Asymptotically Optimal Kinodynamic Planning Using Bundles of Edges

Rahul Shome and Lydia E. Kavraki

***Abstract*— Using sampling to estimate the connectivity of high-dimensional configuration spaces has been the theoretical underpinning for effective sampling-based motion planners. Typical strategies either build a roadmap, or a tree as the underlying search structure that connects sampled configurations, with a focus on guaranteeing completeness and optimality as the number of samples tends to infinity. Roadmap-based planners allow preprocessing the space, and can solve multiple kinematic motion planning problems, but need a steering function to connect pairwise-states. Such steering functions are difficult to define for kinodynamic systems, and limit the applicability of roadmaps to motion planning problems with dynamical systems. Recent advances in the analysis of single-query tree-based planners has shown that forward search trees based on random propagations are asymptotically optimal. The current work leverages these recent results and proposes a multi-query framework for kinodynamic planning. Bundles of kinodynamic edges can be sampled to cover the state space before the query arrives. Then, given a motion planning query, the connectivity of the state space reachable from the start can be recovered from a forward search tree reasoning about a local neighborhood of the edge bundle from each tree node. The work demonstrates theoretically that considering any constant radial neighborhood during this process is sufficient to guarantee asymptotic optimality. Experimental validation in five and twelve dimensional simulated systems also highlights the ability of the proposed edge bundles to express high-quality kinodynamic solutions. Our approach consistently finds higher quality solutions compared to `SST`, and `RRT`, often with faster initial solution times. The strategy of sampling kinodynamic edges is demonstrated to be a promising new paradigm.**

I. Introduction

Sampling-based roadmaps [1] have been a popular and effective strategy for motion planning in high-dimensional spaces. Initial advances focused on the properties of probabilistic completeness to asymptotically guarantee the discovery of a solution to the motion planning problem, if one exists. Tree-based methods [2] construct tree data structures rooted at the start of the motion planning problem, as opposed to connected roadmaps. Typically roadmap-based methods [1] are multi-query frameworks [3] that can reuse the same graphical structure across different motion planning queries in the same scene. Tree-based methods are single-query, and rooted at the start state of the current problem. Roadmaps have a limitation arising from the neighborhood connectivity required for the graph structure. Such precise connections require the robotic system to be *steerable*, or in other words, the local planner needs access to a *steering function* that can connect pairs of nearby states. Such steering functions are typically not available for systems with complex dynamics, and therefore limit the use of roadmaps-based multi-query methods. For the past two decades, tree-based planners [4]–[6] have therefore been the mainstay in kinodynamic motion planning. Tree edges can be *forward propagated* to grow outwards into the space, without needing to precisely hit any state. Some previous work [5] has focused on sampling path segments, instead of tree nodes within planning. Nonetheless, since the tree is rooted at the start of the motion planning problem, in most cases, for a new problem, the tree is reconstructed from scratch and all the previous computation is discarded.

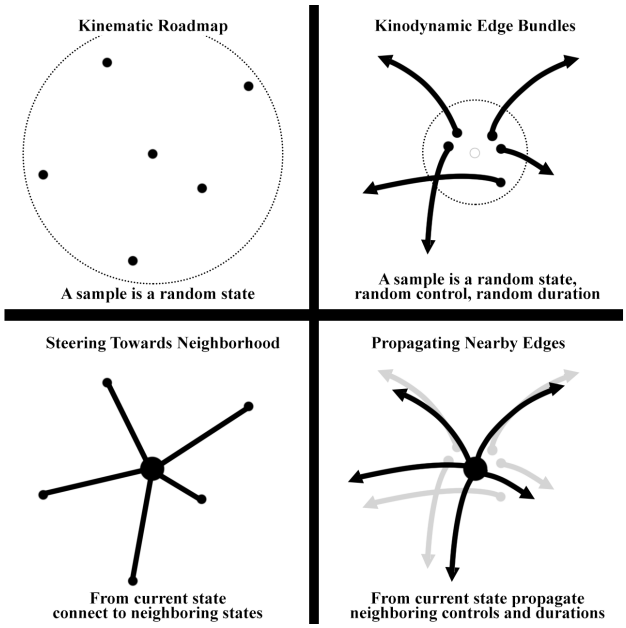

Fig. 1. The key idea of sampling edges instead of configurations, and propagating nearby edges instead of steering towards neighborhoods.

Recently, there has been a push beyond probabilistic completeness into *asymptotic optimality*, which affords guarantees of the motion planner to converge to the optimal cost in addition to finding a solution. The original work [7] and subsequent improvements [8] used results from random geometric graph literature to provide meticulous bounds on how to design neighborhoods and connection strategies for both roadmap and tree-based methods in kinematic problems. Extensions to kinodynamic domains relied on assumptions about the availability of steering [9]–[11]. Eventually a kinodynamic asymptotically optimal planner [12] was devised based off a naive random forward search tree, that forsook steering. It was shown that the theoretical property of asymptotic optimality is guaranteed as long as enough *Monte Carlo propagations* (random controls and durations) are added to the search tree. More recent extensions have

The authors are affiliated with the Dept. of Computer Science, Rice University. email:{rahul.shome, kavraki}@rice.edu. The authors have been supported in part by NSF 2008720 and Rice Academy of Fellows.

refined the understanding of the properties of asymptotically optimal kinodynamic planning [13]–[15].

