Metric cells: towards complete search for optimal trajectories

Devin Balkcom[1], Ajay Kannan[2], Yu-Han Lyu[3], Weifu Wang[4], Yinan Zhang[5]

***Abstract*— This paper presents a definition of convexity useful for describing local optimality in configuration spaces, proves that finding convex regions is relatively easy, and presents an algorithm for approximating the free configuration space using a set of such convex regions. The paper examines simple but interesting systems: serial planar arms with revolute joints, and a Reeds-Shepp car. The paper experimentally explores an approach for finding good (although not necessarily optimal) trajectories using the derived data structure.**

I. Introduction

This paper examines how an *optimal steering method*, which finds optimal trajectories if there are no obstacles, may be used to build a good approximate cell-based representation of a configuration space with obstacles. The paper also presents algorithms that make use of this representation to find good, although not necessarily optimal, paths.

Approaches to finding optimal paths among obstacles can be loosely classified into two types: cell decomposition (for example, early work by Barraquand and Latombe [1] and Xavier and Donald [2]), and sampling methods, such as PRM* [3]. Cell-based methods have the advantage that it can be easier to prove results about path quality after finite computation time, but typically require division of the space into a number of cells that is explicitly exponential in the dimensionality of the space. Sampling methods, on the other hand, can quickly return reasonable paths if there are large open spaces in the environment, even in high dimensions.

The current paper begins to explore the idea that some easy spaces (with large open regions, or large obstacle regions) can be represented easily, without giving up optimality, using a variable-sized cell decomposition approach. Figures **??** and 1 show an example of such a decomposition for the configuration space for a serial planar arm with two revolute joints, as well as a path obtained by a simple A* search across points on the boundaries of the cells. Note that because optimal trajectories are not unique for the chosen metric (Section III-A) some "wiggle" can occur even in optimal trajectories.

Variable-sized cell-decomposition methods are hardly new; the unique contribution of this paper is an exploration of the implications of optimality under some metric, for systems with and without non-holonomic constraints.

The key idea is to exploit convexity of regions in configuration space. Intuitively, between any pair of boundary points of a convex region, there should exist a path that does not leave the region. So, paths within the region need not be sampled or stored.

[1,2,3,4,5] Department of Computer Science, Dartmouth College
*This work was supported in part by NSF grant IIS-0643476.

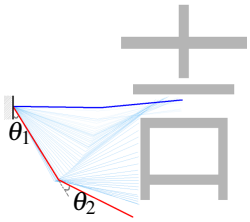


Fig. 1: Initial and final positions are represented by thick lines, while others are intermediate positions of the arm.

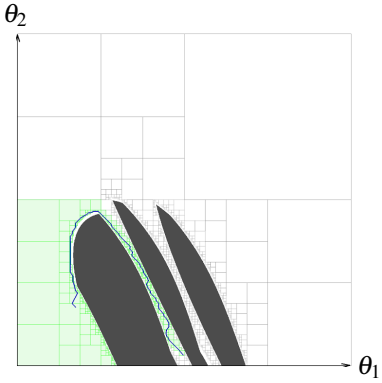


Fig. 2: Decomposition of C-space and the corresponding trajectory in C-space. Shaded cubes are explored by the search algorithm.

In this paper, we show that a form of convexity can be defined (Section II) that is suitable for configuration spaces with optimal steering methods and corresponding shortest-path metrics, and that a variant on this form of convexity, subconvexity, is available locally everywhere there is a reachable ball in the free configuration space. We consider some example systems, including a planar arm and a Reeds-Shepp car [4], and present an algorithm for decomposing the free configuration space into cells (Section IV). We search this data structure for good paths (Section VI), and show some results for a few simple systems.

The work in this paper is quite preliminary, but we believe that the theoretical framework hints at new ways to tackle issues such as ensuring good path quality after finite computation time (*vs.* weaker guarantees such as asymptotic optimality), topological analysis of configuration spaces, and compact representations of configuration spaces. Concretely, the current work suggests some bounds on how densely a space may need to be sampled to capture optimality over

"most of" the configuration space. We believe that this is an important step to begin to understand the behavior of sampling-based planners near obstacles (a primary focus of, for example, variations such as obstacle-based PRM [5] and Toggle PRM [6]), after finite computation time.

A. Related work

One inspiration for this paper is early work by Xavier and Donald [2] that showed that shortest paths can be provably approximated for a particle among obstacles in R^3 subject to kinodynamic constraints; by growing the obstacles slightly, a lower bound on the required size of cells used to represent configuration space can be computed. Similarly, in collision detection, *temporal coherence* is frequently used to compute the frequency at which collisions need to be checked for; LaValle [7] provides a survey.

