Current Issues in Sampling-Based Motion Planning

Stephen R. Lindemann   Steven M. LaValle

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

Abstract

In this paper, we discuss the field of sampling-based motion planning. In contrast to methods that construct boundary representations of configuration space obstacles, sampling-based methods use only information from a collision detector as they search the configuration space. The simplicity of this approach, along with increases in computation power and the development of efficient collision detection algorithms, has resulted in the introduction of a number of powerful motion planning algorithms, capable of solving challenging problems with many degrees of freedom. First, we trace how sampling-based motion planning has developed. We then discuss a variety of important issues for sampling-based motion planning, including uniform and regular sampling, topological issues, and search philosophies. Finally, we address important issues regarding the role of randomization in sampling-based motion planning.

1 Introduction

In recent years, a number of motion planning algorithms have been introduced which have had remarkable success in solving challenging motion planning problems, including ones with many degrees of freedom (DOFs). Examples include the Randomized Path Planner (RPP), Probabilistic Roadmap planners (PRMs), Ariadne's Clew, and Rapidly-exploring Random Trees (RRTs). Each of these methods, and many others, can be seen as belonging to a field we call *sampling-based motion planning*. We believe a fundamental distinction exists between sampling-based motion planners and earlier planners that built explicit representations of the obstacle boundary in the configuration space.

While the problem of motion planning has been recognized for many decades (e.g, Nilsson's work in the late 1960s [56]), it can be argued that the epoch marking the beginning of modern motion planning was the introduction in 1979 of the idea of the *configuration space* by Lozano-Pérez and Wesley [50]. In the configuration space, the robot is reduced to a point; hence, the motion planning problem becomes that of finding a path for a point from an initial point to a goal point in the configuration space. This contrasts with previous methods which planned directly in the workspace, using methods such as swept volumes to determine whether or not a path was feasible (i.e., did not collide with an obstacle). However, planning in C-space poses a problem: unlike the obstacles in the workspace, which are well-defined, how does one represent invalid configurations in C-space (C-space obstacles[1])? Lozano-Pérez and Wesley gave methods for constructing representations of $\mathcal{C}_{obs}$ based on contact conditions between the robot and the obstacles, and their ideas formed the foundation for many motion planning algorithms in the decade to follow.

However, constructing explicit representations of $\mathcal{C}_{obs}$ from the geometry of the problem has several disadvantages. The first complete, general motion planning algorithm, by Schwartz and Sharir, used cylindrical algebraic decomposition, whose running time was doubly exponential in the degrees of freedom [62]. Canny introduced a roadmap algorithm which improved this to "only" singly exponential [17]. Both methods employ general-purpose techniques from computational real algebraic geometry [8], which are very difficult to implement correctly, especially due to numerical considerations. Furthermore, the running times of these algorithms also grow quickly with the number of primitives in the obstacle and robot representations, as opposed to the true difficulty of a particular motion planning task. Most realistic motion planning models require thousands of primitives, which is well beyond what can be handled by the motion planning algorithms that work directly with algebraic constraints on the obstacle region. In some special cases, efficient combinatorial algorithms have been developed, but these usually apply to low-dimensional C-spaces and/or simplified geometries. The PSPACE-hardness bound [61] provides further discouragement from attempting to develop and use exact, combinatorial solutions, such as those in [17, 62], to solve motion planning problems that have many degrees of freedom and model primitives.

In light of these difficulties, sampling-based motion planning has emerged over the past fifteen years as a way to avoid explicit constructions of $\mathcal{C}_{obs}$. By sampling, it is hoped that many practical, high-dimensional problems

[1]The set of all invalid configurations is often denoted $\mathcal{C}_{obs}$ and its complement, the set of all free configurations, as $\mathcal{C}_{free}$

with complicated models can be solved efficiently. Instead of an explicit representation of $\mathcal{C}_{obs}$, imagine that one has access only to limited information about the configuration space. For example, given a test configuration, suppose one can efficiently determine whether the configuration lies in $\mathcal{C}_{free}$ or $\mathcal{C}_{obs}$. Alternatively, one may have access to the distance (typically, Hausdorf distance) between the robot and the workspace obstacles for a given configuration. This information is exactly that which is provided by modern collision detection algorithms. Hence, we define sampling-based motion planners as those whose only information about $\mathcal{C}_{obs}$ is obtained by sampling C-space through a collision detector.

Sampling-based motion planning is thus fundamentally different from earlier approaches to motion planning since its model of available information about C-space is substantially restricted. This restriction eliminates many of the problems encountered in methods that constructed a representation of $\mathcal{C}_{obs}$. Since there is no explicit model of $\mathcal{C}_{obs}$, there is no need to characterize all possible contact conditions for particular classes of problems, nor to compute the contacts to solve a given problem. Also, a sampling-based motion planner can apply to a broad class of motion planning problems because it treats collision detection as a separate module, which may be tailored to a particular kind of problem. For these reasons, sampling-motion planning algorithms often seem strikingly simple in comparison to combinatorial motion planners, such as Canny's roadmap algorithm. The simplicity and generality of these planners is likely a significant factor contributing toward their success and applicability to high-DOF problems.

2 The rise of sampling-based motion planning

To fully understand the continuing evolution of sampling-based motion planning and its current issues, it is helpful to understand how sampling-based algorithms have developed and changed over time. In this section, we will describe how sampling-based algorithms began to emerge, and how they have continued to develop up to the present time.

