ANNUAL REVIEWS

Annual Review of Control, Robotics, and Autonomous Systems

Sampling-Based Methods for Motion Planning with Constraints

Zachary Kingston, Mark Moll, and Lydia E. Kavraki

Department of Computer Science, Rice University, Houston, Texas 77005, USA;
email: zak@rice.edu, mmoll@rice.edu, kavraki@rice.edu

Annu. Rev. Control Robot. Auton. Syst. 2018. 1:3.1–3.27

The *Annual Review of Control, Robotics, and Autonomous Systems* is online at control.annualreviews.org

https://doi.org/10.1146/annurev-control-060117-105226

Keywords

robotics, robot motion planning, sampling-based planning, constraints, planning with constraints, planning for high-dimensional robotic systems

Abstract

Robots with many degrees of freedom (e.g., humanoid robots and mobile manipulators) have increasingly been employed to accomplish realistic tasks in domains such as disaster relief, spacecraft logistics, and home caretaking. Finding feasible motions for these robots autonomously is essential for their operation. Sampling-based motion planning algorithms are effective for these high-dimensional systems; however, incorporating task constraints (e.g., keeping a cup level or writing on a board) into the planning process introduces significant challenges. This survey describes the families of methods for sampling-based planning with constraints and places them on a spectrum delineated by their complexity. Constrained sampling-based methods are based on two core primitive operations: (*a*) sampling constraint-satisfying configurations and (*b*) generating constraint-satisfying continuous motion. Although this article presents the basics of sampling-based planning for contextual background, it focuses on the representation of constraints and sampling-based planners that incorporate constraints.

1. INTRODUCTION

Motion planning problem: the problem of finding a continuous path $\tau : [0, 1] \rightarrow Q_{\text{free}}$ that connects $q_{\text{start}} = \tau(0)$ and $\tau(1) \in Q_{\text{goal}}$

Consider a mobile manipulator tasked with carrying out various chores and maintenance tasks. Robonaut 2 (R2) (1), shown in **Figure 1**, is an example of such a high-dimensional robotic platform designed to work with humans in spacecraft. Future concept NASA missions involve spacecraft that will be uncrewed for long periods of time but still require maintenance. R2 and other robots could fill a unique role in such missions and are mechanically capable of executing the complex tasks necessary for spacecraft upkeep, such as turning valves (**Figure 1*a***), opening doors, and extracting cargo from a rack (**Figure 1*b***). However, for these robots to do so autonomously, a motion planning system that can generate feasible motion from high-level requirements is needed. Additionally, this motion planning system must respect the constraints on a task in order to achieve success, such as turning a valve only about its axis, rotating a door about its hinge, or extracting cargo linearly from its hold. The requirements produced by this scenario and in many other problem domains (e.g., household caretaking and disaster recovery) motivate the study of sampling-based motion planners with constraints.

Motion planning is an essential tool for autonomous robots, as it enables the description and execution of motion as high-level goals rather than lower-level primitives (e.g., manual specification of joint angles for a manipulator). However, motion planning is a PSPACE-hard problem (2), with complexity growing with the number of a robot's degrees of freedom. Although exact algorithms exist, they are difficult to implement and scale poorly to high-dimensional robots. Before going into detail on constrained sampling-based planning algorithms, we first provide a brief summary of historical approaches to the motion planning problem and point out which approaches have considered constraints.

Early on, potential field methods were proposed, which follow the gradient of a potential that guides a robot to its goal (3). It is difficult, though, to come up with a general mechanism to

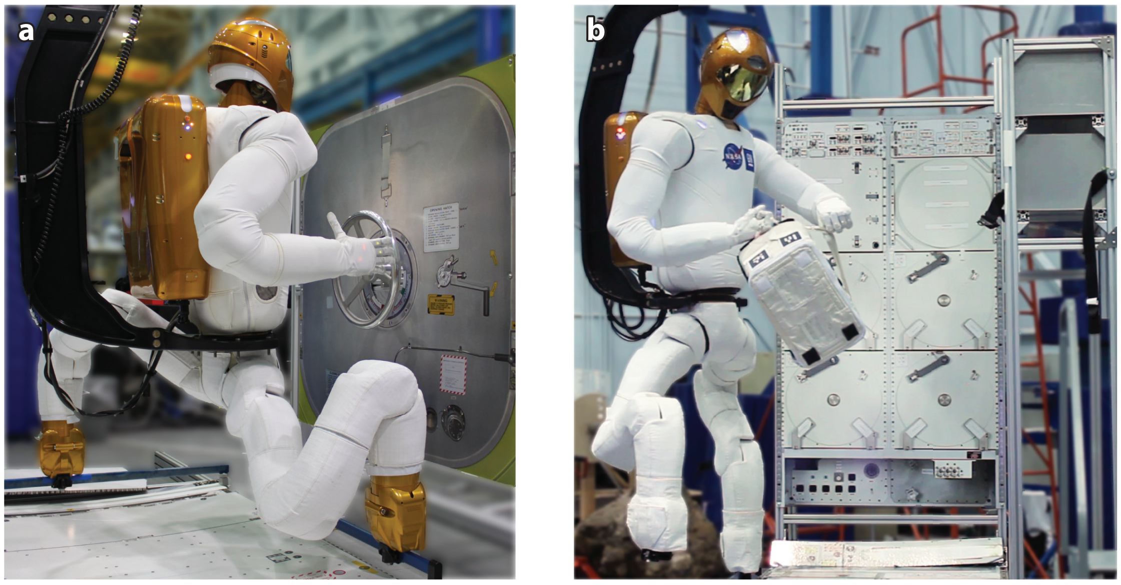

Figure 1

Examples of motion planning with constraints. (*a*) NASA's Robonaut 2, a highly dexterous humanoid robot, opens a valve, which constrains the end effector to follow a circular motion. Next, it slides open the door, which requires a whole-body sideways movement while maintaining its stance. (*b*) Robonaut 2 holds a bag with two hands and moves its entire body. Photos courtesy of NASA.

escape local minima of a potential function (4) or design a potential function that has only one minimum (5). Another family of planning algorithms is composed of heuristic search techniques [e.g., A* (6)] that operate over a discretization of possible robot configurations. These algorithms provide resolution completeness: A path will be found if the discretization is fine enough (7, 8). A careful choice of resolution and heuristics is critical for efficient heuristic search. In principle, the classes of motion planning algorithms described above could be adapted to incorporate constraints [e.g., discretized search with constraints (9)]. However, owing to the complications of scaling these methods to higher-dimensional systems such as R2, they are generally not applied to modern systems.

The techniques presented so far do not consider task constraints. In general, specialized methods are required to cope with motion constraints on robotic systems. The rich history of motion constraints began in industrial control, with Cartesian constraints on manipulator end effectors to describe assembly tasks (10–12). One of the most common constraints is Cartesian curve tracking, which requires a manipulator to follow a preplanned end-effector path—a constraint on the end effector's motion (e.g., welding and painting tasks in manufacturing). As robotic manipulators on the factory floor became more complex and gained degrees of freedom redundant to the task at hand, more advanced techniques for Cartesian curve tracking required resolution of the degrees of freedom of the robot (13) using inverse kinematic techniques (14, 15). However, using these techniques to generate paths is difficult, as it is hard to guarantee path continuity (16). The first applications of geometric constraints to planning techniques in low-dimensional spaces were reduced to problems of finding geodesics on polyhedral structures (17), similar to finding shortest paths of visibility graphs (18, 19). However, as motion planning was applied to more complex, higher-dimensional robotic systems, geometric constraints increased the difficulty of the motion planning problem and required additional consideration for effective planning.

In recent years, approaches that use penalty functions and optimize over full trajectories have been proposed (20–23). These approaches relax the hard constraints of the task (those that must be satisfied exactly, e.g., geometric task constraints and obstacle avoidance) into soft constraints (corresponding to some cost to optimize), combining the constraints into the formulation of the penalty function. Additionally, probabilistic completeness can still be preserved by combining optimization with random trajectory initialization in an appropriate functional space (20). However, tuning the penalty functions to avoid local minima and avoid paths that go through thin obstacles (as obstacle avoidance is relaxed) is challenging for robots with many degrees of freedom in complex scenes. This becomes even more challenging with multiple constraints and complex task goals.

Sampling-based algorithms take a very different approach. They randomly sample valid robot configurations and form a graph of valid motions (7, 8). Many algorithms provide probabilistic completeness: The probability of finding a solution goes to 1 with the run time of the algorithm, provided a solution exists (7, 8). Sampling-based motion planning algorithms are effective at solving motion planning problems in a broad range of settings with minimal changes, including very-high-dimensional systems. They have also been used in very different contexts, such as computer graphics and computational structural biology (e.g., 24, 25).

Among the classes of algorithms presented, sampling-based planning algorithms have emerged as the basis of several constrained planning approaches. This is in part because such algorithms are inherently modular and adaptive: Constraints can be easily incorporated into the core of a sampling-based algorithm without affecting its method for solution finding. With these methods, systems like R2 (26) can successfully accomplish complex, constrained tasks and find motions that respect constraints. However, there have been a variety of sampling-based methods proposed to handle constraints, each with a distinct methodology for handling and incorporating constraints into the planning process.

The rest of this review is organized as follows. First, we give a more formal description of the motion planning problem and the specification of constraints (Section 2). Next, we provide a brief overview of the main varieties of sampling-based planning algorithms and their core components (Section 3). After that, we describe in detail the main algorithms for sampling-based planning with constraints (Section 4). This includes a description of how the core components of sampling-based planning can be modified to handle constraints. Finally, we conclude with a discussion of the state of the art and possible avenues for future research (Section 5).

Configuration space (Q): the set of all robot configurations

Free space (Q_{free}): the set of configurations where the robot is not colliding with any obstacles or itself

Constraint function (F): $F : Q \to \mathbb{R}^k$ such that $F(q) = \mathbf{0}$ when q satisfies k given constraints

2. MOTION PLANNING AND CONSTRAINTS

This section develops the notation and mathematical objects necessary to understand sampling-based motion planning with constraints. Motion planning, particularly motion planning with constraints, draws from concepts in differential geometry to describe the various spaces utilized in the planning process. Spivak (27) provided a good reference for these topics.

This section is organized as follows. First, Section 2.1 introduces the mathematical notation. Next, Section 2.2 discusses how constraints have been expressed in the literature. Finally, Section 2.3 presents some methods for constraint composition.