The current work poses the question: *can we attain benefits from a multi-query algorithmic framework in the kinodynamic domain?*

Our work proposes sampling a set of kinodynamic edges for a new environment. Each edge comprises of a sampled state, from where a sampled control is propagated for a sampled duration. These edges represent a *kinodynamic edge bundle*. When a new motion planning query arrives, a forward search tree is grown from the start state. The out-edges from each tree node can be inferred from the precomputed edge bundle. Bundle edges starting within a neighborhood of the tree node, parameterized by radius θ, are propagated from the tree node to grow the search tree. This is visualized in Fig 1. For new motion planning queries in the same scene, the edge bundle can be reused.

It should be noted that the edges in the bundle are independent of each other and are completely disconnected. This makes their computation relatively straightforward. Most of the computational overhead is delegated to the forward search over the bundle of edges, that attempts to reconstruct a solution from the start state, to the goal region.

We leverage the theoretical tools outlined in these recent lines of work, for our analysis. Our chief contribution in Lemma 4 is to show that as long as the θ used for computing neighboring edges remains a positive constant, as the number of edges in the *pre-sampled bundle* increases, solutions that are arbitrarily close to the optimal can be recovered from the edge bundle. This is the key component in our proof of asymptotically optimality. Beyond the theoretical investigation, our experimental results indicate that bundles of sampled edges can express higher quality solutions, faster than competing methods. This motivates our algorithmic paradigm of *sampling edges instead of sampling configurations.*

II. Problem Setup

The problem formulation and associated assumptions are mostly identical to those set up in previous work [12], [15].

The workspace $\mathcal{W}$ contains a robot and obstacles. A robot can be described in terms of a d-dimensional state $x \in \mathcal{Q} \subset \mathbb{R}^d$, and D-dimensional controls $u \in \mathcal{U} \subset \mathbb{R}^D$. It should be noted that the state space in kinodynamic problems typically consists of a geometric component like positions or angles, and higher order dynamical components like velocities and accelerations. Each of these add to the d dimensions. A collision-free subset of the state space $\mathcal{Q}_{\text{free}} \subseteq \mathcal{Q}$ does not induce intersections between the robot and the obstacles.

Consider problem domains where the changes to the robotic system can be described in terms of differential equations of the following form (where t denotes time):

$$\dot{x} = f(x(t), u(t)), \quad x \in \mathcal{Q}, u \in \mathcal{U} \tag{1}$$

*Assumption 1 (**Topology**):* It is assumed that $\mathcal{Q}_{\text{free}}$ is locally Euclidean, and admits the Euclidean norm ($\|\cdot\|$), and hyperball regions $\mathcal{B}_r(x)$ of radius r at state x.

*Assumption 2 (**Small-time local controllability** [12]-5):* The system in Eq 1 is assumed to be small-time locally controllable, with a bounded second derivative, and Lipschitz continuous in both x and u.

A **control function** $\mathcal{Y} : [0, T] \rightarrow \mathcal{U}$ maps timestamps within the range $[0, T]$ to a control. When starting from a start state x_0, a valid trajectory $\pi : [0, T] \rightarrow \mathcal{Q}_{\text{free}}$ is obtained by forward-integrating Eq 1 from $x_0 = \pi(0)$.

Trajectories can be defined using **piecewise-constant** control functions $\hat{\mathcal{Y}}$ with resolution Δt such that $\hat{\mathcal{Y}} : i \rightarrow \mathcal{U}, \quad \mathcal{Y}(t) = \hat{\mathcal{Y}}(\lfloor \frac{t}{\Delta t} \rfloor), \quad i \in \mathbb{Z}^+$, where i is the indexing.

We define ε-similar trajectories π, π^ε when $\pi^\varepsilon(t) \in \bigcup_t \mathcal{B}_\varepsilon(\pi(t))$. We will also denote this phenomenon as the trajectory π^ε *observing* π. It is also assumed that we restrict the analysis to trajectories that have a positive clearance δ from obstacles. A valid δ-**robust trajectory** π means that $\bigcup_t \mathcal{B}_\delta(\pi(t)) \subset \mathcal{Q}_{\text{free}}$.

The cost of the trajectory of duration T is assumed to be additive, and real-valued over a trajectory, and can be defined as: $\text{cost}(\pi) : \pi \rightarrow \mathbb{R}_{\geq 0}$. Let us assume this cost to be the Euclidean arc length along the trajectory.

Similarly distances can be described as the Euclidean norm between points in the state and control space as $\|x_0 - x_1\|$ or $\|u_0 - u_1\|$ respectively. The distance between trajectories can be defined by the maximum deviation along configurations over two trajectories $\sup_{\forall t}\{\|\pi_0(t) - \pi_1(t)\|\}$.

*Assumption 3 (**Smooth Cost Function**):* The cost function defined over a trajectory is assumed to be Lipschitz continuous with respect to both x and u.