In the 1980s, constructing a representation of $\mathcal{C}_{obs}$, either completely or in part, was the predominate approach to motion planning. Examples include the planner by Brooks and Lozano-Pérez for a polygon rotating and translating in the plane [15], work by Donald for planning for a 3D rigid body [22, 23], and a planner by Lozano-Pérez for manipulator arms [49]. References to many combinatorial planners and a few early sampling-based ones can be found in Hwang and Ahuja's survey [34]. Glimpses of sampling-based ideas can be seen even in Donald's work; he placed a six-dimensional lattice over the configuration space and attempted to find a connected sequence of lattice points. Lacking the collision detection abstraction, however, he relied on equations representing the $\mathcal{C}_{obs}$ boundaries during the search process. Greater movement toward sampling-based motion planning began in the late 1980s. Algorithms in this direction typically centered around advances in efficient calculation of distance between polyhedra. Faverjon and Tournassoud introduced a manipulator planner which computed local collision-free motions using distance computation and hierarchical CAD models [25, 24]. The introduction of algorithms such as the Gilbert-Johnson-Keerthi algorithm [29] made sampling-based approaches more common. A good example of an approach is the manipulator planner of Paden *et al.* [58]. They create a 2^d-tree representation of the configuration space, labelling cells as "freespace," "obstacle," or "not sure or mixed." To classify cells correctly (or at least, conservatively), they find the uniform bound on the Jacobian for the given manipulator. Then, based on this information and the workspace distance returned by the GJK algorithm, they can determine whether or not an entire cell can be classified as freespace or obstacle. If neither apply, then the cell is labelled mixed and will be subdivided, if a predefined minimum resolution has not yet been reached. After preprocessing the environment in such a way, it is simple to find a path, if one exists in the tree, or to determine that greater resolution is required to resolve small mixed cells.

The use of distance information from a collision detector permits hierarchical grid-based approaches as in Paden *et al.*, but computing this information is more expensive than simply returning the boolean result of an intersection test (the most basic form of collision detection). A less-expensive grid-based approach might discretize the space at a sufficiently fine resolution and use an inexpensive collision detection method to determine whether each cell belongs to $\mathcal{C}_{free}$, thus creating a bitmap of C-space. The resulting data structure can then be searched by classical AI search techniques to find a path, if one exists. In fact, this very approach was taken by Lengyel *et al.* [45]. Their algorithm uses graphics hardware to plan for a polygonal robot translating and rotating in the plane. They divide the rotational degree of freedom, θ, into a number of slices, and use graphics hardware to calculate the Minkowski sum of the robot and obstacles for a particular value of θ. They combine all resulting slices and have a bitmap representation of the three-dimensional C-space, which they then search with a dynamic programming technique.

In general, however, this kind of approach is limited to lower dimensions since the number of resultant grid cells grows exponentially with the number of DOFs of the problem, and the a fine resolution is required. Hence, checking them all for collision becomes impractical. Nevertheless, when general sampling-based motion planning algorithms began to proliferate in the early 90's, several of these were clearly influenced by the grid search approach. We will consider two of this type, along with two other early sampling-based algorithms, before describing several more recent, state-of-the-art sampling-based motion planners.

One early planner that strongly reflects classical grid search techniques is that of Kondo [41]. Kondo's planner is based on the observation that even if a fine grid is placed over the configuration space, it may be possible to find a solution without visiting large portions of that grid. Hence, if one delays collision checking until needed–a "lazy" approach–only (relatively) few collision checks will need to be performed, thus avoiding the expensive preprocessing step of naive grid search. The planner searches a grid bidirectionally, assigning cost $f(C) = g(C) + h(C)$ to each expanded grid cell, in which $g(C)$ is the standard cost-to-come and $h(C)$ is a heuristic weighted sum-of-squares cost. Kondo's planner uses multiple heuristics (i.e., different assignments of the heuristic weight constants), and adaptively selects between them based on an estimate of their effectiveness. Hence, the effectiveness of the planner strongly depends on the quality of the heuristic functions, and on the planner's ability to choose the appropriate one to apply. If either of these are poor, then performance will degrade greatly. Kondo gives several six-dimensional examples, with the resolution of the grid being 2^7 points per axis yielding 2^{42} total grid cells. However, for the results reported, typically less than 20000 collision checks were needed to solve the problem. The influence of Kondo's multiple-heuristic approach can be seen in recent PRM-related work by Isto [35].

In 1990, Barraquand and Latombe introduced the planner that came to be called the Randomized Path Planner [7]. This planner is important for three primary reasons: first, it was the perhaps the first well-known sampling-based motion planner; second, it solved problems with many DOFs, typically many more than other planners at the time were capable of handling; and third, it advocated randomization as a means of efficiently finding solutions in the high-dimensional configuration space. Its influence in this third respect can hardly be overestimated, since for the following decade virtually every significant sampling-based motion planning algorithm used randomization. In fact, only recently has the role of randomization in sampling-based motion planning begun to be studied in depth. We will discuss this issue in some depth in subsequent sections. RPP operates as follows: first, the planner defines several potential fields over a grid imposed on the workspace; each potential field corresponds to a "control point" on the robot. A finer-resolution grid is also defined over the configuration space, and the potential value of each configuration-space grid cell is defined by the following non-negative, real-valued function on C_{free}:

$$U(q) = G\left(U_{p_1}(X(p_1, q)), \ldots, U_{p_n}(X(p_n, q))\right),$$

in which $p_1, \ldots, p_n$ are the control points, X is a function mapping a point on the the robot to its position in the workspace at the given configuration, and G is an arbitration function. Then, beginning at the initial state, the planner descends the gradient of the C-space potential field, until a local minimum is reached. If the minimum is the global minimum, the goal state has been attained; else, the planner executes a series of random walks with the aim of escaping the local minimum. After this, the planner again descends the potential field gradient, continuing this process until the goal state has been reached or a user-specified amount of time has elapsed. This latter condition is necessary because unlike combinatorial planners, sampling-based planners are typically unable to recognize that a problem has no solution; in such a situation, they will never terminate. The key to this planner's performance is the construction of good potential fields and a good arbitration function, which can be quite difficult to construct in practice. If the potential fields result in many local minima, the planner can perform poorly.

Another early sampling-based motion planner is the SANDROS planner of Chen and Hwang [18], which was developed for manipulator arms. This planner searches in a multi-resolution manner over a non-uniform grid (i.e., the resolution on the coordinate axes may differ). The axes are given different resolutions because for manipulator arms, links near the base have the greatest impact on end effector position. Just as Paden *et al.*, this algorithm uses the GJK algorithm [29] for collision detection. It also uses the distance information to place links of the arm at maximal distance from the workspace obstacles.

