Tangent Space RRT with Lazy Projection: An Efficient Planning Algorithm for Constrained Motions

Terry Taewoong Um, Beobkyoon Kim, Chansoo Suh and Frank Chongwoo Park

School of Mechanical and Aerospace Engineering,
Seoul National University, Seoul 151-744, Korea;
e-mail: {terryum, beokuni0, cssuh}@robotics.snu.ac.kr, fcp@snu.ac.kr

Abstract. Rapidly-Exploring Random Trees (RRT) have been successfully used in motion planning problems involving a wide range of constraints. In this paper we develop a more robust and efficient version of the constrained RRT planning algorithm of [1]. The key idea is based on first constructing RRTs on tangent space approximations of constraint manifold, and performing lazy projections to the constraint manifold when the deviation exceeds a prescribed threshold. Our algorithm maintains the Voronoi bias property characteristic of RRT-based algorithms, while also reducing the number of projections. Preliminary results of a numerical study, together with a discussion of the potential strengths and weaknesses of our algorithm, are presented.

Key words: Rapidly-exploring random tree, constrained motion planning, lazy projection.

1 Introduction

In order for robots to function autonomously in unstructured environments, they must be able to perform a wide range of constrained motions, *e.g.*, holding a tray with two hands while maneuvering through a dynamically changing obstacle cluttered environment, or pushing a heavy object through a narrow passageway with intermittent gaps, all the while satisfying various constraints on, *e.g.*, the velocities, accelerations, forces and torques on the joints and the object being manipulated, as well as various contact constraints among the robot, manipulated object, and the environment.

A mathematical formulation of the constrained motion planning problem in its most general form is highly complex, and its solution even more challenging. In this paper we focus on a simpler and more tractable version of this problem that focuses on holonomic constraints; the formulation is still reasonably general, and should readily admit extensions to various nonholonomic and other input constraints not explicitly considered here. The setup is as follows. Let $\mathscr{M}$ be configuration space of the robot, with local coordinates q. Denote by $\mathscr{N}$ its task space, with local coordinates X, and let $f : \mathscr{M} \rightarrow \mathscr{N}$ denote its forward kinematics; in local coordinates we thus have $X = f(q)$. Finally, let $q_{init} \in \mathscr{M}$ and $q_{final} \in \mathscr{M}$ be the user-supplied initial and final configurations, respectively. The objective then is to seek a continuous path from q_{init} to q_{final} that lies entirely on M, while satisfying a further set of task constraints of the form

$$\psi(X) = 0, \tag{1}$$
$$\nu(X) \leq 0, \tag{2}$$

where $\psi : \mathscr{N} \rightarrow \Re^p$ and $\nu : \mathscr{N} \rightarrow \Re^k$ are given and assumed differentiable. Note that $\psi(X) = c$ can also be expressed via the forward kinematics f as $\psi(f(q)) = g(q) = 0$. The inequality constraints can be similarly expressed in the form $\nu(f(q)) = h(q) \leq 0$. Assuming the forward kinematics is differentiable everywhere, $g(q) = 0$ and those elements of $h(q) \leq 0$ that are active (that is, those elements of $h(q)$ for which strict equality holds) then constitute a set of holonomic constraints on $\mathscr{M}$.

As an illustration, a planar six-bar linkage has a configuration space that can be regarded as a three-dimensional surface embedded in $\Re^6$ (more accurately, a three-dimensional surface embedded in a six-dimensional flat torus). If a large vertical obstacle were present in the middle of the workspace, an inequality constraint can be imposed requiring the robot to remain clear of this obstacle; such a constraint may, e.g., separate the configuration space into two disjoint three-dimensional regions, possibly connected by a two-dimensional surface. The resulting configuration space taking into account all the task constraints may result in a highly complex subset of $\mathscr{M}$.

More generally, trying to determine *a priori* the set of configurations defined by $g(q) = 0$ and $h(q) \leq 0$, especially for robots with a large number of degrees of freedom, is in most cases computationally prohibitive. Moreover, equality constraints of the form $g(q) = 0$ define surfaces that we shall refer to as constraint manifolds; discovering configurations that lie on constraint manifolds via random sampling of the configuration space (e.g., with sampling based planners such as RRTs, or probabilistic road maps that directly sample the configuration space) is extremely unlikely. As pointed out in [1], the lack of prior knowledge of the constraint manifold structure also precludes the use of task space control techniques of the type described in [2] as a complete solution. Alternative approaches, such as first planning a path in a lower-dimensional space (like the constraint manifold) and then attempting to follow this reference path in the full configuration space of the robot [3, 4] suffers from feasibility problems; the lower-dimensional path may not be trackable becuase of joint limits or collisions.

