Path Planning Under Kinematic Constraints by Rapidly Exploring Manifolds

Léonard Jaillet and Josep M. Porta

***Abstract*—The situation arising in path planning under kinematic constraints, where the valid configurations define a manifold embedded in the joint ambient space, can be seen as a limit case of the well-known narrow corridor problem. With kinematic constraints, the probability of obtaining a valid configuration by sampling in the joint ambient space is not low but null, which complicates the direct application of sampling-based path planners. This paper presents the AtlasRRT algorithm, which is a planner especially tailored for such constrained systems that builds on recently developed tools for higher-dimensional continuation. These tools provide procedures to define charts that locally parametrize a manifold and to coordinate the charts, forming an atlas that fully covers it. AtlasRRT simultaneously builds an atlas and a bidirectional rapidly exploring random tree (RRT), using the atlas to sample configurations and to grow the branches of the RRTs, and the RRTs to devise directions of expansion for the atlas. The efficiency of AtlasRRT is evaluated in several benchmarks involving high-dimensional manifolds embedded in large ambient spaces. The results show that the combined use of the atlas and the RRTs produces a more rapid exploration of the configuration space manifolds than existing approaches.**

***Index Terms*—Higher-dimensional continuation, kinematic constraints, manifolds, path planning.**

I. Introduction

THE problem of path planning, i.e., the determination of how to move a robotic system from an initial to a goal configuration avoiding collisions, is ubiquitous in robotics as it appears in most of the addressed tasks [1], [2]. Sampling-based path planners [3], [4] have been largely successful and are the standard for industry-level solutions [5]. These planners rely on the fact that while representing the obstacles in configuration space is hard, checking whether a particular configuration is in collision or not is relatively simple. However, these planners have difficulties in the so-called narrow corridor problems, where the solution must necessarily traverse a tiny area, i.e., an area with relatively low probability of being sampled. The case where the problem includes kinematic constraints can be seen as a limit case of the narrow corridor problem. In this case, the constraints define a configuration space that is a zero-measure manifold embedded in the ambient space formed by the robot joint variables [6]. Thus, the probability of directly sampling on the configuration space by selecting random joint values is not just low, but null, which complicates the use of sampling-based path planners.

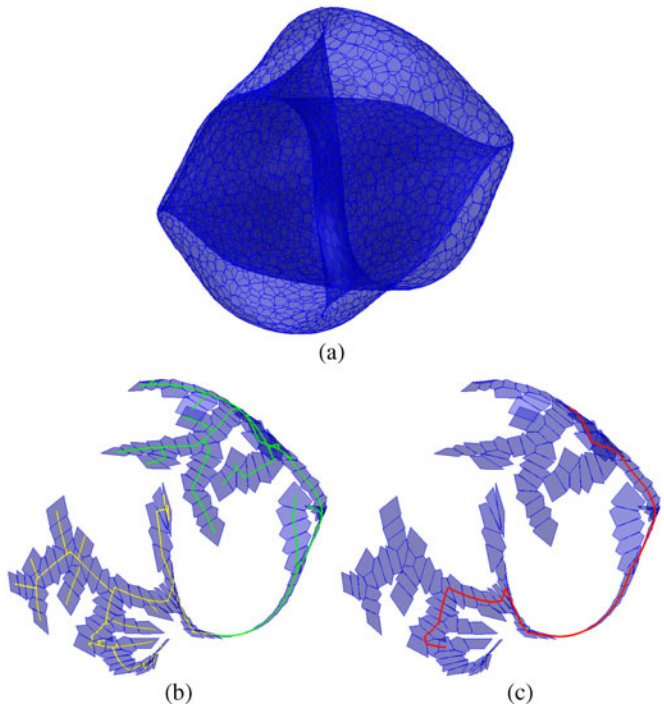

Fig. 1. Example of exploration with AtlasRRT. (a) Full atlas of the bidimensional configuration space of the cyclooctane. (b) AtlasRRT intertwines the construction of a bidirectional RRT with an atlas construction. The trees rooted at the start and goal configurations are represented in yellow and green, respectively. (c) When the two RRTs are connected, a solution path (represented in red) can be readily computed. Observe that only a small fraction of the full atlas is necessary to connect the query configurations.

Path planning under kinematic constraints is a classical topic in robotics [7]–[10], and it appears, for instance, in complex manipulation problems [11], parallel robots [12], robot grasping [13], constraint-based object positioning [14], or surgery robots [15]. This problem is also crucial in Biochemistry, when analyzing the conformational changes in molecular loops [16]. Moreover, the popularization of robots, such as two-armed service robots [17], anthropomorphic hands [18], or humanoid robots [19], where loop closures appear very often, has strongly pushed the research in this field over the past few years [20]–[25]. In the quest of a general and efficient solution, most of the existing approaches try to adapt the sampling-based planners to the constrained case.

Manuscript received March 9, 2012; revised July 17, 2012; accepted September 26, 2012. Date of publication November 16, 2012; date of current version February 1, 2013. This paper was recommended for publication by Associate Editor D. Hsu and Editor W. K. Chung upon evaluation of the reviewers' comments. This work was supported in part by the Spanish Ministry of Economy and Competitiveness under Project DPI2010-18449. The work of L. Jaillet was supported by the Spanish Council for Scientific Research under a JAE-Doc fellowship partially funded by the European Social Fund.

The authors are with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona 08028, Spain (e-mail: ljaillet@iri.upc.edu; porta@iri.upc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2012.2222272

The efficiency of the sampling-based path planning approaches relies on the ability of drawing samples and of growing branches that cover the configuration space. If some information about the environment is available, the distribution of samples can be adapted to the problem [26]. However, in the absence of such information, a uniform distribution is preferred. This distribution of samples can only be easily obtained using an isometric parametrization of the configuration space. Moreover, the branch extension is typically based on linear interpolation between samples, which also relies on a parametrization of the configuration space. Although, for unconstrained systems, such parametrization is straightforward, this is not the case when the kinematic constraints reduce the dimensionality of the configuration space. Due to this issue, existing attempts to adapt the sampling-based planners to problems with kinematic constraints are either limited to particular families of mechanisms or unable to efficiently explore the configuration space.

To address these limitations, we propose here a method called AtlasRRT based on a coordinated construction of an atlas and a bidirectional rapidly exploring random tree (RRT). On the one hand, the atlas is used to adequately sample configurations and grow branches on the configuration space manifold and, thus, to retain the RRT exploration efficiency, despite working on a non-Euclidean configuration space. On the other hand, the RRT is used to determine directions of expansion for the atlas so that the charts generated are those useful to find solution paths. An example of problem solved with AtlasRRT is shown in Fig. 1, which represents the 2-D configuration space of the cyclooctane molecule, a bidirectional RRT, and the solution path obtained with the proposed approach. AtlasRRT is an evolution of the algorithm introduced in [27]. Here, the change of the underlying formulation, the improvements in the coordination between the RRT and the atlas, and the new strategies to control the growth of the trees result in a significant speedup with respect to the preliminary version of the algorithm. Moreover, small modifications are introduced in the algorithm to guarantee its probabilistic completeness.

This paper is organized as follows. Section II frames the proposed planner in the context of the previous work. Then, Section III introduces the concepts of charts and atlas and how to use them to represent an implicitly defined manifold. Section IV presents a way to integrate the atlas-based manifold representation with an RRT exploration. Section V formally describes the algorithms implementing the AtlasRRT planner, and Section VI compares its performance to state-of-the-art methods for several benchmarks involving high-dimensional manifolds embedded in large ambient spaces. Finally, Section VII summarizes the contributions and limitations of this study and indicates points that deserve further attention.

II. Related Work

As already mentioned, the main issue for path planning under kinematic constraints is to devise proper ways to sample the configuration space manifold and to connect the samples between them.

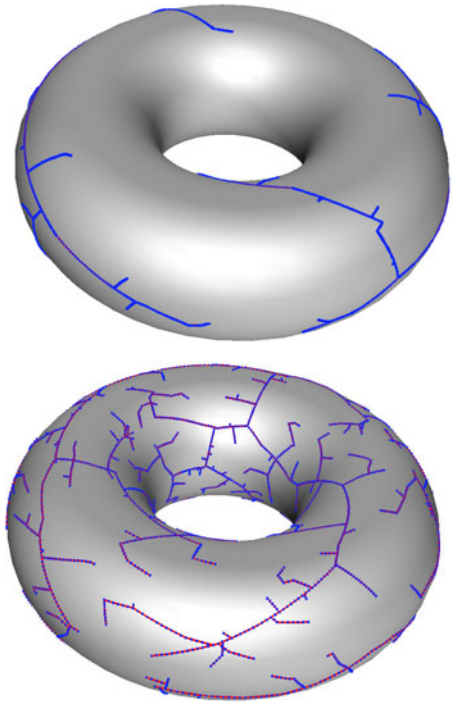

Fig. 2. Two RRTs of 500 samples built on a torus-like manifold. (Top) With an ambient space sampling method, the exploration focuses on the outer parts of the torus, and many samples do not produce a tree extension. (Bottom) With an AtlasRRT, the diffusion process is largely independent of the ambient space, which improves the coverage.

For some particular families of mechanisms with kinematic loops, distance-based formulations provide a global parametrization that can be readily used to sample and to grow branches in the configuration space [23], [28]. Other approaches try to learn the parametrization from large sets of samples on the configuration space [22], but they can only deal with relatively simple manifolds.

In the absence of a global parametrization, Kinematics-PRM [8] samples a subset of joint variables and uses inverse kinematics to find values for the remaining ones. Unfortunately, this strategy is not valid for all the mechanisms, and although some improvements have been proposed [29], the probability of generating invalid samples is significant. Finally, the various solutions of the inverse kinematic functions and the presence of singularities complicate the approach [30]. Task-space planners [24], [31], [32] are similar in the sense that they sample on a subset of variables (those related with the end-effector), although they typically determine values for the remaining degrees of freedom using numerical techniques instead of closed kinematic functions. Thus, they share the problems of kinematic-based approaches regarding the multiple solutions for the variables not fixed.

