Provably Good Approximation Algorithms for Optimal Kinodynamic Planning: Robots with Decoupled Dynamics Bounds[1]

Bruce R. Donald
Department of Computer Science, Cornell University, Ithaca, NY 14853-7501
Patrick Xavier
Sandia National Laboratories, Albuquerque NM 87185-0951

Keywords: robot motion planning, optimal control, polynomial-time ϵ-approximation algorithm, time-optimal trajectory, shortest path, kinodynamics, polyhedral obstacles

Abstract: We consider the following problem: given a robot system, 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. In [1] we developed a provably good approximation algorithm for the minimum-time trajectory problem for a robot system with decoupled dynamics bounds (e.g., a point robot in $\mathbb{R}^3$). This algorithm differs from previous work in three ways. It is possible (1) to bound the goodness of the approximation by an error term ϵ; (2) to polynomially bound the computational complexity of our algorithm; and (3) to express the complexity as a polynomial function of the error term. Hence, given the geometric obstacles, dynamics bounds, and the error term ϵ, the algorithm returns a solution that is ϵ-close to optimal and requires only a polynomial (in $(\frac{1}{\epsilon})$) amount of time.

We extend the results of [1] in two ways. First, we modify it to halve the exponent in the polynomial bounds from $6d$ to $3d$, so that that the new algorithm is $O\left(c^d N \left(\frac{1}{\epsilon}\right)^{3d}\right)$, where N is the geometric complexity of the obstacles and c is a robot-dependent constant. Second, the new algorithm finds a trajectory that matches the optimal in time with an ϵ factor sacrificed in the obstacle-avoidance safety margin. Similar results hold for polyhedral Cartesian manipulators in polyhedral environments.

The new results indicate that an implementation of the algorithm could be reasonable, and a preliminary implementation has been done for the planar case.

[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 grants No. IRI-8802390, IRI-9000532 and by a Presidential Young Investigator award, and in part by the Mathematical Sciences Institute, Intel Corporation, and AT&T Bell laboratories.

1 Introduction

The *kinodynamic planning problem* is to synthesize a robot motion subject to simultaneous kinematic constraints, such as avoiding obstacles, and dynamics constraints, such as modulus bounds on velocity, acceleration, and force. A kinodynamic solution is a trajectory specification: a start state and a mapping from time to generalized forces or accelerations. The resulting motion is governed by a dynamics equation. In robotics, a long-standing open problem has been to synthesize *time-optimal* kinodynamic solutions, by which we mean solutions that require minimal time with respect to the kinodynamic constraints.

While there has been much work on this problem in the robotics community, there have been no exact algorithms except in the one-dimensional case.[2] In three dimensions, finding exact solutions is known to be NP-hard [3]; this straightforward extension of a result from [4] is also described in Appendix A. Therefore, it is reasonable to pursue *approximation algorithms* — algorithms that compute kinodynamic solutions that are "close" to optimal. However, for the many proposed approximate or heuristic techniques previous to [1, 5],[3] there exist no bounds on the goodness of the resulting solutions, or on the time-complexity of the algorithms.

The primary measure of optimality is time. Because of uncertainty in control and error in models, we believe that a *planned* robot motion can only be considered safe if it avoids obstacles by an appropriate margin. Thus, it is also natural to incorporate a safety measure into the meaning of "optimal". The problem formulation we introduced in [1] therefore includes a speed-dependent obstacle-avoidance margin in the problem parameters, along with start and goal states, dynamics bounds, and a set of obstacles. This margin of error is specified by a *safety function.* An optimal trajectory is thus a minimum-time trajectory that respects this safety criterion.

An approximation version of the problem allows an algorithm for kinodynamic planning to trade off running time against optimality in terms of: (a) execution time of the trajectory, (b) strictness in observing the safety margin, and (c) closeness to the desired start and goal states. (We note that (c) is implied in [1] but clarified in [6].) To analytically express this trade-off we parameterize closeness to an optimal solution with a tolerance ϵ and bound algorithm running time in terms of this ϵ. For example, if there exists a "safe" optimal-time kinodynamic solution requiring time T_{opt}, then the algorithm must find a "nearly-as-safe" solution that requires time at most $(1+\epsilon)T_{opt}$.

[1] described the first provably-good polynomial-time approximation algorithm for 2- and 3-dimensional optimal kinodynamic planning, which they restricted to particle dynamics. Here, we modify and reanalyze the algorithm to improve both its its complexity bound and its accuracy. The new algorithm is has time complexity $O\left(c^d N \left(\frac{1}{\epsilon}\right)^{3d}\right)$, where N is the geometric complexity of the environment and c depends on the dynamics and safety margin parameters; this halves the previous exponent of the $(\frac{1}{\epsilon})$ term. Furthermore,

[2] [2] have recently provided an exact algorithm for the 2D L_∞ case. The algorithm runs in exponential time and polynomial space.

[3] [5] is the journal revision of [1].

we show that if there exists an optimal kinodynamic solution requiring time T_{opt}, then the new algorithm will find an approximately optimal solution that requires time T_{opt}, whereas [1] only show a bound of $(1+\epsilon)T_{opt}$.

These new results indicate that our theoretical algorithm might be reasonable for offline motion planning, and we have performed simple experiments with a preliminary implementation of this algorithm, as reported in [3]. Our companion paper [7] extends our approach to robots with coupled dynamics and coupled dynamics bounds, including open-chain manipulators.

2 Kinodynamic Motion Planning

2.1 The Kinodynamic Planning Problem

Kinematic constraints, such as joint limits and obstacles, limit the configuration (position) of a robot. *Dynamics constraints* govern the time-derivatives of configuration and are independent of obstacles. They include dynamics laws and bounds on velocity, acceleration, and applied force. *Strictly kinodynamic* constraints are obstacle-dependent constraints that govern configuration *and* its time-derivatives but do not fall into either of the previous categories. An example of such a constraint is a speed-dependent obstacle-avoidance margin. A constraint is a *kinodynamic constraint* if it belongs to any of the above categories. The *state* of a robot at a given time is its configuration and velocity. The *general kinodynamic planning problem* is, for a given robot, to find a motion that goes from a start state to a goal state while obeying kinodynamic constraints.

We consider the following Cartesian problem from [1]. (See Figure 1.) A point mass in $\mathbb{R}^d$, where $d \in \{2,3\}$, must be moved from a start state $\mathbf{S} = (\mathbf{s}, \dot{\mathbf{s}})$ to a goal state $\mathbf{G} = (\mathbf{g}, \dot{\mathbf{g}})$. In the course of the motion, the point must avoid a set of polyhedral obstacles. Movement is controlled by applying forces or commanding accelerations, which are equivalent for a point mass. By using a configuration space approach, this problem is readily extended to cover polyhedral robots obeying decoupled dynamics and decoupled dynamics bounds.

We will denote the configuration space $\mathbb{R}^d$ by C, and its phase space by TC. Phase space TC is the robot state space and is isomorphic to $\mathbb{R}^{2d}$. Thus, a point in TC is a *(position, velocity)* pair such as $\mathbf{S}$ or $\mathbf{G}$.

A robot motion over a time interval $[0, T_f]$ can be specified by a twice-differentiable map $\mathbf{p} : [0, T_f] \to C$. This map is the *path* of the motion. In kinodynamic planning, the motion must obey dynamics and dynamics constraints, and it is convenient to specify $\dot{\mathbf{p}}$ explicitly. The *trajectory* of a robot motion is the map $\Gamma : [0, T_f] \to TC$ given by $\Gamma(t) = (\mathbf{p}(t), \dot{\mathbf{p}}(t))$. We denote the position and velocity components of a subscripted trajectory Γ_r by $\mathbf{p}_r$ and $\dot{\mathbf{p}}_r$, respectively. While a motion $\mathbf{p}$ can be given directly as a twice-differentiable function of time, two equivalent specifications are useful: (a) an initial position $\mathbf{p}_0$ and a velocity function $\mathbf{v} = \dot{\mathbf{p}}$, and (b) an initial state $(\mathbf{p}_0, \mathbf{v}_0)$ and an acceleration function $\mathbf{a} = \ddot{\mathbf{p}}$.

The motion must respect upper bounds on the magnitudes of the acceleration and velocity. At all times t the acceleration $\ddot{\mathbf{p}}(t)$ and the velocity $\dot{\mathbf{p}}(t)$ must obey

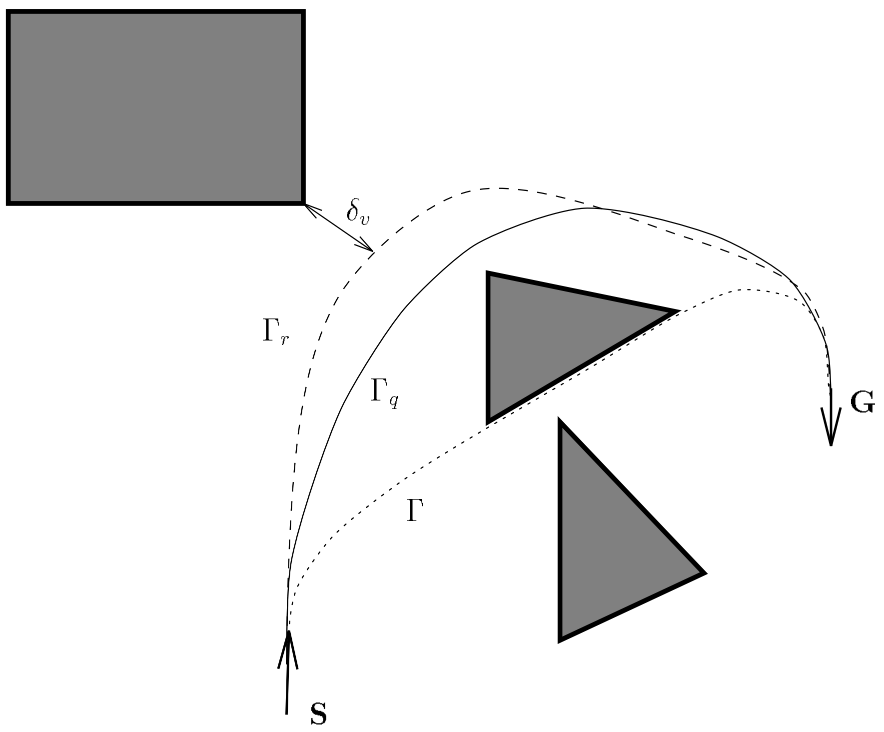


Figure 1: A kinodynamic planning problem for a point robot, showing the obstacles, the start $\mathbf{S} = (\mathbf{s}, \dot{\mathbf{s}})$, and the goal $\mathbf{G} = (\mathbf{g}, \dot{\mathbf{g}})$. Γ is a time-optimal solution (trajectory), with no safety-margin. Γ_r is an optimal kinodynamic solution, and Γ_q is an ϵ-approximately optimal kinodynamic solution.

$$\begin{aligned} \|\dot{\mathbf{p}}(t)\|_\infty &\le v_{max}, \qquad \text{and} & (1)\\ \|\ddot{\mathbf{p}}(t)\|_\infty &\le a_{max}. & (2) \end{aligned}$$

Eqs. (1) and (2) are the *dynamics bounds*, and since the L_∞-norm is used, we call (1) and (2) *L_∞ dynamics bounds.*

We assume that the obstacles $\mathcal{O}$ are represented by a set of convex, possibly overlapping polyhedra. If these convex polyhedra have a total of N faces overall, we call N the *combinatorial complexity* of $\mathcal{O}$. *Free space* is the complement of these obstacles. Finally, we assume that the set of free configurations is bounded by a d-cube of side length l. Thus, a tuple $(\mathcal{O}, \mathbf{S}, \mathbf{G}, l, a_{max}, v_{max})$ is an instance of the *Cartesian kinodynamic planning problem.*

An *exact solution* to the kinodynamic planning problem is a trajectory Γ such that $\Gamma(0) = \mathbf{S}$, $\Gamma(T_f) = \mathbf{G}$, and Γ obeys the kinodynamic constraints. That is, the path $\mathbf{p}$ avoids all obstacles, the velocity $\dot{\mathbf{p}}$ respects (1), and $\ddot{\mathbf{p}}$ respects (2). The *time* for a solution Γ is simply T_f. The *time-optimal* kinodynamic planning problem is to find a minimal-time kinodynamic solution, which is represented as a suitable encoding of the start state $\Gamma(0)$ and the acceleration function $\mathbf{a}$.

2.2 Optimal and Approximately Optimal Kinodynamic Plans

Following a theoretically time-optimal solution closely enough to avoid obstacles may require unrealizable precision in control or sensing. The exact time-optimal solution may thus be unexecutable by a physical robot. For this reason, an optimal solution should observe a safety margin; the margin we define is speed-dependent. The safety margin ensures the existence of a "tube" or family of solutions "nearby" in time and in phase-space that "approximate" the optimal safe solution. The existence of such a "tube" of approximating solutions is essential for our approach. Safety margins are both practically motivated and mathematically necessary.

A *δ_v-safe kinodynamic solution* avoids all obstacles by a safety margin δ_v. In this paper, we define this safety margin to be an affine function of the trajectory speed. This first-order choice roughly corresponds to how accurately and quickly a robot senses its position and velocity, combined with how quickly it can correct for velocity errors. Two scalars $c_0 > 0$ and $c_1 \ge 0$ characterize the safety margin, which one can view as an obstacle-free tube centered about the path. Formally, a *δ_v-safe kinodynamic solution* has the property that for all times t in $[0, T_f]$, there exists a ball about $\mathbf{p}(t)$ in free space of radius

$$\delta_v(c_0, c_1)(\dot{\mathbf{p}}(t)) = c_0 + c_1\|\dot{\mathbf{p}}(t)\|. \tag{3}$$

We will omit the parameters c_0 and c_1 in the discussion when confusion will not arise. Note that δ_v-safety is is an example of a kinodynamic constraint that is neither a pure kinematic constraint nor a pure dynamics constraint. A *δ_v-safe kinodynamic planning problem*, then, is a tuple $(\mathcal{O}, \mathbf{S}, \mathbf{G}, a_{max}, v_{max}, l, c_0, c_1)$. We call a_{max}, v_{max}, l, c_0, and c_1 the *kinodynamic bounds.*

For fixed c_0 and c_1, consider the class of all δ_v-safe kinodynamic solutions. We define an *optimal δ_v-safe kinodynamic solution* to be a solution whose time is minimal in this class. We will henceforth employ the term *optimal kinodynamic solution* as including δ_v-safety as one of the kinodynamic constraints that optimal trajectories must obey.

Let us say that an approximating state $(\mathbf{x}', \dot{\mathbf{x}}')$ is "ϵ-close" to a reference state $(\mathbf{x}, \dot{\mathbf{x}})$ if

$$\begin{aligned} \|\mathbf{x} - \mathbf{x}'\| &= O(\epsilon), \qquad \text{and} && (4) \\ \|\dot{\mathbf{x}} - \dot{\mathbf{x}}'\| &= O(\epsilon). && (5) \end{aligned}$$

We now specify what it means for a kinodynamic solution Γ_q to be *ϵ-approximately optimal*, where a positive $\epsilon < 1$ parameterizes the closeness of the approximation. First of all, Γ_q must obey the safety margin

$$\delta_v'(c_0, c_1)(\dot{\mathbf{p}}_q) = (1 - \epsilon)\delta_v(c_0, c_1)(\dot{\mathbf{p}}_q). \tag{6}$$

Second, if an optimal safe trajectory takes time T_{opt}, then we require that for an ϵ-approximately optimal trajectory Γ_q

$$T_q \leq (1 + \epsilon)T_{opt}. \tag{7}$$

Thirdly, we require that $\Gamma_q(0)$ and $\Gamma_q(T_q)$ be ϵ-close to the desired start and goal states $\mathbf{S}$ and $\mathbf{G}$, respectively.

In order to obtain our result, we must assume four conditions: L_∞-norm acceleration and velocity bounds, a bounded world diameter, and a non-zero safety margin. Each of these can be plausibly motivated in physical terms. For example, any physical robot will have bounded velocity and acceleration. However, the proofs in this paper do not go through if any of these assumptions is dropped, and the safety margin assumption is particularly crucial.

2.3 Statement of Results

We describe a provably good approximation algorithm for the optimal Cartesian kinodynamic planning problem $(\mathcal{O}, \mathbf{S}, \mathbf{G}, a_{max}, v_{max}, l, c_0, c_1)$, where l is the workspace diameter. Concisely stated, we show:

Theorem 2.1 *Let $(\mathcal{O}, \mathbf{S}, \mathbf{G}, a_{max}, v_{max}, l, c_0, c_1)$ be an optimal kinodynamic planning problem for a point mass robot obeying L_∞ dynamics bounds. Let $0 < \epsilon < 1$.*

Suppose there is a $\delta_v(c_0, c_1)$-safe trajectory from $\mathbf{S}$ to $\mathbf{G}$ taking time T_{opt}. Then the algorithm finds a $(1 - \epsilon)\delta_v(c_0, c_1)$-safe trajectory taking time at most T_{opt} 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 within $O(\epsilon)$ of $\mathbf{S}$ and $\mathbf{G}$, respectively. The approximation approximation error at $\mathbf{S}$ and $\mathbf{G}$ can be controlled independently of v_{max} and l.

