SIAM J. COMPUT.
Vol. 30, No. 1, pp. 161–190
© 2000 Society for Industrial and Applied Mathematics

NONUNIFORM DISCRETIZATION FOR KINODYNAMIC MOTION PLANNING AND ITS APPLICATIONS*

JOHN H. REIF† AND HONGYAN WANG†

Abstract. The first main result of this paper is *a novel nonuniform discretization approximation method for the kinodynamic motion-planning problem.* The kinodynamic motion-planning problem is to compute a collision-free, time-optimal trajectory for a robot whose accelerations and velocities are bounded. Previous approximation methods are all based on a uniform discretization in the time space. On the contrary, our method employs a nonuniform discretization in the configuration space (thus also a nonuniform one in the time space). Compared to the previously best algorithm of Donald and Xavier, the running time of our algorithm reduces in terms of $1/\varepsilon$, roughly from $O((1/\varepsilon)^{6d-1})$ to $O((1/\varepsilon)^{4d-2})$, in computing a trajectory in a d-dimensional configuration space, such that the time length of the trajectory is within a factor of $(1+\varepsilon)$ of the optimal. More importantly, our algorithm is able to take advantage of the obstacle distribution and is expected to perform much better than the analytical result. This is because our nonuniform discretization has the property that it is coarser in regions that are farther from all obstacles. So for situations where the obstacles are sparse, or the obstacles are unevenly distributed, the size of the discretization is significantly smaller.

Our second main result is the *first known polynomial-time approximation algorithm for the curvature-constrained shortest-path problem in three and higher dimensions.* We achieved this by showing that the approximation techniques for the kinodynamic motion-planning problem are applicable to this problem.

Key words. robotic motion planning, nonholonomic motion planning, kinodynamic motion planning, nonuniform discretization

AMS subject classifications. 68W25, 68W40, 65K10

PII. S0097539798331975

1. Introduction.

Nonholonomic motion planning involves planning a collision-free path (or trajectory) for a robot subject to *nonholonomic* constraints on its dynamics. A *holonomic* constraint is one that can be expressed as an equation of the robot's configuration parameters (a placement of a robot with k degrees of freedom can be uniquely specified by k such parameters), while a *nonholonomic* one can only be expressed as a *nonintegrable* equation involving also the derivatives of the configuration parameters (see [30] for a more detailed discussion on nonholonomic constraints). Examples of nonholonomic constraints are bounds on velocities, accelerations, and curvatures. Although there has been considerable recent work in the robotics literature (see [2, 3, 4, 5, 6, 7, 19, 24, 27, 28, 29, 31, 33, 34, 39, 42, 45, 47, 48] and references therein) on nonholonomic motion-planning problems, relatively little theoretical work has been done on these problems. Nonholonomic motion planning is considerably harder than holonomic. For one thing, a robot with k degrees of freedom cannot be described completely by k parameters. A complete description has to include the k parameters and their derivatives. The *configuration-space* approach, which is widely used for the holonomic motion-planning problems, does not apply to the problems with nonholonomic constraints because such constraints are not expressed by the

*Received by the editors December 31, 1997; accepted for publication (in revised form) March 18, 1999; published electronically April 25, 2000. This work was supported by NSF grant NSF-IRI-91-00681, Rome Labs contracts F30602-94-C-0037, ARPA/SISTO contracts N00014-91-J-1985, and N00014-92-C-0182 under subcontract KI-92-01-0182.

http://www.siam.org/journals/sicomp/30-1/33197.html

†Department of Computer Science, Duke University, Box 90129, Durham, NC 27708-0129 (reif@cs.duke.edu, hongyan@cs.duke.edu).

configuration-space representation. On the other hand, these problems bear major significance in robotic engineering. In reality, a robot arm has to move not only in a collision-free fashion, but also in conformation of the dynamic bounds due to limited force or torque from motors.

It is also desirable, in many cases, to minimize the cost of a motion plan under some cost function (e.g., path length, time length, number of turns, clearance, etc.). Finding an optimal solution is significantly harder than just finding any solutions. For example, the seemingly easy problem of finding the shortest path for a point robot in three dimensions among polyhedral obstacles is already NP-hard (see Canny and Reif [10]), even without constrained dynamics.

In this paper, we study two optimal nonholonomic motion-planning problems: the *kinodynamic motion-planning* problem and the *curvature-constrained shortest-path* problem. The kinodynamic motion-planning problem studies the problem of finding collision-free time-optimal trajectories for a robot whose motion is governed by Newtonian dynamics and whose accelerations and velocities are bounded. A *trajectory* of a robot with d degrees of freedom is a map $\Gamma\colon [0,T] \to \mathbb{R}^d \times \mathbb{R}^d$ given by $\Gamma(t) = (p(t), \dot{p}(t))$, where $p(t)$ and $\dot{p}(t)$ give the location and the velocity at time t, respectively, in a d-dimensional configuration space. $\ddot{p}(t)$ is the acceleration function, which determines a trajectory uniquely once an initial state is fixed. The constraints on dynamics are given by bounding the norms of the accelerations and velocities. The most studied norms are the L_∞-norm (called the *decoupled* case) and the L_2-norm (called the *coupled* case). In this paper, we study the coupled kinodynamic problem, which happens to be harder than the decoupled one. The decoupled case is simpler because each dimension is independent of the others, and a d-dimensional problem can be reduced to a one-dimensional (1D) one.

The curvature at one point on a path describes how fast the direction of the path changes at that point. Generally, a curvature constraint requires that a path has a curvature of at most c at every point along the path, where $c > 0$ is a given parameter. The curvature-constrained shortest-path problem is to compute a shortest collision-free path such that the path satisfies the given curvature constraint.

Our major contribution in this paper is a *nonuniform* discretization approximation method for the kinodynamic motion-planning problem. The discretization is nonuniform in the sense that it is coarser in regions which are farther from all obstacles. The intuition behind this is that in regions that are far away from all obstacles, even with a coarse discretization, we are still able to find an approximation trajectory which does not intersect obstacles. The nonuniform discretization on one hand reduces the search space and the running time, and on the other hand still enables us to obtain a collision-free trajectory whose time length is within a given factor of the optimal time length. Nonuniform discretization is widely used in solving PDEs (see Miller et al. [35, 36] and references therein), but research in that area focuses on quite different issues. Applying the idea to kinodynamic motion planning was first suggested by Xavier [49] but without a rigorous proof. As it happens, provably good bounds given by us are quite intriguing to obtain.

Our nonuniform discretization is based on a box decomposition of the configuration space. Other geometric algorithms using similar box decompositions include the work of Mitchell, Mount, and Suri [38] on ray-shooting problems and that of Hershberger and Suri [21] on two-dimensional (2D) shortest-path problems without constraints. Using nonuniform discretization for geometric planning with nonholonomic constraints presents a new challenge.

Instead of using a deterministic discretization, Kavraki et al. [25, 26] developed a *random sampling technique*, where in preprocessing, grid-points (called *milestones*) are chosen *randomly* and are connected by feasible paths to form a network. The method is similar to ours in that both do a preprocessing to obtain a set of valid paths (or trajectories) which are used later in answering planning queries. However, they have to preprocess for each new environment, where in our method, the preprocessing is only done once for some fixed parameters, and can be used repeatedly for different environments.

1.1. Previous work on the kinodynamic motion-planning problem. With the exceptions of 1D and 2D cases (see Ó'Dúnlaing [40] and Canny, Rege, and Reif [9]), there are no exact solutions for the kinodynamic motion-planning problem. In fact, as an implication of the result of [10], the problem is at least NP-hard in three and higher dimensions. In light of this lower bound, most study has been focusing on finding approximation solutions. The earlier approximation algorithms of Sahar and Hollerbach [44] did not guarantee goodness of their solutions. Moreover, the running time was exponential in the resolution. Canny et al. [8, 16] developed the first provably good, polynomial-time approximation algorithm for the decoupled kinodynamic case. Their work was followed up by a series of work in which Donald and Xavier [11, 15] improved the running time for the decoupled case, Heinzinger et al. [20] and Donald and Xavier [13, 14] investigated the problem for open chain manipulators, and independent work of Donald and Xavier [12, 13, 15] and Reif and Tate [43] gave approximation algorithms for the coupled kinodynamic problem. The best-known result for the coupled case, given by [15], is that given an $\varepsilon > 0$, one can compute in time $O(c(d)p(n, \varepsilon, d)L^d(1/\varepsilon)^{6d-1})$ an approximation trajectory whose time length is at most $(1+\varepsilon)$ times the time length of an optimal safe trajectory (roughly speaking, a safe trajectory is one that can be perturbed without intersecting obstacles), where d is the dimension, L is the size of the configuration space to which the robot is confined, n is the number of equations describing the configuration obstacles, $c(d)$ is a function depending solely on d, and $p(n, \varepsilon, d)$ is a lower-order polynomial in n, ε, and d.

1.2. Previous work on the curvature-constrained shortest-path problem. Dubins [17] was perhaps the first to study the curvature-constrained shortest paths who gave a characterization of the shortest paths in 2D in the absence of obstacles. Reeds and Shepp [41] extended the obstacle-free characterization to robots that can make reversals. (Boissonnat, Cerezo, and Leblond [4] gave an alternative proof for both cases, using ideas from control theory.) In the presence of obstacles, Fortune and Wilfong [18] gave a $2^{poly(n,m)}$-time algorithm, where n is the number of vertices and m is the number of bits of precision with which all points are specified; their algorithm only decides whether a path exists, without necessarily finding one. Jacobs and Canny [23] gave an $O((\frac{n+L}{\varepsilon})^2 + (\frac{n+L}{\varepsilon})n^2 \log n)$-time algorithm that computes an approximation path whose length is at most $(1+\varepsilon)$ times the length of an optimal path, where n is the number of obstacle vertices and L is the total edge length of the obstacles. This running time was later improved significantly by Wang and Agarwal [46] to $O((n^2/\varepsilon^2)\log n)$. Agarwal, Raghavan, and Tamak [1] studied a special case when the boundaries of obstacles are also constrained to have a curvature of at most 1. There has also been work on computing curvature-constrained paths when the robot is allowed to make reversals [2, 32, 37]. However, no result has been obtained for the curvature-constrained problem in three or higher dimensions, even in the absence of obstacles.

1.3. Models and results. Let B be a point robot moving in a d-dimensional configuration space. The velocity of B is bounded by $v_{\max}$ in L_2-norm and the acceleration of B is bounded by $a_{\max}$ in L_2-norm, where $v_{\max} > 0$ and $a_{\max} > 0$ are arbitrary. In section 2.7, we show that we can always scale $v_{\max}$ and $a_{\max}$ to 1 by scaling both time and the size of the configuration space. Therefore, without loss of generality, we will set $v_{\max} = 1$ and $a_{\max} = 1$.

A *state* x of B is a pair $(\text{LOC}(x), \text{VEC}(x))$, where $\text{LOC}(x)$ is a point representing the location of B in the d-dimensional configuration space and $\text{VEC}(x)$ is a vector representing the velocity of B. A trajectory is an $(\bar{a}, \bar{v})$-trajectory if at any time during the trajectory, the acceleration is bounded by $\bar{a}$ and the velocity is bounded by $\bar{v}$, both in L_2-norm. Thus only $(1, 1)$-trajectories are valid trajectories for robot B. Let Ω be a set of configuration obstacles. A trajectory is *collision free* if its path does not intersect the interior of Ω. A trajectory from state x to state y is *optimal* if it is a collision-free $(1, 1)$-trajectory with a minimum time length, where the minimum is taken over all collision-free $(1, 1)$-trajectories from x to y. The kinodynamic motion-planning problem is to compute such optimal trajectories.

Since we have seen that computing exact solutions is hard, we focus on developing fast approximation algorithms. To discuss approximation solutions, the notion of a safe trajectory needs to be introduced. A point p has a *clearance* of μ if for any point $q \in \Omega$, $\|p - q\|_\infty \geq \mu$, where $\|\cdot\|_\infty$ denotes the L_∞-norm operation. A path is *μ-safe* if for any point p along the path, p has a clearance of μ. A trajectory is a *μ-safe trajectory* if its path is μ-safe. A trajectory is an *optimal μ-safe* trajectory from state x to state y if its time length is the minimum over all μ-safe $(1, 1)$-trajectories from x to y.

The first main result of this paper is a faster algorithm for computing approximations to optimal safe trajectories. Let W be a d-dimensional configuration space of size L (without loss of generality, we can assume that W is a d-dimensional cube), where the point robot is confined to move. Let Ω be a set of configuration obstacles defined by a total of n algebraic equations, each of $O(1)$ degrees. Given the initial and the final states i and f and two parameters $l > 0$ and $\varepsilon > 0$, we can compute an approximation to the optimal $3l$-safe $(1, 1)$-trajectory from i to f in time $O(nN + N \log N(1/\varepsilon)^{4d-2})$, where $N = O((L/l)^d)$. The time length of the computed trajectory is at most $(1 + \varepsilon)$ times the time length of the optimal trajectory. The computed trajectory connects two states which are close to i and f, respectively. More precisely, if the computed trajectory is from state i' to state f', then $\|\text{LOC}(i') - \text{LOC}(i)\|_\infty \leq \varepsilon l$, $\|\text{VEC}(i') - \text{VEC}(i)\|_\infty \leq \varepsilon$, $\|\text{LOC}(f') - \text{LOC}(f)\|_\infty \leq \varepsilon l$, and $\|\text{VEC}(f') - \text{VEC}(f)\|_\infty \leq \varepsilon$.

Here we introduce two parameters ε and l, which are independent except that in general $\varepsilon < l$. ε describes how close the approximation should be to the optimal safe trajectory in time, while l describes how safe the optimal trajectory is. If we require that l be as small as ε, the running time of our algorithm in terms of ε is roughly $O((1/\varepsilon)^{5d-2})$, which improves over the running time $O((1/\varepsilon)^{6d-1})$ of the previously best algorithm of Donald and Xavier. If we choose a big l, our algorithm may fail to find a trajectory because $3l$-safe trajectories connecting the initial state to the final state may not exist due to, for example, the crowded obstacles. However, we can still choose the parameters such that $l \gg \varepsilon$, with both parameters being small. In this case, the running time of our algorithm in terms of ε is $O((1/\varepsilon)^{4d-2})$. Moreover, because of our nonuniform discretization, there are cases when the number of boxes is much smaller than the worst-case bound of $N = O((L/l)^d)$. In these cases, our

algorithm is expected to perform much better than the given time bound. Such cases include sparse obstacle distributions and uneven obstacle distributions. As will be pointed out in the conclusion section, one future work is to look for more accurate bounds on the number of boxes with respect to different obstacle distributions.