An alternative strategy to get valid configurations is to sample in the joint ambient space and to converge to the configuration space either implementing random walks [9] or the more efficient Jacobian pseudo-inverse method [20], [21], [33]. These approaches are probabilistically complete [34] and easy to implement, but a uniform distribution of samples in the ambient

space does not necessarily translate to a uniform distribution in the configuration space [24], and the branch extensions are many times prematurely blocked, as noted in [35], which reduces their efficiency. This problem is illustrated in Fig. 2 where the configuration space to be explored is a torus-like manifold of diameter four times smaller than the ambient space width. The top part of Fig. 2 shows an RRT built from points sampled in the ambient space that has a poor coverage of the manifold. With the Atlas-RRT presented in this paper, the process of diffusion is largely independent of the configuration space shape and the ambient space bounds which improves the coverage of the manifold, as shown at the bottom of Fig. 2.

To increase the quality of the exploration, one can focus on a subset of the ambient space around the configuration space [36]. With this method, however, points are still sampled in the ambient space, which can be of much higher dimensionality than the configuration space. Suh *et al* [35] introduce a lazy RRT scheme, where loosely coordinated RRTs are built on tangent spaces that locally approximate the manifold and that have the same dimensionality as the configuration space. However, the quality of the resulting RRT is affected by the possible overlap of tree branches that belong to different tangent spaces.

From differential geometry, it is well known that a manifold can be described by a collection of local parametrizations called charts, which can be coordinated within an atlas [37]. Higher-dimensional continuation techniques provide principled numerical tools to compute the atlas of an implicitly defined manifold starting from a given point, whereas minimizing the overlap between neighboring charts [38], [39]. One-dimensional continuation methods have been strongly developed in the context of dynamical systems [40], whereas in robotics, they have been mainly used to solve problems related to kinematics [41], [42]. To the best of our knowledge, higher-dimensional continuation methods have only been used in robotics to evaluate the dexterity of mechanisms [43], [44]. In a previous work [25], we introduced a resolution complete path planner on manifolds based on higher-dimensional continuation tools. Despite its efficiency, this planner relies on a discretization of the manifold and the exploration could be blocked in the presence of narrow corridors, unless using a fine-enough resolution, with the consequent loose in performance. Moreover, the number of charts generated with this planner scales exponentially with the dimension of the configuration space, hindering its application to complex problems. In [27], we preliminarily explored the possibility of combining the atlas construction with an RRT. An improved strategy with stronger theoretical background is presented here.

III. Representing Implicitly Defined Configurations Spaces

Let us consider a system described by an n-dimensional joint ambient space $\mathcal{A}$ and a k-dimensional configuration space $\mathcal{X} \subset \mathcal{A}$ implicitly defined by a set of equality constraints

$$\mathcal{X} = \{\mathbf{x} \in \mathcal{A} \mid \mathbf{F}(\mathbf{x}) = \mathbf{0}\} \tag{1}$$

with $\mathbf{F} : \mathcal{A} \to \mathbb{R}^{n-k}$, and $n > k > 0$. We adopt here the convention where the configuration space $\mathcal{X}$ is defined as the set of points fulfilling the constraints [6] (this is sometimes called constrained configuration space) that is embedded in the ambient space of the joint variables (called configuration space in some approaches). The constraints defining the configuration space may either come from the mechanics of the system itself (e.g., the assembly restrictions of a parallel robot) or from the task to be performed (e.g., a tray that must remain horizontal during a given task). Note that we only consider kinematic constraints, and other constraints, such as those involving dynamical aspects, are not taken into account. Moreover, we assume that $\mathcal{X}$ is a smooth manifold everywhere, without considering the presence of singularities.

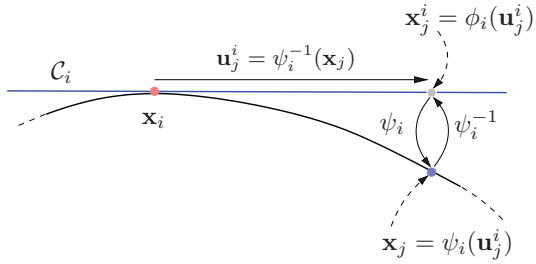

Fig. 3. In this paper, a chart $\mathcal{C}_i$ defines an exponential map $\mathbf{x}_j = \psi_i(\mathbf{u}_j^i)$ between the tangent space at $\mathbf{x}_i$ and the manifold, as well as a logarithmic map $\mathbf{u}_j^i = \psi_i^{-1}(\mathbf{x}_j)$ between the manifold and the tangent space. The mapping ϕ_i provides the ambient space coordinates of the tangent space parameters $\mathbf{u}_j^i$.

Let $\mathcal{O}$ be the obstacle region of the manifold, such that $\mathcal{F} = \mathcal{X} \setminus \mathcal{O}$ is the open set of the non-colliding configurations. Let also assume that $\mathbf{x}_s$ and $\mathbf{x}_g$ are the given start and goal configurations, respectively, both in $\mathcal{F}$. Then, the path planning problem consists of finding a collision free path linking the query configurations while staying on the manifold, i.e., to find a continuous function $\sigma : [0, 1] \to \mathcal{X}$ with $\sigma(0) = \mathbf{x}_s$, $\sigma(1) = \mathbf{x}_g$, and with $\sigma(\tau) \in \mathcal{F}$ for each $\tau \in [0, 1]$.

Representing an implicitly defined manifold with a global isometric parametrization is infeasible even for simple manifolds such as a sphere in 3-D. However, we can represent it as a set of local parametrizations, called charts, that form an atlas that covers the manifold.

Formally, a chart $\mathcal{C}_i$ locally parametrizes the k-dimensional manifold around a given point $\mathbf{x}_i$ with a bijective map, $\mathbf{x}_j = \psi_i(\mathbf{u}_j^i)$, between parameters $\mathbf{u}_j^i \in \mathbb{R}^k$ and points $\mathbf{x}_j$ on the manifold $\mathcal{X}$, with $\psi_i(\mathbf{0}) = \mathbf{x}_i$. The map from the parameter space to the manifold is known as the exponential map, and the inverse is referred as the logarithmic map (see Fig. 3). Following [38], an approximation of the exponential and logarithmic maps can be implemented using the k-dimensional space tangent at $\mathbf{x}_i$. An orthonormal basis for this tangent space is given by the $n \times k$ matrix $\mathbf{\Phi}_i$, satisfying

$$\begin{bmatrix} \mathbf{J}(\mathbf{x}_i) \\ \mathbf{\Phi}_i^\top \end{bmatrix} \mathbf{\Phi}_i = \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \tag{2}$$

where $\mathbf{J}(\mathbf{x}_i)$ is the Jacobian of $\mathbf{F}$ evaluated at $\mathbf{x}_i$, and $\mathbf{I}$ is the $k \times k$ identity matrix. Using this basis, the exponential map ψ_i is determined by first computing the mapping ϕ_i from parameters in the tangent space to coordinates in the joint ambient space

$$\mathbf{x}_j^i = \phi_i(\mathbf{u}_j^i) = \mathbf{x}_i + \mathbf{\Phi}_i \, \mathbf{u}_j^i \tag{3}$$

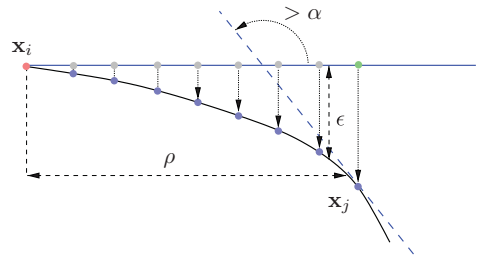

Fig. 4. Validity area of a chart $\mathcal{V}_i$ defined from (7)–(9). To ensure a correct mapping between the chart and the manifold, the distance and the curvature with respect to the tangent space must be bounded. Moreover, $\mathcal{V}_i$ is bounded to cover an area of maximum radius ρ to obtain a regular paving of the manifold. In the figure, the green point is not part of $\mathcal{V}_i$ since it does not fulfill these conditions.

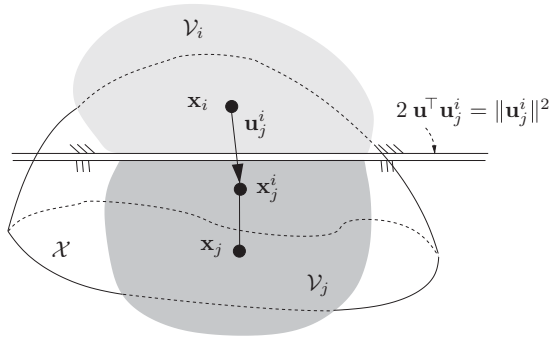

Fig. 5. When a new chart is defined at $\mathbf{x}_j$ the validity areas of the two charts $\mathcal{V}_i$ and $\mathcal{V}_j$ are coordinated to avoid overlaps.

and then orthogonally projecting this point on the manifold to obtain $\mathbf{x}_j$. This projection can be computed by solving the system

$$\begin{cases} \mathbf{F}(\mathbf{x}_j) = \mathbf{0} \\ \mathbf{\Phi}_i^\top\,(\mathbf{x}_j - \mathbf{x}_j^i) = \mathbf{0} \end{cases} \tag{4}$$

using a Newton procedure where $\mathbf{x}_j$ is initialized by an approximation of the solution ($\mathbf{x}_j^i$ if no better estimation is available) and is iteratively updated by the $\Delta\mathbf{x}_j$ increments fulfilling

$$\begin{bmatrix} \mathbf{J}(\mathbf{x}_j) \\ \mathbf{\Phi}_i^\top \end{bmatrix} \Delta\mathbf{x}_j = - \begin{bmatrix} \mathbf{F}(\mathbf{x}_j) \\ \mathbf{\Phi}_i^\top\,(\mathbf{x}_j - \mathbf{x}_j^i) \end{bmatrix} \tag{5}$$

until the error is negligible or for a maximum number of iterations [45]. The logarithmic mapping ψ_i^{-1} can be computed as the projection of a point on the tangent space

$$\mathbf{u}_j^i = \psi_i^{-1}(\mathbf{x}_j) = \mathbf{\Phi}_i^\top(\mathbf{x}_j - \mathbf{x}_i). \tag{6}$$

Note that this projection can also be applied to points not on the manifold to obtain their parameters in a given tangent space.

