Atlas + X: Sampling-based Planners on Constraint Manifolds

Caleb Voss, Mark Moll, and Lydia E. Kavraki

***Abstract*—Sampling-based planners struggle when the valid configurations are constrained to an implicit manifold. Special planners have been proposed for this problem recently. Our new framework is decoupled from any particular planner and augments existing algorithms not explicitly designed for constraint planning. We demonstrate the advantages of our generalized approach.**

***Index Terms*—Motion planning; constraint manifolds; sampling-based planners; atlas.**

I. Introduction

A robotic system whose motion is subjected to a set of hard kinematic constraints poses a challenge to traditional sampling-based planning techniques because the set of feasible configurations forms a low-dimensional manifold embedded in a higher dimensional space. Examples where such manifolds arise include problems in which the robot carries an object in a fixed orientation, in which the robot maintains contact with the environment, or in which the robot performs bimanual manipulation. In all these cases the robot cannot exercise its full range of motion due to the constraints. The resultant configuration space manifold of lower dimension poses several challenges for many sampling-based motion planners, which explore the space by interpolating toward sampled configurations [1]. The first difficulty is the sampling of valid configurations. The manifold of constrained configurations has zero measure with respect to the ambient space metric; therefore the space is almost everywhere invalid. Direct sampling of a valid configuration is often impossible because the manifold is defined implicitly by the constraints and lacks an analytic formula. The second difficulty is interpolation through the manifold space. Ambient-space interpolation departs the manifold, and projecting a sequence of such points back onto the manifold may not give a continuous path. Because of these difficulties, a standard sampling-based planner cannot solve such a problem unless it is augmented with a specialized sampler and interpolator.

One approach is to sample a target uniformly from ambient space and then repeatedly step towards it and project to the manifold. This is the approach of the planner CBiRRT2 [2], which is explicitly designed to operate on constraint manifolds. However, there is no guarantee that the manifold is covered uniformly by this process. For example, when samples are projected onto a torus with a small hole, the area around the hole will be sampled much more sparsely than other parts of the surface since a smaller fraction of the bounding box projects there. A solution is to use a representation of the manifold called an *atlas*. Formally, an atlas is a collection of charts that cover a manifold, where each chart is a diffeomorphism between a subset of the manifold and an open subset of Euclidean space. In this paper, we approximate an atlas (with bounded error) using a piecewise linear approximation [3]. In a slight abuse of terminology, we will refer to this approximation as the atlas throughout the paper. Projecting samples from an atlas onto the manifold gives much more uniform coverage of the manifold, to the benefit of sampling-based planners. Moreover, the process of constructing the atlas itself can be guided by a sampling-based planner, giving rise to the AtlasRRT algorithm [4], which incorporates a bidirectional variant of RRT [5]. AtlasRRT is successful and has been demonstrated to outperform CBiRRT2, which does not use an atlas, on several kinematically constrained problems.

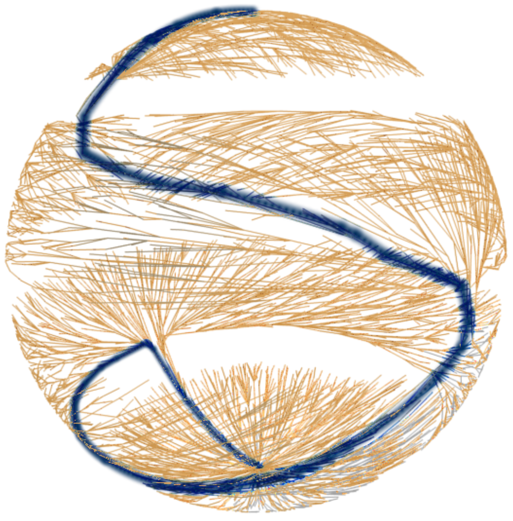
Fig. 1: Planner graph (orange) and best path (blue) generated by Atlas + RRT* in 0.5 seconds. RRT* minimizes path length during the allotted time. The manifold is the surface of a sphere. Three narrow passages occur at different heights due to obstacles encircling the sphere.

While AtlasRRT ties the use of an atlas to a specific planning algorithm, we will supply a collection of planner-independent algorithms for interpolation and for two kinds of sampling on an atlas-backed configuration space. These algorithms enable many existing sampling-based planners—without modifications—to operate on an implicitly defined

Work on this paper has been supported in part by NSF 1139011, NSF 1317849, and NSF 1514372.

The authors are with the Department of Computer Science at Rice University, Houston, TX 77005, USA. {cav2, mmoll, kavraki}@rice.edu

constraint manifold with all the atlas details hidden behind our work. This is possible because many sampling-based planners interact with the space solely through interpolation and sampling. Within our framework, there is no need to develop a sampling-based planner with special mechanisms for dealing with a manifold. As a consequence of our contribution, pre-existing sampling-based planners that were not able to operate under kinematic constraints can now solve such problems.

Our work is inspired by the excellent results in [4]. We devise a generalization of AtlasRRT, whose design philosophy does not rely on specific properties of RRT other than its sampling-based nature. Expecting that we can substitute another sampling-based planner that may perform better, we isolate operations that directly involve the manifold from any planner decisions, allowing the planning algorithm to be agnostic of the atlas. It is not immediately clear that such encapsulation is possible, as the atlas and the planner rely on one another for functionality. The atlas is constructed only because it is driven by the planner's exploration attempts. But the planner can only explore near the volume already covered by the atlas. For the ensemble to work, a typical sampling-based planner needs two operations to be provided by the atlas: sampling and interpolation.

Not all planners sample in the same way. Some use uniform sampling, which is approached in [4]. Others instead specify a ball in which to sample, so we devise a means of performing this on an atlas. There are also two distinct ways that a planner can use interpolation. The first is to query a single state that lies between two endpoints. The second is to query many such states at high resolution for collision checking. In a Euclidean space, these are one and the same. But on a manifold, the atlas can naturally supply the second through incremental stepping and projection. We are then able to implement the first on top of the second by reasoning over the intermediate points.

Our framework allows rapid substitution of different planners into the same problem specification. We call our work Atlas + X, where X is the chosen planner. The advantage is that we are no longer constrained to a single sampling-based planning algorithm that is integrated with the atlas. By providing functionality to sample and interpolate using an atlas, many existing sampling-based planners become suitable candidates for solving problems on implicit manifolds in high-dimensional spaces, without any modification to adapt them to an atlas approach. One can then choose, for example, an appropriate and performant flavor of RRT, such as the bidirectional RRTConnect [6] or the asymptotically optimal RRT* [7], or use different planner algorithms altogether, like PRM [8], which can produce a roadmap instead of a single path, or EST [9], which directs it search toward less explored regions. See Figure 1 for an example of Atlas + RRT* finding the shortest path on a spherical manifold with obstacles.

