sigma_1, \ldots, \sigma_R$ are robust in the sense that there is sufficient clearance for the individual robots to not collide with each other given a fixed location of a single robot. A Lemma is derived using proof techniques from the literature [6], and it implies every $G_i$ contains a single-robot path $\sigma_i(n)$ that converges to $\sigma_i$.

**Lemma 1:** For every robot $i$, $G_i$ constructed with $n$ samples and a connection radius $r(n) \geq r^*(n)$ contains a path $\sigma_i(n)$ with the following attributes a.a.s.:

(i) $\sigma_i(0) = s_i, \sigma_i(n) = t_i$;
(ii) $|\sigma_i(n)| \leq (1 + o(1))|\sigma_i|$;
(iii) $\forall q \in \text{Im}(\sigma_i(n)), \exists \tau \in [0, 1]$ s.t. $\|q - \sigma_i(\tau)\|_2 \leq r^*(n)$.

**Proof:** The first property (i) follows from the fact that $s_i, t_i$ are directly added to $G_i$. The rest follows from the proof of Theorem 4.1 in [6], which is applicable here since $r(n) \geq r^*(n)$.

Lemma 1 also implies that $\hat{G}$ contains a path in $G_i$ that represents robot-to-obstacle collision-free motions, and minimizes the multi-robot metric cost. In particular, define $\Sigma(n) = (\sigma_1(n), \ldots, \sigma_R(n))$, where $\sigma_i(n)$ are obtained from Lemma 1. Then

$$|\Sigma(n)| = \sum_{i=1}^R |\sigma_i(n)| \leq (1 + o(1))\sum_{i=1}^R |\sigma_i| \leq (1 + o(1))|\Sigma|.$$

However, it is not clear whether this ensures the existence of a path where robot-robot collisions are avoided.

---

4Note that the given definition of robust optimum is similar to that in previous work [17].

5Let $A_1, A_2, \ldots$ be random variables in some probability space and let $B$ be an event depending on $A_i$. We say that $B$ occurs asymptotically almost surely (a.a.s.) if $\lim_{n \to \infty} \Pr(B \mid A_i) = 1$.

6Note that $r^*(n)$ can be refined to incorporate the proportion of $C_i$, which would reduce this expression.

---

CONFIDENTIAL. Limited circulation. For review only.
That is, although \( \text{Im}(\sigma^{(n)}_i) \subset C_i \), it might be the case that \( \text{Im}(\Sigma^{(n)}) \cap C_0 \neq \emptyset \). Next it is shown that \( \sigma^{(n)}_1, \ldots, \sigma^{(n)}_R \) can be reparametrized to induce a composite-space path whose image is fully contained in \( C_i \), with length equivalent to \( \Sigma^{(n)} \).

For each robot \( i \), denote by \( V_i = (v_i^1, \ldots, v_i^{\ell_i}) \) the chain of \( G_i \) vertices traversed by \( \sigma^{(n)}_i \). For every \( v_i^j \in V_i \) assign a timestamp \( \tau^j_i \) of the closest configuration along \( \sigma^i \), i.e.,

\[
\tau^j_i = \arg \min_{\tau \in [0,1]} \| v_i^j - \sigma_i(\tau) \|_2.
\]

Also, define \( T_i = (\tau^1_i, \ldots, \tau^{\ell_i}_i) \) and denote by \( T \) the ordered list of \( \bigcup_{i=1}^R T_i \) according to the timestamp values. Now, for every \( i \), define a global timestamp function \( T S_i : T \rightarrow V_i \) which assigns to each global timestamp in \( T \) a single-robot configuration from \( V_i \). It thus specifies in which vertex robot \( i \) resides at time \( \tau \in T \). For \( \tau \in T \), let \( j \) be the largest index such that \( \tau^j_i \leq \tau \). Then simply assign \( T S_i(\tau) = \tau^j_i \). From property (iii) in Lemma 1 and Claim 1 it follows that no robot-robot collisions are induced by the reparametrization, concluding the proof of Theorem 1.