The area of the manifold properly parametrized by a given chart is limited. As the norm of $\mathbf{u}_j^i$ increases, the distance to the manifold and the difference in curvature typically increases as well, and the Newton process implementing ψ_i could even diverge. Thus, the validity area $\mathcal{V}_i$ of chart $\mathcal{C}_i$ is defined as the set of points $\mathbf{u}_j^i$ such that ψ_i can be safely computed and where

$$\|\mathbf{x}_j - \phi_i(\mathbf{u}_j^i)\| \leq \epsilon \tag{7}$$

$$\|\mathbf{\Phi}_i^\top\,\mathbf{\Phi}_j\| \geq \cos(\alpha) \tag{8}$$

$$\|\mathbf{u}_j^i\| \leq \rho \tag{9}$$

where $\mathbf{\Phi}_j$ is the basis of the tangent space at $\mathbf{x}_j = \psi_i(\mathbf{u}_j^i)$. The first condition limits the maximal distance error between the tangent space and the manifold, whereas the second condition ensures a bounded curvature in the part of the manifold covered by $\mathcal{C}_i$, as well as a smooth transition between charts (see Fig. 4). Combined, these two conditions bound the distortion introduced by the exponential map. Thus, they ensure that a uniform distribution of samples in $\mathcal{V}_i$ translates to an approximately uniform distribution of configurations on the manifold, and that a straight line in the tangent space gives a path on the manifold that has a bounded error with respect to the geodesic [46]. Finally, the third condition is introduced to obtain a more regular paving of the manifold.

The applicability area of a given chart, $\mathcal{V}_i$, is never empty, and it always includes the center of the chart, $\mathbf{x}_i$, but its shape can be arbitrarily intricate. Thus, $\mathcal{V}_i$ is conservatively approximated with a convex polytope, $\mathcal{P}_i \subset \mathcal{V}_i$. This polytope is represented with a set of linear inequalities, $\mathcal{L}_i$, defined in the tangent space associated with the chart [38]. This set is initially empty and is enlarged as the borders of $\mathcal{V}_i$ are discovered. If a parameter vector $\mathbf{u}_j^i$ is in $\mathcal{V}_i$ but $\mathbf{u}_j^i + \Delta\mathbf{u}$ is not in this set for a small $\Delta\mathbf{u}$, then a new chart $\mathcal{C}_j$ is added to the atlas at $\mathbf{x}_j = \psi_i(\mathbf{u}_j^i)$ and the following inequality:

$$2\,\mathbf{u}^\top \mathbf{u}_j^i \leq \|\mathbf{u}_j^i\|^2 \tag{10}$$

is included in $\mathcal{L}_i$, as shown in Fig. 5. Such inequality reduces the validity regions of the chart by cropping a half space defined in the tangent space associated with $\mathcal{C}_i$ given by the plane orthogonally bisecting vector $\mathbf{u}_j^i$. Actually, the new chart is coordinated not only with $\mathcal{C}_i$ but with all the charts $\mathcal{C}_l$ in the atlas such that $\mathbf{x}_j \in \mathcal{V}_l$. Thus, the set of inequalities for $\mathcal{C}_j$ is initialized with the inequalities defined from $\mathbf{u}_l^j = \phi_j^{-1}(\mathbf{x}_l)$, which is the projection of the centers of the nearby charts, $\mathcal{C}_l$, on the new chart $\mathcal{C}_j$.

Using the exponential and logarithmic maps, a full atlas of the manifold can be defined from where a roadmap whose edges are given by the neighboring relations between charts can be extracted (see Fig. 6). In principle, one could implement a resolution complete planner using such a roadmap to determine an optimal path between any two given configurations. However, this process would be computationally too demanding, especially in high dimensions [25]. Therefore, we propose to intertwine the atlas construction with the definition of an RRT to focus the exploration on the regions of the manifold that are relevant to the considered path planning problem.

IV. Rapid Exploration of Implicitly Defined Configuration Spaces

Instead of extending the atlas by following a predefined sequence of steps, we propose to use an RRT to devise random directions of expansions for the atlas. This trades off completeness

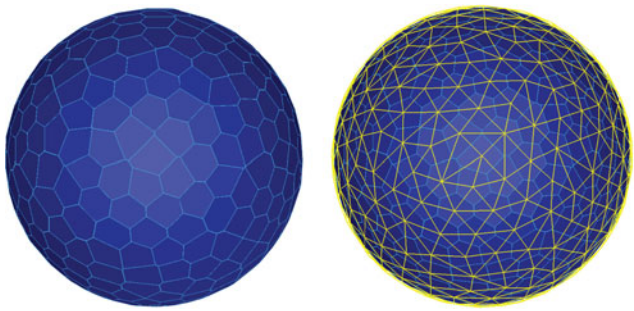

Fig. 6. (Left) Atlas of a sphere. Each polygonal patch corresponds to a given $\mathcal{P}_i$: a conservative approximation of the validity area for the associated chart. (Right) A roadmap can be extracted from the atlas where the nodes are the chart centers and where the edges are given by the neighborhood relations between charts. This roadmap could be used to devise collision free paths between any two given configurations.

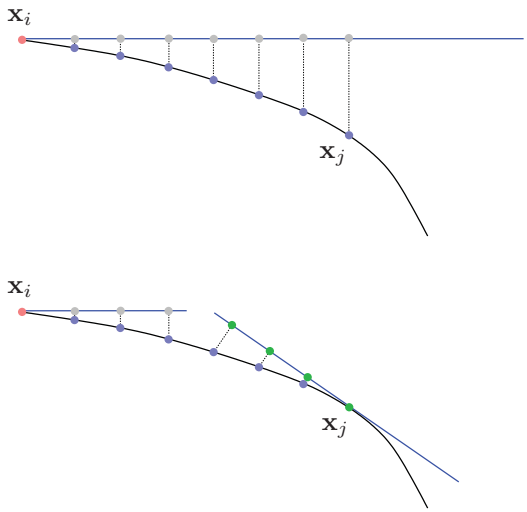

Fig. 7. (Top) RRT nodes parametrized by the chart at $\mathbf{x}_i$. (Bottom) When a new chart is created at $\mathbf{x}_j$, some of the nodes are now parametrized by the new chart. The nodes whose parametrization changes are represented in green.

in the exploration by efficiency: The RRT drives the growth of the atlas toward as-yet unexplored regions of the configuration space, delaying the refinement of already explored areas.

An RRT construction mechanism is based on three basic operations that are described next for the case of implicitly defined manifolds: sampling in the space to explore, detection of nearest neighbors, and extension of the tree toward a given point.

A. Random Sampling

To sample in $\mathcal{X}$, we take advantage of the partially built atlas. First, a chart $\mathcal{C}_r$ is randomly selected, and then, a point is randomly sampled within the ball of radius $\rho_s > \rho$ defined on the tangent space associated with this chart. If the random point is not in $\mathcal{P}_r$, which is the polytope that approximates the validity region of $\mathcal{C}_r$, it is discarded. This process is repeated until a valid sample $\mathbf{u}_r$ is eventually found. When the sampling is successful, $\mathbf{u}_r$ can be translated to coordinates in the ambient space using the corresponding ϕ_r mapping to get $\mathbf{x}_r$. Note that there is no need to map the parameters to $\mathcal{X}$ since the branch extension is actually performed in the tangent space, projecting on $\mathcal{X}$ when necessary.

For a given atlas, the probability of generating a valid random point in chart $\mathcal{C}_i$ is proportional to the volume of $\mathcal{P}_i$, and therefore, the sampling process selects points uniformly distributed within the region covered by the atlas. Since the atlas is defined incrementally, chart generated at the beginning of the process can be selected more times. These charts, however, typically have a larger rejection ratio than charts at the frontier of exploration, which have sampling areas much larger than their actual validity areas. Therefore, in practice, the exploration is pushed toward unexplored regions of the configuration space. Actually, the ratio $1 - (\rho/\rho_s)^k$ gives an upper bound of the proportion of sampled points that are outside the validity region of a given frontier chart and, thus, the ratio of points that will trigger the creation of new charts. Therefore, by changing ρ_s, we can directly tune the balance between exploration and refinement, which is an important feature of the RRT-based algorithms [47].

B. Nearest Neighbor

The second basic operation to build an RRT is the identification of the node $\mathbf{x}_n$ in the tree closer to the random node $\mathbf{x}_r$. This should be done using the intrinsic metric of the configuration space. In a parametrizable space, this metric is simple, but on a manifold, the geodesic distance should be used. The implementation of an efficient nearest neighbor procedure for implicitly defined manifolds is difficult, and it has only been addressed recently in an approximated way, relying on a representation of the manifold that is very similar to a partial atlas [48]. However, the approximation is only adequate for dense set of samples, which is not the case when building an RRT. An alternative solution, which will be adopted here, is to resort to the ambient space nearest neighbor as an approximation of the geodesic nearest neighbor, despite this may sometimes lead to inadequate tree extensions.

C. Tree Extension

When the nearest node $\mathbf{x}_n$ has been selected, the RRT proceeds to grow a branch toward the random sample $\mathbf{x}_r$. Note that $\mathbf{x}_n$ and $\mathbf{x}_r$ can be in different charts and, thus, given in different local parametrizations. Therefore, the direction of extension is obtained by projecting $\mathbf{x}_r$ on the chart $\mathcal{C}_c$ parametrizing $\mathbf{x}_n$ and eventually translating the resulting point to ensure that it is at least at distance $\|\mathbf{x}_n - \mathbf{x}_r\|$ from $\mathbf{x}_n$. Once the nearest node and the random point are represented in the same parameter space, a new branch can be generated by linear interpolation between the parameters of these two points. After each interpolation step, the parameters are projected on the configuration space, and if the new configuration is collision free, a new node is added to the tree.

