Algorithmica (1995) 14: 480–530

Algorithmica
© 1995 Springer-Verlag New York Inc.

Provably Good Approximation Algorithms for Optimal Kinodynamic Planning for Cartesian Robots and Open-Chain Manipulators[1]

B. R. Donald[2] and P. G. Xavier[3]

Abstract. In *optimal kinodynamic planning*, given a robot system, we must find a minimal-time trajectory that goes from a start state to a goal state while avoiding obstacles by a speed-dependent safety margin and respecting dynamics bounds. With Canny and Reif [1], we approached this problem from an ε-approximation standpoint and introduced a provably good approximation algorithm for optimal kinodynamic planning for a robot obeying particle dynamics. If a solution exists, this algorithm returns a trajectory ε-close to optimal in time polynomial in both $(1/\varepsilon)$ and the geometric complexity.

We extend [1] and [2] to d-link three-dimensional robots with full rigid-body dynamics amidst obstacles. Specifically, we describe polynomial-time approximation algorithms for Cartesian robots obeying L_2 dynamic bounds and for open-kinematic-chain manipulators with revolute and prismatic joints. The latter class includes many industrial manipulators. The correctness and complexity of these algorithms rely on new trajectory tracking lemmas for robots with coupled dynamics bounds.

Key Words. Robot motion planning, Optimal control, Polynomial-time ε-approximation algorithm, Time-optimal trajectory, Full dynamics, Shortest path, Kinodynamics, Polyhedral obstacles.

1. Introduction. It has been the hope of theoretical computer science that by making simplifying assumptions in robotics problems, precise combinatorial algorithms could be obtained, and that these results could later be generalized to cover "real" robots with control uncertainty, full dynamics, and imperfect models. For example, it is common to assume point robots (often planar), perfect position-control, and trivial dynamics. Optimization issues—such as finding the "fastest" path—are often ignored. Generalizing the early results and relaxing these assumptions are essential if algorithmic analysis is to have an impact in the theory and practice of robotics, particularly in motion planning.

As described in our companion paper [3], the *kinodynamic planning problem*[4] [1], [2] is to synthesize a robot motion subject to simultaneous kinematic constraints and dynamics constraints, including the dynamics law that governs

[1] This paper describes research done at the Computer Science Robotics Laboratory at Cornell University. Support for our robotics research there is provided in part by the National Science Foundation under Grant Nos. IRI-8802390 and IRI-9000532, by a Presidential Young Investigator award, and in part by the Mathematical Sciences Institute, Intel Corporation, and AT&T Bell Laboratories.
[2] Department of Computer Science, Cornell University, Ithaca, NY 14853-7501, USA.
[3] Sandia National Laboratories, Albuquerque, NM 87185-0951, USA.
[4] Reference [2] is the journal revision of [1].

Received February 16, 1993; revised December 10, 1993. Communicated by J.-D. Boissonnat.

the motion. A *kinodynamic solution* is a trajectory specification, such as a start state and a mapping from time to generalized forces or accelerations. Because of errors in control, models, and sensing, a trajectory plan can only be considered safe if it avoids obstacles by some safety margin. This margin is incorporated into the kinodynamic constraints. An important problem is to synthesize *time-optimal kinodynamic solutions*, which require minimal time with respect to the kinodynamic constraints. This problem is NP-hard in three dimensions [3], [4], and thus it is reasonable to develop approximation algorithms.

A *provably good polynomial-time approximation algorithm* for kinodynamic planning is guaranteed to find a solution that is provably close to optimal when a solution exists. Suppose an optimal trajectory that avoids obstacles by the speed-dependent margin δ_v (as in our companion paper [3]; also see (6)) takes time T_{opt}. Then, given an encoding of the problem and an approximation parameter ε, the algorithm will find a trajectory that:

(a) Takes at most time $(1 + \varepsilon)T_{\text{opt}}$.
(b) Approximates the start and goal states to within a factor of ε.
(c) Avoids obstacles by the margin $(1 - \varepsilon)\delta_v$.

Furthermore, the running time of the algorithm is polynomial in both $1/\varepsilon$ and the geometric complexity of the problem.

Canny *et al.* [1] provided a provably good polynomial-time approximation algorithm for two- and three-dimensional optimal kinodynamic planning in the restricted case of particle dynamics. Donald and Xavier [3], [5] modify this algorithm to improve the accuracy and complexity.

Here, we extend our approach to robots with coupled dynamics bounds, in particular to a class of robot systems that includes d-link open-chain manipulators with revolute and prismatic joints as well as Cartesian robots obeying L_2-norm dynamics bounds. Our algorithms find two types of trajectories: ones that obey piecewise-constant extremal controls, and ones that obey piecewise-constant near-extremal accelerations. The algorithms have asymptotic complexity bounds and branching factors (in the search) lower than those for the approximation algorithm of [6]–[8], which also generalizes the results of [1] to d-link open-chain manipulators.

2. Kinodynamic Motion Planning for Robots with Coupled Dynamics Bounds

2.1. A More General Kinodynamic Planning Problem.

We now reformulate the optimal and ε-optimal kinodynamic planning problems to accommodate a wider class of robots than covered by our companion paper [3].

We again denote robot configuration space by C, and its phase space, the robot state space, by TC. A robot motion taking time T_f can be specified by a twice-differentiable map $\mathbf{p}$: $[0, T_f] \to C$, called the *path* of the motion. The *trajectory* of a robot motion is the map Γ: $[0, T_f] \to TC$ given by $\Gamma(t) = (\mathbf{p}(t), \dot{\mathbf{p}}(t))$. Thus, a motion is determined by an initial state $(\mathbf{p}_0, \mathbf{v}_0)$ and an *acceleration function*

$\mathbf{a} = \ddot{\mathbf{p}}$. We retain the convention of denoting the position and velocity components of a subscripted trajectory Γ_r by $\mathbf{p}_r$ and $\dot{\mathbf{p}}_r$, respectively.

As before, a robot with d degrees of freedom must move from a start state $\mathbf{S} = (\mathbf{s}, \dot{\mathbf{s}})$ to a goal state $\mathbf{G} = (\mathbf{g}, \dot{\mathbf{g}})$ while avoiding a set of obstacles and configuration (e.g., joint) limits; these are the *kinematic constraints*. However, we now allow the dynamics constraints to be more general than in our previous work.

The robot motion is governed by a *dynamics law*, which relates applied generalized forces $\mathbf{f}$ to states, accelerations, and forces $\mathbf{G}(\mathbf{p})$ induced by gravity. For open kinematic chains [9], [10],

$$\mathbf{f}(t) = \mathbf{M}(\mathbf{p}(t))\mathbf{a}(t) + [\dot{\mathbf{p}}^T(t)\mathbf{C}(\mathbf{p}(t))\dot{\mathbf{p}}(t)] + \mathbf{G}(\mathbf{p})(t)). \tag{1}$$

$\mathbf{M}(\mathbf{p}(t))$, the robot intertia tensor, is orthogonal, symmetric, and positive-definite. $\mathbf{C}(\mathbf{p}(t))$ is a tensor of rank three, and $[\dot{\mathbf{p}}^T(t)\mathbf{C}(\mathbf{p}(t))\dot{\mathbf{p}}(t)]$ denotes the column vector in which

$$[\dot{\mathbf{p}}^T(t)\mathbf{C}(\mathbf{p}(t))\dot{\mathbf{p}}(t)]_i = \dot{\mathbf{p}}^T(t)\mathbf{C}^i(\mathbf{p}(t))\dot{\mathbf{p}}(t), \quad \text{where } \mathbf{C}^i(\mathbf{p}(t))_{j,k} = \frac{\partial M_{jk}(\mathbf{p})}{\partial p_k} - \frac{1}{2}\frac{\partial M_{jk}(\mathbf{p})}{\partial p_i}. \tag{2}$$

(See [9] for a deviation.) It is important to note that each component of $\mathbf{M}(\mathbf{p})$ and $\mathbf{G}(\mathbf{p})$ is a sum of products of components of $\mathbf{p}$ and their sines and cosines (e.g., p_i, $\cos(p_j)$, etc.).

We call a robot whose inertia tensor is constant and whose dynamics law simplifies to

$$\mathbf{f}(t) = \mathbf{M}\mathbf{a}(t) \tag{3}$$

a *Cartesian robot.*

A robot motion $\mathbf{p}$ obeys dynamic bounds $(\bar{\mathbf{v}}, \bar{\mathbf{f}})$ if for all times t the joint velocities $\dot{\mathbf{p}}$ and the applied generalized forces $\mathbf{f}$ obey the following at each joint i:

$$|\dot{p}_i(t)| \le \bar{v}_i \quad \text{and} \quad |f_i(t)| \le \bar{f}_i. \tag{4}$$

These bounds imply global acceleration bounds $\bar{\mathbf{a}}$ (via (10)), and we define

$$A_{\max} = \max_i \bar{a}_i.$$

In practice, acceleration bounds are sometimes used instead of force bounds for Cartesian robots; because of (3), an algorithm that works for one formulation suffices for the other. In a further simplification, all the dynamics bounds are sometimes given in an L_p-norm, e.g., for $p = 2$,

$$\|\mathbf{v}(t)\|_2 \le \bar{v} \quad \text{and} \quad \|\mathbf{a}(t)\|_2 \le \bar{a}. \tag{5}$$

We consider the L_2-norm in this paper, whereas in [1] and [3] we used the L_∞-norm.

The dynamics laws and dynamics bounds that apply to a robot are its *dynamics constraints.* Note that in this paper we use "Cartesian" only to describe the dynamics laws obeyed by the robot, whereas in [3] and [4] the "Cartesian Kinodynamic Planning Problem" refers to a point robot obeying L_∞-norm dynamics bounds.

The problem parameters must include an encoding $\mathscr{M}$ of the robot's dynamics equation. Since the general form of the equation is given by (1), this involves supplying an encoding of the matrices $\mathbf{M}(\mathbf{p})$ and $\mathbf{C}(\mathbf{p})$ and the vector $\mathbf{G}(\mathbf{p})$ in terms of configuration $\mathbf{p} \in C$. In addition, there must be an encoding $\mathscr{O}$ of the workspace obstacles. An instance of the *general kinodynamic planning problem,* then, is a tuple $\mathscr{K} = (\mathscr{O}, \mathbf{S}, \mathbf{G}, \bar{\mathbf{f}}, \bar{\mathbf{v}}, \mathscr{M})$. An *exact solution* is a trajectory Γ such that $\Gamma(0) = \mathbf{S}$, $\Gamma(T_f) = \mathbf{G}$ at some time T_f, and Γ obeys the kinematic and dynamics constraints. Thus, the corresponding map $\mathbf{p}$ must avoid all obstacles and respect (1) and (4). The *time* for solution Γ is simply T_f. The *time-optimal kinodynamic planning problem* is to find a minimal-time solution, which is represented as a suitable encoding of the start state and the acceleration function $\mathbf{a}$.

We assume that the robot and obstacles are polyhedral, and that they have been mapped to N configuration-space (C-space) constraints, as in [11]–[14], that give rise to the C-space obstacles. *Free space* is the complement of the C-space obstacles in C. For a polyhedral robot of geometric complexity m and a set of polyhedral obstacles with geometric complexity n, the number of configuration space constraints $N = O(m(m + n))$, since an arm must avoid self collisions [11]. Finally, we assume that all linear (i.e., nonrevolute) degrees of freedom are bounded from above by a length l.

As before, we define a trajectory to be *δ_v-safe* if and only if, for all times t in $[0, T_f]$, all real-space obstacles are avoided by a distance of at least

$$\delta_v(c_0, c_1)(\dot{\mathbf{p}}(t)) = c_0 + c_1 \|\dot{\mathbf{p}}(t))\|, \tag{6}$$

where $c_0 > 0$ and $c_1 \geq 0$ are problem parameters. In other words, we can think of each obstacle as having a shell whose thickness grows with the trajectory speed $\|\dot{\mathbf{p}}(t)\|$; contact with these grown obstacles is avoided under a δ_v-safe trajectory. This is different from requiring that there be a ball about $\mathbf{p}(t)$ in free space at all times t. Note that any rational p-norms can be used for distance and speed in (6), since the expanded obstacles would still be semi-algebraic.

For any scalars $c_0 > 0$ and $c_1 \geq 0$ we define an *optimal (δ_v-safe) kinodynamic solution* to be a δ_v-safe solution whose time is minimal. An instance of the optimal (δ_v-safe) kinodynamic planning problem is a tuple $(\mathscr{O}, \mathbf{S}, \mathbf{G}, \bar{\mathbf{f}}, \bar{\mathbf{v}}, l, \mathscr{M}, c_0, c_1)$. $\bar{\mathbf{f}}, \bar{\mathbf{v}}, c_0$, and c_1 are the *kinodynamic bounds.* Together, the kinodynamic bounds and $\mathscr{M}$ are the *kinodynamic specifications* of the robot.

As before [1], [3], the quality of a solution is measured in terms of an approximation parameter ε. A solution Γ_q: $[0, T_q] \to TC$ is *ε-optimal* if:

(a) $\Gamma_q(0)$ and $\Gamma_q(T_q)$ are within $O(\varepsilon)$ tolerances of **S** and **G**, respectively.
(b) $T_q \leq (1 + \varepsilon)T_{\mathrm{opt}}$, where T_{opt} is the time of an optimal solution.
(c) Γ_q is $\delta'_v(c_0, c_1)$-safe, where

$$\delta'_v(c_0, c_1)(\dot{\mathbf{p}}(t)) = (1 - \varepsilon)\delta_v(c_0, c_1)(\dot{\mathbf{p}}(t)). \tag{7}$$

2.2. Statement of Results: Robots with Coupled Dynamics Bounds. A d-degree-of-freedom (d-DOF) robot system obeys *coupled dynamics bounds* if:

(a) Its dynamics equations cannot be separated into the dynamics equations of d independent one-dimensional systems.
(b) There is no fixed coordinate transformation such that the velocity and acceleration bounds in each axial direction are independent of the bounds in all other axial directions.

We describe provably good approximation algorithms for the optimal (safe) kinodynamic planning problems for two classes of robots with coupled dynamics bounds: Cartesian robots with L_2 (-norm) dynamics bounds and open-chain manipulators with revolute and prismatic joints. Given a problem instance and an approximation parameter ε, these algorithms will find an ε-optimal solution if a $\delta_v(c_0, c_1)$-safe solution exists.

These algorithms run in time polynomial in the geometric complexity N of the configuration space obstacles and in the resolution $(1/\varepsilon)$. In the following two theorems c_D and c_E are constants dependent only on the kinodynamic specifications of the robot and are polynomial in d, and thus c_D^d and c_E^d are constant for any particular robot. Our algorithms are ε-approximation schemes that are *fully polynomial* in the combinatorial complexity of the geometry and *pseudopolynomial* in the kinodynamic specifications. As in [1], a key intuition is to reduce the problem to searching a graph whose edges correspond to "primitive" trajectory segments.

Formally stated, we show the following:

THEOREM 2.1. *Let $\bar{a}$ and $\bar{v}$ be velocity and acceleration bounds, respectively. Let $(\mathcal{O}, \mathbf{S}, \mathbf{G}, \bar{a}, \bar{v}, l, c_0, c_1)$ be an optimal kinodynamic planning problem for a d-degree of freedom Cartesian robot obeying L_2 dynamics bounds. Let $0 < \varepsilon < 1$.*

Suppose there is a $\delta_v(c_0, c_1)$-safe trajectory from **S** *to* **G** *taking time T_{opt}. Then the algorithms we describe each find a $(1 - \varepsilon)\delta_v(c_0, c_1)$-safe trajectory taking time at most $T_{\mathrm{opt}}(1 + \varepsilon)$ and going from some $\mathbf{S}^* = (\mathbf{s}^*, \dot{\mathbf{s}}^*)$ to some $\mathbf{G}^* = (\mathbf{g}, \dot{\mathbf{g}}^*)$ such that* **S*** *and* **G*** are *ε-close to* **S** *and* **G**, *respectively.*

The asymptotic running time of the algorithms is $O(c_D^d p(N, \varepsilon, d)(1/\varepsilon)^{6d-1})$, where N is the geometric complexity of the C-space obstacles, c_D is a constant dependent on the algorithm and the kinodynamic specifications and is polynomial in d, and $p(N, \varepsilon, d)$ is the time-complexity of checking the $(1 - \varepsilon)\delta_v(c_0, c_1)$-safety of one of the "primitive" trajectory segments the algorithms consider.

THEOREM 2.2. *Let $(\mathcal{O}, \mathbf{S}, \mathbf{G}, \bar{\mathbf{f}}, \bar{\mathbf{v}}, \mathcal{M}, l, c_0, c_1)$ be an optimal kinodynamic planning problem for an open-chain manipulator with revolute and/or prismatic joints. Let $0 < \varepsilon < 1$.*

Suppose there is a $\delta_v(c_0, c_1)$-safe trajectory from $\mathbf{S}$ *to* $\mathbf{G}$ *taking time T_{opt}. Then the algorithms we describe each find a $(1 - \varepsilon)\delta_v(c_0, c_1)$-safe trajectory taking time at most $T_{\text{opt}}(1 + \varepsilon)$ and going from some $\mathbf{S}^* = (\mathbf{s}^*, \dot{\mathbf{s}}^*)$ to some $\mathbf{G}^* = (\mathbf{g}^*, \dot{\mathbf{g}}^*)$ such that $\mathbf{S}^*$ and $\mathbf{G}^*$ are ε-close to* $\mathbf{S}$ *and* $\mathbf{G}$*, respectively.*

The asymptotic running time of the algorithms is $O(c_E^d p(N, \varepsilon, d)(1/\varepsilon)^{6d-1})$ where N is the geometric complexity of the C-space obstacles, c_E is a constant dependent on the algorithm and the kinodynamic specifications and is polynomial in d, and $p(N, \varepsilon, d)$ is the time-complexity of checking the $(1 - \varepsilon)\delta_v(c_0, c_1)$-safety of one of the "primitive" trajectory segments the algorithms consider.

We have omitted the complexity factors containing kinodynamic specifications because they are constant for a given robot and because the terms can be very complicated. We express their overall contribution to the complexity bounds as a factor of c_D^d (or c_E^d). Our claim that the complexities are pseudopolynomial in the kinodynamic specifications and $O((1/\varepsilon)^{6d-1})$ is substantiated in the sections that follow. In Section 3 we sketch our approach, and in Section 4 we prove two lemmas key to our results. Detail is provided in Section 5, with further detail given in Appendix A.

In Section 5.4 we give an approach for safety checking and argue that $p(N, \varepsilon, d)$ is roughly $O(N(d + \log N))$. As we discuss there, for non-Cartesian open-chain manipulators amidst polyhedral real-space obstacles, exact collision detection for quadratic paths requires the solution of mixed trigonometric equations that cannot be transformed into algebraic equations using the usual substitution methods, such as [12]. Furthermore, for these manipulators, trajectory segments corresponding to constant extremal controls are solutions to systems of ordinary trigonometric differential equations, and the trajectories found by the corresponding algorithm are extremal to the accuracy of the solution method; see Section 3.1.2. In both of these cases, certain parameters would be adjusted, and numerical techniques would be used to do safety checking approximately.

For a choice of C-space coordinates in which the path is algebraic in time and in which the obstacles are represented as semialgebraic sets *in configuration space*, $p(N, \varepsilon, d)$ would be $O(N \log N)$ for the True-Extremal Algorithm (applied to Cartesian robots) and for the Near-Extremal Algorithm (in general). Finally, we note that the True-Extremal Algorithm is more theoretical than the Near-Extremal Algorithm, and that for a given robot it yields a larger c_D or c_E.

2.3. Previous and Related Work. Canny *et al.* [1] and Donald *et al.* [2] address the problem of kinodynamic planning for a point robot with L_∞-normal velocity bound $\bar{v}$ and acceleration bound $\bar{a}$ in an environment with polyhedral obstacles. Canny *et al.* [1] provide the first formulation of kinodynamic planning as an approximation problem and obtain the first provable polynomial-time approximation algorithm for kinodynamic planning in more than one dimension. A review of the body of previous and related work on planning robotic motions subject to dynamics constraints and obstacle avoidance can be found in [3] and in [1] itself. The key innovation that differentiates [1] from previous graph-search-based algorithms for motion planning is that the parameters of the reachability graph guarantee that if the kinodynamic planning problem instance is solvable, then the shortest path in the graph from the root vertex to some vertex approximating the problem goal will yield an ε-optimal kinodynamic trajectory.

By improving the algorithm and its complexity anlaysis, we obtained the better accuracy and complexity results in our companion paper [3], which were initially reported in [5] and [15]. Through a coordinate transformation, all these algorithms can be applied to kinodynamic planning for a Cartesian robot with L_∞-norm dynamics bounds and polyhedral C-space obstacles.

Jacobs *et al.* [6]–[8][5] built on the methods in [1] to obtain the first polynomial time approximation algorithm for optimal kinodynamic planning for open-chain manipulators—the first for robot systems with state-dependent dynamics. Their work introduces several techniques to the kinodynamic planning literature, including:

(a) Discretizing acceleration-space according to the problem parameters.
(b) Reducing state-dependent dynamics to being locally constant.

Our approach (see the early description in [15]–[17]) toward state-dependent dynamics is similar, but we obtain better complexity results.[6] In concurrent work, Reif and Tate [18] used a parameter-dependent acceleration-space discretization implicitly to obtain a polynomial-time approximation algorithm for robots with decoupled dynamics, L_2 dynamics bounds, and polyhedral C-space obstacles. For this L_2 problem we also obtain an algorithm with a lower complexity bound.

3. Robots with Coupled Dynamics Bounds. In this section we provide an overview of the algorithms and describe the key concepts used in obtaining our results.

3.1. Algorithms Overview. We describe the basic ideas behind two general algorithms for finding near-optimal kinodynamic trajectories for Cartesian robots with L_2-norm dynamics bounds and for open-chain manipulators. The first algorithm searches a reachability graph corresponding to piecewise-constant, extremal forces and torques, and we refer to it as "the True-Extremal Algorithm." The second uses piecewise-constant, near-extremal accelerations, and we call it "the Near-Extremal Algorithm."

[5] We refer to this body of work as [6]–[8].
[6] The result in [6]–[8] preceded the result in [15]–[17].

3.1.1. The Basic Idea. Our algorithms transform the problem of finding an approximately optimal trajectory to that of finding the shortest path in a directed graph. In a general sense, we follow the technique from [1]:

(a) The vertices of the graph are states in TC.
(b) The graph is a *reachability graph*, whose *root vertex* approximates the start state and whose edges are $(1-\varepsilon)\delta_v(c_0, c_1)$-safe trajectory segments.
(c) These $(1-\varepsilon)\delta_v(c_0, c_1)$-safe trajectory segments are generated by a finite set of control primitives and each takes duration τ, the *timestep* chosen by the algorithm.
(d) The graph is explored from the root vertex, and the search terminates when either a vertex approximating the goal state is found or when no new vertices are generated (found).

Since we use the notion of a "slowed-down" trajectory often, we now formalize two definitions, as in [1].

DEFINITION 3.1. Let Γ_r: $[0, T_f] \rightarrow TC$, and suppose that $\varepsilon \geq 0$. Then we define the *ε-time rescaled* trajectory Γ_r': $[0, (1+\varepsilon)T_f] \rightarrow TC$ as follows:

$$\Gamma_r'(t) = \left(\mathbf{p}_r\left(\frac{t}{1+\varepsilon}\right), \left(\frac{1}{1+\varepsilon}\right)\dot{\mathbf{p}}_r\left(\frac{t}{1+\varepsilon}\right)\right). \tag{8}$$

Thus, if Γ_{opt} is an optimal kinodynamic trajectory, then Γ_{opt}' will be ε-optimal when $\varepsilon < 1$.