By showing that the approximation techniques for kinodynamic motion planning are applicable to the curvature-constrained shortest-path problem, we are able to obtain the second main result of this paper—the first known polynomial-time approximation algorithm for the curvature-constrained shortest-path problem in three and higher dimensions. The detailed model and result are presented in section 3.

2. The kinodynamic motion-planning problem.

2.1. Preliminaries. In this paper, we generally use $\|\cdot\|_\infty$ to denote the L_∞-norm operation and $\|\cdot\|$ the L_2-norm one.

Given a trajectory Γ, let $T(\Gamma)$ be the time length of Γ. Let $p_\Gamma(t), v_\Gamma(t), a_\Gamma(t)$ for $0 \le t \le T(\Gamma)$ be the location, the velocity, and the acceleration of Γ, respectively, at time t.

Let Π be a path and A a subset of the configuration space. We say that $d(\Pi, A) \le \rho$ for some real number $\rho \ge 0$ if for every point $p \in \Pi$, there is a point $q \in A$ such that $\|p - q\|_\infty \le \rho$. $d(\Pi, A) = 0$ if every point $p \in \Pi$ is also in A. Similarly, for two paths Π and Π', we say that $d(\Pi', \Pi) \le \rho$ if for every point $p \in \Pi'$, there is a point $q \in \Pi$ such that $\|p - q\|_\infty \le \rho$. Let Γ and Γ' be two trajectories and let Π and Π' be the paths of Γ and Γ', respectively. $d(\Gamma', \Gamma) \le \rho$ if $d(\Pi', \Pi) \le \rho$. Also $d(\Gamma', A) \le \rho$ if $d(\Pi', A) \le \rho$, where A is a subset of the configuration space.

For a state x, define

$$ngb(x, \rho, \nu) = \{x' \mid \|\textsc{loc}(x') - \textsc{loc}(x)\|_\infty \le \rho \text{ and } \|\textsc{vec}(x') - \textsc{vec}(x)\|_\infty \le \nu\}$$

for some $\rho, \nu > 0$. Thus $ngb(x, \rho, \nu)$ defines a set of states which are close to x. Notice that if $x' \in ngb(x, \rho, \nu)$, then $x \in ngb(x', \rho, \nu)$.

Lemma 2.1 (time-rescaling lemma, Hollerbach [22]). *Let Π be the path of an $(\bar{a}, \bar{v})$-trajectory Γ. Π can be traversed by an $(\bar{a}/(1+\varepsilon)^2, \bar{v}/(1+\varepsilon))$-trajectory Γ' in time $(1+\varepsilon)\,T(\Gamma)$ for any $\varepsilon > 0$.*

The time-rescaling lemma shows a trade-off between time and the dynamic bounds. Later we will see that in our approximation algorithm, we first compute a trajectory whose acceleration and velocity bounds are slightly bigger than 1. Then applying the time-rescaling lemma, we can reduce the dynamic bounds to 1 by sacrificing the time length by a small factor.

The TC-graph method developed in [13] by Donald and Xavier applies a graph-searching technique to compute approximations to time-optimal trajectories (for a brief description of the method, see the appendix). The following corollary states a result of the TC-graph method.

Corollary 2.2 (see [13]). *Let W be a d-dimensional configuration space of size L. Let Γ be a $(1,1)$-trajectory from state i to state f such that Γ lies inside W. Given any $\varepsilon > 0$ and $\rho = O(1)$, applying the TC-graph method with appropriate parameters, we can compute in time $O(L^d(1/\varepsilon)^{6d-1})$ a $(1,1)$-trajectory Γ' from a state i' to a state f' such that $i' \in ngb(i, \varepsilon\rho/2, \varepsilon/2)$, $f' \in ngb(f, \varepsilon\rho/2, \varepsilon/2)$, $T(\Gamma') \le (1+\varepsilon)\,T(\Gamma)$, and $d(\Gamma', W) \le \varepsilon\rho/2$.*

The above corollary states that by applying the TC-graph method, one can compute in polynomial time (in terms of L and ε) a trajectory which obeys the dynamic constraints and which approximates the given trajectory in the following ways. The

computed trajectory connects two states which are close to the given initial and final states, respectively. The time length of the computed trajectory is within $(1+\varepsilon)$ times that of the given trajectory. Also the path of the computed trajectory stays close to the region W, within which the path of the given trajectory lies.

We will see that in later sections we apply the TC-graph method to precompute a set of trajectories which in turn become the building-block trajectories in our approximation method. Here we present the result for cases where there are no obstacles. This is sufficient for our purpose, though the TC-graph method works also in the presence of obstacles.

In the rest of the paper, we fix $l > 0$ and $\epsilon > 0$ unless otherwise stated, where l gives the size of the smallest box in the box decomposition and ϵ describes the fineness of the discretization.

2.2. Overview of the algorithm. Following the framework of many motion-planning algorithms, our algorithm reduces the problem to that of computing and then searching on a graph. The nodes of the graph correspond to a set of states (see section 2.3) which are induced by a nonuniform discretization over the configuration space. Therefore, we term our method the CS-graph method, where C stands for the configuration space and S the state space.

Section 2.5 describes how to compute the edge set of the graph. Each edge corresponds to a collision-free, near-optimal trajectory between two nodes. It is shown in this section that the graph satisfies the property that if there is an optimal safe trajectory, then there exists a path in the graph that corresponds to a collision-free trajectory whose time length is near optimal. Thus the problem is reduced to a shortest-path search on the computed graph.

In order to compute the edge set efficiently, we perform precomputations. We derive a set of canonical trajectories (section 2.4), which are sufficient for building the graph. These trajectories, once computed, can be used repeatedly for different problem instances with different distributions of obstacles.

A technical lemma, the correcting lemma (Lemma 2.15), is described in section 2.6. This lemma is a precise statement of our intuition that the discretization can be coarser in regions farther away from all obstacles.

2.3. Nonuniform grids. The set of nonuniform grids is generated based on a box decomposition of the configuration space (see section 2.3.1). The decomposition is such that the size of a box is roughly proportional to the smallest distance between the box and the obstacles. Each box contributes the same number of grids, despite its size; the distances among grids induced by larger boxes are larger. Thus the grids are nonuniform (see section 2.3.2).

2.3.1. A box decomposition. Let $\tilde{\Xi}(s)$ be a set $\{p = (p_1, \ldots, p_d) \mid -s/2 \le p_1, \ldots, p_d \le s/2\}$. $\tilde{\Xi}(s)$ is called a d-dimensional *canonical box* of size s. Each set of $\{p \mid p \in \tilde{\Xi}(s) \text{ and } p_j = -s/2\}$ and $\{p \mid p \in \tilde{\Xi}(s) \text{ and } p_j = s/2\}$ for $1 \le j \le d$ is called a *face* (of size s) of $\tilde{\Xi}(s)$. Ξ is a d-dimensional *box* of size s if it is a region which can be obtained from $\tilde{\Xi}(s)$ by translation and rotation (reflection is not necessary because of the symmetry of the box). Similarly we can define the faces of Ξ. Notice that a face of a d-dimensional box Ξ of size s is really a $(d-1)$-dimensional box of size s. A d-dimensional box has $2d$ faces.

Given a d-dimensional box of size s, we can decompose it into 2^d boxes, each of size $s/2$. Each of them can be further decomposed into 2^d boxes of size $s/4$ and so on. We stop further decomposing until certain conditions hold. We refer to this procedure,

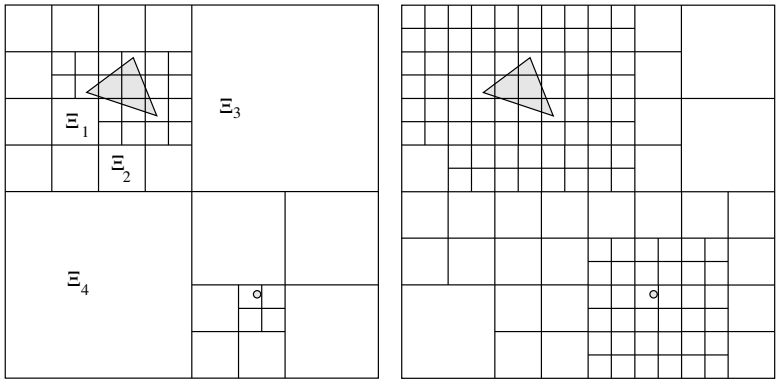

FIG. 2.1. (a) *A box decomposition satisfying* C1; (b) *one also satisfying* C2 *and* C3.

as well as the collection of nondecomposed boxes obtained, as a *box decomposition*. When we refer to a box of a decomposition, we mean a nondecomposed box.

Let W be a d-dimensional configuration space (assuming without loss of generality that W is a d-dimensional box) of size L, with a set Ω of configuration obstacles. (We consider the initial and final locations also as obstacle vertices.) A box is *free* if it does not intersect the interior of Ω. A box is *occupied* if it is a subset of Ω. Otherwise a box is called *partially occupied*. A box B_1 is *adjacent* to a box B_2 (or B_1 and B_2 are *neighbors*) if B_1 and B_2 share common points. We perform a box decomposition on W until the following conditions are satisfied:

C1. The size of every partially occupied box is l.

C2. If a free box is adjacent to a nonfree box, or to a free box with nonfree boxes as neighbors, its size is l.

C3. The decomposition is *balanced*, i.e., a free box has only adjacent free boxes whose sizes are either twice as large or half as small.

Figure 2.1 shows an example of a box decomposition in two dimensions. The shaded triangle is an obstacle, and the shaded disk represents the initial location. Figure 2.1(a) gives a box decomposition only satisfying C1. In Figure 2.1(b), boxes are further decomposed to satisfy C2 (e.g., boxes Ξ_1 and Ξ_2) and C3 (e.g., boxes Ξ_3 and Ξ_4).

We use a tree structure to represent the box decomposition, where an internal node represents a decomposed box and a leaf node a nondecomposed one. Each internal node has 2^d children. We can perform the box decomposition in two stages. In the first stage, decompose until C1 is satisfied. Let M be the number of boxes obtained after this stage. It is easy to see that this stage can be computed in time $O(nM)$, where n is the number of obstacle constraints. In the second stage, we further decompose certain boxes until C2 and C3 are satisfied. Basically, we check all the leaf nodes in a bottom-up manner, working from small boxes to large ones. For each leaf node, find its neighbors and further decompose them if their sizes are too large. Let N be the size of the final box decomposition. We claim that the time spent in this stage is $O(Nlog(L/l))$. Since the size of the smallest box is l, the box-decomposition tree has a depth of $O(\log(L/l))$. Thus finding the neighbors of a box takes at most $O(\log(L/l))$ time. Since we find neighbors for at most N boxes, the total running time is $O(N \log(L/l))$.

Notice that in the worst case, $N = O((L/l)^d)$, but N tends to be much smaller for cases where the obstacles are sparse.

Let $\mathcal{B}$ be the final box decomposition of W. Later, when we refer to a box (resp., a face) of $\mathcal{B}$, we mean a nondecomposed box (resp., face) of $\mathcal{B}$.

2.3.2. Discretization. Let $\mathcal{B}$ be the box decomposition of W (with respect to Ω) satisfying C1–C3. For each face Δ of $\mathcal{B}$ such that Δ is not further decomposed (recall that Δ is a $(d-1)$-dimensional box), we select $(2/\epsilon)^{d-1}$ uniformly spaced points on Δ, with spacing $\epsilon s/2$, where s is the size of Δ. The set of points obtained in this way are called the *CS grid points* of $\mathcal{B}$. Notice that the number of grid points on each face is the same, while the spacing of grid points grows linearly with the size of the faces. Thus we obtain a nonuniform discretization of the configuration space.

Let

$$\mathcal{V} = \{v = (j_1\epsilon, \ldots, j_d\epsilon) \mid j_1, \ldots, j_d \in \mathbb{Z} \text{ and } \|v\| \leq 1\}. \tag{2.1}$$

$\mathcal{V}$ is called the set of *grid velocities*. A state a is called a *CS grid state* of $\mathcal{B}$ if $\textsc{loc}(a)$ is a *CS* grid point of $\mathcal{B}$ and $\textsc{vec}(a) \in \mathcal{V}$. The grid states of $\mathcal{B}$ (including the initial and the final states) are the nodes of the graph to be constructed.

For a state a that lies on a face Δ, let $s(a)$ be the function returning the size of the face Δ. Define $ngb(a) = ngb(a, \epsilon s(a)/4, \epsilon/2)$. By the construction of the grid states, it is easy to show the following result.

LEMMA 2.3. *If a is a state such that $\textsc{loc}(a)$ lies on a face Δ of $\mathcal{B}$, there exists a CS grid state a' such that $a \in ngb(a')$.*

2.4. Precomputing canonical trajectories. In this section, we derive a set of *canonical trajectories*, which once computed can be used repeatedly for different problem instances with different obstacle distributions. These canonical trajectories are from grid states to grid states which lie on the faces of an *extended box*. The extended boxes are to guarantee that each canonical trajectory has a length at least proportional to the size of the box it lies in (see section 2.4.1). Section 2.4.2 describes the canonical trajectories and how to compute them.

2.4.1. Extended boxes. Let $\mathcal{B}$ be the box decomposition of W. Given a state a, define $\xi(a)$ to be the *extended box* of a, which is the smallest region composed of boxes such that (i) the boundary of the union of these boxes forms a closed $(d-1)$-dimensional surface and (ii) for any point q that lies on the closed surface, $\|\textsc{loc}(a) - q\|_\infty \geq s(a)/2$.

Let Δ be a face of $\mathcal{B}$ of size s and let Ξ be the box that contains Δ (pick one arbitrarily if both of the two boxes containing Δ have size s). Decompose Δ into 2^{d-1} $(d-1)$-dimensional boxes, each of size $s/2$, and label them $\Delta_1, \ldots, \Delta_{2^{d-1}}$. Given any two states a and b, if both $\textsc{loc}(a)$ and $\textsc{loc}(b)$ lie in the interior of Δ_j, then $\xi(a)$ and $\xi(b)$ are the same. This implies that we can extend the definition of extended boxes for faces. Define $\xi(\Delta_j)$ to be the extended boxes of Δ_j, which are the same as the extended boxes of $\xi(a)$, for any state a such that $\textsc{loc}(a)$ lies in the interior of Δ_j. $\xi(\Delta_j)$ is called an extended box of *size* $s/2$, since Δ_j is a face of size $s/2$.

An extended box (resp., a box) is said to have a *clearance* of μ if for any point p that lies in the extended box (resp., the box), p has a clearance of μ. Notice that a free box has a clearance of s if all its neighbors are free boxes and have a size of at least s. The following two lemmas describe the clearance properties of extended boxes of different sizes.