2.1. Notation

The classical version of the motion planning problem can be defined as follows. One of the key ideas in motion planning is to lift the problem of planning for a dimensioned, articulated robotic system into planning for a single point that represents the robot in a higher-dimensional space. This space, called the configuration space, Q, includes all configurations for a given robot. The configuration space is a metric space (distance is defined between all points) and is usually a differentiable manifold. The dimensionality n of Q corresponds to the number of degrees of freedom of the robot. Let $Q_{\text{free}} \subseteq Q$ be the free space—the set of configurations where the robot is not colliding with any obstacles or itself. Given a start configuration $q_{\text{start}} \in Q_{\text{free}}$ and a set of goal configurations $Q_{\text{goal}} \subseteq Q_{\text{free}}$, the motion planning problem is then to find a continuous path $\tau : [0, 1] \to Q_{\text{free}}$ that connects $q_{\text{start}} = \tau(0)$ and $\tau(1) \in Q_{\text{goal}}$. The algebraic complexity of Q_{free} determines the PSPACE-hard complexity of this problem (2). A key idea of sampling-based planning is to avoid computing Q_{free} exactly (as described in Section 3).

The constraints we consider in this review are geometric and rely only on the configuration of the robot q, not on other properties of the motion, such as velocities or accelerations. Constraints reduce the effective number of degrees of freedom, denoted by m. Here, m is less than n, the number of degrees of freedom. It is usually not possible to reparameterize the system in terms of its effective degrees of freedom. Instead, the constraints are often written in terms of a constraint function $F : Q \to \mathbb{R}^k$ such that $F(q) = \mathbf{0}$ when q satisfies the constraints. Here, $k = n - m$ is the number of equality constraints imposed. Generally, F must be a continuous and differentiable function. For example, take a robot with its end effector constrained to remain at one position. That end-effector constraint could be encoded as

$$F(q) = \text{distance of end effector to position},$$

which evaluates to $\mathbf{0}$ when the constraint is satisfied. **Figure 2** shows a more complicated example with R2. Inequality constraints can be dealt with in much the same way as collision-avoidance constraints, as will become clear in the next section.

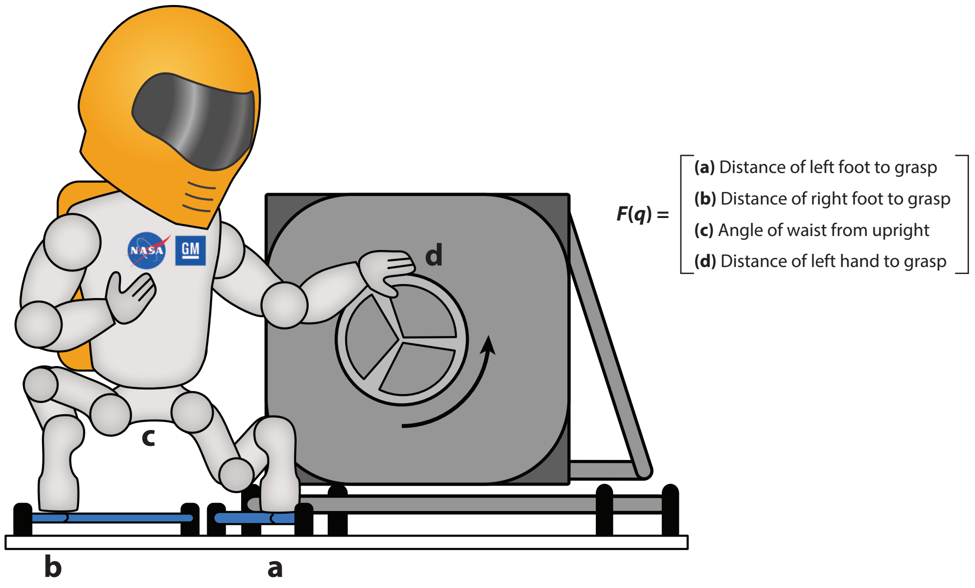
F(q) =
(a) Distance of left foot to grasp
(b) Distance of right foot to grasp
(c) Angle of waist from upright
(d) Distance of left hand to grasp
a
b
c
d

Figure 2

A breakdown of the constraints shown in **Figure 1*a***, encoded within a constraint function. For Robonaut 2 to make a full-body motion, all four constraints must be satisfied, i.e., $F(q) = \mathbf{0}$.

The constraint function defines an m-dimensional implicit constrained configuration space within the ambient configuration space (shown in **Figure 3**):

$$X = \{q \in Q \mid F(q) = \mathbf{0}\},$$

which consists of all configurations that satisfy the constraint. Although hard to visualize in higher dimensions, it is clear that there is a much smaller subset of allowable motion a robot can make when attempting to achieve a task (e.g., keeping a cup level or a welding tip on the surface of a piece of metal). The relative measure (volume) of X compared with Q is small and usually 0, which highlights the problem of using uninformed sampling to obtain valid configurations. If F is continuous and differentiable and Q is a differentiable manifold, then X is a differentiable manifold as well. The number of equality constraints, k, is also referred to as the codimension of

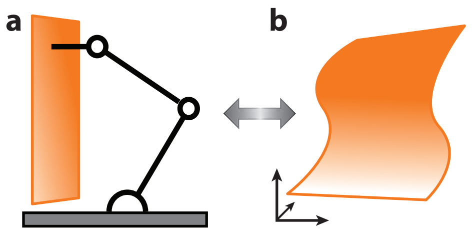

Figure 3

(*a*) An end-effector constraint (*orange plane*) imposed on a simple three-link manipulator. (*b*) The end-effector constraint corresponding to the lower-dimensional constraint manifold X (*orange sheet*) within the manipulator's configuration space $Q \subset \mathbb{R}^3$ (*black axes*).

the manifold. The problem of motion planning with constraints can now be defined as finding a continuous path in $X_{\text{free}} = X \cap Q_{\text{free}}$ that connects a given q_{start} and $q \in Q_{\text{goal}}$.

Constraint manifold (X): an implicit configuration space within the ambient configuration space Q: $X = \{q \in Q \mid F(q) = \mathbf{0}\}$

2.2. Constraint Expression

Specialized constraints most commonly take the form of end-effector constraints. These are constraints phrased in terms of the position and orientation of the robot's end effector. End effectors are generally the component of the robot that carries out the task, and as such, they are usually subject to the task constraints (e.g., maintaining contact or not rotating past a limit). End-effector constraints originate in industrial control with Cartesian constraints between interacting objects (10), which was later developed into general end-effector constraints (11, 12). There are many modern incarnations of end-effector constraints (28), such as the task-space-region formulation (29). Task-space regions are a general representation for many, but not all, Cartesian end-effector constraints (such as a screwing motion). End-effector constraints can also be extended to closed-chain systems by decomposing the loop into two manipulators that must maintain contact throughout the entire motion, closing the chain (30). Task-space regions have also been extended to task-space-region chains (29), which can model articulated kinematic structures such as doors and drawers in a scene as virtual kinematic chains. Task-space-region chains form a closed-chain system with the robot's manipulator.

More abstractly, many approaches have been taken to specify motion constraints in the form of a constraint function $F(q)$. The most general approach is to not assume any properties of the constraint function in relation to the kinematic structure of the robot and assume that the function encodes the distance to the surface of the constraint manifold (31). This general representation comes at the price of the ability to exploit features of the constraint that might be employed by a system utilizing end-effector constraints. Solver speed can be improved and satisfying configurations can be generated faster if constraints can be cast as functions that can be automatically differentiated or have analytic derivatives (32).

2.3. Constraint Composition

The representation of a constraint function and its derivative is critical to efficiently solving an instance of a constraint motion planning problem. Composing constraints that must all be simultaneously satisfied into one constraint function is nontrivial. For example, given a humanoid robot, what is the best way to combine and encode balance, the task objective, and a visibility region that must be maintained? This "and"-ing of constraints together into one function can be thought of as computing the intersection of sets of configurations that satisfy each constraint. If the constraint is phrased as a constraint function $F(q)$, multiple constraints are composed as additional equality constraints, increasing the dimension k (shown in **Figure 2**). However, the addition of new equality constraints can potentially introduce singularities in the constraint function. When the structure of the constraints is known and their importance and dependence can be deduced, it is possible to order the constraints and use null-space projection to attempt to solve for satisfying configurations hierarchically (33; e.g., see 26). Another approach is to use more advanced gradient descent techniques or cyclic projection, as discussed by Berenson (29). How multiple constraints are encoded has dramatic effects on the performance of a constrained motion planning method. Care must be taken when selecting an encoding; otherwise, planner completeness or efficiency is sacrificed.

What if the composition of tasks we wish to achieve has more than one modality? Intermittent contact, an essential component of manipulation and legged locomotion, requires the constant

a GraphPlanner(q_{start}, q_{goal})

```
    $\mathcal{G}$.init($q_{start}$, $q_{goal}$);
    while no path from $q_{start}$ to $q_{goal}$ do
        $q_{rand} \longleftarrow$ Sample();
        $\mathcal{Q} \longleftarrow$ SelectNghbrs($\mathcal{G}$, $q_{rand}$)
        for all $q_{near} \in \mathcal{Q}$ do
            if Connect($q_{near}$, $q_{rand}$) then
            $\mathcal{G}$.Add($q_{near}$, $q_{rand}$)
```

b TreePlanner(q_{start}, q_{goal})

```
    $\mathcal{T}$.init($q_{start}$);
    while no path from $q_{start}$ to $q_{goal}$ do
        $q_{rand} \longleftarrow$ Sample();
        $q_{near} \longleftarrow$ Select(T, $q_{rand}$)
        $q_{new} \longleftarrow$ Extend($q_{near}$, $q_{rand}$);
        if Connect($q_{new}$, $q_{near}$) then
        $\mathcal{T}$.Add($q_{near}$, $q_{new}$)
```

Figure 4

Prototypical examples of (*a*) graph-based and (*b*) tree-based sampling-based planners.

addition and removal of constraints (34, 35). Combining constraints where only a subset need to be satisfied at any given time is an "or"-ing of the constraints together—creating a union of constraint manifolds that potentially intersect and overlap. This creates singularity points, changes in dimension, and other problematic changes that most constraint methodologies cannot handle. One approach is to use a higher-level discrete representation of a graph to handle mode switching between different constraints, which might have different numbers of equality constraints imposed on the system (32). This problem can also be thought of as a special instance of hierarchical planning, with a discrete selection of constraint modality followed by geometric constrained planning. Footstep planning and other task and motion planning problems can all be thought of within this framework (36–40). Each of these planners employs domain-specific knowledge to solve the problem efficiently, but to the best of our knowledge, no general-purpose solutions have been proposed.

3. SAMPLING-BASED MOTION PLANNING

It is helpful to first describe the general structure of (unconstrained) sampling-based planning algorithms and the common primitives they rely on (for a more in-depth review of sampling-based planning, see 7, 8, 41). The general idea behind sampling-based planning is to avoid computing the free space exactly and instead sample free configurations and connect them to construct a tree or graph that approximates the connectivity of the underlying free space. Most sampling-based algorithms provide probabilistic completeness guarantees (42): If a solution exists, then the probability of finding a path goes to 1 with the number of samples generated by the algorithm. If no solution exists, then most sampling-based algorithms cannot recognize this [although it is possible in some cases (43)].