We show that Atlas + X planning times are often an order of magnitude faster than CBiRRT2, noting that CBiRRT2 is designed for constraint manifolds, while X is not aware of the constraints. We also find examples of Atlas + X planners, including a unidirectional one, that can outperform the original bidirectional AtlasRRT. We demonstrate that there can be significant disparity in planning time across different choices of X, emphasizing that one planner may be better suited for a given problem than another.

Our contribution is beneficial in allowing the planner to be selected from among many known algorithms without requiring any modification to use the atlas. This result effectively extends the pre-existing sampling-based planners to handle constraint manifold problems, even where no prior effort has been made to do so. The chosen planning algorithm can operate without understanding the atlas-based mechanics of the configuration space.

II. Background and Related Work

Much of the literature in robotic motion planning for high-dimensional systems is devoted to sampling-based planning techniques. These are techniques that randomly explore the configuration space of the system by attempting to reach sampled states. PRM [8] constructs a roadmap of the free space by connecting neighboring samples. RRT [5] incrementally constructs a tree of motions by iteratively expanding the tree from a node that is closest to a uniformly randomly sampled configuration. Other sampling-based planning algorithms guide exploration based on an estimation of sample density by iteratively biasing tree growth toward less densely sampled parts of configuration space. Examples of algorithms that use this general technique include EST [9], KPIECE [10], and STRIDE [11].

Some robotic systems are subjected to constraints for particular tasks, restricting the set of valid configurations to a manifold, which has measure zero with respect to the ambient space. Naturally, sampling can no longer effectively produce valid configurations. Some approaches target specific kinds of constraint problems, such as open- or closed-loop kinematic linkage constraints [12], [13], end-effector pose constraints [14], or n-point contact constraints [15]. Some approaches project invalid samples through gradient descent [16] or by more robust means [17]. A successful algorithm for planning under abstract constraints is CBiRRT2 [2]. Like RRT it steps toward sampled points, but after every step it projects the current point back to the manifold.

In [4] the AtlasRRT algorithm was proposed. During planning, it constructs a piecewise linear approximation of the manifold by computing tangent spaces. This approximation allows the planner to reason about a difficult, implicitly defined space by projecting regions onto simpler spaces. AtlasRRT resolves some problems that arise with CBiRRT2. Namely, CBiRRT2 struggles to achieve uniform exploration since the ambient space does not project uniformly to the manifold; AtlasRRT approaches uniform sampling of the manifold directly by sampling from the tangent spaces. AtlasRRT has been demonstrated to outperform CBiRRT2 on a number of problems, so we will likewise compare Atlas + X with CBiRRT2.

In similar spirit to AtlasRRT, a very recent paper [18] also proposes to use a set of tangent spaces to address the problem of reasoning on a curved domain. A key contribution of Tangent Bundle RRT (TB-RRT), the algorithm described in [18], is that projection from the tangent space to the manifold is done lazily. This saves significant computation time; however, it requires the collision detector to operate on preliminary points in a path that may be adjusted later during projection. TB-RRT is suitable if one can dilate the obstacles to suppress the discrepancy or if the issue is tolerably rare. TB-RRT has a bidirectional RRT integrated into the algorithm, very much like AtlasRRT. Unlike AtlasRRT, which coordinates the tangent spaces patches to minimize overlap, TB-RRT samples from tangent space domains that can overlap significantly, so uniformity is not achieved. In contrast, our atlas-based approach samples uniformly over the explored part of the manifold and, in the limit, achieves uniform coverage of the entire manifold.

Another approach to good manifold sampling, which works for the configuration space of a closed-loop kinematic chain [19], is to sample a subset of the dimensions, leaving only finitely many possible configurations to choose from. However, this technique requires an appropriate IK solver to find such solutions, which we do not assume to be available.

Interpolating smoothly along a manifold defined by contact points between the robot and its environment has been addressed [15], with the goal of finding a parameterization that satisfies the dynamic constraints of a system. The constraint manifolds we address include, but are not limited to, contact constraints. We do not consider systems with dynamics in this work, so we are satisfied with C^0 paths; however, one could refine the output path or roadmap with a more advanced interpolator.

III. Definitions

We consider an n-dimensional Euclidean state space with $n-k$ constraints. More precisely, $n-k$ is the number of degrees of freedom lost due to the constraints. These constraints are given by a differentiable constraint function $\mathbf{F} : \mathbb{R}^n \to \mathbb{R}^{n-k}$ such that $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ precisely defines the points where the constraints are satisfied. Thus, $\mathbf{F}$ implicitly defines a manifold $\mathcal{M}$ of dimension k. We say $\mathbf{x}$ is on the manifold when $||\mathbf{F}(\mathbf{x})|| < \tau$, for some tolerance τ. We refer to n as the *ambient* dimension, k as the *manifold* dimension, and $n-k$ as the *codimension*.

A *chart* is a k-dimensional hyperplane tangent to $\mathcal{M}$ at some point, called the chart's *center*. Points on the chart are represented using an orthonormal basis in which $\mathbf{0}$ coincides with the center. We will follow the notation of [4]. First, the exponential map, $\psi_c : \mathbb{R}^k \to \mathbb{R}^n$, projects a point in chart c onto the manifold in the direction orthogonal to the chart. Second, the change-of-basis function, $\phi_c : \mathbb{R}^k \to \mathbb{R}^n$, rewrites a chart point in ambient space coordinates. We will also invoke the inverse of ψ_c. The only user-supplied information required to compute them is $\mathbf{F}$. An explicit formulation of the Jacobian of $\mathbf{F}$, if available, is also useful, but not necessary since it can instead be computed numerically. The basis for a given chart is the orthonormalized kernel of the Jacobian at its center. The maps ϕ_c and ψ_c^{-1} are operations using this basis and its transpose, respectively. The map ψ_c is implemented as Newton's method on a system involving the chart basis and $\mathbf{F}$. Refer to [4] for a full mathematical treatment of this computation.

The *validity region* $\mathcal{V}_c$ of a chart c is the neighborhood of the chart center in which the chart provides an acceptable approximation of $\mathcal{M}$. We control this by ϵ, the maximum distance between a chart point and its projection on the manifold; α, the maximum angle between the chart and the manifold as it curves away from the chart; and ρ, the maximum radius from the chart center. A polytope, $\mathcal{P}_c \supset \mathcal{V}_c$ is maintained to approximate the validity region. Finally, the *atlas* itself is a collection of charts that covers $\mathcal{M}$ with its validity regions.

