2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Refined Analysis of Asymptotically-Optimal Kinodynamic Planning in the State-Cost Space

Michal Kleinbort[1], Edgar Granados[2], Kiril Solovey[3], Riccardo Bonalli[3], Kostas E. Bekris[2], and Dan Halperin[1]

***Abstract*— We present a novel analysis of AO-RRT: a tree-based planner for motion planning with kinodynamic constraints, originally described by Hauser and Zhou (AO-X, 2016). AO-RRT explores the state-cost space and has been shown to efficiently obtain high-quality solutions in practice without relying on the availability of a computationally-intensive two-point boundary-value solver. Our main contribution is an optimality proof for the single-tree version of the algorithm—a variant that was not analyzed before. Our proof only requires a mild and easily-verifiable set of assumptions on the problem and system: Lipschitz-continuity of the cost function and the dynamics. In particular, we prove that for any system satisfying these assumptions, any trajectory having a piecewise-constant control function and positive clearance from the obstacles can be approximated arbitrarily well by a trajectory found by AO-RRT. We also discuss practical aspects of AO-RRT and present experimental comparisons of variants of the algorithm.**

I. Introduction

Motion planning is a fundamental problem in robotics, concerned with allowing autonomous robots to navigate in complex environments while avoiding collisions with obstacles. The problem is already challenging in the simplified geometric setting, and even more so when considering the kinodynamic constraints that the robot has to satisfy. This work is concerned with the latter setting, and consider the case where the robot's system is specified by differential constraints of the form

$$\dot{x} = f(x,u), \quad \text{for } x \in \mathcal{X}, u \in \mathcal{U}, \tag{1}$$

where $\mathcal{X} \subseteq \mathbb{R}^d$ is the robot's state space, and $\mathcal{U} \subseteq \mathbb{R}^D$ is the control space, for some $d, D \geqslant 2$. The objective of motion planning is thus to find a control function $\Upsilon : [0,T] \rightarrow \mathcal{U}$, which induces a *valid* trajectory $\pi : [0,T] \rightarrow \mathcal{X}$, such that (i) Equation (1) is satisfied, (ii) π is contained in the free space $\mathcal{F} \subseteq \mathcal{X}$, and (iii) the motion takes the robot from its initial state x_{init} to the goal region $\mathcal{X}_{\text{goal}} \subseteq \mathcal{X}$.

In *optimal motion planning*, the objective is to find a control function Υ and a trajectory π satisfying the constraints (i), (ii), (iii), which also minimize the trajectory cost, specified by

$$\text{COST}(\pi) = \int_0^T g(\pi(t), \Upsilon(t))dt, \tag{2}$$

where $g : \mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R}_+$ is a cost derivative. Depending on the precise formulation of g, $\text{COST}(\pi)$ may represent the distance traversed by the robot, the energy required to execute the motion, or other metrics.

Almost thirty years of research on motion planning have led to a variety of approaches to tackle the problem, ranging from computational-geometric algorithms, potential fields, optimization-based methods, and search-based solutions [1], [2]. To the best of our knowledge, the only approach that is capable of satisfying global optimality guarantees, while still being computationally practical, is sampling-based planning. Sampling-based algorithms capture the connectivity of the free space of the problem via random sampling of states (and sometimes controls) and connecting nearby states, to yield a graph structure.

The celebrated work of Karaman and Frazzoli [3] laid the foundations for optimality in sampling-based motion planning. They introduced several new algorithms and proved mathematically that they converge to the optimal solution as the number of samples generated by the algorithms tends to infinity. This property is termed asymptotic optimality (AO). Many researchers have followed their footsteps, and designed new algorithms, which can be used in various applications [4], [5], [6], [7].

Unfortunately, the applicability of most of the aforementioned results to optimal planning with kinodynamic constraints remains limited. In particular, the majority of results only apply to the geometric (holonomic) setting of the problem. While a small subset of results do consider the kinodynamic case, they assume the existence of a two point *boundary value problem (BVP) solver*, which given two states $x, x' \in \mathcal{X}$ returns the lowest-cost trajectory connecting them (see, [8], [9], [10], [11], [12], [13], [14]). In practice BVP solvers are usually only available for simple robotic systems, and in many cases they are prohibitively costly to use, which limits their applicability.

Recently, there have been sampling-based approaches that do not rely on the existence of a BVP solver [15]. These methods employ forward propagation instead. Li et al. [16], [17] provided an analysis of tree sampling-based planners that perform random propagation from first principles and proposed the `SST` algorithm. `SST` is in practice computationally efficient and achieves asymptotic near-optimality, which is the property of converging toward a path with bounded suboptimality. True AO properties can be achieved by `SST`*, which sacrifices computational efficiency by progressively shrinking a pruning radius parameter. The approach proposed here aims for AO properties and computational efficiency,

Work by D.H. and M.K. has been supported in part by the Israel Science Foundation (grant nos. 825/15,1736/19), by the Blavatnik Computer Science Research Fund, and by grants from Yandex and from Facebook. K.B. was supported by NSF awards IIS-1734492, IIS-1723869, CCF-1934924.

[1]Blavatnik School of Computer Science, Tel-Aviv University, Israel.
[2]Computer Science Department, Rutgers University, NJ 08854, USA.
[3]Aeronautics and Astronautics Department, Stanford University, CA 94305, USA.

978-1-7281-7395-5/20/$31.00 ©2020 IEEE

while avoiding the critical dependence on parameters, such as pruning radii, that are difficult to tune for a variety of motion planning problems.