For $d = 2, 3$, the running time of the algorithm is

$$O\left(c_A^d N \left[\frac{v_{max}(a_{max}c_1 + v_{max})^3 l}{a_{max}^2 c_0^3 \epsilon^3}\right]^d\right),$$

where N is the geometric complexity of the problem and c_A is a constant. Thus, for a given point robot in a 2- or 3-dimensional world of fixed diameter, the algorithm has an asymptotic time bound of $O\left(c^d N \left(\frac{1}{\epsilon}\right)^{3d}\right)$.

Thus, the algorithm will find a trajectory that takes at most as long as the optimal kinodynamic solution but that might be a factor of ϵ less safe. The theorem states that the algorithm runs in time polynomial in the geometric complexity N and in the resolution $(\frac{1}{\epsilon})$, as does the result from [1]. Our new result is a significant improvement over the [1] result in both the approximation accuracy and the complexity bounds, as described in Section 1.

Observing more closely, we note that an optimal kinodynamic planning problem $\mathcal{K}$ has three components: The *combinatorial complexity* of $\mathcal{K}$ is the number N of faces in the arrangement of obstacles $\mathcal{O}$. The *algebraic complexity of the geometry* is the number of bits necessary to encode the coordinates of the vertices of $\mathcal{O}$, and the start and goal states. The *algebraic complexity of the kinodynamic bounds* is the number of bits necessary to encode the kinodynamic bounds $(a_{max}, v_{max}, c_1, c_0)$. In the language of combinatorial optimization [8], we show that our algorithm is an ϵ-approximation scheme that is *fully polynomial* in the combinatorial and algebraic complexity of the geometry, and *pseudo-polynomial* in the kinodynamic bounds.

We also note that neither our algorithm nor the [1] algorithm guarantees that the approximate optimal safe solution will be near optimal safe solution except at its endpoints. In this respect, these *trajectory* planning algorithms are similar to Papadimitriou's fully polynomial approximation scheme for 3D Euclidian shortest paths [9]. Again, the closeness of the approximation is strictly in terms of the optimization measure, so the optimal solution might not appear spatially similar to the truly optimal. In fact, the results of [4] imply that finding a path that is is homotopic to the optimal is $\mathcal{NP}$-hard. (See Appendix A.)

We have completed a preliminary Common Lisp implementation of this algorithm in two dimensions. In Section 5 we briefly report on this and describe extensions to the main result.

2.4 Previous and Related Work

For a review of issues in robotics and algorithmic motion planning, see [10, 11]. There exists a large body of work on optimal control in the control theory and robotics literature. For example, see [12–16]. Much of this work provides partial analytic characterizations of time-optimal solutions. Among significant results, [12, 13] show how to *time-rescale* the velocity profile of given a *particular* trajectory to obtain a trajectory that is time-optimal with respect to dynamics constraints. This flavor of theoretical work has led to algorithms that attempt to find nearly time-optimal trajectories, notably [17] and [18]. None of these results provided analytically guaranteed closeness to global optimality, and assuring one could control the accuracy of these algorithms by increasing the number of gridpoints required that their running-time bounds be exponential this number.

The polyhedral Euclidian shortest path problem can be viewed as a version of optimal kinodynamic planning in which the acceleration bound a_{max} is set to to infinity. This observation may be used to extend the results of [4] to show that in 3D, optimal kinodynamic planning is $\mathcal{NP}$-hard; a proof sketch is given in the Appendix of this paper. [9] gives a fully-polynomial approximation algorithm for the shortest path problem. Ó'Dúnlaing [19] provides an exact algorithm for one-dimensional kinodynamic planning. These methods may extend to the 2- and 3D cases as well. Kinodynamic planning in 2D is related to the problem of planning with non-holonomic constraints, as studied by Fortune and Wilfong [20, 21] and Jacobs and Canny [22]. In this problem, a robot with wheels and a bounded minimum turning radius must be moved. To make the analogy clear, in our case, the minimum turning radius is $\frac{1}{a_{max}}\|\dot{\mathbf{p}}\|^2$.

[1] (see also its revision, [5]) introduce the use of an ϵ-approximation problem formulation to kinodynamic planning and provide the first provably good approximation algorithm for 2- and 3-dimensional optimal kinodynamic planning. The work presented in this paper improves on their result in both the accuracy of the approximation and in the complexity of the algorithm. These stronger results are obtained by modifying the earlier algorithm and by using new constructive trajectory proof techniques that utilize the velocity bounds in the problem. In Section 3, we describe the improved algorithm, with the changes presented in Section 3.2. In Section 4 we present the complexity analysis.

3 Kinodynamic Planning with L_∞ Dynamics Bounds

3.1 The Basic Idea

The basic idea behind our approach, beginning with [1], is to reduce the problem of finding an approximately minimal-time trajectory to finding the shortest path in a directed graph. The vertices of the graph "discretize" the statespace TC, and the edges of the graph correspond to trajectory segments that each take time τ, a parameter computed by the algorithm.

Given the acceleration bounds a_{max}, let $\mathcal{A}$ be the set of constant accelerations whose components are members of $\{-a_{max}, 0, a_{max}\}$. Let us choose a timestep τ such that velocity bound v_{max} is a multiple of $a_{max}\tau$. Applying a member of $\mathcal{A}$ for duration τ is called an (a_{max}, τ)-*bang*. (See Figures 2 and 3.) We also use this term to refer to the resulting trajectory segment: we say that there is an (a_{max}, τ)-*bang* from state $\mathbf{X}$ to state $\mathbf{Y}$ if following an (a_{max}, τ)-bang moves from $\mathbf{X}$ to $\mathbf{Y}$.

Suppose $\mathbf{S}^* = (\mathbf{s}^*, \dot{\mathbf{s}}^*) \in TC$ is a state such that $\dot{\mathbf{s}}^*$ is a vector of integer multiples of $a_{max}\tau$. Suppose that $(\mathbf{x}, \dot{\mathbf{x}})$ is a state reachable from $\mathbf{S}^*$ by some sequence of (a_{max}, τ)-bangs. Then for each coordinate i,

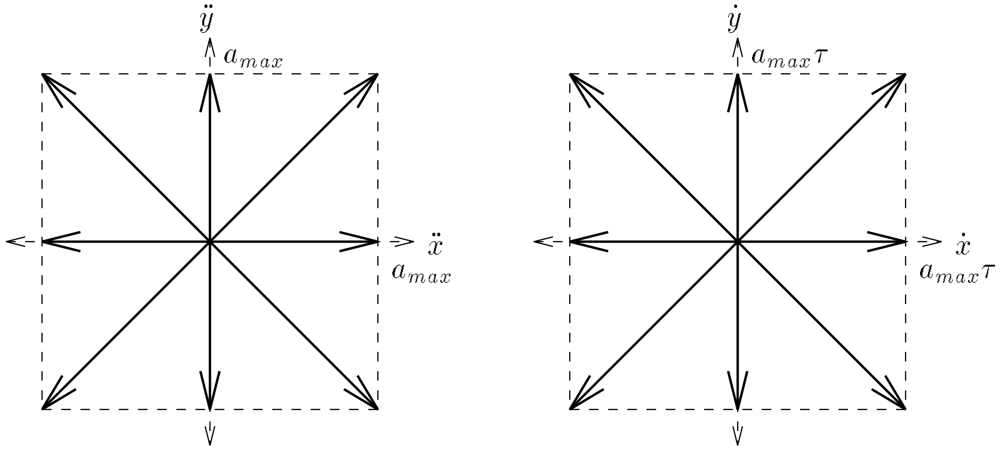


Figure 2: For L_∞ dynamics bounds: extremal accelerations (left) that generate (a_{max}, τ)-bangs, with the velocity components shown here (right).

$$
\begin{aligned}
x_i &= s_i^* + \tfrac{m_i}{2} a_{max} \tau^2 \quad \text{and} \\
\dot{x}_i &= \dot{s}_i^* + n_i a_{max} \tau
\end{aligned} \tag{8}
$$

for some integers m_i and n_i. Thus, all states reachable from $\mathbf{S}^*$ by a sequence of (a, τ)-bangs belong to a set of states that lie at the interstices of an underlying, regular grid embedded in TC. This grid has spacings of $\frac{a_{max}\tau^2}{2}$ in position and $a_{max}\tau$ in velocity. We call this set of interstitial states the *TC-grid*, and each of these states a *TC gridpoint.* We call a trajectory that results from a sequence of (a_{max}, τ)-bangs between TC-gridpoints an *(a_{max}, τ)-grid-bang trajectory.* Its velocity function is a *grid-bang velocity function.*

Recall the definition of δ'_v-safety (6). We say that state $(\mathbf{x}, \dot{\mathbf{x}})$ obeys δ'_v-safety if the ball of radius $\delta'_v(\dot{\mathbf{x}})$ about $\mathbf{x}$ lies in free space. $\mathbf{S}^*$, v_{max}, δ'_v, $\mathcal{O}$, and (a_{max}, τ)-bangs determine a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ embedded in TC. The vertices $v_i \in \mathcal{V}$ are the *TC-gridpoints.* Each edge $e_j \in \mathcal{E}$ is a δ'_v-safe (a_{max}, τ)-bang between two of these vertices. We say that τ, $\mathbf{S}^*$, a_{max}, v_{max}, c_0, c_1, $\mathcal{O}$, and ϵ *induce the reachability graph* $\mathcal{G}(\mathcal{V}, \mathcal{E})$, which we say is *rooted* at $\mathbf{S}^*$.

The smaller τ is, the finer the underlying TC-grid, and the better some (a_{max}, τ)-grid-bang trajectory will approximate an arbitrary trajectory that starts at $\mathbf{S}^*$ and obeys the kinodynamic constraints. Thus, it is intuitively plausible that if τ is small enough and Γ_{opt} is an optimal trajectory from a state $\mathbf{S}$ sufficiently near $\mathbf{S}^*$ to a state $\mathbf{G}$, then there will be a δ'_v-safe (a_{max}, τ)-grid-bang trajectory Γ_q going from $\mathbf{S}^*$ to a state $\mathbf{G}^*$ near $\mathbf{G}$ in approximately time T_{opt}. Furthermore, since this trajectory need only obey $\delta'_v(c_0, c_1)$-safety, it is conceivable that it might take time $T_q \leq T_{opt}$.

A naïve algorithm might therefore do the following. First it would choose a time-step τ as a function of a_{max}, v_{max}, ϵ, c_0, and c_1. Then, it would choose a start state $\mathbf{S}^*$ that approximates $\mathbf{S}$, with the restriction that $a_{max}\tau$ divides $\dot{s}^*$. Finally, it would search for the shortest path in the induced graph to any of a set of vertices that approximate $\mathbf{G}$, and return the trajectory corresponding to this path. The closeness of the approximations to $\mathbf{S}$ and $\mathbf{G}$ would improve as τ decreases.

This describes the gist of the [1] algorithm. The main burden in proving the correctness is to show how to choose an adequately small τ that induces a reachability graph only polynomially large in $\frac{1}{\epsilon}$. The disadvantage of this algorithm is that, in general, it is only guaranteed to approximate the goal to $O(v_{max}\tau)$ in position, which means that the approximation closeness is dependent on the "size" of the statespace.

3.2 The Algorithm

3.2.1 Modifications to the Naïve Algorithm

Recall that edges in the reachability graph correspond to (a_{max}, τ)-bangs between vertices. A state in the image of one of these trajectory segments is an *edge-state* if it is not a vertex. Since an (a, τ)-bang trajectory might come closest to the goal state at a time that is not an integer multiple of τ, we might expect that an algorithm that checks the closeness of edge-states to the goal will find a better approximation than the naïve algorithm would find. (See Figure 4.) This intuition is correct, and by considering edge-states as well as

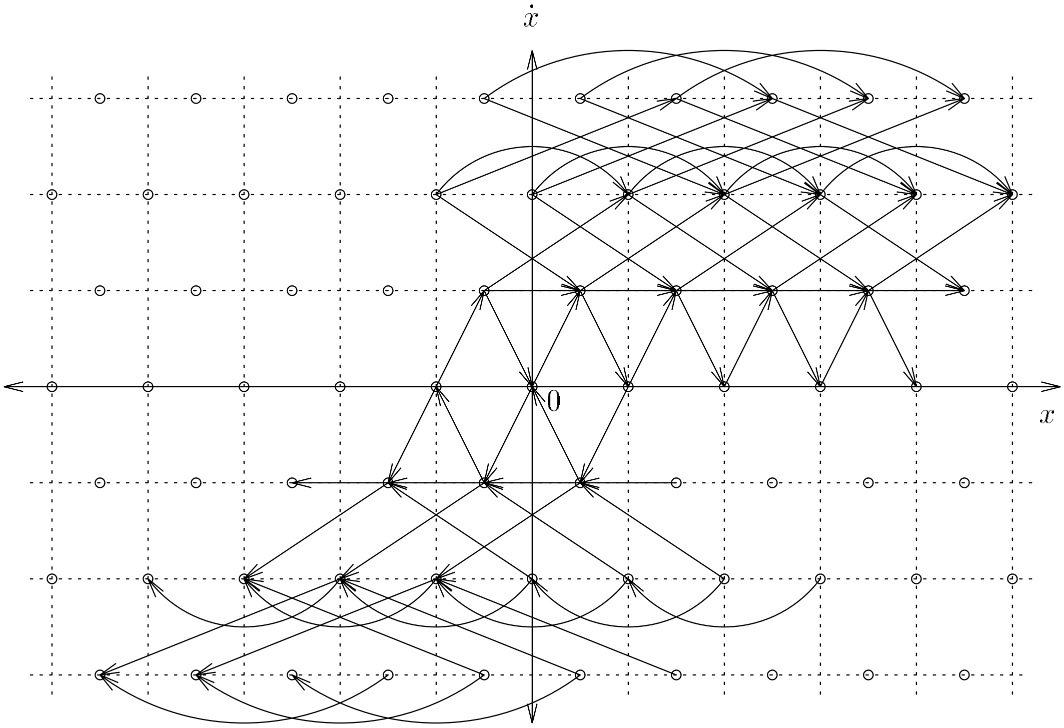


Figure 3: TC-gridpoints lying in phase-space for a one dimensional problem. TC-gridpoints are the vertices of a reachability graph. Shown here are *some* of the edges for a reachability graph rooted at the origin, with no obstacles nearby. Note that an edge does not show the actual trajectory corresponding to an (a_{max}, τ)-bang, which traces a quadratic curve in TC.

vertices, our algorithm can find a trajectory that approximates $\mathbf{G}$ to within $O(a_{max}\tau^2)$ in position.

We define a *graph trajectory* to be a trajectory that begins at the root $\mathbf{S}^*$ of the reachability graph $\mathcal{G}$ and is a sub-trajectory of some trajectory corresponding to a path in the graph. Thus, the algorithm looks for a minimal-time graph trajectory to an edge-state or vertex within the appropriate closeness of $\mathbf{G}$.

If $\dot{s}_i$ is not a multiple of $a_{max}\tau$ for some coordinate i, then it is hard to choose the root vertex $\mathbf{S}^*$ that approximates $\mathbf{S}$ and that is the best for finding an approximately optimal graph trajectory without a cumbersome case analysis. We attempt to simplify the description by using the following trick: whenever some $\dot{s}_i$ is not a multiple of $a_{max}\tau$ the algorithm chooses $\mathbf{S}^*$ so that $\mathbf{S}$ will be approximated by $\Gamma_q(\tau)$, for any graph trajectory Γ_q beginning at $\mathbf{S}^*$. If Γ_q is the minimal-time graph trajectory to a state adequately close to the goal, the algorithm will then return the sub-trajectory that begins at time τ.

3.2.2 Our Algorithm Step by Step

We first introduce terminology to describe how closely one state approximates another. Let $\mathbf{X} = (\mathbf{x}, \dot{\mathbf{x}})$ and $\mathbf{Y} = (\mathbf{y}, \dot{\mathbf{y}})$ be two states, and suppose that

$$\begin{aligned} \|\mathbf{x} - \mathbf{y}\|_\infty &\leq \eta_x \quad \text{and} \\ \|\dot{\mathbf{x}} - \dot{\mathbf{y}}\|_\infty &\leq \eta_v. \end{aligned}$$

Then we say that $\mathbf{X}$ *is within* (η_x, η_v) *of* $\mathbf{Y}$, or that $\mathbf{X}$ *approximates* $\mathbf{Y}$ *to tolerance* (η_x, η_v).

Our improved algorithm does the following

1. Chooses a timestep τ as a function of a_{max}, v_{max}, ϵ, c_0, and c_1. Specifically, the algorithm chooses the largest τ such that $\tau \leq \frac{v_{max}}{a_{max}}$, $a_{max}\tau | v_{max}$, and $\tau \leq \frac{c_0 \epsilon}{2a_{max}c_1(1-\epsilon)+5v_{max}}$.