IV. Algorithms

The original AtlasRRT works by running a bidirectional RRT with special sampling and interpolation techniques that rely on the atlas. The atlas is not constructed in advance but simultaneously with the planner trees. New charts are created as needed when new regions of the manifold are explored by the trees. The output is a path on the manifold connecting the start and goal states.

Our new scheme is to decouple the atlas from the planner. At a high level, we differ from AtlasRRT by isolating the process of moving along the manifold between two points into TraverseManifold (Algorithm 1). Two points are given as start and end points. We step across the manifold from one to the other in increments of δ (line 8). At each step we move within the local chart in the direction that maximizes progress toward the goal. This is done by projecting both points onto the chart (lines 4–6, 16–18). The chart, of course, is exactly tangent to $\mathcal{M}$ at its center, by construction. Since a chart is only valid within a neighborhood of its center, defined by the parameters ϵ, α, ρ, we switch to another existing or new chart whenever we depart it (line 15). It is simple to check if the manifold point is more than ϵ from the chart plane or if the chart point is more than ρ from the chart center. The angle between the manifold and the chart is estimated using the ratio between the distance traveled in the chart, δ, and the distance traveled in ambient space, d_s, during the most recent step. With appropriate limits on this validity region, local motions in the chart are nearly tangent to the manifold. We proceed until we are within δ of the goal (line 7), or some failure condition occurs, such as a collision, diverging from the goal, or traveling excessively far along the manifold (line 12). Finally, we return the list of every valid point visited on the manifold (line 19).

Note the differences from AtlasRRT[1]. We do not maintain our own tree of motions here, as that is left for the planner. Readers familiar with AtlasRRT know that its manifold traversal algorithm behaves in one of two modes depending on if the purpose is to explore toward a sampled configuration or to connect the two planner trees. It would be impossible for us to make the explore/connect distinction as not all planners use two trees, and our goal is to keep the space logic agnostic of the planning logic. We found that a single traversal mode suffices.

[1]We believe there to be a typographical error on lines 8, 13, 37 of AtlasRRT in [4] (p. 111). The text indicates that the updated point should be a certain distance from the initial point. The formula on its line 13, as given, finds an appropriately scaled vector in the direction of the target, but fails to add this to the initial point. Lines 8 and 37 contain related errors.

Algorithm 1

```
TRAVERSEMANIFOLD($A, \mathbf{x}_a, \mathbf{x}_b$, COLLIDE)
Input: Partial atlas $A$ for some manifold $\mathcal{M}$, states $\mathbf{x}_a$, $\mathbf{x}_b$, with $\mathbf{x}_a$ on $\mathcal{M}$, and COLLIDE : $\mathbb{R}^n \to \{\text{TRUE}, \text{FALSE}\}$ a function to recognize states that are in collision.
Output: A sequence $\{\mathbf{x}_i\}$ of interpolated points from $\mathbf{x}_a$ toward $\mathbf{x}_b$ in free space. $\{\mathbf{x}_i\}$ may not attain $\mathbf{x}_b$.
 1: $j \leftarrow 0$
 2: $d_{trav} \leftarrow 0$
 3: $\mathbf{x}_j \leftarrow \mathbf{x}_a$
 4: $c \leftarrow$ GETORCREATECHART($A, \mathbf{x}_j$)
 5: $\mathbf{u}_j \leftarrow \psi_c^{-1}(\mathbf{x}_j)$
 6: $\mathbf{u}_b \leftarrow \psi_c^{-1}(\mathbf{x}_b)$
 7: while $||\mathbf{u}_b - \mathbf{u}_j|| > \delta$ do
 8:     $\mathbf{u}_{j+1} \leftarrow \mathbf{u}_j + (\mathbf{u}_b - \mathbf{u}_j)\delta/||\mathbf{u}_b - \mathbf{u}_j||$
 9:     $\mathbf{x}_{j+1} \leftarrow \psi_c(\mathbf{u}_{j+1})$
10:     $d_s \leftarrow ||\mathbf{x}_j - \mathbf{x}_{j+1}||$
11:     $d_{trav} \leftarrow d_{trav} + d_s$
12:     if (COLLIDE($\mathbf{x}_{j+1}$)) or $||\mathbf{x}_{j+1} - \mathbf{x}_a|| > ||\mathbf{x}_b - \mathbf{x}_a||$
        or $d_{trav} > 2||\mathbf{x}_b - \mathbf{x}_a||$ then
13:         break
14:     $j \leftarrow j + 1$
15:     if $||\phi_c(\mathbf{u}_j) - \mathbf{x}_j|| > \epsilon$ or $\delta/d_s < \cos\alpha$ or $||\mathbf{u}_j|| > \rho$
        or $\mathbf{u}_j \notin \mathcal{P}_c$ then
16:         $c \leftarrow$ GETORCREATECHART($A, \mathbf{x}_j$)
17:         $\mathbf{u}_j \leftarrow \psi_c^{-1}(\mathbf{x}_j)$
18:         $\mathbf{u}_b \leftarrow \psi_c^{-1}(\mathbf{x}_b)$
19: return $\{\mathbf{x}_i\}_0^j$
```

Algorithm 2

```
GEODESICINTERPOLATE($A, \mathbf{F}, \mathbf{x}_a, \mathbf{x}_b, t$)
Input: Partial atlas $A$ for some manifold $\mathcal{M}$ with constraint function $\mathbf{F}$, states $\mathbf{x}_a, \mathbf{x}_b$ on $\mathcal{M}$, and time $t \in [0, 1]$.
Output: A point on $\mathcal{M}$ occurring at time $t$ on the approximated geodesic between $\mathbf{x}_a, \mathbf{x}_b$. Returns $\mathbf{x}_a$ upon failure.
 1: $\{x_i\}_0^j \leftarrow$ TRAVERSEMANIFOLD($A, \mathbf{x}_a, \mathbf{x}_b$, FALSE)
 2: if $||x_b - x_j|| > \delta$ then
 3:     return $\mathbf{x}_a$
 4: $d_0 \leftarrow 0$
 5: for $i \leftarrow 1..j$ do
 6:     $d_i \leftarrow d_{i-1} + ||\mathbf{x}_i - \mathbf{x}_{i-1}||$
 7: Choose $k$ s.t. $d_k/d_j \leq t$ and $d_{k+1}/d_j > t$.
 8: $s \leftarrow (t\ d_j - d_k)/(d_{k+1} - d_k)$
 9: $\mathbf{y} \leftarrow \psi_c(\mathbf{x}_k + s\ (\mathbf{x}_k - \mathbf{x}_{k+1}))$
10: if $||\mathbf{F}(\mathbf{y})|| \geq \tau$ then
11:     return $\mathbf{x}_a$
12: return $\mathbf{y}$
```