DEFINITION 3.2. Let $\mathbf{X} = (\mathbf{x}, \dot{\mathbf{x}}) \in TC$, and suppose that $\varepsilon \geq 0$. Then the *ε-time-rescaled state* $\mathbf{X}' = (\mathbf{x}', \dot{\mathbf{x}}')$ is defined by

$$\begin{aligned} \mathbf{x}' &= \mathbf{x}, \\ \dot{\mathbf{x}} &= \left(\frac{1}{1+\varepsilon}\right)\dot{\mathbf{x}}. \end{aligned} \tag{9}$$

These two definitions are important because our proofs extensively utilize dynamical properties associated with states in rescaled trajectories.

As in the algorithm of [1], in our new algorithms the root vertex of the reachability graph will approximate the ε-time-rescaled start state in the kinodynamic planning problem instance. The *basic* idea is that if a solution Γ_r to the problem instance exists, then some graph trajectory will track Γ_r' closely enough to be ε-optimal.

The algorithms presented here differ from the algorithms in [1]–[3] in three important ways.

First, the control primitives our new algorithms use to generate the reachability graph correspond to extremal or near-extremal accelerations that differ from their "neighbors" by $O(\varepsilon)$. This is necessary to guarantee, as required by our proof technique, that a trajectory constrained to the control primitives can out-accelerate an ε-time-rescaled optimal trajectory. Let us consider a point robot obeying L_2-norm acceleration bound $\bar{a}$. For any fixed finite set $\mathscr{A}$ of acceleration vectors obeying this bound, an $\varepsilon > 0$ exists such that, in some direction $\mathbf{n}$, there will be no convex combination $\mathbf{a}$ of vectors in $\mathscr{A}$ such that $\mathbf{a} \cdot \mathbf{n} \geq \bar{a}/(1 + \varepsilon)^2$. Thus, a point obeying acceleration bound $\bar{a}/(1 + \varepsilon)^2$ would be able to out-accelerate a point limited to accelerations from $\mathscr{A}$.

Second, the control primitives used by the True-Extremal Algorithm generally do not yield a finite reachability graph. While our previous algorithms would potentially explore the entire reachability graph, the True-Extremal Algorithm instead bounds its search by only exploring nodes that are not too close to previously found nodes. Hence, a state density condition limits its search size.

Third, the True-Extremal Algorithm uses constant applied force/torques, which can yield nonconstant accelerations, as its control primitives. Although the main purpose of our algorithms is only to guarantee ε-approximately optimal trajectories, it would be desirable for an algorithm possibly to find a truly optimal trajectory with respect to δ'_v-safety, even for a "nearby" problem (as in numerical analysis). In robotics and control theory [19] there is a family of results (e.g., [20]—[22]) on the feasibility of planning and approximating optimal trajectories using only piecewise-extremal controls; these results are often called "bang-bang" theorems. Among other things, they imply that such an algorithm would have to include piecewise-extremal trajectories in its search. Under the less restrictive dynamics model we use here, this generally excludes piecewise-constant accelerations.

The use of a grid of $O(\varepsilon)$-spaced accelerations as control primitives is shared by [18], and [6]–[8]. Our use of near-extremal and extremal accelerations as control primitives distinguishes our algorithm from the latter work and, with our proof techniques, contributes to our lower complexity bound. Our use of state density (or spatial hashing) to prune the reachability graph is a first in provably good kinodynamic planning.

3.1.2. Control Primitives Used by the True-Extremal Algorithm. Although the correctness of the True-Extremal Algorithm is harder to prove than the correctness of the Near-Extremal Algorithm, the former is conceptually simpler.

We now describe the control primitives the True-Extremal Algorithm uses to generate the edges in its reachability graph. Let $\mathscr{U}$ be the set of generalized (control) forces that are extremal with respect to given bounds $\bar{\mathbf{f}}$. (We can define similar sets for L_p bounds $\bar{f}$ and $\bar{a}$.) Given a scalar *discretization parameter* $\bar{\mu} > 0$, we say that $\mathscr{U}_{\bar{\mu}} \subset \mathscr{U}$ is *a* $\bar{\mu}$*-discretization of the control extremals* if for every $\mathbf{u}_a \in \mathscr{U}$ there is some $\mathbf{u}_b \in \mathscr{U}_{\bar{\mu}}$ such that $\frac{1}{2}\|\mathbf{u}_a - \mathbf{u}_b\|_\infty \leq \bar{\mu}$. As long as $\bar{\mu}$ is small enough, our algorithm may choose any such $\mathscr{U}_{\bar{\mu}}$, so we henceforth refer to $\mathscr{U}_{\bar{\mu}}$ as *the* $\bar{\mu}$-discretization of control extremals.

The application of a member of $\mathscr{U}_{\bar{\mu}}$ for duration τ is called a *true* $(\bar{\mu}, \tau)$*-bang*. (The "true" distinguishes it from uses of "bang" which refer to controls that are constant, but only *nearly* extremal.) Recall that the dynamics equation maps states and generalized forces to accelerations. From (1), the acceleration is given by

$$\mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f}) = \mathbf{M}^{-1}(\mathbf{p})(\mathbf{f} - [\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}] - \mathbf{G}(\mathbf{p})). \tag{10}$$

Thus, a true $(\bar{\mu}, \tau)$-bang and a state **X** together determine a trajectory segment of duration τ starting at **X**, i.e., the solution to (10) with initial state **X** and **f** constant over the duration of the $(\bar{\mu}, \tau)$-bang. If this trajectory ends at state **Y**, then we say that there is *a true* $(\bar{\mu}, \tau)$*-bang from* **X** *to* **Y**. Thus, we sometimes use "true $(\bar{\mu}, \tau)$-bang" to refer to the trajectory segment generated by a bang when the meaning is clear.

Furthermore, the *acceleration functions* associated with true $(\bar{\mu}, \tau)$-bangs beginning at a given state **X** are important to our discussion. We call these functions the *true* $(\bar{\mu}, \tau)$*-bangs at* **X**, and the set of them is denoted by $\mathscr{H}(\mathbf{X}, \tau)$. A *true* $(\bar{\mu}, \tau)$*-bang trajectory* is a concatenation of true $(\bar{\mu}, \tau)$-bangs. If Γ_r is a true $(\bar{\mu}, \tau)$-bang trajectory, then $\mathbf{a}_r(t - n\tau) \in \mathscr{H}(\Gamma_r(n\tau), \tau)$ during each open time interval $(n\tau, (n + 1)\tau)$.

For an open-chain manipulator or an L_2 Cartesian robot, the number of states reachable from a root vertex $\mathbf{S}^*$ via a sequence of m $(1 - \varepsilon)\delta_v$-safe $(\bar{\mu}, \tau)$-bangs can, in general, be exponential in the path length m. Worse yet, the total number states reachable via an infinite number of such bangs can be unbounded, even in a bounded state space. Therefore, we apply (b_x, b_v)*-bucket pruning*, a spatial hashing technique, in the search. (See Figure 1.) In (b_x, b_v)-bucket pruning TC is divided into voxels of diameter b_x in spatial dimensions and b in velocity dimensions. When the graph search finds a vertex in a voxel in which another vertex has previously been found, the edges from the new vertex are simply not explored, and we say they are *pruned* from the search. (Compare this with the search in [13], Figure 31 on p. 315.) Since the state space is bounded and the reachability

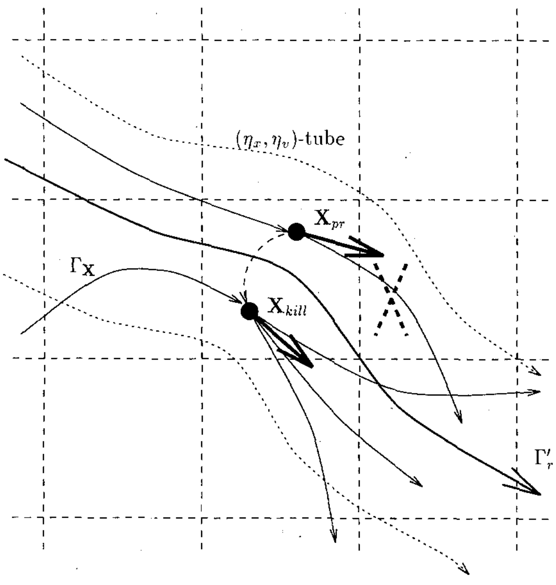

Fig. 1. (b_x, b_v)-bucket pruning search pruning. During the search, an edge (trajectory segment) to vertex (state) $\mathbf{X}_{pr}$ is found, but a previously found vertex $\mathbf{X}_{\text{kill}}$ is in the same bucket. Consequently, edges from $\mathbf{X}_{pr}$ are pruned from the search. (Compare with Figure 31 on p. 315 of [13]. In Section 4.3 we show that the True-Extremal Algorithm still has the desired completeness with this search pruning.

graph has an $O((\bar{\mu})^{-(d-1)})$ branching factor, the size of the subgraph searched is $O((\bar{\mu})^{-(d-1)}(b_x b_v)^{-d})$.

We note that for non-Cartesian open-chain manipulators, (10) generally has no closed-form solution. Thus, we consider the True-Extremal Algorithm is purely theoretical: we either choose a model of computation in which (10) is solved by an oracle, or we settle for approximate, numerically computed $(\bar{\mu}, \tau)$-bangs. Under the latter choice, the planned trajectory segments cannot really be extremal without risking their being unexecutable. (See Section 5.1.)

3.1.3. Control Primitives Used by the Near-Extremal Algorithm: Cartesian Robots. As an alternative to (b_x, b_v)-bucket pruning and the problem of computing true $(\bar{\mu}, \tau)$-bangs, the Near-Extremal Algorithm builds a reachability graph using trajectory segments corresponding to constant near-extremal accelerations that each last one timestep. These accelerations and a root vertex are chosen so that the size of the reachability graph is automatically bounded. Under our current proof techniques, the algorithm has a much lower constant term in its complexity than the True-Extremal Algorithm. We now describe the accelerations and outline the algorithm.

A positive vector $\boldsymbol{\mu} \in \mathbb{R}^d$ determines a grid that discretizes the acceleration space $\mathbb{R}^d$. Precisely, let μ_i be the grid-spacing in dimension i, and let the grid be aligned with the coordinate axes so that the origin $\mathbf{0}$ is a gridpoint; i.e., it lies at one of the interstices of the grid. We call the set of gridpoints the *$\boldsymbol{\mu}$-grid.* For now, suppose that timestep τ has been chosen and that the set of accelerations has been discretized. Given a state $\mathbf{X} \in TC$, let $\mathscr{A}_\mathbf{X}$ denote the set of instantaneous accelerations possible at $\mathbf{X}$ under the dynamics constraints. (Recall that we assume that the set of control forces $\mathscr{U}$ is state-invariant, but that these forces map to accelerations via (10).)

Let $\hat{\mathscr{A}}_\mathbf{X} \subset \mathscr{A}_\mathbf{X}$ denote the set of constant accelerations (i.e., vectors) possible for trajectories of duration τ beginning at state $\mathbf{X}$:

$$\hat{\mathscr{A}}_\mathbf{X} = \left\{ \mathbf{a} \in \mathscr{A}_\mathbf{X} \,\middle|\, \mathbf{a} \in \bigcap_{t \in [0,\tau]} \mathscr{A}_{\Gamma(\mathbf{X},\mathbf{a},t)} \right\},$$

where $\Gamma(\mathbf{X}, \mathbf{a}, t)$ is the trajectory that results in applying constant acceleration $\mathbf{a}$ at state $\mathbf{X}$. Then a "constant acceleration" analogue of the set of true $(\bar{\mu}, \tau)$-bang accelerations would be the set of $\boldsymbol{\mu}$-gridpoints that lie within a $\boldsymbol{\mu}$-grid-spacing of the boundary $\partial\hat{\mathscr{A}}_\mathbf{X}$ of $\hat{\mathscr{A}}_\mathbf{X}$.

For now, we call these accelerations and the corresponding trajectory segments *$(\boldsymbol{\mu}, \tau)$-bangs.* Suppose that, for a state $\mathbf{S}^*$ and in each dimension i, the acceleration discretization μ_i divides the state velocity $\dot{s}_i^*$. Then all states reachable from $\mathbf{S}^*$ via a sequence of $(\boldsymbol{\mu}, \tau)$-bangs lie at the interstices of an underlying grid covering TC, with a grid spacing $\mu_i \tau^2/2$ in position dimension i and $\mu_i \tau$ in velocity dimension i. Thus, a $\boldsymbol{\mu}$-grid, a timestep τ, and a choice of origin in C *induce* an underlying regular *TC-grid.* If $\mathbf{S}^*$ is the root vertex of a reachability graph, then the number of its vertices is bounded above by the number of TC-gridpoints. (See [1], [2] and ensuing work such as [3], [6], [17], and [18].)

For Cartesian robots, the sets $\hat{\mathscr{A}}_{\mathbf{X}}$ are easily computed, and hence the Near-Extremal Algorithm for Cartesian robots uses $(\boldsymbol{\mu}, \tau)$-bangs as described above to generate a reachability graphs. $(\boldsymbol{\mu}, \tau)$-bangs are defined formally in the section below, which also covers the non-Cartesian case.

3.1.4. Control Primitives Used by the Near-Extremal Algorithm: More General Dynamics. Unfortunately, it appears that $\hat{\mathscr{A}}_{\mathbf{X}}$ is very difficult to compute for the general dynamical case (10). For non-Cartesian robots, we therefore define sets of near-extremal constant accelerations using conservative subset approximations of the sets $\hat{\mathscr{A}}_{\mathbf{X}}$.

Let $\mathscr{D}_{\mathbf{X}} \subset TC$ denote the set of states that can be reached from $\mathbf{X}$ within time τ without violating the dynamics constraints. Then one conservative approximation to $\hat{\mathscr{A}}_{\mathbf{X}}$ is

$$(11) \qquad \tilde{\mathscr{A}}_{\mathbf{X}} = \bigcap_{\mathbf{Y} \in \mathscr{D}_{\mathbf{X}}} \mathscr{A}_{\mathbf{Y}} \subset \hat{\mathscr{A}}_{\mathbf{X}}.$$

This set still might be difficult to compute when the robot dynamics are nonlinear, since the intersection is taken over a set of states.

The dynamics and dynamics bounds determine global bounds on the magnitudes Δx and Δv of the possible position and velocity changes during a duration of length τ. Using these bounds, we can compute an analogous "acceleration space" bound E_A, which has the following property: if $\mathbf{a}^{\mathbf{Y}} \in \mathscr{A}_{\mathbf{Y}}$ for some $\mathbf{Y}$ within $(\Delta x, \Delta v)$ of $\mathbf{X}$, then there is some $\mathbf{a}^{\mathbf{X}} \in \mathscr{A}_{\mathbf{X}}$ such that $\|\mathbf{a}^{\mathbf{Y}} - \mathbf{a}^{\mathbf{X}}\| \leq E_A$. (See Section 5.3.1 and Appendix A.)

We now define:

$$(12) \qquad \hat{\mathscr{A}}^{*}_{\mathbf{X}} = \bigcap_{\Delta\mathbf{a} \in B_{E_A}(\mathbf{0})} \{\mathbf{a} \in \mathscr{A}_{\mathbf{X}} \mid \exists \mathbf{a}' \in \mathscr{A}_{\mathbf{X}}, \mathbf{a} = \mathbf{a}' + \Delta\mathbf{a}\},$$

where $B_\delta(\mathrm{y})$ denotes the δ-ball about y. Since $\mathscr{A}_{\mathbf{X}}$ is a parallelepiped or a closed L_p-norm d-ball, this set has a simple geometry. Thus,

$$\hat{\mathscr{A}}^{*}_{\mathbf{X}} \subset \tilde{\mathscr{A}}_{\mathbf{X}} \subset \hat{\mathscr{A}}_{\mathbf{X}} \subset \mathscr{A}_{\mathbf{X}},$$

with equality among the first three terms in the case of a Cartesian robot, except at the velocity bounds. (See Figure 2.)

We call $\hat{\mathscr{A}}^{*}_{\mathbf{X}}$ the set of *conservatively τ-feasible constant accelerations at* $\mathbf{X}$. Our algorithms treat $\partial\hat{\mathscr{A}}^{*}_{\mathbf{X}}$, the boundary of $\hat{\mathscr{A}}^{*}_{\mathbf{X}}$, as the set of "most-extremal" constant accelerations feasible for a trajectory of duration τ starting at state $\mathbf{X}$.

We now define the *$(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations at* $\mathbf{X}$ to be

$$(13) \qquad \mathscr{H}(\mathbf{X}, \tau) = \{\mathbf{a} \in \boldsymbol{\mu}\text{-grid} \mid \exists \mathbf{a}' \in \partial\hat{\mathscr{A}}^{*}_{\mathbf{X}}, \forall i, |a_i - a_i'| \leq \mu_i\}.$$

This notation and this term also refer to the corresponding set of constant acceleration functions.

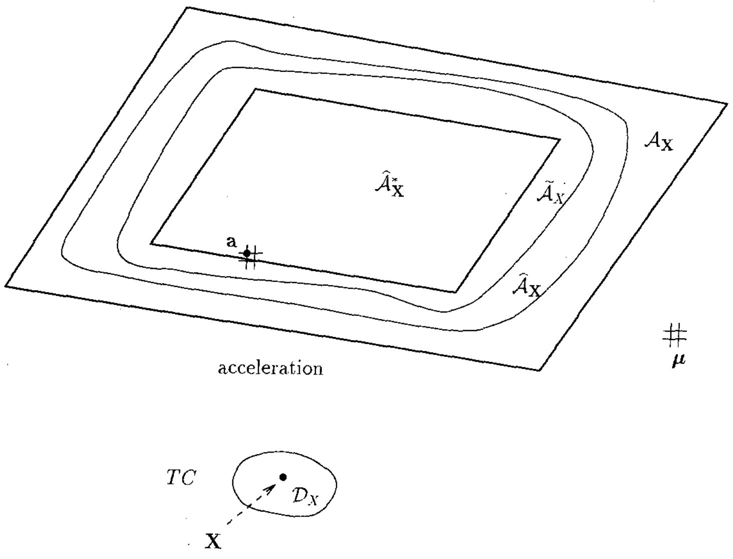

Fig. 2. Sets of accelerations. $\mathscr{A}_X$ is the set of possible instantaneous accelerations at state **X**. $\hat{\mathscr{A}}_X$ is the set of accelerations that can be maintained for duration τ by trajectories starting at **X**. $\tilde{\mathscr{A}}_X = \bigcap \mathscr{A}_Y$, where the intersection is taken over all $Y \in \mathscr{D}_X$, is a conservative estimate of this set. $\hat{\mathscr{A}}^*_X$, used by the Near-Extremal Algorithm to obtain the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations $\mathscr{H}(\mathbf{X}, \tau)$, is "geometrically computable." **a** is a member of $\mathscr{H}(\mathbf{X}, \tau)$. This figure is only a Venn diagram.

The application of a $(\boldsymbol{\mu}, \tau)$-extremal acceleration for duration τ is called a $(\boldsymbol{\mu}, \tau)$-*bang*. Terminology concerning $(\bar{a}, \tau)$-bangs and true $(\bar{\mu}, \tau)$-bangs will be analogously extended to $(\boldsymbol{\mu}, \tau)$-bangs. For example, if a $(\boldsymbol{\mu}, \tau)$-bang results in the transition from state **X** to state **Y**, we say there is a $(\boldsymbol{\mu}, \tau)$-bang from **X** to **Y**.

3.2. Key Concepts. To show the correctness of our algorithms, we follow the general technique from [1]. We formalize the notion of an acceleration advantage and generalize it to sets of acceleration functions, and we use and prove two tracking lemmas that are more general than the ones in [1], [3], [6]–[8], and [18] and yield tighter complexity bounds for our algorithms than those obtained in [6]–[8], [18].

3.2.1. Overview. As in [1], tracking is a key concept.

DEFINITION 3.3. Let Γ_a and Γ_b be trajectories, and suppose that, for all $t \in [0, T]$,

$$\|\mathbf{p}_a(t) - \mathbf{p}_b(t)\|_\infty \leq \eta_x \quad \text{and} \quad \|\dot{\mathbf{p}}_a(t) - \dot{\mathbf{p}}_b(t)\|_\infty \leq \eta_v. \tag{14}$$

Then we say that Γ_a *tracks* Γ_b *to tolerance* (η_x, η_v). Furthermore, for all $t \in [0, T]$, $\Gamma_a(t)$ *approximates* $\Gamma_b(t)$ *to tolerance* (η_x, η_v).

The *Safe Tracking Lemma* [1] relates c_0, c_1, and ε to a family of tracking tolerances. Specifically, given c_0, c_1 and ε, the lemma provides a set of tracking tolerances (η_x, η_v) such that if a trajectory Γ_a is $\delta_v(c_0, c_1)$-safe and Γ_b tracks Γ_a to tolerance (η_x, η_v), then Γ_b will be $(1 - \varepsilon)\delta_v(c_0, c_1)$-safe. This lemma is stated in Section 3.2.2, and it ensures that the algorithms can choose sufficiently small ε-safe tracking tolerances η_x and η_v that are $\Theta(\varepsilon)$.

Recall that, for a trajectory Γ, Γ' denotes Γ ε-time-rescaled (8). To prove the correctness of each algorithm we show that if a given problem has an optimal solution Γ_{opt} taking time T_{opt}, then the algorithm's choice of root vertex state $\mathbf{S}^*$, timestep τ, and discretization $\boldsymbol{\mu}$ (or $\bar{\mu}$) guarantee that it will, at worst, find a graph trajectory Γ_q that tracks Γ'_{opt} to tolerance (η_x, η_v). Γ_q will therefore be $(1 - \varepsilon)\delta_v(c_0, c_1)$-safe and approximate $\mathbf{G}'$ to within $O(\varepsilon)$ at time $(1 + \varepsilon)T_{\mathrm{opt}}$. Since $\varepsilon < 1$, $\mathbf{S}'$ and $\mathbf{G}'$ are within $O(\varepsilon)$ of $\mathbf{S}$ and $\mathbf{G}$. It follows that $\Gamma_q(0)$ and $\Gamma_q((1 + \varepsilon)T_{\mathrm{opt}})$ will also be within $O(\varepsilon)$ of $\mathbf{S}$ and $\mathbf{G}$.

Following an analysis similar to that found in [1], we observe that if a Cartesian robot trajectory Γ_r respects L_2 dynamics bounds $\bar{a}$ and $\bar{v}$, then the ε-time-rescaled trajectory Γ'_r will respect bounds $\bar{a}'$ and $\bar{v}'$ that are smaller than $\bar{a}$ and $\bar{v}$ by a $\Theta(\varepsilon)$ factor. This is also true for open-chain manipulators as long as the force bound $\bar{\mathbf{f}}$ is great enough to overcome gravity in all configurations when the robot is stationary.

At this point, we introduce techniques that significantly extend those in [1]. In Sections 3.2.3 and 3.2.4 we generalize from feasible instantaneous accelerations to *state-dependent sets of acceleration functions* and formalize an appropriate notion of *acceleration advantage* over a time interval.

The *Coupled Tracking Lemma* (Lemma 3.2) presented in Section 3.2.5 relates such an acceleration advantage κ_l, a maximum acceleration, and a desired tracking tolerance to a timestep τ. Using the lemma we can choose $\Theta(\varepsilon)$ parameters μ_i, $\bar{\mu}$, and τ that guarantee the existence of graph trajectories that will track Γ'_{opt} to the $\Theta(\varepsilon)$ tolerances obtained via the Safe Tracking Lemma. For the Near-Extremal Algorithm, this enables us to find conditions that guarantee that *some* graph trajectory will track an ε-time-rescaled optimal trajectory Γ'_{opt} closely enough. In Section 3.2.6 we present the *Robust Coupled Tracking Lemma* (Lemma 3.3), which gives conditions ensuring that (b_x, b_v)-pruning during a breadth-first search of the reachability graph will not eliminate trajectories the True-Extremal Algorithm needs to find. Thus, Lemmas 3.2 and 3.3 are the key to showing algorithm correctness and complexity, modulo safety checking.

