Path Planning with Loop Closure Constraints using an Atlas-based RRT

Léonard Jaillet and Josep M. Porta

Abstract In many relevant path planning problems, loop closure constraints reduce the configuration space to a manifold embedded in the higher-dimensional joint ambient space. Whereas many progresses have been done to solve path planning problems in the presence of obstacles, only few work consider loop closure constraints. In this paper we present the AtlasRRT algorithm, a planner specially 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 manifolds and to coordinate them forming an atlas. AtlasRRT simultaneously builds an atlas and a Rapidly-Exploring Random Tree (RRT), using the atlas to sample relevant configurations for the RRT, and the RRT to devise directions of expansion for the atlas. The new planner is advantageous since samples obtained from the atlas allow a more efficient extension of the RRT than state of the art approaches, where samples are generated in the joint ambient space.

1 Introduction

In the recent years, there has been a growing interest around the problem of path planning with loop closure constraints [3, 6, 10, 20, 27, 31]. The reason behind this interest is that this problem appears in many relevant problems in Robotics such such as coordinated manipulation [19], motion planning with parallel robots [34], robot grasping [25], constraint-based object positioning [24], or surgery robots [1]. This problem is also crucial in Biochemistry, when searching for conformational changes in molecular loops [37].

The mainstream of research in path planning in the two last decades [4, 16] has focused on variants of sampling-based path planners [14, 17] to efficiently solve

Léonard Jaillet and Josep M. Porta
Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Llorens i Artigas 4-6, Barcelona, Spain, e-mail: {ljaillet,porta}@iri.upc.edu

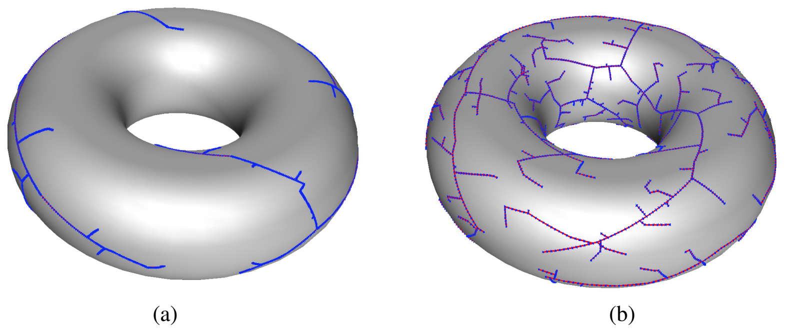


Fig. 1 Two RRTs built on a torus-like manifold after throwing 500 samples. (a) With an ambient space sampling, the exploration is focused on the exterior regions of the torus and many samples do not produce any tree extension. (b) With an AtlasRRT, the diffusion process is largely independent of the ambient space which improves the coverage.

the problem of robots with open loop kinematics operating in environments cluttered with obstacles. Obstacles induce a set of inequality constraints and planning motions that respect these constraints can be a very tough task, especially when it requires passing through narrow passages. Here, we address a more challenging situation where, beside the obstacles, the problem includes loop closure constraints represented by a set of equalities that must be fulfilled. Such constraints reduce the configuration space to a manifold embedded in the higher-dimensional ambient space defined by the joint variables involved in the problem.

The efficiency of sampling-based path planning approaches such as the Rapidly Exploring Random Trees (RRTs) relies in the so called Voronoi exploration bias [17] which can only be obtained if the space to explore can be properly sampled. Thus, ideally, these approaches would require an isometric parametrization of the configuration space from which a uniform distribution of samples can be generated. Whereas for non-constrained systems such parametrization is straightforward, this is not the case when the loop closure constraints reduce the dimensionality of the configuration space.

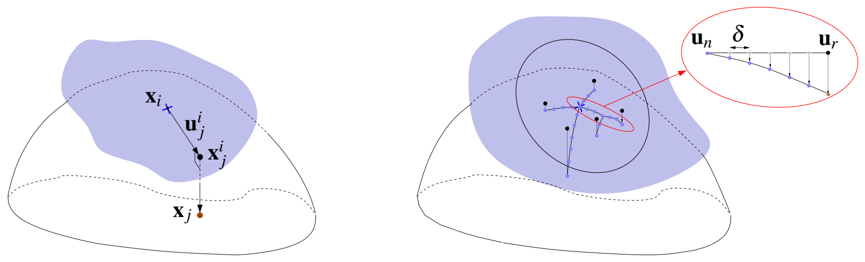
Distance-based formulations [9, 31] can provide a global parametrization of the constrained configuration space for some particular families of mechanism with kinematic loops. Other approaches try to infer the parametrization from large sets of samples on the manifold [10], and task-space planners assume that a subset of variables related to the end-effector are enough to parametrize the configuration space [3, 27, 40].

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 only valid for a limited class of mechanisms, and although some improvements have been proposed [5], the probability of generating invalid samples is significant. An alternative strategy to get valid configurations is to sample in the joint ambient space and to converge to the configuration space after each tree extension using numerical iterative techniques, either implementing random walks [38], or the more efficient Jacobian pseudoinverse method [2, 6, 30]. Despite being probabilistically complete [3], a uniform distribution of samples in the ambient space does not necessarily translate to a uniform distribution in the configuration space, which reduces the efficiency of these approaches. This problem is illustrated in Fig. 1 where the configuration space to be explored is a torus-like manifold of diameter four times smaller than the ambient space width. Fig. 1(a) shows a RRT built from points sampled in the ambient space that has a poor coverage of the manifold. With the AtlasRRT presented in this paper, the process of diffusion is largely independent of the configuration space shape and of the ambient space bounds which improves the coverage of the manifold, as shown in Fig. 1(b).