From Assumptions 2 and 3, there exists smoothness constants [12], [14], [15] for the underlying dynamics function f and two constant-control trajectories (π_0, and π_1), such that the following bounds hold for some $K_u, K_x, K_c > 0$:

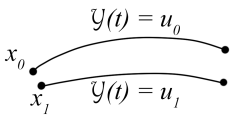

Fig. 2. Lipschitz smoothness.

$$\|f(x_0, u_0) - f(x_0, u_1)\| \leq K_u \|u_0 - u_1\| \tag{2}$$

$$\|f(x_0, u_0) - f(x_1, u_0)\| \leq K_x \|x_0 - x_1\| \tag{3}$$

$$\text{cost}(\pi_0) - \text{cost}(\pi_1) \leq K_c \sup_{\forall t}\{\|\pi_0(t) - \pi_1(t)\|\} \tag{4}$$

The smoothness bounds essentially imply that there exists small enough scales of the space wherein *close enough starting configurations and controls* executed for *similar time durations* will result with trajectories that end in *close enough ending configurations*, and have *similar costs*.

Definition 1 (Motion Planning Problem): Given a starting state $x_0$and a goal region $\mathcal{Q}_{goal} \in \mathcal{Q}$, the motion planning problem is defined by $(x_0, \mathcal{Q}_{goal}, \mathcal{Q}_{\text{free}})$, and admits a feasible solution π. A feasible motion planning solution π connects the starting state $x_0 = \pi(0)$ and ends in a goal region $\mathcal{Q}_{goal}$ such that $\pi(T) \in \mathcal{Q}_{goal}$.

Definition 2 (Optimal Motion Planning): Given a motion planning problem, an optimal solution $\pi^{\texttt{OPT}}$ is a feasible solution with the lowest cost, $\pi^{\texttt{OPT}} = \underset{\pi}{argmin}\ \text{cost}(\pi)$.

*Assumption 4 (**Robust Optimal Motion Planning**):* It is assumed that there exists a small enough $\delta' > 0$ such that all

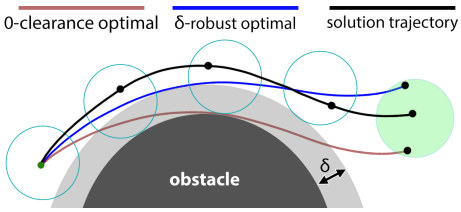

Fig. 3. δ-robust optimal motion planning to a goal region (green). For some small $\delta > 0$, the robust optimal stays δ away from the obstacle, and can be *observed* by a solution in its neighborhood (blue hyperballs).

values of clearance in the range $\delta \in (0, \delta']$ admit solutions π^* that are at least δ-robust.

It is assumed that the feasible δ-robust optimal solution exists, and can be approximated by a piecewise-constant control function. This is illustrated in Fig 3. This property is necessary for any sampling-based technique to have fully dimensional voluminous regions where good enough solutions can exist, and can be sampled, Previous work [15] has shown that this approximation is guaranteed even for feasible solutions that are not piecewise-constant controlled.

Definition 3 (Asymptotic Optimality): Consider an optimal motion planning solution $\pi^{\texttt{OPT}}$ with a corresponding cost $c^{\texttt{OPT}}$, and an algorithm that reports a solution π_n after n iterations, the algorithm is called asymptotically optimal if the solution cost can be shown to converge to the optimal.

$$\lim_{n\to\infty} Pr(\{\text{cost}(\pi_n) < (1+\epsilon)c^{\texttt{OPT}}\}) = 1, \quad \forall \epsilon > 0 \qquad (5)$$

Specifically the statement says that the event that the reported solution cost has its suboptimality contained within any arbitrarily small error ϵ is assured as n is allowed to increase.

For the purposes of our arguments we will focus on π^*, as it allows the existence of volumes around it for analysis. Hereafter we will refer to some robust π^* as the robustly optimal solution we care about. [1]

III. Algorithm

The approach is outlined in Algo 1. The method first covers the state space with n valid sampled edges in the loop over Lines 3-4. The SampleEdge subroutine is invoked to augment $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with additional edges, creating a disconnected bundle of edges. SampleEdge first randomly samples a state, and then performs a *MonteCarloProp* sampling similar to previous work [12] by randomly sampling a control and a duration. The edge trajectory is described by the combination of sampled configuration, control, and duration using forward propagation in SimulateControl on Line 4. The edge is retained only if it is collision-free. This is checked by the ValidEdge subroutine. After the high level loop of Algo 1, there now exists a bundle of valid kinodynamic edges connecting different parts of the space. The larger the number of samples, the denser this coverage will be. An illustration of such an edge bundle is shown in Fig 4(*left*). It should be noted that $\mathcal{G}$ *is expected to be fully disconnected.* since no care has been taken thus far in ensuring any neighborhoods are connected.

[1] Typically arguments of converging in cost for all δ-robust optimal solutions rely on δ being made arbitrarily small. This way the error stays similarly bounded w.r.t. $\pi^{\texttt{OPT}}$ as in Def 3.