Recently, Bialkowski *et al.* have reduced the time cost of collision detection with RRT*, by building balls in free (Euclidean) space in which collision detection needs to be performed only once [8]. The current paper extends these ideas to compute a nearly complete representation of the space, under a general shortest-path metric.

Recent sampling-based algorithms by Li, Littlefield, and Bekris [9] achieve asymptotic optimality of path length in parameter space without the need for a steering method. Deits shows a numerical optimization approach to computing large convex regions, also in a Euclidean space [10]; a key contribution of the current work is extending this notion of convexity to non-Euclidean metrics.

Extensive work has been done on *optimal steering methods*. In fact, for some specific systems, the exact optimal trajectories can be found analytically, or by relatively efficient algorithms [11], [4], [12], [13], [14]. Work by Venditelli [15] demonstrated that for the Dubins car [11], the shortest distance to obstacles in configuration space can be computed exactly; the current paper is strongly motivated by the idea of finding bounds on such distance functions, in a way that can be easily extended to a variety of systems.

Lafferriere and Sussman [16] and others [17] have used the property of small-time local controllability (STLC) to approximate paths that do not respect differential constraints by paths that do; the definition of shortest-path subconvexity in this paper is inspired by the "inner ball/outer ball" aspect of the definition of STLC. The Ball-Box Theorem [18] shows that, given a locally-defined metric satisfying Lipschitz continuity properties, the reachable configurations from a given point contain and are contained by a pair of boxes in the configuration space parameters.

II. Path convexity and subconvexity

In this paper, we assume that an optimal steering method is available, together with a shortest-path local metric, d, that would describe the minimum cost of traveling between two configurations if there were no obstacles. The steering method and metric might be known exactly, or in practice, might be computed approximately using numerical techniques. Metrics are typically only available for symmetric systems.

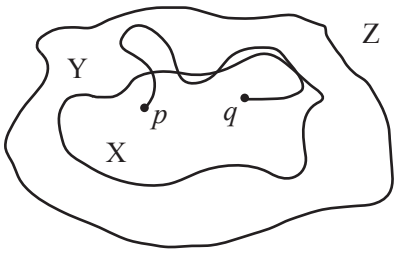


Fig. 3: Subconvexity of set X with respect to set Y, under an optimal steering method. The path between p and q does not leave Y.

The central idea is to cover most of the obstacle-free portion of the configuration space, C_{f}, with some convex closed cells, so that optimal paths cross through boundary points of the cells, and are well-behaved within the cells. However, for systems with a given metric, or subject to non-holonomic constraints, the straight lines used to define convexity in the usual sense might not be geodesics or even feasible.

It is natural to define convexity in terms of the steering method for the system. Let a steering method S of a metric space be a family of continuous curves (parametrized by arc length) such that for each ordered pair of points (a,b) in the metric space there is a corresponding curve in S from a to b.

We say that a set X of a metric space with steering method S is **path convex** if between any two points in X, the corresponding path from S is contained entirely within X. (This definition is somewhat related to definitions of *geodesic convexity* described in [19].)

As a simple example, let the metric space be the surface of a solid globe that has been cut in half through the equator, with the metric defined by the shortest path along the surface, measured using the Euclidean distance in R^3. Consider an optimal steering method S that contains some shortest path on the surface for each pair of points.

First consider the set F, the flat disc where the globe was cut. The shortest paths between points in F are straight lines, and they do not leave the disc, so F is path convex (under S). Now consider P, consisting of all points on the surface within 15 degrees of the pole. The shortest paths between points in P are arcs of great circles on the sphere and those arcs do not leave P, so P is path convex.

Finally, consider the set H, the entire curved hemisphere within 90 degrees of the pole. For some pairs of points in H, the connecting shortest path is an arc of a circle contained in H. But for other points (along the equator, for example), the connecting shortest path takes a shortcut through F, leaving H. So H is not a path convex set.

For the purpose of motion planning, we would like to be able to place a convex cell almost anywhere in the space, but for many systems, path convexity is too strong a requirement. We cannot cover H with path convex cells contained within H under an optimal steering method, so if there are obstacles

in F, motion planning requires knowledge of F even if we care only about planning paths between points in H.

Given sets X and Y of a metric space, under a steering method S, we say that X is **path subconvex to** Y **under** S if between any two points in X, the corresponding path in S is contained within Y. For the purpose of motion planning, Y may be an obstacle-free region large enough to allow optimal maneuvering of the system between points of X.

A closed metric ball of radius c at a point x in a metric space Z under metric d, defined in the usual way as $B_c^d[x] = \{z \in Z : d(x,z) \leq c\}$, represents the set of points reachable from x with cost no greater than c. Although we might like to sample some points using closed metric balls at those points to cover some part of the free configuration space, we can see from the half-globe example (H is a closed metric ball centered at the pole) that a closed metric ball is not necessarily path convex with respect to an optimal steering method; the shortest path between points on the boundary might leave the ball, into regions we don't necessarily know anything about.