In response to these difficulties, Berson et al. [1] have proposed a constrained bi-directional RRT algorithm (CBiRRT) that addresses the problem of sampling on constraint manifolds (we refer the reader to [5–7] and the references for a survey of RRT methods, and randomized motion planning methods in general). The CBiRRT algorithm first samples in the configuration space $\mathscr{M}$, then uses projection operations to move samples onto constraint manifolds according to some criterion. The projection step can be performed in a number of ways, e.g., the randomized gradient descent method (RGD) [8] originally developed for closed chain motion planning and its extension to general end-effector constraints [9], or the Jacobian pseudo-

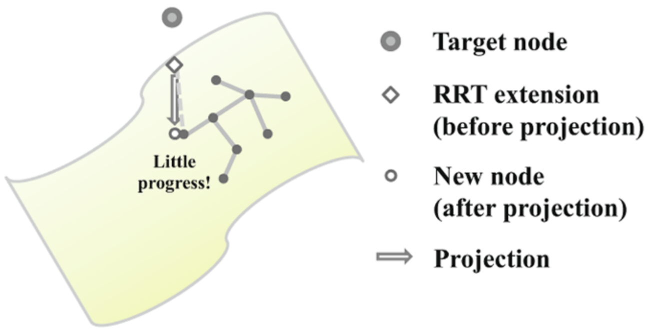
Little progress!
Target node
RRT extension (before projection)
New node (after projection)
Projection

Fig. 1 Example of a target node whose projection contributes very little to extending the tree in the constraint manifold.

inverse projection method of [10]. Whatever the choice of projection method, the fact remains that the manner in which the random configuration is projected to the constraint manifold is in most cases the computational bottleneck, particularly since the Voronoi bias property of RRT algorithms – in a nutshell, the tendency of RRT algorithms to seek out unexplored regions [11] – is considerably diminished. At a more fundamental level, because the shape of the constraint manifold is not considered when generating random samples in the configuration space, performance of the algorithm can be significantly degraded (see Fig. 1).

To address the shortcomings of current RRT-based algorithms for constrained motion planning, in this paper we propose a new algorithm, the Tangent Space RRT (TS-RRT) algorithm. Whereas the CBiRRT algorithm samples target nodes in the configuration space, and projects this node to the constraint manifold, the key idea in our approach is to first sample and construct an RRT in the tangent space of the constraint manifold. When the RRT constructed in the tangent space deviates from the constraint manifold by a certain prescribed threshold, we project to the constraint manifold, construct the tangent space approximation at the projected point, and repeat the above sample-projection procedure. Three additional distinctive features of our algorithm are as follows: (i) we develop a "lazy projection" procedure to reduce the overall number of projections, inspired in part by the lazy collision checking procedure [12] that attempts to reduce the number of collision tests; (ii) we choose the initial direction for RRT construction in the tangent plane to maintain the Voronoi bias toward unexplored regions; (iii) we use local curvature information to scale the tangent space domain over which sampling is performed (for example, nearly zero curvature indicates that the constraint manifold is close to being flat, so that the tangent space approximation is valid over a larger region of the manifold).

2 Tangent Space RRT Algorithm

For space reasons we omit a discussion of the basic RRT construction algorithm; the reader is referred to [5,6] for a review. We shall take the basic RRT construction algorithm as our point of departure, and describe the Tangent Space RRT (TS-RRT) algorithm as an extension of the basic RRT algorithm. Let us first assume that the given q_{init} and q_{final} are already on $\mathscr{M}$. Recall that the main distinguishing feature of TS-RRT is that branches of exploring trees are constructed on the tangent planes of the constraint manifold instead of in the configuration space. Two tangent planes to the constraint manifold are initially constructed at q_{init} and q_{final}. Starting with q_{init} and q_{final} as the root nodes, two trees are then grown on the tangent planes via the basic RRT algorithm. We now discuss the basic components of the TS-RRT algorithm.