Finally, a planner (later termed the ZZ-method) was introduced by Glavina in 1990 [30] which foreshadows PRMs in many respects. The ZZ-method first attempts to connect the initial and goal queries using a "straight-and-slide" local planner (a method which does not allow backtracking but is more powerful than the straight-line local planner). If this fails, which is usually the case, then a new configuration is chosen as a subgoal (Glavina advocates using jittered sampling), and attempts to connect the subgoal to the initial and goal configurations using the same local planner. If this fails, new subgoals are added and attempts are made to connect them with previously existing subgoals, as well as the initial and goal configurations. Edges between subgoals are checked for collisions at a pre-defined subsampling resolution. Glavina also identifies the well-known "narrow corridor" problem and uses connected component analysis to speed up his planner. However, he uses a primitive collision detection method which prevents him from applying his algorithm to challenging, high-DOF problems (this was remedied in some extensions of his work [4, 5]); also, the straight-and-slide local planner becomes expensive in high dimensions. In principle, however, the ZZ-method contains many elements which have become common in more recent algorithms.

Since the introduction of these early algorithms, sampling-based motion planning has continued to develop. Changes have been made to deal with failings of previous planners, and new exploration paradigms have been investigated. We discuss four well-known recent motion planning algorithms: PRMs, Ariadne's Clew, the expansive-space planner by Hsu *et al.*, and RRTs.

In recent years, the most popular paradigm for sampling-based motion planning has been the probabilistic roadmap [38]. The original PRM, along with its numerous extensions and variants (e.g., [1, 11, 46, 60, 63, 69, 70]), have been successfully applied to problems in robotics, computer animation, and computational biology [40, 59, 65]. While there is a strong connection to Glavina's work, there are several important differences. Foremost among these is that the PRM is designed for multiple-queries rather than a single-query. Hence, the placement of landmarks is seen as constructing a reusable roadmap in the PRM method, not as generating query subgoals as in the ZZ-method. Second, the ZZ-method attempted to connect each new landmark (subgoal) to all previous ones; PRMs attempt to connect to a more carefully-chosen subset of these, which is typically the K nearest landmarks from each connected component, or all subgoals within some specified radius. Third, the PRM uses a more simple local planner, often either straight-line or rotate-at-s [3], unlike the ZZ-method's more expensive straight-and-slide local planner. Finally, methods are used to identify difficult regions of C-space and sample in those regions (the "roadmap enhancement" phase). Along with the use of more sophisticated collision detection methods, these factors make the PRM more effective for challenging motion planning problems.

Ariadne's Clew is a single-query algorithm that grows a tree from the initial configuration toward the goal configuration [52, 53]. At each step, it searches for a new "landmark," reachable from a current landmark by a Manhattan path of a certain order, which is maximally distant from the set of all current landmarks. They use highly-parellelized genetic algorithms to search for a solution to this optimization problem. Once a new landmark has been added to the tree, the planner attempts to connect this new landmark to the goal. To improve performance, when the algorithm encounters an obstacle in trajectory calculation it "bounces" off it. Experimental results give fast solution times for motion of a 6-DOF arm in a dynamic environment. One limitation, however, is the difficult heuristic choices required for the genetic algorithm.

Hsu *et al.* introduced a single-query path planner[2] for "expansive" configuration spaces in [33]. The notion of expansiveness is related to how much of the free space is visible from a single free configuration or connected set of free configurations, and extends the idea of ϵ-goodness [6]. The expansive-space planner grows a tree from the initial configuration. Each node x in the tree has an associated weight, which is defined to be the number of nodes inside $N_d(x)$, the ball of radius d centered at x. At each iteration, it picks a node to extend; the probability that a given node x will be selected is $1/w(x)$, in which w is the weight function. Then, K points are sampled from $N_d(x)$ for the selected node x, and the weight function value for each is calculated. Each new point y is retained with probability $1/w(y)$, and the planner attempts to connect each retained point to the node x. Hence, we see a similarity between this planner and Ariadne's clew, in that they each try to "push" the tree toward unexplored areas of free space. The main drawback of the approach is that the required d and K parameters may vary dramatically across problems, and they are difficult to estimate for a given problem.

Finally, we describe Rapidly-exploring Random Trees (RRTs) [42, 44], which were developed for problems with differential constraints, such as kinodynamic planning and nonholonomic planning. Its introduction has stimulated a flurry of recent applications and extensions (e.g., [13, 16, 19, 20, 21, 27, 36, 37, 39, 47, 67, 70]). In its basic form, the RRT attempts to grow a tree from the initial configuration to the goal configuration as follows: take a random sample, and find its nearest neighbor in the search tree. Then, grow toward the sample from its nearest neighbor. This process is repeated until the initial and goal configurations are connected. The best-performing RRT planner uses a more greedy connection strategy (at each iteration, attempt to make a complete connection from the nearest neighbor to the sample) and searches bidirectionally. This planner rapidly explores the configuration space because it is Voronoi-biased: at each iteration it tends to grow from the node with the largest Voronoi area. This is because the probability that a node is selected for expansion is directly proportional to the volume of its Voronoi cell. In contrast to Ariadne's Clew and the expansive-space planner, which work hard to push the tree toward unexplored regions, RRTs are *pulled* into these regions by virtue of the sampling and connection strategy. This avoids the need for complicated parameter tuning, but comes at the expense of performing nearest neighbor queries.

3 Uniform sampling issues

All of the recent methods from Section 2 rely on some method for generating samples over the configuration space. This section covers ways to sample uniformly, which applies many methods, including the ZZ-method, PRMs and RRTs. Others, such as RPP and the expansive-space planner, sample uniformly over neighborhoods of certain points. Typically, the samples are taken at random from a statistically uniform distribution; however, this method of sampling is not as uniform as some deterministic methods. Several weaknesses of random sampling were shown in the context of the PRM in [14, 28, 43]. One problem is that it is not optimal with respect to rigorous criteria, such as those introduced in Section 3.1. In this section we discuss some of these general issues, and defer discussion of non-uniform sampling (or importance sampling) until Section 4.1.