To improve the quality of the sampling, one can focus on a subset of the ambient space around the configuration space [43]. However, with this method points are still sampled in the ambient space, which can be of much higher dimensionality than the configuration space. Um *et al* [35] sketch a lazy RRT scheme where loosely coordinated RRTs are built on tangent spaces that locally approximates the manifold and that have the same dimensionality as the configuration space. However, the fact that the subtrees in different tangent spaces overlap affects the quality of the resulting RRT.

From Differential Geometry, it is well known that a manifold can be described by a collection of local parametrizations called charts, that can be coordinated within an atlas [22]. 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 [11, 12]. One-dimensional continuation methods, have been strongly developed in the context of Dynamical Systems [15], whereas in Robotics, they have been mainly used for solving problems related to Kinematics [26, 29]. To the best of our knowledge, higher-dimensional continuation methods have been only used in Robotics to evaluate the dexterity of mechanisms [39]. In a previous work [20], 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 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. To overcome these limitations, we propose here a probabilistic complete planner based on RRTs with the consequent gain of efficiency, specially for high dimensional configuration spaces. The new method called AtlasRRT is based on a coordinated construction of an atlas and a bidirectional RRT. On the one hand, the atlas is used to adequately sample new configurations and thus, to retain the RRT Voronoi bias, despite exploring 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.

This paper is organized as follows. Next section introduce the main mechanisms of the approach, showing how the atlas and the RRT expansions are coordinated. Then, Section 3 formally describes the algorithms implementing the AtlasRRT plan-



(a) (b)

Fig. 2 (a) A chart $\mathcal{C}_i$ is implemented as a mapping $\mathbf{x}_j = \psi_i(\mathbf{u}^i_j)$ between the tangent space at $\mathbf{x}_i$ and the manifold. (b) Growing a RRT on the manifold from a single chart. Black dots are the samples thrown in the tangent space and blue dots are points on the manifold forming the RRT branches. The inset shows the process of generating a branch by linear interpolation in the tangent space with steps of size δ and successive projection.

ner and in Section 4 we compare its performance to other state of the art methods for several benchmarks. Finally, Section 5 summarizes the contributions of this work and indicates points that deserve further attention.

2 Building an RRT on a Manifold

In this section, we first describe the elements describing a chart and how to build a RRT within it. Next, we describe how to define an atlas properly coordinating the charts to obtain a RRT covering the whole configuration space manifold.

2.1 RRT on a Chart

Let us consider a n-dimensional joint ambient space and a k-dimensional configuration space implicitly defined by a set of constraints

$$\mathbf{F}(\mathbf{x}) = 0, \tag{1}$$

with $\mathbf{F} : \mathbb{R}^n \rightarrow \mathbb{R}^{n-k}$, and $n > k > 0$. Note that we adopt the standard convention in Kinematics [18] where the configuration space is defined as the set of points fulfilling the constraints (this is sometimes called constrained configuration space) that is embedded in the ambient space of the joint variables (called configuration space in some approaches). Moreover, we assume that the configuration space is manifold everywhere, without considering the presence of singularities.

A chart, $\mathcal{C}_i$, locally parametrizes the k-dimensional manifold around a given point $\mathbf{x}_i$ with a bijective map, $\mathbf{x}_j = \psi_i(\mathbf{u}^i_j)$, between points $\mathbf{u}^i_j$ in $\mathbb{R}^k$ and points $\mathbf{x}_j$

on the manifold, with $\psi_i(\mathbf{0}) = \mathbf{x}_i$. Following [23], this mapping can be implemented using the k-dimensional space tangent at $\mathbf{x}_i$ (see Fig. 2). An orthonormal basis for this tangent space is given by the $m \times k$ matrix, Φ_i, satisfying

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

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

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

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

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

using a Newton procedure where $\mathbf{x}_j$ is initialized to $\mathbf{x}_j^i$ and is iteratively updated by the $\Delta\mathbf{x}_j$ increments fulfilling

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

until the error is negligible or for a maximum number of iterations.

The inverse mapping ψ_i^{-1} can be computed as the projection of a point on the tangent subspace

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

Using the mapping provided by a chart, $\mathcal{C}_i$, we can define a RRT on the part of the manifold covered by this chart, as shown in Fig. 2(b). This can be achieved using the standard RRT exploration mechanism and projecting to the manifold whenever necessary. Thus, the tree is initialized at $\mathbf{x}_i$, a random point, $\mathbf{u}_r$, is drawn in a ball of radius R in $\mathbb{R}^k$ and their coordinates in ambient space are obtained as $\mathbf{x}_r = \phi_i(\mathbf{u}_r)$. Then, the point, $\mathbf{x}_n$, already in the RRT and closer to $\mathbf{x}_r$ is determined. The tree is extended from $\mathbf{x}_n$ by interpolating between $\mathbf{u}_n = \psi_i^{-1}(\mathbf{x}_n)$ and $\mathbf{u}_r$, using steps of a small size δ and projecting to the manifold after each step, as shown in the inset of Fig. 2(b). If the projected sample is collision free, it is added to the RRT. Otherwise the tree extension is stopped. The result is a tree with points in the n-dimensional ambient space, but actually defined from a k-dimensional space.