During the tree extension, the new branch might leave $\mathcal{P}_c$ or $\mathcal{V}_c$. If it leaves $\mathcal{P}_c$, then the branch extension must continue on a neighboring chart. If the branch reaches a yet unknown region that borders $\mathcal{V}_c$, a new chart has to be created at the last point

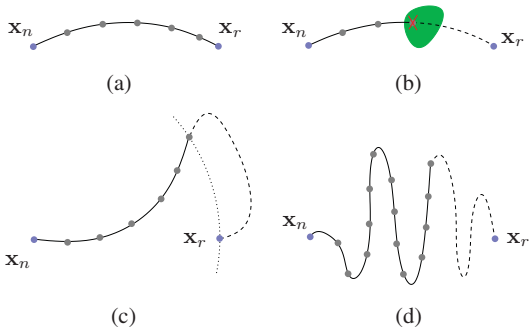

Fig. 8. Four conditions to stop the AtlasRRT branch extension process. (a) The branch reaches the random configuration. (b) A collision is detected. (c) The distance from the end branch to its origin is larger than the distance to the random point. (d) The distance traveled is larger than λ times the distance to the random point.

within $\mathcal{V}_c$. After creating a chart, the nodes in existing charts not fulfilling the inequalities introduced by the new chart move to the area of validity of the new chart (see Fig. 7). Whenever the branch reaches a new or a neighboring chart, the direction of expansion is recomputed, projecting $\mathbf{x}_r$ on this chart and ensuring that it is far enough from the last node added to the RRT. This projection strategy is different from the one in [27], where branch extensions were prematurely stopped in some occasions.

The most expensive step in the branch generation is the evaluation of (8) since it requires to build the tangent space at each considered point. This cost can be alleviated evaluating the curvature along the direction of expansion of the branch. This operation only requires the distance between the previous and the new nodes in parameter and in ambient spaces, which are readily available. Such simplified curvature evaluation is not present in [27] and yields a large execution speedup. Note, however, that the full curvature test is used when a new chart is added to the atlas. Thus, neighboring relations between charts always fulfill (8).

When exploring an Euclidean space with an RRT, a branch extension stops either when the random sample is reached or when a collision is detected. When exploring a non-Euclidean space, a branch can be arbitrarily long due to the manifold curvature, even when using the two conditions above. To avoid this issue, the branch extension is also stopped if

$$\|\mathbf{x}_j - \mathbf{x}_r\| > \|\mathbf{x}_n - \mathbf{x}_r\| \tag{11}$$

or if

$$d > \lambda \|\mathbf{x}_n - \mathbf{x}_r\| \tag{12}$$

where $\mathbf{x}_j$ is the last node added to the branch, $\mathbf{x}_r$ is the random sample, $\mathbf{x}_n$ is the nearest node in the tree, and d is the length of the branch, as illustrated in Fig. 8. The first condition prevents the branch to escape the ball centered at $\mathbf{x}_n$ with radius $\|\mathbf{x}_n - \mathbf{x}_r\|$. The second condition avoids the length of the branch to be larger than a scale factor of the distance between $\mathbf{x}_r$ and $\mathbf{x}_n$. These two branch termination conditions were not present in the preliminary version of the algorithm [27] and largely help to improve the quality of the RRT.

Algorithm 1: The AtlasRRT algorithm.

AtlasRRT$(\mathbf{x}_s, \mathbf{x}_g, \mathbf{F})$
input : The query configurations, $\mathbf{x}_s$ and $\mathbf{x}_g$, a set of constraints, $\mathbf{F}$.
output: A path connecting $\mathbf{x}_s$ and $\mathbf{x}_g$.
1 $T_s \leftarrow$ INITRRT$(\mathbf{x}_s)$
2 $T_g \leftarrow$ INITRRT$(\mathbf{x}_g)$
3 $A \leftarrow$ INITATLAS$(\mathbf{F}, \mathbf{x}_s, \mathbf{x}_g)$
4 DONE $\leftarrow$ FALSE
5 **while not** DONE **do**
6 $\quad \mathbf{x}_r \leftarrow$ SAMPLEONATLAS(A, T_s)
7 $\quad \mathbf{x}_n \leftarrow$ NEARESTNODE$(T_s, \mathbf{x}_r)$
8 $\quad \mathbf{x}_l \leftarrow$ EXTENDTREE$(A, T_s, \mathbf{x}_n, \mathbf{x}_r,$ TRUE$)$
9 $\quad \mathbf{x}'_n \leftarrow$ NEARESTNODE$(T_g, \mathbf{x}_l)$
10 $\quad \mathbf{x}'_l \leftarrow$ EXTENDTREE$(A, T_g, \mathbf{x}'_n, \mathbf{x}_l,$ FALSE$)$
11 $\quad$ **if** $\|\mathbf{x}_l - \mathbf{x}'_l\| < \delta$ **then**
12 $\quad\quad$ DONE $\leftarrow$ TRUE
13 $\quad$ **else**
14 $\quad\quad$ SWAP(T_s, T_g)
15 RETURN(PATH$(T_s, \mathbf{x}_l, T_g, \mathbf{x}'_l)$)

Algorithm 2: Sampling on an atlas.

SampleOnAtlas(A, T)
input : The atlas, A, the tree currently extended, T.
output: A sample on the atlas.
1 **repeat**
2 $\quad r \leftarrow$ RANDOMCHARTINDEX(A, T)
3 $\quad \mathbf{u}_r \leftarrow$ RANDOMONBALL(ρ_s)
4 **until** $\mathbf{u}_r \in \mathcal{P}_r$
5 **forall** $l_k \in \mathcal{L}_r$ **do**
6 $\quad$ **if** DISTANCE$(\mathbf{u}_r, l_k) < \|\mathbf{u}^r_k\|/20$ **then**
7 $\quad\quad j \leftarrow$ GENERATINGCHART(l_k)
8 $\quad\quad \mathcal{P}_j \leftarrow$ ENLARGEPOLYTOPE$(\mathcal{P}_j, \psi_j^{-1}(\psi_r(\mathbf{u}_r)))$
9 RETURN$(\phi_r(\mathbf{u}_r))$

V. ATLASRRT ALGORITHM

Algorithm 1 gives the pseudocode for the AtlasRRT planner implementing the path planning approach introduced in the previous section. The algorithm takes $\mathbf{x}_s$ and $\mathbf{x}_g$ as start and goal configurations, respectively, and tries to connect them with a path on the manifold implicitly defined by a given set of constraints $\mathbf{F}$. The algorithm implements a bidirectional RRT with one tree rooted at $\mathbf{x}_s$ and another at $\mathbf{x}_g$ (lines 1 and 2). An atlas is also initialized with one chart centered at each one of these points (line 3). Next, the algorithm iterates, trying to connect the two trees (lines 5–14). First, a configuration is sampled from the atlas (line 6), and an RRT branch is extended from the nearest node already in the tree (line 7) toward the random sample (line 8). Then, an extension attempts to reach the last node of the new branch (line 10) from the nearest node in the other tree (line 9). If this second extension achieves its objective (line 11), the trees are connected, and their nodes are used to reconstruct a path between $\mathbf{x}_s$ and $\mathbf{x}_g$ (line 15). Otherwise, the two trees are swapped (line 14), and the extension process is repeated.

In the sampling process, we use the atlas, as described in Algorithm 2. A chart is selected at random with uniform distribution among the set of charts reached by the tree to extend (line 2). Then, a point $\mathbf{u}_r$ is sampled within the ball of radius ρ_s bounding the sampling region of this chart (line 3). The process

Algorithm 3: AtlasRRT tree extension.

ExtendTree$(A, T, \mathbf{x}_n, \mathbf{x}_r, \text{EXPLORE})$
input : An atlas, A, a tree, T, the nearest node in the tree, $\mathbf{x}_n$, the random sample, $\mathbf{x}_r$, and EXPLORE, a flag that is TRUE when extending a tree and FALSE when trying to connect the two trees.
output: The last node added to the tree or $\mathbf{x}_n$ if no extension was performed.

1 $c \leftarrow \text{CHARTINDEX}(\mathbf{x}_n)$
2 $\mathbf{u}_n \leftarrow \psi_c^{-1}(\mathbf{x}_n)$
3 $\mathbf{u}_r \leftarrow \psi_c^{-1}(\mathbf{x}_r)$
4 $d_0 \leftarrow \|\mathbf{x}_n - \mathbf{x}_r\|$
5 $d \leftarrow 0$
6 $\mathbf{x}_0 \leftarrow \mathbf{x}_n$
7 **if** EXPLORE **then**
8 $\quad \mathbf{u}_r \leftarrow \mathbf{u}_r(d_0 - d)/\|\mathbf{u}_r\|$
9 $\quad \mathbf{x}_r \leftarrow \phi_c(\mathbf{u}_r)$
10 STOP $\leftarrow$ FALSE
11 CHARTCREATED $\leftarrow$ FALSE
12 **while not** STOP **and** $\|\mathbf{u}_r - \mathbf{u}_n\| > \delta$ **do**
13 $\quad \mathbf{u}_j \leftarrow (\mathbf{u}_r - \mathbf{u}_n)\,\delta/\|\mathbf{u}_r - \mathbf{u}_n\|$
14 $\quad \mathbf{x}_j \leftarrow \psi_c(\mathbf{u}_j)$
15 $\quad d_s \leftarrow \|\mathbf{x}_n - \mathbf{x}_j\|$
16 $\quad$ NEW $\leftarrow$ FALSE
17 $\quad$ **if** COLLISION$(\mathbf{x}_j)$ **then**
18 $\quad\quad$ STOP $\leftarrow$ TRUE
19 $\quad$ **else**
20 $\quad\quad$ **if** $\|\mathbf{x}_j - \phi_c(\mathbf{u}_j)\| > \epsilon$ **or** $\delta/d_s < \cos(\alpha)$ **or** $\|\mathbf{u}_j\| > \rho$ **then**
21 $\quad\quad\quad \mathcal{C} \leftarrow \text{NEWCHART}(\mathbf{x}_n)$
22 $\quad\quad\quad c \leftarrow \text{ADDTOATLAS}(A, \mathcal{C})$
23 $\quad\quad\quad$ NEW $\leftarrow$ TRUE
24 $\quad\quad\quad$ CHARTCREATED $\leftarrow$ TRUE
25 $\quad\quad$ **else**
26 $\quad\quad\quad$ **if** $\mathbf{u}_j \notin \mathcal{P}_c$ **then**
27 $\quad\quad\quad\quad c \leftarrow \text{NEIGHBORCHART}(\mathcal{C}_c, \mathbf{u}_j)$
28 $\quad\quad\quad\quad$ **if** CHARTCREATED **or** (**not** EXPLORE **and** INTREE(T, c)) **then**
29 $\quad\quad\quad\quad\quad$ STOP $\leftarrow$ TRUE
30 $\quad\quad\quad\quad$ **else**
31 $\quad\quad\quad\quad\quad$ NEW $\leftarrow$ TRUE
32 $\quad$ **if not** STOP **then**
33 $\quad\quad$ **if** NEW **then**
34 $\quad\quad\quad \mathbf{u}_j \leftarrow \psi_c^{-1}(\mathbf{x}_j)$
35 $\quad\quad\quad \mathbf{u}_r \leftarrow \psi_c^{-1}(\mathbf{x}_r)$
36 $\quad\quad\quad$ **if** EXPLORE **then**
37 $\quad\quad\quad\quad \mathbf{u}_r \leftarrow \mathbf{u}_r(d_0 - d)/\|\mathbf{u}_r\|$
38 $\quad\quad\quad\quad \mathbf{x}_r \leftarrow \phi_c(\mathbf{u}_r)$
39 $\quad\quad$ $T \leftarrow \text{ADDNODE}(T, A, \mathbf{x}_j, \mathbf{x}_n)$
40 $\quad\quad$ $d \leftarrow d + d_s$
41 $\quad\quad$ **if** $\|\mathbf{x}_0 - \mathbf{x}_j\| > d_0$ **or** $d > \lambda\, d_0$ **then**
42 $\quad\quad\quad$ STOP $\leftarrow$ TRUE
43 $\quad\quad$ $\mathbf{u}_n \leftarrow \mathbf{u}_j$
44 $\quad\quad$ $\mathbf{x}_n \leftarrow \mathbf{x}_j$
45 RETURN$(\mathbf{x}_n)$