Algorithm 1: ConstructEdgeBundle

Input: Number of samples n, Start x_0, Goal Region $\mathcal{Q}_{goal}$, Search Neighborhood θ
Output: Path π

1 $\mathcal{G}(\mathcal{V}, \mathcal{E}) \leftarrow (\emptyset, \emptyset)$;
2 $\pi \leftarrow \emptyset$;
3 **for** $1 \dots n$ **do**
4 | SampleEdge($\mathcal{G}$);
5 $\pi \leftarrow$RetracePath(x_0,$\mathcal{Q}_{goal}$,$\mathcal{G}$,θ);
6 **return** π

Algorithm 2: SampleEdge

Input: Graph $\mathcal{G}$

1 $v_{new} \leftarrow$ SampleState();
2 $\mathcal{E}_{new} \leftarrow \emptyset$;
3 $u_{\text{new}}, t \leftarrow$ SampleControlAndDuration(v_{new});
4 $e_{\text{new}} \leftarrow$ SimulateControl(v_{new}, u_{new}, t);
5 **if** ValidEdge(e_{new}) **then**
6 | $\mathcal{G} \leftarrow (\mathcal{V} \cup v_{\text{new}}, \mathcal{E} \cup e_{\text{new}})$;

Algorithm 3: RetracePath

Input: Start x_0, Goal Region $\mathcal{Q}_{goal}$, Graph $\mathcal{G}$, Search Neighborhood θ
Output: Path π

1 $\pi \leftarrow \emptyset$;
2 $\mathbb{Q} \leftarrow x_0$;
3 **for** $\mathbb{Q} \neq \emptyset$ **do**
4 | $v \leftarrow$ Select($\mathbb{Q}$);
5 | **if** $v \in \mathcal{Q}_{goal}$ **then**
6 | | **if** cost(π) > cost(PathTo(v)) **then**
7 | | | $\pi \leftarrow$PathTo(v);
8 | $\mathcal{V}_{\text{near}} \leftarrow$RadialNN($v, \theta, \mathcal{G}$);
9 | $\mathcal{V}_{\text{next}} \leftarrow \emptyset$;
10 | **for** $v_{\text{near}} \in \mathcal{V}_{\text{near}}$ **do**
11 | | $u_{\text{near}}, t \leftarrow$GetControl(OutEdge($v_{\text{near}}$));
12 | | $e_{\text{new}} \leftarrow$SimulateControl($v, u_{\text{near}}, t$);
13 | | **if** ValidEdge(e_{new}) **then**
14 | | | $\mathcal{V}_{\text{next}} \leftarrow \mathcal{V}_{\text{next}} \cup e_{\text{new}}$;
15 | $\mathbb{Q} \leftarrow$Add($\mathbb{Q}, \mathcal{V}_{\text{next}}$);
16 **return** $\emptyset$

Since $\mathcal{G}$ is disconnected the onus of reconstructing a usable solution trajectory lies on the search. A high-level tree search is proposed in Algo 3, where neighbor expansions are obtained by accumulating all the edges in $\mathcal{G}$ that start within a parameterized θ-neighborhood of a tree node. RetracePath maintains a search queue $\mathbb{Q}$. The minimal guarantee required from the Select operation is to ensure *every feasible node that can lead to the optimal solution in $\mathbb{Q}$ is given an opportunity for expansion*. The expansion set is computed from all the nodes in $\mathcal{G}$ that lie in the θ-neighborhood of the selected node. These controls and durations are propagated from the current node to obtain $\mathcal{V}_{\text{next}}$, which is added to the queue. The loop updates the solution when the goal region is reached in Lines 5-7, and exits when the queue is empty. A retraced solution is shown in Fig 4(*right*). Some implementation choices for Algo 3, relevant to practical performance, are outlined in Sec V.

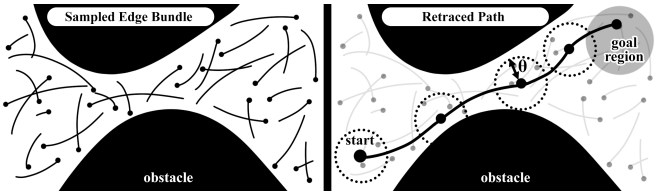

Fig. 4. An illustration of the underlying idea of the proposed method. The figure on the left shows a bundle of disconnected collision-free edges (random state, random control, random duration) . The figure on the right shows a motion planning problem from the start to a goal region (gray). Controls and durations from the bundle edges starting within the dotted θ neighborhoods can be reused to create the sequence of black edges that reaches the goal region.

IV. Analysis

The current set of arguments reason over the invocation of Algo 1 for an input n, and then calling Algo 3 to retrace a solution. The idea is that the set of n edges sampled using SampleEdge needs to sufficiently recover the connectivity of different regions in the free space. These edges are kinodynamic controls, and are not connected to other specific samples. The RetracePath subroutine uses θ-neighborhoods to propagate sampled valid controls from nearby parts of the state space. *It needs to be proven that retracing along such sampled edges can connect the start to the goal region, while converging to the optimal cost as the number of sampled edges increases.*

The proof first lays out the theoretical tools required, primarily those set up in previous work [12], [14], [15]. The rest of the section will go into the details.