Figure 4 shows in pseudocode the two main varieties of sampling-based planners. **Figure 4*a*** shows a basic version of the first sampling-based planner, the probabilistic road-map method (PRM) (44). It incrementally constructs a road map embedded in Q_{free} by repeatedly sampling collision-free configurations via rejection sampling. For each sampled configuration, it computes nearby configurations sampled during previous iterations. If there exists a collision-free motion between the new sample and a neighbor, a new edge is added to the road map. This process continues until the start and goal configurations are in the same connected component of the graph, at which point the shortest path in the road map can be extracted via, e.g., A*. The PRM algorithm grows a road map that can be reused to solve many motion planning problems in the same environment.

In many cases, however, we are interested only in solving one particular problem (e.g., when the environment is changing, is very large, or has many different connected components). In such cases, a tree planner (**Figure 4*b***) might be more appropriate. The most well-known variant of this type of planner is the rapidly exploring random trees (RRT) algorithm (45, 46), but several other tree-based planners have been proposed, such as expansive-space trees (EST) (47) and others (48, 49). RRT grows a tree of configurations from the start to the goal. At each iteration, a random sample is generated (which may be in collision). The existing tree is extended toward the random sample from the existing configuration nearest to the sample. If the new tree branch can be connected to the goal, then the algorithm terminates. A popular variant of tree planners is to grow two trees simultaneously, one from the start and one from the goal (45). Connection between the two nearest states in the two trees is tested after every tree extension. The bidirectional tree search terminates once a connection between the two trees is found.

Despite their differences, many sampling-based planners have similar requirements from the robot's configuration space. The following are some of the components that are commonly used in sampling-based algorithms:

- *Samplers*: Typically, uniform sampling is used, but various heuristics have been proposed to sample (approximately) in lower-dimensional spaces to improve the odds of sampling in narrow passages, which is key to solving the motion planning problem. Sampling near the surface of configuration-space obstacles, a codimension 1 manifold, can be justified by the fact that configurations in narrow passages tend to be close to this surface. Although this surface is not computed analytically, various techniques have been proposed to sample near the surface (50, 51). Alternatively, one could sample near the medial axis, a one-dimensional structure formed by all configurations that have more than one closest point on the boundary of Q_{free}, i.e., exactly between two obstacles. Configurations on the medial axis tend to "see" more of Q_{free} than other configurations (52, 53). There has also been work on sampling entire lower-dimensional manifolds of the configuration space (54), which can better capture the connectivity of the free space in some problems. Finally, deterministic quasi-random samples improve the spread of samples (dispersion) compared with uniformly random sampling (55).
- *Metrics and nearest-neighbor data structures*: The choice of distance measure is often critical to the performance of sampling-based planners. Intuitively, a good distance measure reflects the difficulty of connecting configurations. If the measure is a proper metric, then various data structures can be used to efficiently find nearest neighbors. In many cases, approximate neighbors are sufficient, which can be computed much more efficiently in high-dimensional spaces than exact nearest neighbors (56).
- *Local planners*: A local planner is a fast but not necessarily complete method for finding paths between nearby configurations. In many cases, interpolation is used [or spherical linear interpolation (SLERP) for rotations (57)]. Tree-based planners can also be easily extended for kinodynamic motion planning, where the dynamics can often be written as $\dot{q} = f(q, u)$. During the extension, a steering function that drives the system toward a randomly sampled state is used as a local planner. Randomly sampling a control input u is generally sufficient for probabilistic completeness (46).
- *Coverage estimates*: Several sampling-based planners use the density of samples in a grid defined in a low-dimensional projection of the configuration space as a way to measure coverage and guide the exploration (49, 58). Although random projections often work well in practice (59), for constrained planning it may be difficult to define a projection over Q that approximates the density of sampling in the implicit configuration space X.

Note that the above is not a complete list of all sampling-based planner components but a list of important components for constrained sampling-based planning. The Open Motion Planning Library (60) provides various implementations of these core components as well as implementations of all the sampling-based planning algorithms mentioned in this section.

The paths produced by sampling-based planning are feasible but sometimes far from optimal. There are various techniques that postprocess paths to optimize them locally (61). This tends to work well in practice. However, with some small modifications, planners like PRM and RRT can be proven to be asymptotically optimal: The solution path will converge to the globally optimal solution over time (62). Subsequent work has improved the convergence rate (see, e.g., 63), but in practice, repeatedly running a nonoptimizing planner, smoothing the solution path, and keeping the best one seems to work surprisingly well in comparison (64, 65). Asymptotic optimality can be extended to kinodynamic planning (66–68). It is also possible to create sparse road maps or trees that guarantee asymptotic near optimality (68, 69)—that is, the solution paths converge to paths whose length is within a small constant factor approximation of the shortest path. There are complex trade-offs among the time to first feasible solution, convergence rate, and degree of optimality that are highly problem dependent. Systematic benchmarking is needed to determine a good algorithm for a given problem domain (70).

Finally, an idea that can be combined with many of the planning algorithms above is lazy evaluation of the validity of configurations and the motions that connect them (58, 71). Collision checking is the most expensive operation in sampling-based planning. By postponing this step until a candidate solution is found, collision checking can be avoided for all configurations and motions that are never considered to be part of a candidate solution.

4. SAMPLING-BASED MOTION PLANNING WITH CONSTRAINTS

4.1. Challenges

Constraints introduce another element of difficulty to the problem: the need to find configurations that satisfy the constraint function. The core concepts that enable a sampling-based planner to perform effectively require adaptation to appropriately handle the constraint function and generate a satisfying path. Let us reconsider each of the concepts introduced in the previous section in light of the need to satisfy the constraint function:

- *Samplers*: Sampling valid configurations is crucial to guiding the exploration of a planner through a robot's free space. The structure of the implicit region defined by a constraint function is not known a priori and is thus hard to sample from without careful consideration or preprocessing. A planner needs to be able to either sample configurations that satisfy the constraint function [solutions to $F(q) = \mathbf{0}$] or guide the search toward valid regions of the space.
- *Metrics and nearest-neighbor data structures*: Normally, the distance metric utilized by a sampling-based planner is defined by the configuration space, such as the Euclidean metric for $\mathbb{R}^n$. However, the constraint function defines a subset of the configuration space that can be curved and twisted relative to the ambient configuration space. In this case, the configuration-space distance metric bears very little resemblance to distance on the manifold, as is the case in the "Swiss roll" function (72). A more appropriate metric for this space would be something like the Riemannian metric, as the constraint function usually defines a Riemannian manifold (27). However, implicitly defined metrics such as the Riemannian metric are expensive to compute because they require computing the shortest geodesic on the manifold between two points. Computing the Riemannian metric is infeasible for any motion planning application that is concerned with speed of execution. However, if we already had a road map constructed on the constraint manifold, then the shortest path length within the road map could be used as an approximation of the Riemannian metric, as was done by Tenenbaum et al. (72). A sampling-based planner needs to consider its choice of

Tangent space: a real vector space that contains all possible tangent motions to a point on the constraint manifold

metric to effectively plan with constraints. Additionally, for nearest-neighbor computation, there are approximate methods that have been employed for curved data (73). Unfortunately, these methods require precomputation and are not suited for the rapidly updating structures employed in planning, and the adaptation of approximate methods is an open problem.

- *Local planners*: Normally, the local planner is a fast procedure to generate the intermediate configurations between two configurations. This, in the case of interpolation, corresponds to computing the geodesic between the configurations. However, within implicit spaces defined by constraint functions, interpolation becomes very difficult because the curvature and structure of the space is unknown a priori. On manifolds, there are many existing approaches to compute the minimum-length geodesic (74–77). How a planner employs a local planner that respects constraints is crucial to its success in the constrained planning problem.
- *Coverage estimates*: To the best of our knowledge, no constrained sampling-based planner has employed a planning methodology that utilizes a coverage estimate in its planning process. An interesting direction for future research would be to gather knowledge of the structure of the constraint manifold to direct the planning process.

For a sampling-based planner to plan with constraints, sampling and local planning must be augmented to satisfy constraints, as both of these elements directly affect whether the planning generates a valid path; as such, most methods focus on these two elements. In existing work, the metric from the ambient configuration space is typically used, which works well in practical applications. The ambient configuration space's metric defines a semimetric (78) for the constrained space, as the triangle inequality may not hold given sufficient curvature of the implicit space. However, semimetrics are good enough for most sampling-based planners because this discrepancy affects only the effectiveness of exploration. In this case, some theoretical guarantees may not hold.

4.2. Methodology Overview

The approaches to handling constraints within a sampling-based framework can be organized into a spectrum based on the complexity of the algorithmic machinery necessary to compute satisfying samples and connect them to the motion graph. The families of methods lie upon the spectrum as follows, from least to most complex:

- *Relaxation*: Because the critical limitation of sampling-based planners in solving constrained problems is their inability to find satisfying configurations (owing to the lower dimension of X), a simple idea is to relax the surface of the constraint manifold by increasing the allowed tolerance of the constraint function, changing $F(q) = \mathbf{0}$ to $\|F(q)\| < \varepsilon$. With this relaxation, sampling-based planners that have no additional machinery to handle the constraint can plan and generate a path.
- *Projection*: Finding satisfying configurations of the constraint function $F(q) = \mathbf{0}$ requires finding solutions to the constraint's system of equations. A projection operator takes a configuration and projects it onto the surface of the implicit manifold, iteratively retracting the point to a minimum of the constraint function and solving a linear system of equations at each iteration. The projection operator is a heavy hammer available to the planner to use for both sampling and local planning.
- *Tangent space*: From a known satisfying configuration, a tangent space of the constraint function can be generated. The tangent space is constructed by finding the basis for the null space of the derivative of the constraint function. Satisfying configurations and valid local motions can be generated within the tangent space.

- *Atlas*: Instead of recomputing tangent spaces at all points when an expansion is needed, the tangent spaces can be kept and composed together to create a piecewise linear approximation of the manifold, which can then be readily used to sample satisfying configurations or compute geodesics for local planning. This is called an atlas, in a slight abuse of terminology from differential geometry.
- *Reparameterization*: For certain constraints, it is possible to compute a new parameterization of the robot's configuration, allowing direct sampling of constraint-satisfying configurations. Using the reparameterized space, a new configuration space can be generated or a local motion computed and then mapped back into the robot's previous configuration space.

Atlas: a concept borrowed from differential geometry (27) in which a manifold is modeled as a collection of charts, each of which can be approximated by a tangent space

We discuss each methodology in detail in Section 4.3. Notably, most techniques for constrained sampling-based motion planning do not alter the core mechanics used by sampling-based planners. Generally, constrained sampling-based algorithms are adaptations of existing algorithms that incorporate a methodology for constraint satisfaction. RRT-based planners are often used as the basis for a constrained planner, perhaps in part owing to the straightforward steps in the algorithm. Note that RRT-based planners rely on uniform sampling, which is typically not possible with implicit manifolds. It is an open question whether other sampling-based planners that do not depend on uniform sampling (e.g., 47–49) might have an advantage, assuming they can be adapted to deal with constraints. Some recent work (79) described in Section 4.3.5 suggests that this is the case for some systems, but further study is needed.