2. Next, it chooses the starting TC-gridpoint $\mathbf{S}^*$ that roots the reachability graph. For each coordinate i:

$$\begin{aligned} \dot{s}_i^*(0) &= \text{the multiple of } a_{max}\tau \text{ closest to } \quad \dot{s}_i \\ s_i^*(0) &= s_i(0) - \tfrac{\tau^2}{2}(\dot{s}_i(0) + \dot{s}_i^*(0)). \end{aligned} \tag{9}$$

3. It then searches for a minimal-time graph trajectory from $\mathbf{S}^*$ to any state that is within $(\frac{5a_{max}\tau^2}{2}, 2a_{max})$ of $\mathbf{G}$. This search is basically a breadth-first search of the induced reachability graph $\mathcal{G}$. In the n^{th} generation of the search the algorithm finds vertices n edges from $\mathbf{S}^*$. However, as the algorithm explores an edge out of a vertex, it checks whether the corresponding (a_{max}, τ)-bang comes adequately close to $\mathbf{G}$. If during a generation n more than one such (a_{max}, τ)-bangs are found, the algorithm chooses one that comes adequately close to $\mathbf{G}$ at the earliest time. The algorithm uses backpointers to construct a minimal-time graph trajectory when the search succeeds.

4. Let Γ_q be the minimal-time graph trajectory found above. The algorithm returns Γ_q^τ, defined by:

$$\Gamma_q^\tau(t) = \Gamma_q(t + \tau). \tag{10}$$

3.3 Analyzing the Algorithm

We claim that if an optimal δ_v-safe trajectory Γ_{opt} from $\mathbf{S}$ to $\mathbf{G}$ takes time T_{opt}, then the algorithm will find a δ'_v-safe trajectory taking at most time T_{opt} and approximating $\mathbf{S}$ and $\mathbf{G}$ to within tolerances that are $O(\epsilon)$ and that do not grow with l or v_{max}. A key lemma shows how to choose a timestep τ that is $\Omega(\epsilon)$ such that the choice of $\mathbf{S}^*$ (9) assures that there will be a δ'_v-safe graph trajectory Γ_q taking time

$$T_q \leq T_{opt} + \tau$$

and going from $\mathbf{S}^*$ to a state $\mathbf{G}^*$ approximating $\mathbf{G}$ to the appropriate tolerance given in step 3 above. The lemma also bounds $\|\Gamma_q(\tau) - \mathbf{S}\|$. This shows that searching for the shortest graph trajectory will be adequate.

There are two components to the algorithm's complexity: the size of the reachability graph $\mathcal{G}$ and the cost of checking whether an (a_{max}, τ)-bang (potential graph edge) is δ'_v-safe. The maximal out-degree in $\mathcal{G}$ is 3^d, and the number of vertices is bounded by the number of TC-gridpoints, which is proportional to $\left(\frac{l v_{max}}{a_{max}^2 \tau^3}\right)^d$. Safety-checking using simple computational-geometric techniques described in Appendix B costs $O(N)$ per (a_{max}, τ)-bang, as in [5]. With minor modifications, the technique for checking safety with respect to a single obstacle is also used to check closeness to the goal. Thus we get the complexity bound $O(c^d N (\frac{1}{\epsilon})^{3d})$.

4 Better Bounds for a Cartesian Robot with L_∞ Dynamics Bounds

4.1 Overview

We now outline the argument proving the parts of Theorem 2.1 concerning the goodness of the approximation and the running-time of the algorithm. The general idea is that the choice of timestep τ and root vertex $\mathbf{S}^*$ will guarantee that for every δ_v-safe trajectory beginning at start state $\mathbf{S}$, there will be a graph trajectory that "tracks" it closely enough to be $(1-\epsilon)\delta_v$-safe and to take the same amount of time. We now formalize our notion of "tracking".

Definition 4.1 *Consider two trajectories Γ_a, $\Gamma_b : [0, T] \to TC$. Given two scalars η_x and η_v, we say that we say that Γ_a* approximately tracks Γ_b to tolerance (η_x, η_v) in the L_∞-norm *if for all times t,*

$$
\begin{array}{rcll}
\|\mathbf{p}_a(t) - \mathbf{p}_b(t)\|_\infty & \leq & \eta_x, & \textit{and} \\
\|\dot{\mathbf{p}}_a(t) - \dot{\mathbf{p}}_b(t)\|_\infty & \leq & \eta_v. &
\end{array}
\tag{11}
$$

It is useful to think of a trajectory Γ inducing a tube of diameter at least η_x in position and η_v in velocity around its image in TC, and we call this tube the (η_x, η_v)*-tube induced by* Γ. If Γ_a approximately tracks Γ_b to tolerance (η_x, η_v), then each trajectory lies in the (η_x, η_v)-tube induced by the other. (See Figure 5.)

Now, suppose that Γ_r is a trajectory from $\mathbf{S}$ to $\mathbf{G}$ taking time T_r, and suppose furthermore that Γ_r obeys dynamics bounds a_{max} and v_{max} and is $\delta_v(c_0, c_1)$-safe. Let $0 < \epsilon < 1$.

To prove the theorem, we first describe a one-parameter family of *safe tracking tolerances* (η_x, η_v) such that if trajectory Γ_q tracks Γ_r to a tolerance in this family, then Γ_q will be $(1-\epsilon)\delta_v(c_0, c_1)$-safe. This family of tolerances is given by the following lemma, which is proven in [1, 3, 5]. Note that because the lemma holds for any L_p-norm, our results are easily extendible to safety margin given in any such norm.

Lemma 4.1 (The Safe Tracking Lemma) *Suppose that δ_v is specified by c_0 and c_1 and that Γ_r is a δ_v-safe trajectory. Let $0 < \epsilon < 1$, and let ${\delta_v}' = (1-\epsilon)\delta_v$. Then there exists a tolerance (η_x, η_v) 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:*

$$
\begin{array}{rcl}
\eta_v & \leq & \frac{c_0 \epsilon}{c_1(1-\epsilon)+\beta} \\
\eta_x & \leq & \beta \eta_v.
\end{array}
\tag{12}
$$

Second, we show how to choose a timestep τ and a root vertex state $\mathbf{S}^*$ so that in the absence of obstacles there will be some graph-trajectory Γ_q such that (a) $\Gamma(\tau)$ is within (η_x, η_v) of $\mathbf{S}$, and (b) Γ_q^τ tracks Γ_r to tolerance (η_x, η_v). (Recall that $\Gamma_q^\tau(t) = \Gamma_q(t+\tau)$ from (10).) This choice of τ and $\mathbf{S}^*$ is given in Lemma 4.2 (The Strong Tracking Lemma). If (η_x, η_v) is a safe tracking tolerance, then Γ_q^τ will be $(1-\epsilon)\delta_v(c_0, c_1)$-safe in the presence of the obstacles.

To complete the proof, we show how to choose a safe tracking tolerance (η_x, η_v) so that there will be a timestep τ that both is $\Omega(\epsilon)$ and satisfies Lemma 4.2. It follows that the number of TC-gridpoints will be $O\left(\left(\frac{1}{\epsilon}\right)^{3d}\right)$. Since each TC-graph vertex has a maximum out-degree of 3^d, and checking the $(1-\epsilon)\delta_v(c_0, c_1)$-safety of an (a_{max}, τ)-bang is $O(N)$ (as proven in Appendix B), it follows that searching the TC-graph for a shortest path from $\mathbf{S}^*$ to $\mathbf{G}^*$ in the TC-graph takes time $O\left(3^d N \left(\frac{1}{\epsilon}\right)^{3d}\right)$. The lemmas and theorem that follow and the terms in the algorithm give us Theorem 2.1.

We note that the structure of the proof of the main result is similar to that in [1]. However, the Strong Tracking Lemma (4.2) shows how to choose a timestep and root vertex so that some graph trajectory will track a *an optimal trajectory*, and not merely an optimal trajectory that has been ϵ-time-rescaled, i.e., slowed down by a factor of ϵ. Furthermore, the proof of the lemma is significantly different from that of the Tracking Lemma of [1]. The main part of the our proof is based on tracking velocity functions and makes use of the velocity limits in the problem to bound the duration over which tracking error can increase, unlike the corresponding proof in the earlier work. The construction is done with explicit inductive definition. Finally, additional constructions involving the initial and terminal segments guarantee that the approximation closeness at the start and goal is achieved independently of v_{max}, and with an $\Omega(\epsilon)$ timestep.

4.2 The Strong Tracking Lemma

4.2.1 Preliminary Discussion of the Lemma

Our tighter approximation and time-complexity bounds rely on the following lemma.

Lemma 4.2 (The Strong Tracking Lemma) *Suppose a trajectory Γ_r respects acceleration bound a_{max} and velocity bound v_{max} and takes time T_r. Furthermore, suppose that $a_{max}\tau$ divides v_{max}. Then in the absence of obstacles, the following hold.*

1. *For any positive η_x, η_v, there exists a timestep size τ and a choice of root vertex $\mathbf{S}^*$ such that a graph-trajectory Γ_q with the following properties exists:*
   - *(a) $\Gamma_q(\tau)$ is within $(a_{max}\tau^2, 2a_{max}\tau)$ of $\Gamma_r(0)$, and for all $t \in [0, \tau]$, $\Gamma_q(t)$ is within (η_x, η_v) of $\Gamma_r(0)$;*
   - *(b) $\Gamma_q(t + \tau)$ is within tolerance (η_x, η_v) of $\Gamma_r(t)$ for all $t \in [0, T_r]$, and $\Gamma_q(T_r + \tau)$ is within $(\frac{5a_{max}\tau^2}{2}, 2a_{max}\tau)$ of $\Gamma_r(T_r)$*
2. *Moreover, τ is polynomial in η_x, η_v, $\frac{1}{v_{max}}$, and $\frac{1}{a_{max}}$. Specifically, τ can be chosen as the largest τ such that $a_{max}\tau$ divides v_{max} and*

$$\tau \leq \min\left(\frac{\eta_x}{5v_{max}}, \frac{\eta_v}{2a_{max}}\right). \tag{13}$$

As in [1], it will be sufficient to consider the one-dimensional case, since we are using the L_∞-norm for dynamics bounds. Assume that trajectory Γ_r obeys the velocity and acceleration bounds. Let us call a function that describes the velocity of some grid-bang trajectory a *grid-bang velocity function.* We initially assume that $T_r = K\tau$ for some integer K, and that v_{max}, $v_r(0)$ and $v_r(T_r)$ are multiples of $a_{max}\tau$.

The proof has four stages. First, we show there exists a pair of grid-bang velocity functions bounding v_r from above and below while staying within a constant of v_r. (See Figure 6.) Second, we use these bounding trajectories to show there exists a grid-bang trajectory $\widetilde{\Gamma}_q$ that tracks Γ_r to a tolerance that is a function of the dynamics bounds and τ;

$\widetilde{\Gamma}_q(t)$ approximates $\Gamma_r(t)$ to within $(2v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$ for all times $t \in [0, T_r]$. (See Figure 8.) Third, we show that there exists a grid-bang trajectory Γ_q that tracks Γ_r only slightly less closely than $\widetilde{\Gamma}_q$ but that approximates Γ_r better at T_r. (See Figure 9.) Finally, we relax our assumptions about Γ_r, except for the condition that $a_{max}\tau$ divide v_{max}. To relax the assumptions, we show how to prepend and append trajectory segments of length τ or less to Γ_r to reduce the proof to the restricted case. (See Figures 10 and 11.)

We now introduce a notation to describe grid-bang trajectories. Observe that a grid-bang trajectory Γ lasting for $K\tau$ is uniquely described by the initial state $(\mathbf{p}(0), \dot{\mathbf{p}}(0))$ and a sequence of vectors $\{\mathbf{c}^0, \ldots, \mathbf{c}^{K-1}\}$ whose individual components are members of $\{-1, 0, 1\}$, i.e., $c_i^n \in \{-1, 0, 1\}^d$. Then we have for for $t \in [n\tau, (n+1)\tau)$:

$$\begin{aligned} \mathbf{v}(t) &= \mathbf{v}(n\tau) + (t - n\tau)a_{max}\mathbf{I}\mathbf{c}^n \\ \mathbf{p}(t) &= \mathbf{p}(n\tau) + \textstyle\int_{n\tau}^{t} \mathbf{v}(s)ds, \end{aligned} \tag{14}$$

where $\mathbf{I}$ is the identity matrix.

4.2.2 Bounding Velocity Functions

The following lemma shows the existence of two "bounding" grid-bang velocity functions for every one-dimensional velocity function v_r that obeys bounds v_{max} and a_{max}. (See Figure 6.) For a velocity function $\mathbf{v}_r$ in d dimensions there will exist d such pairs of functions. In the construction of an (a_{max}, τ)-bang trajectory that approximates Γ_r, we will use these bounding functions to guarantee velocity-tracking.

Lemma 4.3 *Let τ be fixed such that $a_{max}\tau | v_{max}$. Let v_r be a velocity function obeying dynamics bounds v_{max} and a_{max}, and let $\tau | T_r$. Suppose that $a_{max}\tau$ divides $v_r(0)$ and $v_r(T_r)$.*

Then there exist two grid-bang velocity functions $v_{slow}, v_{fast} : [0, T_r] \to [-v_{max}, v_{max}]$ that satisfy the following five conditions:

1. *$v_{slow}(0) = v_r(0) = v_{fast}(0)$,*
2. *$v_{slow}(T_r) = v_r(T_r) = v_{fast}(T_r)$,*
3. *$v_{slow}(t) \le v_r(t) \le v_{fast}(t)$ for all $t \in [0, T_r]$,*
4. *$v_{fast}(t) - v_r(t) \le 2a_{max}\tau$ and $v_r(t) - v_{slow}(t) \le 2a_{max}\tau$ for all $t \in [0, T_r]$, and*
5. *for all n, $v_{fast}(n\tau) - v_r(n\tau) \le \frac{3a_{max}\tau}{2}$ and $v_r(n\tau) - v_{slow}(n\tau) \le \frac{3a_{max}\tau}{2}$.*

Proof: We first define a grid-bang velocity function $\tilde{v}_r$ that approximates v_r. (See Figure 7.) Let us define $\tilde{v}_r(n\tau)$ to be the integral multiple of $a_{max}\tau$ that most closely approximates[4] $v_r(n\tau)$. We then inductively construct $\tilde{v}_r$:

[4] In cases when $v_r(\tau) - \left\lfloor \frac{v_r(n\tau)}{a_{max}\tau} \right\rfloor a_{max}\tau = \frac{a_{max}}{2}$, the floor or ceiling must be chosen consistently.

- Initialization: $\tilde{v}_r(0) = v_r(0)$.
- Stage n: for $t \in [n\tau, (n+1)\tau)$,

$$\tilde{v}_r(t) = \tilde{v}_r(n\tau) + \frac{(\tilde{v}_r((n+1)\tau) - \tilde{v}_r(n\tau))(t - n\tau)}{\tau}. \tag{15}$$

Thus, for all n, $|v_r(n\tau) - \tilde{v}_r(n\tau)| \leq \frac{a_{max}\tau}{2}$. It follows that $\tilde{v}_r$ also obeys acceleration bound a_{max}, or else v_r would have to violate a_{max}. Furthermore, because of the closeness at multiples of τ, $|v_r(t) - \tilde{v}_r(t)| \leq a_{max}\tau$ for all $t \in [0, T_r]$. It is obvious that $\tilde{v}_r$ obeys velocity bound v_{max}.

We now define:

1. $\tilde{v}_{fast}(t) = \min(v_{max}, \tilde{v}_r(t) + a_{max}\tau)$.
2. $\tilde{v}_{slow}(t) = \max(-v_{max}, \tilde{v}_r(t) - a_{max}\tau)$.

It is straightforward to verify that these grid-bang velocity functions satisfy the last three claims of the lemma.

v_{fast} and v_{slow} are similar to $\tilde{v}_{fast}$ and $\tilde{v}_{slow}$. First, we set

$$v_{fast}(0) = v_{slow}(0) = \tilde{v}_r(0) = v_r(0), \text{ and}$$

$$v_{fast}(T_r) = v_{slow}(T_r) = \tilde{v}_r(T_r) = v_r(T_r).$$

We then construct the functions forwards in time from $t = 0$ and backwards from $t = T_r$ until they match $\tilde{v}_{fast}(t)$. For the v_{fast} case at the $t = 0$ end, we set

$$\frac{d}{dt}(v_{fast}(t)) = \min(a_{max}, \frac{d}{dt}\tilde{v}_{fast}(t) + a_{max})$$

until the minimal time when $v_{fast}(t) = \tilde{v}_{fast}(t)$. If there is no such time, then $v_{fast} = \tilde{v}_r$. The other cases are similar. $\square$

4.2.3 Tracking a Restricted Trajectory

Given a trajectory Γ_r that begins and ends with velocities that are multiples of $a_{max}\tau$ and that takes a multiple of τ time, we show that there exists an (a_{max}, τ)-bang trajectory that matches it at the start, approximates it at the end, and tracks it to a tolerance that is independent of trajectory time. Lemma 4.4 establishes that the start state can be matched and that the tracking tolerance can be obeyed for an arbitrarily long trajectory. (See Figure 8.) The trajectories v_{slow} and v_{fast} are used in the construction as markers to bound velocity tracking error and enforce exact matches at $t = 0$ and $t = T_r$. According to Lemma 4.5, with an increase in tracking error, an (a_{max}, τ)-bang trajectory can also approximate $\Gamma_r(T_r)$ to within $a_{max}\tau^2$ in position.