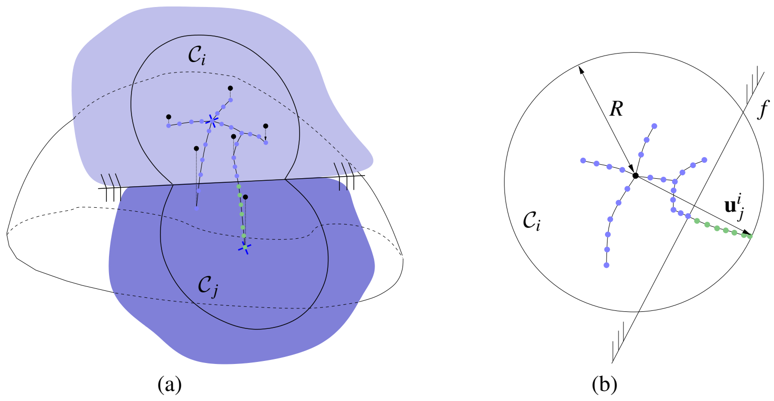


Fig. 3 (a) When a RRT leaves the validity area of chart $\mathcal{C}_i$, a new chart, $\mathcal{C}_j$, is created and their respective sampling areas are coordinated so that they do not overlap. (b) $\mathcal{C}_j$ adds an inequality, f, to $\mathcal{C}_i$ reducing its actual validity area. Note that after adding f, the nodes in green move from $\mathcal{C}_i$ to $\mathcal{C}_j$.

2.2 RRT on an Atlas

Unless for particularly simple problems, a single chart is not enough to parametrize the whole configuration space. The validity area, $\mathcal{V}_i$, for a chart, $\mathcal{C}_i$, is defined as the set of points $\mathbf{u}^i_j$ in the tangent space associated with $\mathcal{C}_i$ such that

$$\|\mathbf{u}^i_j\| \leq \beta\, R\,, \tag{7}$$

$$\|\mathbf{x}_j - \phi(\mathbf{u}^i_j)\| \leq \varepsilon\,, \tag{8}$$

$$\|\Phi_i^\top \, \Phi_j\| \leq 1-\varepsilon\,, \tag{9}$$

where $0 < \beta < 1$, R is the radius of the sampling ball, $\mathbf{x}_j = \psi_i(\mathbf{u}^i_j)$ is the projection of $\mathbf{u}^i_j$ on the manifold, and Φ_j is the basis of the tangent space at $\mathbf{x}_j$.

The first condition in the definition of $\mathcal{V}_i$ ensures that new charts are eventually created if the boundaries of the search area are reached. The second condition bounds the error between the tangent space and the manifold. The third condition ensures a smooth curvature in the part of the manifold covered by $\mathcal{C}_i$ and a smooth transition between charts. These two last conditions bound the distorsion introduced by the chart map and, thus, ensure that a uniform distribution of samples in $\mathcal{V}_i$ translates to an approximately uniform distribution of configurations on the manifold.

The validity area of a chart is not precomputed, but discovered as the RRT grows. Whenever the RRT reaches a point outside the validity area of a given chart $\mathcal{C}_i$, a new chart is added to the atlas on the last point still in $\mathcal{V}_i$. To avoid the overlap between the validity areas of neighboring charts we introduce a mechanism to reduce the validity areas, similar to that in [11] (see Fig. 3). We associate a set of inequalities, $\mathcal{F}_i$, to each chart. This set is initially empty and when two charts $\mathcal{C}_i$ and $\mathcal{C}_j$ are

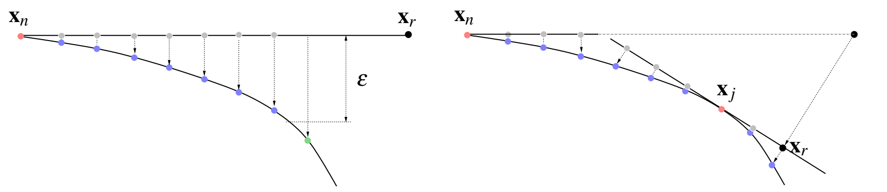


Fig. 4 RRT extension on a one-dimensional cut. The RRT is extended towards $\mathbf{x}_r$, the randomly selected point, from $\mathbf{x}_n$, the closest point already in the tree. In this case, $\mathbf{x}_n$ is the center of a chart. When the extension reaches a point (the green dot) where the error with respect to the chart at $\mathbf{x}_n$ is larger than ε, a new chart is created from the previous node already in the tree and $\mathbf{x}_r$ is projected to the new chart. Then, the RRT extension continues in the new chart until the projected $\mathbf{x}_r$ is reached.

neighbors (i.e., when the center of one of the charts are inside the validity area of the other chart) the following inequality is added to $\mathcal{F}_i$

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

with $\mathbf{u}^i_j = \psi_i^{-1}(\mathbf{x}_j)$. This reduces the validity area to a half space defined in the tangent space associated to $\mathcal{C}_i$ given by the plane orthogonally bisecting vector $\mathbf{u}^i_j$. Similarly, the validity area of $\mathcal{C}_j$ is cropped with the inequality defined from $\mathbf{u}^j_i = \psi_j^{-1}(\mathbf{x}_i)$, the projection of $\mathbf{x}_i$ on the tangent space associated with $\mathcal{C}_j$. When a given chart $\mathcal{C}_i$ is fully surrounded by neighboring charts, the intersection of the half spaces defined by the inequalities in $\mathcal{F}_i$ conform a polytope that conservatively bounds the actual validity area for the chart, taking into account the presence of neighboring charts.