Most recently, Hauser and Zhou [18] proposed a meta algorithm AO-x, which allows to adapt any *well-behaved* non-optimal kinodynamic sampling-based planner, denoted by x, into an AO algorithm. This is achieved by substituting the d-dimensional state space $\mathcal{X}$ on which the former is run with the $(d+1)$-dimensional space $\mathcal{Y} = \mathcal{X} \times \mathbb{R}$, where the last coordinate encodes the solution cost. Then, x is iteratively applied to shrinking subsets $\mathcal{Y}_i$ of $\mathcal{Y}$ for $i \in \mathbb{N}_+$, where the maximal value of the last coordinate (representing the cost) is gradually decreased with i, and hence the cost of the returned solution. The authors combined their framework with the forward-propagating versions of RRT [19] and EST [20], to yield AO-RRT and AO-EST, both of which demonstrated favorable performance over competitors. The observation that the cost induced by a system can be analyzed by augmenting the state space in the above manner was first considered by Pontryagin (see, [21]).

We follow up on Hauser and Zhou's approach. We augment their work by addressing aspects of the analysis that we believe require more attention, namely what are the precise conditions under which using the augmented-space approach will lead to provably AO solutions. The main issue that we address is the assumption [18] that x is well-behaved, without proving this property for neither RRT nor for EST. Well behavedness consists of two requirements: (i) x must find a feasible solution eventually within each $\mathcal{Y}_i$—a property corresponding to probabilistic completeness (PC) [22]— and (ii) the cost of the solution found in $\mathcal{Y}_i$ is smaller (with non-negligible probability) than the maximal cost value over $\mathcal{Y}_i$. Note that requirement (ii) is a particularly strong assumption, essentially requiring x to be "nearly" AO, i.e., gradually reducing the cost of the solution when applied to the bounded subspaces $\mathcal{Y}_i$ for $i \in \mathbb{N}_+$.

In this context, it should be noted that some variants of RRT are not even PC [23] (and thus not well behaved). Furthermore, it is not specified for what types of robotic systems [18], with respect to $\mathcal{X}, f, g$, or problem instances $\mathcal{F}, x_{\text{init}}, \mathcal{X}_{\text{goal}}$ this property holds. Another logical gap that has not been adequately addressed is that the proof focuses on a version of AO-RRT which grows multiple trees, and does not seem to directly extend to the single-tree version of AO-RRT used in the experiments of that paper.

A. Contribution

We present a novel analysis of AO-RRT: a tree-based planner for motion planning with kinodynamic constraints, originally described by Hauser and Zhou [18]. We focus on a variant that constructs a single tree, rather than multiple trees, embedded in the augmented state space $\mathcal{Y}$, and which was not analyzed before. We note that this variant was used in the experiments in [18]. The approach does not require a BVP solver and can be viewed as an AO generalization of the non-AO RRT planner [19].

Our main contribution is a rigorous optimality proof for the single-tree AO-RRT. Our proof only requires an easily-verifiable set of assumptions on the problem and system: we require Lipschitz-continuity of the cost function and the dynamics. In particular, we prove that for any system satisfying these assumptions, any trajectory having a piecewise-constant control function and positive clearance from obstacles can be approximated arbitrarily well by a trajectory found by AO-RRT. Furthermore, we develop explicit bounds on the convergence rate of the algorithm. Our AO proof relies on the theory that we have recently developed for the probabilistic completeness of RRT [24].

We also discuss practical aspects of AO-RRT, namely node pruning and a hybrid approach that combines the algorithm with other planners, while still maintaining AO. Then we present an experimental comparison of AO-RRT variants with the vanilla RRT, and SST for a fixed-wing plane and a rally car.

The paper is organized as follows. The AO-RRT algorithm is described in Section II. Section III proceeds with the theoretical properties of AO-RRT and gives the asymptotic optimality proof. Practical aspects of the algorithm are discussed in Section IV and experiments are presented in Section V. Finally, in Section VI we discuss further research.

II. THE SINGLE-TREE AO-RRT ALGORITHM

We describe the single-tree AO-RRT approach. Henceforth we will refer to this algorithm simply as AO-RRT. Recall that $\mathcal{X}, \mathcal{F}, \mathcal{U}$ denote the state, free, and control spaces, respectively. We assume that $\mathcal{X}$ is compact, and $\mathcal{F}$ is open. The AO-RRT algorithm is very similar to the (kinodynamic) RRT algorithm, based on [19]. Whereas RRT grows a tree embedded in $\mathcal{X}$, AO-RRT (see Algorithm 1) does so in the state-cost space. In particular, we define the augmented (state) space $\mathcal{Y} := \mathcal{X} \times \mathbb{R}_+$, which is $(d+1)$-dimensional, where the additional coordinate represents the cost of the (non-augmented) state. That is, a point $y \in \mathcal{Y}$ can be viewed as a pair $y = (x, c)$, where $x \in \mathcal{X}$ and $c \geqslant 0$ represents the cost of the trajectory from x_{init} to x over the tree $\mathcal{T}(\mathcal{Y})$. Given a point $y \in \mathcal{Y}$ we use the notation $x(y), c(y)$ to represent its component of $\mathcal{X}$ and cost, respectively.

The AO-RRT algorithm has the following inputs: In addition to an initial start state x_{init}, goal region $\mathcal{X}_{\text{goal}}$, number of iterations k, maximal total duration for propagation T_{prop}, and control space $\mathcal{U}$, which RRT accepts, AO-RRT also accepts a maximal cost $c_{\max}$. See Section IV for more information on how to choose $c_{\max}$.