3.1 Sampling criteria

A very brief overview is given here; more details and references appear in [43]. Motivated by applications in numerical integration and optimization, the most common criteria are discrepancy and dispersion. Let $X = [0,1]^d \subset \mathbb{R}^d$ define a space over which to generate samples. Consider evaluating the uniformity of a set, P,

[2]Some authors refer to this and virtually all planning algorithms that use randomization as PRMs. To avoid confusion, we do not use this term for single-query planners, such as the planner of Hsu *et al.*, even though it is called a PRM by its authors.

of N d-dimensional sample points $\{p_0, \ldots, p_{N-1}\}$. The *discrepancy* is defined as

$$D(P, \mathcal{R}) = \sup_{R \in \mathcal{R}} |\mu(R) - |P \cap R|/N| \qquad (1)$$

in which $|\cdot|$ of a set denotes its cardinality, μ denotes the Lebesgue measure, and $\mathcal{R}$ is a *range space*, which will be taken by default in this paper to be the set of all axis-aligned rectangular subsets of X. The *dispersion* is defined as

$$\delta(P, \rho) = \sup_{x \in X} \min_{p \in P} \rho(x, p), \qquad (2)$$

in which ρ denotes any metric; unless otherwise stated, the default metric in this paper will be ℓ^∞. Dispersion can also be considered as the "radius" of the largest empty ρ-ball, among all balls whose centers lie in X. Both discrepancy and dispersion seem very relevant in the PRM context because range queries are repeatedly performed, and these criteria ensure that either the appropriate number of samples (discrepancy) or at least one sample (dispersion) will fall within a range.

A *low-discrepancy* point set or sequence is one that yields the best-possible asymptotic discrepancy, which is $O(N^{-1} \log^d N)$ for infinite sequences and $O(N^{-1} \log^{d-1} N)$ for finite point sets. The simplest low-discrepancy point sets and sequences are Hammersley and Halton points, respectively, which were applied to motion planning in [14]. Other low-discrepancy techniques exist that produce smaller constants in the asymptotic convergence rate. The best family of methods are the (t,m,s)-nets and (t,s)-sequences [54], and the current best within this family are the Niederreiter-Xing sequences [55].

Regarding ℓ^∞ dispersion, the *Sukharev sampling criterion* [66] states that for any point set P, $\delta(P) \geq \frac{1}{2}\lfloor N^{\frac{1}{d}} \rfloor$. Thus, to keep dispersion fixed, it is impossible to avoid exponentially-many samples in dimension. A *low-dispersion* point set or sequence is one that produces the best possible asymptotic dispersion, which is $O(N^{-1/d})$. For a fixed N, if $N^{\frac{1}{d}}$ is an integer, k, the *Sukharev grid* yields the best possible dispersion, which is precisely $\frac{1}{2}N^{-1/d}$. In this case, the grid is constructed by partitioning $[0,1]^d$ into N cubes of width $1/k$ so that a tiling of $k \times k \times \cdots \times k$ is obtained, and a sample is placed at the center of each cube. Nongrid, low-dispersion infinite sequences exist that have $\frac{1}{\ln 4}$ as the constant in the asymptotic convergence rate [54].

If a sample sequence is used that has a tight upper bound on dispersion, resolution completeness guarantees can be made on most sampling-based planning algorithms, expressed in terms of corridor width [43]. Using a random sequence, one typically bounds expected performance or constructs high-probability bounds [6]. It is possible, however, to provide both kind of bounds, if desired, by randomizing low-dispersion sequences.

3.2 Lattices and other regular structures

Regular structures often have desirable uniformity characteristics. For example, the Sukharev grid is optimal for ℓ^∞-dispersion [66], and low-discrepancy lattices have been a subject of interest in the sampling community for some time [64]. In addition to good uniformity, regular sampling can have additional benefits. For example, regular structures have an implicit neighborhood structure: given a vertex in the structure, one may easily find its neighbors. Additionally, regular structures often admit hierarchical or multi-resolution representations, which can also be used advantageously for planning purposes (e.g., Paden *et al.* [58]). However, regular structures suffer from two problems: first, they are point *sets*, not point *sequences*; and second, they are axis-aligned. We discuss each of these in turn.

Point sets of a fixed size are generally unsuitable for motion planning purposes becuase one cannot determine *a priori* how many samples will be necessary to solve the problem (and hence how large the point set needs to be). Hence, regular structures seem to be at a disadvantage as opposed to infinite sequences such as Halton points or uniform random samples. However, it is possible to construct infinite sequences based on regular structures, which incrementally enhance their resolution. Sequences of this type periodically look like the point sets they are based on (at particular resolution levels), and gradually "fill in the gaps" between one resolution level and the next. The foremost requirement of sequences of this type is that it have incremental quality. This means that after every sample (or, perhaps, each small set of samples) the sequence should be as uniform as possible. Secondarily, these sequences should be easy to generate and allow easy neighbor-finding, to fully exploit the advantages of regular structures. Examples of infinite sequences based on regular structures include the extensible grid sequence of the authors [48] and the extensible low-discrepancy lattices of Hickernell *et al.* [31]. The derandomized version of the Lazy PRM (see [11]) also progressively increases the resolution of the grid used to build the roadmap [10]. Rather than adding points one at a time as in the previous methods, Bohlin adds an entire hyperplane of samples, chosen to fill the largest gap present in the existing grid.

By their nature, regular structures are axis-aligned. In the case of grids, the structures are aligned with the coordinate axes; other regular structures have alignments on other axes. The effect of this on motion planning algorithms should be understood. To date, only limited efforts have been undertaken to examine the effect of these axis alignments [43]. It may be observed from the outset that the effect of axis alignments is greatest when $\mathcal{C}_{obs}$ also has axis alignments. While this should not be the case in general, it often happens when the workspace environment is human-designed. When this is the case, small changes in workspace obstacle position can move an entire hyperplane of samples from $\mathcal{C}_{free}$ to $\mathcal{C}_{obs}$, greatly increasing the number of samples required to solve the problem and thus its (apparent) difficulty. This is undesirable, since it is intuitively clear that, in general, changing the workspace slightly should not greatly change the difficulty of the problem. Several comments on this front can be made. First, it has already been noted that this problem is more practical than theoretical, since it probably occurs most often when the workspace is human-designed. Second, axis alignments may *help* for some problems, and hurt for others. Thus, axis alignments may not be bad in an absolute sense (such as increasing execution time across the board), but may simply result in higher "variance" in execution times across a set of similar problems. Third, it may be possible to reduce this variance through limited use of randomization (e.g., choosing a random start index for the sample sequence from a set of possibilities).