Lemma 4.4 *Let τ be fixed such that $a_{max}\tau | v_{max}$. Let Γ_r have velocity function v_r obeying dynamics bounds v_{max} and a_{max}, and let $\tau | T_r$. Suppose $a_{max}\tau$ divides $v_r(0)$ and $v_r(T_r)$.*

Then there is an (a_{max}, τ)-bang trajectory $\widetilde{\Gamma}_q$ such that for all times $t \in [0, T_r]$,

$$
\begin{aligned}
|v_r(t) - \tilde{v}_q(t)| &\leq 2a_{max}\tau, \quad \textit{and} &(16)\\
|p_r(t) - \tilde{p}_q(t)| &< 2v_{max}\tau + a_{max}\tau^2. &(17)
\end{aligned}
$$

Proof:

We inductively define an (a_{max}, τ)-bang trajectory $\widetilde{\Gamma}_q$ that tracks Γ_r to the tolerance in the lemma. The definition works for semi-infinite trajectories, and it is similar to simple finite injury constructions from recursion theory.

In stage n of the construction, we define $\tilde{v}_q(t)$ for all $t \in [n\tau, (n+1)\tau)$ and possibly alter a previously defined section of the function. Our choice of $\widetilde{\Gamma}_q(0)$ serves as the root vertex of the reachability graph and determines the position-space alignment of the TC-grid coordinates. Let $\tilde{v}_r$, v_{fast} and v_{slow} be defined as in Lemma 4.3.

- Initially, set $\tilde{p}_q(0) = p_r(0)$ and $\tilde{v}_q(0) = \tilde{v}_r(0)$. The origin of the grid is chosen so that $(\tilde{p}_q(0), \tilde{v}_q(0))$ is a gridpoint.
- Stage n (≥ 0): We choose $c^n \in \{-1, 0, 1\}$, and set $\tilde{v}_q(t) = \tilde{v}_q(n\tau) + c^n a_{max}(t - n\tau)$ for $t \in [n\tau, (n+1)\tau)$. The following rules determine c^n.
    1. Initially, c^n is chosen to minimize $|p_r((n+1)\tau) - \tilde{p}_q((n+1)\tau)|$ subject to the condition $v_{slow}(t) \leq \tilde{v}_q(t) \leq v_{fast}(t)$ and subject to the previously chosen $c^0, \ldots, c^{n-1}$.
    2. Let $y^{(n)} = p_r((n+1)\tau) - \tilde{p}_q((n+1)\tau)$. We call $y^{(n)}$ the *lag during stage n*. If $|y^{(n)}| > 2v_{max}\tau$, then
        (a) for the greatest integer $m < n$ such that $c^m \neq \mathrm{sgn}(y^{(n)})$, set $c^m = c^m + \mathrm{sgn}(y^{(n)})$,
        (b) if there is a greatest integer $j, m < j \leq n$, such that setting $c^j = c^j - \mathrm{sgn}(y^{(n)})$ minimizes $|p_r((n+1)\tau) - \tilde{p}_q((n+1)\tau)|$, or if otherwise $\tilde{v}_q((n+1)\tau)$ would not be bounded by $v_{fast}((n+1)\tau)$ and $v_{slow}((n+1)\tau)$, then set $c^j = c^j - \mathrm{sgn}(y^{(n)})$.

By this definition, for all $t \in [0, T_r]$, $v_{slow}(t) \leq \tilde{v}_q(t) \leq v_{fast}(t)$, so $\tilde{v}_q(t)$ satisfies the first claim in the lemma.

Before we show $\tilde{v}_q$ satisfies the second claim, we observe that if we only use the first rule at each stage n, then

$$|p_r(t) - \tilde{p}_q(t)| < 4v_{max}\tau + a_{max}\tau^2$$

for all $t \in [0, T_r]$. To see this, first note that Γ_r can maintain maximum acceleration for at most time $\frac{2v_{max}}{a_{max}}$, since otherwise it would violate the velocity bounds. Therefore, the rule implies that if at some time t_b

$$p_r(t_b) - \tilde{p}_q(t_b) \geq a_{max}\tau^2,$$

then $\tilde{v}_q$ will equal or exceed v_r at some time no later than $t_b + \frac{2v_{max}}{a_{max}}$. The second rule gives us the tighter approximation in the lemma; although this does not effect the asymptotic complexity bounds, it significantly affects the running time of any implementation.

We now verify that the second rule for stage n of the construction is consistent and guarantees (17). Let

$$\widehat{\Gamma}_q^{(n)} = (\hat{p}_q^{(n)}, \hat{v}_q^{(n)})$$

be the trajectory just before the rule is applied, and let $\tilde{\Gamma}_q^{(n)}$ denote the trajectory at the end of stage n. Assume that $\tilde{p}^{(n-1)}$ obeys the lemma for all $t \in [0, n\tau]$. To see that the condition $|y^{(n)}| \geq 2v_{max}$ is adequate for triggering changes to the $\{c^k\}$, we observe that if $|p_r(t) - \hat{p}_q(t)| > 2v_{max}\tau + a_{max}\tau^2$ for some $t \in (n\tau, (n+1)\tau]$, then $|y^{(n)}| \geq 2v_{max}$. Thus, if a c^m is found as specified, then $|p_r(t) - \tilde{p}_q^{(n)}(t)| \leq 2v_{max}\tau + a_{max}\tau^2$ for all $t \in [n\tau, (n+1)\tau]$.

Wlog we will assume that the rule is applied when the lag $y^{(n)}$ exceeds $2v_{max}\tau$; the negative lag case is symmetrical. Since the lag $y^{(n)} > 2v_{max}\tau$, there is some minimal t_w such that

$$\hat{v}_q^{(n)}(t) < v_r(t) \leq v_{fast}(t) \quad \text{for all } t \in (t_w, n\tau).$$

Otherwise, the first rule would have been violated at a previous stage. Furthermore, there is an integer m such that $m\tau < (n-1)\tau$, $t_w < (m+1)\tau$, and $c^m \leq 0$, and $c^{m+1}, ..., c^{n-1} = 1$ before the rule is triggered.

Finally, we must ensure that the rule does not cause the new $\tilde{p}_q^{(n)}(t)$ to become too much larger than $p_r(t)$ in the interval $[m\tau, (n+1)\tau]$. Recall that for all integers k, $v_r(k\tau) - v_{slow}(k\tau) \leq \frac{3a_{max}\tau}{2}$. Because $\widehat{\Gamma}_q^{(n)}$ accelerates maximally (respecting a_{max} and v_{max}) over $[(m+1)\tau, (n+1)\tau]$, it follows that $v_r(t) - \hat{v}_q^{(n)}(t) \leq \frac{3a_{max}}{2}$ for all t in this interval. Since $y^{(n)} > 2v_{max}\tau$ and $n\tau - m\tau \leq \frac{2v_{max}}{a_{max}}$, it then follows (with some algebraic manipulation) that

$$p_r(t) - \tilde{p}_q^{(n)}(t) \geq -v_{max}\tau - a_{max}\tau^2 \quad \text{for} \quad t \in [m\tau, (n+1)\tau]$$

Thus $\tilde{p}_q^{(n)}(t)$ obeys (17) for all $t \in [0, (n+1)\tau]$. $\square$

Using the $\widetilde{\Gamma}_q$ defined above, we show the existence of an (a_{max}, τ)-bang trajectory Γ_q that tracks Γ_r a little less closely in position but that is within $(a_{max}\tau^2, a_{max}\tau)$ of $\Gamma_r(T_r)$ at time T_r. Specifically, we modify the velocity function $\tilde{v}_q$ to improve the approximation while staying between v_{fast} and v_{slow}. (See Figure 9.)

Lemma 4.5 *Let τ be fixed such that $a_{max}\tau | v_{max}$. Let Γ_r have velocity function v_r obeying dynamics bounds v_{max} and a_{max}, and let $\tau | T_r$. Suppose $a_{max}\tau | v_r(0), v_r(T_r)$.*

Then, there exists a grid-bang Γ_q that tracks Γ_r to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$ such that $\Gamma_q(0) = \Gamma_r(0)$, $v_q(T_r) = v_r(T_r)$, and $|p_q(T_r) - p_r(T_r)| < a_{max}\tau^2$.

Proof: Given Γ_r obeying the hypothesis of the lemma, let us define v_{fast}, v_{slow}, and $\widetilde{\Gamma}_q$ as in the two previous proofs. We show by construction how to incrementally modify $\widetilde{\Gamma}_q$ into a Γ_q that satisfies the lemma. Let $y = p_r(T_r) - \tilde{p}_q(T_r)$.

- Initialization: Let $\Gamma_q^{(0)} = \tilde{\Gamma}_q$. Go to stage 0.
- Stage n: If $|p_r(T_r) - p_q^{(n)}(T_r)| < a_{max}\tau^2$, then $\Gamma_q = \Gamma_q^{(n)}$. Otherwise, we obtain $\Gamma_q^{(n+1)}$ by modifying $v_q^{(n-1)}$. Let m be the least integer such that

$$\begin{aligned} \operatorname{sgn}(y)\,(v_r(m\tau) - \tilde{v}_q(m\tau)) &> 0 \\ \operatorname{sgn}(y)c^m &\geq 0, \quad \text{and} \\ \operatorname{sgn}(y)c^{m-1} &\leq 0. \end{aligned}$$

  Set $c^{m-1} = c^{m-1} + \operatorname{sgn}(y)$ and $c^m = c^m - \operatorname{sgn}(y)$. Proceed to the next stage.

For $n > 0$ we verify that at stage n we can always find the necessary integer m. We describe the case where $y > 0$; the $y < 0$ case is similar. If $v_q^{(n)}(t) = v_{fast}(t)$ for all t, then $p_q^{(n)}(T_r) \geq p_r(T_r)$. We show that as long as $v_q^{(n)}(t) < v_{fast}(t)$ at some time t, the stage can proceed. Suppose that we have followed the construction through $n-1$ stages, reach stage n, and find that $p_r(T_r) - p_q^{(n)}(T_r) \geq a_{max}\tau^2$. Then, since $v_q^{(n)}(0) = v_{fast}(0)$, there is a least integer k such that $v_q^{(n)}(k\tau) < v_{fast}(k\tau)$. Either $c^{k-1} = 0$ or $c^{k-1} = -1$. Suppose that $c^{k-1} = -1$. Then there must be a least integer $m \geq k$ such that $c_m \geq 0$, since $v_q^{(n)}(k\tau) < v_{fast}(k\tau)$ but $v_q^{(n)}(T_r) = v_{fast}(T_r)$. Suppose that $c^{k-1} = 0$. If $c_k \geq 0$, then $m = k$. Otherwise, by the previous argument, there must be a least integer $m > k$ such that $c_m \geq 0$.

By inspection, we can see that for each n and all $t \in [0, T_r]$,

$$\begin{aligned} p_q^{(n)}(t) \leq p_q^{(n+1)}(t) &\leq p_q^{(n)}(t) + a_{max}\tau^2, \quad \text{and} \\ p_q^{(n+1)}(T_r) &= p_q^{(n)}(T_r) + a_{max}\tau^2. \end{aligned}$$

This ensures that the construction reaches termination and the position-tracking bounds are achieved. The velocity-tracking bounds follow from the fact that $v_{slow}(t) \leq v_q(t) \leq v_{fast}(t)$ for all t. □

The following corollary is immediate.

Corollary 4.6 *Let τ be fixed such that $a_{max}\tau | v_{max}$. Let Γ_r have velocity function v_r obeying dynamics bounds v_{max} and a_{max}, and let $\tau | T_r$. Suppose that $\Gamma_r(T_r)$ is a TC-gridpoint relative to $\Gamma_r(0)$, τ, and the dynamics bounds.*

Then, there exists a grid-bang trajectory Γ_q that tracks Γ_r to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$ such that $\Gamma_q(0) = \Gamma_r(0)$ and $\Gamma_q(T_r) = \Gamma_r(T_r)$.

Proof: Lemma 4.5 asserts all the claims except that $p_q(T_r) = p_r(T_r)$. The corollary follows from the property that (a_{max}, τ)-grid-bang trajectories with the same start states and final velocities will have final positions that differ by a multiple of $a_{max}\tau^2$. □

4.2.4 Removing Restrictions

We will now remove certain restrictions on Γ_r. (See Figures 10 and 11.) First, we remove the restriction that T_r be a multiple of τ. The basic idea is that if we extend the trajectory and obtain a trajectory that takes $\left\lceil \frac{T_r}{\tau} \right\rceil \tau$ time, we can obtain a trajectory that satisfies the hypotheses of Lemma 4.5. The acceleration bound and the timestep then limit how much greater the tracking error is at time T_r than at $\left\lceil \frac{T_r}{\tau} \right\rceil \tau$.

Corollary 4.7 *Let τ be fixed such that $a_{max}\tau | v_{max}$. Let Γ_r have velocity function v_r obeying dynamics bounds v_{max} and a_{max} such that $a_{max}\tau | v_r(0)$ and $a_{max}\tau | v_r(T_r)$.*

Then, there exists a grid-bang Γ_q that tracks Γ_r to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$ such that $\Gamma_q(0) = \Gamma_r(0)$, $|v_q(T_r) - v_r(T_r)| < a_{max}\tau$, and $|p_q(T_r) - p_r(T_r)| < \frac{3a_{max}\tau^2}{2}$.

Proof: Suppose Γ_r is a trajectory obeying the hypotheses of the corollary. Define $T'_r = \left\lceil \frac{T_r}{\tau} \right\rceil \tau$. Then there is a trajectory Γ'_r such that:

$$\begin{array}{rcll} \Gamma'_r(t) & = & \Gamma_r(t) & \text{if} \quad t \in [0, T_r], \\ v'_r(t) & = & v_r(T_r) & \text{if} \quad t \in [T_r, T'_r]. \end{array}$$

By Lemma 4.5, there is an (a_{max}, τ)-grid-bang trajectory Γ_q obeying v_{max} that tracks Γ'_r to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$ such that $\Gamma_q(0) = \Gamma'_r(0)$, $v_q(T'_r) = v'_r(T'_r)$, and $|p_q(T'_r) - p'_r(T'_r)| < a_{max}\tau^2$. Since $\Gamma_r(T_r) = \Gamma'_r(T_r)$ and $v'_r(t) = v_r(T_r) = v_q(T'_r)$ for all $t \in [T_r, T'_r]$, and since v_q obeys acceleration bound a_{max}, it follows that $|p_r(T_r) - p_q(T_r)| < \frac{3a_{max}\tau^2}{2}$ and $|v_r(T_r) - v_q(T_r)| < a_{max}\tau$. □

Next, we remove the restriction that $a_{max}\tau | v_r(T_r)$. Again, we show how to modify the "tail" of Γ_r to get a trajectory that satisfies the hypotheses of Lemma 4.5.

Corollary 4.8 *Let τ be fixed such that $a_{max}\tau | v_{max}$. Let Γ_r have velocity function v_r obeying dynamics bounds v_{max} and a_{max} such that $a_{max}\tau | v_r(0)$.*

Then, there exists a grid-bang Γ_q that tracks Γ_r to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$ such that $\Gamma_q(0) = \Gamma_r(0)$, $|v_q(T_r) - v_r(T_r)| < 2a_{max}\tau$, and $|p_q(T_r) - p_r(T_r)| < \frac{5a_{max}\tau^2}{2}$.

Proof: Suppose Γ_r is a trajectory obeying the hypotheses of the corollary. Define $T'_r = \left\lceil \frac{T_r}{\tau} \right\rceil \tau$, and $T''_r = T'_r - \tau$. Define v_0 to be the multiple of $a_{max}\tau$ closest to $v_r(T_r)$ subject to the condition that $|v_r(T''_r) - v_0| \le a_{max}\tau$.

We then define a trajectory Γ'_r:

$$\begin{array}{rcll} \Gamma'_r(t) & = & \Gamma_r(t) & \text{if} \quad t \in [0, T''_r], \quad \text{and ,} \\ v'_r(t) & = & v'_r(T''_r) + \frac{(t - T''_r)}{\tau}(v_0 - v'_r(T''_r)) & \text{if} \quad t \in [T''_r, T'_r]. \end{array}$$

Γ'_r obeys a_{max}. It follows that $|p_r(T_r) - p'_r(T_r)| < a_{max}\tau^2$, since $T'_r - T''_r = \tau$. By Lemma 4.5, there is an (a_{max}, τ)-grid-bang trajectory Γ_q obeying v_{max} that tracks Γ'_r to to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$ such that $\Gamma_q(0) = \Gamma'_r(0)$, $v_q(T'_r) = v'_r(T'_r)$, and

$|p_q(T'_r) - p'_r(T'_r)| < a_{max}\tau^2$. It follows that $|p_q(t) - p'_r(t)| < \frac{5a_{max}\tau^2}{2}$ for all $t \in [T''_r, T_r]$. Because $|v_q(T''_r) - v_r(T''_r)| \leq 2a_{max}\tau$ and $|v_q(T'_r) - v_r(T'_r)| \leq a_{max}\tau$, it follows that $|v_q(t) - v_r(t)| \leq 2a_{max}\tau$ for all $t \in (T''_r, T_r]$. □

Finally, we remove the restriction that $v_r(0)$ be a multiple of $a_{max}\tau$. We modify Γ_r by prepending a trajectory segment that has an initial velocity that is a multiple of $a_{max}\tau$, ends at $\Gamma_r(0)$, and takes time τ. (This construction is "symmetrical" to the construction in the proof of Corollary 4.7. See Figure 11.) The modified trajectory satisfies the hypotheses of Corollary 4.8.