Lemma 1 (Near-optimal construction): Given π^* there exists a finite sequence of bounded regions $\mathcal{Q}_0, \mathcal{Q}_1, \cdots \mathcal{Q}_M$ along the trajectory where $\pi^*(0) \in \mathcal{Q}_0, \pi^*(1) \in \mathcal{Q}_M$, and M is a constant independent of n, such that a feasible motion planning solution π crossing these regions in sequence has a bounded error in cost from c^*, i.e., $\text{cost}(\pi) \leq (1+\epsilon)c^*$. Such a construction exists from every value of $\epsilon \in (0, \epsilon_0]$ for some positive value of ϵ_0.

Proof: This follows from arguments made in [12] Sec 5.1. The near-optimality arises from the smooth cost function. For each of the $M-1$ *segments* connecting consecutive regions, the robust optimal trajectory executes a constant control function. If π has a constant control along the same pair of regions (of maximum radius δ), then cost smoothness dictates the trajectories differ at most by $K_c\delta$. This error can accumulate over all $M-1$ segments, leading to $\text{cost}(\pi) = c^* + (M-1)K_c\delta$. There exists some small enough value of δ such that it must be that $(M-1)K_c\delta \leq \epsilon c^*$. The result immediately follows. The following arguments show that such a finite sequence of regions can be connected by constant-control trajectories for any arbitrarily small value of δ. *This would allow the cost of the solution to be arbitrarily close to the optimal cost.* ■

Lemma 2 (Probability of Connecting Regions): The probability of connecting across the regions $\mathcal{Q}_{i=1\cdots M}$ can be expressed in terms of the event of sampling a state within one of the regions $\mathcal{Q}_i$, and sampling a transition between consecutive regions, i.e., an edge $e_i(u,v)$ such that $u \in \mathcal{Q}_i, v \in \mathcal{Q}_{i+1}$. Let the probabilities be Pr^i_{select} and Pr^i_{extend} respectively. If $Pr^i_{select} > 0, Pr^i_{extend} > 0, Pr^i_{select} \perp\!\!\!\perp n, Pr^i_{extend} \perp\!\!\!\perp n \quad \forall i$, i.e., both probabilities are strictly positive, and independent of n, then the event attached to sampling a sequence of the edges connecting across the regions happens asymptotically almost surely.

Proof: The arguments follow those laid out in previous work [16] Theorem 11.3. Construct an absorbing Markov chain corresponding to the events of finding a sequence of samples and controls that connect up to the i'th region $\mathcal{Q}_i$, where the absorbing state is the final region attached to the goal $\mathcal{Q}_M$. With probability $p_i = Pr^i_{select} \times Pr^i_{extend}$ describing the transition probability between the nodes i and $i+1$ in the chain. As long as $p_i > 0, p_i \perp\!\!\!\perp n, \forall i$ the process is asymptotically assured to reach the absorbing state as the number of trials (samples and edges in this case) n increases. Note that the independence of these geometric probability measures from n is dictated by [16] Theorem 11.3. ■

It should be noted that the arguments in Lemma 2 work out because the construction (number and volume of regions in the construction) is independent of n. This principle has been heavily used in similar arguments in previous work [14], [15], [17], [18]. This is a departure from the evolving constructions derived from Random Geometric Graph literature [7], [8], where the number of regions increases, and radius of hyperball regions reduce asymptotically with n.

Lemma 3 (Sampling Edges to Connect Regions): Given two consecutive regions along a constant control ($\mathcal{Y}(t) = u$) segment along π^*, represented by $\mathcal{B}_{\kappa\delta}(x_i)$, and $\mathcal{B}_{\kappa\delta}(x_{i+1})$, $\forall \kappa \in (0,1)$, starting at a state $x'_i \in \mathcal{B}_{\kappa\delta}(x_i)$, the probability of sampling a random control u' and random duration T' that ends up at some $x'_{i+1} \in \mathcal{B}_{\kappa\delta}(x_{i+1})$ is positive, and independent of n.

Proof: The construction of the edge is shown in red in Fig 5 (top). The proof is a slight restatement of arguments made in [15] Lemma 3, and follows due to the Lipschitz bounds. Essentially, *there exists a positive volume of constant controls, and a positive range of durations, selecting* ***any combination of which*** *almost surely connects between $\mathcal{B}_{\kappa\delta}(x_i)$, and $\mathcal{B}_{\kappa\delta}(x_{i+1})$, for any $\kappa \in (0,1)$.* These edges are generated almost surely via Algo 2. ■

Lemma 3 demonstrates that an edge bundle comprising of randomly sampled edges will almost surely discover an edge starting from the interior of the i^{th} $\kappa\delta$ ball, and reach the $i+1^{th}$ ball in the construction, as n increases, for all i. One such edge from the bundle might look like the red edge from Fig 5 (*top*). This only shows that consecutive regions have an edge connecting them, while a motion planning solution requires retracing through exact states. Algo 3 describes a forward search tree that can use the bundle for forward propagating each tree node using controls and durations from bundle-edges arising within a *θ-neighborhood*. Such a tree node x_i^{tree} arriving inside the i^{th} region is shown in Fig 5.