AO-RRT constructs a tree $\mathcal{T}(\mathcal{Y})$, embedded in $\mathcal{Y}$ and rooted in $y_{\text{init}} = (x_{\text{init}}, 0)$, by performing k iterations of the following form. In each iteration, it generates a random sample y_{rand} in $\mathcal{Y}$, by randomly sampling $\mathcal{X}$ and the cost space $[0, c_{\max}]$ (lines 3-4). In addition a random control u_{rand} and duration t_{rand} are generated by calling the routine SAMPLE (lines 5-6). For a given set S, the procedure SAMPLE(S) produces a sample uniformly and randomly from S.

Next, the nearest neighbor y_{near} of y_{rand} in $\mathcal{T}(\mathcal{Y})$ is retrieved (line 7). We emphasize that this operation is performed in the $(d+1)$-dimensional space $\mathcal{Y}$ using a suitable distance metric such as the Euclidean metric in the augmented space (see Section IV). Then, in line 8, the algorithm uses a forward propagation approach (using PROPAGATE) from y_{near} to generate a new state y_{new}: the random control input u_{rand} is applied for time duration t_{rand} from $x(y_{\text{near}})$ reaching a new state $x_{\text{new}} \in \mathcal{X}$ through a trajectory π_{new}. The state $x(y_{\text{near}})$ is then coupled with the cost of executing π_{new} together with $c(y_{\text{new}})$ (line 9). Mathematically, for $x \in \mathcal{X}, u \in \mathcal{U}, t > 0$, we have that

$$\text{PROPAGATE}(x, u, t) := \int_0^t f(x(t), u)\, \mathrm{d}t.$$

Finally, COLLISION-FREE(π_{new}) checks whether the trajectory reaching y_{new} from y_{near} using the control u_{rand} and duration t_{rand} is collision free. This operation is known as *local planning*, and is typically achieved by densely sampling the trajectory and applying a dedicated collision detection mechanism [25]. If indeed the trajectory is collision free, y_{new} is added as a vertex to the tree and is connected by an edge from y_{near} (lines 10-12). The trajectory π_{new} is also added to the edge. If y_{new} is in the goal region and its cost is the smallest encountered so far, then y_{min} is substituted with this point (lines 13,14). Finally, a lowest-cost trajectory (if exists) is returned in line 15. Note that the algorithm maintains the lowest-cost trajectory discovered so far by keeping track of the last vertex y_{min} on such a trajectory.

III. THEORETICAL PROPERTIES OF AO-RRT

We spell out the assumptions that we make with respect to the system and the cost function, and state our main theorem. Then, in Section III-A, we describe the problem in the augmented space $\mathcal{Y}$, define the augmented system F, and study its properties. We then leverage this in the proof of the main theorem in Section III-B.

Throughout this section we use the following notations. For simplicity, in our proofs we use the standard Euclidean norm, denoted by $\|\cdot\|$. We note, however, that all proofs can be generalized to work with the weighted Euclidean norm. Given a set $S \subseteq \mathbb{R}^{d'}$, for some $d' > 0$, we denote by $|S|$ its Lebesgue measure. For a given point $y \in \mathbb{R}^{d'}$, and a radius $r > 0$, we use $\mathcal{B}_r^{d'}(y)$ to denote the d'-dimensional Euclidean ball of radius r centered at y.

We make the following assumption concerning f (Eq. (1)):

Assumption 1 Lipschitz continuity of the system**.** *The system f is Lipschitz continuous for both of its arguments. That is, $\exists K_u^f, K_x^f > 0$ s.t. $\forall\ x_0, x_1 \in \mathcal{X}, \forall u_0, u_1 \in \mathcal{U}$:*

$$\|f(x_0, u_0) - f(x_0, u_1)\| \leqslant K_u^f \|u_0 - u_1\|,$$
$$\|f(x_0, u_0) - f(x_1, u_0)\| \leqslant K_x^f \|x_0 - x_1\|.$$

We make the following assumption concerning g (Eq. (2)):

Assumption 2 Lipschitz continuity of the cost**.** *The cost derivative g is Lipschitz continuous for both of its arguments. That is, $\exists K_u^g, K_x^g > 0$ s.t. $\forall\ x_0, x_1 \in \mathcal{X}, \forall u_0, u_1 \in \mathcal{U}$:*

$$\|g(x_0, u_0) - g(x_0, u_1)\| \leqslant K_u^g \|u_0 - u_1\|,$$
$$\|g(x_0, u_0) - g(x_1, u_0)\| \leqslant K_x^g \|x_0 - x_1\|.$$

Definition 1. A piecewise constant control function $\overline{\Upsilon}$ with resolution Δt is the concatenation of constant control functions $\bar{\Upsilon}_i : [0, \Delta t] \to u_i$, where $u_i \in \mathcal{U}$, and $1 \leqslant i \leqslant k$, for some $k \in \mathbb{N}_{>0}$.

From this point on, when we say a *valid trajectory* we mean a valid trajectory as described in Section I, with the extra proviso that the control function is piecewise constant.

Definition 2. Let π be a valid trajectory, and let T be its duration. We define the clearance of π to be the maximal value $\delta > 0$ such that

$$\bigcup_{t \in [0,T]} \mathcal{B}_\delta^d(\pi(t)) \subset \mathcal{F} \text{ and } \mathcal{B}_\delta^d(\pi(T)) \subset \mathcal{X}_{\text{goal}}.$$

We say that a trajectory is *robust* if its clearance is positive.

We arrive to our main contribution that establishes the rate of convergence of AO-RRT.