4.3. Methodologies

Below, we describe in detail the different constraint methodologies. They differ primarily in the way that they perform sampling and local planning.

4.3.1. Relaxation. The primary challenge facing sampling-based planning approaches with constraints is generating configurations that satisfy the constraint equation $F(q) = \mathbf{0}$. Sampling-based planners generally sample within a configuration space in which the constraint function—a set of equality relations—defines a zero-volume subset. Hence, there is zero probability that an uninformed random sample will satisfy the constraint. To resolve this issue, relaxation-based approaches to solving constrained problems relax the constraint function, growing the subset of satisfying configurations by introducing an allowable tolerance to the constraint, $\|F(q)\| < \varepsilon$. The set of satisfying configurations has a nonzero probability of being sampled within the relaxed constraint, albeit with chances similar to narrow passages in the unconstrained instance of the motion planning problem. **Figure 5*a*** depicts sampling with relaxation-based approaches. Bonilla et al. (80, 81) applied this technique to bimanual manipulation problems. These works leverage execution-level controllers with compliant, closed-chain control to ensure successful execution despite using configurations that are not within the zero set of the constraint function. Because the subset of constraint-satisfying configurations defines a narrow band around the manifold, techniques well suited to motion planning problems in difficult domains can be adapted to improve performance (e.g., 82, 83). Local planning is also simple in a relaxation-based method (**Figure 5*b***). The local planner of the ambient configuration space is used. Since connections made to the planner's motion graph are generally local, the curvature of the manifold is respected because motions do not go far enough to invalidate themselves.

Relaxation-based approaches to constrained planning bridge unconstrained instances of the motion planning problem to the constrained incarnation. The sampling strategies employed by

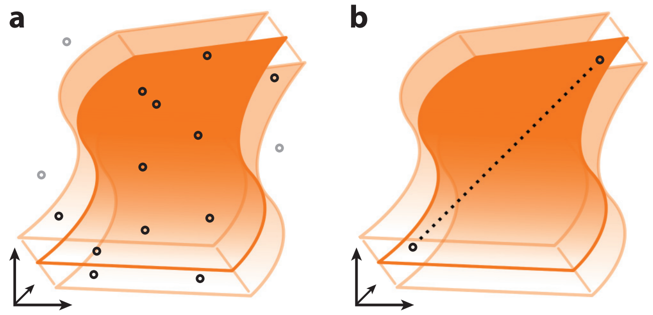

Figure 5

Sampling and local planning with relaxation-based constraint handling. The constraint manifold (*orange*) is given nonzero volume by relaxing the constraint according to some tolerance; boundaries are shown by faded extensions of the manifold. (*a*) Standard rejection sampling used to find close-to-satisfying configurations. Invalid samples are in gray; valid samples are in black. (*b*) Standard local planning (*dashed line*).

a relaxation-based planner can use algorithmic techniques meant to exploit lower-dimensional structures within planning, such as obstacle-based sampling and medial axis sampling. Because relaxation-based approaches do not fundamentally change the planning problem from the perspective of the motion planner (as they encode the constraint as a narrow passage), the constraint methodology is already decoupled from the planning approach. As such, very little adaptation of any sampling-based planner is needed to handle the inflated constraint. Sampling-based planning methodologies that better suit the planning problem could be used to solve relaxed constrained planning problems. Sampling-based planners also retain their probabilistic completeness when using the relaxed constraint. However, execution success is no longer a given, as it is now determined by the controller's ability to handle plans that deviate from the geometrically defined constraint. Much of the complexity of handling the constraint is pushed onto the controller rather than the planner. Despite these bonuses, this approach is not usually taken because it is inefficient when given complex constraints. Sampling narrow passages is still inefficient compared with other approaches, such as projection- or approximation-based methods. Additionally, these methods rely on properties of the robot and its controller, and the compliance of the mechanism must be sufficient to retain the constraint in the face of geometric inaccuracy.

Note that, in general, each of the constraint-solving techniques has a tolerance on constraint satisfaction, but this number is usually tuned to achievable numerical precision in order to obtain accurate results. In this technique, the constraint is purposefully relaxed far more than it is in other techniques.

4.3.2. Projection.

In a constrained motion planning problem, a satisfying path contains only configurations that satisfy the constraint function, $F(q) = \mathbf{0}$. One method to find satisfying configurations is with a projection operator. Projection takes a configuration and projects it into the set of satisfying configurations, retracting the point to a minimum of the constraint function. Formally, a projection operator is an idempotent mapping $P(q) : Q \to X$, where if $q \in X$, $P(q) = q$. Projection is typically an iterative optimization-based procedure that finds solutions to the constraint equation, $F(q) \approx \mathbf{0}$. A common implementation of projection is a Newton procedure with Jacobian (pseudo)inverse gradient descent, using the Jacobian of the constraint function $J(q)$ (15, 84). This is shown in Algorithm 1.

Algorithm 1 (projection through iteration). This algorithm is an iterative procedure that uses the Jacobian pseudoinverse $[J(q)^{+}]$ of the constraint function $F(q)$:

```
procedure Projection($q$)
    $x \leftarrow F(q)$
    while $\|x\| > \epsilon$ do
        $\Delta q \leftarrow J(q)^{+}x$
        $q \leftarrow q - \Delta q$
        $x \leftarrow F(q)$
    return $q$
```

Note that the Jacobian does not need a full inversion if solutions Δq to $J(q)\Delta q = F(q)$ can be found. For certain problems, QR factorization (85) and other matrix decompositions might be more efficient.

The constraint function must encode the distance of the configuration from the solution of the equation so that the gradient adequately represents progress toward the manifold. As such, it is actually not strictly necessary that the underlying subspace defined by the constraint be a manifold, as long as this distance is properly encoded. Projection requires only piecewise differentiability of the constraint function so that gradient descent can converge successfully.

Projection-based approaches utilize the projection operator heavily within the sampling and local planning components of the planner. **Figure 6*a*** shows sampling with a projection operator. Samples are drawn from the ambient configuration space and are projected to solve the constraint function. As time trends to infinity, the constraint function's satisfying subset of configurations will be fully covered by projection sampling, as proved by Berenson (29). This property of projection sampling preserves the probabilistic completeness of RRT-like projection-based algorithms (29). **Figure 6*b*** shows an example of local planning using a projection operator. In this method, the curvature of the constraint function is captured by small incremental steps interleaved with projection. From an initial satisfying configuration to a goal configuration, a small interpolation is done within the ambient configuration space. The interpolated point is projected to generate a satisfying configuration. The process is repeated from each successive point until the goal is reached.

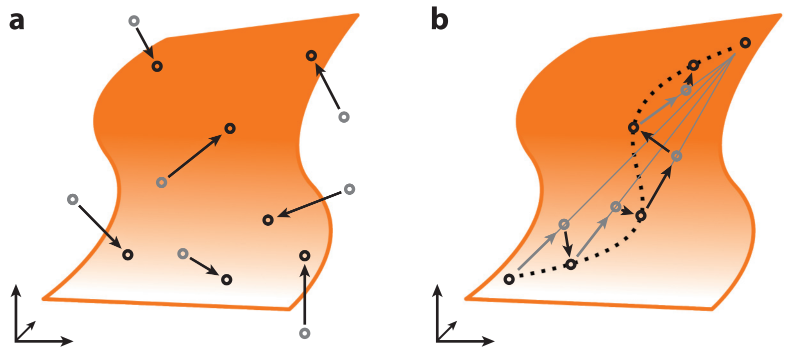

Figure 6

Sampling and local planning with projection-based constraint handling. The samples are projected (*black arrows*) onto the constraint manifold (*orange*) using a projection operator. (*a*) Sampling. After a sample is drawn from the configuration space (*gray circles*), it is retracted to the surface of the constraint manifold (*black circles*) using the projection operator. (*b*) Local planning. From a starting configuration (*black circle at bottom left*), a new unsatisfying configuration (*first gray circle*) closer to the goal is generated by interpolation (*first gray arrow*). That configuration is then projected onto the manifold (*first black arrow*), and the process repeats until the goal is reached or another termination condition is met.

This is the core of the mechanism introduced by Yakey et al. (30), who considered constrained motion planning for closed-chain planar chains. Many modern methods (e.g., 29, 32) adopted this approach for local planning. Recently, work has gone into investigating continuous local planning using projection (77, 86), avoiding issues of discontinuity with naive discretization of the geodesic.

Self-motion manifold: the set of configurations that all result in the same end-effector pose for a redundant manipulator; its tangent space is the null space of the manipulator Jacobian

Historically, projection-based methods saw early adoption in solving loop-closure problems for parallel manipulators, an intrinsic constraint. Planning with loop closures was (and continues to be) very relevant in structural biology for protein analysis (87), and complex loop-closure problems in robotics were solved with PRM variants using active/passive-chain methods (30, 88). In active/passive-chain methods, the projection operator uses inverse kinematic techniques to join the passive chain to the active chain, closing the loop and creating a satisfying configuration. Projection operators were also used to solve curve-tracking problems in early industrial applications (89).

The idea of using projection to satisfy constraints was applied to general end-effector constraints by Yao & Gupta (90). Task-constrained RRT (28) further generalized the idea of constraints and utilized Jacobian gradient descent (15) for projection. Recently, constrained bidirectional RRT (CBiRRT2) (29), the motion planner implemented for the Humanoid Path Planner System (32), and other planners [such as one for the HRP-2 humanoid (91)] have utilized projection with general constraints.

A special application of projection-based approaches to sampling-based motion planning with constraints is the domain of regrasping problems. In these problems, a manipulated object must be released by a manipulator (because of some obstacle within the current homotopy group of the path) and regrasped to continue progress. These problems generally define a constraint manifold that can be described as a foliated manifold, where the constraint manifold is divided into slices for each pose the end effector of the robot can take (34). In a foliation, a manifold X is decomposed into a disjoint union of connected submanifolds called leaves. Each leaf corresponds to the self-motion manifold of the robot at a particular end-effector pose, of which there are infinitely many. The self-motion manifold is the set of all configurations where the end effector remains in the same pose. Its tangent space is the null space of the manipulator Jacobian. This property has been exploited for manipulation planning (34) and by a few constrained planners (92, 93) to achieve manipulation tasks with regrasping. Note that regrasping problems are not the exclusive domain of projection-based approaches but have not been attempted by other types of sampling-based planners with constraints.