It is shown that propagating the control and duration of the red edge from x_i^{tree} ***is guaranteed to reach the*** *$i+1^{th}$* ***ball****, and that the red edge is almost surely sampled in the bundle* ***within θ-neighborhoods of*** *x_i^{tree}.*

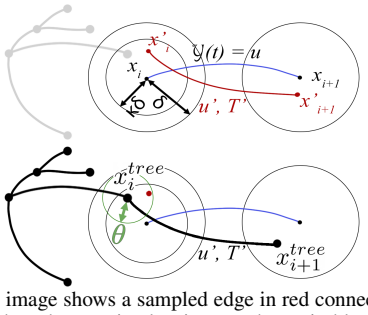

Fig. 5. The top image shows a sampled edge in red connecting consecutive δ regions tiling the robust optimal trajectory shown in blue. The tree shown in black enters the i'th region. The bottom image shows the search tree that can propagate the red edge to connect to the next region.

Lemma 4 (Existence of θ neighborhoods): Given a state x_i^{tree} from a search tree expansion reaching $\mathcal{B}_\delta(x_i)$, there is a region $\mathcal{B}_\theta(x_i^{tree})$ for some $\theta > 0$ such that asymptotically almost surely there would exist some constant controls and durations starting from $\mathcal{B}_\theta(x_i^{tree})$, that connected to $\mathcal{B}_\delta(x_{i+1})$, which when propagated from $\mathcal{B}_\theta(x_i^{tree})$ will also reach $\mathcal{B}_\delta(x_{i+1})$.

Proof: We focus on the i^{th} constant control segment of the robust optimal trajectory π^*, defined by two states on the trajectory x_i, and x_{i+1}. Two collision free open balls $\mathcal{B}_\delta(x_i)$, and $\mathcal{B}_\delta(x_{i+1})$ are centered at the ends of the segment, both of which lie in $\mathcal{Q}_{\text{free}}$. We are interested in tracing through these regions along the search tree constructed in Algo 3. Consider a tree node x_i^{tree} that reached the interior of $\mathcal{B}_\delta(x_i)$.

$$\exists \kappa \in (0,1),\ x_i^{tree} \in \mathcal{B}_{\kappa\delta}(x_i) \tag{6}$$

According to Lemma 3, from any state inside $\mathcal{B}_{\kappa\delta}(x_i)$, some control u' and time duration T' can be sampled that reaches a state $x'_{i+1} \in \mathcal{B}_{\kappa\delta}(x_{i+1})$. This means that the edge bundle is assured to possess such edges as n increases.

Now, given the state x_i^{tree}, and a region defined by some positive $\theta > 0$, we need to show that an edge can be sampled in the edge bundle characterized by a state x'_i, a control u', and a duration T', such that a) $x'_i \in \mathcal{B}_\theta(x_i^{tree})$, and b) when u', T' is propagated from x_i^{tree}, an end state $x_{i+1}^{tree} \in \mathcal{B}_\delta(x_{i+1})$ is reached. We only choose θ-near edges for propagation, so

$$x'_i \in \mathcal{B}_\theta(x_i^{tree}),\ \|x_i^{tree} - x'_i\| < \theta \tag{7}$$

The sampled edge in the bundle x'_i, u', T' ends at $x'_{i+1} \in \mathcal{B}_{\kappa\delta}(x_{i+1})$. When propagating this edge from x_i^{tree}, the end state x_{i+1}^{tree} will be different from x'_{i+1}. We need to show that there exists some θ, within which propagating edges connecting to the next $\kappa\delta$ ball keeps $x_{i+1}^{tree} \in \mathcal{B}_\delta(x_{i+1})$. From the smoothness of the system, we can characterize the maximum deviation of the end state. From [14](*Lemma 2*), we get for a constant control segment of duration T' between time parameters t_i and $t_i + T'$

$$\|\pi(t_i + T') - \pi'(t_i + T')\| \leq e^{K_x T'} \|\pi(t_i) - \pi'(t_i)\| \tag{8}$$

Given a trajectory generated from a control u' and time duration T' sampled from some $x'_i \in \mathcal{B}_\theta(x_i^{tree})$, if we use the same control function from x_i^{tree}, using Eq. 7 and 8:

$$\|x_{i+1}^{tree} - x'_{i+1}\| \leq e^{K_x T'} \|x_i^{tree} - x'_i\| \leq e^{K_x T'} \theta \tag{9}$$

By construction the intersecting volume between the θ-neighborhood and $\kappa\delta$-ball is a positive constant, i.e.,

$$\mu(\mathcal{B}_{\kappa\delta}(x_i) \cap \mathcal{B}_\theta(x_i^{tree})) > 0 \tag{10}$$