Theorem 1. *Assume that Assumptions 1, 2 hold and fix $\varepsilon \in (0,1)$. Denote by π_k the solution obtained by* AO-RRT *after k iterations. For every robust trajectory π having a piecewise-constant control function there exist a finite $k_0 \in \mathbb{N}, a > 0, b > 0$, such that for every $k > k_0$ it holds that*

$$\Pr[\text{COST}(\pi_k) > (1+\varepsilon)\text{COST}(\pi)] \leqslant a e^{-bk}.$$

A. Properties of the augmented system

It would be convenient to view the problem of optimal planning with respect to f, g, as a feasible motion planning for an augmented system F, which is defined as follows. The *augmented* system F encompasses both types of transitions in f and g, respectively. The control space for this system is simply $\mathcal{U}$, and its state space is $\mathcal{Y} = \mathcal{X} \times \mathbb{R}_+$. Formally,

$$\dot{y} = (\dot{x}, \dot{c}) = F(y, u) = (f(x, u), g(x, u)), \tag{3}$$

for $y = (x, c)$, where $x \in \mathcal{X}, c \in \mathbb{R}_+, u \in \mathcal{U}$.

We have the following claim with respect to F:

Algorithm 1 $\texttt{AO-RRT}(x_{\text{init}}, \mathcal{X}_{\text{goal}}, k, T_{\text{prop}}, \mathcal{U}, c_{\text{max}})$

1: $y_{\text{init}} \leftarrow (x_{\text{init}}, 0); \mathcal{T}(\mathcal{Y}).\text{init}(y_{\text{init}}); y_{\text{min}} = (\text{NULL}, \infty)$
2: **for** $i = 1$ to k **do**
3: $\quad x_{\text{rand}} \leftarrow \text{SAMPLE}(\mathcal{X})$ ▷ sample state
4: $\quad c_{\text{rand}} \leftarrow \text{SAMPLE}([0, c_{\text{max}}])$ ▷ sample cost
5: $\quad t_{\text{rand}} \leftarrow \text{SAMPLE}([0, T_{\text{prop}}])$ ▷ sample duration
6: $\quad u_{\text{rand}} \leftarrow \text{SAMPLE}(\mathcal{U})$ ▷ sample control
7: $\quad y_{\text{near}} \leftarrow \text{NEAREST}(y_{\text{rand}} = (x_{\text{rand}}, c_{\text{rand}}), \mathcal{T}(\mathcal{Y}))$
8: $\quad (x_{\text{new}}, \pi_{\text{new}}) \leftarrow \text{PROPAGATE}(x(y_{\text{near}}), u_{\text{rand}}, t_{\text{rand}})$
9: $\quad c_{\text{new}} \leftarrow c(y_{\text{near}}) + \text{COST}(\pi_{\text{new}})$
10: $\quad$ **if** COLLISION-FREE(π_{new}) **then**
11: $\quad\quad \mathcal{T}(\mathcal{Y}).\text{add\_vertex}(y_{\text{new}} = (x_{\text{new}}, c_{\text{new}}))$
12: $\quad\quad \mathcal{T}(\mathcal{Y}).\text{add\_edge}(y_{\text{near}}, y_{\text{new}}, \pi_{\text{new}})$
13: $\quad\quad$ **if** $x(y_{\text{new}}) \in \mathcal{X}_{\text{goal}}$ and $c(y_{\text{new}}) < c(y_{\text{min}})$ **then**
14: $\quad\quad\quad y_{\text{min}} \leftarrow y_{\text{new}}$
15: **return** TRACE-PATH($\mathcal{T}(\mathcal{Y}), y_{\text{min}}$)

Claim 1. *Under Assumptions 1,2, the augmented system F is Lipschitz continuous for both of its arguments. That is, $\exists K_u, K_x > 0$ s.t. $\forall\ y_0, y_1 \in \mathcal{Y}, u_0, u_1 \in \mathcal{U}$:*

$$\|F(y_0, u_0) - F(y_0, u_1)\| \leqslant K_u \|u_0 - u_1\|,$$
$$\|F(y_0, u_0) - F(y_1, u_0)\| \leqslant K_x \|y_0 - y_1\|.$$

Proof. It follows that

$$\begin{aligned}
&\|F(y_0, u_0) - F(y_0, u_1)\| \\
&\quad = \sqrt{\|f(x_0, u_0) - f(x_0, u_1)\|^2 + \|g(x_0, u_0) - g(x_0, u_1)\|^2} \\
&\quad \leqslant \sqrt{(K_u^f \|u_0 - u_1\|)^2 + (K_u^g \|u_0 - u_1\|)^2} \\
&\quad = \sqrt{(K_u^f)^2 + (K_u^g)^2} \cdot \|u_0 - u_1\| = K_u \cdot \|u_0 - u_1\|,
\end{aligned}$$

for $K_u := \sqrt{(K_u^f)^2 + (K_u^g)^2}$.

The second inequality requires an additional transition since $\|F(y_0, u_0) - F(y_1, u_0)\| \leqslant K_x \cdot \|x_0 - x_1\|$. It remains to use the fact that $\|x_0 - x_1\| \leqslant \|y_0 - y_1\|$. □

We can think of AO-RRT planning for the system f with cost g, state space $\mathcal{X}$ and control space $\mathcal{U}$, as the standard RRT operating over the system F, state space $\mathcal{Y}$, and control space $\mathcal{U}$. Lines 8-9 in Algorithm 1 are identical to propagating with F. This equivalence allows to exploit useful properties of RRT recently developed in [24].