Proofs for Lemmas 3.2 and 3.3 are presented in Section 4. Details of how the algorithms choose the reachability graph parameters are given in Sections 5.2 and 5.3. An approximate but sufficiently precise computational method of safety checking is described in Section 5.4.

3.2.2. Safe Tracking. Assuming that a optimal trajectory Γ_{opt} exits, we must determine a tracking tolerance that guarantees that any trajectory tracking Γ'_{opt} to that tolerances will be $(1 - \varepsilon)\delta_v(c_0, c_1)$-safe.

LEMMA 3.1 (The Safe Tracking Lemma [1]). *Suppose that δ_v is specified by c_0 and c_1 and that Γ_r is a δ_v-safe trajectory relative to the C-space obstacles. Let*

$0 < \varepsilon < 1$, *and let* $\delta'_v = (1 - \varepsilon)\delta_v$. *Then a tolerance* (η_x, η_v) *exists such that, for any trajectory* Γ_q, *the following hold*:

1. *If* Γ_q *tracks* Γ_r *to tolerance* (η_x, η_v), *then* Γ_q *is* δ'_v*-safe.*
2. *Furthermore, for any positive* β, *the following choices suffice*:

$$\eta_v \leq \frac{c_0\varepsilon}{c_1(1-\varepsilon)+\beta}, \qquad \eta_x \leq \beta\eta_v. \tag{15}$$

The lemma is independent of norm, as long as a particular norm is consistently used, and distances and velocities are in *TC*. A proof is found in [1].

If we wish the safety margin to correspond to the distance between the physical robot and the obstacles, then η_x can be divided by the maximum modulus of the forward kinematic map. Similarly, if we wish the velocity to correspond to the velocity of a point on the physical robot, then η_v can be divided by $\max_{\mathbf{p},\dot{\mathbf{p}}}\|\mathbf{J}(\mathbf{p})\dot{\mathbf{p}}\|$, where $\mathbf{J}(\mathbf{p})$ is the manipulator Jacobian. We denote these divisors by $P_{\max}$ and $J_{\max}$, respectively. Then we (conservatively) require

$$\eta_v \leq \frac{c_0\varepsilon}{J_{\max}(c_1(1-\varepsilon)+\beta)}, \qquad \eta_x \leq \frac{\beta c_0\varepsilon}{P_{\max}(c_1(1-\varepsilon)+\beta)}. \tag{16}$$

By choosing $\beta = 1$ we see that η_x and η_v can both be $O(\varepsilon)$. The algorithm chooses β to maximize the timestep and thus minimize the overall complexity. Given c_0, c_1 and ε, we call a tolerance (η_x, η_v) that obeys (16) a *physical* ε*-safe-tracking tolerance.*

Finally, because the safety-checking procedure is a numerical method, when computing the parameters that determine the reachability graph, the η_x given by (15) or (16) must be reduced fractionally to account for the numerical error.

3.2.3. Time-Rescaling and Instantaneous Acceleration Advantages. If trajectory Γ_r obeys L_∞ dynamics bounds $\bar{a}$ and $\bar{v}$, then the ε-time-rescaled trajectory Γ'_r obeys L_∞ dynamics bounds $\bar{a}/(1+\varepsilon)^2$ and $\bar{v}/(1+\varepsilon)$. For any d-vector σ of 1's and -1's there is a d-vector $\hat{\sigma}$ of 1's and -1's such that if $\|\mathbf{a}(t)\|_\infty \leq \bar{a}/(1+\varepsilon)^2$, then, for each dimension i,

$$\sigma_i(\hat{\sigma}_i\bar{a} - a_i(t)) \geq \frac{\bar{a}(2\varepsilon+\varepsilon^2)}{(1+\varepsilon)^2}.$$

Therefore, a trajectory obeying $(\bar{a}, \tau)$-bang control [1], [3] can always out-accelerate the ε-time-rescaled trajectory Γ'_r by an $\bar{a}(2\varepsilon+\varepsilon^2)/(1+\varepsilon)^2$ margin *in any direction over the entirety of a timestep*, as long as this would not violate the velocity bound. We can thus immediately reduce the analysis for L_∞ Cartesian robots to the one-dimensional case, where we exploit this *acceleration advantage* to relate the timestep τ to how closely some $(\bar{a}, \tau)$-bang trajectory is guaranteed to track Γ'_r (in the absence of obstacles). We use an acceleration advantage together with the dynamics bounds, Lemma 3.1, and Lemma 3.2 or 3.3 to choose a timestep τ.

As the first step toward defining a type of acceleration advantage appropriate for sets of *acceleration functions*, as used in the Coupled Tracking Lemma, we describe a generalization of "instantaneous acceleration advantage" to both L_2 Cartesian robots and open-chain manipulators.

Recall that the robot motion obeys the following equation:

$$\text{(1)} \qquad \mathbf{f}(t) = \mathbf{M}(\mathbf{p}(t))\mathbf{a}(t) + [\dot{\mathbf{p}}^T(t)\mathbf{C}(\mathbf{p}(t))\dot{\mathbf{p}}(t)] + \mathbf{G}(\mathbf{p}(t)).$$

Suppose a trajectory Γ_r obeys the bounds $\bar{\mathbf{f}}$. Recall that Γ'_r is Γ_r ε-time-rescaled (8). We find [10] that at time $(1+\varepsilon)t$:

$$\text{(17)} \qquad \mathbf{f}'_r((1+\varepsilon)t) = \frac{\mathbf{f}_r(t)}{(1+\varepsilon)^2} + \frac{2\varepsilon+\varepsilon^2}{(1+\varepsilon)^2}\mathbf{G}(\mathbf{p}_r(t)).$$

This immediately means that Γ'_r obeys some tighter bounds $\bar{\mathbf{f}}'$ if the robot is Cartesian or if there is no gravity. For an open-chain manipulator, Γ'_r obeys tigher bounds $\bar{\mathbf{f}}'$ as long as the bounds $\bar{\mathbf{f}}$ are large enough for the robot to be held stationary in the presence of gravity in all configurations. In particular, suppose that the forces necessary to balance gravity obey the bound $(1-\kappa_f)\bar{\mathbf{f}}$, so that the scalar κ_f describes the advantage over gravity. Then Γ'_r obeys the bounds

$$\text{(18)} \qquad \bar{\mathbf{f}}' = \frac{1+(1-\kappa_f)(2\varepsilon+\varepsilon^2)}{(1+\varepsilon)^2}\bar{\mathbf{f}},$$

in addition to

$$\text{(19)} \qquad \bar{\mathbf{v}}' = \left(\frac{1}{1+\varepsilon}\right)\bar{\mathbf{v}}.$$

Then, for every degree of freedom i,

$$\text{(20)} \qquad \bar{f}_i - \bar{f}'_i \geq \frac{\kappa_f(2\varepsilon+\varepsilon^2)}{(1+\varepsilon)^2}\bar{f}_i > \frac{\kappa_f\varepsilon}{2}\bar{f}_i.$$

Recall that $\mathscr{A}_\mathbf{X}$ denotes the sets of feasible instantaneous accelerations at the state $\mathbf{X}$ under the force bound $\bar{\mathbf{f}}$. Let $\mathscr{A}'_\mathbf{X}$ denote the corresponding set under the force bounds $\bar{\mathbf{f}}'$, and let $f_{\min} = \min_i \bar{f}_i$. Recall that, for any $\mathbf{y} \in \mathbb{R}^d$,

$$\text{(21)} \qquad \|\mathbf{y}\|_\infty \leq \|\mathbf{y}\|_2 \leq \sqrt{d}\|\mathbf{y}\|_\infty.$$

Now, because $\mathbf{M}(\mathbf{x})$ in (1) is orthogonal, (20) implies that, for all $\mathbf{a} \in \mathscr{A}_{\mathbf{X}}$ and $\mathbf{a}' \in \mathscr{A}'_{\mathbf{X}}$,

$$\text{(22)} \qquad \|\mathbf{a} - \mathbf{a}'\|_2 \geq \frac{\kappa_f f_{\min}\varepsilon}{2\lambda_{\max}},$$

where $\lambda_{\max}$ is the maximum eigenvalue of $\mathbf{M}(\mathbf{x})$ taken over all of C. Equation (22) is also true for a Cartesian manipulator obeying L_2-norm dynamics bounds; in this case it follows from (3) that $\kappa_f = 1$ and $f_{\min} = \bar{f}$. The L_2 acceleration bound case is trivial, since it is equivalent to setting $\mathbf{M}$ to the identity matrix.

In all three cases $\mathscr{A}'_{\mathbf{X}}$ lies entirely inside $\mathscr{A}_{\mathbf{X}}$ by some $\Theta(\varepsilon)$ margin. In particular, we can always find a scalar $\kappa_l > 0$ such that, for any $\mathbf{a} \in \mathscr{A}'_{\mathbf{X}}$ and any d-vector σ of 1's and -1's, there is an $\hat{\mathbf{a}} \in \partial\mathscr{A}_{\mathbf{X}}$ such that, for all degrees of freedom i,

$$\sigma_i(\hat{a}_i - a_i) \geq \kappa_l.$$

Hence, we say that $\mathscr{A}_{\mathbf{X}}$ has a κ_l *instantaneous acceleration advantage* over $\mathscr{A}'_{\mathbf{X}}$. (See Figure 3.)

3.2.4. Acceleration Functions and Uniform Advantages. We now introduce a form of acceleration advantage useful for comparing sets of acceleration functions. We say that a *trajectory* Γ_r *respects a set* $\mathscr{Y}$ *of acceleration functions in interval*

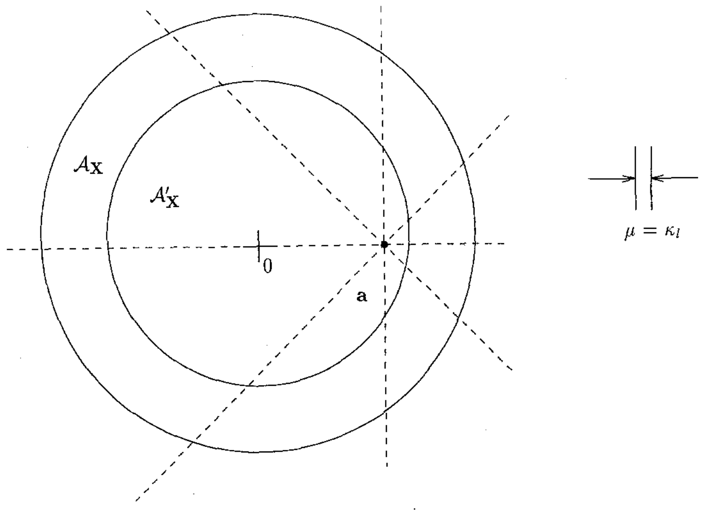

Fig. 3. Allowable accelerations for the L_2 Cartesian robot at any state $\mathbf{X}$. $\mathscr{A}_{\mathbf{X}}$ obeys $\bar{a}$, $\mathscr{A}'_{\mathbf{X}}$ obeys $\bar{a}/(1+\varepsilon)^2$. For any $\mathbf{a} \in \mathscr{A}'_{\mathbf{X}}$, in each direction given by a vector of 1's and -1's, there is some $\hat{\mathbf{a}} \in \mathscr{A}_{\mathbf{X}}$ that has a κ_l advantage over $\mathbf{a}$; the intersections of the dotted lines and the outer circle ($\partial\mathscr{A}_{\mathbf{X}}$) represent "witnesses." Thus, $\mathscr{A}_{\mathbf{X}}$ has a uniform κ_l advantage over $\mathscr{A}'_{\mathbf{X}}$ for any duration τ (as long as the velocity bound is not violated). If the actual uniform advantage is large enough and all μ_i are small enough, then the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations $\mathscr{H}(\mathbf{X}, \tau)$ (see (13)) has a uniform κ_l advantage over $\mathscr{A}'_{\mathbf{X}}$.

$[t_1, t_2]$ if there is an acceleration function $\mathbf{a} \in \mathscr{Y}$ such that $\mathbf{a}_r(t) = \mathbf{a}(t - t_1)$ for all $t \in [t_1, t_2]$. Suppose that $\hat{\mathscr{Y}}$ is also a set of acceleration functions, and suppose that, for every $\mathbf{a} \in \mathscr{Y}$ and each d-vector σ of 1's and -1's, there is some $\hat{\mathbf{a}} \in \hat{\mathscr{Y}}$ such that, for all $t \in [0, \tau]$,

$$\sigma_i\left(\hat{a}_i(t) - \frac{\int_0^\tau a_i(s)\,ds}{\tau}\right) \geq \kappa_l. \tag{23}$$

Then we say that $\hat{\mathscr{Y}}$ *has a uniform* κ_l advantage over $\mathscr{Y}$ *for duration* τ.

Because the set of feasible accelerations for a Cartesian robot is essentially state-invariant, we can find an acceleration advantage κ_l and discretizations $\mu_i, \bar{\mu}$ that are $\Theta(\varepsilon)$, and such that both the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations $\mathscr{H}(\mathbf{X}, \tau)$ and the set of true $(\bar{\mu}, \tau)$-bangs $\bar{\mathscr{H}}(\mathbf{X}, \tau)$ at every state $\mathbf{X}$ have a uniform κ_l advantage over the acceleration of $\Gamma'_{\mathrm{opt}}(t)$ at every time t for any duration τ. We note that this discussion about the set of feasible accelerations does not consider velocity bounds, and that the inclusion of velocity bounds changes the set of feasible accelerations for states on the velocity boundary. This technical detail is minor because of the effect ε-time rescaling has on a trajectory's velocity bound.

We now generalize the notion of a set of instantaneous accelerations $\mathscr{A}_{\mathbf{X}}$ to a mapping

$$\mathscr{A}\colon TC \mapsto \{\mathscr{Y} \mid \mathscr{Y} \text{ is a set of acceleration functions}\}$$

from TC to sets of acceleration functions. For a given robot $\mathscr{A}(\mathbf{X})$ is defined to be the set of acceleration functions obeying the dynamics bounds $\bar{\mathbf{f}}$ and $\bar{\mathbf{v}}$ for a robot motion (in the absence of obstacles) beginning at state $\mathbf{X}$. Thus, $\mathbf{a}_r \in \mathscr{A}(\mathbf{X})$ if and only if there is a trajectory Γ_r that begins at state $\mathbf{X}$ and obeys the dynamics bounds. Furthermore, an instantaneous acceleration (vector) $\mathbf{a}^{\mathbf{X}} \in \mathscr{A}_{\mathbf{X}}$ if and only if there is some acceleration function $\mathbf{a} \in \mathscr{A}(\mathbf{X})$ such that $\mathbf{a}^{\mathbf{X}} = \mathbf{a}(0)$. Strictly speaking, $\mathscr{H}(\mathbf{X}, \tau) \subset \mathscr{A}(\mathbf{X})$, and for each $\mathbf{a} \in \mathscr{H}(\mathbf{X}, \tau)$ there is a constant $\mathbf{a}^{\mathbf{X}} \in \mathscr{A}_{\mathbf{X}}$ such that $\mathbf{a}(t) = \mathbf{a}^{\mathbf{X}}$ for all times $t \in [0, \tau]$.

We say that Γ_r *respects* $\mathscr{A}$ if, for all intervals $[t_1, t_2]$, $\Gamma_r(t_1) = \mathbf{X}$ implies that Γ_r respects $\mathscr{A}(\mathbf{X})$ in $[t_1, t_2]$. $\mathscr{A}'$ denotes the map from TC to all acceleration functions feasible under the bounds $\bar{\mathbf{f}}'$ and $\bar{\mathbf{v}}'$ given in (18) and (19).

Since the set of feasible instantaneous accelerations for an open-chain manipulator is state-dependent, to choose the discretization parameter $\bar{\mu}$ (or $\boldsymbol{\mu}$) and a timestep τ, we consider the relationship between feasible acceleration functions (such as the members of $\mathscr{A}(\mathbf{X})$) and feasible sets of instantaneous accelerations (such as $\mathscr{A}_{\mathbf{X}}$). Suppose Γ_r respects $\mathscr{A}'$. Then the average acceleration of Γ_r over the interval $[n\tau, (n+1)\tau]$ is

$$\frac{\mathbf{v}_r((n+1)\tau) - \mathbf{v}_r(n\tau)}{\tau} \in \bigcup_{t \in [n\tau, (n+1)\tau]} \mathscr{A}'_{\Gamma_r(t)}. \tag{24}$$

Suppose that Γ_q is a $(\boldsymbol{\mu}, \tau)$-bang trajectory. Then, for all $t \in [n\tau, (n+1)\tau]$, $\mathbf{a}_q(t) = \mathbf{a}^{(n)}$ for some constant $\mathbf{a}^{(n)}$ such that, for all i, u_i divides $a_i^{(n)}$ and

$$(25) \qquad \mathbf{a}^{(n)} \in \bigcap_{t \in [n\tau, (n+1)\tau]} \mathscr{A}_{\Gamma_q(t)}.$$

If $\Gamma_q(n\tau)$ and $\Gamma_r(n\tau)$ are close enough and τ is small enough, then

$$(26) \qquad \bigcup_{t \in [n\tau, (n+1)\tau]} \mathscr{A}'_{\Gamma_r(t)} \subset \bigcap_{t \in [n\tau, (n+1)\tau]} \mathscr{A}_{\Gamma_q(t)} \subset \hat{\mathscr{A}}^*_{\Gamma_q(n\tau)},$$

and, furthermore, the boundaries of $\bigcup_{t \in [n\tau, (n+1)\tau]} \mathscr{A}'_{\Gamma_r(t)}$ and $\hat{\mathscr{A}}^*_{\Gamma_q(n\tau)}$ are separated by some distance. (Recall (12).)

If this separation distance is great enough compared with κ_l and $\|\boldsymbol{\mu}\|_\infty$ and $\bar{\mu}$, then both the $(\boldsymbol{\mu}, \tau)$-extremal shell at $\Gamma_q(n\tau)$ and the set of true $(\bar{\mu}, \tau)$-bangs at $\Gamma_q(n\tau)$ will have uniform κ_l advantages over $\mathscr{A}'(\Gamma_r(n\tau))$. We use this relationship between sets of instantaneous accelerations as a criterion for comparing maps from *TC* to sets of acceleration functions.

Since we can bound the minimal distance between members of $\partial\mathscr{A}_{\mathbf{X}}$ and members of $\partial\mathscr{A}'_{\mathbf{X}}$, we now consider how points on $\partial\mathscr{A}_{\mathbf{X}}$ and $\partial\mathscr{A}'_{\mathbf{X}}$ "move" in response to small, continuously realizable changes, or *perturbations*, to state **X**. It is straightforward (but tedious) to show that physically possible state perturbations $(\Delta\mathbf{x}, \Delta\mathbf{v})$ cause points on $\partial\mathscr{A}_{\mathbf{X}}$ and $\partial\mathscr{A}'_{\mathbf{X}}$ to move by distances that are $O(\Delta\mathbf{x})$ and $O(\Delta\mathbf{v})$. Therefore, because we assume global dynamics bounds, points on $\partial\mathscr{A}_{\Gamma_q(t)}$ and points on $\partial\mathscr{A}_{\Gamma'_r(t)}$ move by $O(\Delta t)$ over a duration of length Δt. Specifically, we can globally bound the effect of perturbations $(\Delta\mathbf{x}, \Delta\mathbf{v}, \Delta t)$ on $\partial\mathscr{A}_{\mathbf{X}}$ by $h_{q0}(\|\Delta t\|) + h_{q1}(\|\Delta\mathbf{x}\|, \|\Delta\mathbf{v}\|)$ and the effect of a perturbation Δt on $\partial\mathscr{A}'_X$ by $h_r(\|\Delta t\|)$, where h_{q0}, h_{q1}, and h_r are linear. (See Figure 4 and Appendix A.)

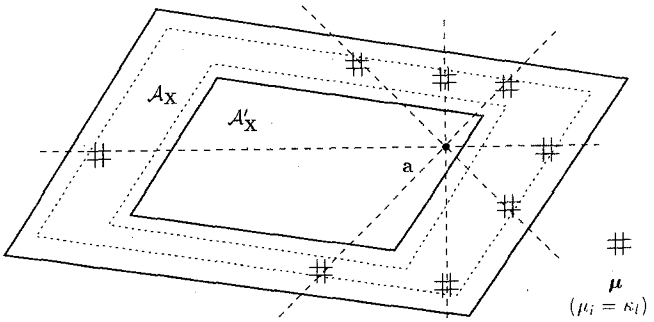

Fig. 4. Sets in acceleration space. $\mathscr{A}_{\mathbf{X}}$ obeys bound $\bar{\mathbf{f}}$; $\mathscr{A}'_{\mathbf{X}}$ obeys $\bar{\mathbf{f}}'$. The outer dotted parallelogram represents the boundary of the set obtained by considering all "perturbations" of $\mathscr{A}_{\mathbf{X}}$ by up to h_{q1} $(\eta_x, \eta_v) + h_{q0}(\tau)$ and taking the intersection. The intersection of this set with the $\boldsymbol{\mu}$-grid has a κ_l advantage over the set (represented by the inner dotted parallelogram) obtained by considering all "perturbations" of $\mathscr{A}'_{\mathbf{X}}$ by up to $h_r(\tau)$ and taking the union. The eight μ-grid patches crossed by dashed lines show this κ_l advantage with respect to an acceleration (vector) **a**.

This means that we can find $\Theta(\varepsilon)$ uniform acceleration advantage κ_l, discretization $\boldsymbol{\mu}$, fundamental timestep τ_0, and maximal tracking tolerances η_{x0} and η_{v0} such that if $\mathbf{X}$ is within (η_{x0}, η_{v0}) of $\mathbf{Y}$, then the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations at $\mathbf{X}$, $\mathscr{H}(\mathbf{X}, \tau)$, has a uniform κ_l advantage over $\mathscr{A}(\mathbf{Y})$ for duration $\min(\tau, \tau_0)$. Hence, the fundamental timestep is defined to be the maximum that guarantees *some* $(\boldsymbol{\mu}, \tau)$-bang trajectory will be able to stay within *some* constant tracking tolerance of an ε-time-rescaled trajectory. Following similar steps, we can choose $\bar{\mu}$ to guarantee that the set of true $(\bar{\mu}, \tau)$-bangs at $\mathbf{X}$, $\bar{\mathscr{H}}(\mathbf{X}, \tau)$, also has a uniform κ_l advantage. (Details in Section 5.3.)

In other words, we can choose a set of *consistent parameters* $(\mu, \tau_0, \eta_{x0}, \eta_{v0}, \kappa_l)$: a maximal discretization parameter μ for either acceleration or control, a maximal timestep τ_0, a maximal tracking tolerance (η_{x0}, η_{v0}), and a uniform acceleration advantage κ_l. For any timestep $\tau \le \tau_0$ and any trajectory Γ_r that respects $\mathscr{A}'$, the following will be true: if state $\mathbf{X}$ is within (η_{x0}, η_{v0}) of $\Gamma_r(n\tau)$, and both $\bar{\mu}$, $\|\boldsymbol{\mu}\|_\infty \le \mu$, then both the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations $\mathscr{H}(\mathbf{X}, \tau)$ and the set $\bar{\mathscr{H}}(\mathbf{X}, \tau)$ of true $(\bar{\mu}, \tau)$-bangs have uniform κ_l advantages over $\mathscr{A}'(\Gamma_r(t - n\tau))$ for duration τ.

For this general case, we describe how to choose sufficiently small μ, κ_l, τ_0, η_{x0}, and η_{v0} that are $\Theta(\varepsilon)$ in Section 5.3. The Cartesian case, in which τ_0, η_{x0}, and η_{v0} do not arise, is handled in Section 5.2.