B. Asymptotic Optimality of dRRT*

Finally, dRRT* is shown to be AO. Denote by \( m \) the time budget in Algorithm 1, i.e., the number of iterations of the loop. Denote by \( \Sigma^{(n,m)} \) the solution returned by dRRT* for \( n \) and \( m \).

**Theorem 2:** If \( r(n) > r^*(n) \) then for every fixed \( \epsilon > 0 \) it holds that

\[
\lim_{n,m \to \infty} \Pr \left[ |\Sigma^{(n,m)}| \leq (1 + \epsilon)c^* \right] = 1.
\]

Since \( \hat{G} \) is AO (Theorem 1), it suffices to show that for any fixed \( n \), and a fixed instance of \( \hat{G} \), defined over \( R \) PRMs with \( n \) samples each, dRRT* eventually (as \( m \) tends to infinity), finds the optimal trajectory over \( \hat{G} \). This can be shown using the properties of a Markov chain with absorbing states [5, Theorem 11.3]. While a full proof is omitted here, the high-level idea is similar to what is presented in previous work [16, Theorem 3], and expanded upon in Appendix A. By restricting the states of the Markov chain to being the graph vertices along the optimal path, setting the target vertex to be an absorbing vertex, and showing that the probability of transitioning along any edge in this path is nonzero (i.e. the probability is proportional to \( \frac{\mu(V_i)}{\rho(V_i)} \)), then the probability that this process does not reach the target state along the optimal path converges to 0 as the number of dRRT* iterations tends to infinity. The final step is to show that the above statements hold when both \( m \) and \( n \) tend to \( \infty \). A proof for this phenomenon can be found in [16, Theorem 6].

V. Experimental Validation

This section provides experimental validation of dRRT* by demonstrating practical convergence, scalability, and applicability to dual-arm manipulation. The approach is run on three benchmarks, executed on a cluster with Intel(R) Xeon(R) CPU E5-4650 @ 2.70GHz processors, and 128GB of RAM. Additional data are provided in Appendix B and in an accompanying video.

2 Disk Robots among 2D Polygons: This base-case test involves 2 disks (\( C_i := \mathbb{R}^2 \)) of radius 0.2, in a \( 10.2 \times 10.2 \) region, as in Figure 5. The disks start at \((0,0)\), and \((9,9)\), and swap positions.

This is a setup where it is possible to compute the explicit roadmap, which is not practical in more involved scenarios. In particular, dRRT* is tested against: a) running \( A^* \) on the implicit tensor roadmap \( \hat{G} \) (referred to as “Implicit \( A^* \)” defined over the same individual roadmaps with \( N \) nodes each as those used by dRRT*; and b) an explicitly constructed PRM* roadmap with \( N^2 \) nodes in the composite space.

Results are shown in Figure 6. dRRT* converges to the optimal path over \( \hat{G} \), similar to the one discovered by Implicit \( A^* \), while quickly finding an initial solution of high quality. Furthermore, the implicit tensor product roadmap \( \hat{G} \) is of comparable quality to the explicitly constructed roadmap.

![Fig. 5. The 2D environment where the 2 disk robots operate.](image)

![Fig. 6. Average solution cost over iterations. Data averaged over 10 roadmap pairs. dRRT* (solid line) converges to the optimal path through \( G \) (dashed line).](image)

Table I presents running times. dRRT* and implicit \( A^* \) construct \( 2 \) \( N \)-sized roadmaps (row 3), which are faster to construct than the PRM* roadmap in \( C \) (row 1). PRM* becomes very costly as \( N \) increases. For \( N = 500 \), the explicit roadmap contains 250,000 vertices, taking 1.7GB of RAM to store, which was the upper limit for the machine used. When the roadmap can be constructed, it is quicker to query (row 2). dRRT* quickly
returns an initial solution (row 5), and converges within 5% of the optimum length (row 6) well before Implicit A* returns a solution as $N$ increases (row 4). The next benchmark further emphasizes this point.