The construction of an RRT on an Atlas proceeds as follows. First, a point $\mathbf{u}_r$ is sampled on the atlas. For this purpose, a chart $\mathcal{C}_r$ is randomly selected and a point is sampled on the ball of radius R defined on the tangent space associated with this chart. Random points not in the actual validity area (i.e., points not fulfilling the inequalities in $\mathcal{F}_r$) are rejected and the sampling process is repeated. Thus, the probability of generating a valid random point in a chart is proportional to the volume of its actual validity area and therefore, the sampling process selects points uniformly distributed within the area covered by the entire atlas.

The coordinates of sample $\mathbf{u}_r$ in ambient space, x_r, are computed using ϕ_r. Then, $\mathbf{x}_n$, the node already in the RRT closer to $\mathbf{x}_r$, is determined, the random point $\mathbf{x}_r$ is projected to the tangent space of the chart $\mathcal{C}_c$ including $\mathbf{x}_n$, and the tree extension proceeds in this chart as detailed in Section 2.1. The extension only stops if the random point is reached or if the path is blocked by an obstacle. However, during the tree extension the growing branch can leave $\mathcal{C}_c$. If one of the inequalities in $\mathcal{F}_c$ is not fullfilled by the new point to be added to the RRT, the extension entered in the validity area of the neighboring chart that generated the violated inequality in $\mathcal{F}_c$. In this case, the tree extension continues in this neighboring chart by projecting $\mathbf{x}_r$ on it. If the RRT extension leaves the validity area of $\mathcal{C}_c$ but the new point fulfills all the inequalities in $\mathcal{F}_c$ (if any), a new chart is generated and the branch

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       $\mathbf{x}_r \leftarrow$ SAMPLEONATLAS($A$)
7       $n_r \leftarrow$ NEARESTNODE($T_s, \mathbf{x}_r$)
8       $\mathbf{x}_l \leftarrow$ EXTENDTREE($T_s, A, n_r, \mathbf{x}_r$)
9       $n_l \leftarrow$ NEARESTNODE($T_g, \mathbf{x}_l$)
10      $\mathbf{x}'_l \leftarrow$ EXTENDTREE($T_g, A, n_l, \mathbf{x}_l$)
11      if $\|\mathbf{x}_l - \mathbf{x}'_l\| < \delta$ then
12          DONE $\leftarrow$ TRUE
13      else
14          SWAP($T_s, T_g$)
15  RETURN(PATH($T_s, \mathbf{x}_l, T_g, \mathbf{x}'_l$))
```

extension continues on it (see Fig. 4). Note that after creating a chart, some of the nodes assigned to neighboring charts might move to the area of validity of the new chart. These nodes are readily identified since they are the ones that do not fulfill the inequality introduced to by the new chart.

In the absence of other constraints, parameter β in Eq. (7) gives the ratio of sampled points that are outside the validity area and, thus, the ratio of points that will trigger the creation of new charts. Therefore, attending only to this criterion, the probability of creating new charts is

$$p = 1 - \beta^k \; . \tag{11}$$

3 AtlasRRT Algorithm

Algorithm 1 gives the pseudo-code for the AtlasRRT planner implementing the path planning approach described 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 search method.To this end, two RRTs are initialized (lines 1 and 2), with the start and the goal configurations as respective root nodes. An atlas is also initialized with two charts centered at each one of these points (line 3). Next, the algorithm iterates trying to connect the two trees (lines 5 to 14). First, a configuration is sampled using the atlas and one of the RRT is extended as much as possible towards this random sample from the nearest node already in the tree, mea-

Algorithm 2: Sampling on an atlas.

```
   SampleOnAtlas($A$)
   input  : The atlas, $A$.
   output: A sample on the atlas.
1  repeat
2      $r \leftarrow$ RandomChartIndex($A$)
3      $\mathbf{u}_r \leftarrow$ RandomOnBall($R$)
4  until $\mathbf{u}_r \in \mathcal{F}_r$
5  Return($\phi_r(\mathbf{u}_r)$)
```

sured using the Euclidean distance. The other RRT is then extended towards the last node added to the first tree, from the nearest node already in this second tree. If this second extension reaches its objective, the trees are connected and the nodes in the RRTs are used to reconstruct a path between $\mathbf{x}_s$ and $\mathbf{x}_g$. Otherwise, the two RRTs are swapped and the extension process is repeated. Note that this top level search algorithm is the same as that in [3]. The differences appears in how to sample random points and how to add branches to the RRTs.

In the sampling process we take advantage of the atlas, as shown in Algorithm 2. A chart is selected at random with uniform distribution and then, a point is sampled within the ball of radius R bounding the sampling area of this chart. The process is repeated until a point is inside the actual validity area associated with its selected chart, i.e., until it fulfills all the inequalities associated with the chart, if any. Finally, the sample returned is formed by the ambient space coordinates for the selected point computed using the mapping ϕ_r for the selected chart, r.

The addition of a branch to a tree T is done following the steps detailed in Algorithm 3. This procedure operates in the chart $\mathcal{C}_c$ including the node to be extended, that initially is the chart associated to the nearest node n (line 1). The sample to reach in the tree extension is projected on $\mathcal{C}_c$ (lines 2 and 3) and the branch extension is iteratively performed while the branch is not blocked and $\mathbf{x}_r$ is not reached (lines 5 to 23). At each iteration, a node is added to the tree. To define the node to add, a small step of size δ is performed in the parameter space of $\mathcal{C}_c$ from the parameters of the current node $\mathbf{u}_n$ towards the parameters for the goal sample, $\mathbf{u}_r$. The resulting parameters $\mathbf{u}_j$ are projected to the manifold to obtain the configuration $\mathbf{x}_j$ (line 7). If the projection is not too far away from $\mathbf{x}_n$ and if it is collision free, the point is added to the tree. First, however, we verify if the new point is still inside $\mathcal{C}_c$. On the one hand, if the parameters for the point are outside the validity area of this chart (line 12), a new chart is added to the atlas (line 13) which becomes the chart where to operate. Procedure NewChart implements the chart creation, the detection of neighboring charts, and the reassignment of tree nodes, if necessary. On the other hand, if the parameters for the point are inside the validity area, but outside the area defined by the inequalities associated with $\mathcal{C}_c$, the point is in the validity area of a neighboring chart. Procedure MoveToChart (line 17) identifies this chart. Whenever the current chart changes (line 19), we compute the parameters for the current sample and for the goal one in the new $\mathcal{C}_c$. Note that this affects the computation of

Algorithm 3: Adding a branch to the AtlasRRT.