3.3 Topological considerations

In the discussion so far, little attention has been given to the interaction between topology and sampling issues. For example, most QMC literature is dedicated to sampling in an n-dimensional cube, which does not reflect the topology of most configuration spaces. Special concern must be given for samples near the boundary of the cube, but if all robot motions are obtained from $360°$ revolute joints, the configuration space is a manifold without boundary (i.e., a torus). A sampling technique that was optimized for a cube might perform well on a torus, but better techniques might still exist because the boundary effects are different. This becomes very important in high dimensions. For example, for a lattice, there will be very few points per axis. It would be unfortunate to place points close to what would be the boundary of the cube, when in fact the opposite faces are identified, and there is no boundary.

It is also important to note that sampling issues are affected by the particular parameterization used to represent the C-space. A homeomorphism can always be introduced that "enlarges" one part of the space, while "shrinking" others. Thus, which parameterization would accurately preserve the notion of uniformity? In many cases, an intrinsic notion of uniformity exists. For any locally compact topological group, there exists a unique (up to scale) measure, called the *Haar measure*, that is invariant with respect to group actions [26]. For $SO(2)$, this is achieved by standard Lebesque measure on the interval $[0, 2\pi)$. For $SO(3)$ this is obtained by defining a uniform density function over the upper half of the unit sphere, $S^3 \subset \mathbb{R}^4$, which corresponds to the set of unit quaternions in which the first coordinate is positive. Thus, for uniform, random sampling, it is best to generate samples uniformly on S^3. Optimal dispersion sampling, however, remains a challenging problem for SO(3) and other transformation groups that arise in motion planning.

In addition to uniformly sampling the entire configuration space, operations are frequently used which require uniformity over local directions or neighborhoods, which yields new topologies. For example, Barraquand and Latombe sample the n-neighbors of the current grid cell for the purposes of gradient descent. Glavina chooses several direction vectors to use in the straight-and-slide local planning method. Wilmarth *et al.* choose k directions in which to try to push a sample toward the medial axis of free space. In the same way as in the general case, it is possible to obtain more uniform results with deterministic techniques than with simple random ones. For example, one may precompute k points placed with high or optimal uniformity on S^{d-1}, corresponding to direction vectors in d-dimensional space. It would require some work to compute these vectors for different values of d and k, and to do so is non-trivial; however, these only need to be computed a single time, and can then be stored for later use. To use the computed sample sets, one need only apply a rotation to each one to eliminate bias, possibly selected at random or from a uniform deterministic sequence.

4 Search and exploration issues

Complementary to uniform sampling is the problem of how to explore the configuration space, which determines the way in which samples are used. We discuss several issues related to search in this section.

4.1 Non-uniform sampling

Having discussed some of the issues related to uniform sampling, we have argued that deterministic techniques may be beneficial compared to the randomized methods which have thus far dominated sampling-based motion planning. However, it is much more challenging to address these issues in the context of *non*-uniform sampling. The primary motivation for non-uniform sampling is simple: if it is possible to determine that certain regions of the configuration space are more important than others, then one would like to be able to sample these at a higher density, as efficiently as possible. The importance of generating samples around narrow corridors was recognized in [2, 30, 32].

In general, there are two approaches to non-uniform sampling: importance sampling (*a priori* non-uniform sampling) and adaptive sampling. Importance sampling is based on the prior belief that solutions will be found more quickly by concentrating sampling in certain areas of C-space. Importance sampling has played a significant role in recent PRM literature; a simple example is the technique of goal-biased sampling [65], in which samples are drawn from Gaussian distributions centered at the goal configurations in addition to some uniform sampling. Other examples of importance sampling used in PRMs include obstacle-based sampling, in which samples are taken from the boundary of $\mathcal{C}_{obs}$ [1], medial-axis sampling, in which samples are taken from the medial axis of $\mathcal{C}_{free}$ [69], and Gaussian sampling, in which sampling is biased to be near C-space obstacles [12]. Given the limited information about C-space to which sampling-based planners have access, it is not immediately clear how to sample this way. The way this problem has been dealt with is to draw samples uniformly from the configuration space, and then apply a transformation or rejection rule to them to achieve the desired distribution. We define a transformation rule as a rule which takes uniform samples and transforms them to have the desired characteristics. Medial-axis sampling implements a transformation rule, and obstacle-based sampling can be implemented in this way as well.

The other category of non-uniform sampling is adaptive sampling. In this technique, the distribution from which new samples are drawn is modified based on information gained from previous samples. An example of adaptive sampling is the Visibility PRM [63]. This algorithm adapts its sampling-based on the visibility regions of the different connected components of the computed roadmap. By sampling only in unexplored regions (in the visibility region of no component) and in regions where connections between different components can be made (in the visibility region of more than one component), good exploration can be achieved without creating large numbers of nodes in the roadmap (as in most PRM methods). However, since no explicit information about the visibility regions is available to the planner, it takes samples uniformly from the configuration space and applies a rejection rule to them. A sample is accepted if it satisfies one of the criteria given above, and rejected if it does not.

As we have seen, both importance and adaptive sampling techniques often implicitly depend on uniform sampling, due to the lack of C-space information inherent in the sampling-based approach. Hence, utilizing high-quality uniform sampling techniques (including deterministic ones) is not only important in a general sense, but also impacts algorithms whose performance hinges on non-uniform sampling.

4.2 Single-query vs. multiple-query

An important distinction between single-query and multiple-query planning was made in [38]. For earlier sampling based methods, such as the ZZ-method, Ariadne's clew, or RPP, searching for the solution to a query was the main focus. The PRM was introduced as a precomputed structure that could be used to quickly answer many queries for the same set of obstacles. The combinatorial methods, such as Canny's roadmap or cylindrical algebraic decomposition, also follow this philosophy because the searching part is straightforward, once substantial effort has been invested in building a roadmap that captures the connectivity of $\mathcal{C}_{free}$. Many recent sampling-based planning works, including the RRT and the expansive C-space planner, have returned to the single query model, particularly for the most challenging problems. A single-query version of the PRM was even introduced in [11]. Efforts have also been made to build multiple-query data structures using single-query primitives [9].