LEMMA 2.4. *An extended box of size $s \geq 2l$ has a clearance of at least $s/2$; an extended box of size l has a clearance of at least l.*

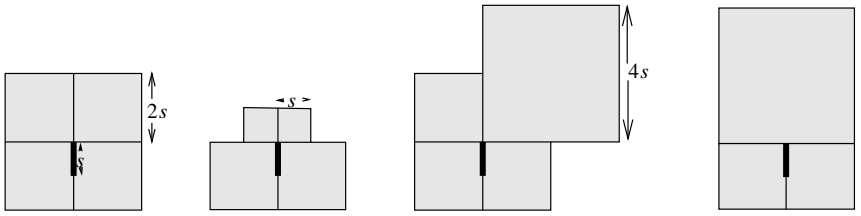

FIG. 2.2. *Some canonical extended boxes of* $\tilde{\Delta}(s)$*, where* $\tilde{\Delta}(s)$ *is a line segment drawn in thick lines.*

Proof. Suppose that the extended box is an extended box of face Δ_j whose size is s. Let Ξ be the box containing Δ_j and the size of Ξ be $2s$. Each other box in the extended box is a neighbor of Ξ. Thus their sizes are at least s, by the balanced property of the box decomposition.

If $s \geq 2l$, for any box Ξ' in the extended box, its neighbors must have sizes $\geq s/2$. By property C2 of the box decomposition, these neighbor boxes must be free. Otherwise, the size of Ξ' cannot be larger than l. Thus Ξ' has a clearance of $\geq s/2$. This implies that the extended box has a clearance of $\geq s/2$.

If $s = l$, then Ξ is a box of size $2l$. Since the box decomposition satisfies C2, we know that all Ξ's neighbors and all their neighbors are free boxes whose sizes are at least l. This implies that the extended box has a clearance of l. This completes the proof. □

LEMMA 2.5. *Let* $\xi(\Delta_j)$ *be an extended box of a face* Δ_j *of size* $l/2$*. If there exists a point* p *on* Δ_j *such that* p *has a clearance of* $3l$*, then the clearance of the extended box is at least* l*.*

Proof. It suffices to prove that for each point q on the boundary of $\xi(\Delta_j)$, q has a clearance of l. If q lies on the face of a box of size l, then $\|q - p\|_\infty \leq 2l$. Since the clearance of p is $3l$, the clearance of q is at least $3l - 2l = l$. If q lies on the face of a box of size $\geq 2l$, its clearance is at least l, since the clearance of this box is at least l. □

Let $\tilde{\Delta}(s/2) = \{p = (p_1, \ldots, p_d) \,|\, p_1 = -s/2 \text{ and } 0 \leq p_2, \ldots, p_d \leq s/2\}$. $\tilde{\Delta}(s/2)$ is called the *canonical face* of size $s/2$. We can transform Δ_j, Ξ, and $\xi(\Delta_j)$ by translation, rotation, and reflection such that Ξ becomes $\tilde{\Xi}(s)$, the canonical box of size s, and Δ_j becomes $\tilde{\Delta}(s/2)$. The transformed extended box $\xi(\Delta_j)$ is called a *canonical extended box* of $\tilde{\Delta}(s/2)$, or a canonical extended box of size $s/2$. Since the box decomposition $\mathcal{B}$ is balanced, the number of canonical extended boxes of a fixed size is a function depending only on the number of dimensions d, but not on the size or the location. Let $\xi(s)$ be the set of canonical extended boxes of $\tilde{\Delta}(s)$. We can give an order to $\xi(s)$ and let $\xi(s, j)$ be the jth canonical extended box of $\tilde{\Delta}(s)$ for some valid j. Figure 2.2 gives some examples of canonical extended boxes in two dimensions.

Let $\mathcal{P}(s)$ be the set of $(1/\epsilon)^{d-1}$ uniformly spaced points on $\tilde{\Delta}(s)$ with spacing ϵs. Define $\mathcal{Q}(s, j)$ as follows. If Δ is a face of size s' belonging to the boundary of $\xi(s, j)$, include the set of $(2/\epsilon)^{d-1}$ uniformly spaced points with spacing $\epsilon s'/2$ on Δ in $\mathcal{Q}(s, j)$. Let $\mathcal{I}(s) = \mathcal{P}(s) \times \mathcal{V}$ and $\mathcal{F}(s, j) = \mathcal{Q}(s, j) \times \mathcal{V}$. A state in the set $\mathcal{I}(s)$ is called a *canonical starting state* and a state in the set $\mathcal{F}(s, j)$ is called a *canonical ending state* of $\xi(s, j)$. It is easy to see that $|\mathcal{I}(s)| = O((1/\epsilon)^{2d-1})$. It also holds that $|\mathcal{F}(s, j)| = O(d(1/\epsilon)^{2d-1})$. This is because of the balanced property of the box decomposition. The boundary of $\xi(s, j)$ can contain at most $O(d)$ faces, each contributing $O((1/\epsilon)^{2d-1})$ states.

2.4.2. Computing connection tables. Let $\xi(s,j)$ be the jth canonical extended box of $\tilde{\Delta}(s)$ for some valid s and j. Let $CT(s,j)$ be the *connection table* of $\xi(s,j)$. The connection table contains precomputed trajectories which connect canonical starting states and canonical ending states. In this section we describe how to compute the connection tables.

First we introduce our correcting lemma (whose proof can be found in section 2.6), which states a result essential to the proofs of other lemmas.

LEMMA 2.6 (correcting lemma). *Fix a constant $c \leq 1$ and let $\rho_c = \bar{v}^2/(c\bar{a})$, where $\bar{a}, \bar{v} > 0$ are arbitrary. Let Γ be an $(\bar{a},\bar{v})$-trajectory from i to f. Given any $\rho > 0$ and $\varepsilon > 0$ and given any two states g and h, if $\rho > \rho_c$, $\|\text{LOC}(f) - \text{LOC}(i)\|_\infty \geq \rho$, $g \in ngb(i, \varepsilon\rho/2, \varepsilon\bar{v}/2)$, and $h \in ngb(f, \varepsilon\rho/2, \varepsilon\bar{v}/2)$, we can construct a trajectory Γ' from g to h by correcting Γ such that Γ' satisfies the following properties:*

P1. $T(\Gamma') = T(\Gamma)$.

P2. *Γ' is a $((1+4c\sqrt{d}\varepsilon)^2\bar{a}, (1+4c\sqrt{d}\varepsilon)\bar{v})$-trajectory.*

P3. $d(\Gamma', \Gamma) \leq (17/16)\varepsilon\rho$.

Here onwards, we fix

$$c = \max(2/l, 1), \tag{2.2}$$

where l is the size of the smallest box. In our later application of this lemma, it always holds that $\bar{v}^2/\bar{a} = 1$, thus $\rho_c = 1/c$. This implies that

$$l/2 \geq \rho_c. \tag{2.3}$$

Also let

$$e = 4c\sqrt{d}. \tag{2.4}$$

Consider a pair of states (i,f) that lie on the boundary of an extended box ζ. A $(1,1)$-trajectory from i to f is called *legal* if its path lies inside ζ. The pair (i,f) is *legal* if there exists at least a legal trajectory from i to f. We will see later that these legal trajectories are the potential trajectories that we need to approximate. We call a pair of grid states (i,f) *good* if there exists at least a legal pair (g,h) such that $g \in ngb(i)$ and $h \in ngb(f)$. Recall that $ngb(i) = ngb(i, \epsilon s(i)/4, \epsilon/2)$, where $s(i)$ gives the size of the face where the location of state i lies. Roughly speaking, the next lemma (existence lemma) states that if (i,f) is a good pair, then there *exists* a trajectory Γ from i to f such that Γ approximates *any* legal trajectories for any legal pair (g,h), where $g \in ngb(i)$ and $h \in ngb(f)$. Thus for our purpose of approximation, it is enough to compute only the trajectories for good pairs. For a pair of legal states (a,b), let $\hat{T}(a,b)$ be the time length of the legal trajectory from a to b whose time length is the smallest among all legal trajectories from a to b.

LEMMA 2.7 (existence lemma). *Let $\xi(s,j)$ be the jth canonical extended box of $\tilde{\Delta}(s)$ for some valid s and j. For any $i \in \mathcal{I}(s)$ and $f \in \mathcal{F}(s,j)$, if (i,f) is good, then there exists a trajectory Γ from i to f such that the trajectory satisfies the following properties:*

P1. *For any $i' \in ngb(i)$ and $f' \in ngb(f)$, if (i',f') is legal, $T(\Gamma) \leq \hat{T}(i',f')$.*

P2. *Γ is a $((1+e\epsilon)^2, (1+e\epsilon))$-trajectory.*

P3. $d(\Gamma, \xi(s,j)) \leq (17/8)\epsilon s$.

Proof. Let $g \in ngb(i)$ and $h \in ngb(f)$ be such that (g,h) is legal and that $\hat{T}(g,h)$ is the smallest among those of all such legal pairs;let Γ' be the $(1,1)$-trajectory from g

to h whose time length is $\hat{T}(g,h)$. We will show the existence of a trajectory satisfying P1–P3 by constructing one from Γ'. There are three cases depending on whether $s(f)$ (the size of the face where f lies) is s, $2s$, or $4s$. We will prove the case when $s(f)=4s$; this case gives the worst bounds. The other two cases can be handled in a similar way.

Let Γ be the trajectory obtained by correcting Γ'. Let $\bar{a}=1$, $\bar{v}=1$, and $\rho=2s$ in the correcting lemma (Lemma 2.6). Since $s \geq l/2$, $\rho=2s \geq l > \rho_c$. We can show that $\|\textsc{loc}(h)-\textsc{loc}(g)\|_\infty \geq 2s=\rho$. Since $g \in ngb(i)$, $g \in ngb(i,\epsilon\, s(i)/4,\epsilon/2)$. Since $\rho=2s=s(i)$, it is also true that $g \in ngb(i,\epsilon\rho/2,\epsilon/2)$. This implies that $i \in ngb(g,\epsilon\rho/2,\epsilon/2)$. Similarly, $f \in ngb(h,\epsilon\rho/2,\epsilon/2)$. Thus the conditions of the correcting lemma are satisfied. This implies that $T(\Gamma) \leq \hat{T}(g,h)$, satisfying P1. Γ is a $((1+e\epsilon)^2,(1+e\epsilon))$-trajectory, and $d(\Gamma,\Gamma') \leq (17/16)\epsilon\rho = (17/8)\epsilon s$. Since $d(\Gamma',\xi(s,j))=0$, $d(\Gamma,\xi(s,j)) \leq (17/8)\epsilon s$, proving P3. □

For a good pair (i,f) (with respect to $\xi(s,j)$), our goal is to approximate (since we do not know how to compute exactly) such a trajectory Γ as stated in the existing lemma (Lemma 2.7). The approximation is done in two steps. First, we apply the TC-graph method to compute a trajectory Γ' which is close to Γ timewise and spatially. However, the TC-graph method does not guarantee that Γ' is from i to f. Instead, we only know that Γ' is from some i' close to i to some f' close to f. Second, we correct Γ' to obtain a trajectory Γ'' which is really from i to f. By the correcting lemma, Γ'' approximates Γ', and thus approximates Γ. In the first step, by setting $\rho=l/2$ and $\varepsilon=\epsilon$ in Corollary 2.2, we are able to guarantee that

- Γ' is a $((1+e\epsilon)^2,(1+e\epsilon))$-trajectory. (Notice that even though Corollary 2.2 is about $(1,1)$-trajectories, the results apply to $((1+e\epsilon)^2,(1+e\epsilon))$-trajectories.)
- $T(\Gamma') \leq (1+\epsilon)T(\Gamma)$.
- Γ' is from some $i' \in ngb(i,\epsilon l/4,\epsilon/2)$ to some $f' \in ngb(f,\epsilon l/4,\epsilon/2)$.
- $d(\Gamma',\xi(s,j)) \leq (17/8)\epsilon s+\epsilon l/2$. The existence of a trajectory satisfying this condition in the TC graph is due to the fact that $d(\Gamma,\xi(s,j)) \leq (17/8)\epsilon s$ and to the existence of a trajectory $\tilde{\Gamma}$ in the graph such that $d(\tilde{\Gamma},\Gamma) \leq \epsilon l/2$.

In the second step, setting $\bar{a}=(1+e\epsilon)^2$, $\bar{v}=1+e\epsilon$, and $\rho=l/2 \geq \rho_c$, we can see that the conditions of the correcting lemma are satisfied. Thus Γ'' is a $((1+e\epsilon)^4,(1+e\epsilon)^2)$-trajectory, $T(\Gamma'')=T(\Gamma') \leq (1+\epsilon)T(\Gamma)$, and $d(\Gamma'',\Gamma') \leq (17/32)\epsilon l$. Since $d(\Gamma',\xi(s,j)) \leq (17/8)\epsilon s+\epsilon l/2$, $d(\Gamma'',\xi(s,j)) \leq (17/8)\epsilon s+(33/32)\epsilon l \leq 5\epsilon s$ (using the fact that $s \geq l/2$). In summary, we obtain the following result.

LEMMA 2.8 (loose-tracking lemma). *Let $\xi(s,j)$ be the jth canonical extended box of $\tilde{\Delta}(s)$ for some valid s and j. Let $i \in \mathcal{I}(s)$ and $f \in \mathcal{F}(s,j)$. If (i,f) is good, then Γ'' computed as above satisfies the following properties:*

P1. *For any $g \in ngb(i)$ and $h \in ngb(f)$, if (g,h) is legal, $T(\Gamma'') \leq (1+\epsilon)\,\hat{T}(g,h)$.*

P2. *Γ'' is a $((1+e\epsilon)^4,(1+e\epsilon)^2)$-trajectory.*

P3. *$d(\Gamma'',\xi(s,j)) \leq 5\epsilon s$.*

Fix a canonical extended box $\xi(s,j)$ for some valid s and j. For each $i \in \mathcal{I}(s)$ and $f \in \mathcal{F}(s,j)$, we precompute a trajectory from i to f as described above. We store the trajectory (i.e., the initial state i, the final state f, the acceleration function, and the time length) in the connection table $CT(s,j)$. A connection table computed in this way satisfies the following property.

LEMMA 2.9 (connection-table property). *Let $\xi(s,j)$ be the jth canonical extended box of $\tilde{\Delta}(s)$ for some valid s and j and let $CT(s,j)$ be the connection table for $\xi(s,j)$. For any canonical starting state $i \in \mathcal{I}(s)$ and any canonical ending state $f \in \mathcal{F}(s,j)$, if (i,f) is good, then $CT(s,j)$ contains a trajectory Γ from i to f and Γ satisfies the*

following properties:

P1. *For any* $g \in ngb(i)$ *and* $h \in ngb(f)$*, if* (g,h) *is good,* $T(\Gamma) \leq (1+\epsilon)\,\hat{T}(g,h)$.