3.2.5. The Coupled Tracking Lemma. The Coupled Tracking Lemma is the key to choosing, for a given tracking tolerance, a timestep τ and other parameters that guarantee the following: for any ε-time-rescaled trajectory Γ'_r beginning at $\mathbf{S}'$, *some* $(\boldsymbol{\mu}, \tau)$-bang trajectory and *some* true $(\bar{\mu}, \tau)$-bang trajectory will track Γ'_r to that tolerance. Note that the set of acceleration functions associated with a given state can be arbitrary, as long as that set has the necessary uniform acceleration advantage. This allows us to apply the lemma to sets of true $(\bar{\mu}, \tau)$-bangs without knowing their forms as path functions; i.e., we do not need a series or closed-form representation. The lemma can be applied more generally to state-dependent sets of bounded acceleration functions whose exact form is unknown but whose time-derivatives are also bounded.

LEMMA 3.2 (The Coupled Tracking Lemma). *Let $A_{\max}$ be the global L_∞ acceleration bound. Consider functions $\mathscr{P}$, $\mathscr{Q}$: $TC \to \{\mathscr{Y} : \mathscr{Y}$ is a set of acceleration functions$\}$. Let $\mathscr{P}$ have the property that if, for some state $\mathbf{X}$, $\mathbf{a} \in \mathscr{P}(\mathbf{X})$, then, for all times t, $\|\mathbf{a}(t)\|_\infty \le A_{\max} - \kappa_l$. Let $\mathscr{Q}$ have the property that if acceleration function $\mathbf{a} \in \mathscr{Q}(\mathbf{X})$, then, for all times t, $\|\mathbf{a}(t)\|_\infty \le A_{\max}$. Let $\Gamma_r(t)$ respect $\mathscr{P}$.*

Suppose that a maximal tracking tolerance (η_{x0}, η_{v0}) and a fundamental timestep τ_0 exist such that for all $\tau \le \tau_0$ the following is true: if state $\mathbf{X}$ is within (η_{x0}, η_{v0}) of state $\mathbf{Y}$, then the set of acceleration functions $\mathscr{Q}(\mathbf{X})$ has a uniform κ_l advantage[7] *over $\mathscr{P}(\mathbf{Y})$ for duration τ.*

[7] Recall (23).

Then, for any L_∞ tracking tolerance (η_x, η_v), a timestep τ and a velocity $\mathbf{v}_0$ exist such that if trajectory Γ_r respects $\mathscr{P}$ and $\|\dot{\mathbf{p}}_r(0) - \mathbf{v}_0\|_\infty \leq 2A_{\max}\tau$, then a trajectory Γ_q exists such that:

1. $\mathbf{v}_q(0) = \mathbf{v}_0$.
2. $\mathbf{p}_r(0) = \mathbf{p}_q(0)$.
3. *Γ_q tracks Γ_r to tolerance (η_x, η_y).*
4. *For each n, Γ_q respects $\mathscr{Q}(\Gamma_q(n\tau))$ in $[n\tau, (n+1)\tau]$.*

Moreover, it is sufficient that

$$\begin{aligned} &\tau \leq \tau_0, \\ &\tau \leq \sqrt{\frac{\min(\eta_x, \eta_{x0})\kappa_l}{A_{\max}(8A_{\max} + 6\kappa_l)}}, \qquad (27) \\ &\tau \leq \frac{\min(\eta_v, \eta_{v0})}{4A_{\max}}. \end{aligned}$$

PROOF. Presented in Section 4.

Recall from Section 3.2.2 that there is an ε-safe tracking tolerance (η_x, η_v) where both η_x and η_v are $\Theta(\varepsilon)$. The Near-Extremal algorithm uses the Coupled Tracking Lemma (above) to choose a timestep τ that is $\Theta(\varepsilon)$ and that guarantees that if a solution Γ_{opt} exists, then there will be a $(\boldsymbol{\mu}, \tau)$-graph trajectory that tracks Γ'_{opt} to tolerance (η_x, η_v). The True-Extremal Algorithm can choose τ similarly, but because of the search pruning, this alone will not guarantee it can *find* such a trajectory. For the True-Extremal Algorithm, we must apply a stronger tracking lemma.

3.2.6. Coupled Tracking and (b_x, b_v)-Bucket Pruning. In the True-Extremal Algorithm, the kinodynamic constraints, ε, and the choices of discretization parameter $\bar{\mu}$, timestep τ, and root vertex $\mathbf{S}^*$ determine a reachability graph $\mathscr{G}$ whose vertices are states and whose edges correspond to $(1-\varepsilon)\delta_v(c_0, c_1$-safe $(\bar{\mu}, \tau)$-bangs.

Recall that in (b_x, b_v)-bucket pruning, the state space TC is divided into voxels with diameter b_x in spatial dimensions and b_v in velocity dimensions. A breadth-first search with (b_x, b_v)-bucket pruning proceeds as a normal breadth-first search does, except that when the search finds a vertex in a voxel that contains another vertex found in a previous generation or earlier in the current generation, the edges out of the newly found vertex will be pruned from the search; i.e., no edges out of that vertex will be explored.

Consider any breadth-first search of $\mathscr{G}$ with (b_x, b_v)-bucket pruning. We call removing all edges and vertices in $\mathscr{G}$ that are not explored during this search *a breadth-first (b_x, b_v)-bucket pruning of $\mathscr{G}$*. We say that a graph-trajectory Γ *remains* after that breadth-first (b_x, b_v)-bucket pruning if the path in $\mathscr{G}$ corresponding to it is not affected by the pruning.

Like the Coupled Tracking Lemma, the Robust Coupled Tracking Lemma applies to sets of acceleration functions more general than $(\bar{\mu}, \tau)$-bangs and

$(\boldsymbol{\mu}, \tau)$-bangs. We introduce three new terms used in its statement. First, let $\mathscr{Q}$ be a mapping $\mathscr{Q}$: $TC \to \{\mathscr{Y}: \mathscr{Y}$ is a finite set of acceleration functions$\}$. We say that a trajectory segment from state $\mathbf{X}$ to state $\mathbf{Y}$ *is a $(\mathscr{Q}, \tau)$-trajectory segment* if there is a trajectory Γ_r such that $\Gamma_r(0) = \mathbf{X}$, $\Gamma_r(\tau) = \mathbf{Y}$, and $\mathbf{a}_r \in \mathscr{Q}(\mathbf{X})$. Second, a vertex $\mathbf{S}^*$ and the set of $(\mathscr{Q}, \tau)$-trajectory segments determine, via transitive closure, the *$(\mathscr{Q}, \tau)$-reachability graph $\mathscr{G}$* rooted at $\mathbf{S}^*$. Finally, if $\mathscr{R}$ is a subset of TC, then the *maximal subgraph of $\mathscr{G}$ lying in $\mathscr{R}$* is the maximal subgraph $\mathscr{G}'$ of $\mathscr{G}$ such that all vertices and all edges (as trajectory segments) in $\mathscr{G}'$ lie in $\mathscr{R}$.

LEMMA 3.3 (The Robust Coupled Tracking Lemma). *Consider functions $\mathscr{P}, \mathscr{Q}$: $TC \to \{\mathscr{Y}: \mathscr{Y}$ is a set of acceleration functions$\}$. Let $\mathscr{P}$ have the property if $\mathbf{a} \in \mathscr{P}(\mathbf{X})$ for some state $\mathbf{X}$, then, for all times t, $\|\mathbf{a}(t)\|_\infty \le A_{\max} - \kappa_l$. Let $\mathscr{Q}$ have the properties that $\mathscr{Q}(\mathbf{X})$ is finite for all $\mathbf{X} \in TC$ and that if $\mathbf{a} \in \mathscr{Q}(\mathbf{X})$, then, for all times t, $\|\mathbf{a}(t)\|_\infty \le A_{\max}$.*

Suppose that a maximal tracking tolerance (η_{x0}, η_{v0}) and a fundamental timestep τ_0 exist such that for all $\tau \le \tau_0$ the following is true: if state $\mathbf{X}$ is within (η_{x0}, η_{v0}) of state $\mathbf{Y}$, then the set of acceleration functions $\mathscr{Q}(\mathbf{X})$ has a uniform κ_l advantage over $\mathscr{P}(\mathbf{Y})$ for duration τ.

Let (η_x, η_v) be an L_∞ tracking tolerance, and let timestep τ obey the following inequalities:

$$\begin{aligned} &\tau \le \tau_0, \\ (28) \qquad &\tau \le \sqrt{\frac{4\min(\eta_{x0}, \eta_x)\kappa_l}{24A_{\max}\kappa_l + 2\kappa_l^2 + (8A_{\max} + \kappa_l)^2}}, \\ &\tau \le \frac{2\min(\eta_{v0}, \eta_v)}{8A_{\max} + \kappa_l}. \end{aligned}$$

Let $\mathscr{G}$ be the $(\mathscr{Q}, \tau)$-reachability graph rooted at some state $\mathbf{S}^*$.

Now, suppose that Γ_r respects $\mathscr{P}$, $\Gamma_r(0)$ is within $2A_{\max}\tau$ of $\dot{\mathbf{s}}^*$, and $\mathbf{p}_r(0) = \mathbf{s}^*$. Suppose that $\mathscr{R}$ is a subset of TC containing the (η_x, η_v)-tube induced by Γ_r, and that $\mathscr{G}'$ results from some breadth-first $(\kappa_l\tau^2/4, \kappa_l\tau/2)$-pruning of the maximal subgraph of $\mathscr{G}$ that lies in $\mathscr{R}$. Then there is a Γ_q in $\mathscr{G}'$ such that $T_q \le T_r$, $\Gamma_q(0) = \mathbf{S}^*$, and $\Gamma_q(T_q)$ approximates $\Gamma_r(T_r)$ to tolerance (η_x, η_v).

PROOF. Presented in Section 4.

Note that $\mathscr{R}$ is introduced to model the union of the (η_x, η_v)-tube induced by the arbitrary Γ_r and the $\delta_v'(c_0, c_1)$-safe region of TC. The lemma guarantees that if $\mathscr{R}$ contains the (η_x, η_v)-tube, then some trajectory with the necessary endpoint properties will survive the pruning of the reachability graph in $\mathscr{R}$. The lemma does *not* guarantee that a trajectory that tracks Γ_r to tolerance (η_x, η_v) or that lie close to Γ_r for its entirety survives the pruning.

Having chosen $\bar{\mu}$ as described in Section 3.2.4 (and in detail in Sections 5.2 and 5.3), the True-Extremal Algorithm uses Lemma 3.3 to choose a timestep τ

that is $\Theta(\varepsilon)$, and pruning bucket dimensions b_x and b_v that are $\Theta(\varepsilon^3)$ and $\Theta(\varepsilon^2)$, respectively. The parameters chosen will ensure that if a δ_v-safe solution Γ_{opt} exists, then a $(\bar{\mu}, \tau)$-true-bang graph trajectory that tracks Γ'_{opt} to the ε-safe tracking tolerance (η_x, η_v) also exists, and the algorithm will find such a trajectory if no sufficient shorter or equivalent-length graph trajectory survives the pruning.

3.3. Asymptotic Bounds. In the preceding sections we have described the key concepts we use in obtaining asymptotic bounds for parameters that guarantee our algorithms will find an ε-optimal solution when a solution exists. We now sketch how to arrive at the complexity bounds of these algorithms.

Given a problem and an approximation parameter ε, the Safe Tracking Lemma shows how to find a family of ε-safe-tracking tolerances (η_x, η_v) such that η_x and η_v are $\Theta(\varepsilon)$. Sections 3.2.3 and 3.2.4 sketch why, for Cartesian robots, there will be sufficiently small discretization parameters $\bar{\mu}$ and μ_i and an acceleration advantage κ_l that are $\Theta(\varepsilon)$. For open-chain manipulators we find consistent parameters $\boldsymbol{\mu}$ (or $\bar{\mu}$), τ_0, η_{x0}, η_{v0}, and κ_l that are $\Theta(\varepsilon)$, where κ_l is a uniform acceleration advantage over τ_0.

Using the lemmas from Sections 3.2 and 3.3 we can then show that there are correct choices of τ that are $\Theta(\varepsilon)$. Section 3.3 implies that since κ_l and τ are both $\Theta(\varepsilon)$, we can choose (b_x, b_v)-pruning bucket dimensions that are $\Theta(\varepsilon^3)$ and $\Theta(\varepsilon^2)$, respectively.

It follows that for a given problem, the number of TC-gridpoints that can be considered by the Near-Extremal Algorithm and the number of (b_x, b_v)-pruning buckets that can be visited by the True-Extremal Algorithm is $O((1/\varepsilon)^{5d})$. Since $\bar{\mu}$ and the μ_i are $\Theta(\varepsilon)$, the out-degree of each graph is $O((1/\varepsilon)^{d-1})$.

The complexities of each algorithm, then, are $O(c_D^d p(N, \varepsilon, d)(1/\varepsilon)^{6d-1})$ and $O(c_E^d p(N, \varepsilon, d)(1/\varepsilon)^{6d-1})$, where $p(N, \varepsilon, d)$ is the cost of checking the $(1-\varepsilon)\delta_v(c_0, c_1)$-safety of a $(\bar{\mu}, \tau)$-bang or a $(\boldsymbol{\mu}, \tau)$-bang. c_D and c_E are constants dependent on the particular robot and the algorithm and are polynomial in d. In Section 5.4 we review numerical techniques sufficient for safety checking that will have a cost of $O(N(d + \log N)$ per bang when ε is sufficiently small. Our algorithms therefore have overall asymptotic complexity bounds of $O(c^d N(d + \log N)(1/\varepsilon)^{6d-1})$.

4. Proving the Coupled Tracking Lemmas.

We now present the proofs of the Coupled Tracking Lemma (Lemma 3.2) and the Robust Coupled Tracking Lemma (Lemma 3.3). These two lemmas are fundamental to obtaining our results. We use them to show how the algorithm can choose discretization parameters $\boldsymbol{\mu}$ and $\bar{\mu}$ and a timestep τ, and how we can calculate a uniform acceleration advantage κ_l.

If a system has decoupled dynamics bounds, then a set of coordinate axes exists such that the acceleration or velocity chosen along any one axis never affects what accelerations or velocities are possible along any other axis. This is why tracking the trajectory of a Cartesian robot obeying L_∞ bounds reduce trivially to the one-dimensional case. On the other hand, if a system has coupled dynamics bounds, then, for any set of coordinate axes, choosing a maximal acceleration along one axis can limit the accelerations along another. Furthermore, if the dynamics equations are state dependent, then the set of instantaneous extremal accelerations is state dependent. The definition of a uniform κ_l acceleration advantage over a timestep is thus crucial to our proofs of Lemmas 3.2 and 3.3.

4.1. The Tracking Game. To prove Lemmas 3.2 and 3.3, we begin by considering a game against an adversary in a one-dimensional space. In this game certain rules simulate dynamics. When simultaneous independent games are played, the adversary can force our movements to be governed by accelerations in the $(\boldsymbol{\mu}, \tau)$-extremal shell or by true $(\bar{\mu}, \tau)$-bangs.

4.1.1. Defining the Game. Consider a game in which you are trying to track an adversary in a one-dimensional space. The game is a series of rounds, each of which simulates motions taking duration τ. The game begins with the adversary in state (x_0, v_0) and you in state (x'_0, v'_0). A round simulates an interval of time with length τ. Note that in our discussion of tracking games, the "$'$" symbol does not denote ε-time rescaling.

Let game parameters $A_{\max}$ and κ_l be given such that $A_{\max} > \kappa_l > 0$. When discussing the game, we use x_n, v_n, etc., to denote game variables that correspond to round n.

In each round the following takes place:

Play for Round n.

1. The adversary plays first by choosing an acceleration function $a_n(t)$ for himself such that, for all $t \in [n\tau, (n+1)\tau]$, $|a_n(t)| \le A_{\max} - \kappa_l$. His state (x_{n+1}, v_{n+1}) at the end of the round (beginning of round $n+1$) is computed from this state (x_n, v_n) and this acceleration function by integration over the time interval $[n\tau, (n+1)\tau]$.
2. For your play in round n, you then choose either HIGH or LOW. This choice limits the acceleration function that the adversary can choose *for you*, which determines your state (x'_{n+1}, v'_{n+1}) at the end of the round. Note that you play only HIGH or LOW; the adversary chooses *your* acceleration *as well as his own.* However, his choice of your acceleration is constrained by your play.
3. Let $a_n^\circ = (v_{n+1} - v_n)/\tau$. If you played HIGH, then your adversary can choose any function $a'_n(t)$ such that $a'_n(t) \ge a_n^\circ + \kappa_l$ for all $t \in [n\tau, [n\tau,(n+1)\tau]$. If instead you played LOW, then your adversary's choice must obey the condition $a'_n(t) \le a_n^\circ - \kappa_l$ for all $t \in [n\tau,(n+1)\tau]$. In both cases, $a'_n(t)$ must obey the condition $|a'_n(t)| \le A_{\max}$ for all $t \in [n\tau, (n+1)\tau]$. (See Figure 5.)
4. Your state (x'_{n+1}, v'_{n+1}) at the end of the round is computed by using your state at the end of the previous round and integrating $a'_n(t)$ over the time interval $[n\tau, (n+1)\tau]$.

4.1.2. A Winning Simple Strategy for the Tracking Game. A simple high-level strategy for the Tracking Game is: try to go faster than the adversary if you have fallen behind; try to go slower than the adversary if you are ahead; but never go much faster or much slower than the adversary. The uniform κ_l advantage you

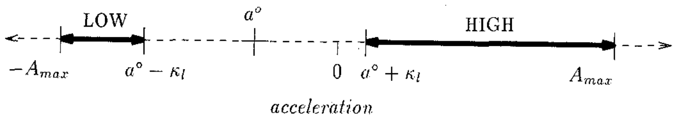

Fig. 5. Details from a game against an adversary. Suppose in the current round the adversary's average acceleration is a_n°. Then if you choose HIGH, the adversary assigns you an acceleration function whose range lies in $[a^\circ + \kappa_l, A_{\max}]$; if you choose LOW, the adversary assigns you an acceleration function whose range lies in $[-A_{\max}, a^\circ - \kappa_l]$. This function controls your motion for the current round, which covers a time interval of length τ.

have over the adversary's average acceleration during a round assures that we can follow such a strategy. (See Figure 6.)

We now give the detailed strategy in terms of the game parameters.

Simple Strategy. In round n choose HIGH or LOW according to the following rules:

1. If $|x_n - x'_n| \le 2A_{\max}\tau^2$, then choose HIGH if $v_{n+1} \ge v'_n$, LOW otherwise.
2. Else if $x_n - x'_n > 2A_{\max}\tau^2$, then choose HIGH if $v'_n \le v_{n+1} + 2A_{\max}\tau$, LOW otherwise.
3. Else choose LOW if $v'_n \ge v_{n+1} - 2A_{\max}\tau$, HIGH otherwise.

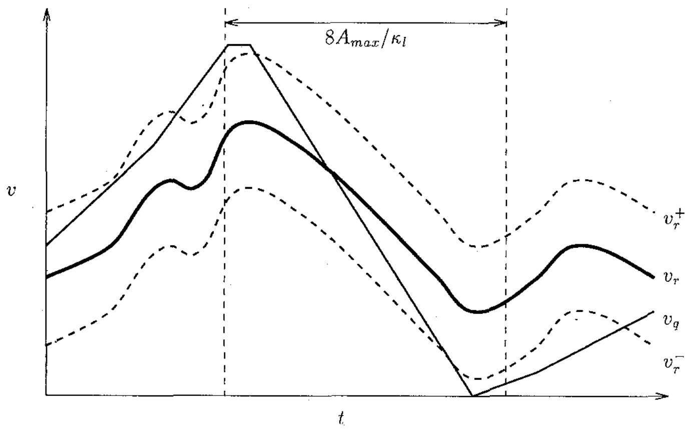

Fig. 6. Intuition for proving Lemma 3.2: a game in one dimension. We try to track an adversary whose acceleration a_r obeys the condition $|a_r(t)| \le A_{\max} - \kappa_l$. We generate our velocity function v_q timestep by timestep, trying to limit $|v_q(t) - v_r(t)|$ and $|\int(v_r(t) - v_q(t))\,dt|$. During each round we can only choose whether our acceleration $a_q(t)$ is above or below the adversary's average acceleration over the timestep by κ_l. The adversary otherwise controls our acceleration, except for the restriction that $|a_q(t)| \le A_{\max}$. Mimicking the adversary's velocity v_r to within $2A_{\max}\tau$ is straightforward, but in making up for position error, our strategy can result in a velocity error of $4A_{\max}\tau$; $v_r^+ = v_r + 4A_{\max}\tau$ and $v_r^- = v_r - 4A_{\max}\tau$ in the figure. Following a good strategy will keep $|\int(v_r(t) - v_q(t))\,dt|$ within a constant bound dependent on τ. The game conditions give us a uniform κ_l acceleration advantage over each timestep.

We make the following claim:

Claim 4.1. *Suppose that you play the Tracking Game against an adversary and follow the Simple Strategy above, and suppose that* $|x_0 - x'_0| \leq 2A_{\max}\tau^2$ *and* $|v_0 - v'_0| \leq 2A_{\max}\tau$. *Let two trajectories* Γ_r *and* Γ_q *be defined as follows*:

$$\begin{aligned} &\Gamma_r(0) = (x_0, v_0), \\ &a_r(t) = a_n(t - n\tau) \qquad \textit{for all} \quad t \in [n\tau, (n+1)\tau), \\ &\Gamma_q(0) = (x'_0, x'_0), \\ &a_q(t) = a'_n(t - n\tau) \qquad \textit{for all} \quad t \in [n\tau, (n+1)\tau). \end{aligned}$$

Then, for all t,

$$\begin{aligned} &|v_r(t) - v_q(t)| \leq 4A_{\max}\tau, \\ &|x_r(t) - x_q(t)| < \frac{8A_{\max}\tau^2}{\kappa_l} + 6A_{\max}\tau^2. \end{aligned} \tag{29}$$

Proof. Suppose the hypotheses hold. (See Figure 6.) The velocity condition in (29) holds by inspection, since $|a_r(t) - a_q(t)| \leq 2A_{\max}$ for all t. We now define

$$\begin{aligned} x_{\text{err}}(t) &= x_r(t) - x_q(t), \\ v_{\text{err}}(t) &= v_r(t) - v_q(t). \end{aligned}$$

To obtain a bound on $|x_{\text{err}}(t)|$, we first bound the length of the interval for which x_{err} and v_{err} can have the same nonzero signs. Suppose that $|x_{\text{err}}(n\tau)| > 2A_{\max}\tau^2$ and that $v_{\text{err}} > 0$. Then following the strategy ensures that $v_{\text{err}}(t_c) = 0$ at some least time $t_c > n\tau$, and, in particular, $t_c \leq n\tau + 4A_{\max}\tau/\kappa_l$. Then

$$|x_{\text{err}}(t_c) - x_{\text{err}}(n\tau)| \leq \frac{8A_{\max}^2\tau^2}{\kappa_l}.$$

The velocity condition ensures that if $|x_{\text{err}}(n\tau)| \leq 2A_{\max}\tau^2$ but $|x_{\text{err}}((n+1)\tau)| > 2A_{\max}\tau^2$, then we know that

$$|x_{\text{err}}((n+1)\tau)| < 6A_{\max}\tau^2.$$

Therefore,

$$\max_t |x_{\text{err}}(t)| < \frac{8A_{\max}^2\tau^2}{\kappa_l} + 6A_{\max}\tau^2. \tag{30}$$

4.2. Applying the Game To Prove Lemma 3.2. We now use facts proven about the Tracking Game to obtain a proof of Lemma 3.2.