Projection by Newton–Raphson Method

We regard $\mathscr{M}$ as a surface embedded in some higher dimensional normed Euclidean space. Given a configuration q not on $\mathscr{M}$, the natural way to compute the distance between q and $\mathscr{M}$ is to find the point $p \in \mathscr{M}$ that minimizes the distance $||p-q||$, where $||\cdot||$ is a suitably chosen norm on the Euclidean space. Because this nonlinear optimization problem is often difficult and computationally intensive, in practice one settles for easily obtained solutions that approximately minimize the distance function. If the constraint manifold is parametrized implicitly as $g(q)=0$, one easily implementable optimization procedure is to define the error vector e according to

$$e = g(q), \tag{3}$$

i.e., if q lies on $\mathscr{M}$ then e is zero, and nonzero otherwise. For q sufficiently close to $\mathscr{M}$, $||e||^2$ is a valid distance function that can be easily minimized via a Newton–Raphson root-finding procedure for $g(q)$ (see Table 1.) Specifically, a first-order Tayler expansion of (3) leads to

$$e + \delta e \simeq g(q) + \frac{\partial g}{\partial q}(q)\delta q. \tag{4}$$

Here we denote the Jacobian matrix $\partial g/\partial q$ by $J(q)$. Setting the right-hand side of (4) to zero and solving for δq, we obtain the update rule

$$q_{new} \leftarrow q - J(q)^{\dagger}\delta e, \tag{5}$$

where $J(q)^{\dagger}$ denotes the pseudo-inverse of $J(q)$:

$$J(q)^{\dagger} = J(q)^T[J(q)J(q)^T]^{-1}. \tag{6}$$

Table 1 Projection by Newton–Raphson method.

```
Projection($q$)
1  $e \leftarrow g(q)$
2  while( $||e|| < \varepsilon$)
3      $q \leftarrow q - J(q)^{\dagger} e$
4      $e \leftarrow g(q)$
5  end
6  return $q$
```

Random Sampling on Tangent Planes

Given a root node q_{root} lying on $\mathscr{M}$, a basis for the tangent space to $\mathscr{M}$ at q_{root} can be obtained by applying a Gram–Schmidt procedure to the following projection matrix $P(q_{root}) \in \Re^{n \times n}$:

$$P(q_{root}) = I - J(q_{root})^{\dagger} J(q_{root}), \tag{7}$$

where n denotes the dimension of the the configuration space. The basis is used to generate a random sample on the tangent plane; note that the basis needs to be computed only once for each root node. To illustrate the procedure, suppose the constraint equation $g(q) = 0$ consists of m independent equations; in this case the tangent space is of dimension $n - m$, and an orthonormal basis $\{d_1, \ldots, d_{n-m}\}$ can be constructed at each point of $\mathscr{M}$, where each $d_i \in \Re^n$. A random sample q_{rand} on the tangent space can then be generated straightforwardly given q_{root} and the orthonormal basis.

Given a random sample node q_{rand} on the tangent plane, we find the nearest neighbor node q_{near} on the same tangent plane, and extend the tree a fixed length segment in the direction from q_{near} to q_{rand}. The extended node is then tested to see if the inequality constraints are satisfied. If it fails to satisfy the inequality constraints, the node is then abandoned, and the algorithm samples another random node on the tangent plane.

Determining the length of the fixed segment can be done in a number of ways. One effective way uses the local curvature information of the constraint manifold; the basic premise is that at points where the principal curvatures are nearly zero, the manifold is nearly flat, and thus relatively larger steps can be taken in the principal directions without deviating significantly from the manifold.

Creating a New Tangent Plane

When the distance $||e||$ from the extended node q_{new} to $\mathscr{M}$ exceeds a certain threshold, denoted $E_{\mathscr{M}}$, the tangent plane no longer approximates $\mathscr{M}$ with the desired accuracy. In such cases we project the extended node onto $\mathscr{M}$ using our previously described projection algorithm in Table 1, and treat the projected point as

Table 2 Create a tangent plane.

```
CreateTangentPlane($q$)
1  $q \leftarrow \text{Projection}(q)$
2  $P \leftarrow I - J(q)^{\dagger} J(q)$
3  $d \leftarrow q - q_{root,parent}$
4  $d \leftarrow P(q) * d$
5  $TB \leftarrow \text{GramSchmidt}(P)$
6  return $(TB, q)$
```

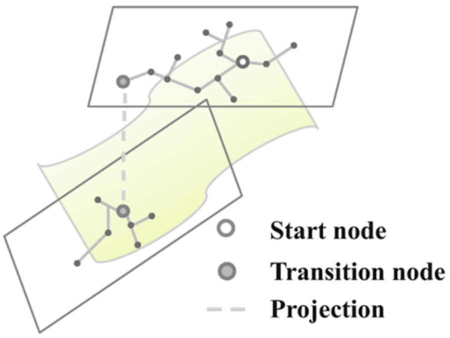

Fig. 2 In TS-RRT, tree branches are generated on tangent planes. When a newly sampled node exceeds a certain prescribed distance from the constraint manifold, that node is projected onto the manifold, and a new tangent plane is created.

a new root node for a new tangent plane generated via another Gram–Schmidt procedure. The process for generating a new root node and the accordant tangent plane is shown in Table 2 and Fig. 2.