Corollary 4.9 *Let τ be fixed such that $a_{max}\tau | v_{max}$. Let Γ_r have velocity function v_r obeying dynamics bounds v_{max} and a_{max}.*

Suppose that

$$\begin{aligned} \dot{s}^* &= \text{the multiple of } a_{max}\tau \text{ closest to } v_r(0) \\ s^* &= p_r(0) - \tfrac{\tau}{2}(\dot{v}_r(0) + \dot{s}^*). \end{aligned}$$

Then, there exists a grid-bang Γ_q such that

1. $\Gamma_q(0) = (s^*, \dot{s}^*)$,
2. *$\Gamma_q(\tau)$ approximates $\Gamma_r(0)$ to tolerance $(a_{max}\tau^2, 2a_{max}\tau)$,*
3. *for all $t \in [0, \tau]$, $\Gamma_q(t)$ is within $(v_{max}\tau, 2a_{max}\tau)$ of $\Gamma_r(0)$,*
4. *for all $t \in [0, T_r]$, $\Gamma_q(t+\tau)$ tracks $\Gamma_r(t)$ to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$,*
5. *for all $t \in [0, T_r]$, $\Gamma_q(T_r + \tau)$ approximates $\Gamma_r(T_r)$ to tolerance $(\frac{5a_{max}\tau^2}{2}, 2a_{max}\tau)$, and*
6. *$\Gamma_q(0)$ is computable from a_{max}, v_{max}, τ, and $\Gamma_r(0)$.*

Proof: Suppose Γ_r is a trajectory obeying the hypotheses of the corollary, and suppose $v_r(0)$ is not a multiple of $a_{max}\tau$. Let v_0 be the multiple of $a_{max}\tau$ closest to $v_r(0)$. We can then define a trajectory Γ'_r such that $\Gamma'_r(\tau) = \Gamma_r(0)$:

$$\begin{aligned} v'_r(0) &= v_0, & & \\ v'_r(t) &= v_0 + \tfrac{t}{\tau}(v_r(0) - v_0) & \text{if} \quad & t \in [0, \tau], \\ p'_r(0) &= p_r(0) - \textstyle\int_0^\tau v'_r(t)dt, & \text{and,} & \\ \Gamma'_r(t) &= \Gamma_r(t - \tau) & \text{if} \quad & t \in [\tau, T_r + \tau]. \end{aligned}$$

Clearly, $\Gamma'_r(0) = (s^*, \dot{s}^*)$ as defined in the lemma.

By Corollary 4.8, there is an (a_{max}, τ)-bang trajectory Γ_q such that

1. $\Gamma_q(0) = \Gamma'_r(0)$,
2. Γ_q tracks Γ'_r to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$,

3. $|p_q(T_r+\tau) - p'_r(T_r+\tau)| < \frac{5a_{max}\tau^2}{2}$, and

4. $|v_q(T_r+\tau) - v'_r(T_r+\tau)| < 2a_{max}\tau$.

Since $\Gamma'_r(t) = \Gamma_r(t-\tau)$ for all $t \in [\tau, T_r+\tau]$, this establishes the last three conditions of the corollary. Since $\Gamma_q(0) = \Gamma'_r(0)$ and $\Gamma'_r(\tau) = \Gamma_r(0)$, the first three conditions follow from the a_{max} and v_{max} bounds. □

Proof (of lemma 4.2): We apply Lemma 4.5 and the corollaries that follow it.

Specifically, suppose that a trajectory Γ_r respects acceleration bound a_{max} and velocity bound v_{max} and takes time T_r, and that $a_{max}\tau$ divides v_{max}. Using Corollary 4.9, for any τ such that $a_{max}\tau$ divides v_{max}, we can compute the components $(s_i^*, \dot{s}_i^*)$ of a state $\mathbf{S}^*$ such that for some (a_{max}, τ)-bang trajectory Γ_q beginning at $\mathbf{S}^*$

1. $\Gamma_q(\tau)$ approximates $\Gamma_p(0)$ to tolerance $(a_{max}\tau^2, 2a_{max}\tau)$,

2. $\Gamma_q(t+\tau)$ tracks $\Gamma_r(t)$ to tolerance $(4v_{max}\tau + a_{max}\tau^2, 2a_{max}\tau)$,and

3. $\Gamma_q(T_r+\tau)$ approximates $\Gamma_p(T_r)$ to tolerance $(\frac{5a_{max}\tau^2}{2}, 2a_{max}\tau)$.

Therefore, if $4v_{max}\tau + a_{max}\tau^2 \leq \eta_x$ and $2a_{max}\tau \leq \eta_v$, then $\Gamma_q(t+\tau)$ tracks $\Gamma_r(t)$ to tolerance (η_x, η_v). Thus the choice of τ in equation (13) is sufficient. □

4.3 A δ'_v-Safe ("Also Safe") Grid-Bang Trajectory

Recall that by lemma 4.2, τ is polynomially dependent on η_x and η_v. Applying lemmas 4.2 and 4.1 and choosing β in (12) to maximize the upper bound on τ yields the following theorem:

Theorem 4.10 *Given dynamics bounds a_{max} and v_{max}, obstacles $\mathcal{O}$, scalars $c_0 > 0$ and $c_1 \geq 0$, and positive scalar $\epsilon \leq 1$, let τ be chosen such that $a_{max}\tau$ divides v_{max} and*

$$\tau \leq \frac{c_0\epsilon}{2a_{max}c_1(1-\epsilon) + 5v_{max}}. \tag{18}$$

Then for any start state $\mathbf{S}$ we can choose a root vertex state $\mathbf{S}^*$ such that if Γ_r begins at $\mathbf{S}$, is δ_v-safe, and takes time T_r, there exists a δ'_v-safe trajectory Γ_q with the following properties:

1. *$\Gamma_q(0) = \mathbf{S}^*$,*

2. *$\Gamma_q(\tau)$ approximates $\Gamma_r(0)$ to within $(a_{max}\tau^2, 2a_{max}\tau)$, and*

3. *$\Gamma_q(T_r+\tau)$ approximates $\Gamma_r(T_r)$ to within $(\frac{5a_{max}\tau^2}{2}, 2a_{max}\tau)$.*

Proof: Suppose Γ_r is a $\delta_v(c_0, c_1)$-safe trajectory taking time T_r obeying acceleration bound a_{max}. Then by Lemma 4.1 the choice of a tolerance (η_x, η_v) given in (12) ensures that if a trajectory Γ_q approximately tracks Γ_r to tolerance (η_x, η_v), then the δ'_v-tube induced by Γ_q lies entirely inside the δ_v-tube induced by Γ_r. Since the δ_v-tube induced by Γ_r intersects no obstacles in $\mathcal{O}$, Γ_q is therefore δ'_v-safe. Given the tolerance (η_x, η_v), we use Lemma 4.2 to choose the time-spacing τ and the root vertex $\mathbf{S}^*$ (via Corollary 4.9.)

To get the desired bounds, we must choose β so that using (18) yields a maximal τ as given by (13). Let us therefore define for $\beta \geq 0$

$$\begin{aligned} \tau_x(\beta) &= \tfrac{c_0 \epsilon \beta}{5 v_{max}(c_1(1-\epsilon)+\beta)} \\ \tau_v(\beta) &= \tfrac{c_0 \epsilon}{2 a_{max}(c_1(1-\epsilon)+\beta)} \\ \tau(\beta) &= \min(\tau_x(\beta), \tau_v(\beta)). \end{aligned} \tag{19}$$

By inspection, $\tau_x(0) < \tau_v(0)$, τ_x is monotonically increasing, and τ_v is monotonically decreasing. Thus, $\tau(\beta)$ is maximized when $\tau_x(\beta) = \tau_v(\beta)$. Requiring β to be positive and doing a little computation, we find that $\tau(\beta)$ is maximized when

$$\beta = \frac{2 v_{max}}{5 a_{max}}. \tag{20}$$

Applying either τ_x or τ_v to this β yields the desired τ in (18). Finally, we see that the choice of τ and the closeness of $\Gamma_q(\tau)$ to $\mathbf{S}$ guarantee that the trajectory segment from the root vertex to $\Gamma_q(\tau)$ will be δ'_v-safe. $\square$

4.4 Approximation Goodness and Overall Complexity Bounds

As the algorithm's choices of timestep and root vertex agree with Theorem 4.10 and Corollary 4.9, we have shown the approximation bounds in Theorem 2.1.

To determine the total complexity of the algorithm we must bound the number of (a_{max}, τ)-gridpoints for a Cartesian robot with maximum (L_∞) speed v_{max} in a d-dimensional free-space of diameter l. We so this by simply calculating the number of gridpoints reachable under grid-bang commanded motions from the root vertex state in the absence of obstacles. Let $G_\infty(a_{max}, \tau, v_{max}, l, d)$ denote this bound.

First we consider the $d = 1$ case. It is clear that $G_\infty(a_{max}, \tau, v_{max}, l, 1)$ is equal to the maximum number of possible velocities at any given time $k\tau$ multiplied by the maximum number of possible positions at that time. Since at each timestep the change in velocity is $a_{max}\tau$, $-a_{max}\tau$, or 0, the number of possible velocities is at most $\frac{2v_{max}}{a_{max}\tau} + 1$. To see that the number of possible positions at a given velocity is at most $\frac{l}{a_{max}\tau^2} + 1$, let $v^{(i)}$ denote the velocity and $x^{(i)}$ the position at timestep i for any i. Then $x^{(k+1)} = v^{(k)}\tau + \sigma(k)\frac{a_{max}\tau^2}{2} + x^k$, where $\sigma(k) \in \{-1, 0, 1\}$. *Wlog*, let $v^{(0)} = 0$ and $x^{(0)} = 0$. Since $v^{(k)} = c^{(k)} a_{max}\tau$ for some integer $c^{(k)}$, by using induction we can show that

$$
\begin{aligned}
x^{(k)} &= \tfrac{(2\Upsilon^{(k)}+1)a_{max}\tau^2}{2} && \text{if } c^{(k)} \text{ odd} \\
x^{(k)} &= \tfrac{2\Upsilon^{(k)}a_{max}\tau^2}{2} && \text{if } c^{(k)} \text{ even,}
\end{aligned}
\tag{21}
$$

for some integer $\Upsilon^{(k)}$. It follows that

$$
G_\infty(a_{max}, \tau, v_{max}, l, d) = \left(\left(\frac{2v_{max}}{a_{max}\tau} + 1 \right) \left(\frac{l}{a_{max}\tau^2} + 1 \right) \right)^d . \tag{22}
$$

Hence, in a bounded workspace with velocity limits, we can use a polynomial-sized reachability graph to find an approximate optimal safe solution. Given a problem for which a δ_v-safe solution exists, the algorithm finds a trajectory Γ_q that satisfies the time approximation $T_q \leq T_{opt}$, in addition to respecting the kinodynamic constraints and being δ'_v-safe. Noting that the maximum out-degree of the reachability graph is 3^d and that checking the safety of trajectory segments requires time $O(N)$, we substitute the algorithm's choice of τ (18) into (22) to complete obtain the complexity bound in Theorem 2.1.

5 Implementation, Discussion, and Extensions

5.1 Implementation Results

To get some idea for how our algorithm might behave in practice, we have completed a COMMON LISP implementation for a point robot in two dimensions obeying L_∞ dynamics bounds. This planner, which we have run on a Sun SparcStation2, is the first implementation[5] of any algorithm that generates provably good, provably near-optimal kinodynamic plans for problems in more than one dimension.

Our planner is a simple implementation of the algorithm and uses no search-pruning heuristics. It basically does a breadth-first graph search of the TC-grid, computing a state's (vertex's) neighbors only when it is on the search frontier. A bit array is used to record whether a state has been reached, and each vertex found keeps a pointer to its "parent". The planner implements collision-avoidance and δ'_v-safety, so each trajectory segment is δ'_v-safe.

An example of a solution found by our planner is shown in Figure 12. The start position is at lower left, and the start velocity is in the positive y direction; the goal, at the upper right, has velocity in the negative y direction. The small dots indicate the position component of the planned trajectory at each time step. The line segments attached to each circle in Figure 12(a) indicate the velocity at that position. The longer segments in Figure 12(b) indicate the acceleration direction. In this problem, the world is 3.5 by 1.9 units; $v_{max} = 0.12$; $a_{max} = 0.1$; $c_0 = 0.31$; $c_1 = 0.1$; and $\epsilon = 0.8$. τ was chosen as 0.4, resulting of a TC-grid of approximately 800,000 states. The implementation searched over 740,000 states

[5] Actually, it's the second incarnation of our first implementation.

and required approximately three hours of actual elapsed time, which includes swapping, to solve this problem; CPU run time was consistently under an hour.

5.2 Discussion

The careful reader might observe that, in general, there may be many minimal-time (a_{max}, τ)-bang trajectories from $\mathbf{S}$ to a given TC-gridpoint, and the algorithm as described in previous sections might find any one of these. These trajectories can differ from each other in many properties, such as homotopy class, maximum curvature, and average speed. Certain secondary performance measures can be used to choose among these during each round, but this is unrelated to our theoretical result and does not affect the number of states visited. This idea is explored in [3], and Figure 13 illustrates the effect of a different choice.

Despite the slowness of the our current implementation, we do not believe that the algorithm is inherently impractical. First, if we allow solutions that take at most time $(1+\epsilon)T_{opt}$ instead of T_{opt}, where T_{opt} is the time for an optimal kinodynamic trajectory, we can increase the timestep size. This single change in the algorithm dramatically reduces the size of the reachability graph and the running time. (See Figure 14.) The analysis in [3, 23] closely parallels the one described here. Finally, we note that because of the particular graph-search nature of the algorithm, we could greatly exploit parallelism in a practical implementation; additional, though limited, parallelism can be extracted in safety-checking.

5.3 Extensions

Our results can be directly extended in several ways.

Via a transformation to configuration space [24], our results can be applied to a rigid, non-rotating robot whose geometry is given by a union $\mathcal{R}$ of convex polyhedra. This configuration space transformation has been discussed extensively in the literature (see, e.g., [24]). The algorithm of [24] could be used as a preprocess to reduce the planning problem for $\mathcal{R}$ amidst obstacles $\mathcal{O}$ to the point navigation problem we have discussed.

Since the Safe Tracking Lemma (4.1) and Strong Tracking Lemma (4.2) are independent of spatial dimension, the main theorem can be extended to arbitrary d, with additional complexity resulting from the increased cost of safety-checking and d-dimensional arithmetic. In particular, safety-checking would involve collision detection with the Minkowski sum of convex d-polytopes and a speed-parameterized d-cube. (See Appendix B for the 3-dimensional case.) Letting $p(N, d)$ be the complexity of this procedure, the extended main result would be have cost $O\left(c^d p(N,d)\left(\frac{1}{\epsilon}\right)^{3d}\right)$. The details of the result appear to hinge on bounding the number of $(d-1)$-faces (since a $(d-2)$-face bounds exactly two $(d-1)$-faces) of the Minkowski sum and constructing their incidence relation. We conjecture that applying and extending work from computational geometry such as [25, 26] would be fruitful.

In turn, this extended d-dimensional algorithm would apply, via robot-dependent constant linear transforms, to other robots with constant, decoupled dynamics equations and decoupled dynamics, when the configuration space obstacles are expressed as unions of convex polytopes. For such robots with revolute and translational degrees of freedom and polyhedral *workspace* obstacles, the only change in the algorithm would again be in the safety-checking step. For a description of the modified safety-checking step, which extends [27] and [28], see our companion paper [7] or [3], which present our results for robots with coupled dynamics.

Finally, the [1] approach and its descendents reduce the problem of finding an approximately optimal trajectory to that of finding the shortest path between two vertices of a uniform-cost graph whose vertices correspond to system states. On such graphs the single-source or single-sink shortest path problem can be solved with nearly the same asymptotic time-complexity as finding a path between two vertices. For a k-source or k-sink problem, the complexity of the algorithm is $O(c^d(N+k)(\frac{1}{\epsilon})^{3d})$, with the increased cost resulting from checking closeness to the sources or sinks.

6 Conclusions

In this paper we obtain a provably good approximation algorithm for kinodynamic planning that extends the results of [1]. We modify their algorithm for Cartesian kinodynamic planning under L_∞ dynamics bounds and apply new analysis techniques. We are able to (a) tighten the complexity bound to $O\left(c^d N \left(\frac{1}{\epsilon}\right)^{3d}\right)$, or $O\left(N \left(\frac{1}{\epsilon}\right)^{3d}\right)$ for a fixed robot and world diameter, from $O\left(N^2 \log N \left(\frac{1}{\epsilon}\right)^{6d}\right)$, and (b) show that our algorithm finds a trajectory taking at most time T_{opt} (the optimal time) instead of time $T_{opt}(1+\epsilon)$. In addition, the approximation closeness at the start and goal states is not affected by the velocity bounds.

We have reported on a preliminary implementation. This is the first implementation of a polynomial-time, provably good approximation algorithm for kinodynamic planning. While the current implementation runs slowly, eventually an improved implementation may be reasonable for practical off-line motion planning. Finally, we have described probable, if not definite, direct extensions to our work.