Proof of Lemma 3.2. Let $\mathscr{P}$ and $\mathscr{Q}$ be functions $\mathscr{P}, \mathscr{Q}: TC \rightarrow \{\mathscr{Y}: \mathscr{Y}$ is a set of acceleration functions$\}$. Let $\mathscr{P}$ have the property that if $\mathbf{X} \in TC$ and $\mathbf{a} \in \mathscr{P}(\mathbf{X})$, then, for all times t, $\|\mathbf{a}(t)\|_\infty \leq A_{\max} - \kappa_l$. Let $\mathscr{Q}$ have the property that if $\mathbf{X} \in TC$ and $\mathbf{a} \in \mathscr{Q}(\mathbf{X})$, then, for all times t, $\|\mathbf{a}(t)\|_\infty \leq A_{\max}$. Let $\Gamma_r(t)$ respect $\mathscr{P}$.

Suppose that a maximal tracking tolerance (η_{x0}, η_{v0}) and fundamental timestep τ_0 exist such that for all $\tau \leq \tau_0$ the following is true: if state $\mathbf{X}$ is within (η_{x0}, η_{v0}) of state $\mathbf{Y}$, then the set of acceleration functions $\mathscr{Q}(\mathbf{X})$ has a uniform κ_l advantage over $\mathscr{P}(\mathbf{Y})$ for duration τ.

Suppose that (η_x, η_y) is given, and that τ obeys the conditions

$$\begin{aligned} &\tau \leq \tau_0, \\ (31) \qquad &\tau \leq \sqrt{\frac{\min(\eta_x, \eta_{x0})\kappa_l}{(8A_{\max} + 6\kappa_l)A_{\max}}}, \\ &\tau \leq \frac{\min(\eta_v, \eta_{v0})}{4A_{\max}}. \end{aligned}$$

Consider d simultaneous independent playings of the Tracking Game in which your adversary's trajectory in the ith game is $\Gamma_{r,i}$ and yours is $\Gamma_{q,i}$, and in which $\Gamma_{r,i}(0)$ and $\Gamma_{q,i}(0)$ meet the starting $(t = 0)$ closeness hypotheses of Claim 4.1.

We consider play during round n, assuming that you have followed the Simple Strategy and play has proceeded legally through the end of the previous round. By Claim 4.1, in each game i, $\Gamma_{q,i}(n\tau)$ is within (η_{x0}, η_{v0}) of $\Gamma_{r,i}(n\tau)$. Then during round n, for each of the 2^d combinations of HIGH and LOW, there is some function $\mathbf{a}^{\mathscr{Y},n} \in \mathscr{Y}_q((\Gamma_q(n\tau))$ such that the adversary can legally return[8] $a_i^{\mathscr{Y},n}$ in the ith game, provided you follow the Simple Strategy. Suppose you do so. Then by Claim 4.1, in each game i, $\Gamma_{q,i}(t)$ is within (η_{x0}, η_{v0}) of $\Gamma_{r,i}(t)$ for all $t \in [n\tau, (n+1)\tau]$. By induction, we see that (29) holds for the duration of the game.

Therefore, if, for all n, for all $t \in [n\tau, (n+1)\tau]$, $\mathbf{a}_q(t) = \mathbf{a}^{\mathscr{Y},n}(t)$, then Γ_q tracks Γ_r to L_∞ tolerance (η_x, η_v). □

4.3. Altering the Game to Prove the Robust Lemma (Lemma 3.3). We now consider a version of the Tracking Game in which the adversary is allowed to perturb your state between moves. This perturbation will correspond to a branch of the search of the reachability graph being eliminated by (b_x, b_v)-bucket pruning, and a proof of the lemma follows from a successful strategy for the game. (See Figure 7.)

4.3.1. The Tracking Game with Perturbations. Let us take the Tracking Game and alter the rules:

4. A temporary state $(\tilde{x}_{n+1}, \tilde{v}_{n+1})$ is computed for you by using your state at the

[8] That is, he can choose it as your acceleration in part 3 of the rounds; see Section 4.1.1.

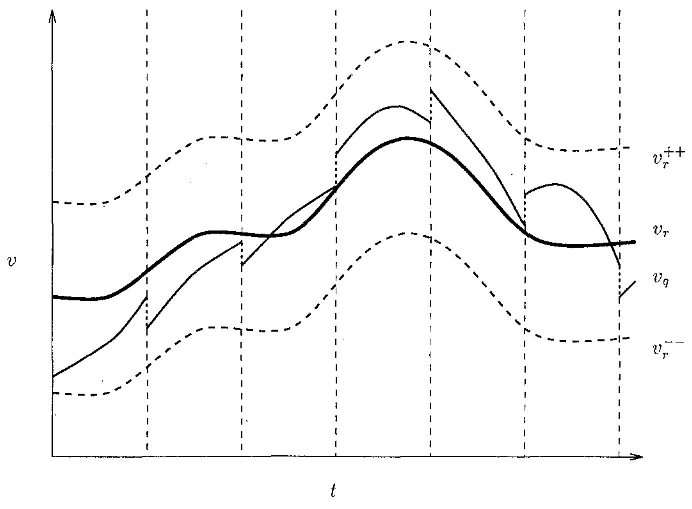

Fig. 7. Intuition for proving Lemma 3.3: another tracking game in one dimension. This time the adversary can perturb our state at the end of each round, by $\kappa_l\tau^2/2$ in position and $\kappa_l\tau/2$ in velocity; the discontinuities in v_q correspond to the perturbations. However, we can still track him, though to a looser tolerance. $v_r^{++} = v_r + 4A_{\max}\tau + \kappa_l/2$, and $v_r^{--} = v_r - 4A_{\max}\tau - \kappa_l\tau/2$.

beginning of the round (end of the previous round) (x_n, v_n) and integrating $a'_n(t)$ over the time interval $[n\tau, (n+1)\tau]$ from (x_n, v_n).

5. The adversary chooses a position perturbation x_n^Δ and a velocity perturbation v_n^Δ to be applied to your temporary position and velocity. Your position at the end of round n is

$$x'_{n+1} = \tilde{x}_{n+1} + x_n^\Delta,$$
$$v'_{n+1} = \tilde{v}_{n+1} + v_n^\Delta.$$

We call the resulting game the *Tracking Game with Perturbations*

CLAIM 4.2. *Suppose that you play the Tracking Game with Perturbations against an adversary and follow the Simple Strategy above. Suppose that at the start of the game $|x_0 - x'_0| \le 2A_{\max}\tau^2$ and $|v_0 - v'_0| \le 2A_{\max}\tau$. Furthermore, suppose that the adversary always chooses perturbations such that $|x_n^\Delta| \le \kappa_l\tau^2/2$ and $|v_n^\Delta| \le \kappa_l\tau/2$.*

Let two trajectories Γ_r and Γ_q be defined as follows:

$$\Gamma_r(n\tau) = (x_n, v_n),$$
$$a_r(t) = a_n(t - n\tau) \qquad \textit{for all} \quad t \in (n\tau, (n+1)\tau),$$
$$\Gamma_q(n\tau) = (x'_n, v'_n),$$
$$a_q(t) = a'_n(t - n\tau) \qquad \textit{for all} \quad t \in (n\tau, (n+1)\tau).$$

Then, for all t,

$$|v_r(t) - v_q(t)| \le 4A_{\max}\tau + \frac{\kappa_l \tau}{2},$$

(32)

$$|x_r(t) - x_q(t)| < \frac{(8A_{\max} + \kappa_l)^2\tau^2}{4\kappa_l} + 6A_{\max}\tau^2 + \frac{\kappa_l\tau^2}{2}.$$

PROOF. The analysis proceeds similarly to that for Claim 4.1. Again, we let $x_{\text{err}}(t) = x_r(t) - x_q(t)$, and $v_{\text{err}}(t) = v_r(t) - v_q(t)$, and bound the length of an interval for which x_{err} and v_{err} can have the same nonzero sign. Because of the velocity perturbation, our effective acceleration advantage is only $\kappa_l/2$. Therefore, if $|x_{\text{err}}(n\tau)| > 2A_{\max}\tau^2$, following the strategy ensures that $v_{\text{err}}(t_c) = 0$ at some later time $t_c \le n\tau + (8A_{\max} + \kappa_l)\tau/2\kappa_l$. Then

$$|x_{\text{err}}(t_c) - x_{\text{err}}(n\tau)| \le \frac{(8A_{\max} + \kappa_l)^2\tau^2}{4\kappa_l}.$$

Because of the perturbations at the end of a round, the $\kappa_l\tau/2$ term must be added to the velocity-tracking tolerance and $\kappa_l\tau^2/2$ must be added to the position term in the final tracking accuracy. Therefore

(33) $$\max_t |x_{\text{err}}(t)| < \frac{(8A_{\max} + \kappa_l)^2\tau^2}{4\kappa_l} + 6A_{\max}\tau^2 + \frac{\kappa_l\tau^2}{2}. \qquad \square$$

4.3.2. Using the Game with Perturbations To Prove the Lemma. We can now prove Lemma 3.3.

PROOF OF LEMMA 3.3. Let $\mathscr{P}$ and $\mathscr{Q}$ be functions $\mathscr{P}, \mathscr{Q}$: $TC \to \{\mathscr{Y}$: $\mathscr{Y}$ is a set of accelerations$\}$. Let $\mathscr{P}$ have the property that, for all $\mathbf{X} \in TC$, if $\mathbf{a} \in \mathscr{P}(\mathbf{X})$, then, for all times t, $|\mathbf{a}(t)|_\infty \le A_{\max} - \kappa_l$. Let $\mathscr{Q}$ have the properties that $\mathscr{Q}(\mathbf{X})$ is finite for all $\mathbf{X} \in TC$ and that if $\mathbf{a} \in \mathscr{Q}(\mathbf{X})$, then, for all times t, $|\mathbf{a}(t)|_\infty \le A_{\max}$.

Suppose that a maximal tracking tolerance (η_{x0}, η_{v0}) and a fundamental timestep τ_0 exist such that for all $\tau \le \tau_0$ the following is true: if state $\mathbf{X}$ is within (η_{x0}, η_{v0}) of state $\mathbf{Y}$, then the set of acceleration functions $\mathscr{Q}(\mathbf{X})$ has a uniform κ_l advantage over $\mathscr{P}(\mathbf{Y})$ for duration τ.

Suppose that (η_x, η_v) is an L_∞ tracking tolerance and that

$$\tau \le \tau_0,$$

(28)

$$\tau \le \sqrt{\frac{4\min(\eta_{x0}, \eta_x)\kappa_l}{24A_{\max}\kappa_l + 2\kappa_l^2 + (8A_{\max} + \kappa_l)^2}},$$

$$\tau \le \frac{2\min(\eta_{v0}, \eta_v)}{8A_{\max} + \kappa_l}.$$

Let $\mathscr{G}$ be the obstacle-free $(\mathscr{Q}, \tau)$-reachability graph rooted at some $\mathbf{S}^*$. Suppose that Γ_r respects $\mathscr{P}$, $\Gamma_r(0)$ is within $2A_{\max}\tau$ of $\dot{\mathbf{s}}^*$, and $\mathbf{p}_r(0) = \mathbf{S}^*$. Suppose that $\mathscr{R}$ is a subset of TC containing the (η_x, η_v) tube induced by Γ_r, and that $\mathscr{G}'$ results from some breadth-first $(\kappa_l\tau^2/2, \kappa_l\tau/2)$-pruning of the maximal subgraph of $\mathscr{G}$ that lies in $\mathscr{R}$.

Consider d simultaneous independent playings of the Tracking Game with Perturbations in which the adversary's trajectory in the ith game is $\Gamma_{r,i}$ and yours is $\Gamma_{q,i}$. Suppose that all the $\Gamma_{r,i}(0)$ and $\Gamma_{q,i}(0)$ meet the starting $(t = 0)$ closeness hypothesis of Claim 4.2. Let $(\mathbf{x}_n, \mathbf{v}_n)$ denote your state (i.e, the vector of your states in the individual games) at the beginning of round n.

Now, suppose you follow the Simple Strategy. Because the breadth-first pruning buckets and perturbation size are both $(\kappa_l\tau^2/2, \kappa_l\tau/2)$, by induction on n, Claim 4.2 and the definitions of $\mathscr{P}$, $\mathscr{Q}$, $\mathscr{G}$, and $\mathscr{G}'$ assure that the following can legally occur in each round n of the game:

1. The adversary chooses his acceleration in each game, yielding a vector $\Gamma_r((n + 1)\tau)$ of his new states.
2. You then chose LOW or HIGH separately for each game using the Simple Strategy, yielding a d-vector of choices.
3. Then the adversary chooses a member of $\mathscr{Q}(\Gamma_q(n\tau))$ for *your* (vector of) accelerations, obeying your choices.
4. Then the adversary perturbs your state $(\tilde{\mathbf{x}}_{n+1}, \tilde{\mathbf{v}}_{n+1})$ if and only if it is not in $\mathscr{G}'$. In particular, if he perturbs your state, he perturbs it to the state in $\mathscr{G}'$ that caused $(\tilde{\mathbf{x}}_{n+1}, \tilde{\mathbf{v}}_{n+1})$ to be pruned from $\mathscr{G}'$. Thus, for some graph trajectory $\Gamma_q^{(n+1)}$ in $\mathscr{G}'$ and some $m \leq n + 1$, $\Gamma_q^{(n+1)}(m\tau)$ is this state.

If you and the adversary play in this manner, then by Claim 4.2, for all n:

1. $(\mathbf{x}_n, \mathbf{v}_n)$ will be within (η_x, η_v) of $\Gamma_r(n\tau)$.
2. $(\tilde{\mathbf{x}}_n, \tilde{\mathbf{v}}_n)$ will be within (η_x, η_v) of $\Gamma_r(n\tau)$.
3. Your trajectory $\Gamma_q(t)$ will be within (η_x, η_v) of $\Gamma_r(t)$ for all $t \in [n\tau, (n + 1)\tau]$.

Furthermore, for each n there will be some $\Gamma_q^{(n)}$ in $\mathscr{G}'$ such that, for some $m \leq n$:

1. $\Gamma_q^{(n)}(m\tau) = \Gamma_q(n\tau)$.
2. $\Gamma_q^{(n)}(t - (n - m)\tau) = \Gamma_q(t)$ for all $t \in [n\tau, (n + 1)\tau)$.

Since this holds for all n, there must be some trajectory Γ_q^* in $\mathscr{G}'$ such that, for some $T_q \leq T_r$, $\Gamma_q^*(T_q)$ approximates $\Gamma_r(T_r)$ to within (η_x, η_v). □

5. Algorithms and Bounds

5.1. Algorithm Outlines. We now present the algorithms in outline form. These outlines rely heavily on definitions and descriptions in the preceding chapters. Fuller descriptions of how certain parameters are chosen are given in Sections 5.2 and 5.3.

Recall the definitions of $\bar{\mathscr{H}}(\mathbf{X}, \tau)$ and $\mathscr{H}(\mathbf{X}, \tau)$ from Sections 3.1.2 and 3.1.4. For given $\boldsymbol{\mu}$, $\bar{\mu}$, and τ, we define the maps $\mathscr{H}_\tau$, $\bar{\mathscr{H}}_\tau$: $TC \to \{\mathscr{Y} : \mathscr{Y}$ is a set of acceleration functions} by

$$\mathscr{H}_\tau(\mathbf{X}) = \mathscr{H}(\mathbf{X}, \tau) \tag{34}$$

and

$$\bar{\mathscr{H}}_\tau(\mathbf{X}) = \bar{\mathscr{H}}(\mathbf{X}, \tau). \tag{35}$$

Near-Extremal Algorithm Outline. Given a kinodynamic planning problem $(\mathcal{O}, \mathbf{S}, \mathbf{G}, \bar{\mathbf{f}}, \bar{\mathbf{v}}, l, \mathscr{M}, c_0, c_1)$ for an L_2 Cartesian robot or an open-chain manipulator, the Near-Extremal Algorithm does the following:

1. Computes consistent parameters $(\mu, \tau_0, \eta_{x0}, \eta_{v0}, \kappa_l)$ obeying the conditions in Section 3.2.
2. Chooses a timestep τ and an acceleration-space discretization $\boldsymbol{\mu}$ such that $\mathscr{H}_\tau$ obeys the conditions of Lemma 3.2 with respect to the dynamic bounds of ε-time-rescaled trajectories. τ and all μ_i are $\Theta(\varepsilon)$. If the ith joint has configuration space S^1 (as revolute joints without limits do), then μ_i must be chosen so that $K_i \mu_i \tau^2 = 4\pi$ for some integer K_i.
3. It then chooses the root vertex state $\mathbf{S}^*$ such that $\mathbf{s}^* = \mathbf{s}$ and
$$\left\| \dot{\mathbf{s}}^* - \frac{1}{1+\varepsilon}\,\dot{\mathbf{s}} \right\| \le \min_i \mu_i \tau.$$
4. Let $\mathscr{G}$ be the $(\boldsymbol{\mu}, \tau)$-reachability graph rooted at $\mathbf{S}^*$ whose edges are $(1-\varepsilon)\delta_v(c_0, c_1)$-safe $(\boldsymbol{\mu}, \tau)$-bangs. The algorithm searches for the shortest path from $\mathbf{S}^*$ to any vertex that is within (η_x, η_v) of $(\mathbf{g}, (1(1+\varepsilon))\dot{\mathbf{g}})$. η_x and η_v are both $\Theta(\varepsilon)$. The search is done breadth-first. The algorithm constructs the graph on the fly, so that it only computes what it searches.
5. If a $\delta_v(c_0, c_1)$-safe solution exists, then the algorithm returns the $(1-\varepsilon)\delta_v(c_0, c_1)$-safe graph trajectory that corresponds to the first vertex it finds that meets the approximation conditions at the goal state.

True Extremal Algorithm Outline. Given a kinodynamic planning problem instance $(\mathcal{O}, \mathbf{S}, \mathbf{G}, \bar{\mathbf{f}}, \bar{\mathbf{v}}, \mathscr{M}, c_0, c_1)$ for an L_2 Cartesian robot or an open-chain manipulator, the True-Extremal Algorithm does the following:

1. Computes consistent parameters $(\mu, \tau_0, \eta_{x0}, \eta_{v0}, \kappa_l)$ obeying the conditions in Section 3.2.
2. It chooses a timestep τ and an extremal controls discretization $\bar{\mu}$ such that $\bar{\mathscr{H}}_\tau$ obeys the conditions of Lemma 3.3 with respect to the dynamics bounds of ε-time-rescaled trajectories. Both τ and $\bar{\mu}$ are $\Theta(\varepsilon)$.
3. Chooses the root vertex $\mathbf{S}^*$ by ε-rescaling $\mathbf{S}$; i.e., $\mathbf{s}^* = \mathbf{s}$ and $\dot{\mathbf{s}}^* = (1/(1+\varepsilon))\dot{\mathbf{s}}$.
4. Let $\mathscr{G}$ be the reachability graph rooted at $\mathbf{S}^*$ whose edges are $(1-\varepsilon)\delta_v(c_0, c_1)$-safe $(\bar{\mu}, \tau)$-bangs. The algorithm searches for the shortest path from $\mathbf{S}^*$ to any vertex that is within (η_x, η_v) of $(\mathbf{g}, (1/(1+\varepsilon)\dot{\mathbf{g}})$. η_x and η_v are both $\Theta(\varepsilon)$. The search is done breadth-first with (b_x, b_v)-bucket pruning in which b_x is $\Theta(\varepsilon^3)$ and b_v is $\Theta(\varepsilon^2)$. The algorithm constructs the graph on the fly, so that it only computes what it searches.

5. If a $\delta_v(c_0, c_1)$-safe solution exists, then the algorithm returns the $(1-\varepsilon)\delta_v(c_0, c_1)$-safe graph trajectory that corresponds to the first vertex it finds that meets the approximation conditions at the goal state.

Thus, the controls of a trajectory found by the True-Extremal Algorithm are constant and extremal over each individual timestep, while the acceleration will be extremal but may vary with time, according to the integration of (10). This contrasts with the trajectories found by the Near-Extremal Algorithm, whose *accelerations* are constant and near-extremal over each timestep, but whose controls may be time-varying, as given by (1).

Note that for robots with revolute joints there is generally no closed form for trajectory segments corresponding to true $(\bar{\mu}, \tau)$-bangs. However, using an rth-order numerical integration procedure will yield trajectory segments corresponding to controls within $O(\tau^r)$ (and therefore within $O(\varepsilon^r)$) of being constant and extremal. Thus, as long as ε is sufficiently small, it would appear that we could use, say, $r = 4$ and Runge–Kutta numerical integration. However, in order to guarantee that the trajectory segments are executable under the force bounds, we cannot use truly extremal controls, but ones that are within some polynomial of ε of being extremal. For this reason, we consider the Near-Extremal Algorithm to be only theoretical.

5.2. Search-Space Bounds for Cartesian Manipulators. Here, we first consider an L_2 Cartesian robot with acceleration bound $\bar{a}$. We choose an acceleration discretization $\boldsymbol{\mu}$ and a $\Theta(\varepsilon)$ scalar κ_l such that:

(a) All μ_i are $\Theta(\varepsilon)$.
(b) For all $\mathbf{X}, \mathbf{Y} \in TC$, the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations $\mathscr{H}(\mathbf{X}, \tau)$ at $\mathbf{X}$ has a uniform κ_l advantage over $\mathscr{A}'_{\mathbf{Y}}$ for any τ.

We then derive a similar choice of $\boldsymbol{\mu}$ for cases when an L_2-norm force bound $\bar{f}$ is given instead of an acceleration bound. We then return our focus to the acceleration bound case and examine the Near-Extremal Algorithm in detail to illustrate how the algorithm chooses the reachability graph parameters $\boldsymbol{\mu}$ and τ. We briefly discuss the derivation of the control discretization parameter $\bar{\mu}$ for the True-Extremal Algorithm.

5.2.1. Parameter Choices and Acceleration Advantages for an L_2 Cartesian Robot. The set of constant accelerations obeying the L_2-norm bound $\bar{a}$ has a uniform acceleration advantage of

$$\frac{\bar{a}(2\varepsilon + \varepsilon^2)}{\sqrt{d}(1+\varepsilon)^2} > \frac{\bar{a}\varepsilon}{2\sqrt{d}}$$

over the set of accelerations obeying the L_2-norm bound $\bar{a}/(1+\varepsilon)^2$. Recall that $\|\mathbf{y}\|_\infty \le \|\mathbf{y}\|_2 \le \sqrt{d}\|\mathbf{y}\|_\infty$ for any $\mathbf{y} \in \mathbb{R}^d$ (21). If $\boldsymbol{\mu}$ and κ_l have the property that

(36) $$\|\boldsymbol{\mu}\|_\infty \le \kappa_l \le \frac{\bar{a}\varepsilon}{4\sqrt{d}},$$

then, for any state $\mathbf{X}$ and any duration τ, the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations $\mathscr{H}(\mathbf{X}, \tau)$ has a uniform κ_l advantage over the set of acceleration functions obeying the L_2-norm bound $\bar{a}/(1+\varepsilon)^2$.

For any state $\mathbf{X}$, consider the sets of feasible instantaneous accelerations $\mathscr{A}_{\mathbf{X}}$ and $\mathscr{A}'_{\mathbf{X}}$ that correspond to force bounds $\bar{f}$ and $\bar{f}/(1+\varepsilon)^2$, respectively. Recall that the inertia matrix $\mathbf{M}$ is symmetric and positive definite and thus orthogonal. In addition, recall that $\mathbf{M}$ is constant for a Cartesian robot. If $\lambda_{\min}$ and $\lambda_{\max}$ are the minimum and maximum eigenvalues of $\mathbf{M}$, respectively, and $\mathbf{a} \in \partial\mathscr{A}_{\mathbf{X}}$, then $\|\mathbf{a} - \mathbf{a}'\|_2 \geq \bar{f}(2\varepsilon + \varepsilon^2)/\lambda_{\max}(1+\varepsilon)^2$. More compactly,

$$\|\mathbf{a} - \mathbf{a}'\|_2 > \frac{\bar{f}\varepsilon}{2\lambda_{\max}}. \tag{37}$$

It follows that if

$$\kappa_l \leq \frac{\bar{f}\varepsilon}{4\sqrt{d}\lambda_{\max}} \quad \text{and} \quad \|\boldsymbol{\mu}\|_\infty \leq \frac{\bar{f}\varepsilon}{4\sqrt{d}\lambda_{\max}}, \tag{38}$$

then $\mathscr{H}(\mathbf{X}, \tau)$ has a uniform κ_l advantage over $\mathscr{A}'(\mathbf{Y})$ for any τ, $\mathbf{X}$, and $\mathbf{Y}$.