P2. Γ *is a* $((1+e\epsilon)^4, (1+e\epsilon)^2)$*-trajectory.*

P3. $d(\Gamma, \xi(s,j)) \leq 5\epsilon s$.

Let $\xi(l/2,0)$ be the canonical extended box that consists of four boxes of size l. Apart from $CT(l/2,0)$, we maintain a special connection table CT_0 for this canonical extended box. For each $i \in \mathcal{I}(l/2)$, we construct a TC graph G rooted at i (for a description of the TC-graph method, see the appendix). We add a trajectory Γ from i to f in CT_0, if (i) f is a node of G, (ii) f either lies inside $\xi(l/2,0)$ or its L_∞ distance to the extended box is no more than $\epsilon l/2$, (iii) there is a path on G from i to f (whose corresponding trajectory is Γ). By the property of the TC-graph method, we can show the following.

LEMMA 2.10. *Let* $i \in \mathcal{I}(l/2)$*. For any pair of states* g *and* h *such that* $g \in ngb(i, \epsilon l/2, \epsilon/2)$*,* h *lies inside* $\xi(l/2,0)$*, and the optimal* $(1,1)$*-trajectory from* g *to* h *lies inside* $\xi(l/2,0)$*, there exists a* $(1,1)$*-trajectory in* CT_0 *from* i *to some* f *such that* $f \in ngb(h, \epsilon l/2, \epsilon/2)$ *and the time length of this trajectory is no more than* $(1+\epsilon)$ *times the time length of the optimal* $(1,1)$*-trajectory from* g *to* h*.*

To correct a trajectory takes only $O(1)$ time, since it basically involves computing and adding $O(1)$ number of corrective acceleration terms (see section 2.6). So the time complexity of computing a connection table $CT(s,j)$ is determined by how fast we can compute a trajectory Γ' for each pair of canonical starting and ending states. We can compute the trajectories using the TC-graph method in the following manner. For each canonical starting state i, compute the TC graph rooted at i. Expand the graph until all the canonical ending states of $\xi(s,j)$ are reached, or no new nodes can be added. A shortest graph path from i to a canonical ending state gives a trajectory between these two states. All these paths can be stored in a concise way by storing at each node its preceding node along the paths. Since the extended box is of size s, the graph has $O(s^d(1/\epsilon)^{6d-1})$ edges. (For a brief analysis on the number of TC-graph edges, see the end of the appendix.) This also bounds the time and space for computing the trajectories for a single canonical starting state. Since there are $O((1/\epsilon)^{2d-1})$ canonical starting states for an extended box, it takes $O(s^d(1/\epsilon)^{8d-2})$ time and space to compute a connection table $CT(s,j)$. A similar analysis shows that CT_0 can also be computed in $O(l^d(1/\epsilon)^{8d-2})$ time and space.

2.5. Approximation in the presence of obstacles.

2.5.1. The algorithm. In this section we present our approximation algorithm for computing collision-free near-optimal trajectories in the presence of obstacles. Let W be a d-dimensional configuration space of size L with a set Ω of obstacles. Let i and f be the initial and the final states, respectively. And let $\mathcal{B}$ be the box decomposition of W satisfying C1–C3 (recall that we presented C1–C3 and how to obtain such a box decomposition in section 2.3.1). Our approximation algorithm constructs a weighted directed graph $G = (V,E)$, where V includes i, f, and a subset of CS grid states induced by $\mathcal{B}$. A CS grid state a is included in V if (i) $s(a) \geq 2l$ or (ii) $s(a) = l$ and the clearance of $\xi(a)$ is at least l.

Let a and b be two states that lie within an extended box ζ. To see if there is a precomputed trajectory from a to b, we transform ζ to its canonical form, and also a and b with it. If $\mathcal{T}$ is the transformation, we say that there is a precomputed trajectory from a to b if there is a trajectory from $\mathcal{T} \circ a$ to $\mathcal{T} \circ b$ in the connection table of $\mathcal{T} \circ \zeta$. If Γ is the trajectory from $\mathcal{T} \circ a$ to $\mathcal{T} \circ b$, then $\mathcal{T}^{-1} \circ \Gamma$ is the trajectory

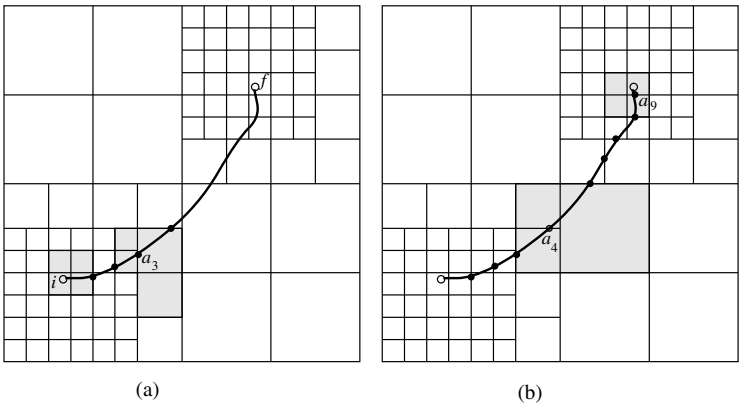

FIG. 2.3. (a) $\xi(a_0)$ *and* $\xi(a_3)$; (b) $\xi(a_4)$ *and* $\xi(a_9)$.

from a to b. It is easy to see that $\mathcal{T}^{-1} \circ \Gamma$ satisfies the connection-table properties P1–P3 (see Lemma 2.9) with respect to ζ.

We construct the edge set in the following way. For node i, add an edge from i to another node a if (i) i lies within $\xi(a)$ and (ii) there is a precomputed trajectory from[1] a^- to i^-. Similarly, for node f, add an edge from a node a to f if (i) f lies within $\xi(a)$ and (ii) there is a precomputed trajectory from a to f. For any other node a, add an edge from a to another node b if (i) b lies on the boundary of $\xi(a)$ and (ii) there is a precomputed trajectory from a to b. The weight of each edge is equal to the time length of its corresponding trajectory.

THEOREM 2.11 (safe loose-tracking theorem). *If there exists an optimal* $3l$*-safe* $(1,1)$*-trajectory* Γ *from state* i *to state* f*, then the graph* G*, constructed as above, contains a path whose corresponding trajectory* Γ' *satisfies the following properties:*

P1. $T(\Gamma') \leq (1+\epsilon)\,T(\Gamma)$.

P2. Γ' *is a* $((1+e\epsilon)^4, (1+e\epsilon)^2)$*-trajectory.*

P3. Γ' *does not intersect the interior of* Ω.

P4. Γ' *is from some* $i' \in ngb(i, \epsilon l/2, \epsilon/2)$ *to some* $f' \in ngb(f, \epsilon l/2, \epsilon/2)$.

Proof. Let Γ be divided into segments of trajectories $\Gamma_{a_0a_1} \| \Gamma_{a_1a_2} \| \dots \| \Gamma_{a_{m-1}a_m}$ such that $a_0 = i$, $a_m = f$, and each a_j is the state where Γ first exits $\xi(a_{j-1})$ for $1 \leq j \leq m-1$. Figure 2.3 gives an example in 2D. The optimal $(1,1)$-trajectory from i to f (drawn in thick curve) is divided into 10 segments (each dark circle represents an a_j for $1 \leq j < 10$). Some of the extended boxes are shown as shaded regions. We consider the following parts $\Gamma_{a_0a_1}$, $\Gamma_{a_1a_2} \| \dots \| \Gamma_{a_{m-2}a_{m-1}}$, and $\Gamma_{a_{m-1}a_m}$ separately.

Since each a_j, for $1 \leq j \leq m-1$, is on some face of $\mathcal{B}$, by Lemma 2.3, we can find a CS grid state b_j such that $a_j \in ngb(b_j)$. If $s(a_j) \geq 2l$, then $s(b_j) \geq 2l$ and b_j is in the node set V. Otherwise, since a_j is $3l$-safe, $\xi(a_j)$ has a clearance of at least l. Since $\xi(b_j) = \xi(a_j)$, $\xi(b_j)$ also has a clearance of at least l and b_j is in the node set V. We will show that the graph path $(b_1, b_2) \| \dots \| (b_{m-2}, b_{m-1})$ corresponds to a trajectory satisfying P1–P3. To this end, we show that there is an edge from b_{j-1} to b_j whose corresponding trajectory $\Gamma_{b_{j-1}b_j}$ satisfies P2, P3, and $T(\Gamma_{b_{j-1}b_j}) \leq (1+\epsilon)\,T(\Gamma_{a_{j-1}a_j})$ for $1 < j < m$.

By the way Γ is divided, $d(\Gamma_{a_{j-1}a_j}, \xi(a_{j-1})) = 0$. Thus each pair (a_{j-1}, a_j) is legal. This implies that each (b_{j-1}, b_j) is good. By the connection table property

[1]For a state a, we use a^- to denote a state of $(\text{LOC}(a), -\text{VEC}(a))$.

(Lemma 2.9), there is a precomputed trajectory $\Gamma_{b_{j-1}b_j}$ from b_{j-1} to b_j. Thus an edge from b_{j-1} to b_j is added in the construction of G. Also by the connection table property, $\Gamma_{b_{j-1}b_j}$ is a $((1+e\epsilon)^4,(1+e\epsilon)^2)$-trajectory, and its time length is no more than $(1+\epsilon)T(\Gamma_{a_{j-1}a_j})$. Next we need to show that $\Gamma_{b_{j-1}b_j}$ is collision free.

Let s be the size of $\xi(b_{j-1})$. We consider the three cases $s \geq 2l$, $s = l$, and $s = l/2$ separately. If $s \geq 2l$ (resp., $s = l$), the extended box $\xi(b_{j-1})$ has a clearance of $s/2$ (resp., l). On the other hand, $d(\Gamma_{b_{j-1}b_j}, \xi(b_{j-1})) \leq 5\epsilon s$. Thus if $\epsilon \leq 1/10$ is chosen small enough, $\Gamma_{b_{j-1}b_j}$ is collision free. When $s = l/2$, $\xi(b_{j-1})$ has a clearance of at least l. This is because a_{j-1} is $3l$-safe. By Lemma 2.4, $\xi(a_{j-1})$ has a clearance of at least l. This implies that $\xi(b_{j-1})$ has a clearance of at least l, since $\xi(b_{j-1}) = \xi(a_{j-1})$. Since $d(\Gamma_{b_{j-1}b_j}, \xi(b_{j-1})) \leq (5/2)\epsilon l$, $\Gamma_{b_{j-1}b_j}$ is collision free if ϵ is chosen small enough.

Now consider the part $\Gamma_{a_{m-1}a_m}$, which lies inside $\xi(a_{m-1})$. When constructing $\mathcal{B}$, $\textsc{loc}(f)$ is considered as an obstacle vertex. Thus the box containing $\textsc{loc}(f)$ and its neighbors all have size l. This means that $\xi(a_{m-1})$ is an extended box consisting of four boxes of size l. By Lemma 2.10, there exists in CT_0 a precomputed trajectory $\Gamma_{b_{m-1}f}$ from b_{m-1} to some $f' \in ngb(f, \epsilon l/2, \epsilon/2)$. Thus an edge from b_{m-1} to f is added in the construction of G. Also $\Gamma_{b_{m-1}f}$ is a $(1,1)$-trajectory whose time length is no more than $(1+\epsilon)\Gamma_{a_{m-1}a_m}$. To show that $\Gamma_{b_{m-1}f}$ is collision free, notice that the clearance of $\xi(a_{m-1})$ is at least l, since a_{m-1} is $3l$-safe. But $d(\Gamma_{b_{m-1}f}, \xi(b_{m-1})) \leq \epsilon l/2$, so this trajectory is also collision free. We can show similar results for $\Gamma_{a_0a_1}$. This completes the proof of this theorem. □

Thus the approximation problem is transformed to one of searching for a shortest path on the graph G. Notice that in order for this algorithm to be correct, each edge introduced in G must correspond to a collision-free trajectory. This is guaranteed by the way we choose the node set V and the connection table properties (a proof similar to the one showing that each $\Gamma_{b_{j-1}b_j}$ is collision free, used in the above theorem). We do not have to explicitly check whether each trajectory segment is collision free. The obtained shortest path corresponds to a $((1+e\epsilon)^4,(1+e\epsilon)^2)$-trajectory. Applying the time-rescaling lemma (Lemma 2.1) with a scaling factor of $(1+e\epsilon)^2$, we can obtain the following.

COROLLARY 2.12. *Let W be a d-dimensional configuration space of size L and Ω a set of obstacles. Let Γ be an optimal $3l$-safe $(1,1)$-trajectory from i to f. We can compute a $(1,1)$-trajectory Γ' from some $i' \in ngb(i, \epsilon l/2, 3e\epsilon)$ to some $f' \in ngb(f, \epsilon l/2, 3e\epsilon)$ such that $T(\Gamma')$ is at most $(1+3e\epsilon)$ times $T(\Gamma)$.*

2.5.2. The time complexity. The running time of the algorithm consists of the following components:

1. Time to generate the graph nodes (i.e., the grid states). If N is the total number of boxes in the final decomposition, the time to perform the decomposition is $O(nN + N\log N)$, where n is the number of constraints defining the configuration obstacles. Since each box contributes at most $O((1/\epsilon)^{2d-1})$ grid states, the time to generate the grid states, after a decomposition, is $O(N(1/\epsilon)^{2d-1})$.
2. Time to compute the graph edges. Since each node is connected to at most $O((1/\epsilon)^{2d-1})$ other nodes, the total number of edges is $O(N(1/\epsilon)^{4d-2})$. This bounds the time to compute the edges, since it takes $O(1)$ time to compute an edge.
3. Time to search for a shortest graph path. Using Dijkstra's algorithm with the priority queue implemented with a binary heap, this time is $O((|V| + |E|)\log|V|)$, where $|V|$ and $|E|$ are the number of nodes and edges, respec-

tively. Plugging in our numbers, the searching time is roughly $O(dN \log N (1/\epsilon)^{4d-2})$.

4. Time to rescale the obtained trajectory to a $(1,1)$-trajectory, which is $O(1)$.

Combining Corollary 2.12 and the time-complexity analysis, and choosing ϵ sufficiently small, we obtain the following result.