**TABLE I**

<table>
<thead>
<tr>
<th>Number of nodes: $N$</th>
<th>50</th>
<th>100</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^<em>$-PRM$^</em>$ construction</td>
<td>3.427</td>
<td>13.293</td>
<td>69.551</td>
</tr>
<tr>
<td>$N^<em>$-PRM$^</em>$ query</td>
<td>0.002</td>
<td>0.004</td>
<td>0.023</td>
</tr>
<tr>
<td>2 $N$-size PRM$^*$ construction</td>
<td>0.135</td>
<td>0.274</td>
<td>0.558</td>
</tr>
<tr>
<td>Implicit A* search over $G$</td>
<td>0.684</td>
<td>2.497</td>
<td>10.184</td>
</tr>
<tr>
<td>dRRT$^*$ over $G$ (initial)</td>
<td>0.343</td>
<td>0.257</td>
<td>0.358</td>
</tr>
<tr>
<td>dRRT$^*$ over $G$ (converged)</td>
<td>3.497</td>
<td>4.418</td>
<td>5.429</td>
</tr>
</tbody>
</table>

Many Disk Robots among 2D Polygons: In the same environment as above, the number of robots $R$ is increased to evaluate scalability. Each robot starts on the perimeter of the environment and is tasked with reaching the opposite side. An $N = 50$ roadmap is constructed for every robot. It quickly becomes intractable to construct a PRM$^*$ roadmap in the composite space of many robots.

**Fig. 7.** Data averaged over 10 runs. (Top): Relative solution cost and success ratio of dRRT$^*$, dRRT and RRT$^*$ for increasing $R$. dRRT$^*$: average iteration and variance for initial solution (top of box), and solution cost and variance after 100,000 iterations (bottom). Similar results for RRT$^*$. Single data point for dRRT (no quality improvement after first solution). (Bottom): Solution costs over time.

Figure 7 shows the inability of alternatives to compete with dRRT$^*$ in scalability. Solution costs are normalized by an optimistic estimate of the path cost for each case, which is the sum of the optimal solutions for each robot, disregarding robot-robot interactions. Implicit A* fails to return solutions even for 3 robots. Directly executing RRT$^*$ in the composite space fails to do so for $R \geq 6$. The original dRRT method (without the informed search component) starts suffering in success ratio for $R \geq 5$ and returns worse solutions than dRRT$^*$. The average solution times for dRRT may decrease as $R$ increases but this is due to the decreasing success ratio, i.e. dRRT begins to only succeed at easy problems.

**Fig. 8.** (Top): dRRT$^*$ is run for a dual-arm manipulator to go from its home position (above) to a reaching configuration (below) and achieves perfect success ratio as $n$ increases. (Bottom): dRRT$^*$ solution quality over time. Here, larger roadmaps provide benefits in terms of running time and solution quality.

**Dual-arm manipulator:** This test shows the benefits of dRRT$^*$ when planning for two 7-dimensional arms. Figure 8 shows that RRT$^*$ fails to return solutions within 100$K$ iterations. Using small roadmaps is also insufficient for this problem. Both dRRT$^*$ and Implicit A* require larger roadmaps to begin succeeding. But with $N \geq 500$, Implicit A* always fails, while dRRT$^*$ maintains a 100% success ratio. As expected, roadmaps of increasing size result in higher quality path. The informed nature of dRRT$^*$ also allows to find initial solutions fast, which together with the branch-and-bound primitive allows for good convergence.

**VI. DISCUSSION**

This work studies the asymptotic optimality of sampling-based multi-robot planning over implicit structures. The objective is to efficiently search the composite space of multi-robot systems while also achieving formal guarantees. This requires appropriate construction of the individual robot roadmaps and search of the tensor product roadmap. Performance is further improved by informed search in the composite space.

These results may extend to more complex settings involving kinodynamic constraints by relying on recent work [14], [13]. Furthermore, the analysis may be valid for different cost functions other than total distance. The tools presented here can prove useful in the context of simultaneous task and motion planning for multiple robots, especially manipulators [1].
REFERENCES


A. Proof of Convergence

This appendix examines the result of Theorem 2, and formally proves the convergence of the dRRT* tree toward containing all optimal paths.