Algorithm 3

```
SAMPLEUNIFORM($A, \rho_s$)
Input: Partial atlas $A$ and sampling radius $\rho_s$.
Output: A point sampled uniformly from the region of the manifold covered by $A$.
 1: repeat
 2:     $c \leftarrow$ RANDOMCHART()
 3:     $\mathbf{u}_s \leftarrow$ SAMPLEINBALL($\rho_s$)
 4: until $\mathbf{u}_s \in \mathcal{P}_c$
 5: ADJUSTBOUNDARY($\mathbf{u}_s, c$)
 6: return $\psi_c(\mathbf{u}_s)$
```

All chart management occurs in GETORCREATECHART, which determines to which chart a point belongs. We do this by searching candidates in a nearest-neighbors data structure, indexed on the chart centers. The method uses the validity polytopes to identify the correct chart and ρ to know when all feasible candidates are exhausted, at which time a new chart is created and the boundaries of its neighbors are updated. For some problems the constraint manifold may have singularities (regions of lower dimension), which the algorithm is technically not able to handle. As long as the singularity is not a necessary part of the solution path, we can circumvent this by detecting when a new chart's basis would not be of full rank and treat the point as an obstacle.

TRAVERSEMANIFOLD incrementally interpolates along the manifold in a way that is useful to a planner as it checks whether two states can be connected by a motion. However, a planner may also interpolate at an explicit time point. For this purpose we provide GEODESICINTERPOLATE in Algorithm 2. The sequence of points returned by TRAVERSEMANIFOLD lends itself to a natural attempt at this computation. Simply pass COLLIDE $\equiv$ FALSE to mark the entire space as free, and verify that $\mathbf{x}_b$ was attained (lines 1–3). The accumulated ambient space distance along the returned sequence of points approximates geodesic distance. So one can find the two appropriate adjacent points (lines 4–7) to use in linear interpolation (lines 8–9). We caution that the chart boundaries will continue to be refined as a side-effect of such computations since the atlas is still being constructed. The sequence of points returned by this algorithm can then vary from call to call during the same execution.

Almost all sampling-based planners rely on one or two kinds of sampling: uniform sampling from the space and sampling near an existing state. As we noted, a classical example of the first is RRT, while EST uses the second. Both sampling methods can be implemented if we make a concession on the uniformity. SAMPLEUNIFORM (Algorithm 3) provides pseudo-uniform sampling in a manner very similar to [4]. Through rejection sampling, a point is chosen uniformly from a chart that is itself selected uniformly. There is a natural bias at the frontier of the atlas because those charts are larger, due to the initial polytope significantly overestimating the validity region. To keep this bias constant regardless of manifold dimension k, we will always fix $\rho_s = 2^{1/k}\rho$. The ADJUSTBOUNDARY routine expands the boundary of neighboring polytopes if the sample is close to the edge of the current chart, as this helps with manifold coverage [4].

The final component needed to complete our interface is the second kind of sampling, given in SAMPLENEAR (Algorithm 4). We simply select the chart to which $\mathbf{x}$ belongs and sample uniformly from a tangent-space ball of radius ρ_s around the projection of $\mathbf{x}$ in the chart (lines 3–4). We do not fix ρ_s here, but leave it to be chosen by the planner.

Algorithm 4

```
SAMPLENEAR($A, \mathbf{F}, \mathbf{x}, \rho_s$)
Input: Partial atlas $A$ with constraint function $\mathbf{F}$, state $\mathbf{x}$ on the manifold, and sampling radius $\rho_s$.
Output: A point projected to the manifold from a uniform sample from a ball of radius $\rho_s$ in the chart for $\mathbf{x}$.
1: $c \leftarrow$ GETORCREATECHART($A, \mathbf{x}$)
2: repeat
3:     $\mathbf{u}_s \leftarrow \psi_c^{-1}(\mathbf{x}) +$ SAMPLEINBALL($\rho_s$)
4:     $\mathbf{x}_s \leftarrow$ GRADIENTDESCENT($\mathbf{F}, \mathbf{u}_s$)
5: until $||\mathbf{F}(\mathbf{x}_s)|| < \tau$
6: ADJUSTBOUNDARY($\mathbf{u}_s, c$)
7: return $\mathbf{x}_s$
```

It is possible, especially near singularities or areas of high curvature, that it is difficult to project the resultant point onto the manifold. We use gradient descent rather than ψ_c (which projects orthogonally to the chart) in line 4 since we found it much more likely to succeed, thereby decreasing sampling time.

We have now supplied approximate interpolation and the two common kinds of sampling using these algorithms. A planner can operate in the space without knowledge of the atlas, and the atlas construction does not depend on a specific planning algorithm. All that is left is to choose a sampling-based planner, and, without adaptation, it can run on an implicitly defined constraint manifold, with all the atlas details hidden behind this interface.

V. RESULTS

We implemented all of the above algorithms in the Open Motion Planning Library (OMPL) [20], which has existing implementations of many popular sampling-based planners. It is a design philosophy of OMPL to provide an abstract interface by which any planner may use nearly any state space; therefore, it is an appropriate platform for our contribution, which seeks to provide an atlas-based state space and use an arbitrary sampling-based planner on it. All timing data was collected on a 1.4 GHz multicore processor with 64 GB of RAM. For consistency, in all our experiments we use $\epsilon = 0.5, \rho = 0.2, \alpha = \pi/8$. Our choice of these parameters is intended to produce a few large charts so that less time is spent constructing the atlas, and more time is spent using the atlas to solve the problem.

Some of the planners we test use SAMPLEUNIFORM, while others use SAMPLENEAR. Two examples are RRT and EST, respectively. The sampling radius ρ_s serves a slightly different purpose in each of these sampling algorithms. In uniform sampling, we fix ρ_s so that the sampled volume is a constant multiple of the maximum possible chart volume. As a result, points can be sampled past the frontier of a partial atlas, which is what drives the atlas construction. By contrast, ρ_s is chosen explicitly by planners that use SAMPLENEAR. Such a planner can use this selective sampling to move toward regions that it believes are less explored. So in the uniform case, the planner cannot receive samples over the entire space until the atlas is complete, and the exploration is a consequence of the atlas. However, in the sample-near case, the planner is in explicit control of the exploration. We therefore hypothesize that a sample-near planner may be better a candidate for atlas-based planning than a uniformly sampling planner, such as the original AtlasRRT.