5.2.2. Timestep Choice and Search-Space Bounds for the Near-Extremal Algorithm. By applying the Safe Tracking Lemma, we find a family of η_x and η_v that are $\Theta(\varepsilon)$ and guarantee that any trajectory tracking a $\delta_v(c_0, \gamma_1)$-safe trajectory to tolerance (η_x, η_v) will be $(1-\varepsilon)\delta_v(c_0, c_1)$-safe. Specifically.

$$\begin{aligned} \eta_v &\leq \frac{c_0\varepsilon}{c_1(1-\varepsilon)+\beta}, \\ \eta_x &= \beta\eta_v. \end{aligned} \tag{15}$$

It is simplest to choose $\beta = 1$, which implies $\eta_x = \eta_v$. However, by using a technique from [1] we show how to choose β to minimize the bound on the possible size of the reachability graph searched by the Near-Extremal Algorithm applied to a Cartesian robot with L_2 acceleration and velocity bounds $\bar{a}$ and $\bar{v}$.

First, we choose an underlying acceleration discretization $\boldsymbol{\mu}$ consistent with a uniform acceleration advantage κ_l (36):

$$\mu_i = \mu = \kappa_l = \frac{\bar{a}\varepsilon}{4\sqrt{d}}. \tag{39}$$

We parametrize our choice of timestep τ as a function of β. We use Lemma 3.2 to obtain

$$\begin{aligned} \tau_x(\beta) &= \frac{\sqrt{c_0\varepsilon^2\beta}}{\sqrt{2d(c_1(1-\varepsilon)+\beta)(8\sqrt{2d}+3\bar{a}\varepsilon)}}, \\ \tau_v(\beta) &= \frac{c_0\varepsilon}{4\bar{a}(c_1(1-\varepsilon)+\beta)}, \\ \tau(\beta) &= \min(\tau_x(\beta), \tau_v(\beta)). \end{aligned} \tag{40}$$

Since τ_x monotonic and β must be positive, $\tau(\beta)$ is maximized when $\tau_x(\beta)$ and $\tau_v(\beta)$ are equal. Solving for β to get a positive τ yields

$$\beta = \tfrac{1}{2}\left(-c_1(1-\varepsilon) + \sqrt{c_1^2(1-\varepsilon)^2 + \frac{c_0(16\sqrt{d}+3\varepsilon)}{2\bar{a}}}\right). \tag{41}$$

We use the Coupled Tracking Lemma (Lemma 3.2) to guarantee that a tracking trajectory would obey the velocity bound. Thus we choose

$$\tau = \min\left(\frac{\bar{v}\varepsilon}{4\bar{a}}, \frac{c_0\varepsilon}{2\bar{a}(c_1(1-\varepsilon) + \sqrt{c_1^2(1-\varepsilon)^2 + c_0(16\sqrt{d}+3\varepsilon)/2\bar{a}})}\right). \tag{42}$$

Given the desired start state S, the Near-Extremal Algorithm chooses the root vertex S* to meet the conditions at $t = 0$ in the Coupled Tracking Lemma, relative to ε-time-rescaled start state S′. (Recall (8).) If an optimal trajectory Γ_{opt} exists, then there will be some graph trajectory Γ_q that tracks Γ'_{opt} to the tolerance (η_x, η_v) determined by the Safe Tracking Lemma and the choice of β above. (See (15) and (41).) Since (η_x, η_v) meets the conditions of the Safe Tracking Lemma, such a Γ_q would be guaranteed to be $(1-\varepsilon)\delta_v(c_0, c_1)$-safe. Since η_x and η_v are $O(\varepsilon)$, Γ_q would meet the approximation criteria at the desired start and goal states.

We can now bound the size of the reachability graph for a Cartesian robot whose maximum speed is $\bar{v}$ and whose configuration space is contained within a d-dimensional cube with diameter l. Let

$$\mu = \min_i \mu_i.$$

Then the total number of possible velocities for reachability graph vertices is bounded above by $(2\bar{v}/\mu\tau + 1)^d$ and the number of configurations by $(l/\mu\tau^2 + 1)^d$. Thus, the total number of reachability graph vertices is

$$G_{\mathscr{V}}(\bar{a}, \mu, \tau, \bar{v}, l, d) < \left(\left(\frac{2\bar{v}}{\mu\tau} + 1\right)\left(\frac{l}{\mu\tau^2} + 1\right)\right)^d. \tag{43}$$

Using the choices of $\boldsymbol{\mu}$ and τ above in (36) (or (38)), (39), and (42), we see that the right-hand side of (43) is $O((d^2\bar{v}l)^d(1/\varepsilon)^{5d})$. Recalling the definition of the $(\boldsymbol{\mu}, \tau)$-extremal shell, the out-degree of this graph is bounded above by $d(\bar{a}/\mu)^{d-1}$. Thus the total number of graph edges is

$$\begin{aligned} G_{\mathscr{E}}(\bar{a}, \mu, \tau, \bar{v}, l, d) &< d\left(\frac{\bar{a}}{\mu}\right)^{d-1}\left(\left(\frac{2\bar{v}}{\mu\tau} + 1\right)\left(\frac{l}{\mu\tau^2} + 1\right)\right)^d \\ &= O\left(d^{3d}\bar{a}^{d-1}(\bar{v}l)^d\left(\frac{1}{\varepsilon}\right)^{6d-1}\right). \end{aligned} \tag{44}$$

We can follow a similar, straightforward development if L_2 force bounds (instead of acceleration bounds) are used. In particular, we choose $\boldsymbol{\mu}$ using (38), define analogues of (40), and choose β to maximize τ to get the necessary reachability graph parameters. We substitute $\bar{f}/\lambda_{\min}$ for $\bar{a}$ in (45) to obtain bounds with the same exponents on the size of the reachability graph.

We summarize these results for a Cartesian robot obeying L_2 dynamics bounds with the following lemma.

LEMMA 5.1. *Given a kinodynamic planning problem for a Cartesian robot with L_2-norm dynamics bounds and given an approximation parameter ε such that $0 < \varepsilon < 1$, the Near-Extremal Algorithm will search a reachability graph with $O(c^d(1/\varepsilon)^{6d-1})$ vertices and edges. c is a constant dependent on the kinodynamics specifications and polynomial in d. If an optimal (safe) solution Γ_{opt} exists, then some graph trajectory is ε-optimal and will be found.*

5.2.3. True-Extremal Algorithm Search Space for a Cartesian Robot. Now, suppose the (Cartesian) robot obeys L_2 force bound $\bar{f}$, and again let $\lambda_{\min}$ and $\lambda_{\max}$ denote the minimum and maximum eigenvalues of its inertia matrix $\mathbf{M}$. If $\|\mathbf{f}_a\|_2, \|\mathbf{f}_b\|_2 \le \bar{f}$ and $\|\mathbf{f}_a - \mathbf{f}_b\|_\infty \le \bar{\mu}$, then

$$\|\mathbf{M}^{-1}\mathbf{f}_a - \mathbf{M}^{-1}\mathbf{f}_b\|_2 \le \frac{\bar{\mu}\sqrt{d}}{\lambda_{\min}}.$$

At the same time, if $\|\mathbf{f}_a\|_2 = \bar{f}$ and $\|\mathbf{f}_b\|_2 = \bar{f}/(1+\varepsilon)^2$, then

$$\|\mathbf{M}^{-1}\mathbf{f}_a - \mathbf{M}^{-1}\mathbf{f}_a\|_2 > \frac{\bar{f}\varepsilon}{2\lambda_{\max}}.$$

Therefore, if we choose κ_l as in (38) and

$$\bar{\mu} \le \frac{\lambda_{\min}\bar{f}\varepsilon}{4d\lambda_{\max}}, \tag{45}$$

we guarantee that the set of true $(\bar{\mu}, \tau)$-bangs $\mathscr{H}(\mathbf{X}, \tau)$ (Section 3.1.2) has a uniform κ_l advantage over $\mathscr{A}'(\mathbf{Y})$ for any two states $\mathbf{X}$ and $\mathbf{Y}$ over any duration τ.

Again using the Safe Tracking Lemma (Lemma 3.1) to find a family of sufficiently close tracking tolerances (η_x, η_v), we can now apply the Robust Coupled Tracking Lemma (Lemma 3.3) to find a maximal timestep τ using the uniform acceleration advantage κ_l above. The algorithm's choice of $\mathbf{S}^*$ trivially satisfies the $t = 0$ condition of the lemma. Clearly, since κ_l, η_x, and η_v are $\Theta(\varepsilon)$, τ will be $\Theta(\varepsilon)$ also. Finally, the algorithm chooses pruning-bucket dimensions prescribed by Lemma 3.3:

$$\begin{aligned} b_x &= \frac{\kappa_l\tau^2}{2} && \text{in configuration, and} \\ b_v &= \frac{\kappa_l\tau}{2} && \text{in velocity.} \end{aligned} \tag{46}$$

These quantities clearly will be $\Theta(\varepsilon^3)$ and $\Theta(\varepsilon^2)$, respectively. The number of buckets is therefore $O((l/\varepsilon)^{5d})$. Recalling the definition of true $(\bar{\mu}, \tau)$-bangs, we see that each vertex has $O((1/\varepsilon)^{d-1})$ out-edges. Since the algorithm explores the out-edges of at most one vertex from each bucket, the True-Extremal Algorithm will search $O((1/\varepsilon)^{6d-1})$ vertices and edges of a reachability graph. Summarizing:

LEMMA 5.2. *Given a kinodynamic planning problem for a Cartesian robot with L_2-norm dynamics bounds and given an approximation parameter ε such that $0 < \varepsilon < 1$, the True-Extremal Algorithm will search a reachability graph with $O(c^d(1/\varepsilon)^{6d-1})$ vertices and edges. c is a constant dependent on the kinodynamic specifications and polynomial in d. If an optimal (safe) solution Γ_{opt} exists, then some graph trajectory is ε-approximately optimal and will be found.*

Since the complexity of each of our algorithms is the number of graph edges that might be explored multiplied by the time it takes to check the $(1 - \varepsilon)\delta(c_0, c_1)$-safety of each edge (as a trajectory segment), we determine the asymptotic complexities by combining this lemma with the bounds from Section 5.4. This yields Theorem 2.1.

5.3. Applying the Coupled Tracking Lemma and Robots with State-Dependent Dynamics. We now derive lower bounds for the discretization and timestep parameters used by the Near-Extremal and True-Extremal Algorithms for robots obeying the open chain dynamics equation (1). The Near-Extremal Algorithm must choose underlying acceleration discretization μ_i and a timestep τ ensuring that if a δ_v-safe solution exists, then the algorithm will find an ε-approximately optimal solution. We describe how to find sufficiently small μ_i and τ that are $\Theta(\varepsilon)$. We also show that the True-Extremal Algorithm can choose a timestep τ and a control discretization parameter $\bar{\mu}$ that are $\Theta(\varepsilon)$ and that ensure the algorithm will find an ε-approximately optimal solution under the same conditions.

Let the problem parameters and ε be given. Recall from Section 3.2.4 that $\mathscr{A}$ and $\mathscr{A}'$ denote mappings from TC to $\{\mathscr{Y}: \mathscr{Y}$ is a set of acceleration functions$\}$ under bounds $\bar{\mathbf{f}}$ and $\bar{\mathbf{f}}'$ (from (18)), respectively. Now, let trajectory Γ_r respect $\mathscr{A}'$, and let Γ_q respect $\mathscr{A}$.

We now sketch how to find a set of *consistent parameters* $(\mu, \tau_0, \eta_{x0}, \eta_{v0}, \kappa_l)$: a maximal discretization parameter μ for either acceleration or control, a maximal timestep τ_0, a maximal tracking tolerance (η_{x0}, η_{v0}), and a uniform acceleration advantage κ_l. For any timestep $\tau \le \tau_0$ and any trajectory Γ_r that respects $\mathscr{A}'$, the following will be true: if state $\mathbf{X}$ is within (η_{x0}, η_{v0}) of $\Gamma_r(n\tau)$, and both $\bar{\mu}$, $\|\boldsymbol{\mu}\|_\infty \le \mu$ then both the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations $\mathscr{H}(\mathbf{X}, \tau)$ and the set $\bar{\mathscr{H}}(\mathbf{X}, \tau)$ of true $(\bar{\mu}, \tau)$-bangs have uniform κ_l advantages over $\mathscr{A}'(\Gamma_r(t - n\tau))$ for duration τ. We will show how to find sufficiently small μ, κ_l, τ_0, η_{x0}, and η_{v0} that are all $\Theta(\varepsilon)$.

Having obtained these parameters, the algorithms can apply the Coupled Tracking Lemmas to choose τ. This leads us to bounds on the size of the reachability graphs the algorithms search.

5.3.1. Sufficient Conditions. Observe that the sets of feasible instantaneous accelerations $\mathscr{A}_{\mathbf{X}}$ and $\mathscr{A}'_{\mathbf{X}}$ are d-dimensional parallelepipeds for all $\mathbf{X} \in TC$. Let $\lambda_{\min}$ and $\lambda_{\max}$ be the minimum[9] and maximum eigenvalues of the inertia tensor $\mathbf{M}(\mathbf{p})$ over all positions. Let $f_{\min} = \min_i \bar{\mathbf{f}}_i$, and let $\bar{f} = \max_i \bar{\mathbf{f}}_i$. Let the force required to hold the robot stationary in the presence of gravity obey the bound $(1 - \kappa_f)\bar{f}$. From Section 3.2.3, the minimum L_2 distance between $\partial\mathscr{A}_{\mathbf{X}}$ and $\partial\mathscr{A}'_{\mathbf{X}}$ is greater than

$$(47) \qquad \begin{aligned} \alpha &= \frac{\kappa_f f_{\min}\varepsilon}{2\lambda_{\max}} && \text{generally, and} \\ \alpha &= \frac{f_{\min}\varepsilon}{2\lambda_{\max}} && \text{in the absence of gravity.} \end{aligned}$$

Suppose that Γ_q respects the mapping $\mathscr{A}$ (see Section 3.2.4) and tracks Γ_r to tolerance (η_{x0}, η_{v0}), and fix $\tau_0 > 0$. For any $\tau \le \tau_0$, consider what happens over a timestep $[n\tau, (n+1)\tau]$. For any $t \in [n\tau, (n+1)\tau]$, $\Gamma_r(t) - \Gamma_r(n\tau)$ and $\Gamma_q(t) - \Gamma_q(n\tau)$ belong to sets of possible state perturbations that are functions of τ_0. That is, minimal sets $\gamma_r(\tau_0), \gamma_q(\tau_0) \subset TC$ exist such that, for all $h \le \tau_0$,

$$(48) \qquad \Gamma_r(t+h) - \Gamma_r(t) \in \gamma_r(\tau_0) \quad \text{and} \quad \Gamma_q(t+h) - \Gamma_q(t) \in \gamma_q(\tau_0).$$

State perturbations about $\Gamma_r(n\tau)$ and $\Gamma_q(n\tau)$ result in perturbations of $\partial\mathscr{A}'_{\Gamma_r(t)}$ and $\partial\mathscr{A}_{\Gamma_q(t)}$ about $\partial\mathscr{A}'_{\Gamma_r(n\tau)}$ and $\partial\mathscr{A}_{\Gamma_q(n\tau)}$. We now bound the magnitudes of these perturbations as functions of η_{x0}, η_{v0}, and τ_0, respectively.

Let $\mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f})$ be given by

$$(10) \qquad \mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f}) = \mathbf{M}^{-1}(\mathbf{p})(\mathbf{f} - [\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}] - \mathbf{G}(\mathbf{p})),$$

and let $B_\delta(\mathbf{y})$ denote the δ-ball about $\mathbf{y}$. We denote global perturbation bounds

$$(49) \qquad \begin{aligned} h_\tau(\tau_0) &= \max \|\mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f}') - \mathbf{a}(\mathbf{p} + \Delta\mathbf{p}_{\gamma_r}, \dot{\mathbf{p}} + \Delta\dot{\mathbf{p}}_{\gamma_r}, \mathbf{f}')\|_2, \\ h_{q0}(\tau_0) &= \max \|\mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f}) - \mathbf{a}(\mathbf{p} + \Delta\mathbf{p}_{\gamma q}, \dot{\mathbf{p}} + \Delta\dot{\mathbf{p}}_{\gamma q}, \mathbf{f})\|_2, \\ h_{q1}(\eta_{x0}, \eta_{v0}) &= \max \|\mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f}) - \mathbf{a}(\mathbf{p} + \Delta\mathbf{p}_{\eta_{x0}}, \dot{\mathbf{p}} + \Delta\dot{\mathbf{p}}_{\eta_{v0}}, \mathbf{f})\|_2, \end{aligned}$$

where the maxima are taken as:

(a) $(\mathbf{p}, \dot{\mathbf{p}})$ ranges over TC.
(b) $\mathbf{f}$ and $\mathbf{f}'$ obey the constraints $f_i \le \bar{f}_i$ and $f'_i \le \bar{f}'_i$ for all coordinates i.
(c) $(\Delta\mathbf{p}_{\gamma_r}, \Delta\dot{\mathbf{p}}_{\gamma_r})$ ranges over $\gamma_r(\tau_0)$.
(d) $(\Delta\mathbf{p}_{\gamma_q}, \Delta\dot{\mathbf{p}}_{\gamma_q})$ ranges over $\gamma_q(\tau_0)$.
(e) $(\Delta\mathbf{p}_{\eta_{x0}}, \Delta\dot{\mathbf{p}}_{\eta_{v0}})$ ranges over $B_{\eta_{x0}}(\mathbf{0}) \times B_{\eta_{v0}}(\mathbf{0})$.

[9] $\lambda_{\min}$ is the only parameter that is neither given in the problem instance nor bounded (below) in the derivation; a loose bound is given by the minimum of the smallest link mass and smallest link inertia (in generalized units). A bound for $\lambda_{\max}$ follows from the bound on $\|\mathbf{M}(\mathbf{p})\|$ found in Appendix A.

Note that $h_{q0}(\tau_0)$ determines the acceleration perturbation bound E_A from Section 3.1.4, modulo norm.

Suppose that $\|\boldsymbol{\mu}\|_\infty = \mu$ and $\tau \leq \tau_0$. Recall (21). If

$$\mu\sqrt{d} + \kappa_l\sqrt{d} + h_r(\tau_0) + h_{q0}(\tau_0) + h_{q1}(\eta_{x0}, \eta_{v0}) \leq \alpha, \tag{50}$$

then for an interval of length τ, the $(\boldsymbol{\mu}, \tau)$-extremal shell of accelerations at $\Gamma_q(n\tau)$, namely $\mathscr{H}(\Gamma_q(n\tau), \tau)$, will have a uniform κ_l advantage over the set of acceleration functions $\mathscr{A}'(\Gamma_r(n\tau))$ feasible under the bound $\bar{\mathbf{f}}$. Similarly, if

$$\frac{\bar{\mu}\sqrt{d}}{\lambda_{\min}} + \kappa_l\sqrt{d} + h_r(\tau_0) + h_{q0}(\tau_0) + h_{q1}(\eta_{x0}, \eta_{v0}) \leq \alpha, \tag{51}$$

then the set of true $(\bar{\mu}, \tau)$-bangs at $\Gamma_q(n\tau)$, $\bar{\mathscr{H}}(\Gamma_q(n\tau), \tau)$ will have a uniform κ_l advantage over $\mathscr{A}'(\Gamma_r(n\tau_0))$.

The two constraints (50) and (51) are obeyed if

$$\mu, \frac{\bar{\mu}}{\lambda_{\min}}, \kappa_l \leq \frac{\alpha}{5\sqrt{d}} \quad \text{and} \quad h_r(\tau_0), h_{q0}(\tau_0), h_{q1}(\eta_{x0}, \eta_{v0}) < \frac{\alpha}{5}. \tag{52}$$

5.3.2. Bounds for Perturbations. We show that $h_r(\tau_0)$ and $h_{q0}(\tau_0)$ are $O(\tau_0)$, and that $h_{q1}(\eta_{x0}, \eta_{v0})$ is $O(\eta_{x0})$ and $O(\eta_{v0})$. We first note that $\|\mathbf{M}(\mathbf{p}) - \mathbf{M}(\mathbf{p} + \Delta\mathbf{p})\|$ is $O(\|\Delta\mathbf{p}\|$, or $O(d^2\|\Delta\mathbf{p})$ if we include the degrees of freedom d. Now, define

$$\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) = \mathbf{f} - [\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}] - \mathbf{G}(\mathbf{p}). \tag{53}$$

Substituting into (49) and differentiating

$$h_r(\tau_0) < \tau_0 \max\left\|\left[\frac{\partial\mathbf{M}^{-1}}{\partial\mathbf{p}}\dot{\mathbf{p}}\right]\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) + \mathbf{M}^{-1}(\mathbf{p})\left[\frac{\partial\mathscr{F}}{\partial\mathbf{p}}\dot{\mathbf{p}} + \frac{\partial\mathscr{F}}{\partial\dot{\mathbf{p}}}\mathbf{a}\right]\right\|,$$

$$h_{q0}(\tau_0) \leq \tau_0 \max\left\|\left[\frac{\partial\mathbf{M}^{-1}}{\partial\mathbf{p}}\dot{\mathbf{p}}\right]\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) + \mathbf{M}^{-1}(\mathbf{p})\left[\frac{\partial\mathscr{F}}{\partial\mathbf{p}}\dot{\mathbf{p}} + \frac{\partial\mathscr{F}}{\partial\dot{\mathbf{p}}}\mathbf{a}\right]\right\|, \tag{54}$$

$$h_{q1}(\eta_x, \eta_v) \leq \max\left\|\left[\frac{\partial\mathbf{M}^{-1}}{\partial\mathbf{p}}\Delta\mathbf{p}\right]\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) + \mathbf{M}^{-1}(\mathbf{p})\left[\frac{\partial\mathscr{F}}{\partial\mathbf{p}}\Delta\mathbf{p} + \frac{\partial\mathscr{F}}{\partial\dot{\mathbf{p}}}\Delta\dot{\mathbf{p}}\right]\right\|.$$

Now, recall a derivation of (1), say from [9], and recall that

$$[\dot{\mathbf{p}}^T(t)\mathbf{C}(\mathbf{p}(t))\dot{\mathbf{p}}(t)]_i = \dot{\mathbf{p}}^T(t)\mathbf{C}^i(\mathbf{p}(t))\dot{\mathbf{p}}(t), \tag{2}$$

where

$$\mathbf{C}^i(\mathbf{p}(t))_{jk} = \frac{\partial M_{jk}(\mathbf{p})}{\partial p_k} - \frac{1}{2}\frac{\partial M_{jk}(\mathbf{p})}{\partial p_i}.$$

Now, $\mathbf{M}(\mathbf{p})$ is simply the inertia tensor, and

$$\mathbf{G}(\mathbf{p}) = \frac{\partial U_G(\mathbf{p})}{\partial \mathbf{p}},$$

where $U_G(\mathbf{p})$ is the gravitational potential energy at state $\mathbf{p}$. Hence, each component of $\mathbf{M}(\mathbf{p})$ and $\mathbf{G}(\mathbf{p})$ is a sum of the products of components of $\mathbf{p}$ and their sines and cosines (e.g., p_i, $\cos(p_j)$, etc.) This implies that $\partial\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}})/\partial\mathbf{p}$ and $\partial\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}})/\partial\dot{\mathbf{p}}$ are globally bounded. Specifically, it follows that there are C_r, C_{q0}, C_{q1}, and C_{q2} such that

$$\begin{aligned} h_r(\tau_0) &< C_r\tau_0, \\ h_{q0}(\tau_0) &\le C_{q0}\tau_0, \\ h_{q1}(\eta_x, \eta_v) &\le C_{q1}\eta_{x0} + C_{q2}\eta_{v0}. \end{aligned} \tag{55}$$

C_r, C_{q0}, C_{q1}, and C_{q2} can be bounded, given the robot parameters, by bounding the norms of the tensors arising from $\partial\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}})/\partial\mathbf{p}$ and $\partial\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}})/\partial\dot{\mathbf{p}}$. This can be done loosely by inspection because all terms are bounded. However, simple expressions bounding the tensor norms have been calculated by [23], and derivations of C_r, C_{q0}, C_{q1} and C_{q2} can be found in Appendix A. Recalling (47) and (50), we therefore choose

$$\begin{aligned} \tau_0 &= \min\left(\frac{\alpha}{5C_r}, \frac{\alpha}{5C_{q0}}\right), \\ \eta_{x0} &= \frac{\alpha}{10C_{q1}}, \\ \eta_{v0} &= \frac{\alpha}{10C_{q2}}. \end{aligned} \tag{56}$$

Recalling the previous section, we can thus choose consistent parameters τ_0, η_{x0}, η_{v0}, μ, and κ_l that are all $\Theta(\varepsilon)$. Now, recall (34) and (35). If $\|\boldsymbol{\mu}\|_\infty \le \mu$ and $\bar{\mu} \le \mu$, then the functions $\mathscr{H}_{\tau_0}$ and $\bar{\mathscr{H}}_{\tau_0}$ respectively satisfy (as functions $\mathscr{Q}$, and with respect to $\mathscr{A}'$) the hypotheses of (Coupled Tracking) Lemmas 3.2 and 3.3. Since we can use the Safe Tracking Lemma to find sufficiently small η_x and η_v that are $\Theta(\varepsilon)$, we obtain a timestep τ that is $\Theta(\varepsilon)$ by applying the appropriate Coupled Tracking Lemma. The pruning bucket size is again given by (46).