B. Proof of Theorem 1

We first provide an outline of the proof. Fix $\varepsilon \in (0, 1)$ and let π_δ be a robust trajectory whose clearance is $\delta > 0$. The clearance is with respect to both distance from the obstacles, and from the boundary of the goal region. Let $c_\delta := \text{COST}(\pi_\delta)$. We draw π_δ in the $(d+1)$-dimensional space $\mathcal{Y}$, such that the new trajectory $\pi_\delta^{\mathcal{Y}}$ begins in $y_{\text{init}} = (x_{\text{init}}, 0)$ and ends in $y_{\text{goal}} = (x_{\text{goal}}, c_\delta)$, where $x_{\text{goal}} \in \mathcal{X}_{\text{goal}}$. Next, similarly to [24], we place a constant number of balls of radius $r = \min(\varepsilon c_\delta, \delta)$ along the trajectory $\pi_\delta^{\mathcal{Y}}$. The balls are constructed in a manner that guarantees that each such transition is collision free. Then we show that with high probability AO-RRT will visit all such balls as the number of samples k tends to infinity, by transitioning from one ball to the next incrementally. Reaching the last ball, centered at y_{goal}, implies that AO-RRT will find a solution whose cost is at most $c_\delta + r \leqslant c_\delta + \varepsilon c_\delta = (1+\varepsilon) c_\delta$, since by definition any trajectory in $\mathcal{Y}$ that terminates in $\mathcal{B}_r^{d+1}(y_{\text{goal}})$ must have a cost (which is its $(d+1)$th coordinate) of at most $c_\delta + r$.

To achieve this, we first adapt with minor changes the following two lemmatta from [24] to the setting of AO-RRT. Lemma 1 shows that there exists a constant $\tau \leqslant T_{\text{prop}}$ such that if we place the centers of the balls along $\pi_\delta^{\mathcal{Y}}$ where the duration between two consecutive centers is τ then the probability for successfully propagating from one ball to the next is positive (assuming that the propagation duration and the control input are chosen uniformly at random). We note that it follows from [24] that τ can be chosen such that $\pi_\delta^{\mathcal{Y}}$ can be divided into sub-trajectories of duration τ, where the control function is fixed during each sub-trajectory.

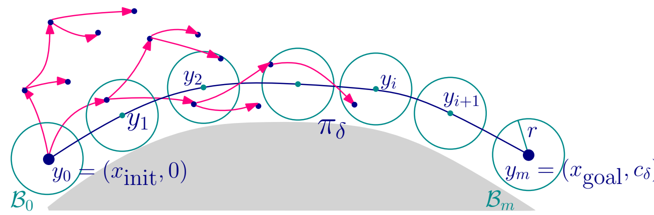

Fig. 1. Illustration of the proof of Theorem 1.

Lemma 1. *There exists $\tau \leqslant T_{\text{prop}}$ for which the following holds: Let π be a trajectory for F with clearance $\delta > 0$ and a control function that is fixed during the interval $[0, \tau]$. Let $r \leqslant \delta$. Let t_{rand} be a random duration sampled uniformly from $[0, T_{\text{prop}}]$, and u_{rand} a uniformly sampled control input from $\mathcal{U}$. Suppose that the propagation step of* AO-RRT *begins at state $y_{\text{near}} \in \mathcal{B}_r^{d+1}(\pi(0))$ and ends in y_{new} (lines 8,9 in Algorithm 1). Then*

$$p_{\text{prop}} := \Pr\left[y_{\text{new}} \in \mathcal{B}_{2r/5}^{d+1}(\pi(\tau))\right] > 0.$$

Lemma 2 shows that the probability that the nearest neighbor of a random sample y_{rand} lies in a specific ball is positive, when y_{rand} is sampled uniformly at random from $\mathcal{Y}$.

Lemma 2. *Let $y \in \mathcal{Y}$ be such that $\mathcal{B}_r^{d+1}(y) \subset \mathcal{F}_{\mathcal{Y}} := \mathcal{F} \times \mathbb{R}_+$. Suppose that there exists an* AO-RRT *vertex $v \in \mathcal{B}_{2r/5}^{d+1}(y)$. Let y_{near} denote the nearest neighbor of y_{rand} among all* AO-RRT *vertices. Then*

$$p_{\text{near}} := \Pr\left[y_{\text{near}} \in \mathcal{B}_r^{d+1}(y)\right] \geqslant |\mathcal{B}_{r/5}^{d+1}|/|\mathcal{Y}|.$$

Note that both probabilities $p_{\text{prop}}, p_{\text{near}}$ are independent of the number k of iterations of the algorithm. Next, we will place $m+1$ balls of radius $r = \min(\varepsilon c_\delta, \delta)$ centered at states $y_0 = y_{\text{init}}, y_1, \ldots, y_m = y_{\text{goal}}$ along the trajectory $\pi_\delta^{\mathcal{Y}}$. See Figure 1 for an illustration.

Denote by t_δ the duration of π_δ. We determine the sequence of points $y_0, \ldots, y_m$ in the following manner: Choose a set of durations $t_0 = 0, t_1, t_2, \ldots, t_m = t_\delta$, such that the difference between every two consecutive ones is τ (see Lemma 1). That is, let $y_0 = \pi_\delta^{\mathcal{Y}}(t_0), y_1 = \pi_\delta^{\mathcal{Y}}(t_1), \ldots, y_m = \pi_\delta^{\mathcal{Y}}(t_\delta)$ be states along the path $\pi_\delta^{\mathcal{Y}}$ that are obtained after duration $t_0, t_1, \ldots, t_m$, respectively. Obviously, $m = t_\delta / \tau$ is some constant independent of the number of samples.