```
ExtendTree($T, A, n, \mathbf{x}_r$)
input : A tree, $T$, an atlas, $A$, the index of the nearest node in the tree, $n$, and the random
        sample, $\mathbf{x}_r$.
output: The last node added to the tree or $\mathbf{x}_n$ if no extension was performed.
1  $c \leftarrow$ ChartIndex($n$)
2  $\mathbf{u}_r \leftarrow \psi_c^{-1}(\mathbf{x}_r)$
3  $\mathbf{x}_r \leftarrow \phi_c(\mathbf{u}_r)$
4  blocked $\leftarrow$ False
5  while not blocked and $\|\mathbf{u}_n - \mathbf{u}_r\| > \delta$ do
6      $\mathbf{u}_j \leftarrow (\mathbf{u}_r - \mathbf{u}_n)\,\delta / \|\mathbf{u}_r - \mathbf{u}_n\|$
7      $\mathbf{x}_j \leftarrow \psi_c(\mathbf{u}_j)$
8      if $\|\mathbf{x}_j - \mathbf{x}_n\| > 2\,\delta$ or Collision($\mathbf{x}_j$) then
9          blocked $\leftarrow$ True
10     else
11         new $\leftarrow$ False
12         if $\mathbf{u}_j \notin \mathcal{V}_c$ then
13             $c \leftarrow$ NewChart($A, \mathbf{x}_n$)
14             new $\leftarrow$ True
15         else
16             if $\mathbf{u}_j \notin \mathcal{F}_c$ then
17                 $c \leftarrow$ MoveToChart($\mathbf{x}_n, \mathbf{u}_j$)
18                 new $\leftarrow$ True
19         if new then
20             $\mathbf{u}_j \leftarrow \psi_c^{-1}(\mathbf{x}_j)$
21             $\mathbf{u}_r \leftarrow \psi_c^{-1}(\mathbf{x}_r)$
22             $\mathbf{x}_r \leftarrow \phi_c(\mathbf{u}_r)$
23         $n \leftarrow$ AddNode($T, A, c, \mathbf{x}_j$)
24 Return($\mathbf{x}_n$)
```

the next point, at line 6. Finally, the new point is added to the tree and associated with the current chart, $\mathcal{C}_c$ (line 23).

Concerning the algorithm complexity and not considering the cost of collision detection, the most expensive steps in the algorithm are the search for nearest nodes in the tree (lines 7 and 9 of Algorithm 1), the identification of neighboring charts when adding a chart to the atlas (line 13 of Algorithm 3), and the computation of the mapping ψ_c in line 7 of Algorithm 3. The first two operations can be implemented using hierarchical structures reducing their cost to be logarithmic in the number of nodes of the corresponding tree and in the number of charts in the atlas, respectively. The cost of computing the mapping ψ_c 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.

The algorithm basically uses four parameters: R, ε, δ, and p that appears in Eq. (11). R is a parameter defining the sampling area around a given point and can be safely set to a large value since the other criteria defining the validity area of a chart will eventually trigger the chart creation. ε regulates the amount of charts in the atlas. To address problems with a configuration space of moderate to large dimensionality it should not be set too small. A small δ should be used to avoid undetected collision between consecutive nodes in the RRT and sudden changes in the manifold curvature. Finally, p regulates the exploration since the larger this parameter the stronger the bias towards unexplored regions. In our experience adjusting the parameters is not an issue since the same values give good results for a large set of problems.

The probabilistic completeness of the RRT generated on a single chart over its sampling area is the same as that of a RRT defined in a k-dimensional Euclidean space. Thus, any point in the collision free region including the root of the tree will be eventually reached by the tree. In particular, points out of its validity area will be eventually reached and, consequently, new charts will be generated. The probabilistic completeness over the area covered by a new chart is also equivalent to that of a plain RRT. Assuming that the transition between charts is continuous and smooth, the local probabilistic completeness for each chart implies a global probabilistic completeness for the overall algorithm.

4 Experiments

We implemented the AtlasRRT planner described through Sections 2 and 3 in C. The planner was integrated as a module of our position analysis toolbox [32], using SOLID [33] as collision detector, the GNU Scientific Library [7] for the linear algebra operations, and the kd-tree described in [42] for the nearest-neighbor queries. In principle, simple formulations are advantageous for continuation methods [36] and, thus, our position analysis toolbox is based on a formulation with redundant variables that yields a system of simple equations only containing linear, bilinear, and quadratic monomials [21]. This is a particularly challenging situation since the manifold here arises not only because of the loop closure constraints, but also due to the equations necessary to compensate for the redundancy in the formulation. Due to this redundancy, the planning system introduced in [35] can not be directly applied to this case. Thus, for the sake of comparison we use the HC-planner [20] and an adaptation of the planner introduced in [3] that is called here CB-RRT. The HC-planner is a resolution completed planner on manifolds that is based on a greedy best first search strategy on the graph implicitly defined by the centers of the charts in the atlas. The CB-RRT planner shares the bi-directional search strategy with AtlasRRT, but randomly samples in the joint ambient space and uses the Jacobian pseudo-inverse procedure to converge to points on the manifold when necessary. Note, however, that in our implementation some aspects of the planner in [3] are not considered (i.e., the Task Space Regions and the direct sampling using them) since they are neither included in the AtlasRRT. The comparison with the HC-Planner is used to evaluate the AtlasRRT search strategy whereas the comparison with the CB-RRT is used to assess the advantage of the atlas-based sampling strategy.