Thus, we have the following lemma:

LEMMA 5.3. *Given a kinodynamic planning problem for an open-chain manipulator and given an approximation parameter ε such that $0 < \varepsilon < 1$, the Near-Extremal Algorithm and the True-Extremal Algorithm will search reachability graphs with $O(c^d(1/\varepsilon)^{6d-1})$ vertices and edges; c is a constant dependent on the kinodynamic parameters and polynomial in d. If an optimal (safe) solution Γ_{opt} exists, then each algorithm will find some graph trajectory that is ε-approximately optimal.*

5.4. *Safety Checking and Final Bounds*

5.4.1. Quadratic Paths and Polyhedral C-Space Obstacles. Consider any kinodynamic solution Γ_q found by the Near-Extremal Algorithm or by the True-Extremal Algorithm for a Cartesian robot. We observe that Γ_q will have a piecewise-constant acceleration $\ddot{\mathbf{p}}_q$. Hence, the solution trajectory Γ_q is piecewise algebraic: $\mathbf{p}_q$ is quadratic and $\dot{\mathbf{p}}_q$ is linear in time t. Therefore, when the C-space obstacles are polyhedral, we can check for safety violations exactly as in the L_∞ dynamics bounds case (Cartesian kinodynamic planning) in [3]. The C-space obstacles can be "grown" affinely with trajectory speed ($\|\mathbf{v}\| \in \mathbb{R}$) to obtain *expanded C-space obstacles* in $C \times \mathbb{R}$. Safety checking for a single $(\tilde{a}, \tau)$-bang or $(\boldsymbol{\mu}, \tau)$-bang can be accomplished by intersecting the (quadratic) time-parametrized trajectory with these surfaces. Hence, we "lift" collision detection to $C \times \mathbb{R}$ to do safety checking, with the $\mathbb{R}$ dimension encoding speed. For $d \leq 3$, this can be done in time $O(N)$. In higher dimensions the basic technique could be extended, but a good complexity bound would require a tight bound on the complexity of computing the Minkowski sum of convex d-polytopes and a d-cube. See [3] for a discussion.

5.4.2. Nonpolyhedral C-Space Obstacles. For kinematic chains, and in cases when the C-space obstacles are nonpolyhedral, safety is checked by affinely growing the workspace obstacles and checking for intersections along the trajectory. (Recall the discussion of δ_v*-safety* in Section 2.1.) We describe a C-space obstacle representation and a robot-obstacle collision detection method that can be generalized to a safety-checking method similar to that described in [1] and [3] and reviewed above. It would be convenient if we could apply an algebraic collision detection predicate such as that described by Canny [12], [24]. (We shall soon describe why we cannot.) In fact, our δ_v-safety predicate uses the *structure* of his predicate and the same logical evaluation method for each pair of polyhedra that could possibly collide.

The nonoverlap condition for two convex polyhedra is given by the nonoverlap predicate

$$\bigwedge_i \bigvee_j \bigwedge_k \bigvee_l (C_{ijkl}(\mathbf{x}) > 0) \tag{57}$$

described in [14], with the exact form of constraint functions $C_{ijkl}: C \to \mathbb{R}$ given in [12]. When the real-space obstacles (polyhedra) are grown affinely with speed, the overlap predicate has the same structure as (57). (See [3].) The signs of the $C_{ijkl}(\mathbf{x})$ correspond exactly to spatial relationships among the vertices, edges, and faces of the possibly overlapping polyhedra. The zeros of the C_{ijkl} are surfaces that contain faces of C-space obstacles, so we call them *C-spaced obstacle surface functions.* To detect collisions along a path $\mathbf{p}$: *time* $\to$ TC, we substitute $\mathbf{p}(t)$ for $\mathbf{x}$ in (57) and "merge" the sign intervals of the resulting functions in time [12]. While the exact form of the C_{ijkl} found in [12] for a robot polyhedron uses quaternions to represent orientation, *we can use other C-space representations and obstacle surface functions that yield the same sign invariant sets in C-space.* For an open kinematic chain, one set of natural C-space surface functions would be mixed trigonometric polynomials—the sums of products of C-space coordinates and their cosines and/or sines [13]. In [24] Canny represents these surfaces algebraically by using quarternion and half-angle substitutions.

Unfortunately, algebraic collision detection requires a C-space coordinate system in which the surfaces of the C-space obstacles are algebraic and in which the robot path is algebraic in time. In the joint coordinate system for an open kinematic chain, the position components p_i of $(\boldsymbol{\mu}, \tau)$-bangs are quadratic in time, and the velocity components v_i are linear. While it is possible to describe the C-space obstacles algebraically by using substitutions such as $u_j = \tan(p_j/2)$ for revolute joints j, for the trajectories generated by our algorithms there is no way (when the configuration space has more than one dimension) to parametrize the *path functions* that result from this substitution simultaneously algebraically. When each p_j is a quadratic polynomial in time t, we can choose either:

(a) A coordinate system in which the C-space obstacle surfaces are the zeros of algebraic functions, and some path-position components p_j are inverse trigonometric functions of time; or
(b) A coordinate system in which the C-space obstacle surfaces are the zeros of trigonometric polynomials, but each path-position component p_i is an algebraic function of time.

Thus, there is no coordinate system in which both the C-space obstacle surface functions and $(\boldsymbol{\mu}, \tau)$-bang robot paths are algebraic, and we cannot in general perform safety checking algebraically.

We can, however, approximately evaluate the collision predicated for $(\boldsymbol{\mu}, \tau)$-bangs by using approximating polynomials for each of the C-space obstacle surface functions. (The set of polynomials would have different sets of coefficients for each $(\boldsymbol{\mu}, \tau)$-bang.) The same can be done for a corresponding δ_v-safety predicate. The degree r of the polynomials we need depend on how accurately we wish to approximate the trigonometric polynomials. Intuitively, we expect that since each timestep is finite, and in practice would be very short, this polynomial approximation is reasonable. More precisely, the resulting error ε_r in checking safety can be bounded and made arbitrarily small by increasing the degree of the polynomial. Then either a conservative algorithm, which only finds solutions that are $(\delta'_v(\mathbf{v}) + \varepsilon_r)$-safe, or an "optimistic" algorithm that finds solutions that are $(\delta'_v(\mathbf{v}) - \varepsilon_r)$-safe could be implemented.

We now argue that for the purpose of deriving an asymptotic complexity we can fix the degree of the approximating polynomials to a constant. Simply put, if we truncate the Taylor expansions of expanded C-space obstacle surface functions to rth-order polynomials, ε_r will be $O(\varepsilon^r)$ because the timestep τ is $\Omega(\varepsilon)$. Thus, if we set $r = 2$, ε_r will be $O(\varepsilon^2)$, and thus the error of the safety-checking approximation will be smaller than $O(\varepsilon)$.

Since the structure of our δ_v-safety violation predicate is similar to that of [12], it contains $O(N)$ polynomials in t and requires $O(N \log N)$ time to evaluate, once the sign-intervals of the polynomials are known. If we restrict ourselves to polynomials of degree r_c, the total number of terms in the polynomials will be $O(r_c N)$, and forming the expressions for expanded C-space obstacle functions will take time $O(r_c dN)$. If finding the sign-intervals for a polynomial of degree r_c has cost $S(r_c)$, then checking the $(1-\varepsilon)\delta_v(c_0, c_1)$-safety of a single trajectory segment could be done in time.

$$O(Nr_c(\log N + d + \log r_c) + NS(r_c)).$$

By the truncated series approximation error argument above, we can choose a constant r_c as long as ε is guaranteed to be sufficiently small. Continuing this reasoning, we argue that safety checking takes time roughly $O(N(d + \log N))$, which is the time required under a model of computation in which finding the sign-intervals of a univariate trigonometric polynomial of constant size and degree takes unit time. This completes the discussion of safety checking for the Near-Extremal Algorithm. We now turn to the True-Extremal case.

For the general open chain manipulator, true $(\bar{\boldsymbol{\mu}}, \tau)$-bangs are the solutions to the set of ordinary differential equations

$$(58)\qquad \begin{pmatrix} \dot{\mathbf{p}}(t) \\ \dot{\mathbf{v}}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{v}(t) \\ \mathbf{M}^{-1}(\mathbf{p}(t))\{\mathbf{f} - [\mathbf{v}(t)^T\mathbf{C}(\mathbf{p}(t))\mathbf{v}(t)] - \mathbf{G}(\mathbf{p}(t))\} \end{pmatrix}.$$

Since the right-hand side of (58) is C^∞ and its time-derivative can be bounded globally, standard integration techniques (e.g., Runge–Kutta) can be used to approximate the image of a $(\bar{\mu}, \tau)$-bang to arbitrary precision for time $t \in [0, \tau]$, with computation time growing sublinearly with the accuracy [25]. Alternatively, we can approximate the trajectory segment with a set of polynomials of fixed order (e.g., 4th order Runge–Kutta) that approximate it in respective subintervals of the timestep; for a given timestep, a kth order integrator, and m subintervals, accuracy will be $O(\tau^k m^{1-k})$. To do safety checking of the True-Extremal Algorithm, then, we can again use approximating polynomials in our δ_v-safety predicate.

To summarize, under a combinatorial model that charges unit cost for finding the roots of a univariate trigonometric polynomial of fixed degree and size, safety checking for the Near-Extremal Algorithm can be done in time $O(N(d + \log N))$ for sufficiently small ε. In practice, numerical methods would be used to find sign-intervals for approximating polynomials in t. The error in these methods can be bounded and incorporated into the safety margin. Approximating polynomials can also be used to check the safety of general open chain manipulator $(\bar{\mu}, \tau)$-bangs. Thus, the algorithms have a time-complexity of $O(N(d + \log N))$ in the asymptotic case, i.e., for sufficiently small ε.

5.4.3. Asymptotic Bounds. For each of our algorithms, the final complexity is the complexity of the number of graph edges explored multiplied by the cost of checking the safety of each edge as a trajectory segment. By combining the graph-size bounds of Lemmas 5.1–5.3 with the safety-checking costs described above, we obtain a final approximate cost of $O(c^d N(d + \log N)(1/\varepsilon)^{6d-1})$, assuming sufficiently small $\varepsilon > 0$. Letting $p(N, \varepsilon, d)$ express the exact cost of safety checking for one bang yields Theorems 2.1 and 2.2; that is, the $O(c^d N(d + \log N)(1/\varepsilon)^{6d-1})$ bound is derived by arguing that $p(N, \varepsilon, d) = O(N(d + \log N))$.

6. Conclusions. In this paper we obtained provably good approximation algorithms for kinodynamic planning that extend the results of [1] to open kinematic chains and to Cartesian robots obeying L_2-norm dynamics bounds. These algorithms find trajectories that are approximately optimal with respect to a possibly speed-dependent safety margin. We presented algorithms that find near-extremal trajectories and algorithms that find truly extremal trajectories. Our True-Extremal Algorithm is the first such provably good algorithm that uses a state density condition to prune an otherwise exponential search to polynomial size. Our techniques yield lower complexity bounds than the earlier algorithms of [6]–[8].

To obtain our results we proved two crucial lemmas by considering simple adversary games. By using the first lemma (Lemma 3.2), given robot dynamics bounds, a safety margin, and ε we can find parameters that determine a state reachability graph such that, for every optimal trajectory whose start is approximated by the root of the graph, there is a graph trajectory that is within ε of being optimal. A second lemma (Lemma 3.3) allows us simultaneously to derive the state density condition for pruning (46).

Although our results directly apply to two classes of robots, they can be easily extended to a larger class. We conjecture this class is the class of robots that have finite degrees of freedom, bounded configuration, convex generalized force and velocity bounds, and acceleration maps that obey the following constraints:

(a) The dimension of the set of feasible accelerations is equal to the dimension of the configuration space.
(b) The set of feasible accelerations is convex at each state.
(c) State perturbations $(\Delta\mathbf{p}, \Delta\dot{\mathbf{p}})$ result in perturbations to the acceleration map that are $O(\|\Delta\mathbf{p}\| + \|\Delta\dot{\mathbf{p}}\|)$.

In addition, there are many directions for future research.

1. We conjecture that the proofs of the Coupled Tracking Lemma (Lemma 3.2) and the Robust Coupled Tracking Lemma (Lemma 3.3) can be adapted to show that we can track an adversary's trajectory as long as the *convex hull* of our allowable accelerations has a κ_l advantage over the adversary's allowable accelerations. This would imply a provably good polynomial-time approximation algorithm for kinodynamic planning using (approximately) bang-bang controls. See [4].
2. Since the tracking lemmas do not require the force bounds to be state-invariant, it should be possible to extend the results to relax this requirement.
3. Because of the use of κ_l acceleration advantages, for Lemmas 3.2 and 3.3 to be applicable (e.g., to obtain polynomial-time approximation algorithms for other classes of robots), it is necessary for the set of feasible instantaneous accelerations $\mathscr{A}(\mathbf{X})$ to have dimension d at every nonextremal state $\mathbf{X} \in TC$. For many robot systems with nonholonomic constraints, such as wheeled mobile robots, this is not so. A tracking lemma for such robots would allow us to extend the general [1] approach to them.
4. We have so far used a single parameter ε to characterize closeness to optimality. In a finer analysis, we would use parameter ε_T and ε_S to describe separately

closeness to optimality in execution time and in observance of the safety margin. We expect that such an analysis would lead to algorithms that allow tradeoff among time-optimality, safety, and running time. Furthermore, while a worst-case analysis is necessary when considering safety, an expected-case analysis would be appropriate for measuring time-optimality versus algorithmic complexity. See [26] and [4].

We have presented provably good approximation algorithms for optimal kinodynamic planning with the lowest known complexity for robots obeying coupled dynamic bounds. While optimal kinodynamic planning has an optimization flavor, our algorithms and proof techniques draw on several branches of computer science and robotics. There is a great deal of challenging theoretical and experimental work to be done, especially in the direction of practical approximation algorithms.

Appendix A. Computing Parameters for the True-Extremal and Near-Extremal Algorithms for Open Chain Manipulators. We wish to show, for any given problem such that the force bounds always exceed the forces necessary to hold the robot stationary, and for a given ε, that the parameters η_{x0}, η_{v0}, and τ_0 will be $\Omega(\varepsilon)$. We use $(\mathbf{p}, \dot{\mathbf{p}})$ to denote both trajectories and arbitrary states.

We first show that $\|\mathbf{M}^{-1}(\mathbf{p}) - \mathbf{M}^{-1}(\mathbf{p} - \Delta\mathbf{p})\|$ are $O(\|\Delta\mathbf{p}\|)$ or $O(d^2\|\Delta\mathbf{p}\|)$ for a d-DOF system. We can then compute bounds for the perturbation magnitudes h_r, h_{q0}, and h_{q1}, defined in (49), in terms of τ_0, η_x, and η_y. Given $\mathbf{M}(\mathbf{p})$ and $\mathbf{U}_G(\mathbf{p})$ (or $\mathbf{G}(\mathbf{p})$), it is possible to bound their derivatives by inspection, because each of their components is a sum of products of components of $\mathbf{p}$ and their sines and cosines. In particular, following any derivation of (1), say from [9], the close relationship to the kinematic map and Jacobian of the robot is noted. For example,

$$G_i(\mathbf{p}) = \sum_j m_j \mathbf{g}^T J_{Li}^{(j)}(\mathbf{p}),$$

where m_j is the mass of the jth link, $\mathbf{g}$ is the gravitational acceleration vector, and $J_{Li}^{(j)}(\mathbf{p})$ is the ith column of the linear velocity Jacobian for link j. More importantly,

$$\mathbf{M}(\mathbf{p}) = \sum_i (m_i \mathbf{J}_L^{(i)}(\mathbf{p})^T \mathbf{J}_L^{(i)}(\mathbf{p}) + \mathbf{J}_A^{(i)}(\mathbf{p})^T \mathbf{I}_i(\mathbf{p}) \mathbf{J}_A^{(i)}(\mathbf{p})),$$

where m_i is the mass of the ith link, $\mathbf{I}_i(\mathbf{p})$ is its inertia tensor, and $\mathbf{J}_L^{(i)}$ and $\mathbf{J}_A^{(i)}$ are its linear velocity and angular velocity Jacobians.

Heinzinger and Paden [23] exploit similar relationships to bound the general derivatives of $\mathbf{M}(\mathbf{p})$ and $\mathbf{G}(\mathbf{p})$. We use their results to show one way to derive the desired bounds.

A.1. Bounding Changes in $\mathbf{M}^{l-1}(\mathbf{p})$. Let $\mathbf{M}(\mathbf{p})$ be the $d \times d$ inertia (tensor) matrix for a d-DOF open kinematic chain. Observe that because all the derivatives of the components of $\mathbf{M}(\mathbf{p})$ are bounded, $\|\mathbf{M}(\mathbf{p}) - \mathbf{M}(\mathbf{p} + \Delta\mathbf{p})\|_2$ is $O(d^2\|\Delta\mathbf{My}\|_2)$.

Let $\mathbf{M}' = \mathbf{M}(\mathbf{p} + \Delta\mathbf{p})$ and $\Delta\mathbf{M} = \mathbf{M} - \mathbf{M}'$. We wish to bound $\|\Delta\mathbf{M}^{-1}\| = \|\mathbf{M}^{-1} - \mathbf{M}'^{-1}\|$. Since $\mathbf{M}$ and $\mathbf{M}'$ are inertia matrices, they are symmetric and positive definite. Let $\lambda_{\min}$ be the minimum of their minimum eigenvalues.

Now, consider the solutions $\mathbf{x}$ and $\mathbf{y}$ to the systems

$$\mathbf{M}\mathbf{x} = \mathbf{b}, \qquad \mathbf{M}'\mathbf{y} = \mathbf{b}. \tag{59}$$

By substitution,

$$\mathbf{M}(\mathbf{y} - \mathbf{x}) = (\mathbf{M} - \mathbf{M}')\mathbf{y} + \mathbf{M}'\mathbf{y} - \mathbf{M}\mathbf{x} = \Delta\mathbf{M}\mathbf{y},$$

and thus $\|\mathbf{M}(\mathbf{x} - \mathbf{y})\| = \|\Delta\mathbf{M}\mathbf{y}\|$. We now choose to use the L_2-norm. Now,

$$\|\Delta\mathbf{M}\mathbf{y}\|_2 \leq \|\Delta\mathbf{M}\|_2\|\mathbf{y}\|_2 \leq \|\Delta\mathbf{M}\|_2 \frac{\|\mathbf{b}\|_2}{\lambda_{\min}}.$$

Since

$$\lambda_{\min}\|\mathbf{x} - \mathbf{y}\|_2 \leq \|\mathbf{M}(\mathbf{x} - \mathbf{y})\|_2 = \|\Delta\mathbf{M}\mathbf{y}\|_2,$$

it follows that

$$\|\mathbf{x} - \mathbf{y}\|_2 \leq \frac{\|\Delta\mathbf{M}\|_2\|\mathbf{b}\|_2}{\lambda_{\min}^2}. \tag{60}$$

Recall that

$$\Delta\mathbf{M}^{-1}\mathbf{b} = (\mathbf{M}^{-1} - \mathbf{M}'^{-1})\mathbf{b} = \mathbf{x} - \mathbf{y}. \tag{61}$$

Since $\Delta\mathbf{M}$ is arbitrary except for the condition that $\mathbf{M}'$ be nonsingular, and $\mathbf{b}$ is arbitrary, (60) and (61) imply that

$$\|\Delta\mathbf{M}^{-1}\|_2 \leq \frac{\|\Delta\mathbf{M}\|_2}{\lambda_{\min}^2}. \tag{62}$$

Therefore, since $\|\Delta\mathbf{M}\|_2$ is $O(d^2\|\Delta\mathbf{p}\|)$, so is $\|\Delta\mathbf{M}^{-1}\|_2$.

A.2. Acceleration Bounds and Perturbations. We first review a notation for tensor-valued functions, as used by [23]. $\mathbf{M}$ is a smooth tensor field, and $\mathbf{M}(\mathbf{p})$ is a tensor of rank 2. For $\mathbf{x} \in C$, $\mathbf{M}(\mathbf{x})(\mathbf{v}_1, \mathbf{v}_2)$ denotes the tensor acting on $(\mathbf{v}_1, \mathbf{v}_2) \in T_\mathbf{x}C \times T_\mathbf{x}C$. We have been representing $\mathbf{M}(\mathbf{p})$ as a matrix. For example, the kinetic energy of the system in state $(\mathbf{p}, \dot{\mathbf{p}})$ can be expressed as $\frac{1}{2}\dot{\mathbf{p}}^T\mathbf{M}(\mathbf{p})\dot{\mathbf{p}}$ or $\frac{1}{2}\mathbf{M}(\mathbf{p})(\dot{\mathbf{p}}, \dot{\mathbf{p}})$.