There are tradeoffs between investing substantial time in precomputation vs. immediately attempting to solve a query. The investment is worthwhile only if the particular application yields numerous planning queries for the same environment. One important middle ground that deserves more research attention is how to incrementally build a data structure that becomes faster as more and more queries are given. Initially, a single query approach could be used, and then some part of the search structure is saved for future use. As more queries are given, the structure could be updated. After numerous queries, it would be ideal if the structure has properties like the Visibility PRM: very few nodes, but most future queries can be answered very quickly.

4.3 Information model

One interesting concept that has involved throughout sampling-based planning is the model of information that is available to a planner. Initially, planners worked directly with C-space constraints. As collision detections algorithms became more powerful, planners were given less access to information regarding the constraints. A collision detector essentially serve as a “black box” that reports collision or possibly yields distance information. This has greatly helped the development of sampling-based planners that can be applied to a broad class of problems. However, in many instances, it may be possible to improve performance by carefully investigating the constraints that arise for particular problems once again. It may be possible to optimize performance of some of the sampling-based planners in particular contexts by carefully considering what information is available directly from the C-space constraints. This would make an interesting direction explore in future research.

4.4 Simplicity and use of heuristics

One welcome trend in sampling-based planning has been a reduction of heuristic parameters that require tuning. In spite of the success of RRP and other earlier planners, many of their ideas were abandoned in recent research because of this difficulty. The original PRM, and some of its variants, such as the Visibility PRM, have become very popular because predictable performance is obtained with little parameter tuning. The main problematic parameter with the PRM is the connection radius (aside from heuristics involved in node enhancement). The RRT has achieved success in recent years because there are no parameters that require tuning for standard path planning (there are metric issues when differential constraints exist), except the collision detection frequency, which is required of all sampling-based algorithms. As more methods are developed, it is hoped that the dependency of solutions on particular parameter settings will be minimized.

5 The value of randomization?

To one degree or another, randomization has been ubiquitous in the field of sampling-based motion planning since Barraquand and Latombe's work on the RPP. They cite successful applications of randomization to NP-hard problems as motivation for their use of randomization as a tool for motion planning, and in turn their planner is highly randomized. Other planners are randomized only through their use of a random sample sequence (e.g., PRMs, RRTs). Since randomized planners have exhibited great success in solving challenging motion planning problems, it is tempting to attribute this success to the "power of randomization." However, little work has been done to carefully articulate the benefits of various uses of randomization in sampling-based motion planning algorithms. We believe that these issues deserve careful study, and doing so will greatly contribute to depth of understanding of sampling-based motion planning in general.

An inescapable feature of randomized algorithms is their lack of repeatability: no two runs will execute identically. In motion planning, this has both positive and negative implications. It is positive in that sometimes the randomized algorithm will be "lucky" and solve a problem very quickly. Hence, if it takes a long time to solve a particular problem, there is hope that it will do better next time; for a deterministic algorithm, this is not the case. If a deterministic algorithm performs poorly once, it will always perform poorly for that problem. On the other hand, the lack of repeatability caused by randomization can easily obscure how well the algorithm performs and can cause flaws to be overlooked. In our experience, flaws in deterministic algorithms often can be discovered quickly because a single execution may be enough to reveal them. Hence, working with deterministic algorithms can result in greater carefulness in both algorithm design and implementation. Also, greater understanding of high-level algorithmic operation is possible since there is no random noise in its performance and operation.

A disadvantage of deterministic sampling and the design of deterministic algorithms is that more caution must be exercised to ensure correct behavior. For example, the fact that deterministic samples are correlated requires that extra attention be paid to how these samples are used. Strange results can occur when patterns in the deterministic sequence interact with the behavior of the algorithm[3]. As a simple example, consider a bidirectional RRT which alternates between growing the two search trees, using Halton points as the input sample sequence. In their typical construction, every even-indexed Halton point has its first coordinate $< 1/2$, and every odd-indexed one $\geq 1/2$; hence, one search tree will always remain in one half of the space, and the second tree in the other half. This is clearly undesirable. This can be fixed with little difficulty; either each tree can use its own sample sequence or one can use a base other than 2 for the first coordinate. Related to this is the observation that one never needs more than one random sample source to supply random numbers to an algorithm. Since each sample is uncorrelated, different portions of the algorithm can all draw random samples from the same place. In the case of deterministic samples, different parts of the algorithm need to instantiate their own deterministic sample generators, to avoid interference with one another and to ensure that the sampling for each one truly is uniform.

[3]Note that the same difficulty sometimes occurs with pseudo-random numbers, which are used instead of truly random numbers.

The best approach may be to use randomized versions uniform point sequences discussed in Section 3. These sequences combine some of the benefits of both deterministic and random sampling, and allow both deterministic and randomized performance guarantees. Several different randomized Halton sequences are possible: one may choose a random shift vector to add to each point, one may use random digit-scrambling techniques [51], or the easily-generated "random-start" Halton sequence by Wang and Hickernell [68]. These sequences all have the uniformity properties of the deterministic Halton sequence, but are randomized as well. Similarly, (t, m, s)-nets and (t, s)-sequences can also be randomized [57]. Several types of random digit scrambling methods are effective; some are fairly inexpensive, and others are more costly to compute.

It has been seen that there are benefits to be gained from examining deterministic alternatives to the randomized approaches so prevalent in sampling-based motion planning. We believe that the achievements of sampling-based motion planning algorithms over earlier combinatorial ones are primarily due to the fact that the are *sampling-based*, not due to the fact that they are usually randomized (which we regard as partly incidental). In fact, the emphasis on and use of randomization may have resulted in less understanding of key issues in motion planning and search. Hence, we believe that new advances in motion planning will occur as a result of careful study of these issues, not through creative uses of randomization. However, there are genuine advantages to some forms of randomization, and these should be embraced. It is possible to design good search methodologies with limited and appropriate randomization.