Although a closed metric ball (reachable set) at a sampled point might not be path convex, we can find a pair of balls at a any point in C_{f} such that the inner, smaller ball is subconvex to the outer:

Theorem 1: Given an optimal steering method S, a corresponding metric d over a metric space Z, a point $x \in Z$, and a positive constant r, the closed metric ball $B_{r/2}^d[x]$ is path subconvex to $B_r^d[x]$ under S.

Proof: Consider two arbitrary points p and q in $B_{r/2}^d[x]$, and a postulated shortest path between them. Let m be an arbitrary point along this shortest path. We will show that $d(x,m)$ is no larger than r, implying that the entire path is contained within the larger ball. By the triangle inequality,

$$d(x,m) \leq d(x,p) + d(p,m) \quad (1)$$
$$d(x,m) \leq d(x,q) + d(q,m). \quad (2)$$

Summing 1 and 2,

$$2d(x,m) \leq d(x,p) + d(x,q) + d(p,m) + d(m,q). \quad (3)$$

Since $d(x,p)$ and $d(p,m)$ are each less than or equal to $r/2$,

$$2d(x,m) \leq r + d(p,m) + d(m,q). \quad (4)$$

Since the path from p to q through m is a shortest path, $d(p,m) + d(m,q)$ must be less than or equal to the length of the path from p to q through x. Therefore, $d(p,m) + d(m,q) \leq r$. Combining with Inequality 4, $d(x,m) \leq r$. ■

III. Computing reachable balls

From a particular configuration, how far, under a given metric, can the robot or system travel before hitting an obstacle? We expect this question to be hard to answer in the configuration space, since we do not typically know the shapes of configuration-space obstacles, and since we do not necessarily expect to even know the shapes of metric balls in configuration space. However, it is perhaps good enough to find a conservative estimate of the size of the safely-reachable ball, and this section will show that computing such an estimate might not be too difficult.

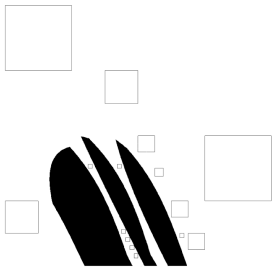


Fig. 4: Different size cells at various configurations in the configuration space of the 2R arm.

Many robotic systems of interest are embedded in two- or three-dimensional Euclidean workspaces, where geometric quantities like distance are much easier to measure. Recall a few classical definitions related to configuration space [20]. A *system* is a collection of particles embedded in a space, perhaps R^2 or R^3. We assume that the system is divided into two closed sets: the robot(s), which we control, and obstacles, which we don't. A *configuration* gives the locations of all particles. There are typically constraints on the possible configurations of the particles; the *configuration space* is the space of configurations satisfying the constraints.

In general, assume we have a configuration space parametrized by a vector $q \in R^n$ (or by an overlapping set of such parametrizations, an *atlas*). We will further assume that the parameters are bounded; let C_{f} be a bounded subset of R^n, or a finite collection of such subsets representing possible values of the parameters for which there is no collision.

In order to compute a lower bound on how much the configuration can change before a collision, we relax constraints on how particles can move. At a particular configuration, there is some minimum Euclidean distance from the obstacles, over all particles, $e(q)$. We also expect that, over all particles, and over all possible configurations, there is some maximum rate of change of location of any particle (measured by Euclidean distance), $v_{\max}$, with respect to a unit rate of change of the metric. Then define

$$d_{\mathrm{safe}}(q) = e(q)/v_{\max}, \quad (5)$$

the lower bound on the change in the metric before a collision; different sizes of cells are show in Figure 3.

A. Example: 2R planar arm

As a concrete example, consider a planar, serial robot arm with a fixed base, and two links that are each line segments of length one, shown in Figure 1. Let the configuration space be parametrized by $q = (\theta_1, \theta_2)$, and for simplicity, place joint limits such that each parameter falls in the range $[0, \pi]$.

We need a locally accurate metric d, which will describe distances in the configuration space with obstacles removed, together with an optimal steering method. Motivated by the observation that if each joint has the same constant

upper bound on velocity, the time cost of moving from one configuration to another is determined by the joint that needs to move the farthest, define

$$d(q,q') = \max(|\theta_1' - \theta_1|, |\theta_2' - \theta_2|). \quad (6)$$

For this simple example, we know the shape of a metric ball: a square in the parameter space (θ_1, θ_2), possibly cut by the joint limits. One simple optimal steering method would be to move the joint that has farther to travel at maximum velocity towards the goal angle, and the other joint at a scaled rate, so that both joints reach their final angles at the same time: a line segment in the parameter space. The metric ball is path convex with respect to this steering method.