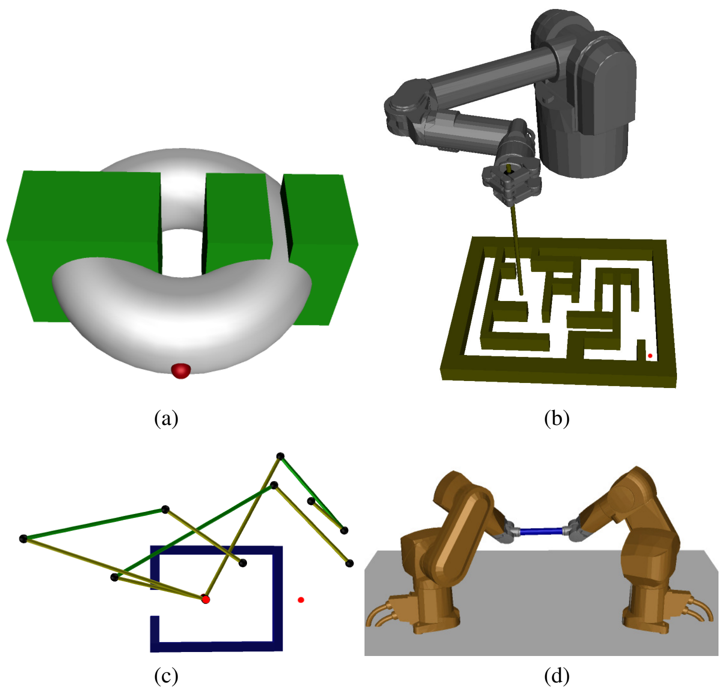


Fig. 5 The four benchmarks used to test the AtlasRRT planner. (a) A point moving on a torus with obstacles and a narrow corridor. (b) The Barret arm solving a maze problem. (c) A planar parallel manipulator moving a peg out of a cul-de-sac. (d) Two Stäubli Rx60 industrial arms collaborating to move a cylindrical bar.

Table 1 Dimension of the configuration and ambient spaces, success rates, execution times, and number of nodes/charts for the three methods compared in this paper.

			HC-Planner			CB-RRT			AtlasRRT			
Benchmark	k	n	Succ.	Time	Charts	Succ.	Time	Nodes	Succ.	Time	Charts	Nodes
Torus	2	7	0.76	0.03	77	1.0	26.07	3177	1.0	0.10	57	1456
Barret	4	84	0.36	279.98	2204	0.04	584.66	7563	0.76	356.36	477	14413
Star	5	18	0.68	49.08	2779	1.0	5.43	13989	1.0	4.70	1028	22394
Two Rx60	6	108	1.0	90.77	158	-	-	-	1.0	14.24	20	366

Fig. 5 shows the four benchmarks used in this paper, ordered by increasing configuration space dimensionality. The first one is a problem where a ball has to pass through a narrow corridor, while staying on the surface of a torus. This example is used to emphasize the fact that AtlasRRT can be advantageous even in simple cases. The second example is the Barret arm solving a maze where the start configuration is depicted in the figure and the goal is marked with a red spot. The stick moved

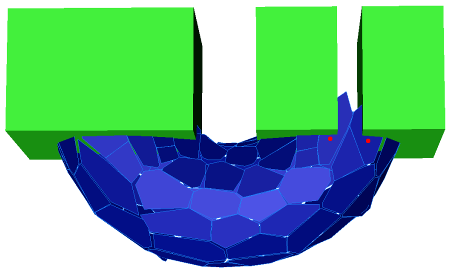


Fig. 6 For given resolution, the HC-planner can get blocked when trying to enter in a narrow corridor since it only considers motions from the centers of chart (the red points in the figure) at the borders of the atlas.

by the arm has to stay in contact with the maze plane and perpendicular to it. Note, however, that the rotation around the stick is not blocked. This problem is specially challenging due to the obstacles. The third example is a planar mechanism similar to that used in [28] except that here, obstacles are considered since the manipulator has to move a peg attached to the point marked in red out of a cul-de-sac. The goal pose of the end effector is also marked with a red spot. This example is of mixed difficulty with respect to the arrangements of obstacles and to the dimensionality of the configuration space. Finally, the fourth example is a manipulation task with two Stäubli Rx60 industrial arms. In this case, collisions are not considered and the difficulty of the task arises only from the loop closure constraints and from the dimensionality of the configuration space.

Table 1 shows the performance comparison between the HC-planner, the CB-RRT and the AtlasRRT, averaged over 25 runs and for a maximal execution time of 600 seconds on a Intel Core i7 at 2.93 Ghz running Mac OS X with parameters set to $R = 0.75$, $\varepsilon = 0.5$, $\delta = 0.05$, and $p = 0.9$ for all the experiments. 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 success (in the Succ. column), and for the successful runs, the execution times (in the Time column), as well as the number of charts and nodes required (given in the Charts and Nodes columns, respectively).

The results show that the AtlasRRT has always a better (or the same) success ratio than the HC-planner. For low dimensionality configuration spaces the HC-planner, when successful, is significantly fast. However, as the complexity of the obstacles grows, the probability of the HC-planner to fail also increases. The reason behind these failures is that when the HC-planner tries to enter a narrow corridor using too large charts, the atlas extension can get blocked as shown in Fig. 6. In these situations, there is not straight line between the center of the charts at the entrance of the corridor (marked with red dots) and the vertexes defining the frontier of expansion of the atlas. These failures can be avoided using a smaller R, but this will suppose an increase in memory use and a decrease in performance since more charts will be generated for the same problem. In the same situation, AtlasRRT will eventually grow a branch passing through the narrow passage, without adjusting the chart size. As the dimensionality of the configuration space increases, the HC-planner is also less efficient than AtlasRRT, even in problems without obstacles as for the manipulation problem with the two Rx60.