There are many additional directions for future research, among these:

1. Reducing "practical complexity" of our algorithms, for example using heuristically-assisted search techniques instead of breadth-first search.

2. Precise lower bounds for kinodynamic planning should be established (especially in the 2D case).

3. Since the structure of the reachability-graph can be computed "locally", there is hope for a parallel implementation, and this should be investigated.

4. We conjecture that if contact is allowed (rather than δ_v-safety) then the complexity of the problem increases considerably.

Kinodynamic planning represents a new direction in algorithmic motion planning. Computational kinodynamics seems a particularly fruitful area in which to pursue provably fast, provably good approximation algorithms, since while the problems are of considerable intrinsic interest, exact solutions may well be intractable. We have presented results with the lowest known complexity for Cartesian kinodynamic planning in two and three dimensions. For our corresponding results for robots having less restricted dynamics and dynamics bounds, we encourage the reader to see our companion paper [7].

Appendices

A Kinodynamic Planning Lower Bounds

This appendix sketches how to extend Canny's and Reif's proof [4] of the $\mathcal{NP}$-hardness of the 3D shortest-path problem to show that Optimal Cartesian Kinodynamic Planning in 3D is also $\mathcal{NP}$-hard. This claim was made in [1], but without proof. While the general description here should convey the flavor of the result, the technical specifics are intended to accompany or follow a close reading Section 2 of [4]. The discussion also includes a brief justification of our statement from Section 2.3 that finding a path that is homotopic to the shortest is $\mathcal{NP}$-hard.

The idea for the (main) extension is simple: if we could set the acceleration bound a_{max} to ∞, the reduction would be trivial because the problems would be identical. However, to do a polynomial-time reduction, we must show that for a given shortest-path problem instance we can find a "sufficiently large" a_{max} that only requires a polynomial number of bits to encode.

A.1 Canny's and Reif's Proof

A.1.1 A Reduction

To prove that the 3D shortest-path problem for a point among polyhedral obstacles is $\mathcal{NP}$-hard, [4] give a polynomial-time reduction from 3SAT. They describe how to take a given instance of 3SAT and construct an instance of the polyhedral path-planning problem such that a shortest path from the start position to the goal position can be used to determine whether the 3CNF formula in the instance is satisfiable. The proof considers a 3CNF formula F of m clauses $C_1, \ldots, C_m$ over n variables $b_1, \ldots, b_n$ and their negations. The polyhedral environment constructed by the reduction has a description of length $O(mn)$. The construction allows one to compute (using $O(mn)$ bits) a path length *lower bound* l and an accuracy δ_{ac} such that the shortest path between the start and goal positions has length greater than $l + \delta_{ac}$ if and only if the formula F is not satisfiable.[6]

A.1.2 Correspondence Between Homotopy Classes

The polyhedral environment will be the interior of a square box separated into levels by flat plates, each having one, two, or three slits. The plate thickness, slit width, and plate separation will all have the same size ϵ_{CR}. In addition, internal walls will divide the spaces between certain plates into rooms having one slit on the "ceiling" and one slit on the "floor". The start position will be above the top plate, and the goal below the lowest.

[6] We mostly follow the notation of [4] in this section. In particular, "l" is used as in [4], not as in the rest of this paper; i.e., it does *not* mean "world diameter". δ_{ac} is not found in [4], but it just takes the place of a more complicated term.

The environment is constructed so that it will have special properties that we describe, avoiding details as much as possible. Recall that each clause C_i in a 3CNF formula is a disjuction of literals $L_{i1} \vee L_{i2} \vee L_{i3}$. If we separate paths that go from the start to the goal without ever visiting a level twice into their homotopy equivalence classes, then each class will encode 1) a truth assignment for $b_1, \ldots, b_n$ and 2) the evaluation of a conjunction of m literals, one from each clause of F, using this truth assignment. In other words, if F' is the disjunctive normal form of F obtained by distribution, then for each term (conjunction of literals) in F' and each truth assignment, there will be a corresponding homotopy class. If the truth assignment fails to satisfy the term formula, then some obstacle will "stretch" the shortest path in the class. Furthermore, this path will have a length greater than than $l + \delta_{ac}$ if and only if the truth assignment fails to satisfy the term formula. Finally, F is satisfiable if and only if the shortest path from start to goal will not be "stretched".

Observation. The careful reader of [4] will note that for any of these homotopy equivalence classes, it can be easily calculated whether the shortest path in that class will be "stretched" in this manner. Thus, given a 3CNF formula F, we could construct the corresponding polyhedral shortest path problem, and once we obtain the homotopy class of the shortest path from start to goal, we could determine whether F is satisfiable. *Thus, the construction in [4] can be used to show that finding the homotopy equivalence class of the shortest path in a polyhedral environment is $\mathcal{NP}$-hard.*

A.1.3 Bit-Counting

To show their reduction is polynomial-time, [4] must show that the number of bits necessary to encode the corresponding shortest-path problem instance will be polynomial in n and m. The main problem is to show that ϵ_{CR} does not have to be too small. Suppose that the construction procedes for a 3CNF formula F. Let $l_{unstretched}$ denote the length of the actual shortest path if $F \in$ *3SAT*, and $l_{stretched}$ if not.

Again, modulo some details, we can view their analysis as having three steps, showing that:

1. If the slit width and plate thickness were zero, the construction guarantees that for path length lower bound $l = 2^{3n}$ and another parameter, which [4] call the minimum virtual source spacing, $\delta_{min} = 2^{-nm}$:

$$\begin{aligned} l_{unstretched} &= l, \quad \text{and} \\ l_{stretched} &\geq l + \frac{\delta_{min}^2}{4l}. \end{aligned}$$

   (Equation (9) in [4].)

2. The total number of plates is bounded from above by

$$7nm + 10n + 2m + 4.$$

Furthermore, each plate can add no more than ϵ_{CR} to the shortest path length, so that if $F \in \mathit{3SAT}$, then

$$l \leq l_{unstretched} \leq l + (7nm + 10n + 2m + 4)\epsilon_{CR}.$$

(Equation (10).)

3. It follows that if

$$\frac{\delta_{min}^2}{4l} > (7nm + 10n + 2m + 4)\epsilon_{CR},$$

which is guaranteed if $\epsilon_{CR} = \frac{2^{-2nm-3n-3}}{(7nm+10n+2m+4)}$, then the reduction works with $\delta_{ac} = \frac{\delta_{min}^2}{4l}$.

Thus, ϵ_{CR} can be specified in $O(nm)$ bits. It then follows from other arguments that any dimension in the environment can be specified in $O(nm)$ bits.

A.2 The Extension for Kinodynamic Lower Bounds

A.2.1 The Idea

To extend the $\mathcal{NP}$-hardness result to kinodynamic planning, we can use essentially the same polynomial-time reduction to reduce an instance of 3SAT to an optimal kinodynamic planning problem instance. Hence, we would use a point amidst polyhedral obstacles. We will also choose the units for time and distance such that they convert trivially. For a given 3CNF formula F, the new reduction constructs the a kinodynamic planning problem instance identical to the path-planning problem instance in the [4] reduction except that

1. the slit width and plate thickness ϵ'_{CR} will be smaller than ϵ_{CR};
2. the start and goal positions will be lifted to the corresponding states with velocity zero;
3. the velocity must obey a unit bound (in the same L_p-norm used for distance in the shortest-path problem);
4. the acceleration will obey some bound a_{max} (in the same norm); and
5. for lower time bound l and time-accuracy δ_T, which we introduce, the minimal-time kinodynamic solution (obstacle-avoiding trajectory) would take more than time $l + \delta_T$ if and only if the given 3CNF formula F is not satisfiable.

The specification of δ_v-safety parameters c_0 and c_1 will be omitted from the discussion; it is easy to choose them, because they can be incorporated linearly into the choices of slit widths and plate separations.

A.2.2 Bit-Counting Again

Similarly to the above, the minimal-time solution will be "stretched" if and only if $F \notin 3SAT$. Let $T_{stretched}$ be the minimal solution time in this case, $T_{unstretched}$ be the least time otherwise. Suppose that we choose times $T_l = l$ and $\delta_T = \delta_{min}$ via unit conversion. It follows the analysis in [4] that if the slit width and plate thickness were zero, and if the acceleration bound were infinite, then

$$\begin{aligned} T_{unstretched} &= l, \quad and \\ T_{stretched} &\geq l + \frac{\delta_{min}^2}{4l}. \end{aligned}$$

With slit width and plate thickness ϵ'_{CR} but infinite acceleration, it follows that

$$l \leq T_{unstretched} \leq l + (7nm + 10n + 2m + 4)\epsilon'_{CR}.$$

Now, if the acceleration bound is a_{max}, $T_{unstretched}$ can increase by at most $\frac{2}{a_{max}}$ for each obstacle edge along the shortest path, since this is twice the amount of time it takes to go from zero velocity to full speed or vice-versa. Since there will be at most two edges along the shortest path for each plate,

$$l \leq T_{unstretched} \leq l + (7nm + 10n + 2m + 4)\left(\epsilon'_{CR} + \frac{4}{a_{max}}\right).$$

By similar reasoning as above, if

$$\begin{aligned} \epsilon'_{CR} &= \frac{2^{-2nm-3n-4}}{(7nm + 10n + 2m + 4)}, \quad \text{and} \\ a_{max} &= 2^{2nm+3n+6}(7nm + 10n + 2m + 4), \end{aligned}$$

then the reduction will work with $\delta_T = \frac{\delta_{min}^2}{4l}$. (Recall that $\delta_{min} = 2^{-nm}$.) Thus, ϵ'_{CR} and a_{max} can be specified in $O(nm)$ bits.

B Safety Checking

We describe how to check whether an (a_{max}, τ)-bang violates the speed-dependent safety-margin $\delta_v(c_0, c_1)$. The procedure runs in $O(N)$ time, where N is geometric complexity of the configuration space obstacles. We review some basic computational geometry developed by [24], describe the special case when $c_1 = 0$, and then extend the method to the general case.

Assume that the configuration space obstacles are the union of convex polyhedra, and recall that $d \leq 3$. For now, let the safety margin be a constant $c_0 > 0$, and define the B_{c_0} to be the L_∞ ball with radius c_0. Staying c_0-safe relative to a convex polyhedron A is then equivalent to avoiding $\overline{A} = A \oplus B_{c_0}$, where "$\oplus$" denotes the Minkowski sum. Let A have

m faces. Then, since B_{c_0} is a d-cube and $d \le 3$, $\overline{A}$ is also a convex polyhedron and has $O(m)$ faces. By taking the Minkowski sum of each of the obstacles with B_{c_0} we obtain the *expanded obstacles*.

Suppose $\overline{A}$ has faces $\{ F_0, \ldots, F_m \}$ lying on the boundary planes of the closed half-spaces $\{ H_0, \ldots, H_m \}$. The boundary plane of each H_i is the kernel of an affine function f_i. If $\mathbf{n}_i$ is a unit vector in the outward normal direction from the boundary plane of H_i, and $\mathbf{y}_i$ is any point on this boundary then

$$f_i(\mathbf{x}) = \langle \mathbf{n}_i, \mathbf{x} \rangle - \langle \mathbf{n}_i, \mathbf{y}_i \rangle.$$

The polyhedron $\overline{A}$ is thus described by a set of functions $\mathcal{F} = \{ f_0, \ldots, f_m \}$.

A point $\mathbf{x}$ is on the boundary of $\overline{A}$ if and only if it lies on some the closure of some face F_k of $\overline{A}$. Equivalently, $f_k(\mathbf{x}) = 0$, and for all f_j that determine an edge of F_k, $f_j(\mathbf{x}) \le 0$. Since for a convex polyhedron the numbers of edges and faces are linearly related and an edge is common to exactly two faces, determining whether $\mathbf{x}$ lies on the boundary of any of the expanded of obstacles takes total time $O(N)$.

Wlog suppose that (a, τ)-bang $\mathbf{p}$ begins at $t = 0$ and that $\mathbf{p}(0)$ is c_0-safe. We then can check the c_0-safety of $\mathbf{p}(t)$ by determining whether $\mathbf{p}(t)$ intersects the boundary of an expanded obstacle. For a face F_k of $\overline{A}$, we only need to solve $f_k(\mathbf{p}(t)) = 0$, and for each solution t_s check whether $f_j(\mathbf{p}(t_s)) > 0$ for some f_j that determines an edge of F_k with f_k.

To check whether an (a_{max}, τ)-bang is $\delta_v(c_0, c_1)$-safe relative to A for a given $c_1 > 0$, we simply "lift" the obstacles from C to TC. A state $(\mathbf{x}, \mathbf{v})$ is δ_v-safe relative to a convex polyhedron A if and only if it lies outside the expanded obstacle $\widehat{A}(\mathbf{v}) = A \oplus B_{\delta_v}(\mathbf{v})$, where $B_{\delta_v}(\mathbf{v})$ is the L_∞ ball with radius $\delta_v(\mathbf{v})$. Thus, a statespace obstacle $\widehat{A}$ is described similarly to $\overline{A}$, by a set of functions $\widehat{\mathcal{F}} = \{ \widehat{f}_0, \ldots, \widehat{f}_m \}$. For each $f_i \in \mathcal{F}$, we define

$$\widehat{f}_i(\mathbf{x}, \mathbf{v}) = \langle \mathbf{n}_i, \mathbf{x} \rangle - \langle \mathbf{n}_i, \mathbf{y}_i + \mathbf{q}_i \|\mathbf{v}\|_\infty \rangle,$$

where $\mathbf{q}_i$ is a constant vector that depends only on $\mathbf{n}_i$. $(\mathbf{x}, \mathbf{v}) \in \widehat{A}$ if and only if $f_i(\mathbf{x}, \mathbf{v}) \le 0$ for each i.

We use the $\widehat{f}_i \in \widehat{\mathcal{F}}$ the same way we use the $f_i \in \mathcal{F}$ above. Specifically, assume that $(\mathbf{p}(0), \dot{\mathbf{p}}(t))$ is δ_v-safe, and for each $f_i \in \mathcal{F}$ define

$$\widetilde{f}_i(t) = \langle \mathbf{n}_i, \mathbf{p}(t) \rangle - \langle \mathbf{n}_i, \mathbf{y}_i + \mathbf{q}_i \|\dot{\mathbf{p}}(t)\|_\infty \rangle.$$

An (a_{max}, τ)-bang $(\mathbf{p}, \dot{\mathbf{p}})$ intersects $\widehat{A}$ if and only if there is some $t \in (0, \tau]$ and some face F_k of $\overline{A}$ such that

$$\begin{array}{rcl} \widetilde{f}_k(t) & = & 0, \text{ and} \\ \widetilde{f}_j(t) & \le & 0 \quad \forall f_j \text{ that determine an edge of } F_k. \end{array}$$

Since $\mathbf{p}(t)$ is quadratic, $\widehat{f}_k(t)$ has zeros of the form $t = a \pm \sqrt{b}$. When computing the inequalities we can square twice to eliminate the radical, and thus it is possible to compute square roots symbolically. This implies that safety checking never requires numbers longer (in the number of bits) than a constant-multiple of the length of the longest number in the input, so taking account of the bit complexity of safety-checking does not raise our complexity bounds from those obtained using the real-RAM model. Therefore, since we need to solve $O(N)$ equations and check $O(N)$ inequalities, the cost of safety-checking is $O(N)$ per (a, τ)-bang.

Footnotes

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 grants No. IRI-8802390, IRI-9000532 and by a Presidential Young Investigator award, and in part by the Mathematical Sciences Institute, Intel Corporation, and AT&T Bell laboratories.

2. [2] have recently provided an exact algorithm for the 2D L_∞ case. The algorithm runs in exponential time and polynomial space.

3. [5] is the journal revision of [1].

4. In cases when $v_r(\tau) - \left\lfloor \frac{v_r(n\tau)}{a_{max}\tau} \right\rfloor a_{max}\tau = \frac{a_{max}}{2}$, the floor or ceiling must be chosen consistently.

5. Actually, it's the second incarnation of our first implementation.

6. We mostly follow the notation of [4] in this section. In particular, "l" is used as in [4], not as in the rest of this paper; i.e., it does *not* mean "world diameter". δ_{ac} is not found in [4], but it just takes the place of a more complicated term.

References

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

[2] J. Canny, A. Rege, and J. Reif. An exact algorithm for kinodynamic planning in the plane. In *Proceedings of the Sixth Annual Symposium on Computational Geometry*, pages 271–280, Berkeley, California, 1990.

[3] P. Xavier. Provably-good approximation algorithms for optimal kinodynamic robot motion plans. CS Dept. Tech. Rep. CUCS-TR92-1279, Cornell University, Ithaca, New York, April 1992. Ph.D. Thesis.

[4] J. Canny and J Reif. New lower bound techniques for robot motion planning. In *Proceedings of the* 28^{th} *Annual Symposium on the Foundations of Computer Science*, Los Angeles, California, 1987.

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

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

[7] B. Donald and P. Xavier. Provably good approximation algorithms for optimal kinodynamic planning for cartesian robots and open-chain manipulators. Companion paper in *Algorithmica,* ? 199?

[8] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Englewood Cliffs, New York, 1982.