This means edges can be sampled in the bundle that start inside this positive volume region. Consider specifically $x'_i \in \mathcal{B}_{\kappa\delta}(x_i) \cap \mathcal{B}_\theta(x_i^{tree})$, then if

$$0 < \theta < \frac{(1-\kappa)\delta}{e^{K_x T'}} \tag{11}$$

$$\implies \|x_{i+1}^{tree} - x'_{i+1}\| < (1-\kappa)\delta \tag{12}$$

$$\implies x_{i+1}^{tree} \in \mathcal{B}_\delta(x_{i+1}) \tag{13}$$

The first implication follows from Eq 9, and the second result follows from Eq 6. This means, for the choice of θ, there is a positive volume of the state space where x'_i can be sampled, and a positive volume of controls and durations to sample u', T', such that when propagated from some x_i^{tree}, allow the search tree to reach $\mathcal{B}_\delta(x_{i+1})$. The arguments hold true for any tree node that reaches the interior of $\mathcal{B}_\delta(x_i)$, and applies to all i. It is important that all the volumes are independent of n, and arise from the smoothness of the system. It should also be noted that the theoretical bounds obtained here for the neighborhoods are only useful to the extent of showing that these volumes exist, and are positive constants independent of n. In practice, any positive $\theta > 0$ works. ■

We now have all the tools for our final arguments.

Theorem 1 (Asymptotic Optimality): Consider the optimal robust motion planning solution π^* with a corresponding cost c^*. The solutions discovered by the algorithm π_n from an edge bundle of size n is shown to asymptotically converge to the optimal cost with an arbitrarily small error.

$$\lim_{n\to\infty} Pr(\{\text{cost}(\pi_n) < (1+\epsilon)c^*\}) = 1, \quad \forall \epsilon > 0 \tag{14}$$

Proof: We reuse the observations from previous work [12], which were further refined [15]. In order to use Lemma 2, $p_i = Pr^i_{select} \times Pr^i_{extend}$ needs to be shown to be positive and independent of n. Lemma 4 shows that Algo 3 will connect between the regions as long as a desired x'_i has been sampled, along with the corresponding controls and durations. By Eq.10, the probability of randomly sampling a desired sample x'_i is positive (i.e., $Pr^i_{select} > 0$). Lemma 3 holds for some range of values of controls and time durations from such a state and the probability of sampling them using Algo 2 is also positive (i.e., Pr^i_{extend}). This holds for all regions, and small δ. Constructing the chain of events from Lemma 2 over the construction, π_n is guaranteed to trace the sequence of regions asymptotically. From Lemma 1 for a small enough δ, we can conclude that $\lim_{n\to\infty}$ cost$(\pi_n) \leq (1+\epsilon)$cost$(\pi^*)$, proving Theorem 1. ■

V. Evaluation

This section evaluates the proposed algorithm on simulated benchmarks, and outlines specific implementation choices for the forward search tree (Algo 3). The primary intent here is to highlight that kinodynamic motion planning solutions can

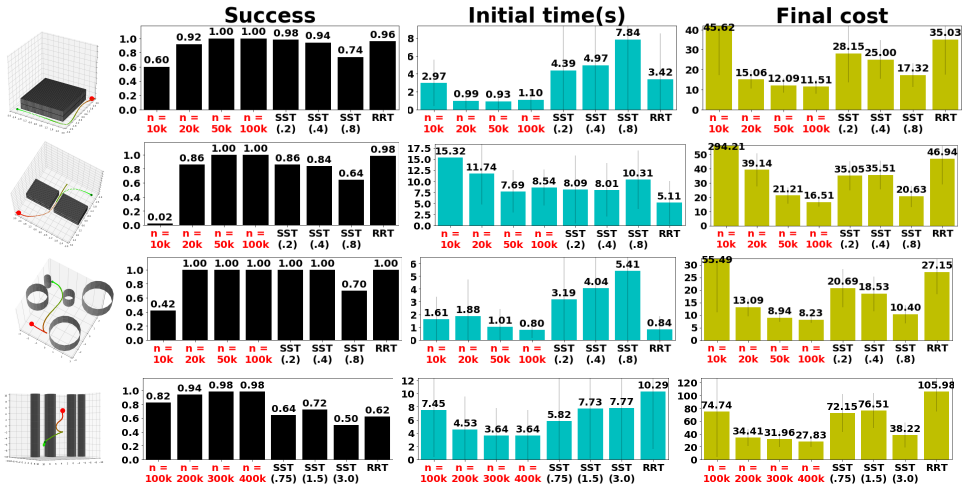

Fig. 6. **Benchmarks (1-4)** The compared methods noted in along the X axis of the plots include different versions of our algorithm using edge bundles of size n, SST with the associated pruning parameter, and RRT. *Top row:* A second order car with an obstacle blocking the center of the space. *Second row:* A second order car with a narrow passage in the center of the space that has to be traversed. *Third row:* A second order car with randomly positioned circular obstacles. *Bottom row:* A quadrotor benchmark with randomly positioned cylindrical obstacles. **Columns:** *Left:* Scenes and visualization of a solution for each benchmark. *Second:* Success rates over 50 runs. *Third:* Time taken to find the initial solution. *Right:* The final solution cost after 30s.

be recovered from edge bundles, and increasing the size of the bundle improves success and solution quality.