Projection-based planners have several notable advantages that contribute to their success as one of the most widely implemented methodologies. Primarily, the projection operator is easy to implement and captures the structure of the constraint function within the planning process. Historically, this has been implemented with randomized gradient descent (30), but modern solvers typically implement a form of Jacobian gradient descent (28). More advanced solvers can be used, such as hierarchical inverse kinematic solvers or cyclical projection for systems under multiple constraints (26, 29). Particularly important is the implementation of the gradient descent routine, as this requires solving a potentially complex system of equations described by the constraint at each step of the iteration. Matrix decompositions can be expensive, and the constraint Jacobian is typically not guaranteed to be invertible.

4.3.3. Tangent space.

If constraint functions define a manifold or if the Jacobian of the constraint function is of full rank, it is possible to locally approximate the manifold using a tangent space of a satisfying configuration. The tangent space defines a locally linear approximation of the constraint manifold to a Euclidean space, which extends until the curvature of the manifold bends sufficiently away. The tangent space is constructed by finding the basis for the null space of $J(q)$, which can be computed through a matrix decomposition. The tangent space T_q is an $(n-k)$-dimensional

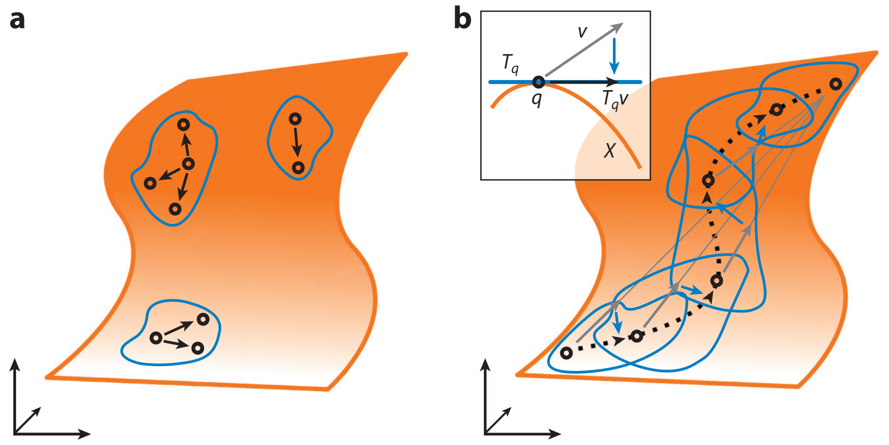

Figure 7

Tangent-space-based constraint handling. Given an initial satisfying configuration, a tangent space to that point can be computed that is the null space of the constraint function's Jacobian (*blue blobs*), and new satisfying configurations can be generated by local perturbations (*black arrows*). (*a*) Sampling. Samples are generated by creating a tangent space at a known configuration and projecting random vectors. (*b*) Local planning. From a starting configuration (*black circle at bottom left*), a tangent space is computed, and the vector v from the current configuration to the goal (*gray arrow*) is projected (*first blue arrow*) into the tangent space (*first black arrowhead*). The projected vector is added to the current configuration to generate a novel configuration close to satisfying the constraint.

space with its origin at a configuration $q \in X$, with an $n \times (n - k)$ orthonormal basis Φ_q. A vector v can also be projected through the tangent space to remove the components orthogonal to the Jacobian, leaving only components that are within the null space of the Jacobian, $T_q v$. This capability is used primarily by techniques based on tangent spaces to generate new configurations. Given a satisfying configuration, the tangent space is calculated, and some random vector within the tangent space is generated (a tangent vector). The tangent vector is added to the configuration from which the tangent space was created to generate a new, local configuration that is close to the manifold. **Figure 7*a*** depicts this process. As the codimension of the constraint manifold approximation increases, sampling in the tangent space becomes more accurate, at the price of increased computational cost per sample. **Figure 7*b*** depicts local planning utilizing tangent spaces. From an initial configuration, a vector to the goal configuration is computed. This vector is projected into the tangent space and decomposed into its components tangent to the manifold. The tangent vector is then added to the initial configuration to generate the next configuration in the local plan, similar to how sampling is done above. From the new configuration, the process is repeated until the goal is reached.

Yakey et al. (30) utilized projection from tangent spaces to generate nearby samples, which are then fixed up with projection. Tangent spaces have been used for manipulators under general end-effector constraints (28, 94). The technique has also seen many applications in curve-tracking constraints for redundant manipulators (95–97) and in structural biology to generate valid motions of proteins with loop closures (98–100).

In tangent-space-based techniques, new satisfying configurations are created by perturbing known satisfying configurations with vectors tangent to the constraint. The small perturbations create configurations that are close to satisfying the constraint and typically can be projected into the satisfying set with few iterations. This works well for heavily constrained systems where the set of valid motions is limited. With tuning of tolerances, it is also possible to not even require reprojection of the constraint onto the manifold. Tangent-space-based methods work particularly well for constraints that are more linear than curved and are well approximated by Euclidean spaces. End-effector constraints in particular have been the target of tangent-space-based methods for exploration (28, 94).

However, tangent-space-based methods are not without their drawbacks. Computing the kernel of the constraint Jacobian is expensive and requires multiple matrix decompositions to solve numerically. The method also breaks down near singularity points because the Jacobian loses rank and no longer maintains a surjective mapping to the ambient configuration space. Additionally, as stated above, tangent-space-based methods break down when the manifold becomes highly curved because the tangent movement rapidly drifts away from the surface of the manifold.

4.3.4. Atlas. Furthering the idea of utilizing tangent spaces to approximate the constraint locally is the idea of building an atlas of the manifold, a concept borrowed from the definition of differentiable manifolds (27). Such methods also require that the constraint function defines a manifold. Unlike the methods described in Section 4.3.3, atlas-based methods store generated tangent spaces to avoid recomputation. The tangent spaces are organized within a data structure called an atlas. The atlas is defined as a piecewise linear approximation of the constraint manifold using tangent spaces, which fully cover and approximate the manifold (101). The key difference between the method described by Henderson (101) and atlas-based planners is the incremental construction of the atlas interleaved with space exploration. These tangent spaces are generated and utilized exactly as described in Section 4.3.3 and allow sampling and mapping to and from the manifold and the tangent space. Atlas-based approaches utilize the tangent spaces to project configurations onto the manifold and lift configurations into the basis defined by the tangent space. A point $t \in T_q$ can be mapped into $q_t \in Q$ by $q_t = q + \Phi_q t$. To map the configuration q_t onto the manifold (an exponential map ψ_q), an orthonormal projection can be computed by solving the system of equations

$$F(q) = \mathbf{0} \qquad \text{and} \qquad \Phi_q^T(q - q_t) = \mathbf{0}.$$

The opposite mapping from the manifold to T_q is much simpler: $\psi_q^{-1}(q_t) = t = \Phi_q^T(q - q_t)$. These procedures were described in detail by Rheinboldt (102), along with other operations on implicit manifolds.

Sampling from an atlas is done as follows. A tangent space is chosen at random from the atlas, and a sample is drawn and projected from the tangent space as described in Section 4.3.3 (**Figure 8*a***). Planners that implement variants of the atlas-based methodology are derived from the procedure described by Henderson (101). AtlasRRT (31) implements the methodology faithfully, computing the tangent spaces and the hyperplanes to separate them (creating tangent polytopes) to guarantee uniform coverage of the part of the manifold covered by the tangent spaces. When traversing the manifold to compute connecting geodesics, AtlasRRT fully evaluates each point, projecting onto the manifold orthogonally and checking feasibility. Interpolation is done within the tangent spaces of the atlas, projecting the interpolated tangent-space configuration at each step to validate the motion (**Figure 8*b***).

Tangent bundle RRT (TB-RRT) (103) is another method that utilizes an atlas-based methodology. In contrast to AtlasRRT, TB-RRT performs a lazy evaluation and does not compute the separating half spaces, simply collecting a set of tangent spaces that cover the manifold. TB-RRT projects onto the manifold only when it needs to switch between tangent spaces, interpolating exclusively within the tangent space. Together, these features make TB-RRT more

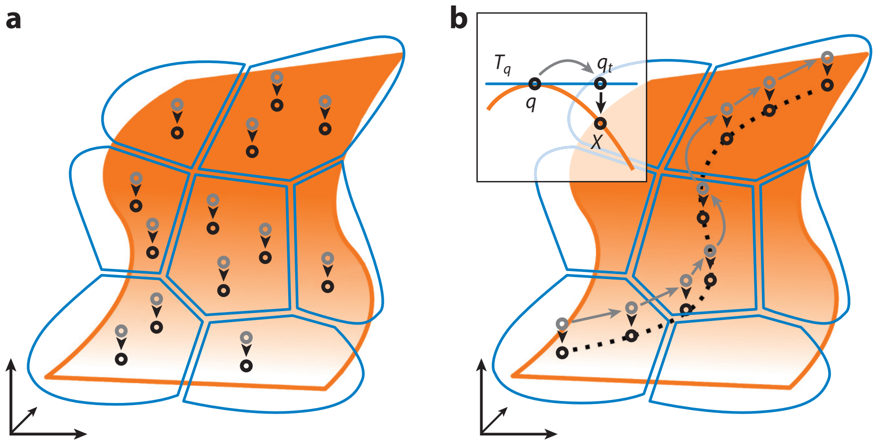

Figure 8

Atlas-based sampling and local planning akin to AtlasRRT. The atlas is a set of tangent polytopes that cover the constraint manifold (*blue blobs*). Here, the atlas has already been computed and covers the space. During planning, the atlas is constructed in tandem with sampling and local planning. (*a*) Sampling. Sampling of the manifold is done by drawing samples from the tangent polytopes by randomly sampling within them (*gray circles*). These points are then orthogonally projected from polytopes onto the surface of the manifold (*black arrowheads pointing to black circles*), ψ_q. (*b*) Local planning. Roughly, interpolation is done with the tangent space (*gray arrows*) from a configuration q to another q_t, and new configurations are projected onto the manifold for validation (*black arrows*). This continues until the goal is reached. For details, see Reference 31.

computationally efficient than AtlasRRT at exploring the constrained space. TB-RRT comes at the cost of overlapping tangent spaces, which leads to less uniform sampling. Furthermore, TB-RRT's lazy interpolation causes potential problems with invalid points, such as failing to check collisions with narrow configuration-space obstacles.

AtlasRRT has been the focus of multiple extensions, improving its capability and augmenting its guarantees. As mentioned above, one of the critical problems with generating a tangent space around a point is handling singularities, as the Jacobian of the constraint function loses rank and a tangent space can no longer be computed. Bohigas et al. (104) introduced a method on top of the AtlasRRT planner to plan for singularity-free paths. AtlasRRT* (105) extends AtlasRRT to be asymptotically optimal, in the vein of RRT* (106), providing the same theoretical guarantees of asymptotic optimality while also respecting the geometric constraints imposed on the system. There has also been an extension to kinodynamic planning, or planning with nonholonomic constraints, utilizing the basic framework of AtlasRRT (107). Additionally, the methodology behind building an atlas incrementally has been extended to general sampling-based motion planning (108), which enables any sampling-based planner to build an atlas approximation while planning with constraints. The extensions made to AtlasRRT emphasize the importance of using sampling-based methods for planning with constraints, as the methods are modular, easily adapted to new problem instances, and extended with features such as asymptotic optimality.