Suppose that there exists an AO-RRT vertex $v \in \mathcal{B}_{2r/5}^{d+1}(y_i) \subset \mathcal{B}_r^{d+1}(y_i)$. We shall bound the probability p that in the next iteration the AO-RRT tree will extend from a vertex in $\mathcal{B}_r^{d+1}(y_i)$, given that a vertex in $\mathcal{B}_{2r/5}^{d+1}(y_i)$ exists, and that the propagation step will add a vertex to $\mathcal{B}_{2r/5}^{d+1}(y_{i+1})$. That is, p is the probability that in the next iteration both $y_{\text{near}} \in \mathcal{B}_r^{d+1}(y_i)$ and $y_{\text{new}} \in \mathcal{B}_{2r/5}^{d+1}(y_{i+1})$. From Lemma 2 we have that the probability that y_{near} lies in $\mathcal{B}_r^{d+1}(y_i)$, given that there exists an RRT vertex in $\mathcal{B}_{2r/5}^{d+1}(y_i)$, is at least p_{near}. Next, we wish to sample duration t_{rand} and control u_{rand} such that a random propagation from y_{near} will yield $y_{\text{new}} \in \mathcal{B}_{2r/5}^{d+1}(y_{i+1})$. According to Lemma 1, the probability for this to occur is at least p_{prop}. Thus, jointly the probability that y_{near} falls in $\mathcal{B}_r^{d+1}(y_i)$ and of sampling the correct propagation duration and control is at least $p = p_{\text{near}} \cdot p_{\text{prop}} > 0$. As we mentioned earlier, this value is also independent of the number of iterations.

It remains to bound the probability of having m successful such steps. This process can be described as k Bernoulli trials with success probability p. The planning problem can be solved after m successful outcomes, where the ith outcome adds an AO-RRT vertex in $\mathcal{B}^{d+1}_{2r/5}(y_i)$. Let X_k denote the number of successes in k trials. As in [24], we have that

$$\Pr[X_k < m] = \sum_{i=0}^{m-1} \binom{k}{i} p^i (1-p)^{k-i} \leqslant a e^{-bk},$$

where a and b are positive constants. This concludes the proof of Theorem 1. In the full version of the paper [26] we show that Theorem 1 is not limited to trajectories with piecewise-constant control functions.

IV. Practical aspects of AO-RRT

We discuss several approaches to potentially speed up the performance of AO-RRT in practice, while retaining its AO property.

Cost sampling. AO-RRT samples $(d+1)$-dimensional points from the augmented space $\mathcal{Y}$ by randomly sampling $\mathcal{X}$ and the cost space $[0, c_{\text{max}}]$, where c_{max} provides an upper bound on the maximal cost of the solution. Setting c_{max} to be much larger than the cost of existing tree vertices may bias the NEAREST procedure towards selecting vertices with high cost, which may affect the time to find an initial solution. Thus, we propose to set c_{max} to be the maximal cost among the tree nodes, until an initial solution is found. Then, we can fix c_{max} to be the cost of the solution.

Augmented-space metric. In some applications the coordinates of the $\mathcal{X}$-component and the cost component in $\mathcal{Y}$ may be on different scales, which can bias NEAREST procedure towards either the cost or the $\mathcal{X}$ component. This in turn may affect the behaviour of the algorithm and its convergence rate. Thus, we propose to use a Weighted Euclidean metric for NEAREST, defined as

$$\text{DIST}(y_a, y_b) := \sqrt{w_x \|x_a - x_b\|^2 + w_c |c_a - c_b|^2}, \qquad (4)$$

where $y_a = (x_a, c_a), y_b = (x_b, c_b) \in \mathcal{Y}$. To avoid biasing, w_x, w_c should be chosen such that the maximal possible squared distance between the $\mathcal{X}$ components and the maximal possible squared distance between costs would be of the same order. Note that this weighted version can be viewed as using an unweighted version on an augmented space $\mathcal{Y}'$ in which the cost coordinate has been rescaled. Thus, the theoretical analysis presented in the previous section holds for the weighted version as-is.

Node pruning. After a solution of some cost $c > 0$ is found, existing tree nodes whose cost-to-come is greater than c cannot participate in the returned solution, or in a solution of better cost. Such vertices can therefore be removed from the tree. We emphasize that the proof of the previous section still applies to this setting, as after pruning, the probability to grow the tree from a certain node whose cost-to-come value is at most c only increases.

Hybrid planning. As the performance of sampling-based planners varies from one scenario to another, we propose a hybrid approach HybAO-RRT, combining AO-RRT with other planners. This approach may perform better in scenarios where AO-RRT struggles to find a solution, and effectively guide the planning and expedite the convergence towards the optimum. HybAO-RRT combines AO-RRT with an additional tree planner, termed PLN, while operating in the augmented space $\mathcal{Y}$. It extends the constructed tree by alternating between AO-RRT and PLN. Each node added to the tree is assigned with a cost value, as in AO-RRT.

HybAO-RRT is AO since by applying AO-RRT every other iteration we still have a positive probability $p' = p/2$ for a successful transition from $\mathcal{B}^{d+1}_i$ to $\mathcal{B}^{d+1}_{i+1}$, for every i. Moreover, the addition of tree nodes due to the steps of the other planner PLN does not affect the transition probability p'. We note that PLN is not required to be AO, nor is it assumed to be PC. This is in the spirit of Multi-Heuristic A* [27], where multiple inadmissible heuristic functions are used simultaneously with a single consistent heuristic to preserve guarantees on completeness of the search.

V. Experimental results

We present an experimental evaluation of the performance of AO-RRT. Our experiments were conducted on an Intel(R) Xeon(R) CPU E5-1660 v33.00GHz with 32GB of memory. We set the optimization objective to be the duration of the trajectory. Throughout the experiments we assign $w_x = w_c$ in the distance metric (Eq. (4)) for all AO-RRT variants. Experiments with different weighting schemes or with additional scenarios are provided in [26].