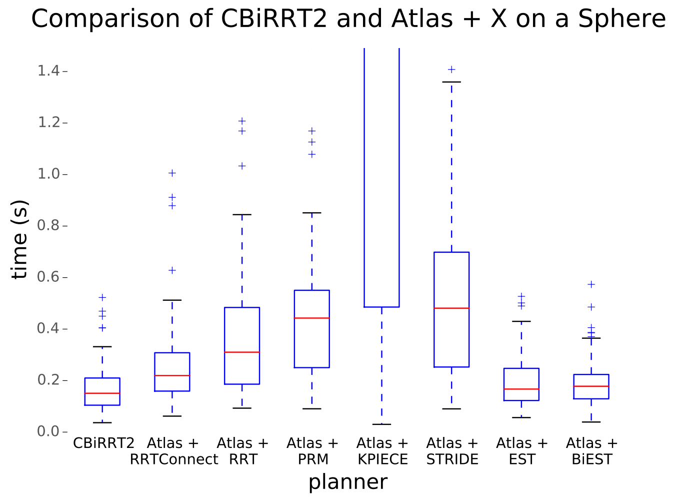

Fig. 2: Comparison of CBiRRT2 against several Atlas + X planners on a 2D spherical manifold (see Figure 1) over 100 runs. Note that RRT(Connect) and PRM invoke SAMPLEUNIFORM, while KPIECE, STRIDE, and (Bi)EST use SAMPLENEAR.

A. Experiments

We begin with a small but illustrative example problem. The unit sphere is defined by a single constraint on 3D ambient space: that the distance to the origin be 1. In this case, both the manifold and the ambient space have low dimension, and the curvature is constant, resulting in charts of approximately equal area, which we have direct control over using the atlas parameters. Uniform samples from ambient space project nearly uniformly onto the sphere in a way that is far from pathological. Mitigating biases in projected samples is a primary motivation for an atlas-based approach [4] over methods like CBiRRT2 that sample in ambient space. We therefore expect CBiRRT2 to perform comparatively well here. We add three latitudinal obstacles around the sphere, creating narrow passages between the start and goal states, which are at the poles. An example solution is shown in Figure 1. The tree of valid motions is constructed by RRT*, which approaches the optimal solution. The best path from start to goal after 0.5 seconds is highlighted in blue. The obstacles are not shown, but their shapes can be inferred from the gaps in the planner tree. The graph conforms to the spherical manifold, and the three empty bands imply where the obstacles lie. For reference, with the given parameters RRT* creates an average of 45 charts in 0.5 s, but if it is allowed to run longer, it will finish covering the freespace of the manifold with a maximum of about 56 charts.

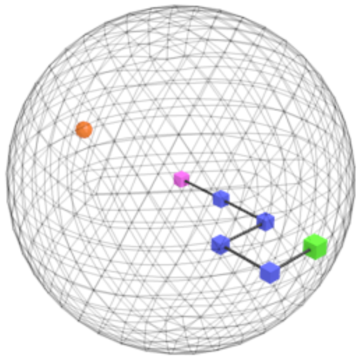

Fig. 3: Diagram of the 5-link kinematic chain, shown in its start configuration. Base (pink) is at the sphere's center. Goal configuration places the end effector (green) at the target on the opposite side (orange). Each link is represented as a distance constraint between adjacent joints. Further constraints can be applied, such as affixing the end effector to the sphere's surface or maintaining a distance between non-adjacent joints.

We compared several Atlas + X planners with CBiRRT2, which does not use an atlas. The execution times of 100 runs are shown in Figure 2. Only CBiRRT2 is designed specifically for planning under constraints. The other planners can only solve the problem because they are augmented by our approach, without which sampling would be nearly impossible. We find CBiRRT2 to be consistently the fastest, narrowly beating the uni- and bidirectional variants of EST. Note that the original AtlasRRT is not equivalent to our Atlas + RRT. AtlasRRT is bidirectional, and it retains every state encountered during traversal of the manifold, rather than just the terminal state in a tree extension. Thus we also compared a variant of Atlas + RRTConnect, which is bidirectional, modified to retain all the intermediate states from a tree extension. So this Atlas + RRTConnect emulates the original AtlasRRT within our framework. It is both expected and observed that Atlas + RRTConnect outperforms the unidirectional Atlas + RRT. We also compared KPIECE and STRIDE using our approach. Like EST, they estimate coverage in different regions of the space to guide the search. In particular, they use SAMPLENEAR to choose where to explore. KPIECE conspicuously struggles to compete with the other planners on this example.

Having measured the relative performance of Atlas + X for various planners in a very simple environment, we go on to explore the relationship between dimensionality and planning time in a more complex system. We define a 5-link kinematic chain robot with universal joints, as depicted in Figure 3. Rather than model the joint kinematics explicitly, we build a 15-dimensional ambient configuration space from the 3D position of each joint and the end effector, and then constrain the distance between adjacent joints to a fixed link length. This gives rise to a 10-dimensional manifold, since there are five links. To create obstacles in the space, we forbid self-intersections.

The codimension of the problem, which is the difference between the ambient space dimension and constraint manifold dimension, is equal to the number of DOFs lost due to constraints. So the codimension of the model is currently 5. To achieve a codimension greater than 5, we can impose additional constraints, giving successively higher codimensions as follows: We first affix the end effector to a sphere (see Figure 3); this extra distance constraint brings the total codimension up to 6. We then additionally require the z coordinates of the first and second joints to be equal, giving a codimension of 7. Next we also require that the x coordinate of the second and third be equal, for a codimension of 8; that the y coordinate of the third and fourth be equal, for a codimension of 9; and finally that the y coordinate of the first joint and the end effector be equal, ending with a codimension of 10.