If we sample a point in the parameter space (θ_1, θ_2), how big of a ball can we place? We can compute the minimum Euclidean distance of points in the arm from the obstacles, $e(q)$, easily. We can also compute $v_{\max}$, the maximum rate of change of $e(q)$ over all trajectories in the configuration space, if configuration-space trajectories are parametrized by arc length measured by the metric. Notice that the "fastest" particle, over all particles, over all configurations, and over all unit-metric velocities of the joints is the point at the far tip of the arm, when the arm is fully extended, as the two joints move in the same direction at equal velocity. Simple differential kinematics indicate that the maximum speed of this particle is 3. So for this arm, $d_{\text{safe}}(q) = e(q)/3$.

B. Reachable balls for systems with Lipschitz continuity

In general, in order to ensure that a $v_{\max}$ exists that gives an upper bound on the rate of change of distances in the workspace with respect to the metric, we can verify that two properties hold for the system, metric, and steering method. First, that there exists a Lipschitz constant that bounds the workspace velocity (over all particles) with respect to change in configuration-space coordinates. Second, that there exists a Lipschitz constant that bounds the rate of change of each of the configuration-space coordinates with respect to the metric along any path returned by the steering method. The upper bound $v_{\max}$ can then be computed as the product of the Lipschitz constants.

IV. Safe covers of free configuration space

There are two structures of interest: the free configuration space C_{f}, which we are trying to approximate, and which will contain the union of "outer balls" described in Theorem 1, and a slightly smaller subset C_ε, all configurations that have distance at least $\varepsilon > 0$ to obstacles in workspace, which we will actually cover with cells that are subconvex with respect to C_{f}. We will call this collection of cells a *safe cover*, since paths from the steering method between boundary points of the cells remain safely within C_{f}.

The shape of a metric ball in the configuration space is typically hard to obtain (although this was easy for the toy example of the serial arm), and it can be hard to understand how metric balls overlap and connect. We would also like to sample the configuration space efficiently, and not place new samples inside cells that have already been explored. Thus, in this section, we show how to construct a cover of C_ε by a set of hypercubes that only overlap along their boundaries; each hypercube is path subconvex with respect to a subset of C_{f}.

A. Reachable hypercubes

Given the maximum velocity, $v_{\max}$, of the robot over all configurations, at any configuration q not in contact with a workspace obstacle ($e(q) > 0$), we can compute a safe metric ball centered at q, and by Theorem 1, a subconvex inner ball with half the radius. We would like to place a hypercube entirely within this subconvex ball.

In general, given a ball of radius r centered at a configuration q, how large a hypercube (also centered at q) can we place within the ball? Let $c_h = \{p \in C_{\text{f}} : p_i \in [q_i - h, q_i + h], \forall 1 \le i \le n\}$ be a hypercube with side length $2h$. We need to choose h so that every point of c_h is reachable in time (or more generally, metric cost) r.

For some systems, there may be particularly good methods for computing such an h. Here is a method which is perhaps more conservative than we might like, but which is a fairly general approach.

Let a coordinate-steering function S_i be a steering method that provides a path between a configuration q and a configuration $q + (0, 0, \ldots, x, 0, 0)$, such that the result of the motion is a change only in the ith parameter. (In the simplest case, each S_i might simply be the optimal steering method that we assume is available for the system, S.) Let $d_i(x)$ be the cost of applying steering method S_i in coordinate direction i to move a coordinate-distance x. For a symmetric system, if

$$\sum_{i=1}^{n} d_i(h) \le r, \quad (7)$$

and if each $d_i(x)$ is non-decreasing in x, then $c_h \subset B_r$. We can find a suitable h value by binary search on h, recomputing costs of coordinate steering methods and adding them, and checking if 7 is satisfied.

For many systems, including systems without nonholonomic constraints, the Reeds-Shepp car, and kinematic differential-drive models, each $d_i(x)$ is non-decreasing. We must note that for non-STLC systems, such as the Dubins car, this is not the case.

B. Uniform grid cover

A simple approach to covering the space is to divide the space into cells of equal size placed on a uniform grid. We want the cells to cover C_ε, while being completely contained within (and at least subconvex to) C_{f}. The following observation indicates that small enough cubes, at configurations for which the workspace distance from obstacles is small enough, are not part of C_ε, and may be safely ignored.

Theorem 2: Let $e(q)$ be the minimum Euclidean distance to obstacles and $2h(e(q))$ be the side length of a subconvex hypercube inside the metric ball $B^d_{e(q)/v_{\max}}[q]$ computed in Section IV-A. For any number $\varepsilon > 0$, and any hypercube Q centered at q with side length $2s$, if $e(q) < \varepsilon/2$ and $s < h(\varepsilon/2)$, then $Q \cap C_\varepsilon = \emptyset$.