Atlas-based approaches make a trade-off between representational complexity and computational efficiency that pays off in many problem instances. For constraint manifolds that have a complex structure and high curvature, maintaining the atlas approximation enables efficient planning regardless of the relative structure of the manifold and configuration space. For example, a constraint manifold that has a toroidal topology with a narrow inner ring would be hard for a projection-based approach to sample and explore owing to the relatively small volume of configuration space that will end up projecting onto that portion of the manifold. Atlas-based approaches would not even notice the difficulty, as they work off the constructed approximation, which is invariant (given appropriate parameters) to the constraint and configuration space. Atlas-based approaches are also probabilistically complete (31).

The primary downside of atlas-based approaches is the difficulty of implementation, as the atlas data structure needs to be efficient and correct. Beyond this, there are also issues of diminishing returns with respect to the codimension of the constraint manifold relative to the ambient configuration space (108). Maintaining an approximation of the manifold is computationally inefficient for constraints that have only a few equality constraints relative to the configuration space. The tangent space does not buy much over doing projection sampling in this case, as there is little difference between the constraint manifold and the ambient space.

4.3.5. Implicit space representation. The techniques discussed above cover part of the spectrum of methods to compute satisfying configurations for constrained motion planning. Each methodology necessarily makes trade-offs between space and time efficiency, performing well in some environments but potentially failing in others. None of the constrained sampling-based planners make dramatic changes to the structure of the underlying augmented motion planning algorithm. Instead, these methods augment primitive operations (e.g., samplers and local planners, as discussed) to generate feasible motion. Additionally, just as trade-offs are made in constrained planners with constraint methodologies, there are many unconstrained sampling-based planners, each with its own heuristics or exploration strategies to perform well in certain environments. Developing a constrained sampling-based planner with a choice of constraint methodology well suited to the constraint and an exploration strategy well suited to the planning problem requires designing a bespoke planner that integrates the two. A recent work has proposed a general framework for constrained sampling-based planning that approaches augmenting primitive operations not from within the planner but from within the representation of the configuration space, leveraging the modularity of sampling-based planners (79). This has the benefit of enabling the composition of any emulated constraint methodology with a broad class of sampling-based planners, allowing choice of the combination best suited to a problem at the cost of losing potential benefits that coupled implementation can bring (e.g., speed and leveraging of planner properties). The benefits of this unified approach can be considerable in some cases. **Figure 9** shows a system

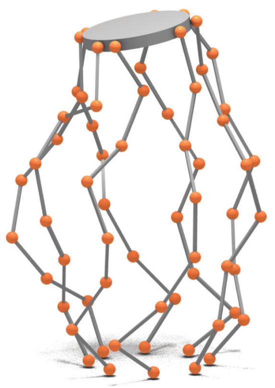

Figure 9

A parallel manipulator with 168 degrees of freedom. The unified approach allows the use of almost any sampling-based planner for this high-dimensional system (79).

described in more detail by Kingston et al. (79). Here, the combination of the KPIECE planner (49) and the projection-based constraint methodology was shown to be orders of magnitude faster than using RRT-Connect (45), which was the planner modified in prior works to accommodate constraints (29, 31, 103).

4.3.6. Reparameterization. In some cases, the constraint function and manipulator topology lend themselves to reparameterization, where another configuration space is generated for the robot and constraint. Configurations within this newly generated space fully describe the robot's state and satisfy the constraint. This allows for any traditional planner to be utilized on top of a new configuration space, enabling the machinery of the planner to function unaffected while satisfying constraints. Tangent-space- and atlas-based approaches can be thought of as reparameterization-based approaches, but the organization presented here distinguishes between precomputation and online exploration and constructions. The reparameterization-based approaches presented in this section are usually computed before planning, while tangent-space-based approaches are generally generated online for computational efficiency. Additionally, reparameterization is distinct from the tangent-space- and atlas-based approaches. Reparameterization creates a global, nonlinearly related space, while tangent-space- and atlas-based methods create local, linear approximations.

Reparameterization-based methods utilize the constraint and properties of the manipulator to generate a new, reparameterized configuration space, which is then mapped back into the original configuration space. Examples of this are the deformation space for planar closed-chain systems (109) and reachable volume space (110) for general kinematic chains. Generally, these methods have their own ways of sampling within their reparameterized space (shown in **Figure 10*a***) and their own ways of stepping within the reparameterized space (shown in **Figure 10*b***). The deformation space reparameterizes closed-chain planar systems by encoding the deformation of the closed chain within the reparameterized space. The deformation space encodes the configuration as a decomposition of triangles that form the polygon formed by the manipulator. The reachable volume space exploits properties of the joints within a kinematic chain (prismatic, revolute, and spherical) to generate reachable volumes of each frame of the

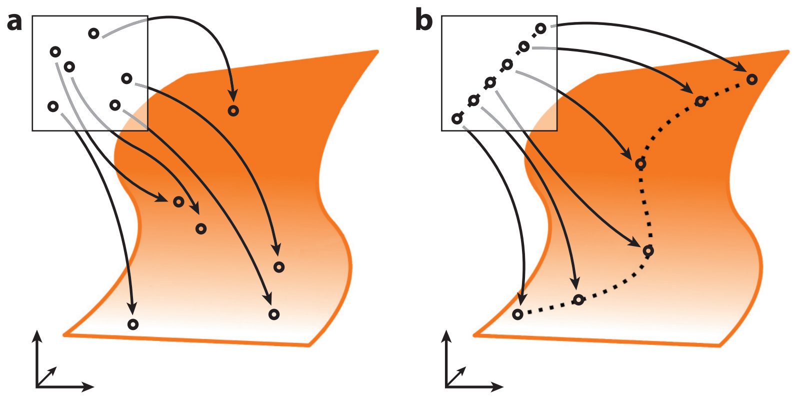

Figure 10

Reparameterization-based constrained handling. A new configuration space is computed from the manipulator and constraint, reparameterizing the space (*white square*). (*a*) Sampling. Samples can be drawn from the reparameterized space and mapped back into the original configuration space (*black arrows*). (*b*) Local planning. Interpolation is done within the reparameterized space and mapped back into the original configuration space.

manipulator. These are akin to Minkowski sums (7) but describe the subset of the workspace a manipulator can reach. The planner requires computing the volumes before planning but is efficient and can scale to very large problem instances (around 70 degrees of freedom) (110).

Reparameterization-based approaches are appealing from a sampling-based planning perspective. If it were possible to simply plan within a space that contained only satisfying configurations, then the constrained planning problem could be reduced to the unconstrained instance. For the unconstrained problem, this would be akin to planning only within the free configuration space. However, each of the reparameterization-based approaches requires a phase of precomputation to generate the reparameterized space. Reparameterization-based approaches rely heavily on geometrical properties of the manipulator and use knowledge of the constraint and manipulator's shape to efficiently encode the problem. They are also generally limited to a specific problem domain (e.g., planar chains and closed-loop systems) and are generally complex to implement, which prevents widespread applicability to many different robotic systems.

4.3.7. Offline sampling. Offline sampling to solve constrained planning problems introduces a methodology orthogonal to those mentioned above. In offline sampling methods, the underlying constraint manifold is sampled before planning takes place, generating a precomputed database of constraint-satisfying samples. The way these samples are generated is generally unimportant to the remainder of the planning approach, and one can utilize any of the methodologies described above. Normally, projection-based approaches are used because of their ease of implementation and guarantee to cover the manifold within the limit of sampling (29). First, samples are drawn to cover the area of interest in the satisfying subset defined by the constraint and placed within a database. Planning then takes place using standard sampling-based planning techniques, but with the planner taking its samples from the precomputed set of configurations. This approach of precomputing a set of constraint-satisfying configurations was used to satisfy balancing constraints on a humanoid robot (111, 112). Additionally, more structure can be imbued to the set of samples to generate a road map of valid motions on the surface of the manifold (113, 114), akin to experience-based planners for the unconstrained instance of the planning problem (115). The self-motion manifold of a robot's end effector can also be precomputed by utilizing road-map-based methods, describing a database of inverse kinematic solutions (116).

Offline sampling-based methods have the benefit of leveraging existing techniques within the sampling-based planning literature, as they generally require minimal adaptation of a planning algorithm after the precomputed set of samples is generated. Planning is also decoupled from database generation, so the constraint-sampling methodology best suited to the particular constrained planning problem can be used. These techniques come with the obvious drawbacks of the need for precomputation and the inflexibility that comes with generating a database offline. However, for intrinsic constraints of the robot, such as dynamic stability for humanoids or satisfying configurations of closed-chain systems, precomputation might be the correct answer to avoid repeating computation online. Additionally, precomputation-based approaches that apply to changing environments require an element of online planning to handle changing obstacle configurations and potentially invalidated edges in an offline-computed road map.

5. DISCUSSION

This review has covered the wide variety of methods that have been proposed to allow sampling-based planning algorithms to incorporate geometric motion constraints. These methods are effective in many real-world scenarios. However, there is no consensus about which approach is best suited for which types of constraints. This likely depends on several factors: the dimensionality of the configuration space, the (co)dimension of the constraint manifold, the degree of clutter in the environment, and so on. One factor that has not been considered in previous work is whether new exploration or exploitation strategies for planning on an implicit constraint manifold are needed. As mentioned above, uniform sampling on implicit spaces and measuring distances along manifolds is either impossible or computationally very expensive. This raises the question of whether a planner that depends less on uniform sampling and distance could be designed for planning with constraints. Additionally, future works should further investigate primitive operations that better represent the underlying constraint manifold, such as using distance metrics that capture the curvature of the manifold, local planners that generate continuous paths, and samplers that can approach uniform sampling of the manifold.

Another avenue for future work is addressing forces while planning with constraints. Intrinsic to many constraints is the application of force in a specific way. For example, writing on a whiteboard is a planar geometric constraint but also requires steady application of force to the board. The direction of force applied is orthogonal to the constraint. For the approaches presented in this work, the constraint is translated into a geometric constraint, because in general, kinodynamic planning (i.e., with forces and dynamics) is much more complicated than planning quasi-statically. A constrained sampling-based approach that leverages information from its constraint methodology could be potentially helpful and make force computations feasible. Finally, there is interesting future research into the development of a general approach to manipulation and locomotion planning that automatically identifies transitions from one set of constraints to another without requiring a hierarchical decomposition. This would require new techniques to simultaneously explore different constraint manifolds as well as ways to transition between them.

SUMMARY POINTS

1. Several methods have been proposed to extend sampling-based algorithms to incorporate geometric constraints on robot motion.
2. These methods are focused primarily on sampling and interpolation on constraint manifolds.
3. There are five categories of constraint methodologies: relaxation, projection, tangent-space sampling, incremental atlas construction, and reparameterization.
4. Unification of the projection, tangent-space sampling, and incremental atlas construction methodologies is possible by using a representation of the implicit space.