AtlasRRT is faster and at least as reliable as CB-RRT in all cases. Since both methods share the same search strategy, the better performance of the AtlasRRT can be explained by the higher quality of the samples obtained from the atlas. Note that, in general, AtlasRRT generates more nodes with a lower computational cost than CB-RRT since it does not suffer from unfruitful extensions, as when sampling in the ambient space. Finally, the advantage of our method is particularly significant in the Barret example where CB-RRT only succeeded once out of 25 attempts and in the manipulation task problem with the two Rx60 where CB-RRT fails in all cases.

5 Conclusions

In this paper, we presented the AtlasRRT algorithm, an approach that uses an atlas to efficiently explore a configuration space manifold implicitly defined by loop closure constraints. The atlas is a collection of charts that locally parametrize the manifold. Samples are thrown uniformly on the charts and, since the error from the chart to the manifold is bounded, the distribution of samples on the manifold is close to be uniform. These samples are then used to efficiently grow a RRT connecting two given configurations and avoiding collisions with obstacles. This strategy contrasts with state of the art approaches that generate samples on the configuration space from uniformly distributed samples in the ambient space.

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 for the RRT, and the RRT is used to determine directions of expansion for the atlas. The approach retains the Voronoi exploration bias typical of RRT approaches in the sense that exploration is strongly pushed towards yet unexplored regions of the configuration space manifold.

The results included in this paper shows that our approach is more efficient than existing state of the art approaches. A more thoughtful evaluation must be carried, though, to fully characterize the performance of the new algorithm. Moreover, several modifications to the basic AtlasRRT algorithm can be devised. In particular, it might be useful to exploit the atlas structure to obtain a more meaningful distance between samples than the Euclidean one. Moreover, the presented approach can be combined with existing strategies to improve path planning in the presence of obstacles such as, for instance, the dynamic domain sampling [41] or the integration of cost functions to focus the planning on the relevant parts of the configuration space [13]. Finally, we would like to explore the possible extension of the proposed planner to problems with differential constraints.

Acknowledgments

We would like to thank L. Ros for pointing us to the higher-dimension continuation tools and for fruitful discussions during the elaboration of this work. This work has been partially supported by the Spanish Ministry of Science and Innovation under project DPI2010-18449.

References

1. Ballantyne, G., Moll, F.: The da Vinci telerobotic surgical system: Virtual operative field and telepresence surgery. Surgical Clinics of North America **83**(6), 1293–1304 (2003)
2. Berenson, D., Srinivasa, S.S., Ferguson, D., Kuffner, J.J.: Manipulation planning on constraint manifolds. In: IEEE International Conference on Robotics and Automation, pp. 1383–1390 (2009)
3. Berenson, D., Srinivasa, S.S., Kuffner, J.J.: Task space regions: A framework for pose-constrained manipulation planning. International Journal of Robotics Research (2011)
4. Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press (2005)
5. Cortés, J., Siméon, T., Laumond, J.P.: A random loop generator for planning the motions of closed kinematic chains using PRM methods. In: IEEE International Conference on Robotics and Automation, pp. 2141–2146 (2002)
6. Dalibard, S., Nakhaei, A., Lamiraux, F., Laumond, J.P.: Whole-body task planning for a humanoid robot: a way to integrate collision avoidance. In: IEEE-RAS International Conference on Humanoid Robots, pp. 355–360 (2009)
7. Galassi, M., et al.: GNU Scientific Library Reference Manual. Network Theory Ltd. (2009)
8. Han, L., Amato, N.M.: A kinematics-based probabilistic roadmap method for closed chain systems. In: Algorithmic and Computational Robotics - New Directions (WAFR2000), pp. 233–246 (2000)
9. Han, L., Rudolph, L.: Inverse kinematics for a serial chain with joints under distance constraints. In: Robotics: Science and Systems II, pp. 177–184 (2006)
10. Havoutis, I., Ramamoorthy, S.: Motion synthesis through randomized exploration of submanifolds of configuration spaces. In: RoboCup 2009: Robot Soccer World Cup XIII. Lecture Notes in Artificial Intelligence, vol. 5949, pp. 92–103 (2009)
11. Henderson, M.E.: Multiple parameter continuation: Computing implicitly defined k-Manifolds. International Journal of Bifurcation and Chaos **12**(3), 451–476 (2002)
12. Henderson, M.E.: Numerical continuation methods for dynamical systems: path following and boundary value problems, chap. Higher-Dimensional Continuation. Springer (2007)