We compare the algorithms within the AO-RRT framework with RRT [19] and SST [17], which have weaker guarantees. The AO-RRT variants that we used are: (i) Multi-tree AO-RRT—the algorithm analyzed in [18], (ii) AO-RRT—an implementation of Algorithm 1, (iii) AO-RRT Pruning, which performs node pruning, and (iv) HybAO-RRT-STRIDE. The latter is a hybrid planner, as described in Section IV, combining AO-RRT and STRIDE [28]. We mention that STRIDE was originally defined for geometric settings and was not shown to be AO. STRIDE uses a data structure that enables it to produce density estimates in the full state space. More precisely, it samples a configuration s, biased towards relatively unexplored areas of the state space. The tree is then grown from s, if possible. HybAO-RRT-STRIDE maintains, as STRIDE does, a data structure for states in the augmented space $\mathcal{Y}$ and alternates between the two methods for choosing the node y_{near} to grow the tree from. Once the node is chosen, the algorithm proceeds as AO-RRT does.

For each planner we report on both the success rate and the minimum solution cost averaged over all successful runs, displaying values within one standard deviation of the mean. Each result is averaged over 50 runs. Note that since we plot

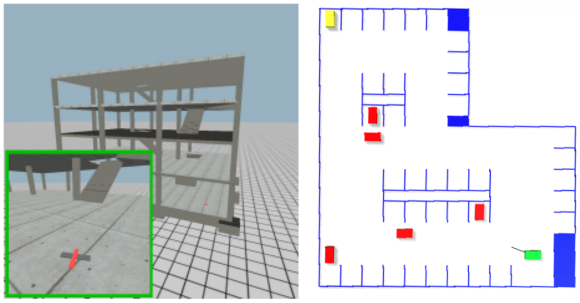

Fig. 2. Scenarios: Fixed-wing airplane in a building (left), and rally car in a parking lot (right).

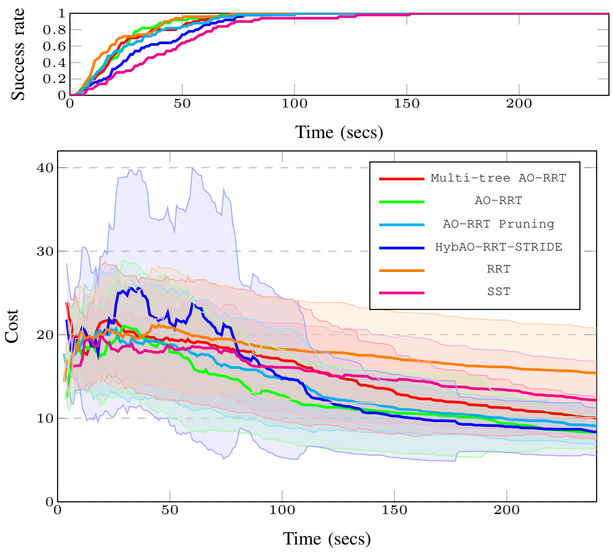

Fig. 3. Plots for a fixed-wing airplane (see Figure 2, left).

the average cost over all successful runs, we may observe for a certain planner an increase in the average cost. This is only possible if the success rate increases as well.

The first scenario involves a fixed-wing 2nd-order airplane moving through a building with tight stairwells to reach the top floor (see Figure 2, left). The state space is nine-dimensional. The task space is the x, y, z location of the fixed-wing airplane. We present the results in Figure 3. An additional scenario (Figure 2, right) involves a rally car (green) moving through a parking lot trying to reach a parking space (yellow) while avoiding other static cars and obstacles. The state space is eight-dimensional, while the task space consists of the 2D pose (x, y, θ) of the car. We present the results in Figure 4.

These two experiments demonstrate that, as expected, all `AO-RRT` variants improve their solution as a function of time. However, single-tree variants within the `AO-RRT` framework perform better than the `Multi-tree AO-RRT` approach. This further justifies the dedicated analysis for the single-tree `AO-RRT`. Additionally, we observe that `AO-RRT` and `AO-RRT Pruning`, differing in the addition of a pruning step, find solutions of similar quality, while the former obtains a slightly better success rate. The variance of the solutions found by the hybrid planner is higher than

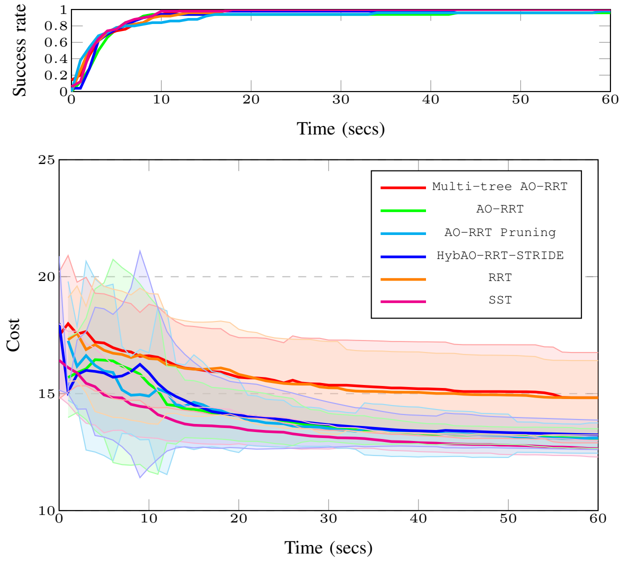

Fig. 4. Plots for a rally car (see Figure 2, right).

that of the other approaches for lower success rates.