We give the median planning times over 100 runs for the increasing set of constraints in Table I. Atlas + X is always an order of magnitude faster than CBiRRT2. We have highlighted the minimum and near-minimum (within 10%) times in each column. Observe that the increasing codimension does not always correspond to an increase in planning time for Atlas + X, but it very clearly does for CBiRRT2. To visualize some of the data, we have plotted the planning times for codimension 6 (Figure 4). Notice that Atlas + KPIECE beats the Atlas + RRTConnect (our stand-in for AtlasRRT) on codimension 6, but that CBiRRT2, the non-atlas competitor, outperforms Atlas + PRM after about 7 seconds. We have also plotted codimension 9 in Figure 5 since it is a problem for which every Atlas + X planner performs exceptionally well (cf. codimensions 8 and 10 in Table I), whereas CBiRRT2 gives no indication that the problem is easier.

We also varied the ambient dimension of the problem. By allowing the chain to move in the hypothetical four-dimensional workspace $\mathbb{R}^4$ instead of $\mathbb{R}^3$, we can increase the ambient and manifold dimensions by 5 (one per joint). Planning in a $\mathbb{R}^5$ increases the dimensionality by another 5. Table II gives the extended data of these higher dimensions for selected codimensions. Across all the dimensions, Atlas + KPIECE still competes well with Atlas + RRTConnect and often outperforms it (highlighted in bold). As we noted before, the behaviors of CBiRRT2 and of Atlas + X do not coincide in response to changing the codimension, especially apparent for codimension 9. However, they do coincide with respect to

TABLE I: Median planning time (seconds) over 100 runs on the kinematic chain problem under increasing constraints (codimension). Best planner(s) in each column are highlighted in bold text. Note that CBiRRT2 performs steadily worse, while Atlas + X exhibits more complicated behavior.

	CODIMENSION				
	6	**7**	**8**	**9**	**10**
CBiRRT2	4.31	6.48	6.93	9.38	11.56
Atlas + RRTConnect	0.47	**0.39**	**0.27**	**0.21**	0.64
Atlas + RRT	1.70	1.80	0.45	0.33	1.64
Atlas + PRM	2.46	1.37	1.84	0.43	0.93
Atlas + KPIECE	**0.40**	**0.41**	0.37	**0.20**	**0.57**
Atlas + STRIDE	0.78	0.78	0.50	0.33	1.36
Atlas + EST	2.20	1.07	1.54	0.47	4.18
Atlas + BiEST	1.12	0.77	1.23	0.42	1.09

TABLE II: Median planning time (seconds) over 100 runs on the kinematic chain problem under selected constraints (codimension) in workspaces of 3, 4, and 5 dimensions. Best planner(s) in bold; easiest workspace dimension in italics.

	CODIMENSION = 6			CODIMENSION = 8			CODIMENSION = 10		
WORKSPACE:	$\mathbb{R}^3$	$\mathbb{R}^4$	$\mathbb{R}^5$	$\mathbb{R}^3$	$\mathbb{R}^4$	$\mathbb{R}^5$	$\mathbb{R}^3$	$\mathbb{R}^4$	$\mathbb{R}^5$
CBiRRT2	4.31	*4.01*	4.30	6.93	*6.11*	7.10	11.56	*10.00*	12.13
Atlas + RRTConnect	*0.47*	0.52	0.68	***0.27***	**0.30**	0.39	0.64	***0.31***	0.40
Atlas + RRT	*1.70*	2.00	2.60	0.45	*0.42*	0.57	1.64	*0.43*	0.54
Atlas + PRM	2.46	*1.44*	1.87	1.84	*1.02*	1.32	0.93	*0.58*	0.71
Atlas + KPIECE	**0.40**	**0.43**	***0.38***	0.37	***0.31***	**0.34**	**0.57**	***0.30***	**0.33**
Atlas + STRIDE	0.78	*0.77*	0.86	0.50	*0.44*	0.56	1.36	*0.50*	0.54
Atlas + EST	2.20	*1.38*	2.17	*1.54*	1.17	1.58	4.18	*1.80*	1.98
Atlas + BiEST	1.12	*0.88*	1.12	1.23	0.49	*0.47*	1.09	*0.42*	0.52

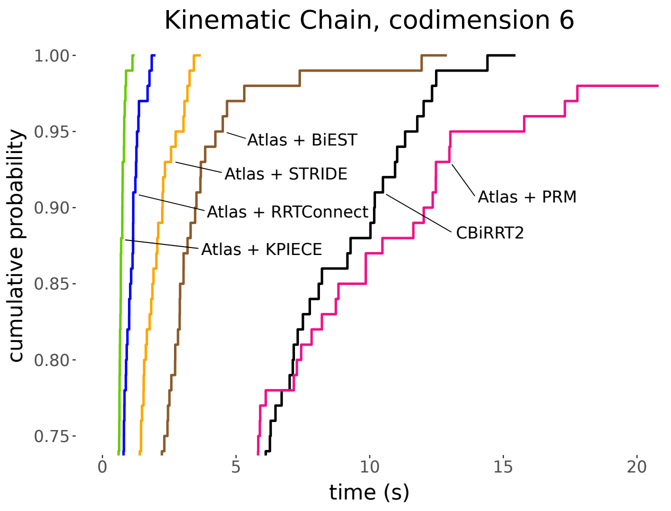

Fig. 4: Kinematic chain with end effector constrained to a sphere. This is modeled as 15 dimensional system with 6 constraints. Cumulative distribution shows what fraction of the 100 runs completed within a given time.

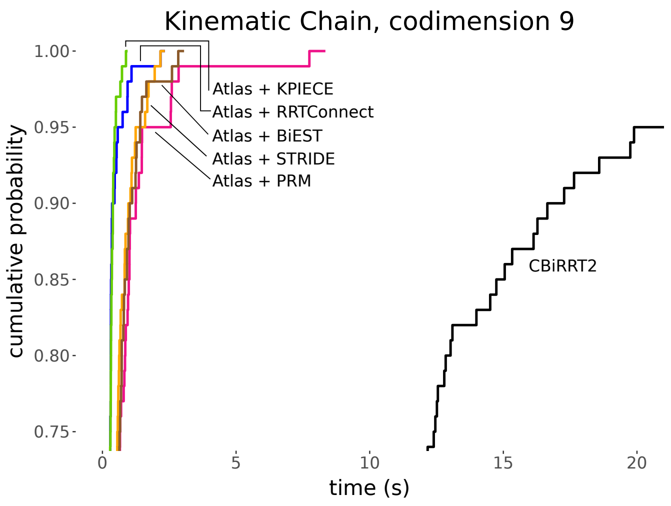

Fig. 5: Kinematic chain with end effector constrained to a sphere and three additional constraints between joints. Cumulative distribution again taken over 100 runs. Note the exceptional disparity between CBiRRT2 and Atlas + X.