6 Conclusion

In conclusion, we have overviewed the field of sampling-based motion planning. We have defined sampling-based motion planning and given an overview of its history. We have also discussed key sampling issues for these motion planners, and mentioned areas for future research. We believe that the key to the success of contemporary motion planning algorithms is not through randomization or clever heuristics. Rather, it is the fact that these algorithms are sampling-based, consequently avoiding the complexities of building $\mathcal{C}_{obs}$ representations, which many earlier planners were faced with. This enabled general-purpose planning algorithm to be developed, while relegating the problem-specific difficulties of analyzing $\mathcal{C}_{obs}$ to collision detection algorithms. In future research, it might be beneficial to investigate ways to improve recent sampling-based planning algorithms in particular contexts by once again considering the interaction between the obstacle constraints and the planning algorithm.

Acknowledgement

We thank Pekka Isto for bringing Glavina's work to our attention. We are grateful for the funding provided in part by NSF awards 9875304, 0118146, and 0208891.

References

[1] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Proceedings of the Workshop on Algorithmic Foundations of Robotics*, pages 155–168, 1998.

[2] N. M. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. In *IEEE Int. Conf. Robot. & Autom.*, pages 113–120, 1996.

[3] N.M. Amato, O.B. Bayazit, L.K. Dale, C. Jones, and D. Vallejo. Choosing good distance metrics and local planners for probabilistic roadmap methods. *IEEE Trans. Robot. & Autom.*, 16(4):442–447, Aug 2000.

[4] B. Baginski. The Z^3 method for fast path planning in dynamic environments. In *Proceedings of IASTED Conference on Applications of Control and Robotics*, pages 47–52, 1996.

[5] B. Baginski. *Motion Planning for Manipulators with Many Degrees of Freedom - The BB-Method*. PhD thesis, Technische Universität München, 1998.

[6] J. Barraquand, L. Kavraki, J.-C. Latombe, T.-Y. Li, R. Motwani, and P. Raghavan. A random sampling scheme for robot path planning. In G. Giralt and G. Hirzinger, editors, *Proc. of the 7th International Symposium on Robotics Research*, pages 249–264. Springer, New York, NY, 1996.

[7] J. Barraquand and J.-C. Latombe. A Monte-Carlo algorithm for path planning with many degrees of freedom. In *IEEE Int. Conf. Robot. & Autom.*, pages 1712–1717, 1990.

[8] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer-Verlag, Berlin, 2003.

[9] K. E. Bekris, B. Y. Chen, A. M. Ladd, E. Plakue, and L. E. Kavraki. Multiple query probabilistic roadmap planners using single query primitives. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2003.

[10] R. Bohlin. Path planning in practice; lazy evaluation on a multi-resolution grid. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2001.

[11] R. Bohlin and L. Kavraki. Path planning using Lazy PRM. In *IEEE Int. Conf. Robot. & Autom.*, 2000.

[12] V. Boor, N. H. Overmars, and A. F. van der Stappen. The gaussian sampling strategy for probabilistic roadmap planners. In *IEEE Int. Conf. Robot. & Autom.*, pages 1018–1023, 1999.

[13] M. S. Branicky and M. M. Curtiss. Nonlinear and hybrid control via RRTs. In *Proc. Intl. Symp. on Mathematical Theory of Networks and Systems*, 2002.

[14] M. S. Branicky, S. M. LaValle, K. Olson, and L. Yang. Quasi-randomized path planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 1481–1487, 2001.

[15] R. A. Brooks and T. Lozano-Pérez. A subdivision algorithm in configuration space for findpath with rotation. *IEEE Trans. Syst., Man, Cybern.*, SMC-15(2):224–233, 1985.

[16] J. Bruce and M. Veloso. Real-time randomized path planning for robot navigation. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2002.

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

[18] P.C. Chen and Y.K. Hwang. Sandros: A motion planner with performance proportional to task difficulty. In *Proc. of IEEE Int. Conf. Robotics and Automation*, pages 2346–2353, Nice, France, 1992.

[19] P. Cheng and S. M. LaValle. Resolution complete rapidly-exploring random trees. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 267–272, 2002.

[20] P. Choudhury and K. Lynch. Trajectory planning for second-order underactuated mechanical systems in presence of obstacles. In *Proceedings of the Workshop on Algorithmic Foundations of Robotics*, 2002.

[21] M. J. de Smith. *Distance and Path: The Development, Interpretation and Application of Distance Measurement in Mapping and Modelling*. PhD thesis, University College, University of London, London, 2003.

[22] B. R. Donald. Motion planning with six degrees of freedom. Technical Report AI-TR-791, Artificial Intelligence Lab., Massachusetts Institute of Technology, 1984.

[23] B. R. Donald. A search algorithm for motion planning with six degrees of freedom. *Artif. Intell.*, 31:295–353, 1987.

[24] B. Faverjon. Hierarchical object models for efficient anti-collision algorithms. In *IEEE Int. Conf. Robot. & Autom.*, pages 333–340, 1989.

[25] B. Faverjon and P. Tournassoud. A local based method for path planning of manipulators with a high number of degrees of freedom. In *IEEE Int. Conf. Robot. & Autom.*, pages 1152–1159, 1987.

[26] G. B. Folland. *Real Analysis: Modern Techniques and Their Applications*. Wiley, New York, 1984.

[27] E. Frazzoli, M. A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance and Control*, 25(1):116–129, 2002.

[28] R. Geraerts and M. H. Overmars. A comparative study of probabilistic roadmap planners. In *Proc. Workshop on the Algorithmic Foundations of Robotics*, December 2002.

[29] E. G. Gilbert, D. W. Johnson, and S. S. Keerth. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE J. of Robot. & Autom.*, RA-4(2):193–203, Apr 1988.

[30] B. Glavina. Solving findpath by combination of goal-directed and randomized search. In *IEEE Int. Conf. Robot. & Autom.*, pages 1718–1723, May 1990.

[31] F. J. Hickernell, H. S. Hong, P. L'Ecuyer, and C. Lemieux. Extensible lattice sequences for quasi-Monte Carlo quadrature. *SIAM Journal on Scientific Computing*, 22:1117–1138, 2000.

[32] D. Hsu, L. E. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. In et al. P. Agarwal, editor, *Robotics: The Algorithmic Perspective*, pages 141–154. A.K. Peters, Wellesley, MA, 1998.

[33] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *Int. J. Comput. Geom. & Appl.*, 4:495–512, 1999.