Moreover, when compared to `RRT`, which is not AO, all `AO-RRT` variants were able to find solutions of better quality. The comparison against `SST`, which is near-AO, yielded different results; for the fixed-wing scenario all `AO-RRT` variants had better success rates and obtained better costs. For the rally car, all the single-tree `AO-RRT` variants and `SST` found comparable solutions, with a slight advantage to `SST`.

VI. Discussion

This work analyzed the desirable theoretical properties of `AO-RRT`, which is a method for kinodynamic sampling-based motion planning. In particular, relaxed sufficient conditions have been identified for which `AO-RRT` is asymptotically optimal.

Future work will extend the framework to manifold-type constraints. In this case, one has to consider the notion of Riemannian distance instead of the more classic Euclidean distance that we use here.

We noticed that `AO-RRT`'s performance strongly depends on the choice of weights w_x, w_c used by the distance function in Eq. (4). It would be desirable to come up with an automatic scheme to choose these weights, or even modify them on-the-fly in order to obtain favorable results.

Another possible research direction involves a deeper examination of the properties of the hybrid approach. This could shed light on the settings in which the hybrid planner has an advantage over `AO-RRT`.

Finally, the following question, concerning the sampling scheme used by the algorithm, arises from this work: Is it possible to replace the uniform sampling of durations, or states with a different sampling method to converge more quickly to the optimum in the state-cost space?

REFERENCES

[1] L. E. Kavraki and S. M. LaValle, "Motion planning," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds., 2008.

[2] S. M. LaValle, *Planning Algorithms*. Cambridge Univ. Press, 2006.

[3] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *IJRR*, vol. 30, no. 7, pp. 846–894, 2011.

[4] L. Janson, E. Schmerling, A. A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *IJRR*, vol. 34, no. 7, 2015.

[5] O. Salzman and D. Halperin, "Asymptotically-optimal motion planning using lower bounds on cost," in *ICRA*, 2015, pp. 4167–4172.

[6] 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 *ICRA*, 2015, pp. 3067–3074.

[7] K. Solovey and D. Halperin, "Sampling-based bottleneck pathfinding with applications to Fréchet matching," in *ESA*, 2016, pp. 76:1–76:16.

[8] E. Schmerling, L. Janson, and M. Pavone, "Optimal sampling-based motion planning under differential constraints: The drift case with linear affine dynamics," in *CDC*, 2015, pp. 2574–2581.

[9] ——, "Optimal sampling-based motion planning under differential constraints: The driftless case," in *ICRA*, 2015, pp. 2368–2375.

[10] S. Karaman and E. Frazzoli, "Optimal kinodynamic motion planning using incremental sampling-based methods," in *CDC*, 2010.

[11] ——, "Sampling-based optimal motion planning for non-holonomic dynamical systems," in *ICRA*, 2013, pp. 5041–5047.

[12] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Pérez, "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics," in *ICRA*, 2012.

[13] D. J. Webb and J. P. van den Berg, "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics," in *ICRA*, 2013, pp. 5054–5061.

[14] C. Xie, J. P. van den Berg, S. Patil, and P. Abbeel, "Toward asymptotically optimal motion planning for kinodynamic systems using a two-point boundary value problem solver," in *ICRA*, 2015.

[15] G. Papadopoulos, H. Kurniawati, and N. M. Patrikalakis, "Analysis of asymptotically optimal sampling-based motion planning algorithms for lipschitz continuous dynamical systems," *CoRR*, vol. abs/1405.2872, 2014.

[16] Y. Li, Z. Littlefield, and K. E. Bekris, "Sparse methods for efficient asymptotically optimal kinodynamic planning," in *WAFR*, 2014.

[17] ——, "Asymptotically optimal sampling-based kinodynamic planning," *IJRR*, vol. 35, no. 5, pp. 528–564, 2016.

[18] K. Hauser and Y. Zhou, "Asymptotically optimal planning by feasible kinodynamic planning in a state-cost space," *IEEE Trans. Robotics*, vol. 32, no. 6, pp. 1431–1443, 2016.

[19] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *IJRR*, vol. 20, no. 5, pp. 378–400, 2001.

[20] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *IJRR*, vol. 21, no. 3, pp. 233–255, 2002.

[21] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko, *The Mathematical Theory of Optimal Processes*. Wiley, New York, 1962.

[22] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press, June 2005.

[23] T. Kunz and M. Stilman, "Kinodynamic RRTs with fixed time step and best-input extension are not probabilistically complete," in *WAFR*, 2014, pp. 233–244.

[24] M. Kleinbort, K. Solovey, Z. Littlefield, K. E. Bekris, and D. Halperin, "Probabilistic completeness of RRT for geometric and kinodynamic planning with forward propagation," *IEEE RA-L*, pp. 1–1, 2018.

[25] D. M. Ming C. Lin and Y. J. Kim, "Collision and proximity queries," in *Handbook of Discrete and Computational Geometry*, 3rd ed. CRC press, 2018, ch. 51.

[26] M. Kleinbort, E. Granados, K. Solovey, R. Bonalli, K. E. Bekris, and D. Halperin, "Refined analysis of asymptotically-optimal kinodynamic planning in the state-cost space," *CoRR*, vol. abs/1909.05569, 2019.

[27] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, "Multi-heuristic A*," *IJRR*, vol. 35, no. 1-3, pp. 224–243, 2016.

[28] B. Gipson, M. Moll, and L. E. Kavraki, "Resolution independent density estimation for motion planning in high-dimensional spaces," in *ICRA*, 2013, pp. 2437–2443.