is repeated until a point in the polytope $\mathcal{P}_r$ is obtained (line 4). Sampling only in the charts reached by the tree being extended produces a more regular exploration than when using all the charts of the atlas [27]. For the samples close to the borders defining $\mathcal{P}_r$ (line 6), we check whether or not they are also covered by the corresponding neighboring chart (lines 7 and 8). In this test, $\mathbf{u}_k^r$ is the point used to generate the linear inequality l_k, and $\|\mathbf{u}_k^r\|/20$ is a tenth of its distance to the chart center. Such a check ensures a small overlap between neighboring charts, and thus, all the configurations are effectively covered by the atlas and can be selected as random samples. The parameters of the random sample in the neighboring chart are obtained using the exponential map for the chart selected at random ψ_r and then the logarithmic map for the neighboring chart ψ_j^{-1}. Finally, the random sample returned by the algorithm is formed by the ambient space coordinates for the selected point computed using the mapping ϕ_r for the selected chart $\mathcal{C}_r$ (line 9).

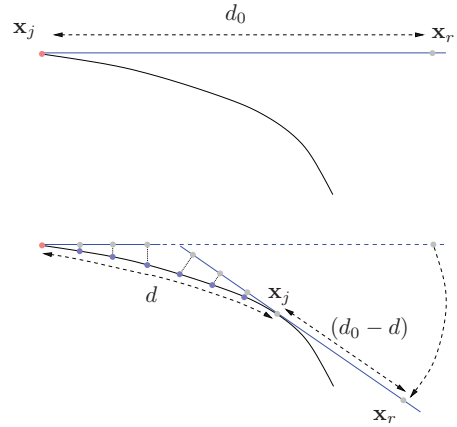

Fig. 9. (Top) Samples $\mathbf{x}_r$ are generated in the tangent space associated with a given chart. (Bottom) When the RRT branch that attempts to reach $\mathbf{x}_r$ generates a chart, the random sample is projected on this new chart, and it is displaced to ensure that it is at an adequate distance from the sample from where to continue the branch extension.

The addition of a branch to a tree T is done by following the steps detailed in Algorithm 3. Note that there are two types of tree extensions. The first one attempts to reach a random sample defined on the tangent space associated with a given chart. The second one tries to reach a node in the other tree and, thus, a point on the configuration manifold. In the first case, the parameter EXPLORE is set to TRUE and this results in some differences in how the random sample is managed during the tree expansion. In any case, the procedure operates in the chart $\mathcal{C}_c$ including the node to be extended, which is initially the chart parametrizing the nearest node $\mathbf{x}_n$ (line 1). The sample from where to start the new branch and the sample to reach in the tree extension are both projected on $\mathcal{C}_c$ (lines 2 and 3), and the distance d_0 between these two samples is computed (line 4). Moreover, the length of the new branch, i.e., d, is initialized to 0 (line 5), and the initial branch point is stored in $\mathbf{x}_0$ (line 6). The values for d_0, d, and $\mathbf{x}_0$ are necessary to eventually stop the branch extension. If EXPLORE is TRUE, i.e., if the random sample is not on the configuration manifold, the parameters for the random sample are displaced to a distance d_0 from $\mathbf{u}_n$ (lines 8 and 9) to ensure a minimum branch length (see Fig. 9). Then, the branch is extended, while none of the possible stop conditions is detected and the random sample is not reached (lines 12 to 44). At each iteration, a node is added to the tree. To define the node to add, a small step of size δ is taken in the parameter space of $\mathcal{C}_c$ from the current node $\mathbf{u}_n$

toward $\mathbf{u}_r$ (line 13). The resulting parameters $\mathbf{u}_j$ are projected on the manifold to obtain the configuration $\mathbf{x}_j$ (line 14). If this new configuration is in collision, the branch extension is stopped (line 18). Otherwise, the algorithm checks if the new configuration triggers the generation of a new chart (line 20). If so, the chart is generated at the previous configuration that is still in $\mathcal{V}_c$ and it is added to the atlas (line 22). Configurations that do not generate a new chart might be out of $\mathcal{P}_c$, i.e., they might be in the area of the manifold parametrized by a neighboring chart (line 26). If a branch that triggered the creation of new charts enters a preexisting chart, or if the branch enters a chart already reached by the tree under expansion when aiming to reach the other tree, the branch extension is stopped. These situations are typically produced when a branch is extended too far or when the nearest neighbor is not actually the closest node to the target sample. By stopping those branch extensions, we avoid an excessive refinement of regions already covered by the tree, which would hinder the efficient exploration of the configuration space manifold. Whenever the branch extension is not stopped, the neighboring chart from where to continue the branch growth is identified (line 27), and the position of the goal sample is recomputed in this new chart (lines 34 to 38). Then, the new node is added to the tree (line 39), and the branch length is updated (line 40). This length is used to determine if the new sample is too far away from the initial branch node or if the branch is too long (line 41). In either of these two cases, the branch extension is stopped (line 42). Finally, the new node is set as the point from where to continue the branch extension (lines 43 and 44).

A. Computational Complexity

Besides the cost of collision detection, the most expensive steps in the algorithm are the search for nearest nodes in the RRTs (lines 7 and 9 of Algorithm 1), the search for the neighboring charts when adding a chart to the atlas (line 22 of Algorithm 3), the creation of new charts (line 21 in Algorithm 3), and the computation of the mapping ψ_c (line 14 of Algorithm 3). The two search operations can be implemented using hierarchical structures reducing their cost to be logarithmic in the number of nodes of the corresponding RRT and the number of charts in the atlas, respectively. The cost of generating a new chart is $O(n^3)$, since the computation of the tangent space basis relies on a QR decomposition. Note, however, that the generation of new charts is seldom necessary, and thus, this cost is amortized over several iterations. Finally, the cost of computing the mapping ψ_c also scales with $O(n^3)$ since it is implemented as a Newton process with a bounded number of iterations, where at each iteration, a QR decomposition is used. This Newton process, though, typically converges in very few iterations since it can be initialized at $\mathbf{x}_n$, which is very close to the solution point.

B. Parameters

The algorithm basically uses six parameters: ϵ, the maximum error with respect to a chart used in (7); α, the maximum angle between neighboring charts used in (8); ρ, the maximum radius of the validity area of a chart used in (9); ρ_s, the sampling radius; δ, the size branch extension steps; and λ, the length factor used to eventually stop the branch extensions. Note that both δ and a bound on the space to sample (the role ρ_s in our case) are also used in RRTs in Euclidean spaces.

The parameters ϵ, α, and ρ control the size of the validity area of the charts and, thus, the number of charts in the atlas. Although ϵ and ρ can safely span over a wide range, α should be limited to avoid a large distortion between the tangent space and the manifold [46]. The ratio between ρ and ρ_s sets the balance between refinement in current charts and exploration since the larger the ρ_s with respect to ρ, the stronger the bias toward unexplored regions. Thus, the ratio $(\rho/\rho_s)^k$ should decrease as the dimensionality of the configuration manifold increases, to emulate the exploration bias of RRT in Euclidean spaces. Since neither collision nor curvature tests are done between two consecutive points in an RRT branch, a small δ should be used to avoid undetected collision or sharp changes in the manifold curvature. Finally, λ must be larger than 1. However, the performance of the algorithm is not very sensitive to its actual value since the branch termination condition using this parameter is seldom active.