[34] Y. K. Hwang and N. Ahuja. A potential field approach to path planning. *IEEE Trans. Robot. & Autom.*, 8(1):23–32, February 1992.

[35] P. Isto. Constructing probabilistic roadmaps with powerful local planning and path optimization. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, pages 2323–2328, 2002.

[36] S. Kagami, J. Kuffner, K. Nishiwaki, and K. Okada M. Inaba. Humanoid arm motion planning using stereo vision and RRT search. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2003.

[37] M. Kallmann, A. Aubel, T. Abaci, and D. Thalmann. Planning collision-free reaching motions for interative object manipulation and grasping. *Eurographics*, 22(3), 2003.

[38] 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.*, 12(4):566–580, June 1996.

[39] J. Kim and J. P. Ostrowski. Motion planning of aerial robot using rapidly-exploring random trees with dynamic constraints. In *IEEE Int. Conf. Robot. & Autom.*, 2003.

[40] Y. Koga, K. Kondo, J. Kuffner, and J.-C. Latombe. Planning motions with intentions. *Computer Graphics (SIGGRAPH'94)*, pages 395–408, 1994.

[41] K. Kondo. Motion planning with six degrees of freedom by multistrategic bidirectional heuristic free-space enumeration. *IEEE Trans. Robot. & Autom.*, 7(3):267–277, 1991.

[42] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. TR 98-11, Computer Science Dept., Iowa State University, Oct. 1998.

[43] S. M. LaValle, M. S. Branicky, and S. R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *International Journal of Robotics Research (to appear)*, 24, 2004.

[44] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In B. R. Donald, K. M. Lynch, and D. Rus, editors, *Algorithmic and Computational Robotics: New Directions*, pages 293–308. A K Peters, Wellesley, MA, 2001.

[45] J. Lengyel, M. Reichert, B. R. Donald, and D. P. Greenberg. Real-time robot motion planning using rasterizing computer graphics hardware. *Computer Graphics*, 24(4):327–335, August 1990.

[46] P. Leven and S. Hutchinson. Real-time motion planning in changing environments. In *Proc. International Symposium on Robotics Research*, 2000.

[47] T.-Y. Li and Y.-C. Shie. An incremental learning approach to motion planning with roadmap management. In *IEEE Int. Conf. Robot. & Autom.*, 2002.

[48] S. R. Lindemann and S. M. LaValle. Incremental low-discrepancy lattice methods for motion planning. In *Proc. IEEE International Conference on Robotics and Automation*, pages 2920–2927, 2003.

[49] T. Lozano-Pérez. A simple motion-planning algorithm for general robot manipulators. *IEEE J. of Robot. & Autom.*, RA-3(3):224–238, Jun 1987.

[50] T. Lozano-Pérez and M. A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979.

[51] J. Matousek. *Geometric Discrepancy*. Springer-Verlag, Berlin, 1999.

[52] E. Mazer, J. M. Ahuactzin, and P. Bessière. The Ariadne's clew algorithm. *J. Artificial Intell. Res.*, 9:295–316, November 1998.

[53] E. Mazer, G. Talbi, J. M. Ahuactzin, and P. Bessière. The Ariadne's clew algorithm. In *Proc. Int. Conf. of Society of Adaptive Behavior*, Honolulu, 1992.

[54] H. Niederreiter. *Random Number Generation and Quasi-Monte-Carlo Methods*. Society for Industrial and Applied Mathematics, Philadelphia, USA, 1992.

[55] H. Niederreiter and C. P. Xing. Nets, (t,s)-sequences, and algebraic geometry. In P. Hellekalek and G. Larcher, editors, *Random and Quasi-Random Point Sets, Lecture Notes in Statistics, Vol. 138*, pages 267–302. Springer-Verlag, Berlin, 1998.

[56] N. J. Nilsson. Shakey the robot. Technical Report TR 223, SRI International, 1984.

[57] A. B. Owen. Monte Carlo variance of scrambled equidistribution quadrature. *SIAM J. Numer. Anal.*, 34(5):1884–1910, 1997.

[58] B. Paden, A. Mees, and M. Fisher. Path planning using a Jacobian-based freespace generation algorithm. In *IEEE Int. Conf. Robot. & Autom.*, pages 1732–1737, 1989.

[59] J. Pettré, J.-P. Laumond, and T. Siméon. A 2-stages locomotion planner for digital actors. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, pages 258–264, 2003.

[60] C. Pisula, K. Hoff, M. Lin, and D. Manoch. Randomized path planning for a rigid body based on hardware accelerated Voronoi sampling. In *Proc. Workshop on Algorithmic Foundation of Robotics*, 2000.

[61] J. H. Reif. Complexity of the mover's problem and generalizations. In *Proc. of IEEE Symp. on Foundat. of Comp. Sci.*, pages 421–427, 1979.

[62] J. T. Schwartz and M. Sharir. On the piano movers' problem: II. General techniqies for computing topological properties of algebraic manifolds. *Communications on Pure and Applied Mathematics*, 36:345–398, 1983.

[63] T. Simeon, J.-P. Laumond., and C. Nissoux. Visibility based probabilistic roadmaps for motion planning. *Advanced Robotics Journal*, 14(6), 2000.

[64] I. H. Sloan and S. Joe. *Lattice Methods for Multiple Integration*. Oxford Science Publications, Englewood Cliffs, NJ, 1990.

[65] G. Song and N. M. Amato. Using motion planning to study protein folding pathways. *Journal of Computational Biology*, 9(2):149–168, 2002.

[66] A. G. Sukharev. Optimal strategies of the search for an extremum. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 11(4), 1971. Translated from Russian, *Zh. Vychisl. Mat. i Mat. Fiz.*, 11, 4, 910-924, 1971.

[67] C. Urmson and R. Simmons. Approaches for heuristically biasing RRT growth. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2003.

[68] X. Wang and F. J. Hickernell. Randomized Halton sequences. *Math. Comp. Modelling*, 32:887–899, 2000.

[69] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *IEEE Int. Conf. Robot. & Autom.*, pages 1024–1031, 1999.

[70] J. Yakey, S. M. LaValle, and L. E. Kavraki. Randomized path planning for linkages with closed kinematic chains. *IEEE Transactions on Robotics and Automation*, 17(6):951–958, December 2001.