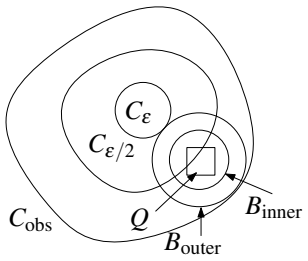


Fig. 5: Illustration of proof of Theorem 2.

Proof: Since $e(q) < \varepsilon/2$, the ball $B_{\text{outer}} = B^d_{\varepsilon/(2v_{\max})}[q]$ is in C_{f}. Consequently, the ball $B_{\text{inner}} = B^d_{\varepsilon/(4v_{\max})}[q]$ does not intersect with C_ε and is subconvex with respect to B_{outer}. If Q's side length is at most $h(\varepsilon/2)$, then Q must lie in the ball B_{inner} by construction and hence $Q \cap C_\varepsilon = \emptyset$; see Figure 4. ■

Theorem 2 suggests a naïve algorithm: partition the configuration space by a set of hypercubes with side length $h(\varepsilon/2)$ and discard all hypercubes with centers with Euclidean distance less than $\varepsilon/2$ to obstacles in the workspace. Although this naïve algorithm creates a safe cover of C_ε, the algorithm is not efficient, since the algorithm generates hypercubes with a uniform size.

C. Cube subdivision cover

In order to obtain some larger hypercubes, we use a recursive approach: for a hypercube Q, if Q is subconvex with respect to C_{f}, then we keep this hypercube. Otherwise, subdivide Q into 2^n sub-hypercubes and find a cover recursively. By Theorem 2, we can stop subdivision when the center has Euclidean distance less than $\varepsilon/2$ and has a size smaller than $h(\varepsilon/2)$. The result is a quadtree-like structure in n dimensions; Figure 1 shows a simple example for the 2R arm.

The crucial part is to test if some cell is subconvex with respect to C_{f}, without false positives. One way to test if a cell is subconvex is to compute a reachable ball centered at the center of the cell. Then divide the radius of this ball in half, to get a subconvex ball. Finally, compute the size h of a hypercube that fits in the smaller ball, and compare h to the size of the cell.

Algorithm 1 shows the approach, which builds the cell cover in depth-first order. Alternatively, one could explore the cover in a breadth-first order, so that large cells would be explored first, and the space could be constructed in an on-line fashion.

We use the 2R planar arm as an example. Since any hypercube in the free space of the arm is convex, testing the subconvexity of a hypercube only requires checking whether the hypercube is collision-free. Because reachable balls are themselves hypercubes for this simple example, if the side length of Q is smaller than $d_{\text{safe}}(q)$, then Q must be collision-free. Thus, the decomposition algorithm can be applied very simply to the 2R planar arm system; see Figure 1.

Algorithm 1: cubeCover

input : Configuration space C, error parameter ε
output: A cover of C_ε by a set of hypercubes.
Let q be the center of C and $2s$ be the side length of C.
if *C passed subconvexity test* **then**
  return a hypercube of C.
else if $e(q) < \varepsilon/2$ *and* $s < h(\varepsilon/2)$ **then**
  return $\emptyset$.
else
  Divide C into 2^n hypercubes and recurse.
  return the union of all results.

D. Covering collision space

Even if sampling a hypercube center results in an actual collision, Algorithm 1 subdivides that cube, since the hypercube may contain smaller hypercubes that are collision free and convex, or subconvex to C_{f}, with respect to the steering method. This is problematic, since the effect is that all of collision space, the part of C inside obstacles, is divided into hypercubes of size that may be as small as $h(\varepsilon/2)$, typically dominating computational and space costs for the algorithm.

Fortunately, there is a relatively easy solution. For each sampled hypercube center that results in a collision, we may compute a conservative bound on *penetration depth*: the minimum distance (as measured by d in the configuration space) required to escape the collision space. We use a technique similar to that used to compute bounds on distances to obstacles in configuration space. First, for a given configuration, we compute the maximum Euclidean distance, over all points on the robot, to the surface of the obstacle in the workspace, e_{escape}, a quantity analogous to $e(q)$. Then the penetration distance is at least equal to $d_{\text{escape}} = e_{\text{escape}}(q)/v_{\max}$.

If the current hypercube fits inside a ball of size d_{escape}, then the hypercube need not be subdivided. The hypercube may be discarded if we are interested only in the free space; in fact, we may add $\varepsilon/2$ to $e_{\text{escape}}(q)$, allowing many of the smallest hypercubes along the boundary be to discarded. Or, we may store these hypercubes if we are interested in problems that require information about topological properties of the obstacle space, such as proving non-existence of paths (McCarthy *et al.* [21]).

E. Larger cells, and cell merging

The computations of d_{safe} and of d_{escape} based on workspace information, as described above, are simple, fast, and too conservative. There are several ways in which the estimates may be improved, allowing larger cells.