C. Probabilistic Completeness

To show the probabilistically completeness of the proposed approach, recall that we assumed the configuration space to be a k-dimensional smooth manifold $\mathcal{X}$ with a Jacobian that is full rank everywhere. Under these conditions, $\mathcal{X}$ and its Jacobians are continuous in an open k-dimensional ball around any given point on $\mathcal{X}$, and thus, $\mathcal{X}$ is singularity free and of constant dimensionality. Thus, a path between any two given configurations never includes points on any particular low-dimensional subset with zero measure. Additionally, by the implicit function theorem, the above conditions also guarantee that the validity area $\mathcal{V}_i$ of a given chart centered at $\mathbf{x}_i$ includes a non-null measure ball around this point, i.e., $\mathcal{V}_i$ is not null in all directions. Moreover, since transitions between different connected components of the manifold are not possible, we assume that the start and goal configurations are in $\mathcal{F}$, which is the connected component of the collision free part of $\mathcal{X}$ including both configurations. For simplicity, we consider a version of AtlasRRT where a single tree is generated (the extension of the argument to bidirectional trees is straightforward), and we assume that for each RRT node, the curvature test in (8) is performed, instead of the simplified version described in Section IV-C. This way, points out of the validity area are detected, independently of the direction from which they are approached. Finally, we also assume that the measure of $\mathcal{F}$, i.e., $\mu(\mathcal{F})$, is finite either because it has a closed topology or because it is cropped to some domain $\mathcal{D}$. Note that this is typically the case in robotics, where $\mathcal{F}$ is bounded by the joint ranges and the presence of obstacles.

As proved in [34], under the above assumptions, any RRT-like method able to densely sample $\mathcal{F}$ is probabilistically complete, even if the samples are not uniformly distributed. In the AtlasRRT, the sampling is obtained using the atlas. Thus, if we prove that the atlas fully covers $\mathcal{F}$, i.e., that any point on $\mathcal{F}$ is mapped from a point on the atlas, the probabilistic completeness

will be guaranteed. To show that an atlas defined using the method presented in Section III fully parametrizes $\mathcal{F}$, we will adapt the argument given in [38]. This argument basically shows that each new chart increases the coverage of $\mathcal{F}$ until it is fully covered.

Let $\mathcal{F}_A$ be the part of $\mathcal{F}$ already parametrized by the current atlas A, i.e., the projection on $\mathcal{F}$ of the validity areas of all the charts in A where the validity area of chart $\mathcal{C}_i$ is

$$\mathcal{V}_i = \mathcal{B}^k(\rho) \cap \mathcal{P}_i \tag{13}$$

where $\mathcal{B}^k(\rho)$ is the k-dimensional ball of radius ρ, and $\mathcal{P}_i$ is the polytope associated with the chart. This area is projected on a patch of $\mathcal{F}$ using (4). Moreover, the sampling area of the chart is

$$\mathcal{S}_i = \mathcal{B}^k(\rho_s) \cap \mathcal{P}_i \tag{14}$$

with $\rho_s > \rho$. For any open chart, i.e., a chart where $\mathcal{S}_i \not\subseteq \mathcal{V}_i$, a sample out of $\mathcal{F}_A$, $\mathbf{x}_r$, will be generated with probability $\mu(\mathcal{S}_i \setminus \mathcal{V}_i) > 0$. The RRT branch toward $\mathbf{x}_r$ necessarily crosses the border of $\mathcal{F}_A$ connecting an RRT node $\mathbf{x}_n \in \mathcal{F}_A$ to $\mathbf{x}_r$. When this happens, a point on the border of $\mathcal{F}_A$ is determined using a dichotomic search, and a new chart is defined on it. Since the validity areas of the charts are not null in all directions and the chart is defined at the border of $\mathcal{F}_A$, this new chart will parametrize an area not previously in $\mathcal{F}_A$, and thus, $\mu(\mathcal{F}_A)$ is increased. Note that $\mu(\mathcal{F}_A)$ is increased even if this chart is close to the border of $\mathcal{F}$, and thus, it also parametrizes regions in collision, i.e., not in $\mathcal{F}$. In practice, new charts are defined on $\mathbf{x}_n$, and the dichotomic search is only performed if $\mathbf{x}_n$ is already the center of a chart. This procedure is not detailed in Algorithm 3 for the sake of clarity. The creation of a new chart removes part of the sampling area for the chart previously parametrizing $\mathbf{x}_n$, and thus, all charts will eventually become closed, i.e., charts where $\mathcal{S}_i = \mathcal{V}_i$.

Closed charts might include areas where their parametrizations are not valid, i.e., where (7)–(9) do not hold. Under the taken assumptions, these areas have non-null measure, and thus, there is a non-null probability of generating a sample on them. When such sample is generated, the same reasoning as that used with open charts leads to an increment of $\mu(\mathcal{F}_A)$.

Special care must be taken to ensure that the addition of a new chart to A does not decrease $\mu(\mathcal{F}_A)$. According to [38] in very curved regions, unmapped gaps between charts might appear. The procedure at the end of the sampling process (lines 5 to 8 in Algorithm 2) allows removing these gaps, if any, ensuring that no part of $\mathcal{F}_A$ is lost. Summarizing, since $\mu(\mathcal{F})$ is finite and $\mathcal{F}_A$ monotonically increases with every new chart, the atlas will eventually cover $\mathcal{F}$. The extension of the atlas cannot be stalled before covering $\mathcal{F}$ since this would imply to have a chart in A including a region not in $\mathcal{F}_A$ with null probability of being sampled, which contradicts the taken hypothesis. When the atlas is fully defined, $\mathcal{F}$ can be densely sampled, and this guarantees the probabilistic completeness of the algorithm. In practice, however, paths are typically found well before the full atlas is defined.

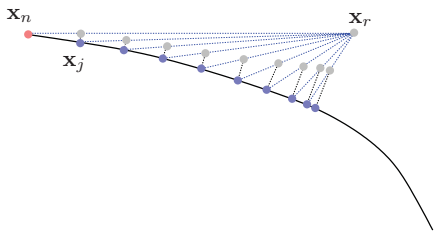

Fig. 10. Branch generation procedure used in the CB-RRT algorithm. The next sample in the RRT branch $\mathbf{x}_j$ is determined by first interpolating a point in the ambient space between the last point in the branch $\mathbf{x}_n$ and the random sample $\mathbf{x}_r$ and then projecting it on the manifold using a Jacobian pseudo-inverse procedure.

VI. Experiments

We implemented the AtlasRRT planner described through Sections IV and V in C. The implementation uses SOLID [49] as collision detector, the GNU Scientific Library [50] for the linear algebra operations, and the kd-tree described in [51] for the nearest neighbor queries. The resulting software package, as well as the examples used in the experiments reported below, can be downloaded from [52]. The preliminary version of AtlasRRT [27] relied on a formulation of the mechanisms with redundant variables that yields a system of simple equations only containing linear, bilinear, and quadratic monomials [53]. In this paper, we propose to use a standard Denavit–Hartenberg formulation for the joints [54]. Due to the high nonlinearity of this formulation, the manifold tends to be more convoluted, which may hinder the projection of samples from the tangent to the configuration space and the trace of this space using continuation-based methods. The experimental results show that this effect is compensated by the speedup obtained due to the reduction in the dimensionality of the ambient space.

The experiments presented herein aim to verify that the proposed planner is able to explore the configuration manifolds arising in realistic problems, even when the solution paths must traverse narrow corridors and avoid local minima. Moreover, we aim to test the sensitivity of the approach to the variations of the different parameters and its scalability with respect to the dimensionality of the configuration space. Finally, we aim to determine the efficiency of the approach, as compared with the most relevant existing methods.

For the sake of comparison, we use the HC-planner [25] and CB-RRT, a planner that includes the mechanisms for path planning on manifolds of the CBiRRT2 planner introduced in [24]. The task-space aspects of CBiRRT2, however, are not included in CB-RRT since AtlasRRT does not use them. The HC-planner is a resolution complete planner on manifolds based on a greedy best first search method within a graph implicitly defined on the atlas that is built along with the search. The CB-RRT planner shares the bidirectional search strategy with AtlasRRT, but the random samples $\mathbf{x}_r$ are generated in the joint ambient space. Then, the algorithm interpolates between the nearest point already in the RRT, $\mathbf{x}_n$, and $\mathbf{x}_r$ to get a point $\mathbf{x}_j$ that is projected on the manifold using the Jacobian pseudo-inverse procedure [55], as illustrated in Fig. 10. The process of interpolation and projection is repeated to grow branches

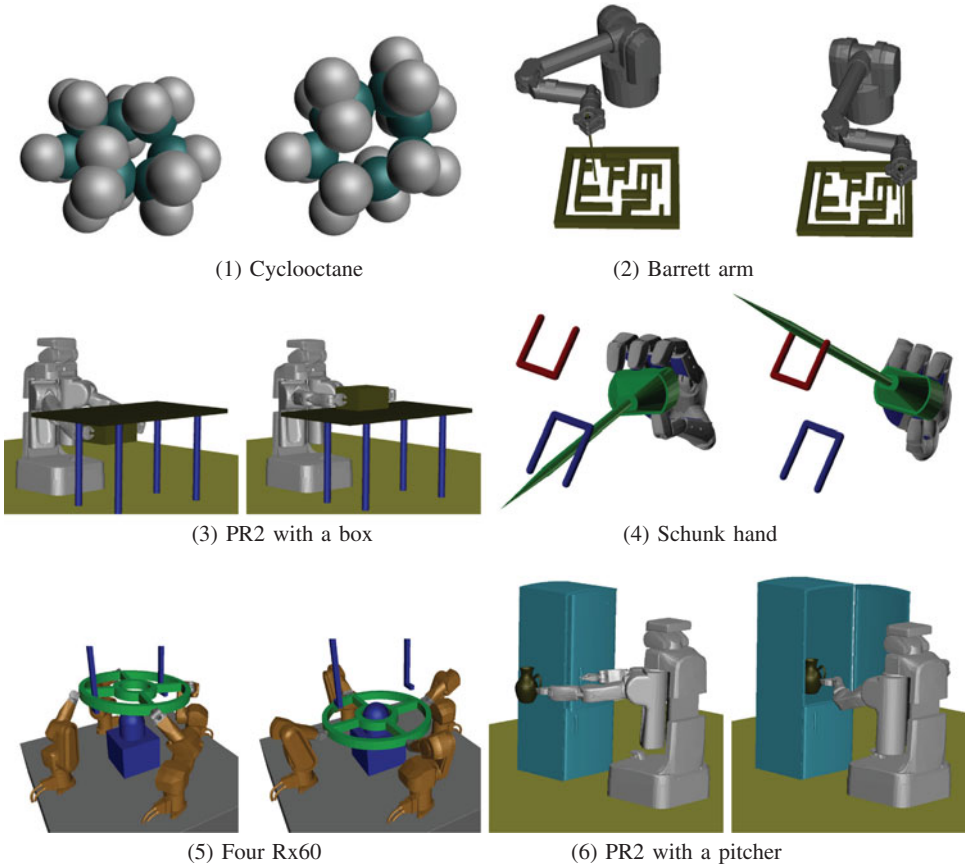

Fig. 11. Six benchmarks are used in this paper. For each benchmark, the left and right pictures correspond to start and goal configurations, respectively. (1) A cyclooctane molecule. (2) The Barrett arm solving a maze problem. (3) The PR2 robot moving a box. (4) The Schunk anthropomorphic grasping a needle. (5) Four Säubli Rx60 arms collaborating to move a circular piece in an industrial environment. (6) The PR2 robot putting a pitcher into a refrigerator.

on the configuration manifold. Note, however, that this process tends to produce short branch extensions when the interpolation direction approaches the orthogonal to the manifold [35] . For a fair comparison, both the HC-planner and the CB-RRT are applied on the same formulation used by AtlasRRT.