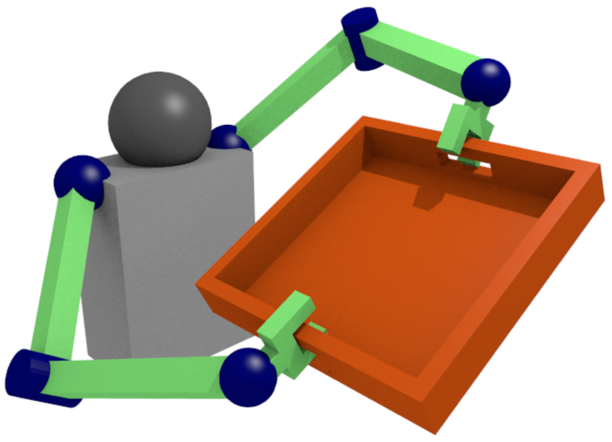

Fig. 6: Diagram of the bimanual robot holding a tray. Each shoulder and wrist has 3 rotational DOFs, while the elbows each have 1. Thus the total ambient dimension is 14. Holding the tray and imposing further constraints on the tray's orientation creates manifolds of successively lower dimension.

the ambient dimension. There is general agreement that the 4D problem is easiest (italicized), but it is not clear whether 3D or 5D is the hardest.

For a more practical example, we modeled a bimanual task performed by a robot with 7 revolute joints in each of its two arms: the shoulder and wrist each have three, and the elbow has one. Thus the unconstrained ambient space of our model has dimension 14. The planning query is to move a tray from a starting position just within reach of the robot to a goal position close to the torso. See the illustration of the robot in Figure 6. To hold the tray, the hands must remain a fixed distance apart, taking away one DOF. Second, we can additionally require that the tray not roll left or right around one axis. Third, we can additionally require the robot to hold the tray level (no rotation in two axes). For this problem, we do not explicitly provide the Jacobian of the constraint function to the program, illustrating that it can be computed numerically if unavailable. Table III shows a very similar story to what was observed with the kinematic chain. CBiRRT2 is an order of magnitude slower than any Atlas + X planner. But notably, this time it is Atlas + PRM and Atlas + BiEST that compete with Atlas + RRTConnect. The three unidirectional SAMPLENEAR planners, KPIECE, STRIDE, and EST, consistently outperform RRT, which is a unidirectional SAMPLEUNIFORM planner.

TABLE III: Median planning time (seconds) over 100 runs on the bimanual manipulator problem under increasing constraints (codimension): hold a tray, refrain from tilting the tray left or right, and hold the tray completely level. Best planners in bold.

	CODIMENSION		
	1	**2**	**3**
CBiRRT2	9.99	12.74	17.94
Atlas + RRTConnect	**0.19**	**0.20**	**0.25**
Atlas + RRT	0.88	0.78	0.63
Atlas + PRM	**0.20**	**0.21**	0.47
Atlas + KPIECE	0.55	0.50	0.58
Atlas + STRIDE	0.43	0.43	0.53
Atlas + EST	0.48	0.52	0.67
Atlas + BiEST	**0.19**	**0.20**	**0.25**

B. Discussion

As we noted, planning on the spherical manifold is the simplest of the problems we considered. The ambient and manifold spaces differ by only one dimension; the manifold exhibits uniform curvature; its embedding lends itself to near-uniform sampling by projecting ambient samples. For all these reasons, it is unsurprising that Atlas + X, which is an elaborate approach requiring a non-trivial amount of bookkeeping, performs worse than CBiRRT2, which takes a more direct but less informed approach without any bookkeeping.

One striking feature of the results is how KPIECE struggles so much more than the others to find a solution on the supposedly easy problem, but is one of the fastest planners on the higher dimensional problem with a kinematic chain. KPIECE maintains a discretization of the space and uses SAMPLENEAR, instead of SAMPLEUNIFORM, to intentionally target specific regions. We believe the performance to be related to this strategy. Planners that do not try to sample selectively will still naturally explore from the frontier charts of the partial atlas. This is because those charts do not yet have fully formed boundaries, and thus are much larger. The atlas effectively provides a discretization of the space that naturally assists in exploring the frontier. However, KPIECE performs much more bookkeeping to maintain its own cellular discretization for this purpose, beyond the one provided by the atlas. The work involved is excessive for a simple problem like a spherical manifold, but can ultimately pay off in higher dimensions.

In understanding why Atlas + X in general does better than CBiRRT2 on the higher dimensional kinematic chain and bimanual problems, and why Atlas + X responds so well to an increase in codimension, while CBiRRT2 does not, we should look at two factors: the quality of samples and the cost of projection. It is difficult to comprehend the shape the manifold takes on in such high dimensions; however we can build some intuition about what happens to ambient-space samples. Consider the location of the first of the five kinematic chain joints. It is constrained to the sphere, $\mathbb{S}^2$. Uniform sampling in an $\mathbb{R}^3$ bounding box already fails to given uniform coverage when projected to $\mathbb{S}^2$. Now consider the second joint. It must lie on a sphere centered at the first joint, forming a torus-like manifold $\mathbb{S}^2 \times \mathbb{S}^2$. It suffers the same deficiency that we described for the ordinary torus ($\mathbb{S}^1 \times \mathbb{S}^1$), with samples rarely projecting onto the region surrounding the hole. Continuing down the chain, this bias is exacerbated by each subsequent joint. Therefore, projecting uniform samples from ambient space to the constraint manifold produces poor coverage of configurations where the chain must bend tightly. This is one of the primary advantages of Atlas + X over CBiRRT2: it approaches uniform coverage of the manifold and encourages exploration of unreached regions.

Secondly, Atlas + X and CBiRRT2 perform projections differently, and the speed is affected by the codimension. For CBiRRT2, the only information known about the manifold is the behavior of the constraint function at the current sample; it can project to the manifold using the pseudoinverse of the Jacobian. The matrix to pseudoinvert is $(n-k)$-by-n, where n and k are the ambient and manifold dimensions, respectively. That is, $n-k$ is the codimension, or degrees of freedom that are constrained. Hence, as codimension increases, so does the size of the system CBiRRT2 must solve. Atlas + X, however, inverts an n-by-n matrix consisting of the Jacobian and the chart basis, since it aims to project orthogonally from the chart. The system to solve is thus larger than that of CBiRRT2; however, it is invariant with respect to codimension. Both projections use an iterative Newton method, but since Atlas + X takes samples from charts which approximate the manifold, we expect an Atlas + X projection to take fewer iterations to converge than a CBiRRT2 projection. As the codimension increases and manifold dimension decreases, the bookkeeping of charts becomes less expensive. It is for these reasons that we believe Atlas + X responds well to an increase in the codimension while CBiRRT2 does not.