13. Jaillet, L., Cortés, J., Siméon, T.: Sampling-based path planning on configuration-space costmaps. IEEE Transactions on Robotics **26**(4), 635–646 (2010)
14. Kavraki, L.E., Svestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation **12**, 566–580 (1996)
15. Krauskopf, B., Osinga, H.M., Galán-Vioque, J.: Numerical continuation methods for dynamical systems: path following and boundary value problems. Springer (2007)
16. LaValle, S.M.: Planning Algorithms. Cambridge University Press, New York (2006)
17. LaValle, S.M., Kuffner, J.J.: Rapidly-exploring random trees: Progress and prospects. In: Algorithmic and Computational Robotics - New Directions (WAFR2000), pp. 293–308 (2000)
18. Milgram, R.J., Trinkle, J.: The geometry of configuration spaces for closed chains in two and three dimensions. Homology, Homotopy and Applications **6**(1), 237–267 (2004)

19. Ott, C., Eiberger, O., Friedl, W., Bauml, B., Hillenbrand, U., Borst, C., Albu-Schafer, A., Brunner, B., Hirschmuller, H., Hirzinger, G.: A humanoid two-arm system for dexterous manipulation. In: IEEE-RAS International Conference on Humanoid Robots, pp. 276–283 (2006)
20. Porta, J.M., Jaillet, L.: Path planning on manifolds using randomized higher-dimensional continuation. 9th International Workshop on the Algorithmic Foundations of Robotics pp. 337–353 (2010)
21. Porta, J.M., Ros, L., Thomas, F.: A linear relaxation technique for the position analysis of multiloop linkages. IEEE Transactions on Robotics **25**(2), 225–239 (2009)
22. Pressley, A.: Elementary Differential Geometry. Springer Verlag (2001)
23. Rheinboldt, W.C.: MANPACK: A set of algorithms of computations on implicitly defined manifolds. Computers and Mathematics with Applications **32**(12), 15–28 (1996)
24. Rodríguez, A., Basañez, L., Celaya, E.: A relational positioning methodology for robot task specification and execution. IEEE Transactions on Robotics **24**(3), 600–611 (2008)
25. Rosales, C., Ros, L., Porta, J.M., Suárez, R.: Synthesizing grasp configurations with specified contact regions. International Journal of Robotics Research **30**(4), 431–443 (2011)
26. Roth, B., Freudenstein, F.: Synthesis of path-generating mechanisms by numerical methods. ASME Journal of Engineering for Industry **85**, 298–307 (1963)
27. Shkolmik, A., Tedrake, R.: Path planning in 1000+ dimensions using a task-space Voronoi bias. In: IEEE International Conference on Robotics and Automation, pp. 2892–2898 (2009)
28. Shvlab, N., Liu, G., Shoham, M., Trinkle, J.C.: Motion planning for a class of planar closed-chain manipulators. International Journal of Robotics Research **26**(5), 457–473 (2007)
29. Sommese, A.J., Wampler, C.W.: The Numerical Solution of Systems of Polynomials Arising in Engineering and Science. World Scientific (2005)
30. Stilman, M.: Task constrained motion planning in robot joint space. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3074–3081 (2007)
31. Tang, X., Thomas, S., Coleman, P., Amato, N.M.: Reachable distance space: Efficient sampling-based planning for spatially constrained systems. International Journal of Robotics Research **29**(7), 916–934 (2010)
32. The CUIK project web page: http://www.iri.upc.edu/research/webprojects/cuikweb
33. The SOLID web page: http://www.dtecta.com
34. Tsai, L.W.: Robot Analysis: The Mechanics of Serial and Parallel Manipulators. John Wiley and Sons (1999)
35. Um, T.T., Kim, B., Suh, C., Park, F.C.: Tangent space RRT with lazy projection: An efficient planning algorithm for constrained motions. In: Advances in Robot Kinematics, pp. 251–260 (2010)
36. Wampler, C.W., Morgan, A.: Solving the 6R inverse position problem using a generic-case solution methodology. Mechanism and Machine Theory **26**(1), 91–106 (1991)
37. Wedemeyer, W.J., Scheraga, H.: Exact analytical loop closure in proteins using polynomial equations. Journal of Computational Chemistry **20**(8), 819–844 (1999)
38. Yakey, J.H., LaValle, S.M., Kavraki, L.E.: Randomized path planning for linkages with closed kinematic chains. IEEE Transactions on Robotics and Automation **17**(6), 951–959 (2001)
39. Yang, F.C., Haug, E.J.: Numerical analysis of the kinematic dexterity of mechanisms. Journal of Mechanical Design **116**, 119–126 (1994)
40. Yao, Z., Gupta, K.: Path planning with general end-effector constraints: Using task space to guide configuration space search. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1875–1880 (2005)
41. Yershova, A., Jaillet, L., Siméon, T., LaValle, S.M.: Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. In: IEEE International Conference on Robotics and Automation, pp. 3856–3861 (2005)
42. Yershova, A., LaValle, S.M.: Improving motion planning algorithms by efficient nearest neighbor searching. IEEE Transactions on Robotics **23**(1), 151–157 (2007)
43. Yershova, A., LaValle, S.M.: Motion planning for highly constrained spaces. In: Robot Motion and Control. Lecture Notes on Control and Information Sciences, vol. 396, pp. 297–306 (2009)