Fig. 11 shows the start and goal configurations for the six benchmarks used in this paper, sorted by increasing dimensionality of its configuration space. These benchmarks have been inspired by examples previously used in path planning under kinematic constraints [10], [24]. The first one is the cyclooctane: a molecule whose kinematics is an eight-revolute loop. Here, we have to find a path between two conformations that avoids collisions between carbon and hydrogen atoms (depicted in the figure in cyan and white, respectively). This is a very constrained problem where the solution path requires to pass through three narrow passages. The second benchmark involves the Barrett arm solving a maze problem. The stick moved by the arm has to stay in contact with the maze plane and perpendicular to it, without rotating about its axis. In the third problem, a PR2 robot with fixed base must move a box located under a table onto this table without rotating it. The gap between the robot and the table is narrow, considering the size of the box, which increases the complexity of the problem. In the fourth problem, the Schunk anthropomorphic hand [18] grasps a needle which must be moved avoiding a couple of U-shaped obstacles that introduce local minima in the planning. In the fifth problem, four Stäubli Rx60 industrial arms must perform complex coordinated motions to extract a large piece from two hooks in the ceiling and insert it into a peg. Finally, in the last problem, the PR2 robot with a fixed base has to put a pitcher into a refrigerator, maintaining it vertical with its left arm and, at the same time, opening the door with its right arm.

Table I shows the performance comparison between the HC-planner, the CB-RRT, and the AtlasRRT, averaged over 100 runs and for a maximal execution time of 600 s on a Intel Core i7 at 2.93 GHz running Mac OS X with parameters set to $\epsilon = 0.1$, $\alpha = 0.45$, $\rho = 1$, $\delta = 0.05$, and $\lambda = 2$ for all the experiments. Two different sampling radius ρ_s are used taking into account that a small ρ_s is more adequate in low-dimensional spaces, or when the problem is very constrained by obstacles, and a large ρ_s is better suited when a large configuration space must be explored. Thus, we set $\rho_s = 2$ for experiments (1), (2), and (4) and $\rho_s = 7$ for experiments (3), (5), and (6). For each benchmark, the table gives the dimensionality of both the configuration space k and the ambient space n. It also provides for each planner the percentage of successful runs (in the Succ. column), and for these runs the number of charts and RRT nodes required (given in the Charts and Nodes columns, respectively), as well as the execution times in seconds (in the Time column).

TABLE 1
DIMENSION OF THE CONFIGURATION AND AMBIENT SPACES, SUCCESS RATES, NUMBER OF NODES/CHARTS, AND EXECUTION TIMES IN SECONDS FOR THE THREE METHODS COMPARED IN THIS PAPER

			HC-Planner			CB-RRT			AtlasRRT			
Benchmark	k	n	Succ.	Charts	Time	Succ.	Nodes	Time	Succ.	Charts	Nodes	Time
(1) Cyclooctane	2	8	0.28	420	19.67	1.0	43333	13.88	1.0	139	5812	1.94
(2) Barrett Arm	3	9	0.20	336	2.59	1.0	49248	30.23	1.0	80	5316	2.43
(3) PR2 box	4	16	0.97	671	135.36	1.0	13657	15.00	1.0	58	1393	0.88
(4) Robot Hand	5	23	0.25	1657	204.13	0.5	55098	256.22	1.0	50	1503	8.23
(5) Four Rx60	6	24	0.01	1355	170.16	1.0	25013	62.87	1.0	566	10838	15.54
(6) PR2 pitcher	8	16	0.01	284	493.99	1.0	8237	12.31	1.0	111	2683	2.37

A first remarkable result is that in problems such as the Barrett arm problem, where the configuration space is of low dimensionality (2 and 3), the HC-Planner can be even faster than the CB-RRT because the search relies on an atlas that captures the structure of the configuration manifold. However, the HC-Planner is only successful in few of the experiments because it tends to be blocked in narrow passages, whereas the CB-RRT can successfully negotiate them. AtlasRRT combines the advantage of the two methods since it takes advantage of the atlas while avoiding being blocked by obstacles. In problems of moderate dimensionality (4 and 5), the performance of HC-Planner starts to decrease since its computational cost is exponential with the dimension of the space, and CB-RRT starts to be more efficient. In these situations, AtlasRRT performs better than the two other methods. A significant case is the Robot hand, where both HC-Planner and CB-RRT have many failures and they are more than 25 times slower than AtlasRRT when they succeed. In higher dimension (6 and 8), HC-Planner is almost unable to find a solution path, whereas both the CB-RRT and the AtlasRRT are successful in all cases, with AtlasRRT being more efficient.

Since CB-RRT and AtlasRRT share the same search strategy, the better performance of the latter can be explained by the higher quality of the samples obtained from the atlas and by the more robust branch extension mechanisms, both possible thanks to the parametrization provided by the atlas. With the aim of elucidating which of these two factors is more relevant, we repeated the experiments with AtlasRRT but sampling in ambient space. Note that line 3 of Algorithm 3 projects the samples on the tangent space associated with the nearest node in the RRT. This cancels part of the bias introduced by sampling in the ambient space. Despite this correction factor, the execution times are double with respect to those obtained with the standard AtlasRRT. The exceptions are the robot hand example where the execution time is ten times larger and the experiment with the four Rx60 robots, where the execution time hardly varies. In this case, the advantage of the AtlasRRT is mostly provided by the more efficient branch generation procedure.

The Barrett arm experiment was used to evaluate the influence of the different parameters required by AtlasRRT. We varied ϵ from 0.05 to 0.25 in steps of 0.05, and we observed that the execution time remained in between 2 s and 3 s in all cases. The number of charts, although, increases to 200 when ϵ is set to the lowest value. Similar results are obtained when varying α from 0.2 to 0.7 in steps of 0.1 rad. When varying ρ from 0.25 to 1.5 in steps of 0.25, we observed that for the lower values of the parameter, the number of charts increases up to 350, and the execution time increases up to 4 s. This increment is due to the overhead of creating the charts. Note, however, that the performance is still remarkably good. When λ varies from 2 to 7 in steps of one unit, neither the execution time nor the number of charts changes significantly. Finally, we performed a series of experiments varying ρ_s from 2 to 6 in steps of one unit. For large values of this parameter, the execution time increases up to 6 s since most of the branches are stopped due to collisions, which hinders the effective exploration of the RRT. This is a well-known issue of sample-based path planning that is addressed by the dynamic-domain approach [47], which adapts the size of the sampling areas in different parts of the configuration space. Such a technique could be incorporated into the AtlasRRT, but we leave this point as a future work.

Finally, note that using the preliminary version of the AtlasRRT, a problem involving two Stäubli Rx60 robots and no obstacles was solved in about 14 s in average [27]. With the improvements introduced in this paper, the same problem is solved in about 0.2 s. These improvements allow addressing significantly more complex problems, such as the four Stäubli Rx60 example.

VII. CONCLUSION

In this paper, we have presented the AtlasRRT algorithm: an approach that uses an atlas to efficiently explore a configuration space manifold implicitly defined by kinematic constraints. Since defining the full atlas for a given manifold is an expensive process, the AtlasRRT algorithm intertwines the construction of the atlas and the RRT: The partially constructed atlas is used to sample new configurations and to generate new branches for the RRT, and the RRT is used to determine directions of expansion for the atlas. The approach retains the exploration bias typical of RRT approaches in the sense that the tree is strongly pushed toward yet unexplored regions of the configuration space manifold. In the experiments reported in this paper, AtlasRRT is more efficient than existing state-of-the–art approaches, although this might not be the case in problems where the configuration space is relatively similar to the ambient space, i.e., not very constrained problems. The computational tools used in AtlasRRT are more complex than the ones used in existing approaches, although we provide an implementation where new benchmarks can be easily tested [52].

A fundamental aspect not considered in this study is the presence of singularities. When reaching a singularity, control problems might appear and the forces in the robot actuators might be

undetermined, possibly leading to failures in the motors. Thus, ideally, the path planner must be able to determine singularity-free paths. This is an aspect not usually addressed in the literature and we are currently working on extensions of the presented planner to determine such paths [56]. Changes in the dimensionality of the configuration manifold also represent an issue for the presented planner. If these changes occur in the transition between consecutive stages of a given task, the presented planner can be used for each one of them separately.

Several extensions to the basic AtlasRRT algorithm can be devised. In particular, it might be useful to exploit the atlas to obtain a more meaningful distance between samples than the Euclidean one, in a way similar to what is done in [48]. Additionally, we would like to integrate cost functions to focus the planning on the relevant parts of the configuration space [57] and to explore the possible extension of the proposed planner to problems with differential constraints. Finally, the solution paths found so far are jagged, as is usually the case with sampling-based path planning methods. We are currently investigating the possible extension of the asymptotically optimal path planners [58] to the context of path planning under kinematic constraints [46].

REFERENCES

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

[2] S. M. LaValle, *Planning Algorithms*. New York: Cambridge Univ. Press, 2006.