Implementation Choices: Algo 3 performs a search over the bundle of edges in a forward tree search, where the branching factor is determined by the number of edges arising within θ-neighborhoods. Performing this naively quickly leads to an explosion in the size of the search queue. Choices are made for the selection, and node addition strategies. For selection, an A^*-like prioritization of *f-values* is used. When a child node improves the heuristic estimate, it would be immediately selected, to greedily make progress [18], [19]. Also, an analog to goal biasing selects the nearest unexplored node from the goal for a fraction of iterations [12]. In typical runs the queue will keep growing. It is important to keep the size of the queue small at any time. A way to do this is employ a cost-to-go based pruning strategy similar to SST [12]. A key difference is that instead of eliminating expansions, they are deferred till the current queue becomes empty, at which point they are added back to the queue.

Comparison Points: Our approach is tested with values of *n=10k up to 400k*, for a constant θ in each benchmark. As a comparison SST is used with three pruning radii that are $0.5\times, 1\times, 2\times$ the pruning used in our method. The SST selection radius is set to $1.5\times$ of the pruning value. We also test RRT, which terminates with the first solution.

Benchmarks: In the benchmarks obstacles are generated in the translational component of the state space. Valid start states, and goal regions are also sampled. Three state spaces are created for the second order car (5-dimensional, $\theta = 0.5$). In *Benchmark 1* a geometric square obstacle is placed in the state space. In *Benchmark 2* a narrow passage separates two halves of the state space. Valid start and goals are sampled at random on opposite sides. In *Benchmark 3* circular obstacles are sampled at random. *Benchmark 4* is modeled with a quadrotor state space (12-dimensional, $\theta = 2$). It has sampled cylindrical state space obstacles. Illustrative examples of the translational projection of the state spaces are shown in Fig 6(left), along with a candidate solution trajectory.

TABLE I

EDGE BUNDLE CONSTRUCTION TIME(S)

Benchmark	10k	20k	50k	100k	200k	300k	400k
1	3.36	7.17	18.18	35.89	—	—	—
2	3.20	6.56	16.18	32.30	—	—	—
3	2.94	6.43	16.22	32.28	—	—	—
4	—	—	—	23.66	47.57	71.46	95.29

The results (Fig 6) show the success rates, initial solution times, and final solution costs after 30s. As the number of edges in the bundle increases, for the same θ, the success increases, and final solution cost decreases. Larger edge bundles outperform both SST and RRT. Initial solution times are typically faster than SST, and in some cases, even faster than RRT. Our planner provides much better trajectories after 30s. This emphasizes the benefits of our strategy. It should be noted that the timings do not include bundle construction (Tables I). The results show trade-offs in the precomputation time versus the perks of reusing the bundle edges.

VI. DISCUSSION

The current work has proposed theoretical arguments in support of a novel paradigm for asymptotically optimal sampling-based kinodynamic planning. The key insight is that complex kinodynamic state spaces can be processed before the search by covering the space with valid randomly sampled edges. Such a bundle of edges is theoretically sufficient to recover asymptotically optimal paths as the number of edges increases if a forward search tree is constructed by generating child nodes propagated using controls and durations arising from edges within a θ-neighborhoods. Experiments show benefits in terms of success rates, solution times and quality of solutions wrt. competing methods. Future work can enhance the efficiency of the search in the context of other methods, explore parallelization of the edge sampling, and incorporate learning. We plan to explore this paradigm to address real-world kinodynamic applications.

REFERENCES

[1] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, 1996.

[2] S. M. LaValle and J. J. Kuffner, "Randomized Kinodynamic Planning," *IJRR*, vol. 20, May 2001.

[3] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[4] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *Proceedings of International Conference on Robotics and Automation*, vol. 3. IEEE, 1997, pp. 2719–2726.

[5] A. M. Ladd and L. E. Kavraki, "Fast tree-based exploration of state space for robots with dynamics," in *Algorithmic Foundations of Robotics VI*. Springer, 2004, pp. 297–312.

[6] I. A. Şucan and L. E. Kavraki, "Kinodynamic motion planning by interior-exterior cell exploration," in *Algorithmic Foundation of Robotics VIII*. Springer, 2009, pp. 449–464.

[7] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *IJRR*, vol. 30, no. 7, June 2011.

[8] L. Janson, A. Schmerling, 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.

[9] S. Karaman and E. Frazzoli, "Sampling-based optimal motion planning for non-holonomic dynamical systems," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 5041–5047.

[10] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "Lqr-rrt*: Optimal sampling-based motion planning with automatically derived extension heuristics," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 2537–2542.

[11] C. Xie, J. 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 *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4187–4194.

[12] Y. Li, Z. Littlefield, and K. E. Bekris, "Asymptotically optimal sampling-based kinodynamic planning," *The International Journal of Robotics Research*, vol. 35, no. 5, pp. 528–564, 2016.

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

[14] 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 Robotics and Automation Letters*, vol. 4, no. 2, pp. x–xvi, 2018.

[15] 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," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6344–6350.

[16] C. Grinstead and J. Snell, *Introduction to Probability*. Providence, RI: American Mathmatical Society, 2012.

[17] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning," *IJRR*, vol. 35, no. 5, 2016.

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

[19] Z. Littlefield and K. E. Bekris, "Efficient and asymptotically optimal kinodynamic motion planning via dominance-informed regions," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.