Lemma 2 (Optimal Tree Convergence of dRRT*): Consider an arbitrary optimal path \( \pi^* \) originating from \( v_0 \) and ending at \( v_T \). Then, let \( O(m) \) be the event such that after \( m \) iterations of dRRT*, the search tree \( T \) contains the optimal path up to segment \( k \). Then,

\[
\lim_{m \to \infty} P(O(m)) = 1.
\]

Proof. This property will be proven using a theorem from Markov chain literature [5, Theorem 11.3]. Specifically, the properties of absorbing Markov chains can be exploited to show that dRRT* will eventually contain the optimal path over \( G \) for a given query. An absorbing Markov chain is one such that there is some subset of states in which the transition matrix only allows that state to transition to itself.

The proof follows by showing that the dRRT* method can be described as an absorbing Markov chain, where the target state of a query is represented as an absorbing state in a Markov chain. For completeness, the theorem is re-stated here.

Theorem 3 (Thm 11.3 in Grinstead & Snell): In an absorbing Markov chain, the probability that the process will be absorbed is 1 (i.e., \( Q(m) \to 0 \) as \( n \to \infty \)), where \( Q(m) \) is the transition submatrix for all non-absorbing states.

There are two steps to using this proof. First, that the dRRT* search can be cast as an absorbing Markov chain, and second, that the probability of transition for from each state to the next in this chain is nonzero (i.e. that each state can eventually be connected to the target).

For query \((S, T)\), let the sequence \( V = \{v_1, v_2, \ldots, v_t\} \) of length \( t \) represent the vertices of \( G \) corresponding to the optimal path through

APPENDIX

The reference list includes papers on multi-robot and multi-agent systems, including:

- Dobson and Bekris (2015) on planning representations and algorithms for prehensile multi-arm manipulation.
- LaValle and Overmars (2001) on geometric robotics.

The appendix provides a proof of convergence for the dRRT* algorithm, showing that it will contain the optimal path with probability 1 as the number of iterations increases.

This appendix examines the result of Theorem 2, and formally proves the convergence of the dRRT* tree toward containing all optimal paths.

Ref:

the graph which connects these points, where $v_t$ corresponds to the target vertex, and furthermore, let $v_t$ be an absorbing state. Theorem 3 operates under the assumption that each vertex $v_i$ is connected to an absorbing state ($v_t$ in this case).

Then, let the transition probability for each state have two values, one for each state transitioning to itself, which corresponds to the dRRT* search expanding along some other arbitrary path. The other value is a transition probability from $v_i$ to $v_{i+1}$. This corresponds to the method sampling within the volume $Vol(v_i)$.

Then, as the second step, it must be shown that this volume has a positive probability of being sampled in each iteration. It is sufficient then to argue that $\frac{\mu(Vol(s_i))}{\mu(C_f)} > 0$. Fortunately, for any finite $n$, previous work has already shown that this is the case given general position assumptions [16, Lemma 2].

Given these results, the dRRT* is cast as an absorbing Markov chain which satisfies the assumptions of 3, and therefore, the matrix $Q(m) \to 0$. This implies that the optimal path to the goal has been expanded in the tree, and therefore $\lim \inf_{m \to \infty} P(O(m)) = 1$. □

B. More Experimental Data

This appendix presents additional experimental data.

1) 2-Robot Benchmark: For the 2 disk robots case, Figure 9 provides the solution cost found by dRRT* over computation time instead of over iterations.

2) $R$ Disk Robots: For the $R$ disk robots, Figure 10 shows query resolution times for the various methods.

To emphasize the lack of scalability for alternate methods, additional experiments were run in this setup using a smaller, manually crafted roadmap. The tests use a 9-node roadmap for each robot shown in Figure 12. Each robot roadmap samples its nodes within the shaded regions indicated in the figure.

Figure 11 indicates that even using a very well-crafted and small roadmap for a problem does not help the alternate methods to scale. While they perform better than with the random roadmaps, Implicit A* times out for $R \geq 5$, and RRT* times out for $R \geq 6$.

3) Manipulator Benchmark: For the dual-arm manipulator benchmark, Figure 13 presents solution quality over iterations instead of over time.