Steps Toward Derandomizing RRTs

Stephen R. Lindemann and Steven M. LaValle

Dept. of Computer Science, University of Illinois, Urbana, IL 61801 USA
\{slindemallavalle\}@uiuc.edu

18.1 Introduction

For over a decade, randomized algorithms such as the Randomized Potential Field Planner (RPP) [3], the Probabilistic Roadmap (PRM) family [1, 9, 21, 24, 25], Rapidly-Exploring Random Trees (RRTs) [10, 12, 13], and others [4, 8, 17], have dominated the field of motion planning. Recently, a great deal of attention has been given to comparing random versus deterministic sampling in the context of PRMs [6,11]. A recent survey of this field, termed sampling-based motion planning, is given in [15]. In this paper, we discuss the role of randomization in RRTs, and introduce two new planners which move toward their derandomization.

Randomization is a common algorithmic technique, and it is of great value in many contexts. Sometimes, it is used to defeat an adversary who might gain an advantage from learning one's deterministic strategy (e.g., cryptographic or sorting algorithms). Randomization is also useful for approximation or in conjunction with amplification techniques (e.g., the randomized min cut algorithm). It also allows for probabilistic performance analysis, which can be very useful. For problems of numerical integration, randomization can sometimes defeat the “curse of dimensionality” [22].

In the context of the original PRM, the primary use of randomization is to uniformly sample the configuration space. Recently, the usefulness of randomization for this purpose has been challenged; proponents of deterministic sampling argue that there are deterministic sequences satisfying other uniformity measures (e.g., discrepancy and dispersion) which perform at least as well as random sampling. Furthermore, these methods give deterministic guarantees of convergence (such as resolution completeness). Currently, work is being done both to study the performance of various random and deterministic sampling sequences in the PRM [11], and to construct new deterministic uniform sequences with other properties that are useful for motion planning [14].

It cannot be denied that contemporary motion planning algorithms, many of which use randomization, are very efficient and able to solve many challenging problems. This might lead one to conclude that randomization is the key to their effectiveness; however, this is not necessarily the case. On the contrary, randomization can easily become a “black box” which obscures the reasons for an algorithm’s success. Hence, attempts to derandomize popular motion planning algorithms do not reflect antipathy toward randomization, but rather the desire
BUILD_RRT ($x_{\text{init}}$)
1 $G_{\text{sub}}$.init($x_{\text{init}}$);
2 for $k = 1$ to $\text{max}$.iterations do
3 $x_{\text{rand}} \leftarrow$ RANDOM_STATE();
4 $x_{\text{near}} \leftarrow$ NEAREST_NEIGHBOR($x_{\text{rand}}, G_{\text{sub}}$);
5 $u_{\text{best}}, x_{\text{new}}, \text{success} \leftarrow$ CONTROL($x_{\text{near}}, x_{\text{rand}}, G_{\text{sub}}$);
6 if success
7 $G_{\text{sub}}$.add_vertex($x_{\text{new}}$);
8 $G_{\text{sub}}$.add_edge($x_{\text{near}}, x_{\text{new}}, u_{\text{best}}$);
9 Return $G_{\text{sub}}$

Fig. 18.1. The basic RRT construction algorithm

to understand the fundamental insights of these algorithms. After studying an algorithm in both its randomized and derandomized forms, it will be possible to intelligently decide which to use, or whether some mixture of the two is appropriate. This approach might even result in algorithms that combine deterministic and randomized strategies in a way that achieves the benefits of both. For example, [20] uses a combined sampling strategy for constructing a Visibility PRM; they recursively divide the space into quadrants (a deterministic strategy) and choose a random sample within each quadrant. In the area of low-discrepancy sampling, Wang and Hickernell have constructed and analyzed randomized Halton sequences [23]. Geraerts and Overmars have also investigated randomizing Halton points [6].

We believe that a great deal of work remains to investigate deterministic variants of contemporary motion planning algorithms. We have already mentioned efforts to derandomize PRMs; namely, those which attempt to use deterministic uniform sampling methods. It is also interesting to consider derandomizing RRTs; this is a very challenging task, due to the way that randomization is used in RRTs.

18.2 Randomization in RRTs

In the case of RRTs, derandomization is more difficult than with PRMs. In the original PRM, the primary use of randomization is to produce a uniformly distributed sample sequence in order to cover the space; for RRTs, the use of randomization is more subtle. As opposed to the former case, simply replacing random samples with deterministic ones will not capture the essence of the exploration strategy of RRTs. In order to understand the best way to derandomize RRTs, we will outline the role of random sampling in the basic RRT algorithm.

The basic RRT algorithm operates very simply; the overall strategy is to incrementally grow a tree from the initial state to the goal state. The root of the tree is the initial state; at each iteration, a random sample is taken and its nearest neighbor in the tree computed. A new node is then created by growing the nearest neighbor toward the random sample. Pseudocode for basic RRT construction can be found in Figure 18.1; for in-depth description and analysis of RRTs, see [13].

In [13], it is argued that RRTs explore rapidly because samples “pull” the search tree toward unexplored areas of the state space. This occurs because the probability that a vertex is selected for expansion is proportional to the area of its Voronoi region. Hence, a node at the frontier of the tree is likely to be chosen to grow into