When a new tangent plane is created, it is important to ensure that the new tangent plane not significantly overlap with the same area of $\mathscr{M}$ covered by the parent tangent plane. To prevent such overlapping, we restrict the domain for sampling a random node as follows. As described by the third and fourth lines of Table 2, we compute the vector from the root node of the parent tangent plane to the projected node that will be the new root node; the projection of this vector onto the new tangent plane is then used as the initial vector in the Gram–Schmidt process for constructing the tangent basis. In the random sampling phase, only positive weights are associated with this initial basis vector, to ensure that we do not "backtrack" to already explored regions of $\mathscr{M}$.

Selection Bias for Tangent Planes

The trees generated by the TS-RRT algorithm consist of multiple tangent planes and branch nodes, such that every extended node belongs to one of the two trees, and also to one of the tangent planes. Thus, when we randomly sample nodes, we first need to choose a tangent plane among all of the tangent planes created. To ensure that the tangent planes created later are selected more often, one can use the number of nodes that belong to the tangent plane; that is, if a certain tangent plane has fewer nodes than the others, it has a greater chance of being selected.

Another means of biasing selection tangent planes created later is to use local curvature information of $\mathscr{M}$; the greater the extrinsic curvature, the greater the error in approximating the constraint manifold by the tangent plane (for example, the mean curvature, given by the trace of the second fundamental form, can be used as a bias factor).

Connection Test

Recall that our proposed TS-RRT algorithm is bidirectional, in the sense that two trees – one each emanating from the initial and goal nodes – are simultaneously generated. After every node extension on the constraint manifold, the TS-RRT algorithm tries to connect the extended node to the nearest node on the opposite tree. This is done by evaluating the inner product between the connecting line vector and the rows of Jq (recall that the rows of $J(q)$ are orthogonal to the surface); if the inner products are sufficiently close to zero at both ends, the connecting segment is deemed to be sufficiently close to the tangent planes. In this case uniformly spaced points along the connecting segment are tested to see if the inequality constraints are satisfied. If at any of these points the constraints are violated, or the error $||e||$ exceeds the prescribed threshold, no connection is established.

Repeating the process described above, the resulting trees rapidly explore the multiple tangent planes that are used to approximate $\mathscr{M}$. Once a successful connection is established, the final path through the collection of tangent planes is extracted, and the nodes in the final path projected to the constraint manifold (see Fig. 3). The pseudo-code description of our TS-RRT algorithm is given in Table 3.

3 Case Study

3.1 Two-Arm Manipulation

In this example, we consider two arms being pulled from the inside of a drawer to the outside, all the while maintaining a fixed distance between the hands. The task constraint is of the form

$$g(X) = (p_{left} - p_{right})^2 - d^2 = 0, \tag{8}$$

Table 3 TS-RRT algorithm.

```
TS-RRT($q_{init}, q_{final}$)
1  Tree[], TangentPlanes[]
2  Tree.AddNode($q_{init}$), Tree.AddNode($q_{final}$)
3  TangentPlanes.Add(CreateTangentPlane($q_{init}$))
4  TangentPlanes.Add(CreateTangentPlane($q_{final}$))
5  while $i < I_{max}$ do $i \leftarrow i+1$
6      $k \leftarrow$ SelectTangentPlane()
7      $q_{rand} \leftarrow$ RandomSampleOnTangentPlane(TangentPlanes[$k$])
8      $q_{near} \leftarrow$ NearestNode($k, q_{rand}$)
9      $q_{new} \leftarrow$ Extend($q_{near}, q_{rand}$)
10     if $h(f(q_{new})) > 0$ then goto 3 end
11     if Connect($q_{new}$) = Success then
12         Path $\leftarrow$ ExtractPath()
13         return LazyProjection(Path)
14     else if $q_{new} > E_{\mathscr{M}}$ then
15         TangentPlanes.Add(CreateTangentPlane($q_{new}$))
16     end
17     Tree.AddNode($q_{new}$), Tree.AddEdge($q_{near}, q_{new}$)
18 end
19 return Fail
```

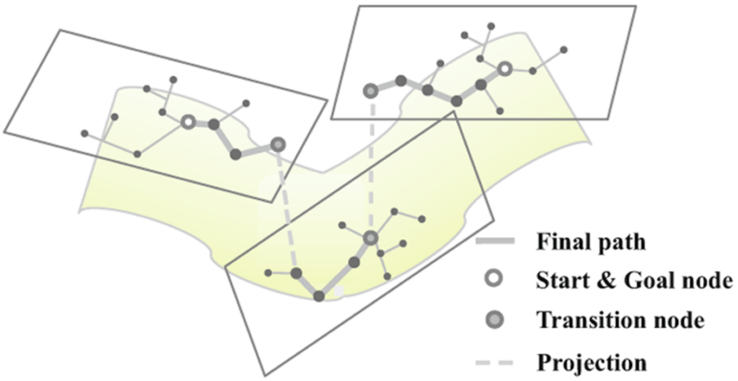

Fig. 3 The final path is extracted through multiple tangent planes. The actual path on the manifold can be obtained by projecting the nodes on the final path.

where p_{left} and p_{right} denote the Cartesian positions of the left and right hands of the robot, respectively. Collision avoidance between the robot arms and the drawer are described by an appropriate set of inequality constraints.