THEOREM 2.13. *Fix an $l > 0$ and an $\varepsilon > 0$. After some precomputation, we can achieve the following.*

Let W be a d-dimensional configuration space of size L. Let Ω be a set of configuration obstacles defined by a total of n algebraic equations, each of $O(1)$ degrees. Given any two states i and f, we can compute a collision-free $(1,1)$-trajectory from i' to f' such that the time length of the trajectory is at most $(1+\varepsilon)$ times the time length of an optimal $3l$-safe $(1,1)$-trajectory from i to f. Furthermore, $\|\textsc{loc}(i') - \textsc{loc}(i)\|_\infty \le \varepsilon l$, $\|\textsc{vec}(i') - \textsc{vec}(i)\|_\infty \le \varepsilon$, $\|\textsc{loc}(f') - \textsc{loc}(f)\|_\infty \le \varepsilon l$, and $\|\textsc{vec}(f') - \textsc{vec}(f)\|_\infty \le \varepsilon$.

The running time of our algorithm is $O(nN + N \log N (1/\varepsilon)^{4d-2})$, where $N = O((L/l)^d)$.

2.6. Correcting a trajectory. In this subsection we prove the correcting lemma (Lemma 2.6). The correcting lemma roughly states the following. Let Γ be an $(\bar{a}, \bar{v})$-trajectory from a state i to a state f. Given another pair of states g and h, we can construct a trajectory Γ' from g to h by correcting Γ. Furthermore, we show that if g and h are close to i and f, respectively, the correction is small. Note that in this subsection, the results are presented in the absence of obstacles.

For simplicity of illustration and analysis, we first look at the 1D case. Let $\delta a(t)$ be the corrective acceleration, i.e., $a_{\Gamma'}(t) = a_\Gamma(t) + \delta a(t)$ for $0 \le t \le T(\Gamma)$. Let $\delta v(t) = v_{\Gamma'}(t) - v_\Gamma(t)$ and $\delta p(t) = p_{\Gamma'}(t) - p_\Gamma(t)$. Also let $\Delta v_i = v_{\Gamma'}(0) - v_\Gamma(0) = \textsc{vec}(g) - \textsc{vec}(i)$, $\Delta v_f = v_{\Gamma'}(T) - v_\Gamma(T) = \textsc{vec}(h) - \textsc{vec}(f)$, $\Delta p_i = p_{\Gamma'}(0) - p_\Gamma(0) = \textsc{loc}(g) - \textsc{loc}(i)$, and $\Delta p_f = p_{\Gamma'}(T) - p_\Gamma(T) = \textsc{loc}(h) - \textsc{loc}(f)$. Thus,

$$\begin{aligned}\delta v(t) &= v_{\Gamma'}(0) + \int_0^t (a_\Gamma(\mu) + \delta a(\mu))\, d\mu - \left(v_\Gamma(0) + \int_0^t a_\Gamma(\mu)\, d\mu\right) \\ &= \Delta v_i + \int_0^t \delta a(\mu)\, d\mu\end{aligned}$$

and

$$\begin{aligned}\delta p(t) &= p_{\Gamma'}(0) + \int_0^t (v_\Gamma(\mu) + \delta v(\mu))\, d\mu - \left(p_\Gamma(0) + \int_0^t v_\Gamma(\mu)\, d\mu\right) \\ &= \Delta p_i + \int_0^t \delta v(\mu)\, d\mu \\ &= \Delta p_i + \int_0^t \left(\Delta v_i + \int_0^\mu \delta a(\nu)\, d\nu\right) d\mu \\ &= \Delta p_i + \Delta v_i t + \int_0^t \int_0^\mu \delta a(\nu)\, d\nu\, d\mu.\end{aligned}$$

Our object is to find $\delta a(t)$ such that

$$\Delta v_i + \int_0^T \delta a(\mu)\, d\mu = \Delta v_f \text{ and}$$

$$\Delta p_i + \Delta v_i T + \int_0^T \int_0^\mu \delta a(\nu)\, d\nu\, d\mu = \Delta p_f,$$

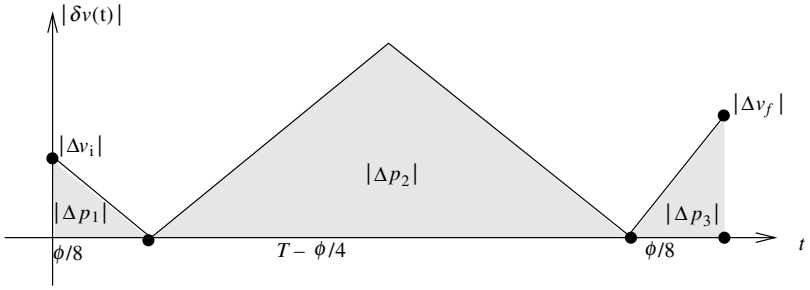

FIG. 2.4. *The correcting scheme.*

where $T = T(\Gamma)$. We also want to keep the absolute values of $\delta v(t)$, $\delta a(t)$, and $\delta p(t)$ small.

Fix a constant $c \geq 1$ and let

$$\rho_c = \bar{v}^2/(c\bar{a}).$$

Assume that

$$|\text{LOC}(f) - \text{LOC}(i)| \geq \rho \geq \rho_c$$

for some ρ, and let

$$\phi = \rho/\bar{v}.$$

Thus $T(\Gamma) \geq \phi$. Our correcting scheme is illustrated in Figure 2.4. Basically, we correct in three phases. In the first phase, we use a constant corrective acceleration to make $\delta v(t)$ become 0, while in the last phase, we use a constant corrective acceleration to make $\delta v(t)$ become Δv_f. The middle phase is used to correct the distance. Notice that at the beginning and the end of the second phase, $\delta v(t) = 0$.

The time length of the first and the last phases is $\phi/8$. Let Δa_1 be the constant corrective acceleration used in the first phase. Then

$$\Delta a_1 = -\frac{\Delta v_i}{\phi/8} = -\frac{8\Delta v_i}{\phi}.$$

If Δp_1 is the distance covered in this phase, then $\Delta p_1 = \Delta v_i \phi/16$. In this phase, $|\delta v(t)|$ is getting smaller and is upper bounded by $|\Delta v_i|$, while $|\delta p(t)|$ is upper bounded by $|\Delta p_i| + |\Delta p_1|$. Let Δa_3 be the constant corrective acceleration used in the last phase. Similarly, we have

$$\Delta a_3 = \frac{8\Delta v_f}{\phi}.$$

$\Delta p_3 = \Delta v_f \phi/16$ is the distance covered in this phase. In this phase, $|\delta v(t)|$ is upper bounded by $|\Delta v_f|$. At the beginning of this phase, $\delta p(t) = \Delta p_f - \Delta p_3$. Thus $|\delta p(t)|$ is upper bounded by $|\Delta p_f| + |\Delta p_3|$ during this phase.

Let $T' = T(\Gamma) - \phi/4$ be the time spent in the second phase. This phase is divided into two subphases of equal length. The corrective accelerations used in these two subphases have the same absolute value but opposite directions. The effect is that at the end of this phase, $\delta v(t)$ becomes 0 again. If Δp_2 is the distance covered in this

phase, $\Delta p_2 = \Delta p_f - \Delta p_i - \Delta p_1 - \Delta p_3$. Notice that $|\Delta p_2|$ and $\delta p(t)$ are both upper bounded by $|\Delta p_i| + |\Delta p_1| + |\Delta p_3| + |\Delta p_f|$. Let Δa_2 be the corrective acceleration used in the first subphase. Then

$$\Delta a_2 = \frac{4\Delta p_2}{(T')^2}.$$

In this phase, $|\delta v(t)|$ is upper bounded by $2|\Delta p_2|/T'$.

LEMMA 2.14. *For any $\varepsilon > 0$ and any two states g and h, if $g \in ngb(i, \varepsilon\rho/2, \varepsilon\bar{v}/2)$ and $h \in ngb(f, \varepsilon\rho/2, \varepsilon\bar{v}/2)$, where ρ is defined as above, then the trajectory Γ', constructed as above, satisfies the following properties:*

P1. $T(\Gamma') = T(\Gamma)$.

P2. Γ' *is a* $((1+4c\varepsilon)^2\bar{a}, (1+4c\varepsilon)\bar{v})$*-trajectory.*

P3. $d(\Gamma', \Gamma) \leq (17/16)\varepsilon\rho$.

Proof. P1 holds obviously. During the whole time period $T(\Gamma)$, the maximum difference in position is upper bounded by

$$\begin{aligned}
|\Delta p_i| + |\Delta p_1| + |\Delta p_3| + |\Delta p_f| &\leq \frac{1}{2}\varepsilon\rho + \frac{|\Delta v_i|\phi}{16} + \frac{|\Delta v_f|\phi}{16} + \frac{1}{2}\varepsilon\rho \\
&\leq \varepsilon\rho + \frac{(\varepsilon\bar{v})(\rho/\bar{v})}{16} \\
&\leq \left(1 + \frac{1}{16}\right)\varepsilon\rho \\
&\leq \frac{17}{16}\varepsilon\rho.
\end{aligned}$$

This implies that $|p_{\Gamma'}(t) - p_\Gamma(t)| \leq (17/16)\varepsilon\rho$ for $0 \leq t \leq T(\Gamma)$. By definition, $d(\Gamma', \Gamma) \leq (17/16)\varepsilon\rho$, proving P3. The maximum difference in velocity is upper bounded by

$$\begin{aligned}
\max\left(|\Delta v_i|, |\Delta v_f|, \frac{2|\Delta p_2|}{T'}\right) &\leq \max\left(\frac{\varepsilon\bar{v}}{2}, \frac{2(17/16)\varepsilon\rho}{(1-1/4)\phi}\right) \\
&\leq \max\left(\frac{\varepsilon\bar{v}}{2}, \frac{3\varepsilon\rho}{\phi}\right) \\
&\leq \max\left(\frac{\varepsilon\bar{v}}{2}, 3\varepsilon\bar{v}\right) \\
&\leq 4c\varepsilon\bar{v}.
\end{aligned}$$

The maximum difference in acceleration is upper bounded by

$$\begin{aligned}
\max\left(|\Delta a_1|, |\Delta a_2|, |\Delta a_3|\right) &\leq \max\left(\frac{8|\Delta v_i|}{\phi}, \frac{4|\Delta p_2|}{(T')^2}, \frac{8|\Delta v_f|}{\phi}\right) \\
&\leq \max\left(\frac{4\varepsilon\bar{v}}{\phi}, \frac{4(17/16)\varepsilon\rho}{((3/4)\phi)^2}\right) \\
&\leq \max\left(\frac{4\varepsilon\bar{v}}{\phi}, \frac{8\varepsilon\bar{v}}{\phi}\right) \\
&= \frac{8\varepsilon\bar{v}}{\phi} \\
&= \frac{8\varepsilon\bar{v}^2}{\rho}
\end{aligned}$$

$$\leq \frac{8\varepsilon\bar{v}^2}{\rho_c}$$
$$= 8c\varepsilon\bar{a}.$$

Thus the velocity of Γ' is upper bounded by $(1+4c\varepsilon)\bar{v}$ and the acceleration is bounded by $(1+4c\varepsilon)^2\bar{a}$, proving P2. □

In higher dimensions, we can add corrective accelerations for each dimension separately. Since the time length of the trajectory is not changed, the corrections can be carried out simultaneously without affecting each other. Let $v_\Gamma^j(t)$ be $v_\Gamma(t)$ projected in the jth dimension for $1 \leq j \leq d$. By Lemma 2.14, we have

$$|v_{\Gamma'}^j(t)| \leq 4c\varepsilon\bar{v} + |v_\Gamma^j(t)|.$$

By the triangle inequality,

$$\begin{aligned}\|v_{\Gamma'}(t)\| &\leq \|v_\Gamma(t)\| + \|v_{\Gamma'}(t) - v_\Gamma(t)\| \\ &\leq \bar{v} + \sqrt{\sum_{j=1}^{d}(v_{\Gamma'}^j(t) - v_\Gamma^j(t))^2} \\ &\leq \bar{v} + \sqrt{\sum_{j=1}^{d}(4c\varepsilon\bar{v})^2} \\ &\leq \bar{v} + 4c\sqrt{d}\varepsilon\bar{v} \\ &= (1 + 4c\sqrt{d}\varepsilon)\bar{v}.\end{aligned}$$

Similarly for the acceleration, we can obtain that

$$\|a_{\Gamma'}(t)\| \leq (1 + 4c\sqrt{d}\varepsilon)^2\bar{a}.$$

In summary, we have Lemma 2.6, reproduced here as Lemma 2.15.

LEMMA 2.15 (correcting lemma). *Fix a constant $c \geq 1$ and let $\rho_c = \bar{v}^2/(c\bar{a})$, where $\bar{a}, \bar{v} > 0$ are arbitrary. Let Γ be an $(\bar{a}, \bar{v})$-trajectory from a state i to a state f. Given any $\rho > 0$ and $\varepsilon > 0$ and given any two states g and h, if $\rho > \rho_c$, $\|\text{LOC}(f) - \text{LOC}(i)\|_\infty \geq \rho$, $g \in ngb(i, \varepsilon\rho/2, \varepsilon\bar{v}/2)$, and $h \in ngb(f, \varepsilon\rho/2, \varepsilon\bar{v}/2)$, then we can construct a trajectory Γ' from g to h by correcting Γ such that Γ' satisfies the following properties:*

P1. $T(\Gamma') = T(\Gamma)$.

P2. *Γ' is a $((1 + 4c\sqrt{d}\varepsilon)^2\bar{a}, (1 + 4c\sqrt{d}\varepsilon)\bar{v})$-trajectory.*

P3. $d(\Gamma', \Gamma) \leq (17/16)\varepsilon\rho$.

2.7. Scaling the dynamic bounds. So far, our presentation considers only the case when both the accelerations and the velocities are upper bounded by 1. This is sufficiently general because, as we mentioned in section 1.3, we can scale arbitrarily given bounds to 1 by scaling time and the configuration space. In this section, we will show how the scaling is done. Basically, for given arbitrary bounds, we scale the configuration space (also the locations and the velocities of the initial and final states) by some appropriate factors decided by the given bounds. We then apply the algorithm described in section 2.5 to compute a trajectory in the scaled configuration space with both dynamic bounds set to 1. Finally we "scale back" the computed trajectory. We will see that the scaled trajectory abides by the given dynamic bounds and is a close approximation to an optimal trajectory.

For a path Π, we use $L(\Pi)$ to denote the path length of Π. Let Π and Π' be two paths. We say that $\Pi' = \alpha\Pi$, or that Π' is Π scaled by a factor of α, if $L(\Pi') = \alpha\, L(\Pi)$ and $\Pi'(\ell) = \alpha\, \Pi(\ell/\alpha)$ for $0 \le \ell \le \alpha\, L(\Pi)$.

In the following, we first present the velocity-scaling lemma, which shows the scaling relation between the velocity and the time and the space. Next, the acceleration-scaling lemma shows how to scale the acceleration by scaling the time and the space. By combining these two lemmas, we obtain the acceleration-velocity scaling lemma. Note that unlike the time-rescaling lemma (Lemma 2.1), the acceleration and the velocity can be scaled by different and unrelated factors in our lemma.