[9] C. H. Papadimitriou. An algorithm for shortest path motion in three dimensions. *Information Processing Letters*, 20:259–263, 1985.

[10] M. Brady, J. Hollerbach, T. Johnson, T. Lozano-Pérez, and M. Mason, editors. *Robot Motion: Planning and Control.* MIT Press, Cambridge, Massachusetts, 1982.

[11] C. Yap. Algorithmic motion planning. In J. Schwartz and C. Yap, editors, *Advances in Robotics: Volume 1.* Lawrence Erlbaum Associates, 1986.

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

[13] J. Bobrow, S. Dubowsky, and J. Gibson. Time-optimal control of robot manipulators along specified paths. *International Journal of Robotics Research*, **4**(3), 1985.

[14] 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.

[15] E. Sontag and H. Sussmann. Remarks on the time-optimal control of two-link manipulators. In *Proceedings of the* 24^{th} *Conference on Decision and Control*, Ft. Lauderdale, Florida, 1985.

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

[17] G. Sahar and J. Hollerbach. Planning of minimum-time trajectories for robot arms. In *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, pages 751–758, St. Louis, Missouri, 1985.

[18] Z. Shiller and S. Dubowsky. Global time-optimal motions of robotic manipulators in the presence of obstacles. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 370–375, Philadelphia, Pennsylvania, 1988.

[19] C. Ó'Dúnlaing. Motion planning with inertial constraints. *Algorithmica*, **2**(4):431–475, 1987.

[20] S. Fortune and G. Wilfong. Planning constrained motion. In *Twentieth Annual ACM Symposium on Theory of Computing*, Chicago, 1988.

[21] G. Wilfong. Motion planning for an autonomous vehicle. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 529–533, Philadelphia, Pennsylvania, 1988.

[22] P. Jacobs and J. Canny. Planning smooth paths for mobile robots. In *Proceedings ot the 1989 IEEE International Conference on Robotics and Automation*, pages 2–7, Scottsdale, Arizona, 1989.

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

[24] T. Lozano-Pérez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, **C–32**(2):108–120, 1983. Also MIT A.I. Memo 605, December 1982.

[25] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* EATCS 10. Springer-Verlag, Berlin, Germany, 1987.

[26] L. Guibas and R. Seidel. Computing convolutions by reciprocal search. In *Proceedings* 4^{th} *ACM Symposium on Computational Geometry*, pages 90–99, 1988.

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

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

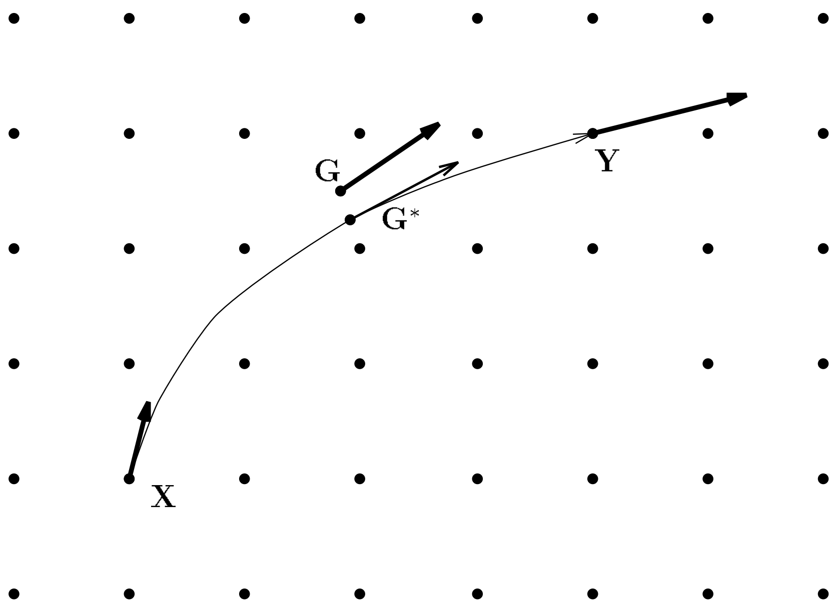


Figure 4: State $\mathbf{G}^*$ on the trajectory of the (a_{max}, τ)-bang from $\mathbf{X}$ to $\mathbf{Y}$ lies much closer to the goal $\mathbf{G}$ than either $\mathbf{X}$ or $\mathbf{Y}$ do.

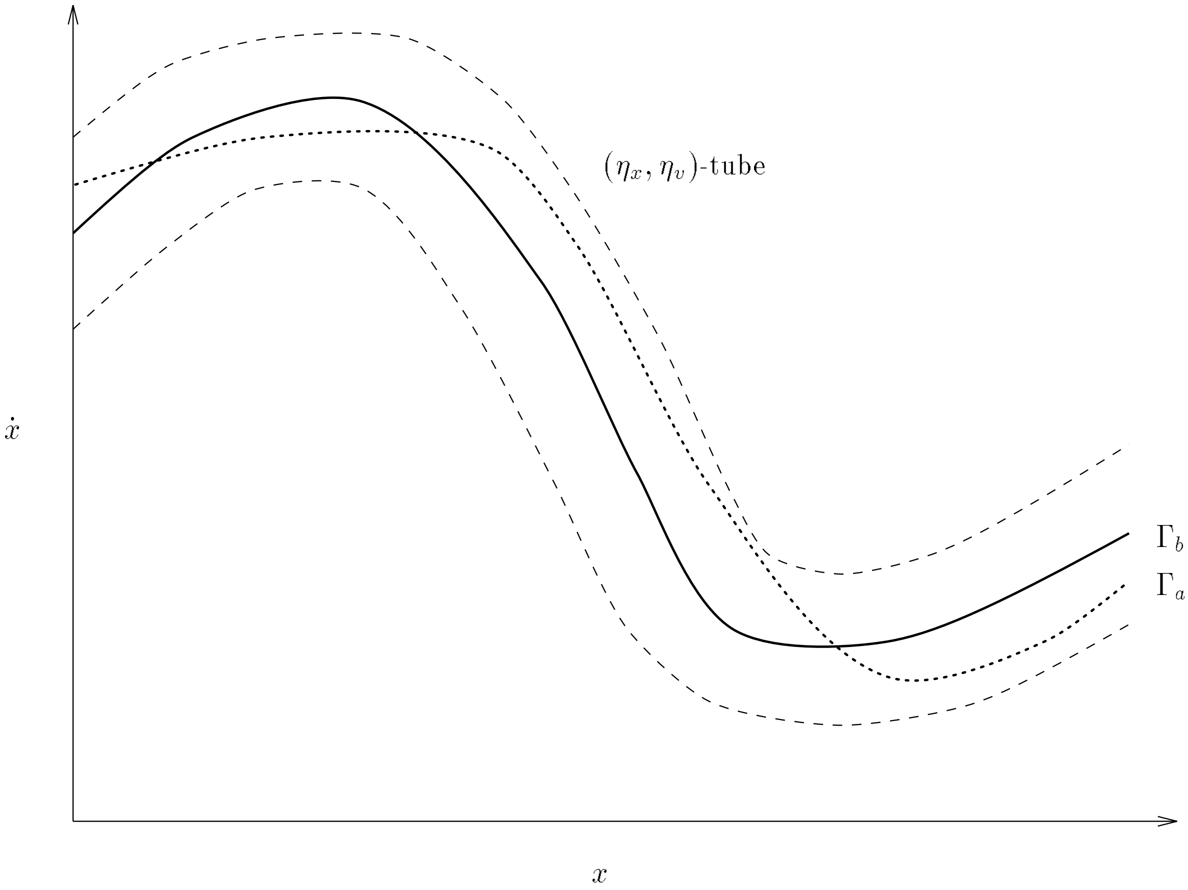


Figure 5: An (η_x, η_v)-tube projected into phase-space. Trajectory Γ_a approximately tracks Γ_b to tolerance (η_x, η_v).

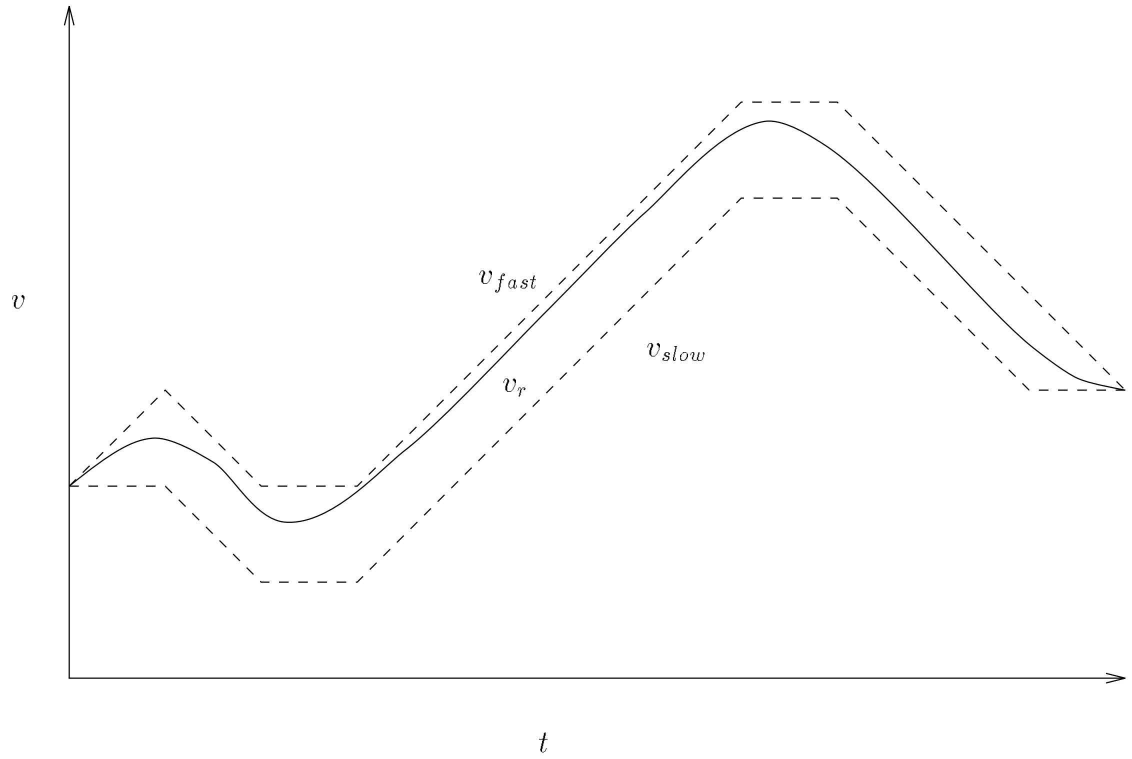


Figure 6: First step in proving Lemma 4.2. Assume that trajectory Γ_r obeys the velocity and acceleration bounds v_{max} and a_{max}. In addition, for now assume that $T_r = K\tau$ for some integer K, and that v_{max}, $v_r(0)$ and $v_r(T_r)$ are multiples of $a_{max}\tau$. Then v_r is bounded above and below by (a_{max}, τ)-bang velocity functions v_{fast} and v_{slow}, which stay within $2a_{max}\tau$ of v_r.

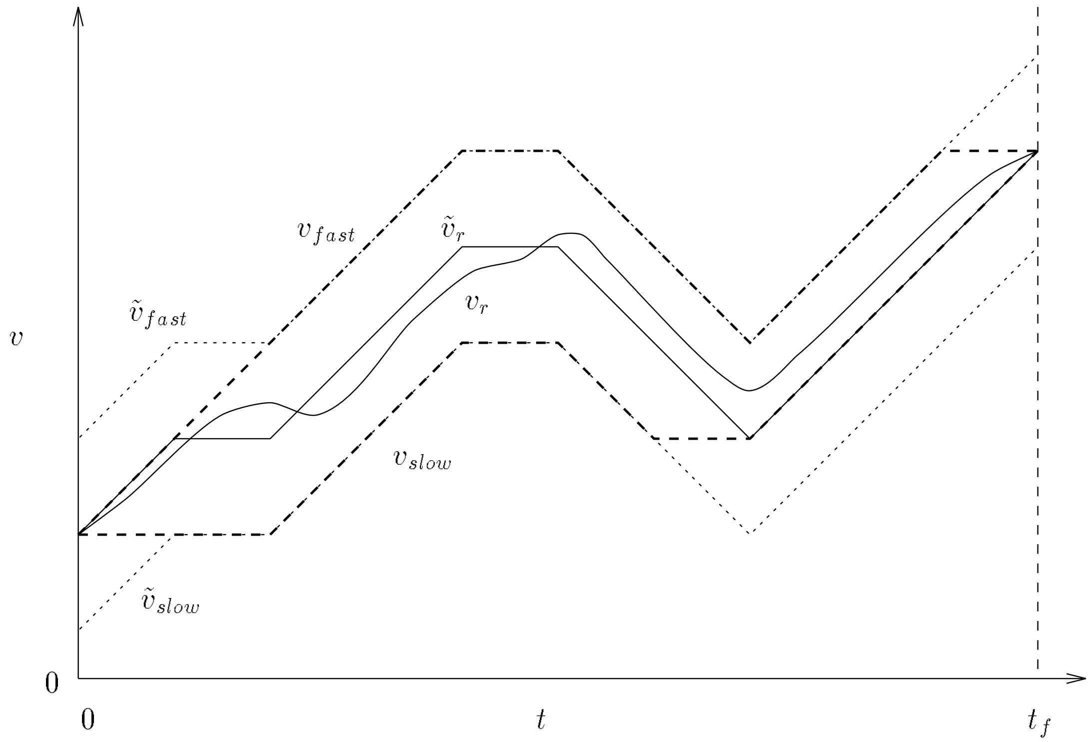


Figure 7: The construction for Lemma 4.3. $\tilde{v}_r$ matches v_r at $t = 0, T_f$. v_{fast} and v_{slow} are visible as dashed lines where they differ from $\tilde{v}_{fast}$ and $\tilde{v}_{slow}$, respectively.

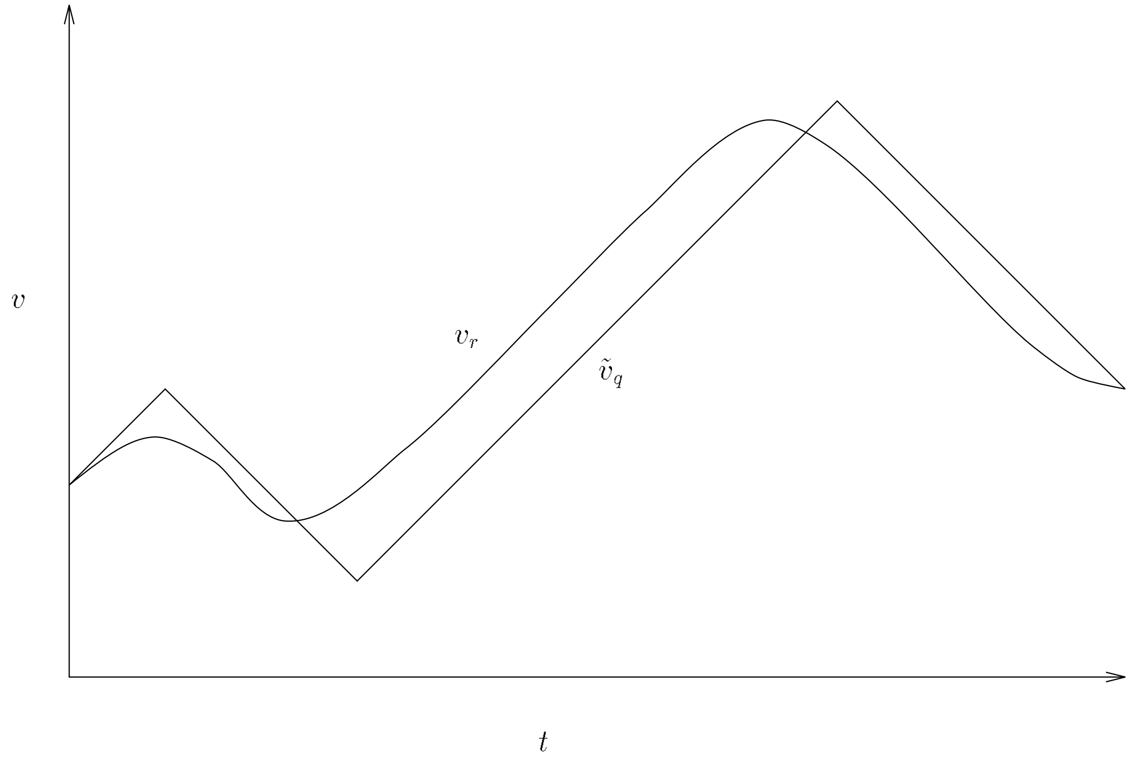


Figure 8: Second step in proving Lemma 4.2. We constructively show the existence of an (a_{max}, τ)-bang trajectory that tracks Γ_r. $\tilde{p}_q(0) = p_r(0)$, and $|p_r(t) - \tilde{p}_q(t)| \leq 2v_{max}\tau + a_{max}\tau^2$. $\tilde{v}_q$ is bounded above and below by (a_{max}, τ)-bang velocity functions v_{fast} and v_{slow}, which stay within $2a_{max}\tau$ of v_r.