Preliminary tests of the TS-RRT algorithm suggest that it works more reliably and efficiently than the CBiRRT algorithm, but more extensive testing under a wider range of scenarios needs to be performed before firmer conclusions can be drawn.

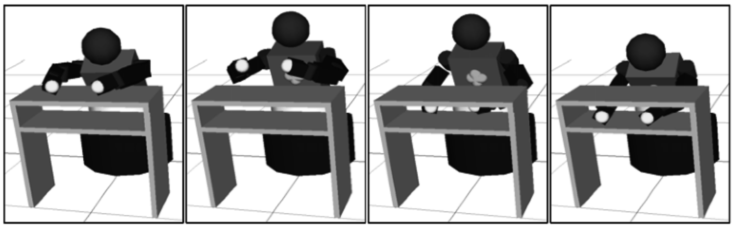

Fig. 4 Constraint on distance between two hands.

4 Conclusions and Future Work

This paper has presented a new randomized algorithm for task constrained motion planning, the TS-RRT algorithm, that is based upon the widely used concept of rapidly exploring random trees (RRT). Unlike existing RRT algorithms for constrained motion planning, the distinguishing feature of the TS-RRT algorithm is that RRTs are constructed in the tangent spaces of the constraint manifold, and projected to the manifold only when the newly sampled nodes exceed a certain threshold distance. Our method can be contrasted with existing approaches that project randomly sample in the configuration space, and project every sample back to the constraint manifold; such a procedure can result in nodes that only extend the tree minimally, possibly revisiting previously explored directions.

Preliminary numerical studies with our algorithm suggest that it is more reliable and computationally efficient than existing algorithms; more extensive testing is currently underway to compare its performance with the existing CBiRRT and other algorithms for constrained motion planning. More effective methods for exploiting local curvature information about the constraint manifold, as well as investigating ways to include, e.g., dynamics, nonholonomic constraints, and actuator force-torque limits into the planning framework, are also being investigated.

References

1. Berson, D., Srinivasa, S., Ferguson, D., and Kuffner, J., Manipulation planning on constraint manifolds, in *Proc. IEEE Int. Conf. Robotics and Automation* (2009).
2. Sentis, S. and Khatib, O., Synthesis of whole-body behaviors through hierarchical control of behavioral primitives, *Int. J. Humanoid Robots* (2005).
3. Koga, Y., Kondo, K., Kuffner, J., and Latombe, J., Planning motions with intentions. In *SIGGRAPH* (1994).
4. Yamane, K., Kuffner, J., and Hodgins, J., Synthesizing animations of human manipulation tasks. In *ACM Transactions on Graphics* (2004).
5. LaValle, S., Rapidly-exploring random trees: a new tool for path planning, TR 98-11, Computer Science Department, Iowa State University (1998).
6. LaValle, S. and Kuffner, J., Rapidly exploring random trees: progress and prospects. In *Workshop on Algorithmic Foundations of Robotics* (2000).

7. Kavraki, L., Svestka, P., Latombe, J., and Overmars, M., Probabilistic roadmaps for path planning in high-dimensional configuration spaces. In *Proc. IEEE Int. Conf. Robotics Automation* (2002).
8. LaValle, S., Yakey, J., and Kavraki, L., A probabilistic roadmap approach for systems with closed kinematic chains. In *Proc. IEEE Int. Conf. Robotics and Automation* (1999).
9. Yao, Z. and Gupta, K., Path planning with general end-effector constraints: Using task space to guide configuration space search. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (2005).
10. Stilman, M., Task constrained motion planning in robot joint space. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (2007).
11. Lindermann, S. and LaValle, S., Incrementally reducing dispersion by increasing Voronoi bias in RRTs. In *Proc. IEEE Int. Conf. Robotics and Automation* (2004).
12. Sanchez, G. and Latombe, J., A single query bi-directional probabilistic roadmap planner with lazy collision checking. In *Int. Symp. Robotics Research*, Lorne, Victoria, Australia (2001).