LEMMA 2.16 (velocity-scaling lemma). *Let Γ be an $(\bar{a}, \bar{v})$-trajectory and Π be its path. The path $(1/\alpha^2)\Pi$ can be traversed by an $(\bar{a}, \bar{v}/\alpha)$-trajectory Γ' in time $T(\Gamma)/\alpha$. Moreover Γ is an optimal $(\bar{a}, \bar{v})$-trajectory if and only if Γ' is an optimal $(\bar{a}, \bar{v}/\alpha)$-trajectory.*

Proof. Define Γ' to be the following: $p_{\Gamma'}(0) = p_\Gamma(0)/\alpha^2$, $v_{\Gamma'}(0) = v_\Gamma(0)/\alpha$, and $a_{\Gamma'}(t) = a_\Gamma(\alpha t)$, $0 \le t \le T(\Gamma)/\alpha$. We will show that, for $0 \le t \le T(\Gamma)/\alpha$,

$$v_{\Gamma'}(t) = v_\Gamma(\alpha t)/\alpha \text{ and}$$

$$p_{\Gamma'}(t) = p_\Gamma(\alpha t)/\alpha^2.$$

This is because

$$\begin{aligned} v_{\Gamma'}(t) &= v_{\Gamma'}(0) + \int_0^t a_{\Gamma'}(s)\, ds \\ &= v_\Gamma(0)/\alpha + \int_0^t a_\Gamma(\alpha s)\, ds \\ &= v_\Gamma(0)/\alpha + \int_0^{\alpha t} a_\Gamma(s')(1/\alpha)\, ds' \\ &= v_\Gamma(0)/\alpha + \left(\int_0^{\alpha t} a_\Gamma(s')\, ds' \right) /\alpha \\ &= v_\Gamma(\alpha t)/\alpha \end{aligned}$$

and

$$\begin{aligned} p_{\Gamma'}(t) &= p_{\Gamma'}(0) + \int_0^t v_{\Gamma'}(s)\, ds \\ &= p_\Gamma(0)/\alpha^2 + \int_0^t v_\Gamma(\alpha s)/\alpha\, ds \\ &= p_\Gamma(0)/\alpha^2 + \int_0^{\alpha t} (v_\Gamma(s')/\alpha^2)\, ds' \\ &= p_\Gamma(0)/\alpha^2 + (1/\alpha^2) \int_0^{\alpha t} v_\Gamma(s')\, ds' \\ &= p_\Gamma(\alpha t)/\alpha^2. \end{aligned}$$

Let Π' be the path traversed by Γ', and let ℓ and t be such that$\Pi'(\ell) = p_{\Gamma'}(t)$.

Thus

$$\begin{aligned}\ell &= \int_0^t v_{\Gamma'}(s)ds \\ &= \int_0^t v_\Gamma(\alpha s)/\alpha ds \\ &= (1/\alpha^2)\int_0^{\alpha t} v_\Gamma(s)ds.\end{aligned}$$

This implies that $\Pi(\alpha^2\ell) = p_\Gamma(\alpha t)$. Since

$$\begin{aligned}\Pi'(\ell) &= p_{\Gamma'}(t) \\ &= p_\Gamma(\alpha t)/\alpha^2 \\ &= \Pi(\alpha^2\ell)/\alpha^2,\end{aligned}$$

therefore Π' is $(1/\alpha^2)\Pi$. The scaling preserves the optimality. This completes the proof of the lemma. □

LEMMA 2.17 (acceleration-scaling lemma). *Let Γ be an $(\bar{a}, \bar{v})$-trajectory and Π be its path. The path $(1/\alpha)\Pi$ can be traversed by an $(\bar{a}/\alpha, \bar{v}/\alpha)$-trajectory Γ' in time $T(\Gamma)$. Moreover Γ is an optimal $(\bar{a}, \bar{v})$-trajectory if and only if Γ' is an optimal $(\bar{a}/\alpha, \bar{v}/\alpha)$-trajectory.*

Proof. Define Γ' to be the following: $p_{\Gamma'}(0) = p_\Gamma(0)/\alpha$, $v_\Gamma(0) = v_\Gamma(0)/\alpha$, and $a_{\Gamma'}(t) = a_\Gamma(t)/\alpha$, $0 \le t \le T(\Gamma)$. We can show that, for any $0 \le t \le T(\Gamma)$,

$$v_{\Gamma'}(t) = v_\Gamma(t)/\alpha \text{ and}$$

$$p_{\Gamma'}(t) = p_\Gamma(t)/\alpha.$$

This is because

$$\begin{aligned}v_{\Gamma'}(t) &= v_{\Gamma'}(0) + \int_0^t a_{\Gamma'}(s)\, ds \\ &= v_\Gamma(0)/\alpha + \int_0^t a_\Gamma(s)/\alpha\, ds \\ &= v_\Gamma(0)/\alpha + \left(\int_0^t a_\Gamma(s)\, ds\right)/\alpha \\ &= v_\Gamma(t)/\alpha\end{aligned}$$

and

$$\begin{aligned}p_{\Gamma'}(t) &= p_{\Gamma'}(0) + \int_0^t v_{\Gamma'}(s)\, ds \\ &= p_\Gamma(0)/\alpha + \int_0^t v_\Gamma(s)/\alpha\, ds \\ &= p_\Gamma(0)/\alpha + \left(\int_0^t v_\Gamma(s)\, ds\right)/\alpha \\ &= p_\Gamma(t)/\alpha.\end{aligned}$$

Let Π' be the path traversed by Γ'. Similarly to the proof of Lemma 2.16, we can show that $\Pi' = (1/\alpha)\Pi$. This completes the proof of the lemma. □

LEMMA 2.18 (acceleration-velocity scaling lemma). *Let Γ be an $(\bar{a}, \bar{v})$-trajectory and let Π be its path. The path $(\beta/\alpha^2)\Pi$ can be traversed by an $(\bar{a}/\beta, \bar{v}/\alpha)$-trajectory Γ' in time $(\beta/\alpha)\,T(\Gamma)$. Moreover Γ is an optimal $(\bar{a}, \bar{v})$-trajectory if and only if Γ' is an optimal $(\bar{a}/\beta, \bar{v}/\alpha)$-trajectory.*

Proof. Let $\Pi'' = (1/\beta)\Pi$. By Lemma 2.17, Π'' can be traversed by an $(\bar{a}/\beta, \bar{v}/\beta)$-trajectory Γ'' in time $T(\Gamma)$.

Let $\gamma = \alpha/\beta$ and let $\Pi' = (1/\gamma^2)\Pi''$. By Lemma 2.16, Π' can be traversed by an $(\bar{a}/\beta, \bar{v}/(\beta\gamma))$-trajectory in time $T(\Gamma'')/\gamma$. Since

$$\Pi' = (1/\gamma^2)\Pi'' = (1/\gamma^2)(1/\beta)\Pi = (\beta/\alpha^2)\Pi,$$

$$\bar{v}/(\beta\gamma) = \bar{v}/\alpha,$$

and

$$T(\Gamma'')/\gamma = T(\Gamma)/\gamma = (\beta/\alpha)T(\Gamma),$$

therefore this trajectory is the one desired. □

By applying the acceleration-velocity scaling lemma, we obtain the following corollary.

COROLLARY 2.19. *Given a kinodynamic motion-planning problem with the following parameters: the size of the configuration space L, the dynamic bounds $\bar{a}$ and $\bar{v}$, the two parameters l and ε, the initial state i, and the final state f, we can scale the problem to one with $\bar{a} = 1$ and $\bar{v} = 1$ by scaling the initial and final velocities by a factor of $(1/\bar{v})$ and scaling the configuration space (including the obstacles, the parameter l, $\text{LOC}(i)$, and $\text{LOC}(f)$) by a factor of $(\bar{a}/\bar{v}^2)$. Let Γ be the trajectory obtained for the scaled problem. Then the trajectory Γ' such that*

$$a_{\Gamma'}(t) = \bar{a} \cdot a_{\Gamma}((\bar{a}/\bar{v})t)$$

for $0 \le t \le (\bar{v}/\bar{a})T(\Gamma)$ is the solution for the original problem.

3. The curvature-constrained shortest-path problem in three and higher dimensions.

3.1. Introduction. Let $P : I \to \mathbb{R}^d$ be a d-dimensional differentiable path parameterized by arclength $s \in I$. The *average curvature* of P in the interval $[s_1, s_2] \subseteq I$ is defined by $\|\dot{P}(s_1) - \dot{P}(s_2)\|/|s_1 - s_2|$. In the curvature-constrained shortest-path problem, we require that the path have an average curvature of at most 1 in every interval. Again, we can always scale an arbitrarily given curvature bound to 1 by scaling the configuration space, thus it is general enough to consider the case when the bound is 1. Notice that we use average curvature instead of curvature because the curvature of a differentiable path may not exist at certain points. Also, the curvature of a path, wherever it is defined, is bounded by 1 if and only if its average curvature is bounded by 1 for all intervals.

A *position* X is a pair $(\text{LOC}(X), \text{ORN}(X))$, where $\text{LOC}(X)$ is a point in the d-dimensional space and $\text{ORN}(X)$ is a vector representing an orientation in the d-dimensional space. Notice that $\|\text{ORN}(X)\| = 1$. Given a set Ω of obstacles in a d-dimensional space, an initial position I, and a final position F, the curvature-constrained shortest-path problem is to find a shortest path from I to F such that the path obeys the curvature constraint and does not intersect Ω.

It has been observed in [18] that the curvature-constrained shortest-path problem is a restricted case of the kinodynamic motion-planning problem, with the L_2-norms of the velocities fixed to be 1. However, if we require that the velocities of the approximation trajectories be fixed in L_2-norm, the techniques developed so far for the general kinodynamic case cannot be applied to this restricted case. As pointed out in [12], a necessary condition for the techniques to apply is that the set of *feasible instantaneous accelerations* (accelerations that can be applied without violating the dynamic constraints) spans d dimensions. On the other hand, if the velocities are fixed in L_2-norm, the set of instantaneous accelerations spans only $d-1$ dimensions, because they have to be perpendicular to the instantaneous velocity.

Our contribution is that we look at the curvature-constrained shortest-path problem from a different viewpoint, which enables us to overcome the difficulty mentioned above. Basically, instead of requiring that the L_2-norms of the velocities be fixed to be 1, we only force them to fall within a small range close to 1. In this way, we are able to obtain a path whose maximum curvature is slightly larger than, but can be arbitrarily close to, 1. It should be noted that under this same variation, the former approximation algorithms of Donald and Xavier [15] and Reif and Tate [43] can also be applied to compute approximating solutions for the curvature-constrained shortest-path problem.

In this section, we use lowercase letters to denote states as defined for the kinodynamic case, but uppercase letters (usually X, Y, U, V) to denote positions (i.e., $\|\textsc{orn}(X)\| = \|\textsc{orn}(Y)\| = \|\textsc{orn}(U)\| = \|\textsc{orn}(V)\| = 1$). A path is called a *c-constrained* path if its average curvature is at most c. We say that Γ is a $(1, \tilde{1})$-trajectory if $\|a_\Gamma(t)\| \leq 1$ and $\|v_\Gamma(t)\| = 1$ for $0 \leq t \leq T(\Gamma)$. Given a path Π, let $L(\Pi)$ be its path length. Notice that the notation of *ngb* extends to positions. Therefore, for a position X, a position $X' \in ngb(X, \rho, \nu)$ if $\|\textsc{loc}(X') - \textsc{loc}(X)\|_\infty \leq \rho$ and $\|\textsc{orn}(X') - \textsc{orn}(X)\|_\infty \leq \nu$.

3.2. Approximating in the absence of obstacles. The following lemma states a result similar to Corollary 2.2 but for the curvature-constrained case. This lemma enables us to apply the method developed in the previous sections to the curvature-constrained shortest-path problem.

LEMMA 3.1. *Let W be a d-dimensional configuration space of size L. Let Π be a 1-constrained path from position X to position Y and lying inside W. Given any $\varepsilon > 0$ and $\rho = O(1)$, we can compute in time $O(L^d(1/\varepsilon)^{6d-1})$ a path Π' such that Π' satisfies the following properties:*

P1. *Π' is $(1+\varepsilon)$-constrained.*

P2. *$L(\Pi') \leq (1+\varepsilon)L(\Pi)$.*

P3. *$d(\Pi', W) \leq \varepsilon\rho/2$.*

P4. *Π' is from some position $X' \in ngb(X, \varepsilon\rho/2, \varepsilon/2)$ to some position $Y' \in ngb(Y, \varepsilon\rho/2, \varepsilon/2)$.*

Proof. Let $\delta = \varepsilon/8$, $\eta_x = \varepsilon\rho/2$, and $\eta_v = \varepsilon/(8\sqrt{d})$.

We consider Π to be the path of a trajectory Γ such that Γ is from state $i = (\textsc{loc}(X), \textsc{orn}(X)/(1+\delta))$ to state $f = (\textsc{loc}(Y), \textsc{orn}(Y)/(1+\delta))$ and $\|v_\Gamma(t)\| = 1/(1+\delta)$. Notice that $T(\Gamma) = (1+\delta)\,L(\Pi)$. Since Π is a 1-constrained path, and at any time t, the curvature of Π is given by $\|a_\Gamma(t)\|/\|v_\Gamma(t)\|^2$, the acceleration of Γ is bounded by

$$\|a_\Gamma(t)\| \leq 1 \cdot \|v_\Gamma(t)\|^2 \leq 1/(1+\delta)^2.$$

This implies that Γ is a $(1/(1+\delta)^2, 1/(1+\delta))$-trajectory. Let $\mathcal{A}_\delta$ be the set of accel-

erations whose L_2-norms are bounded by $1/(1+\delta)^2$. This is the set of accelerations used by Γ.