First, $e(q)$ is the distance of the closest point on the robot to the obstacle, but the fastest point (with respect to the metric) may be some other point. So at some configuration, consider each point on the robot individually. For a given point p, compute the distance to the obstacles in workspace, $e(q,p)$. Now also compute the maximum speed for that point, $v_{\max}(p)$, over all configurations. For example, for

the 2R arm, a point at the end of the first link would have a maximum workspace speed of one, not three. Divide to compute $d_{\text{safe}}(p)$, and choose d_{safe} as the minimum of $d_{\text{safe}}(p)$ over all points p on the robot.

Since we expect the robot to contain a continuum of points, finding this minimum can be difficult. We may, however, divide the robot up into pieces (for example, links on the arm), and for each piece, compute a shortest distance and highest speed. In our implementation of the hypercube cover algorithm for the arm, we took this approach, dividing the arm into links, computing d_{safe} for each link, and taking the minimum.

Second, computing $v_{\max}$ over all configurations is too conservative, since we are only interested in whether configurations near the current sample, q, may collide. For example, if the arm is folded back on itself ($\theta_2 = \pi$), then the maximum speed of any point should be much less than three in the local region. It is not clear how to directly compute a d_{safe} value taking this into account, since the maximum speed over the d_{safe} interval depends on the change in the configuration, and the change in the configuration depends on the speed and the length of the interval.

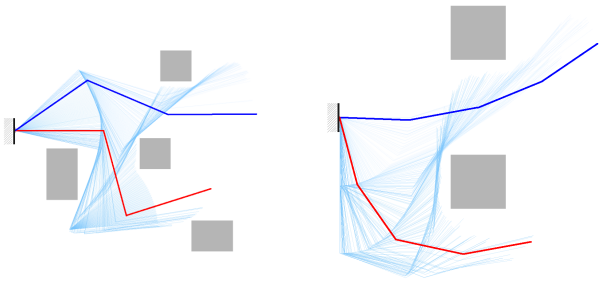
However, we can conduct an exponential search [22] for d_{safe}. First, assume that we're given some interval $\hat{d}$ and an initial configuration q. We can construct an oracle that will tell us whether $\hat{d}$ is either definitely safe, or possibly unsafe. For example, for the arm, we might choose motions that move the maximally extend the arm within the permitted interval, use the farthest point of extension to compute $v_{\max}$, and check if $\hat{d}v_{\max} > e(q)$. If so, the interval $\hat{d}$ is possibly unsafe; divide $\hat{d}$, and repeat until a satisfactory (but conservative) approximation of d_{safe} value has been found. We used such an exponential search in the example hypercube decompositions for the nR arms below.

Finally, it should be noted that adjacent free convex cells may be merged into a single cell, and we used this fact to merge some cells for planar arms. Interestingly, adjacent subconvex cells cannot necessarily be merged; we do not merge adjacent cells for the Reeds-Shepp car example, but would like to understand better how compression of groups of such adjacent cells might be done.

V. Existence of Optimal Paths in the Cover

Sometimes, optimal trajectories do not exist in the free configuration space; for example, Desaulniers showed [23] that for the Reeds-Shepp car among simple polygonal obstacles, there can exist pairs of configurations such that given any connecting trajectory with a finite number of control switches, there is a shorter connecting trajectory: chattering. However, optimal trajectories exist in finite sets of subconvex cells and chattering does not occur:

Theorem 3: Given a pair of points contained within a finite, connected set of cells subconvex to the free configuration space, with corresponding metric that is continuous along the boundaries of each cell, there exists an optimal trajectory that can be described by a non-repeating sequence of cells $c_1, \ldots, c_m$, together with one point $p_i \in c_i$ for each cell except the last, such that this optimal trajectory consists of optimal trajectories connecting the start to p_1, p_1 to p_2, etc, and finally from p_{m-1} to the goal.



(a) 3R planar arm (b) 4R planar arm

Fig. 6: Example trajectories of 3R and 4R planar arms.

Proof: Since the cells are connected, there is a set of curves that connect the given points; let these curves be parameterized by arc length under the given metric. Any such connecting curve will enter a cell for a first time, and exit each cell (except the last) for a last time, s_l. If we replace this section of the curve with an optimal trajectory within the cell, the new curve will certainly be no longer than the original.

Thus, to find an optimal curve, it is sufficient to consider the set of curves connecting the start and goal, with one locally optimal section per cell. Each such curve can be described by the discrete structure of the curve (the non-repeating sequence of cell indices, sorted by increasing order of s_l values), together with parameters describing the configurations at which the curve leaves each cell for the last time. Since there are finitely many structures (upper bounded by the cardinality of the power set over the set of cells), we need only show that for each structure, an optimal trajectory exists.