The nth derivative of $\mathbf{M}(\mathbf{p})$ with respect to $\mathbf{p}$ is defined as follows:

$$D^n\mathbf{M}(\mathbf{p})(\alpha, \beta)(\gamma_1, \ldots, \gamma_n) = \sum_{i,j=1}^{d} \sum_{k_1,\ldots,k_n=1}^{d} \frac{\partial^n \mathbf{M}(\mathbf{p})_{ij}}{\partial \mathbf{p}_{k_1} \cdots \partial \mathbf{p}_{kn}} \alpha^i \beta^j \gamma_1^{k_1} \cdots \gamma_n^{k_n}. \tag{63}$$

Hence, for a given state $(\mathbf{p}, \dot{\mathbf{p}})$ and acceleration $\mathbf{a}$, (1) corresponds to

$$\mathbf{f}(\cdot) = \mathbf{M}(\mathbf{p})(\mathbf{a}, \cdot) + D\mathbf{M}(\mathbf{p})(\dot{\mathbf{p}}, \cdot)(\dot{\mathbf{p}}) - \tfrac{1}{2}D\mathbf{M}(\dot{\mathbf{p}}, \dot{\mathbf{p}})(\cdot) + G(\mathbf{p})(\cdot), \tag{64}$$

with $f_i = \mathbf{f}(\mathbf{e}_i)$, where $\mathbf{e}_i$ is the usual ith basis vector in $\mathbb{R}^d$. The second and third terms on the right-hand side are equivalent to the $[\dot{\mathbf{p}}^T\mathbf{C}(\dot{\mathbf{p}})]$ term in (1).

We use four inequalities ((65)–(68)) that are results from [23]. Let M_i denote the mass of link i, L_i its maximum length from the near joint axis, and $\bar{L}_i$ the greatest distance of its centroid from the first joint axis of the manipulator. Define $\|\mathbf{y}\|_{\hat{\imath}} = \sum_{j=1}^{i} |y_i|$. Then

$$|\mathbf{M}(\mathbf{p})(\mathbf{w}_1, \mathbf{w}_2)| \le \sum_{i=1}^{d} m_i L_i^2 \|\mathbf{w}_1\|_{\hat{\imath}}\, \|\mathbf{w}_2\|_{\hat{\imath}}, \tag{65}$$

$$|D^n\mathbf{M}(\mathbf{p})(\mathbf{w}_1, \mathbf{w}_2)(\mathbf{w}_3, \dots, \mathbf{w}_{n+2})| \le 2^d \sum_{i=1}^{d} m_i L_i^2 \|\mathbf{w}_1\|_{\hat{\imath}} \dots \|\mathbf{w}_{n+2}\|_{\hat{\imath}}, \tag{66}$$

$$|\mathbf{G}(\mathbf{p})(\mathbf{w}_1)| \le \sum_{i=1}^{d} g M_i \bar{L}_i \|\mathbf{w}_1\|_{\hat{\imath}}, \tag{67}$$

$$|D^n\mathbf{G}(\mathbf{p})(\mathbf{w}_1, \dots, \mathbf{w}_{n+1})| \le \sum_{i=1}^{d} g M_i \bar{L}_i \|\mathbf{w}_1\|_{\hat{\imath}} \dots \|\mathbf{w}_{n+1}\|_{\hat{\imath}}. \tag{68}$$

Now, we bound the acceleration. Let us define

$$\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) = \mathbf{f} - [\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}] - \mathbf{G}(\mathbf{p}). \tag{69}$$

Then we can rewrite (10) as

$$\mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f}) = \mathbf{M}^{-1}(\mathbf{p})\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}). \tag{70}$$

Recalling (10),

$$\begin{aligned} \|\mathbf{a}(\mathbf{p}, \dot{\mathbf{p}}, \mathbf{f})\| &\le \|\mathbf{M}^{-1}(\mathbf{p})\|\ \|\mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}})\| \\ &\le \frac{1}{\lambda^M_{\min}} \|\mathbf{f} - [\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}] - \mathbf{G}(\mathbf{p})\| \\ &\le \frac{1}{\lambda^M_{\min}} \{\|\mathbf{f}\| + \|[\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}]\| + \mathbf{G}(\mathbf{p})\|\}. \end{aligned} \tag{71}$$

Using (66) and recalling that $\|\mathbf{x}\|_2 \le \|\mathbf{x}\|_1 \le \sqrt{d}\|\mathbf{x}\|_2$ for any d-vector $\mathbf{x}$, we obtain

$$\begin{aligned} \|[\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}]\| &\le \max_{\|\mathbf{x}\|_2 \le 1} \{|D\mathbf{M}(\mathbf{p})(\dot{\mathbf{p}}, \mathbf{x})(\dot{\mathbf{p}})| + |\tfrac{1}{2}D\mathbf{M}(\mathbf{p})(\dot{\mathbf{p}}, \dot{\mathbf{p}})(\mathbf{x})|\} \\ &\le 2^{d-1}3 \sum_{i=1}^{d} \sqrt{d} M_i L_i^2 \|\mathbf{v}_{\max}\|_{\hat{\imath}}^2 \\ &\le 2^{d-1}3d^{3/2}\|\mathbf{v}_{\max}\|^2 \sum_{i=1}^{d} M_i L_i^2. \end{aligned} \tag{72}$$

Using (67)

$$\|\mathbf{G}(\mathbf{p})\| \le \max_{\|\mathbf{x}\|_2 \le 1} \sum_{i=1}^{d} g M_i \bar{L}_i \|\mathbf{x}\|_i \tag{73}$$

$$\le g\sqrt{d} \sum_{i=1}^{d} M_i \bar{L}_i.$$

Thus we have the following global bound for the acceleration;

$$A_{\max} \le \frac{1}{\lambda_{\min}^{M}} \left\{ \|\mathbf{f}_{\max}\| + 3d^{3/2} \|\mathbf{v}_{\max}\|^2 \sum_{i=1}^{d} M_i L_i^2 + g\sqrt{d} \sum_{i=1}^{d} M_i \bar{L}_i \right\}. \tag{74}$$

We now bound $h_r(\tau_0)$, $h_{q0}(\tau_0)$, and $h_{q1}(\eta_x, \eta_v)$. Recalling definitions (49) we obtain

$$h_r(\tau_0) < \tau_0 \max \left\| \left[\frac{\partial \mathbf{M}^{-1}}{\partial \mathbf{p}} \dot{\mathbf{p}} \right] \mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) + \mathbf{M}^{-1}(\mathbf{p}) \left[\frac{\partial \mathscr{F}}{\partial \mathbf{p}} \dot{\mathbf{p}} + \frac{\partial \mathscr{F}}{\partial \dot{\mathbf{p}}} \mathbf{a} \right] \right\|,$$

$$h_{q0}(\tau_0) \le \tau_0 \max \left\| \left[\frac{\partial \mathbf{M}^{-1}}{\partial \mathbf{p}} \dot{\mathbf{p}} \right] \mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) + \mathbf{M}^{-1}(\mathbf{p}) \left[\frac{\partial \mathscr{F}}{\partial \mathbf{p}} \dot{\mathbf{p}} + \frac{\partial \mathscr{F}}{\partial \dot{\mathbf{p}}} \mathbf{a} \right] \right\|, \tag{54}$$

$$h_{q1}(\eta_x, \eta_v)) \le \max \left\| \left[\frac{\partial \mathbf{M}^{-1}}{\partial \mathbf{p}} \Delta\mathbf{p} \right] \mathscr{F}(\mathbf{f}, \mathbf{p}, \dot{\mathbf{p}}) + \mathbf{M}^{-1}(\mathbf{p}) \left[\frac{\partial \mathscr{F}}{\partial \mathbf{p}} \Delta\mathbf{p} + \frac{\partial \mathscr{F}}{\partial \dot{\mathbf{p}}} \Delta\dot{\mathbf{p}} \right] \right\|.$$

Recall (62), which we use to bound $\|\mathbf{M}^{-1}(\mathbf{p}) - \mathbf{M}^{-1}(\mathbf{p} + \Delta\mathbf{p})\|$, and observe that

$$\|\mathbf{M}(\mathbf{p}) - \mathbf{M}(\mathbf{p} + \Delta\mathbf{p})\| \le \max_{\|x\| = \|y\| = 1} |D\mathbf{M}(\mathbf{p})(\mathbf{x}, \mathbf{y})(\Delta\mathbf{p})| \tag{75}$$

$$\le 2^d d^{3/2} \|\Delta\mathbf{p}\| \sum_{i=1}^{d} M_i L_i^2.$$

By substitution, we obtain

$$\|\Delta\mathbf{M}^{-1}\|_2 \le \frac{2^d d^{3/2} \|\mathbf{p}\| \sum_{i=1}^{d} M_i L_i^2}{(\lambda_{\min}^{M})^2} \tag{76}$$

It follows that

$$\tau_0 \left\| \frac{\partial \mathbf{M}^{-1}}{\partial \mathbf{p}} \dot{\mathbf{p}} \right\| \le \frac{2^d d^{3/2} \|\dot{\mathbf{p}} \tau_0\| \sum_{i=1}^{d} M_i L_i^2}{(\lambda_{\min}^{M})^2} \tag{77}$$

and

$$\left\| \frac{\partial \mathbf{M}^{-1}}{\partial \mathbf{p}} \Delta\mathbf{p} \right\| \le \frac{2^d d^{3/2} \|\Delta\mathbf{p}\| \sum_{i=1}^{d} M_i L_i^2}{(\lambda_{\min}^{M})^2}. \tag{78}$$

Applying (66) and (72), we obtain

$$(79)\quad \left\|\frac{\partial[\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}]}{\partial\mathbf{p}}\Delta\mathbf{p}\right\| \le \max_{\|\mathbf{x}\|\le 1}\{|D^2\mathbf{M}(\mathbf{p})(\dot{\mathbf{p}},\mathbf{x})(\dot{\mathbf{p}},\Delta\mathbf{p})| + |\tfrac{1}{2}D^2\mathbf{M}(\mathbf{p})(\dot{\mathbf{p}},\dot{\mathbf{p}})(\mathbf{x},\Delta\mathbf{p})|\}$$

$$\le 2^{d-1}\cdot 3d^2\|\dot{\mathbf{p}}\|^2\|\Delta\mathbf{p}\|\sum_{i=1}^{d} M_iL_i^2,$$

$$(80)\quad \left\|\frac{\partial[\dot{\mathbf{p}}^T\mathbf{C}(\mathbf{p})\dot{\mathbf{p}}]}{\partial\dot{\mathbf{p}}}\Delta\dot{\mathbf{p}}\right\|$$

$$\le \max_{\|\mathbf{x}\|\le 1}\{|D\mathbf{M}(\mathbf{p})(\dot{\mathbf{p}},\mathbf{x})(\Delta\dot{\mathbf{p}})| + |D\mathbf{M}(\mathbf{p})(\Delta\dot{\mathbf{p}},\mathbf{x})(\dot{\mathbf{p}})| + |D\mathbf{M}(\mathbf{p})(\Delta\dot{\mathbf{p}},\dot{\mathbf{p}})(\mathbf{x})|\}$$

$$\le 2^{d-1}\cdot 3d^{3/2}\|\dot{\mathbf{p}}\|\ \|\Delta\dot{\mathbf{p}}\|\sum_{i=1}^{d} M_iL_i^2,$$

and

$$(81)\quad \left\|\frac{\partial\mathbf{G}(\mathbf{p})}{\partial\mathbf{p}}\Delta\mathbf{p}\right\| \le g\sqrt{d}\sum_{i=1}^{d} M_i\bar{L}_i.$$

We now give the final bounds for $h_r(\tau_0)$, $h_{q0}(\tau_0)$, and $h_{q1}(\eta_{x0}, \eta_{v0})$. First, we define

$$(82)\quad \begin{aligned} H &= \sum_{i=1}^{d} M_iL_i^2,\\ \bar{H} &= \sum_{i=1}^{d} gM_i\bar{L}_i,\\ F &= \|\mathbf{f}\| + 2^{d-1}\cdot 3d^{3/2}\|\mathbf{v}_{\max}\|^2H + \sqrt{d}g\bar{H},\\ F' &= \|\mathbf{f}'\| + 2^{d-1}\cdot 3d^{3/2}\|\mathbf{v}'_{\max}\|^2H + \sqrt{d}g\bar{H}, \end{aligned}$$

In this notation, we can write (74) as

$$(83)\quad A_{\max} \le \frac{F}{\lambda^M_{\min}}.$$

Finally,

$$(84)\quad h_r(\tau_0) < \frac{(2^dd^{3/2}\|\mathbf{v}'_{\max}\|\bar{H}F' + 2^{d-1}\cdot 3d^2\|\mathbf{v}'_{\max}\|^3)\tau_0}{\lambda^M_{\min}} + \frac{2^{d-1}\cdot 3d^{3/2}\|\mathbf{v}'_{\max}\|F'\tau_0}{(\lambda^M_{\min})^2},$$

$$h_{q0}(\tau_0) \le \frac{(2^d d^{3/2} \|\mathbf{v}_{\max} \tau_0\| HF + 2^{d-1} \cdot 3d^2 \|\mathbf{v}_{\max}\|^3)\tau_{0+}}{\lambda^M_{\min}} + \frac{2^{d-1} \cdot 3d^{3/2} \|\mathbf{v}_{\max} \tau_0\| F\tau_0}{(\lambda^M_{\min})^2}, \tag{85}$$

$$h_{q1}(\eta_{x0}, \eta_{v0}) \le \left\{\frac{2^d d^{3/2} HF}{(\lambda^M_{\min})^2} + \frac{2^{d-1} \cdot 3d^2 \|\mathbf{v}_{\max}\| H\sqrt{dg\bar{H}}}{\lambda^M_{\min}}\right\}\eta_{x0} + \frac{2^{d-1} \cdot 3d^{3/2} \|\mathbf{v}_{\max}\| H}{\lambda^M_{\min}} \eta_{v0}. \tag{86}$$

We summarize these bounds by saying that $h_r(\tau_0)$ and $h_{q0}(\tau_0)$ are linear in τ_0, and that $h_{q1}(\eta_{x0}, \eta_{v0})$ is linear in η_{x0} and η_{v0}. In other words, we can rewrite (84)–(86) with obvious substitutions for C_r, C_{q0}, C_{q1}, and C_{q2}, which depend on d, the dynamics equation, and the dynamics bounds:

$$\begin{aligned} h_r(\tau_0) &< C_r \tau_0, \\ h_{q0}(\tau_0) &\le C_{q0}\tau_0, \\ h_{q1}(\eta_{x0}, \eta_{v0}) &\le C_{q1}\eta_{x0} + C_{q2}\eta_{v0}. \end{aligned} \tag{55}$$

Appendix B. Guide to Notation.

C is the configuration space of a particular robot.
TC is the tangent bundle of C.
d is the number of dimensions of the configuration space C.
$\mathcal{O}$ is the encoding of the obstacle set.
δ_v is a safety function parametrized by $c_0 > 0$ and $c_1 \ge 0$; $\delta_v(c_0, c_1)(\mathbf{v}) = c_0 + c_1\|\mathbf{v}\|$.
$\mathbf{p}(t)$ is a path; $\mathbf{p}$: *Time* $\to C$.
$(\mathbf{p}, \dot{\mathbf{p}})$ is a trajectory state; $\mathbf{p}$ is position and $\dot{\mathbf{p}}$ is velocity.
$\mathbf{x}$ is a configuration (also $\mathbf{y}$).
$\mathbf{v}$ is a velocity.
$\mathbf{X}$ is a state (also $\mathbf{Y}$).
Γ is a trajectory; Γ: *Time* $\to TC$.
$\mathbf{S}$ is the start state, usually the desired start state.
$\mathbf{G}$ is the goal state, usually the desired goal state.
(η_x, η_v) is a tracking tolerance.
$\mathbf{a}$ is acceleration.
$\bar{a}$ is an acceleration bound.
$\bar{\mathbf{a}}$ is a vector of acceleration bounds.
$\bar{v}$ is a velocity bound.
$\bar{\mathbf{v}}$ is a vector of velocity bounds.
$A_{\max}$ is a scalar, global (nondimensional) acceleration bound.
$\mathbf{f}$ is a generalized force vector.

$\bar{\mathbf{f}}$ is a vector of generalized force bounds; for each i, $|f_i| \le \bar{f}_i$.
$\mathscr{M}$ is the encoding of the dynamics equation obeyed by a robot.
l is the world diameter or length of the greatest translational degree of freedom.
ε is an approximation parameter; $0 < \varepsilon < 1$.
δ'_v is a safety function for ε-safety; $\delta'_v = (1 - \varepsilon)\delta_v$.
Γ'_r is the trajectory Γ_r, but ε-time rescaled.
$\mathbf{p}'$ is the path (or state) $\mathbf{p}$, but ε-time rescaled.
$\mathbf{S}'$ is the time-rescaled start state.
$\mathbf{G}'$ is the ε-time-rescaled goal state.
$\mathbf{S}^*$ is an ε-close approximation of the start date.
$\mathbf{G}^*$ is an ε-close approximation of the goal state.
$\mathscr{A}_{\mathbf{X}}$ is the set of all instantaneous accelerations possible at state $\mathbf{X}$ obeying dynamics bounds.
$\hat{\mathscr{A}}_{\mathbf{X}}$ is the set of possible constant accelerations of duration τ for trajectories beginning at state $\mathbf{X}$; these trajectories must obey the dynamics bounds.
$\hat{\mathscr{A}}^*_{\mathbf{X}}$ is an easier-to-compute conservative approximation (subset) of $\hat{\mathscr{A}}_{\mathbf{X}}$.
$\mathscr{A}$ is first, a set of acceleration functions, later generalized to a function $\mathscr{A}$: $TC \rightarrow \{\mathscr{Y} : \mathscr{Y}$ is a set of acceleration functions$\}$.
$\mathscr{A}'$ is as above, but corresponding to tighter generalized force bounds.
$\mathscr{P}$, $\mathscr{Q}$ are, with single arguments, arbitrary functions $\mathscr{P}$, $\mathscr{Q}$: $TC \rightarrow \{\mathscr{Y} : \mathscr{Y}$ is a set of acceleration functions$\}$.
τ is a timestep size.
κ_l is a minimal acceleration advantage.
μ is a grid-spacing.
$\boldsymbol{\mu}$ is a vector of grid-spacings; μ_i is the spacing in dimension i.
$\bar{\mu}$ is a discretization (grid-spacing) parameter for generalized forces.
$\mathscr{H}(\mathbf{X}, \tau)$ is the $(\boldsymbol{\mu}, \tau)$-extremal shell at (corresponding to) state $\mathbf{X}$; this is a set of acceleration functions.
$\bar{\mathscr{H}}(\mathbf{X}, \tau)$ is the set of true $(\bar{\mu}, \tau)$-bangs at (corresponding to) state $\mathbf{X}$; another set of acceleration functions.
$\mathscr{H}_\tau(\mathbf{X}) = \mathscr{H}(\mathbf{X}, \tau)$.
$\bar{\mathscr{H}}_\tau(\mathbf{X}) = \bar{\mathscr{H}}(\mathbf{X}, \tau)$.
τ_0, for a given robot and ε, is the fundamental timestep.
(η_{x0}, η_{v0}), for a given robot and ε, is a fundamental tracking tolerance.
$\oplus$ is the Minkowski sum.
Subscripts r, q, and u denote trajectories.
Subscript i usually denotes the ith coordinate axial direction.
Subscript n usually denotes a timestep.

References

[1] J. Canny, B. Donald, J. Reif, and P. Xavier. On the complexity of kinodynamic planning. *Proceedings of the 29th Annual Symposium on the Foundations of Computer Science*, White Plains, New York, 1988, pp. 306–316.

[2] B. Donald, P. Xavier, J. Canny, and J. Reif. Kinodynamic motion planning. *Journal of the ACM*, **40**(5), 1993, 1048–1066. Journal version of [1].

[3] B. Donald and P. Xavier. Provably good approximation algorithms for optimal kinodynamic planning: robots with decoupled dynamics bounds. *Algorithmica*, this issue, pp. 443–479.

[4] P. Xavier. Provably good approximation algorithms for optimal kinodynamic robot motion plans. Technical Report CUCS-TR92-1279, Computer Science Department, Cornell University, Ithaca, New York, April 1992. Ph.D. thesis.

[5] B. Donald and P. Xavier. A provably good approximation algorithm for optimal-time trajectory planning. *Proceedings of the* 1989 *IEEE International Conference on Robotics and Automation*, Scottsdale, Arizona, 1989, pp. 958—963.

[6] P. Jacobs, G. Heinzinger, J. Canny, and B. Paden. Planning guaranteed near-time-optimal planning in a cluttered workspace. Technical Report ESRC 89-20/RAMP 89-15, Engineering Systems Research Center, University of California, Berkeley, California, October 1989.

[7] P. Jacobs, G. Heinzinger, J. Canny, and B. Paden. Planning guaranteed near-time-optimal planning in a cluttered workspace. *Proceedings of the International Workshop on Sensorial Integration for Industrial Robots: Architectures & Applications*, Zaragoza, Spain, 1989.

[8] G. Heinzinger, P. Jacobs, J. Canny, and B. Paden. Time-optimal trajectories for a robot manipulator: a provably good approximation algorithm. *Proceedings of the* 1990 *IEEE International Conference on Robotics and Automation*, Cincinnati, Ohio, May 1990, pp. 150–155.

[9] H. Asada and J. J. Slotine. *Robot Analysis and Control.* Wiley, New York, 1986.

[10] J. M. Hollerbach. Dynamic scaling of manipulator trajectories. A.I. Memo 700, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1983.

[11] T. Lozano-Pérez. Spatial planning: a configuration space approach. *IEEE Transactions on Computers*, **32**(2):108–120, 1983. Also A.I. Memo 605, Massachusetts Institute of Technology, Cambridge, Massachusetts, December 1982.

[12] J. Canny. Collision detection for moving polyhedra. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(2):200–209, 1986.

[13] B. Donald. A search algorithm for motion planning with six degrees of freedom. *Artificial Intelligence*, **31**(3): 295–353, 1987.

[14] J. Canny and B. Donald. Simplified Voronoi diagrams. *Discrete and Computational Geometry*, **3**(3):219–236, 1988.

[15] B. Donald and P. Xavier. Provably good approximation algorithms for optimal kinodynamic planning for cartesian robots and open chain manipulators. *Proceedings of the Sixth Annual Symposium on Computational Geometry*, Berkeley, California, June 1990, pp. 290–300.

[16] B. Donald and P. Xavier. Near-optimal kinodynamic planning for robots with coupled dynamics bounds. In A. C. Sanderson, A. A. Derochers, and K. Valvanis, editors, *Proceedings of the Fourth IEEE International Symposium on Intelligent Control*, Albany, New York, 1989, pp. 354–359.

[17] B. Donald and P. Xavier. Provably good approximation algorithms for optimal kinodynamic planning for cartesian robots and open chain manipulators. Technical Report TR-1095, Department of Computer Science, Cornell University, Ithaca, New York, February 1990. Supersedes TR-971.

[18] J. Reif and S. Tate. Approximate kinodynamic planning using l_2-norm dynamics bounds. Technical Report CS-1990-13, Department of Computer Science, Duke University, Durham, North Carolina, 1990.

[19] Leitman. *An Introduction to Optimal Control.* McGraw-Hill, New York, 1966.

[20] H. M. Schaettler. On the optimality of bang-bang trajectories in $\mathbb{R}^3$. *Bulletin of the American Mathematical Society*, **16**(1):113–116, 1987.

[21] E. Sontag and H. Sussmann. Remarks on the time-optimal control of two-link manipulators. *Proceedings of the 24th Conference on Decision and Control*, Ft. Lauderdale, Florida, 1985, pp. 1646–1652.

[22] E. Sontag and H. Sussmann. Time-optimal control of manipulators. Technical Report, Department of Mathematics, Rutgers University, New Brunswick, New Jersey, 1986.

[23] G. Heinzinger and B. Paden. Bounds on robot dynamics. *Proceedings* 1989 *IEEE International Conference on Robotics and Automation.* Scottsdale, Arizona, 1989, pp. 1227–1232.

[24] J. Canny. *The Complexity of Robot Motion Planning.* MIT Press, Cambridge, Massachusetts, 1988. Book version of Canny's 1986 Ph.D. thesis.

[25] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C.* Cambridge University Press, New York, 1988.

[26] B. Donald and P. Xavier. Time-safety trade-offs and a bang-bang algorithm for kinodynamic planning. *Proceedings of the* 1991 *IEEE International Conference on Robotics and Automation*, Sacramento, California, 1991, pp. 552–557.