Let $\mu = \kappa_l = \delta/(4\sqrt{d})$ and $\mathcal{A}_\mu$ be the set of accelerations as defined in the appendix (see (A.5) and (A.6)). Thus $\mathcal{A}_\mu$ has a uniform advantage of κ_l over $\mathcal{A}_\delta$. By the tracking lemma (see the appendix), there exists a $\tau = O(\varepsilon)$ (satisfying (A.4)) and a τ-bang trajectory Γ' using $\mathcal{A}_\mu$ such that $T(\Gamma') = T(\Gamma)$ and Γ' tracks Γ to a tolerance of (η_x, η_v). This implies that

$$\|v_{\Gamma'}(t) - v_\Gamma(t)\|_\infty \le \eta_v,$$

$$\|v_{\Gamma'}(t) - v_\Gamma(t)\| \le \sqrt{d}\eta_v,$$

$$\|v_\Gamma(t)\| - \sqrt{d}\eta_v \le \|v_{\Gamma'}(t)\| \le \|v_\Gamma(t)\| + \sqrt{d}\eta_v,$$

$$\frac{1}{1+\delta} - \sqrt{d}\frac{\varepsilon}{8\sqrt{d}} \le \|v_{\Gamma'}(t)\| \le \frac{1}{1+\delta} + \sqrt{d}\frac{\varepsilon}{8\sqrt{d}},$$

$$\frac{1}{1+\delta} - \frac{\varepsilon}{8} \le \|v_{\Gamma'}(t)\| \le \frac{1}{1+\delta} + \frac{\varepsilon}{8},$$

$$1 - \frac{\varepsilon}{4} \le \|v_{\Gamma'}(t)\| \le 1 + \frac{\varepsilon}{8}.$$

Let Π' be the path of trajectory Γ'. We will show that Π' satisfies P1–P4. Since $\|a_{\Gamma'}(t)\| \le 1$, this bounds the maximum curvature of Π' to be at most

$$\frac{1}{(1-\varepsilon/4)^2} \le 1 + \varepsilon,$$

proving P1. Since $T(\Gamma') \le T(\Gamma) = (1+\delta)L(\Pi)$, then

$$L(\Pi') \le \left(1 + \frac{\varepsilon}{8}\right) T(\Gamma') \le \left(1 + \frac{\varepsilon}{8}\right)(1+\delta)L(\Pi) \le (1+\varepsilon)L(\Pi),$$

proving P2, where $(1 + \frac{\varepsilon}{8})$ is the upper bound on $\|v_{\Gamma'}(t)\|$. P3 follows directly from the fact that Γ' tracks Γ to a tolerance of (η_x, η_v) with $\eta_x = \varepsilon\rho/2$, and that Γ lies inside W.

Let i' and f' be the initial and the final states of Γ', and X' and Y' be the initial and the final positions of Π'. Thus $\textsc{loc}(X') = \textsc{loc}(i')$, $\textsc{loc}(Y') = \textsc{loc}(f')$, $\textsc{orn}(X') = \textsc{vec}(i')/\|\textsc{vec}(i')\|$, and $\textsc{orn}(Y') = \textsc{vec}(f')/\|\textsc{vec}(f')\|$, and

$$\begin{aligned}
&\|\textsc{orn}(X') - \textsc{orn}(X)\|_\infty \\
&\le \|\textsc{orn}(X') - \textsc{orn}(X)\| \\
&= \|\textsc{orn}(X') - \textsc{vec}(i') + \textsc{vec}(i') - \textsc{vec}(i) + \textsc{vec}(i) - \textsc{orn}(X)\| \\
&\le \|\textsc{orn}(X') - \textsc{vec}(i')\| + \|\textsc{vec}(i') - \textsc{vec}(i)\| + \|\textsc{vec}(i) - \textsc{orn}(X)\| \\
&\le (1 - \|\textsc{vec}(i')\|) + \sqrt{d}\eta_v + \frac{\delta}{1+\delta} \\
&\le \frac{\varepsilon}{4} + \frac{\varepsilon}{8} + \frac{\varepsilon}{8} \\
&= \frac{\varepsilon}{2}.
\end{aligned}$$

Similarly we can show that $\|\textsc{orn}(Y') - \textsc{orn}(Y)\|_\infty \le \varepsilon/2$. This proves P4 for Γ'.

Such a trajectory Γ' (and thus a path Π') can be obtained by the TC-graph method. In constructing the graph, an edge is added if (i) it is a (μ, τ)-bang, (ii) the bang does not diverge from W by more than $\varepsilon\rho/2$, and (iii) the L_2-norms of the velocities of the bang fall into $[1-\varepsilon/4, 1+\varepsilon/8]$. Since μ and τ are $O(\varepsilon)$, the number of edges in this graph is bounded by $O(L^d(1/\varepsilon)^{6d-1})$, and thus so is the running time. This completes the proof of the lemma. □

Notice that we do not explicitly constrain the velocities of the tracking trajectory Γ'. This allows us to apply the tracking lemma (Lemma A.1). However, it happens that $\|v_{\Gamma'}(t)\|$ falls into a small range of $[1-\varepsilon/4, 1+\varepsilon/8]$ by being able to track closely a trajectory Γ whose velocities are fixed to be 1 in L_2-norm. By upper bounding the accelerations and lower bounding the velocities of Γ', we are able to bound the curvature of Π'.

3.3. Approximating with obstacles. Let W be a d-dimensional configuration space of size L with a set Ω of obstacles. Let $\mathcal{B}$ be the box decomposition of W, as described in section 2.3.1. Let Π be an optimal $3l$-safe 1-constrained path from position X to position Y.

As we did in the proof of the safe loose-tracking theorem (Theorem 2.11), we can divide Π into segments of paths, $\Pi_{U_0U_1} \| \dots \| \Pi_{U_{m-1}U_m}$, such that $U_0 = X$, $U_m = Y$, and each U_i is the position where Π first exits $\xi(U_{i-1})$ for $1 \le i \le m$. Each segment $\Pi_{U_iU_{i+1}}$ is a 1-constrained path from U_i to U_{i+1} and lies inside $\xi(U_i)$. These are the paths we need to approximate.

We define the set of CS grid states, the canonical extended boxes, the canonical starting states, and the canonical ending states in the same way as we did in the previous sections, except that we let $\mathcal{V}$ be the set of *unit* vectors that are uniformly spaced with spacing δ. If $\delta = O(\epsilon)$ is chosen small enough, for any unit vector v, there is a unit vector $v' \in \mathcal{V}$ such that $\|v' - v\|_\infty \le \epsilon/2$. Thus $\mathcal{V}$ suffices for the curvature-constrained case, since if we consider U_i as a state then $\|\textsc{vec}(U_i)\| = 1$ for all U_i's along Π. The following corollary can be derived from the correcting lemma (Lemma 2.6).

Corollary 3.2. *Fix a constant $c > 1$ and let $\rho_c = 1/(c\kappa)$, where $\kappa > 0$ is arbitrary. Let Π be a κ-constrained path from position I to position F. Given any $\rho > 0$ and $\varepsilon > 0$ and given any two positions U and V, if $\rho > \rho_c$, $\|\textsc{loc}(F) - \textsc{loc}(I)\|_\infty \ge \rho$, $U \in ngb(I, \varepsilon\rho/2, \varepsilon/2)$, and $V \in ngb(F, \varepsilon\rho/2, \varepsilon/2)$, we can construct a path Π' from U to V by correcting Π such that Π' satisfies the following properties:*

P1. $L(\Pi') = (1 + e\varepsilon)L(\Pi)$,

P2. Π' *is* $((1+e\varepsilon)^2/(1-e\varepsilon)^2)\kappa$*-constrained,*

P3. $d(\Pi', \Pi) \le (17/16)\varepsilon\rho$,

where $e = 4c\sqrt{d}$.

The consequence of Lemma 3.1 and Corollary 3.2 is that we are able to precompute paths which approximate U_{ii+1}'s and store them in the connection tables. Similar to what we did for the kinodynamic case, we can prove a version of the loose-tracking lemma (Lemma 2.8) and the safe loose-tracking theorem (Theorem 2.11) for the curvature-constrained case. Since $|\mathcal{V}| = O((1/\epsilon)^{d-1})$ (as opposed to $O((1/\epsilon)^d)$ for the kinodynamic case), the number of edges in the CS graph is reduced to $O((1/\epsilon)^{4d-4})$.

Combining the above, and choosing small enough $\epsilon = O(\varepsilon)$, we can obtain the following result.

Theorem 3.3. *Fix an $l > 0$ and an $\varepsilon > 0$. After some precomputation, we can achieve the following.*

Let W be a d-dimensional space of size L and let Ω be a set of obstacles with a

total of n vertices. Given any two positions X and Y, we can compute a collision-free $(1+\varepsilon)$-constrained path from a position $X' \in ngb(X, \varepsilon l, \varepsilon)$ to a position $Y' \in ngb(Y, \varepsilon l, \varepsilon)$, such that the path length is at most $(1+\varepsilon)$ times the length of an optimal 1-constrained $3l$-safe path from X to Y.

The running time of our algorithm is $O(nN + N \log N (1/\varepsilon)^{4d-4})$, where $N = O((L/l)^d)$.

4. Conclusions. In this paper we have presented a faster approximation algorithm for the kinodynamic motion-planning problem. Contrary to the previous approximation algorithms which all use a uniform discretization in time, our method employs a nonuniform discretization in the configuration space. The discretization is nonuniform in that it is coarser in regions which are farther from all obstacles. The nonuniform discretization leads to a smaller search space and thus a faster algorithm. Moreover, our method is sensitive to the distribution of obstacles in the configuration space. It is expected to perform better (i.e., faster) than the given theoretical time bound in cases where the obstacles are sparsely or unevenly distributed. In developing the approximation algorithm, we utilize a hierarchical decomposition of the configuration space. We also developed the idea of using trajectories precomputed in the absence of obstacles as building blocks to construct trajectories in the presence of obstacles. Finally, we have applied this algorithm to give the first-known polynomial-time approximation algorithm for the curvature-constrained shortest-path problem in three and higher dimensions. The computed paths may have curvatures slightly larger (but can be arbitrarily close to) the given curvature bound.

It should be pointed out that, though improved, the time complexity of our approximation algorithm is still too high for the method to have real-world applications. At this point, the proposed method and results are of pure theoretical interest.

One future research direction is to give a more precise bound on N, the number of boxes in the box decomposition. Instead of describing N as a function of L, the size of the configuration space, we would prefer to describe N as a function of some parameters that describe the obstacle scene, for example, the aspect ratios of the obstacles.

In the kinodynamic motion-planning problem, we only considered *Cartesian robots*, i.e., robots whose inertia tensor is constant (see [15] for a more precise definition). We believe that our approach can also be applied to more general robots with *open kinematic chains*. In order to do this, we need to generalize our correcting lemma (Lemma 2.6) to open chains.

A key further problem is to determine cases of kinodynamic motion-planning problems that we can solve in closed forms, or for which we can give sufficient characterizations for developing fast algorithms. An example along this line is the case of planning for a point robot moving on a plane amid polygonal obstacles and with decoupled kinodynamic constraints. It is characterized in [9] that the minimum-time trajectory is a sequence of segments, where each segment is a "bang-bang" control between two obstacle boundary points. It is interesting to investigate whether such characterizations extend to coupled cases and higher dimensions.

The complexity of the kinodynamic motion-planning problem in two dimensions is still open.

Appendix. The *TC*-graph method. For the sake of completeness, we describe the gist of the tracking lemma (Lemma A.1) and the TC-graph method in this section.

A trajectory Γ' is said to *track* another trajectory Γ to a tolerance (η_x, η_v) if for

all $t \in [0, T]$,

$$\|p_\Gamma(t) - p_{\Gamma'}(t)\|_\infty \leq \eta_x \text{ and} \tag{A.1}$$

$$\|v_\Gamma(t) - v_{\Gamma'}(t)\|_\infty \leq \eta_v. \tag{A.2}$$

Notice that (A.1) implies that $d(\Gamma', \Gamma) \leq \eta_x$.

A *τ-bang* is a trajectory segment of time duration τ during which a *constant* acceleration is applied.[2] A *τ-bang trajectory* is a trajectory consisting of a sequence of τ-bangs. The set of bang trajectories is a restricted set of trajectories. But they suffice to track any nonrestricted trajectories if the acceleration set used by the bang trajectories has some *advantage* over the acceleration set used by the nonrestricted trajectories. Next we define the notion of advantage more formally.

Let $\mathcal{P}$ and $\mathcal{Q}$ be two sets of accelerations. For any acceleration $a \in \mathcal{P}$ and any direction given by a d-length vector σ of 1's and -1's, if there exists an acceleration $b \in \mathcal{Q}$ such that

$$\sigma^j(b^j - a^j) \geq \kappa_l \tag{A.3}$$

for $1 \leq j \leq d$, then $\mathcal{Q}$ is said to have a *uniform κ_l advantage* over $\mathcal{P}$ (α^j means the jth element of a vector α). The tracking lemma relates the parameter τ to the uniform advantage κ_l and the tracking tolerance (η_x, η_v).

LEMMA A.1 (tracking lemma [15]).[3] *Let $\mathcal{P}$ and $\mathcal{Q}$ be two sets of accelerations such that for each $a \in \mathcal{Q}$, $\|a\| \leq 1$, and $\mathcal{Q}$ has a uniform κ_l advantage over $\mathcal{P}$.*

Let Γ be a trajectory that uses $\mathcal{P}$. Let (η_x, η_v) be a tracking tolerance. There exists a time step τ, and a τ-bang trajectory Γ' that uses $\mathcal{Q}$ such that Γ' tracks Γ to tolerance (η_x, η_v).

Moreover, it is sufficient that

$$\tau = O(\min(\eta_v, \sqrt{\eta_x \kappa_l})). \tag{A.4}$$

Let $\mathcal{A}$ (resp., $\mathcal{A}_\epsilon$) be the set of accelerations whose L_2-norms are bounded by 1 (resp., $1/(1+\epsilon)^2$). Next we show how we can choose a finite set of accelerations such that this set has a uniform advantage over $\mathcal{A}_\epsilon$. Given a parameter μ, a set of grid-points of $\mathcal{A}$ with spacing μ is defined to be

$$\{a = (i_1\mu, \ldots, i_d\mu) \mid i_1, \ldots, i_d = 0, \pm 1, \pm 2, \ldots, a \in \mathcal{A}\}. \tag{A.5}$$

Let $\mathcal{A}_\mu$ be the set of *boundary* grid-points[4] (see Figure A.1; the dark dots represent the boundary grid-points in two dimensions). It is shown in [15] that if μ is chosen

[2]Here we slightly abuse the notion of a *bang*. A bang usually means that its acceleration should be *extremal* in some sense.

[3]This is a simplified version of the tracking lemma presented in [15]. In [15], $\mathcal{Q}$ is a set of *instantaneous accelerations* that depends on the current state, and may be a subset of the set of allowed accelerations. This is because applying certain accelerations may violate the velocity constraint. In the context where we apply this tracking lemma, we do not explicitly bound the velocities of the tracking trajectory. This means that the set of instantaneous accelerations equals the set of all allowed accelerations.

[4]In [15], two approximation algorithms, the true-extremal algorithm and the near-extremal algorithm, are described. They differ in the way of choosing a discretized acceleration set and in the way of defining bangs. The description here follows from the near-extremal algorithm.