The problem of finding the minimum cost for a particular structure is a finite-dimensional optimization (there are finitely many parameters describing final exits from cells) over a compact set (the boundaries of the cells) of a continuous function (the finite sum of metric functions over the cells); an optimal solution therefore exists, as does a corresponding optimal trajectory. ■

We expect Theorem 3 to be useful for designing motion planning algorithms: no cell ever needs to be visited twice, although both the sequence of cells and the final exit points from each need to be determined.

VI. Case Study: Planar Arms

To find a trajectory connecting an initial configuration q_0 and a goal configuration q_g, we create samples along the boundary of all hypercubes at some resolution. We use the steering method to compute the distance between every pair of samples in the same hypercube. Then, we find the shortest path in the graph induced by these samples, using a straightforward A* search.

scenario	ε	number of cells	running time
Figure 5a	0.2	1350697	148
Figure 5b	0.3	4630896	332

TABLE I: Performance for robot arm system.

The heuristic function we used is a lower bound on the distance between the sample represented by the state to the goal, which can be easily computed based on the Euclidean distance in the workspace and the maximum velocity. For specific systems, we expect designing a more accurate heuristic function to be possible.

We take the 3R planar arm 4R planar arm as test cases for the planning algorithm; two example resulting trajectories are shown in Figure 5. We implement the algorithm in C++ and conducted tests on a modern desktop machine (iMac) with an Intel Core i5 2.7 GHz CPU and 16GB RAM. Table I shows the running times, memory, and number of cells generated while constructing representations of c-space for the figures shown. Most of time is spent in the collision detection, since we only use an elementary method to check collision.

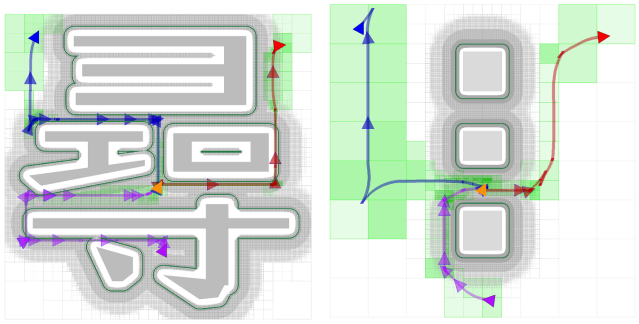
VII. Case study: Reeds-Shepp car

In this section, we show how to apply the decomposition algorithm to other systems, in which metric balls are hard to compute, including non-holonomic systems, using the Reeds-Shepp car as an example. Remember that the crucial part of the decomposition algorithm is to test a given hypercube Q is subconvex with respect to C_{f} without false positive. We gave a general approach based on Lipschitz continuity of the steering method in Sec. IV. However, the resulting hypercubes tend to be smaller than we would like.

Here, we give another (numerical) procedure for testing the subconvexity of a hypercube. Compute the swept volume in the workspace for all trajectories connecting all pairs of configurations in the hypercube. If the swept volume is collision free, then this hypercube is subconvex by definition.

Computing this swept volume analytically is difficult for most systems. Thus, instead of computing swept volume analytically, we approximated the swept volume numerically in the following way: first, densely sample configurations within the hypercube and use the steering method to compute trajectories between all pairs of configurations. Second, compute an approximate bounding volume for all trajectories in the workspace. If the bounding volume is collision free, then this hypercube can be considered subconvex. By using this numerical testing procedure, the size of the subconvex hypercube found by the decomposition algorithm can be greatly increased.

Optimal trajectories for the Reeds-Shepp car can be found analytically, and we use the optimal trajectory solver as the steering method [17]. The configuration space to be $[-\pi,\pi]^3$ and the car is represented as an isosceles triangle with base length 0.25 and height 0.25. The resulting trajectories are shown in Figure 6 and the performance of Algorithm 1 for 4R planar arm system is in Table II.



(a) (b)

Fig. 7: Several example trajectories in different environments of the Reeds-Shepp car. Orange triangles indicate common goal configurations. Shaded polygons are obstacles in workspace. Solid curves around obstacles are the boundary C_ε in work space. Subconvex hypercubes are projected into workspace as cubes. Dark green boundaries show the grown obstacles in the workspace.

scenario	ε	number of cells	running time in seconds
Figure 6a	0.2	24216563	312
Figure 6b	0.15	25218358	66

TABLE II: Performance for Reeds-Shepp car.

VIII. Conclusions and future work

We have presented a definition of convexity that we believe is useful for understanding the interplay between local, optimal steering methods and the global structure of the configuration space. This is our first work on this problem, and we have not yet conducted exhaustive experimental exploration of the properties of the cell decompositions described. Initial results are promising, however, in that for low-DOF systems, we are able to construct apparently very good trajectories from constructed cell-decompositions, and cells corresponding to large open spaces in the workspace are quite large.