FUTURE ISSUES

1. Further study is needed to evaluate the relative merits of each of the constraint methodologies for sampling-based motion planning algorithms.
2. A general method is needed to handle multimodal constrained planning (such as locomotion and manipulation planning), where a planning algorithm needs to explore several different constraint manifolds and possible transitions between them.
3. Application of force is intrinsic to constraints; future methods should leverage information about constraints to consider the forces that will be applied by a robot while planning.

4. More work needs to be done to create primitive operations that better represent constraint spaces (e.g., local planners that are continuous, metrics for implicit spaces, and manifold sampling).

DISCLOSURE STATEMENT

The authors are not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

Work on this review has been supported in part by the National Science Foundation (IIS 1317849 and IIS 1718478) and Rice University. Z.K. is also supported by a NASA Space Technology Research Fellowship (80NSSC17K0163).

LITERATURE CITED

1. Diftler MA, Mehling JS, Abdallah ME, Radford NA, Bridgwater LB, et al. 2011. Robonaut 2 – the first humanoid robot in space. In *2011 IEEE International Conference on Robotics and Automation*, pp. 2178–83. New York: IEEE
2. Canny JF. 1988. *The Complexity of Robot Motion Planning*. Cambridge, MA: MIT Press
3. Khatib O. 1986. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.* 5:90–98
4. Barraquand J, Latombe JC. 1991. Robot motion planning: a distributed representation approach. *Int. J. Robot. Res.* 10:628–49
5. Rimon E, Koditschek DE. 1992. Exact robot navigation using artificial potential functions. *IEEE Trans. Robot. Autom.* 8:501–18
6. Aine S, Swaminathan S, Narayanan V, Hwang V, Likhachev M. 2015. Multi-heuristic A*. *Int. J. Robot. Res.* 35:224–43
7. Choset H, Lynch KM, Hutchinson S, Kantor G, Burgard W, et al. 2005. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press
8. LaValle SM. 2006. *Planning Algorithms*. Cambridge, UK: Cambridge Univ. Press
9. Phillips M, Hwang V, Chitta S, Likhachev M. 2016. Learning to plan for constrained manipulation from demonstrations. *Auton. Robots* 40:109–24
10. Ambler AP, Popplestone RJ. 1975. Inferring the positions of bodies from specified spatial relationships. *Artif. Intell.* 6:157–74
11. Mason MT. 1981. Compliance and force control for computer controlled manipulators. *IEEE Trans. Syst. Man Cybern.* 11:418–32
12. Khatib O. 1987. A unified approach for motion and force control of robot manipulators: the operational space formulation. *IEEE J. Robot. Autom.* 3:43–53
13. Seereeram S, Wen JT. 1995. A global approach to path planning for redundant manipulators. *IEEE Trans. Robot. Autom.* 11:152–60
14. Whitney DE. 1969. Resolved motion rate control of manipulators and human prostheses. *IEEE Trans. Man Mach. Syst.* 10:47–53
15. Buss SR, Kim JS. 2005. Selectively damped least squares for inverse kinematics. *J. Graph. Tools* 10:37–49
16. Beeson P, Hart S, Gee S. 2016. *Cartesian motion planning & task programming with CRAFTSMAN*. Poster presented at Robot. Sci. Syst. Workshop Task Motion Plan., Cambridge, MA, July 15
17. Mitchell JSB, Mount DM, Papdimitrious CH. 1987. The discrete geodesic problem. *SIAM J. Comput.* 16:647–68

18. Asano T, Asano T, Guibas L, Hershberger J, Imai H. 1985. Visibility-polygon search and Euclidean shortest paths. In *26th Annual Symposium on Foundations of Computer Science*, pp. 155–164. New York: IEEE
19. Alexopoulos C, Griffin PM. 1992. Path planning for a mobile robot. *IEEE Trans. Syst. Man Cybern.* 22:318–22
20. Zucker M, Ratliff N, Dragan A, Pivtoraiko P, Klingensmith M, et al. 2013. CHOMP: covariant Hamiltonian optimization for motion planning. *Int. J. Robot. Res.* 32:1164–93
21. Kalakrishnan M, Chitta S, Theodorou E, Pastor P, Schaal S. 2011. STOMP: stochastic trajectory optimization for motion planning. In *2011 IEEE International Conference on Robotics and Automation*, pp. 4569–74. New York: IEEE
22. Schulman J, Duan Y, Ho J, Lee A, Awwal I, et al. 2014. Motion planning with sequential convex optimization and convex collision checking. *Int. J. Robot. Res.* 33:1251–70
23. Dong J, Mukadam M, Dellaert F, Boots B. 2016. Motion planning as probabilistic inference using Gaussian processes and factor graphs. In *Robotics: Science and Systems XII*, ed. D Hsu, N Amato, S Berman, S Jacobs, chap. 1. N.p.: Robot. Sci. Syst. Found.
24. Latombe JC. 1999. Motion planning: a journey of robots, molecules, digital actors, and other artifacts. *Int. J. Robot. Res.* 18:1119–28
25. Gipson B, Hsu D, Kavraki LE, Latombe JC. 2012. Computational models of protein kinematics and dynamics: beyond simulation. *Annu. Rev. Anal. Chem.* 5:273–91
26. Baker W, Kingston Z, Moll M, Badger J, Kavraki LE. 2017. Robonaut 2 and you: specifying and executing complex operations. In *2017 IEEE Workshop on Advanced Robotics and Its Social Impacts*. New York: IEEE. **https://doi.org/10.1109/ARSO.2017.8025204**
27. Spivak M. 1999. *A Comprehensive Introduction to Differential Geometry*. Houston, TX: Publ. Perish
28. Stilman M. 2010. Global manipulation planning in robot joint space with task constraints. *IEEE Trans. Robot.* 26:576–84
29. Berenson D. 2011. *Constrained manipulation planning*. PhD Thesis, Carnegie Mellon Univ., Pittsburgh, PA
30. Yakey JH, LaValle SM, Kavraki LE. 2001. Randomized path planning for linkages with closed kinematic chains. *IEEE Trans. Robot. Autom.* 17:951–58
31. Jaillet L, Porta JM. 2013. Path planning under kinematic constraints by rapidly exploring manifolds. *IEEE Trans. Robot.* 29:105–17
32. Mirabel J, Tonneau S, Fernbach P, Seppälä A-K, Campana M, et al. 2016. HPP: a new software for constrained motion planning. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 383–89. New York: IEEE
33. Sentis L, Khatib O. 2005. Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. *Int. J. Humanoid Robot.* 2:505–18
34. Siméon T, Laumond JC, Cortés J, Sahbani A. 2004. Manipulation planning with probabilistic roadmaps. *Int. J. Robot. Res.* 32:729–46
35. Hauser K, Bretl T, Latombe JC, Harada K, Wilcox B. 2008. Motion planning for legged robots on varied terrain. *Int. J. Robot. Res.* 27:1325–49
36. Perrin N, Stasse O, Lamiraux F, Kim YJ, Manocha D. 2012. Real-time footstep planning for humanoid robots among 3D obstacles using a hybrid bounding box. In *2012 IEEE International Conference on Robotics and Automation*, pp. 977–82. New York: IEEE
37. Reid W, Fitch R, Göktogan AH, Sukkarieh S. 2016. *Motion planning for reconfigurable mobile robots using hierarchical fast marching trees*. Presented at Workshop Algorithmic Found. Robot., 12th, San Francisco, Dec. 18–20. Paper available at **http://wafr2016.berkeley.edu/program.html**
38. Dantam NT, Kingston ZK, Chaudhuri S, Kavraki LE. 2016. Incremental task and motion planning: a constraint-based approach. In *Robotics: Science and Systems XII*, ed. D Hsu, N Amato, S Berman, S Jacobs, chap. 2. N.p.: Robot. Sci. Syst. Found.
39. Lozano-Pérez T, Kaelbling LP. 2014. A constraint-based method for solving sequential manipulation planning problems. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3684–91. New York: IEEE

40. Kaelbling L, Lozano-Perez T. 2011. Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation*, pp. 1470–77. New York: IEEE
41. Elbanhawi M, Simic M. 2014. Sampling-based robot motion planning: a review. *IEEE Access* 2:56–77
42. Ladd AM, Kavraki LE. 2004. Measure theoretic analysis of probabilistic path planning. *IEEE Trans. Robot. Autom.* 20:229–42
43. McCarthy Z, Bretl T, Hutchinson S. 2012. Proving path non-existence using sampling and alpha shapes. In *2012 IEEE International Conference on Robotics and Automation*, pp. 2563–69. New York: IEEE
44. Kavraki LE, Švestka P, Latombe JC, Overmars M. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Autom.* 12:566–80
45. Kuffner J, LaValle SM. 2000. RRT-Connect: an efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, pp. 995–1001. New York: IEEE
46. LaValle SM, Kuffner JJ. 2001. Randomized kinodynamic planning. *Int. J. Robot. Res.* 20:378–400
47. Hsu D, Latombe JC, Motwani R. 1999. Path planning in expansive configuration spaces. *Int. J. Comput. Geom. Appl.* 9:495–512
48. Ladd AM, Kavraki LE. 2005. Motion planning in the presence of drift, underactuation and discrete system changes. In *Robotics: Science and Systems I*, ed. S Thrun, GS Sukhatme, S Schaal, pp. 233–41. Cambridge, MA: MIT Press
49. Şucan IA, Kavraki LE. 2012. A sampling-based tree planner for systems with complex dynamics. *IEEE Trans. Robot.* 28:116–31
50. Amato N, Bayazit O, Dale L, Jones C, Vallejo D. 1999. OBPRM: an obstacle-based PRM for 3D workspaces. In *Robotics: The Algorithmic Perspective*, ed. PK Agarwal, LE Kavraki, MT Mason, pp. 155–68. Wellesley, MA: A.K. Peters
51. Boor V, Overmars MH, van der Stappen AF. 1999. The Gaussian sampling strategy for probabilistic roadmap planners. In *1999 IEEE International Conference on Robotics and Automation*, pp. 1018–23. New York: IEEE
52. Wilmarth SA, Amato N, Stiller PF. 1999. MAPRM: a probabilistic roadmap planner with sampling on the medial axis of the free space. In *1999 IEEE International Conference on Robotics and Automation*, pp. 1024–31. New York: IEEE
53. Hsu D, Jiang T, Reif J, Sun Z. 2003. The bridge test for sampling narrow passages with probabilistic roadmap planners. *2003 IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 4420–26. New York: IEEE
54. Salzman O, Hemmer M, Halperin D. 2015. On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages. *IEEE Trans. Autom. Sci. Eng.* 12:529–38
55. LaValle SM, Branicky MS, Lindemann SR. 2004. On the relationship between classical grid search and probabilistic roadmaps. *Int. J. Robot. Res.* 23:673–92
56. Andoni A, Indyk P. 2017. Nearest neighbors in high-dimensional spaces. In *Handbook of Discrete and Computational Geometry*, ed. JE Goodman, J O'Rourke, CD Tóth, pp. 1135–55. Boca Raton, FL: CRC. 3rd ed.
57. Shoemake K. 1985. Animating rotation with quaternion curves. *ACM SIGGRAPH Comput. Graph.* 19:245–54
58. Sánchez G, Latombe JC. 2001. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In *Robotics Research: The Tenth International Symposium*, ed. RA Jarvis, Z Zelinsky, pp. 403–17. Berlin: Springer
59. Şucan IA, Kavraki LE. 2009. On the performance of random linear projections for sampling-based motion planning. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2434–39. New York: IEEE
60. Şucan IA, Moll M, Kavraki LE. 2012. The Open Motion Planning Library. *IEEE Robot. Autom. Mag.* 19:72–82
61. Geraerts R, Overmars M. 2007. Creating high-quality paths for motion planning. *Int. J. Robot. Res.* 26:845–63
62. Karaman S, Frazzoli E. 2011. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* 30:846–94