[3] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[4] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," in *Proc. Workshop Algorithmic Comput. Robot.—New Directions*, 2000, pp. 293–308.

[5] Kineo Computer Aided Motion web page. [Online]. Available: http://www.kineocam.com, last accessed Oct. 2012.

[6] S. M. Lavalle, "Motion planning. Part I: The essentials," *IEEE Robot. Autom. Mag.*, vol. 18, no. 1, pp. 79–89, Mar. 2011.

[7] J. Canny, *The Complexity of Robot Motion Planning*. Cambridge, MA: MIT Press, 1988.

[8] L. Han and N. M. Amato, "A kinematics-based probabilistic roadmap method for closed chain systems," in *Proc. Workshop Algorithmic Comput. Robot. New Directions*, 2000, pp. 233–246.

[9] 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, no. 6, pp. 951–959, Dec. 2001.

[10] J. Cortés, "Motion planning algorithms for general closed-chain mechanisms" Ph.D. dissertation, Inst. Nat. Polytechnique de Toulouse, Toulouse, France, 2003.

[11] T. Siméon, J. P. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *Int. J. Robot. Res.*, vol. 23, no. 7/8, pp. 729–746, 2004.

[12] J.-P. Merlet, *Parallel Robots*. Boston, MA: Kluwer, 2000.

[13] C. Rosales, L. Ros, J. M. Porta, and R. Suárez, "Synthesizing grasp configurations with specified contact regions," *Int. J. Robot. Res.*, vol. 30, no. 4, pp. 431–443, 2011.

[14] A. Rodríguez, L. Basañez, and E. Celaya, "A relational positioning methodology for robot task specification and execution," *IEEE Trans. Robot.*, vol. 24, no. 3, pp. 600–611, Jun. 2008.

[15] G. Ballantyne and F. Moll, "The da Vinci telerobotic surgical system: Virtual operative field and telepresence surgery," *Surgical Clinics North Amer.*, vol. 83, no. 6, pp. 1293–1304, 2003.

[16] W. J. Wedemeyer and H. Scheraga, "Exact analytical loop closure in proteins using polynomial equations," *J. Comput. Chem.*, vol. 20, no. 8, pp. 819–844, 1999.

[17] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2010, pp. 300–307.

[18] Schunk Anthropomorphic Hand web page. [Online]. Available:http://www.schunk.com, last accessed Oct. 2012.

[19] C. Ott, O. Eiberger, W. Friedl, B. Bauml, U. Hillenbrand, C. Borst, A. Albu-Schafer, B. Brunner, H. Hirschmuller, and G. Hirzinger, "A humanoid two-arm system for dexterous manipulation," in *Proc. IEEE-RAS Int. Conf. Humanoid Robot.*, 2006, pp. 276–283.

[20] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 1383–1390.

[21] S. Dalibard, A. Nakhaei, F. Lamiraux, and J.-P. Laumond, "Whole-body task planning for a humanoid robot: A way to integrate collision avoidance," in *Proc. IEEE-RAS Int. Conf. Humanoid Robot.*, 2009, pp. 355–360.

[22] I. Havoutis and S. Ramamoorthy, "Motion synthesis through randomized exploration of submanifolds of configuration spaces," in *Proc. Robot Soccer World Cup XIII. Lecture Notes Artif. Intell.*, vol. 5949, 2009, pp. 92–103.

[23] X. Tang, S. Thomas, P. Coleman, and N. M. Amato, "Reachable distance space: Efficient sampling-based planning for spatially constrained systems," *Int. J. Robot. Res.*, vol. 29, no. 7, pp. 916–934, 2010.

[24] D. Berenson and S. S. Srinivasa, J. J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. Robot. Res.*, vol. 30, no. 12, pp. 1435–1460, 2011.

[25] J. M. Porta, L. Jaillet, and O. Bohigas, "Randomized path planning on manifolds based on higher-dimensional continuation," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 201–215, 2012.

[26] D. Hsu, J.-C. Latombe, and H. Kurniawati, "On the probabilistic foundations of probabilistic roadmap planning," *Int. J. Robot. Res.*, vol. 25, no. 7, pp. 627–643, 2006.

[27] L. Jaillet and J. M. Porta, "Path planning with loop closure constraints using an atlas-based RRT," in *Proc. Int. Symp. Robot. Res.*, 2011, to appear.

[28] L. Han and L. Rudolph, "Inverse kinematics for a serial chain with joints under distance constraints," in *Proc. Robot.: Sci. Syst. II*, 2006, pp. 177–184.

[29] J. Cortés and T. Siméon, "Sampling-based motion planning under kinematic loop closure constraints," in *Proc. 6th Int. Workshop Algorithmic Found. Robot.*, pp. 75–90, 2004.

[30] M. Gharbi, J. Cortés, and T. Siméon, "A sampling-based path planner for dual-arm manipulation," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, 2008, pp. 383–388.

[31] A. Shkolmik and R. Tedrake, "Path planning in 1000+ dimensions using a task-space Voronoi bias," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 2892–2898.

[32] Z. Yao and K. Gupta, "Path planning with general end-effector constraints: Using task space to guide configuration space search," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2005, pp. 1875–1880.

[33] M. Stilman, "Task constrained motion planning in robot joint space," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2007, pp. 3074–3081.

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

[35] C. Suh, T. T. Um, B. Kim, H. Noh, M. Kim, and F. C. Park, "Tangent space RRT: A randomized planning algorithm on constraint manifolds," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 4968–4973.

[36] A. Yershova and S. M. LaValle, "Motion planning for highly constrained spaces," in *Robot Motion and Control*, (ser. Lecture Notes Control and Information Sciences, vol. 396). New York: Springer, 2009, pp. 297–306.

[37] M. P. do Carmo, *Differential Geometry of Curves and Surfaces*. Englewood Cliffs, NJ: Prentice–Hall, 1976.

[38] M. E. Henderson, "Multiple parameter continuation: Computing implicitly defined k-Manifolds," *Int. J. Bifurcation Chaos*, vol. 12, no. 3, pp. 451–476, 2002.

[39] M. E. Henderson, "Higher-dimensional continuation," in *Numerical Continuation Methods for Dynamical Systems: Path Following and Boundary Value Problems.* New York, Springer, 2007.

[40] B. Krauskopf, H. M. Osinga, and J. Galán-Vioque, *Numerical Continuation Methods for Dynamical Systems: Path Following and Boundary Value Problems*. New York: Springer, 2007.

[41] B. Roth and F. Freudenstein, "Synthesis of path-generating mechanisms by numerical methods," *ASME J. Eng. Ind.*, vol. 85, pp. 298–307, 1963.

[42] A. J. Sommese and C. W. Wampler, *The Numerical Solution of Systems of Polynomials Arising in Engineering and Science*. Singapore: World Scientific, 2005.
[43] F.-C. Yang and E. J. Haug, "Numerical analysis of the kinematic dexterity of mechanisms," *J. Mech. Des.*, vol. 116, pp. 119–126, 1994.
[44] C. Rosales, J. M. Porta, and L. Ros, "Global optimization of robotic grasps," in *Proc. Robot.: Sci. Syst.*, 2011, pp. 289–296.
[45] W. C. Rheinboldt, "MANPACK: A set of algorithms of computations on implicitly defined manifolds," *Comput. Math. Appl.*, vol. 32, no. 12, pp. 15–28, 1996.
[46] L. Jaillet and J. M. Porta, "Asymptotically-optimal path planning on manifolds," in *Proc. Robot.: Sci. Syst.*, 2012, MIT press, to appear.
[47] A. Yershova, L. Jaillet, T. Siméon, and S. M. LaValle, "Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2005, pp. 3856–3861.
[48] R. Chaudhry and Y. Ivanov, "Fast approximate nearest neighbor methods for non-Euclidean manifolds with applications to human activity analysis in videos," in *Proc. Eur. Conf. Computer Vis.*, 2010, pp. 735–748.
[49] The SOLID web page. [Online]. Avaialble:http://www.dtecta.com, last accessed Oct. 2012.
[50] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, P. Alken, M. Booth, and F. Rossi, *GNU Scientific Library Reference Manual*. Godalming, U.K.: Network Theory, 2009.
[51] A. Yershova and S. M. LaValle, "Improving motion planning algorithms by efficient nearest neighbor searching," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 151–157, Feb. 2007.
[52] The Cuik project web page. [Online]. Available: http://www.iri.upc.edu/research/webprojects/cuikweb, last accessed Oct. 2012.
[53] J. M. Porta, L. Ros, and F. Thomas, "A linear relaxation technique for the position analysis of multiloop linkages," *IEEE Trans. Robot.*, vol. 25, no. 2, pp. 225–239, Apr. 2009.
[54] J. Denavit and R. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *Trans. ASME. Ser. E, J. Appl. Mech.*, vol. 23, pp. 215–221, 1955.
[55] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Mach. Syst.*, vol. MMS-10, no. 2, pp. 47–53, Jun. 1969.
[56] O. Bohigas, M. E. Henderson, L. Ros, and J. M. Porta, "A singularity-free path planner for closed-chain manipulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 2128–2134.
[57] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Trans. Robot.*, vol. 26, no. 4, pp. 635–646, Aug. 2010.
[58] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.

Léonard Jaillet received the Engineering degree in mechanical engineering from the Institut Superieur de Mecanique de Paris, Paris, France, and the Ph.D. degree in robotics from the University of Toulouse, Toulouse, France, in 2001 and 2005, respectively.

Since 2008, he has been a Postdoctoral Fellow with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. His current research interests include motion planning for complex robotic systems and molecular simulations for structural biology.

Josep M. Porta received the Engineering degree in computer science in 1994 and the Ph.D. degree (Hons.) in artificial intelligence in 2001, both from the Universitat Politècnica de Catalunya, Barcelona, Spain.

From 2001 to 2003, he was a Postdoctoral Researcher with the University of Amsterdam, Amsterdam, The Netherlands, doing research in autonomous robot localization using vision. He is currently an Associate Researcher with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona. His current research interests include planning under uncertainty and computational kinematics.