It may not be too surprising that a very large number of cells is needed to represent the area of the configuration space near obstacles, or that we do not escape the curse of dimensionality. We intend to explore methods, perhaps exploiting properties such as visibility, that allow sparser representations near obstacles while still allowing some approximation guarantees about optimality to be maintained. We would also like to explore topological properties of configuration spaces using these cell decompositions, along the lines of recent work by, for example, Bhattacharya *et al.* [24].

References

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

[2] B. Donald and P. Xavier, "Provably good approximation algorithms for optimal kinodynamic planning: Robots with decoupled dynamics bounds," *Algorithmica*, vol. 14, no. 6, pp. 443–479, 1995.

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

[4] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forwards and backwards." *Pacific Journal of Mathematics*, vol. 145, no. 2, pp. 367–393, 1990.

[5] H. Yeh, S. L. Thomas, D. Eppstein, and N. M. Amato, "UOBPRM: A uniformly distributed obstacle-based PRM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 2655–2662. [Online]. Available: http://dx.doi.org/10.1109/IROS.2012.6385875

[6] J. Denny and N. M. Amato, "Toggle PRM: A coordinated mapping of c-free and c-obstacle in arbitrary dimension," in *Algorithmic Foundations of Robotics X - Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*, 2012, pp. 297–312. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36279-8_18

[7] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[8] J. Bialkowski, S. Karaman, M. W. Otte, and E. Frazzoli, "Efficient collision checking in sampling-based motion planning," in *Algorithmic Foundations of Robotics X - Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics, WAFR 2012, MIT, Cambridge, Massachusetts, USA, June 13-15 2012*, 2012, pp. 365–380.

[9] Y. Li, Z. Littlefield, and K. E. Bekris, "Sparse methods for efficient asymptotically optimal kinodynamic planning," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, Istanbul, Turkey, August 2014.

[10] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, Istanbul, Turkey, August 2014.

[11] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, July 1957.

[12] H. Sussmann and G. Tang, "Shortest paths for the Reeds-Shepp car: a worked out example of the use of geometric techniques in nonlinear optimal control," Department of Mathematics, Rutgers University, New Brunswick, NJ 08903, SYCON 91-10, 1991.

[13] D. J. Balkcom and M. T. Mason, "Time optimal trajectories for bounded velocity differential drive vehicles," *International Journal of Robotics Research*, vol. 21, no. 3, pp. 199–218, 2002.

[14] W. Wang and D. J. Balkcom, "Analytical time-optimal trajectories for an omni-directional vehicle," in *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, 2012, pp. 4519–4524.

[15] M. Vendittelli, J.-P. Laumond, and P. Soueres, "Shortest paths to obstacles for a polygonal car-like robot," in *Proceedings of the 38th IEEE Conference on Decision and Control*, vol. 1, Dec 1999, pp. 17–22.

[16] G. Lafferriere and H. J. Sussmann, "A differential geometric approach to motion planning," in *Nonholonomic Motion Planning*, ser. The Springer International Series in Engineering and Computer Science, Z. Li and J. Canny, Eds. Springer US, 1993, vol. 192, pp. 235–270.

[17] J. Laumond, *Robot motion planning and control*, ser. Lectures Notes in Control and Information Sciences 229. Springer, N.ISBN 3-540-76219-1, 1998.

[18] A. Bellaïche, "The tangent space in sub-Riemannian geometry," in *Sub-Riemannian Geometry*, ser. Progress in Mathematics, A. Bellaïche and J.-J. Risler, Eds. Birkhäuser Basel, 1996, vol. 144, pp. 1–78.

[19] J. H. C. Whitehead, "Convex regions in the geometry of paths," *The Quarterly Journal of Mathematics*, vol. os-3, no. 1, pp. 33–42, 1932.

[20] M. T. Mason, *Mechanics of Robotic Manipulation*. Cambridge, MA, USA: MIT Press, 2001.

[21] Z. McCarthy, T. Bretl, and S. Hutchinson, "Proving path non-existence using sampling and alpha shapes," in *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, 2012, pp. 2563–2569. [Online]. Available: http://dx.doi.org/10.1109/ICRA.2012.6225300

[22] J. L. Bentley and A. C.-C. Yao, "An almost optimal algorithm for unbounded searching," *Information Processing Letters*, vol. 5, no. 3, pp. 82 – 87, 1976.

[23] G. Desaulniers, "On shortest paths for a car-like robot maneuvering around obstacles," *Robotics and Autonomous Systems*, vol. 17, no. 3, pp. 139 – 148, 1996.

[24] S. Bhattacharya, D. Lipsky, R. Ghrist, and V. Kumar, "Invariants for homology classes with application to optimal search and planning problem in robotics," *Ann. Math. Artif. Intell.*, vol. 67, no. 3-4, pp. 251–281, 2013. [Online]. Available: http://dx.doi.org/10.1007/s10472-013-9357-7