VI. CONCLUSION

We have introduced a new framework, Atlas + X, which provides planner-independent algorithms for interpolation and sampling in an atlas-backed constraint manifold space. The framework allows existing sampling-based planners to operate on such manifolds in a space-agnostic way. While prior work combined an atlas representation with a specific planning algorithm, here we have shown the advantages of decoupling them. Our contribution immediately opens the possibility of applying existing sampling-based planning algorithms to motion problems on constraint manifolds, even where no prior effort has been made to adapt a given planner for such a problem. Future space-agnostic planners will also be suitable for use in our framework without modification.

Our experiments show that such Atlas + X planners vary in their abilities to solve specific problems. On a simple, low-dimensional problem we found the alternative approach of CBiRRT2 to be a better choice. But in higher dimensions and on more complicated manifolds, such as that of a kinematic chain or bimanual manipulator, Atlas + X is an order of magnitude faster for many choices of X, such as RRTConnect, KPIECE, and BiEST. As evidence that some planners are more suited to certain problems, though one expects a bidirectional planner to be faster than a unidirectional one, we found an example of the unidirectional Atlas + KPIECE outperforming the bidirectional Atlas + RRTConnect, which emulates the original AtlasRRT algorithm. This again strongly demonstrates the advantage of our generalized technique.

With our contribution, the robotics community is now equipped to approach constraint manifold problems using a wide variety of planning algorithms. The next step is to apply the atlas approach to multiple intersecting manifolds or stratified manifolds, along with a proper method of handling singularities. A bimanual robot may choose to grasp a large item in any number of configurations, each giving rise to a different manifold. A legged robot moves under multi-modal constraints depending on which feet are currently contacting the ground. Writing on a surface is a simpler example. While the pen is in the air, the robot moves through an n-dimensional manifold, but while it touches the paper for a stroke, motion is confined to an $(n-1)$-dimensional sub-manifold. Such problems may require coordinating motion graphs across multiple atlases, with special care taken at the intersections, which are themselves manifolds of lower dimension. See, for example, [21] for some early work on planning across different intersecting manifolds.

It is an open question whether there will be any advantage to using higher-order approximations of the manifold in an atlas. For example, can we significantly reduce the number of charts required to cover a manifold if the charts are quadratic patches instead of linear? Will this pay off, considering the computational cost of the additional math involved?

With our current formulation, the implicit manifold specification does not suit every constraint-based problem since it assumes the underlying ambient space is Euclidean. However, a given system may be better modeled by an ambient space with different topology. For example, $SE(3)$ has a non-Euclidean subspace, and thus any interesting manifold embedded in this space would no longer be embedded in $\mathbb{R}^n$. Therefore, some care must be taken to ensure the mathematics of the manifold traversal will be correct. Then it will be possible to operate arbitrary sampling-based planners on manifolds embedded in more complicated spaces.

References

[1] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press, 2005.

[8] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, pp. 566–580, 1996.

[2] D. Berenson, S. S. Srinivasa, and J. J. Kuffner, "Task Space Regions: A framework for pose-constrained manipulation planning," *Int. J. Rob. Res.*, vol. 30, pp. 1435–1460, 2011.

[3] M. E. Henderson, "Multiple parameter continuation: Computing implicitly defined k-manifolds," *Int. J. Bifurc. Chaos*, vol. 12, pp. 451–476, 2002.

[4] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *IEEE Trans. Robot.*, vol. 29, pp. 105–117, 2013.

[5] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Rob. Res.*, vol. 20, pp. 378–400, 2001.

[6] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An efficient approach to single-query path planning," in *IEEE Int. Conf. Robot. Autom.*, 2000, pp. 995–1001.

[7] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Rob. Res.*, vol. 30, pp. 846–894, 2011.

[9] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," *Int. J. Comput. Geom. Appl.*, vol. 9, pp. 495–512, 1999.

[10] I. A. Şucan and L. E. Kavraki, "A sampling-based tree planner for systems with complex dynamics," *IEEE Trans. Robot.*, vol. 28, pp. 116–131, 2012.

[11] B. Gipson, M. Moll, and L. E. Kavraki, "Resolution independent density estimation for motion planning in high-dimensional spaces," in *IEEE Int. Conf. Robot. Autom.*, 2013, pp. 2437–2443.

[12] X. Tang, S. Thomas, P. Coleman, and N. M. Amato, "Reachable distance space: Efficient sampling-based planning for spatially constrained systems," *Int. J. Rob. Res.*, vol. 29, pp. 916–934, 2010.

[13] J. Cortés and T. Siméon, "Sampling-based motion planning under kinematic loop-closure constraints," in *Algorithmic Foundations of Robotics VI*, M. A. Erdmann, D. Hsu, M. Overmars, and A. F. van der Stappen, Eds. Springer, 2005, pp. 75–90.

[14] D. Berenson and S. S. Srinivasa, "Probabilistically complete planning with end-effector pose constraints," in *IEEE Int. Conf. Robot. Autom.*, 2010, pp. 2724–2730.

[15] K. Hauser, "Fast interpolation and time-optimization with contact," *Int. J. Rob. Res.*, vol. 33, pp. 1231–1250, 2014.

[16] J. H. Yakey, S. M. LaValle, and L. E. Kavraki, "Randomized path planning for linkages with closed kinematic chains," *IEEE Trans. Robot. Autom.*, vol. 17, pp. 951–958, 2001.

[17] P. Kaiser, D. Berenson, N. Vahrenkamp, T. Asfour, R. Dillmann, and S. S. Srinivasa, "Constellation—An algorithm for finding robot configurations that satisfy multiple constraints," in *IEEE Int. Conf. Robot. Autom.*, 2012, pp. 436–443.

[18] B. Kim, T. U. Taewoong, C. Suh, and F. C. Park, "Tangent bundle RRT: A randomized algorithm for constrained motion planning," *Robotica*, vol. 34, pp. 202–225, 2016.

[19] Y. Zhang, K. Hauser, and J. Luo, "Unbiased, scalable sampling of closed kinematic chains," in *IEEE Int. Conf. Robot. Autom.*, 2013, pp. 2459–2464.

[20] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robot. Autom. Mag.*, vol. 19, pp. 72–82, 2012.

[21] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *Int. J. Rob. Res.*, vol. 23, pp. 729–746, 2004.