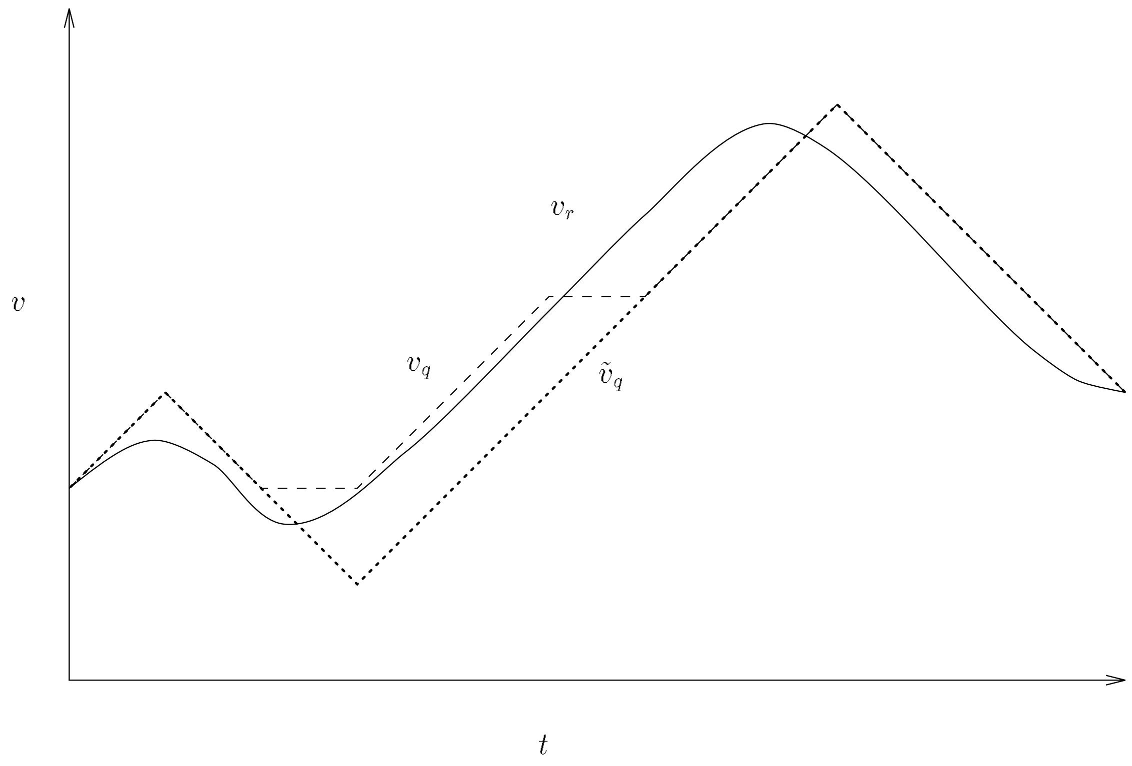


Figure 9: Third step in our proof of Lemma 4.2: $\tilde{v}_q$ (from Lemma 4.4) is modified to obtain v_q such that $\int_0^{t_r} v_q(t)dt$ is within $a_{max}\tau^2$ of $\int_0^{t_r} v_r(t)dt$.

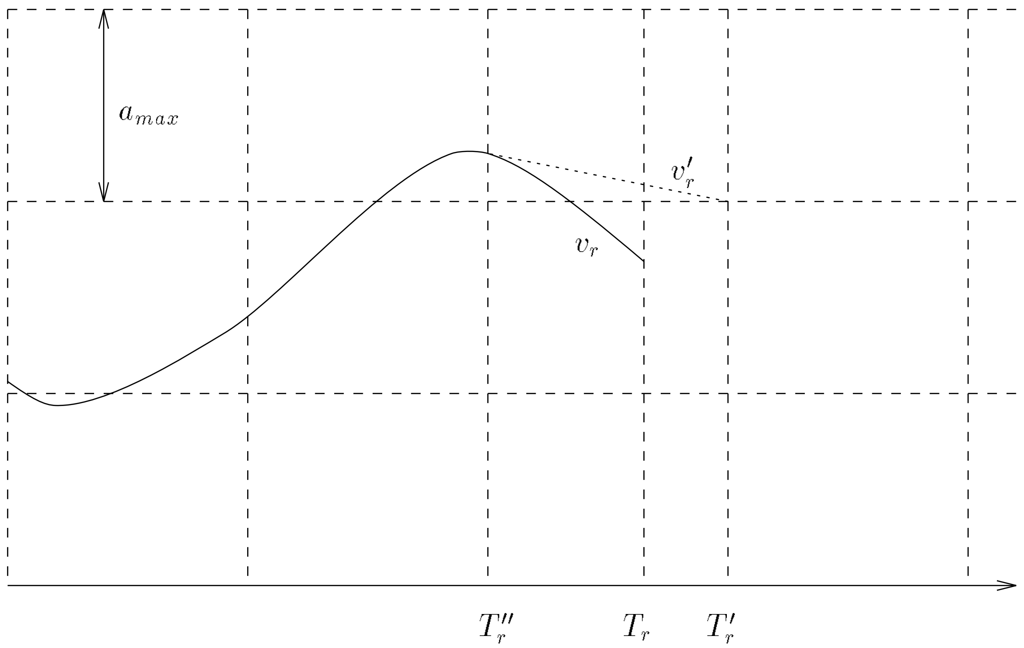


Figure 10: Proving Lemmas 4.6 through 4.8: if $a_{max}\tau$ does not divide $v_r(T_r)$, then there is some a_{max}-bounded v_r' such that $v_r'(t) = v_r(t)$ for $t \in [0, T_r'']$ and $a_{max}\tau | v_r'(T_r')$, where $T_r' = \lceil \frac{T_r}{\tau} \rceil$ and $T_r'' = T_r' - \tau$. If $p_r(0) = p_r'(0)$, then a trajectory that tracks Γ_r' also tracks Γ_r almost as closely.

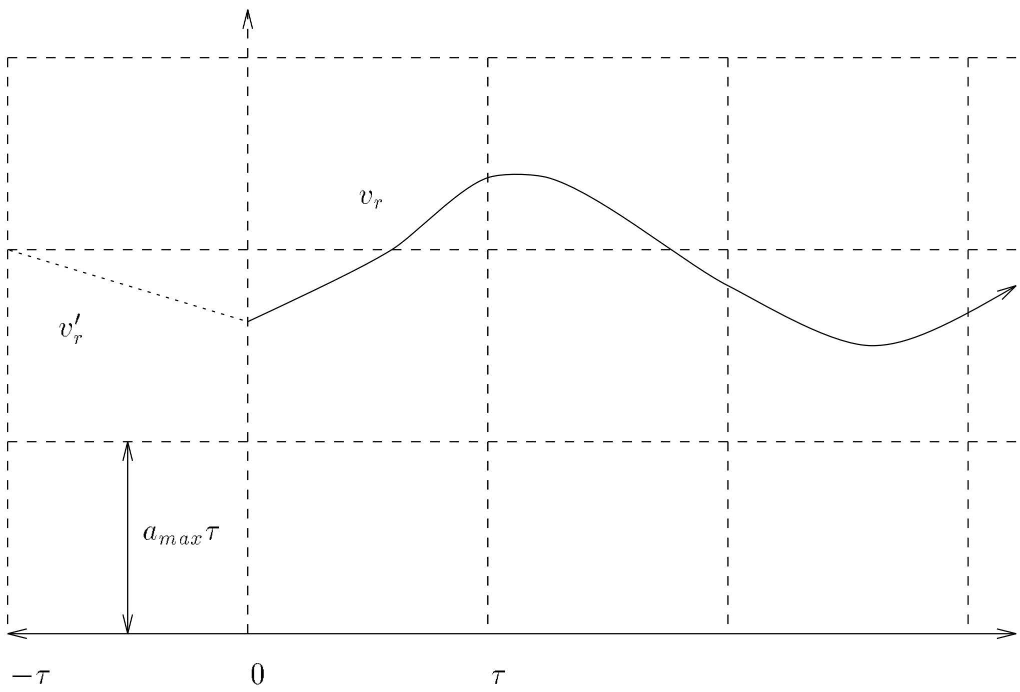
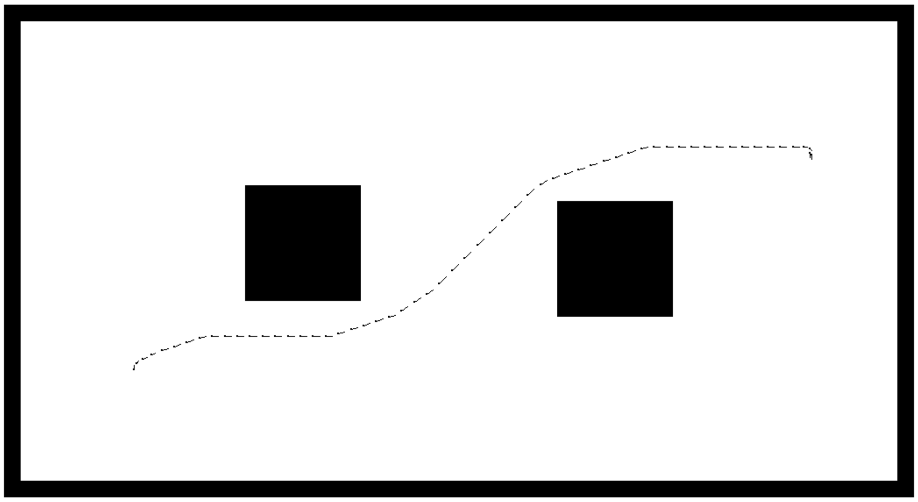


Figure 11: Proving Lemma 4.9: if $a_{max}\tau$ does not divide $v_r(0)$, then there is some a_{max}-bounded v'_r such that $v'_r(t+\tau) = v_r(0)$ for $t \in [0, T_r]$ and $a_{max}\tau$ divides $v'_r(0)$. Here the time-scale for v'_r is offset by $-\tau$ to show this relationship.

(a)



(b)

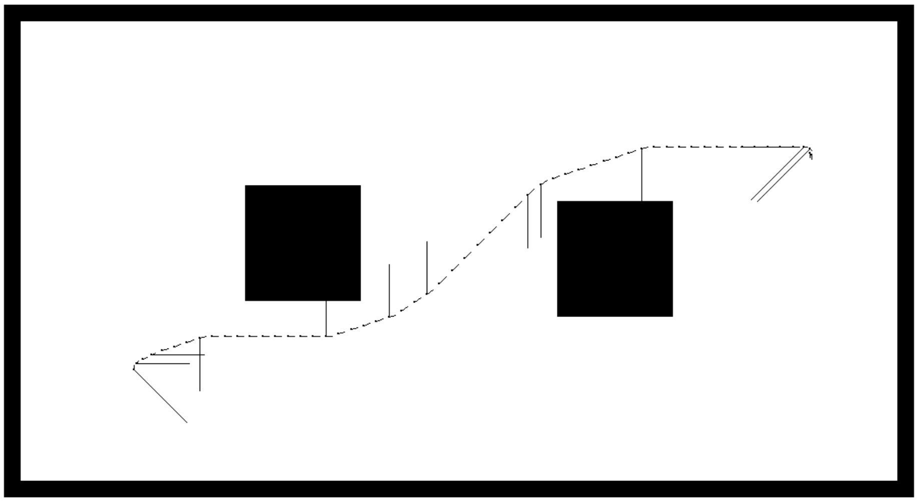


Figure 12: (a) A near-optimal trajectory found by our implementation; the "spikes" indicate the velocities. In (b), the commanded accelerations are shown. E.g., the commanded acceleration for the first time-step is a bang in the $(+x, -y)$ direction.

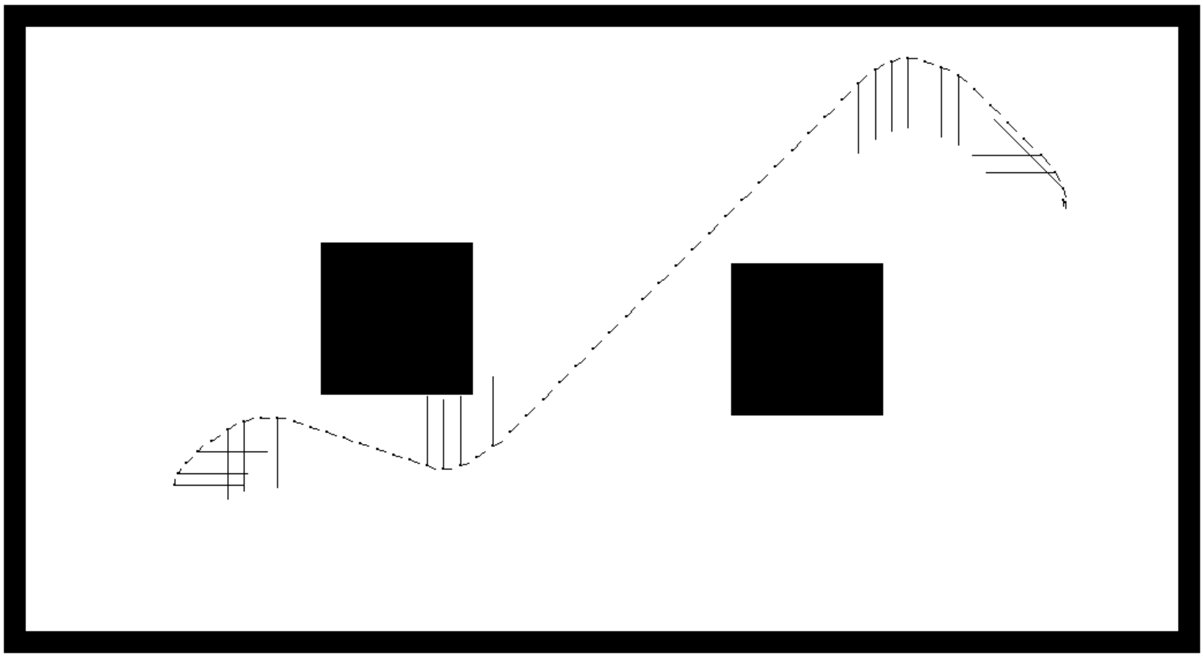


Figure 13: Choosing among multiple trajectories that reach a TC-gridpoint during a round affects properties unrelated to optimality.

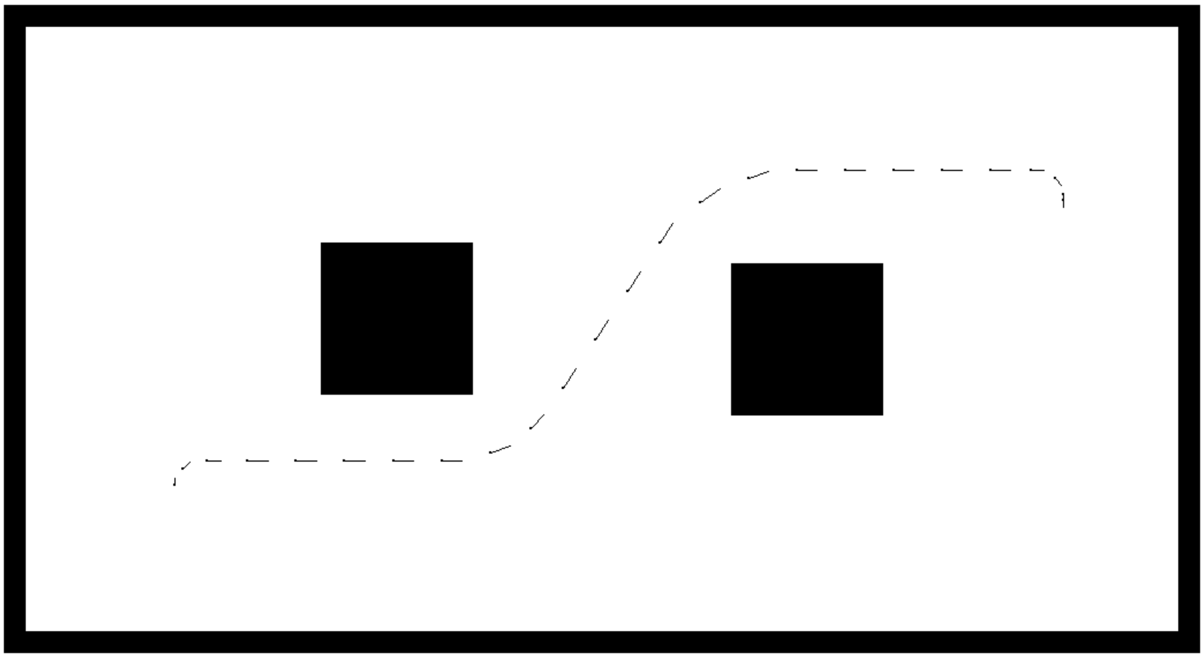


Figure 14: Although the main result of this paper shows that our algorithm can find trajectories that are exactly time-optimal as long as slack is allowed in safety, allowing *approximate* time-optimality results in much faster planning. For example, the implementation on a 32Mb SparcStation2 took five CPU minutes to plan the trajectory above, which obeys $\epsilon_S = \epsilon_T = 0.6$, $c_0 = 0.2$, $c_1 = 0.5$, $a_{max} = 0.2$, and $v_{max} = 0.3$. The environment is the same as in the other examples.