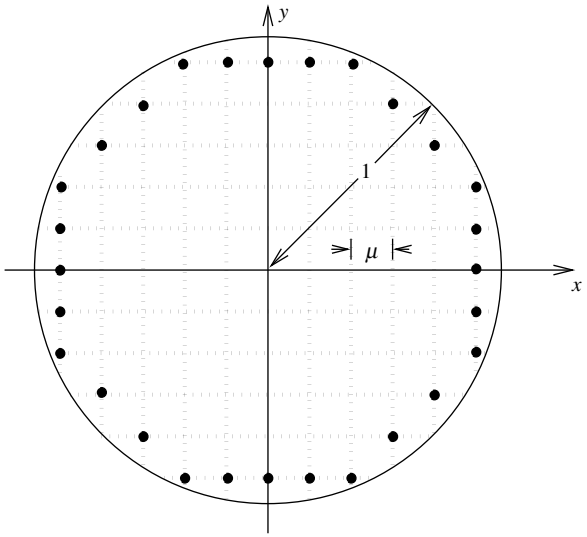

FIG. A.1. *The acceleration set $\mathcal{A}_\mu$ in two dimensions.*

such that

$$\mu \le \kappa_l \le \frac{\epsilon}{4\sqrt{d}}, \tag{A.6}$$

then $\mathcal{A}_\mu$ has a uniform advantage of κ_l over $\mathcal{A}_\epsilon$. Notice that $|\mathcal{A}_\mu| = O(d(1/\mu)^{d-1})$.

A τ-bang (resp., τ-bang trajectory) is a (μ,τ)-*bang* (resp., (μ,τ)-*bang trajectory*) if the acceleration set used is $\mathcal{A}_\mu$. Given a tracking tolerance (η_x,η_v) and any $\epsilon > 0$, there exist μ (satisfying (A.6)) and τ (satisfying (A.4)) such that for any $(1/(1+\epsilon)^2, 1/(1+\epsilon))$-trajectory Γ, there exists a (μ,τ)-bang trajectory Γ' (also a $(1,1)$-trajectory) that tracks Γ to a tolerance of (η_x,η_v). To approximate any given $(1,1)$-trajectory Γ to within a factor of $(1+\epsilon)$ in time length, we first time-rescale Γ to a $(1/(1+\epsilon)^2, 1/(1+\epsilon))$-trajectory Γ', with time length extended by a factor of $(1+\epsilon)$. Since there exists a (μ,τ)-bang trajectory Γ'' that tracks Γ' to a tolerance of (η_x,η_v), Γ'' approximates Γ in that $T(\Gamma'') \le (1+\epsilon)\,T(\Gamma)$. But Γ'' does not track Γ to the tolerance of (η_x,η_v). However, $d(\Gamma'',\Gamma) \le \eta_x$. This is because Γ' and Γ trace the same path and $d(\Gamma'',\Gamma') \le \eta_x$. Also the two end states of Γ'' are close to the two end states of Γ, respectively.

Having discovered the existence of a tracking bang trajectory, the next question is how to compute one. What makes it more difficult is that most of the time, the original $(1,1)$-trajectory Γ is not given except in its initial and final states. The *TC*-graph method transforms this problem to that of finding a shortest path in a directed graph. (In the following, we only describe the *TC*-graph method in the absence of obstacles, where we do not have to do collision-free checking.) The *TC* graph G is roughly as follows: (i) the *root node* of G approximates the initial state of Γ; (ii) a directed edge corresponds to a (μ,τ)-bang whose velocities are also bounded by 1 in L_2-norm; the weight of the edge is always τ; (iii) the graph is generated and explored from the root node in a breadth-first manner, and the search terminates when either a node approximating the final state is found or when no new nodes aregenerated.

Note the following:

- The size of G is finite (see the following analysis).
- The breadth-first search produces a trajectory whose time length is no more than $(1+\epsilon)\,T(\Gamma)$. The breadth-first search suffices for finding a shortest path since each edge has a uniform weight of τ.
- Suppose that Γ lies within a subset W of the configuration space. Since there exists a bang trajectory Γ'' satisfying $d(\Gamma'', \Gamma') \leq \eta_x$, then $d(\Gamma'', W) \leq \eta_x$. We can find such a trajectory by considering only those edges whose corresponding bangs do not diverge from W by more than η_x. Notice that it still holds that $T(\Gamma'') \leq (1+\epsilon)\,T(\Gamma)$.

Each node of G corresponds to a state. By carefully choosing the initial velocity, we can bound the number of possible velocities by $O((1/(\mu\tau))^d)$ and the number of possible locations by $O((L/(\mu\tau^2))^d)$, where L is the size of W. Thus the total number of nodes is bounded by

$$O\left(\left(\frac{L}{\mu^2\tau^3}\right)^d\right). \tag{A.7}$$

Each node can have at most $O(d(1/\mu)^{d-1})$ outgoing edges (because it can have at most this many choices of accelerations), thus the total number of graph edges is

$$O\left(\frac{dL^d}{\mu^{3d-1}\tau^{3d}}\right). \tag{A.8}$$

This also bounds the running time.

If we set

$$\eta_x = \epsilon\rho/2 \text{ and } \eta_v = \epsilon/2, \tag{A.9}$$

where $\rho = O(1)$, we obtain Corollary 2.2. Notice that since $\tau = O(\epsilon)$ and $\mu = O(\epsilon)$, the running time is $O(L^d(1/\epsilon)^{6d-1})$.

REFERENCES

[1] P. K. Agarwal, P. Raghavan, and H. Tamak, *Motion planning for a steering-constrained robot through moderate obstacles*, Proc. Annual Symposium Theory Comput., 27 (1995), pp. 343–352.

[2] J. Barraquand and J. C. Latombe, *Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles*, Algorithmica, 10 (1993), pp. 121–155.

[3] J. D. Boissonnat and X. N. Bui, *Accessibility Region for a Car that Only Moves Forwards along Optimal Paths*, Technical report, INRIA, Sophia Antipolis, France, 1994.

[4] J. D. Boissonnat, A. Cerezo, and J. Leblond, *Shortest paths of bounded curvature in the plane*, in Proceedings of the IEEE International Conference on Robotics and Automation, Nice, France, 1992, pp. 2315–2320.

[5] J. D. Boissonnat, A. Cerezo, and J. Leblond, *A Note on Shortest Paths in the Plane Subject to a Constraint on the Derivative of the Curvature*, Technical report, INRIA, Sophia Antipolis, France, 1994.

[6] X. N. Bui, J. D. Boissonnat, P. Soueres, and J. P. Laumond, *Shortest path synthesis for Dubins non-holonomic robot*, in Proceedings of the IEEE International Conference on Robotics and Automation, San Diego, CA, 1994, pp. 2–7.

[7] J. Canny, *Some algebraic and geometric configuration in* PSPACE, Proc. Annual ACM Sympos. Theory Comput., 20 (1988), pp. 460–467.

[8] J. Canny, B. Donald, J. Reif, and P. Xavier, *On the complexity of kinodynamic planning*, Proc. Sympos. Found. Comput. Sci., 29 (1988), pp. 306–316.

[9] J. Canny, A. Rege, and J. Reif, *An exact algorithm for kinodynamic planning in the plane*, Discrete Comput. Geom., 6 (1991), pp. 461–484.
[10] J. Canny and J. Reif, *New lower bound techniques for robot motion planning problems*, Proc. Sympos. Found. Comput. Sci., 28 (1987), pp. 49–60.
[11] B. Donald and P. Xavier, *A provably good approximation algorithm for optimal-time trajectory planning*, in Proceedings of the IEEE International Conference on Robotics and Automation, Scottsdale, AZ, 1989, pp. 958–963.
[12] B. Donald and P. Xavier, *Near-optimal kinodynamic planning for robots with coupled dynamics bounds*, in Proceedings of the IEEE International Symposium on Intelligent Controls, Albany, NY, 1989, pp. 354–359.
[13] B. Donald and P. Xavier, *Provably good approximation algorithms for optimal kinodynamic planning for Cartesian robots and open chain manipulators*, Proc. Sympos. Comput. Geom., 6 (1990), pp. 290–300.
[14] B. Donald and P. Xavier, *Provably good approximation algorithms for optimal kinodynamic planning: Robots with decoupled dynamics bounds*, Algorithmica, 14 (1995), pp. 443–479.
[15] B. Donald and P. Xavier, *Provably good approximation algorithms for optimal kinodynamic planning for cartesian robots and open chain manipulators*, Algorithmica, 14 (1995), pp. 480–530.
[16] B. Donald, P. Xavier, J. Canny, and J. Reif, *Kinodynamic motion planning*, J. Assoc. Comput. Mach., 40 (1993), pp. 1048–1066.
[17] L. E. Dubins, *On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents*, Amer. J. Math., 79 (1957), pp. 497–516.
[18] S. Fortune and G. Wilfong, *Planning constrained motion*, Ann. Math. Artificial Intelligence, 3 (1991), pp. 21–82.
[19] T. Fraichard, *Smooth trajectory planning for a car in a structured world*, in Proceedings of the IEEE International Conference on Robotics and Automation, Sacramento, CA, 1991, pp. 2–7.
[20] G. Heinzinger, P. Jacobs, J. Canny, and B. Paden, *Time-optimal trajectories for a robot manipulator: a provably good approximation algorithm*, in Proceedings of the IEEE International Conference on Robotics and Automation, Cincinnati, OH, 1990, pp. 150–155.
[21] J. Hershberger and S. Suri, *Efficient computation of Euclidean shortest paths in the plane*, in Proceedings of the 34th Symposium on Foundations of Computer Science, Palo Alto, CA, 1993, pp. 508–517.
[22] J. M. Hollerbach, *Dynamic scaling of manipulator trajectories*, in Proceedings Automatic Control Council, 1983, pp. 752–756.
[23] P. Jacobs and J. Canny, *Planning smooth paths for mobile robots*, in Nonholonomic Motion Planning, Kluwer Academic Publishers, Norwell, MA, 1992.
[24] P. Jacobs, J. P. Laumond, and M. Taix, *Efficient motion planners for nonholonomic mobile robots*, in Proceedings of the IEEE/RSJ International Workshop on Intelligent Robots and Systems, Osaka, Japan, 1991, pp. 1229–1235.
[25] L. Kavraki and J. C. Latombe, *Randomized preprocessing of configuration space for fast path planning*, in Proceedings of the IEEE International Conference on Robotics and Automation, San Diego, CA, 1994, pp. 2138–2145.
[26] L. Kavraki, J. C. Latombe, R. Motwani, and P. Raghavan, *Randomized query processing in robot path planning*, in Proceedings of the 27th Sympos. Theory Computing, Las Vegas, NV, 1995, pp. 353–362.
[27] V. P. Kostov and E. V. Degtiariova-Kostova, *Suboptimal Paths in the Problem of a Planar Motion with Bounded Derivative of the Curvature*, Technical report, INRIA, Cedex, France, 1993.
[28] V. P. Kostov and E. V. Degtiariova-Kostova, *The Planar Motion with Bounded Derivative of the Curvature and Its Suboptimal Paths*, Technical report, INRIA, Cedex, France, 1994.
[29] J. C. Latombe, *A fast path-planner for a car-like indoor mobile robot*, in Proceedings of the Ninth National Conference on Artificial Intelligence, Anaheim, CA, 1991, pp. 659–665.
[30] J. C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Norwell, MA, 1991.
[31] J. P. Laumond, *Finding collision free smooth trajectories for a nonholonomic mobile robot*, Proc. Internat. Joint Conf. Artificial Intelligence, 10 (1987), pp. 1120–1123.
[32] J. P. Laumond, P. E. Jacobs, M. Taix, and R. M. Murray, *A motion planner for nonholonomic mobile robots*, IEEE Trans. Robotics and Automation, 10 (1994), pp. 577–593.
[33] J. P. Laumond and T. Simeon, *Motion Planning for a Two Degrees of Freedom Mobile Robot with Towing*, Technical report, LAAS/CNRS, LAAS, Toulouse, France, 1989.
[34] J. P. Laumond, M. Taix, and P. Jacobs, *A motion planner for car-like robots based on a*

global/local approach, in Proc. IEEE/RSJ International Workshop on Intelligent Robots and Systems, Tsuchira, Japan, 1990, pp. 765–773.

[35] G. L. Miller, D. Talmor, S. Teng, and N. Walkington, *A Delaunay based numerical method for three dimensions: Generation, formulation, and partition*, Proc. Sympos. on Theory of Computing, 27 (1995), pp. 683–692.

[36] G. L. Miller, S. Teng, W. Thurston, and S. A. Vavasis, *Automatic mesh partitioning*, in Graph Theory and Sparse Matrix Computation, A. George, J. Gilbert, and J. Liu, eds., Springer-Verlag, New York, 1993, pp. 57–84.

[37] B. Mirtich and J. Canny, *Using skeletons for nonholonomic path planning among obstacles*, in Proceedings of the IEEE International Conference on Robotics and Automation, Nice, France, 1992, pp. 2533–2540.

[38] J. S. B. Mitchell, D. M. Mount, and S. Suri, *Query-sensitive ray shooting*, Proc. Sympos. Comput. Geom., 10 (1994), pp. 359–368.

[39] Y. Nakamura and R. Mukherjee, *Nonholonomic path planning and automation*, in Proceedings of the IEEE International Conference on Robotics and Automation, Scottsdale, AZ, 1989, pp. 1050–1055.

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

[41] J. A. Reeds and L. A. Shepp, *Optimal paths for a car that goes both forwards and backwards*, Pacific J. Math., 145 (1990), pp. 367–393.

[42] J. Reif, *Complexity of the generalized movers problem*, in Planning, Geometry and Complexity of Robot Motion, J. Hopcroft, J. Schwartz, and M. Sharir, eds., Ablex, Norwood, NJ, 1987, pp. 267–281.

[43] J. H. Reif and S. R. Tate, *Approximate kinodynamic planning using L_2-norm dynamic bounds*, Comput. Math. Appl., 27 (1994), pp. 29–44.

[44] G. Sahar and J. M. Hollerbach, *Planning of Minimum-Time Trajectories for Robot Arms*, Technical report, MIT, Cambridge, MA, 1984.

[45] J. T. Schwartz and M. Sharir, *Algorithmic motion planning in robotics*, in Algorithms and Complexity, J. V. Leeuwen, ed., Handbook of Theoretical Computer Science, Vol. A, Elsevier, Amsterdam, 1990, pp. 391–430.

[46] H. Wang and P. K. Agarwal, *Approximation algorithms for curvature-constrained shortest paths*, in Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, Atlanta, GA, 1996, pp. 409–418.

[47] G. Wilfong, *Motion planning for an autonomous vehicle*, in Proceedings of the IEEE International Conference on Robotics and Automation, 1988, pp. 529–533.

[48] G. Wilfong, *Shortest paths for autonomous vehicles*, in Proceedings of the IEEE International Conference on Robotics and Automation, 1989, pp. 15–20.

[49] P. Xavier, *private communication*, 1998.