63. Gammell JD, Srinivasa SS, Barfoot TD. 2015. Batch informed trees (BIT*): sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In *2015 IEEE International Conference on Robotics and Automation*, pp. 3067–74. New York: IEEE
64. Luna R, Şucan IA, Moll M, Kavraki LE. 2013. Anytime solution optimization for sampling-based motion planning. In *2013 IEEE International Conference on Robotics and Automation*, pp. 5053–59. New York: IEEE
65. Luo J, Hauser K. 2014. An empirical study of optimal motion planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1761–68. New York: IEEE
66. Webb DJ, van den Berg J. 2013. Kinodynamic RRT*: asymptotically optimal motion planning for robots with linear dynamics. In *2013 IEEE International Conference on Robotics and Automation*, pp. 5054–61. New York: IEEE
67. Hauser K, Zhou Y. 2016. Asymptotically optimal planning by feasible kinodynamic planning in a state–cost space. *IEEE Trans. Robot.* 32:1431–43
68. Li Y, Littlefield Z, Bekris KE. 2016. Asymptotically optimal sampling-based kinodynamic planning. *Int. J. Robot. Res.* 35:528–64
69. Dobson A, Bekris KE. 2014. Sparse roadmap spanners for asymptotically near-optimal motion planning. *Int. J. Robot. Res.* 33:18–47
70. Moll M, Şucan IA, Kavraki LE. 2015. Benchmarking motion planning algorithms: an extensible infrastructure for analysis and visualization. *IEEE Robot. Autom. Mag.* 22:96–102
71. Bohlin R, Kavraki LE. 2000. Path planning using lazy PRM. In *2000 IEEE International Conference on Robotics and Automation*, pp. 521–28. New York: IEEE
72. Tenenbaum JB, de Silva V, Langford JC. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290:2319–23
73. Chaudhry R, Ivanov Y. 2010. Fast approximate nearest neighbor methods for non-Euclidean manifolds with applications to human activity analysis in videos. In *Computer Vision – ECCV 2010*, ed. K Daniilidis, P Maragos, N Paragios, pp. 735–48. Berlin: Springer
74. Kimmel R, Sethian JA. 1998. Computing geodesic paths on manifolds. *PNAS* 95:8431–35
75. Ying L, Candes EJ. 2006. Fast geodesic computation with the phase flow method. *J. Comput. Phys.* 220:6–18
76. Crane K, Weischedel C, Wardetzky M. 2013. Geodesics in heat: a new approach to computing distance based on heat flow. *ACM Trans. Graph.* 32:152
77. Hauser K. 2014. Fast interpolation and time-optimization with contact. *Int. J. Robot. Res.* 33:1231–50
78. Deza MM, Deza E. 2016. *Encyclopedia of Distances*. Berlin: Springer
79. Kingston Z, Moll M, Kavraki LE. 2018. Decoupling constraints from sampling-based planners. In *Robotics Research: The 18th International Symposium (ISRR)*. Forthcoming
80. Bonilla M, Farnioli E, Pallottino L, Bicchi A. 2015. Sample-based motion planning for soft robot manipulators under task constraints. In *2015 IEEE International Conference on Robotics and Automation*, pp. 2522–27. New York: IEEE
81. Bonilla M, Pallottino L, Bicchi A. 2017. Noninteracting constrained motion planning and control for robot manipulators. In *2017 IEEE International Conference on Robotics and Automation*, pp. 4038–43. New York: IEEE
82. Rodriguez S, Thomas S, Pearce R, Amato NM. 2008. RESAMPL: a region-sensitive adaptive motion planner. In *Algorithmic Foundations of Robotics VII*, ed. S Akella, NM Amato, WH Huang, B Mishra, pp. 285–300. Berlin: Springer
83. Bialkowski J, Otte M, Frazzoli E. 2013. Free-configuration biased sampling for motion planning. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1272–79. New York: IEEE
84. Dennis J, Schnabel R. 1983. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Englewood Cliffs, NJ: Prentice Hall
85. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. 1992. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge, UK: Cambridge Univ. Press. 2nd ed.
86. Mirabel J, Lamiraux F. 2017. *Manipulation planning: building paths on constrained manifolds*. Tech. Rep., LAAS-GEPETTO – Équipe Mouvement des Systèmes Anthropomorphes, Toulouse, Fr.
87. Wedemeyer WJ, Scheraga HA. 1999. Exact analytical loop closure in proteins using polynomial equations. *J. Comput. Chem.* 20:819–44

88. Cortés J, Siméon T, Laumond JP. 2002. A random loop generator for planning the motions of closed kinematic chains using PRM methods. In *2002 IEEE International Conference on Robotics and Automation*, pp. 2141–46. New York: IEEE
89. Oriolo G, Ottavi M, Vendittelli M. 2002. Probabilistic motion planning for redundant robots along given end-effector paths. In *2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 2, pp. 1657–62. New York: IEEE
90. Yao Z, Gupta K. 2005. Path planning with general end-effector constraints: using task space to guide configuration space search. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1875–80. New York: IEEE
91. Kanehiro F, Yoshida E, Yokoi K. 2012. Efficient reaching motion planning and execution for exploration by humanoid robots. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1911–16. New York: IEEE
92. Kim J, Ko I, Park FC. 2016. Randomized path planning for tasks requiring the release and regrasp of objects. *Adv. Robot.* 30:270–83
93. Xian Z, Lertkultanon P, Pham QC. 2017. Closed-chain manipulation of large objects by multi-arm robotic systems. *IEEE Robot. Autom. Lett.* 2:1832–39
94. Weghe MV, Ferguson D, Srinivasa SS. 2007. Randomized path planning for redundant manipulators without inverse kinematics. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pp. 477–82. New York: IEEE
95. Oriolo G, Vendittelli M. 2009. A control-based approach to task-constrained motion planning. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 297–302. New York: IEEE
96. Vendittelli M, Oriolo G. 2009. Task-constrained motion planning for underactuated robots. In *2015 IEEE International Conference on Robotics and Automation*, pp. 2965–70. New York: IEEE
97. Cefalo M, Oriolo G, Vendittelli M. 2013. Task-constrained motion planning with moving obstacles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5758–63. New York: IEEE
98. Zhang Y, Hauser K. 2013. Unbiased, scalable sampling of protein loop conformations from probabilistic priors. *BMC Struct. Biol.* 13(Suppl. 1):S9
99. Pachov DV, van den Bedem H. 2015. Nullspace sampling with holonomic constraints reveals molecular mechanisms of protein Gαs. *PLOS Comput. Biol.* 11:e1004361
100. Fonseca R, Budday D, van dem Bedem H. 2016. Collision-free Poisson motion planning in ultra high-dimensional molecular conformation spaces. arXiv:1607.07483
101. Henderson ME. 2002. Multiple parameter continuation: computing implicitly defined *k*-manifolds. *Int. J. Bifurc. Chaos* 12:451–76
102. Rheinboldt WC. 1996. MANPAK: a set of algorithms for computations on implicitly defined manifolds. *Comput. Math. Appl.* 32:15–28
103. Kim B, Um TT, Suh C, Park FC. 2016. Tangent bundle RRT: a randomized algorithm for constrained motion planning. *Robotica* 34:202–25
104. Bohigas O, Henderson ME, Ros L, Manubens M, Porta JM. 2013. Planning singularity-free paths on closed-chain manipulators. *IEEE Trans. Robot.* 29:888–98
105. Jaillet L, Porta JM. 2013. Asymptotically-optimal path planning on manifolds. In *Robotics: Science and Systems VIII*, ed. N Roy, P Newman, S Srinivasa, pp. 145–52. Cambridge, MA: MIT Press
106. Karaman S, Frazzoli E. 2011. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* 30:846–94
107. Bordalba R, Ros L, Porta JM. 2017. Kinodynamic planning on constraint manifolds. arXiv:1705.07637
108. Voss C, Moll M, Kavraki LE. 2017. *Atlas + X: sampling-based planners on constraint manifolds*. Tech. Rep. 17-02, Dep. Comput. Sci., Rice Univ., Houston, TX
109. Han L, Rudolph L, Blumenthal J, Valodzin I. 2008. Convexly stratified deformation spaces and efficient path planning for planar closed chains with revolute joints. *Int. J. Robot. Res.* 27:1189–212
110. McMahon T. 2016. *Sampling based motion planning with reachable volumes*. PhD Thesis, Texas A&M Univ., College Station
111. Kuffner JJ, Kagami S, Nishiwaki K, Inaba M, Inoue H. 2002. Dynamically-stable motion planning for humanoid robots. *Auton. Robots* 12:105–18

112. Burget F, Hornung A, Bennewitz M. 2013. Whole-body motion planning for manipulation of articulated objects. In *2013 IEEE International Conference on Robotics and Automation*, pp. 1656–62. New York: IEEE
113. Igarashi T, Stilman M. 2010. Homotopic path planning on manifolds for cabled mobile robots. In *Algorithmic Foundations of Robotics IX*, ed. D Hsu, V Isler, JC Latombe, MC Lin, pp. 1–18. Berlin: Springer
114. Şucan IA, Chitta S. 2012. Motion planning with constraints using configuration space approximations. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1904–10. New York: IEEE
115. Coleman D, Şucan IA, Moll M, Okada K, Correll N. 2015. Experience-based planning with sparse roadmap spanners. In *2015 IEEE International Conference on Robotics and Automation*, pp. 900–5. New York: IEEE
116. Hauser K. 2016. *Continuous pseudoinversion of a multivariate function: application to global redundancy resolution*. Presented at Workshop Algorithmic Found. Robot., 12th, San Francisco, Dec. 18–20. Paper available at **http://wafr2016.berkeley.edu/program.html**