Proceedings of the 2003 IEEE
International Conference on Robotics & Automation
Taipei, Taiwan, September 14-19, 2003

Incremental Low-Discrepancy Lattice Methods for Motion Planning

Stephen R. Lindemann Steven M. LaValle

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

Abstract

We present deterministic sequences for use in sampling-based approaches to motion planning. They simultaneously combine the qualities found in many other sequences: i) the incremental and self-avoiding tendencies of pseudo-random sequences, ii) the lattice structure provided by multiresolution grids, and iii) low-discrepancy and low-dispersion measures of uniformity provided by quasi-random sequences. The resulting sequences can be considered as multiresolution grids in which points may be added one at a time, while satisfying the sampling qualities at each iteration. An efficient, recursive algorithm for generating the sequences is presented and implemented. Early experiments show promising performance by using the samples in search algorithms to solve motion planning problems.

1 Introduction

Sampling the configuration space has been one of the fundamental issues in developing practical motion planners. Some approaches use context-specific heuristics to concentrate samples in critical places [1, 4, 8, 19]. Classical grid-search approaches (see survey in [9]) and also recent "lazy" approaches [3, 2, 5, 14]) have focused more on how to sample the configuration space before taking obstacles into account. In this paper, we consider sampling issues from this perspective, and ask: What is the best way to sample the space?

Here are several desirable criteria that we will consider for an infinite sequence of samples over a bounded configuration space:

1. **Uniformity:** Good covering of the space is obtained without clumping or gaps. This can be formulation in terms of optimizing discrepancy or dispersion [12, 13].
2. **Lattice structure:** For any sample, the location of nearby samples can easily be determined.
3. **Incremental quality:** For any i, if the sequence is suddenly terminated, it has decent coverage. This is an advantage over a sequence that only provides high-quality coverage for a fixed n.

A simple grid generated by scanning has good lattice structure and uniformity, but fails to provide good incremental quality. Quality coverage is only obtained at values of i that yield a complete grid at some resolution. It is intuitive that *closed* sequences (in which the total number of samples is specified in advance) can achieve better coverage than infinite or *open* sequences (in which the total number is not specified), but this sacrifices incremental quality; for many applications, paying a small penalty in coverage to achieve incremental quality is a welcome exchange. A random sequence exhibits incremental quality, but at the expense of lattice structure and even uniformity (clumps and gap are prevalent with high probability [11, 13]). Thus, random sequences and grids appear to be quite complementary. In probabilistic roadmap (PRM) approaches (e.g., [1, 10, 15, 20]), one is usually willing to sacrifice the first two properties to obtain the last.

After considering the tradeoffs, we wondered whether it is possible to define sequences that provide all three qualities listed above. It turns out that this can be done, and the resulting sequences of samples are the primary contribution of this paper. We next provide formal definitions of discrepancy and dispersion, which our sequence will optimize.

2 Uniformity Measures

Uniform sampling criteria and techniques have been developed by numerous mathematicians over the past century. Excellent overviews of the subject include [12, 13]. Here we briefly introduce only the concepts needed for this paper. Let $X = [0,1]^d \subset \Re^d$ define a space over which to generate samples. Define a *range space*, $\mathcal{R}$, as a collection of subsets of X. Let $R \in \mathcal{R}$ denote one such subset. Reasonable choices for $\mathcal{R}$ include the set of all axis-aligned rectangles, the set of all balls, or the set of all convex subsets.

Let $\mu(R)$ denote the Lebesgue measure (or volume) of subset R. If the samples in P are uniform in some ideal

sense, then it seems reasonable that the fraction of these samples that lie in any subset R should be roughly $\mu(R)$ (divided by $\mu(X)$, which is simply one). We define the *discrepancy* [18] to measure how far from ideal the point set P is:

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

in which $|\cdot|$ applied to a finite set denotes its cardinality.

Whereas discrepancy is based on measure, a metric-based criterion, called *dispersion*, can be introduced:

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

Above ρ denotes any metric, such as Euclidean distance or ℓ^∞. Intuitively, this corresponds to the radius of the largest empty ball (assuming all ball centers lie in $[0,1]^d$).

3 One-Dimensional Sampling

To gain an understanding of the issues, it is helpful to first consider the case of sampling a one-dimensional space. In this case, a sequence introduced by van der Corput in 1935 achieves all three desired criteria with beautiful simplicity [17]. Consider a binary representation of points in $[0,1]$. A one-dimensional "grid" can be made by counting in binary. For example, if the resolution is 8, then samples are taken at: 0.000, 0.001, 0.010, 0.011, 0.100, etc. Of course, this scanning behavior of the sequence does not have incremental quality.

The *van der Corput* sequence simply takes the binary counting above and reverses the order of the bits. During the original scan, the least significant bit alternates in every step, but this only yields a small change in value. By reversing bit order, the change is maximized, causing the coverage to be nearly uniform at every point in the sequence. After bit reversal, the sequence is: 0.000, 0.100, 0.010, 0.110, 0.001, 0.101, 0.011, 0.111. An infinite sequence is constructed by using reversed-bit representations of higher binary numbers. The next eight samples are obtained by reversing binary representations of 8 through 15.

This deterministic sequence is ideal in many ways; it satisfies all three of the criteria from Section 1. It is asymptotically optimal in terms of discrepancy ($\mathcal{R}$ is a set of intervals), and also in terms of dispersion (note that this would not be achieved by a random sequence). It has a trivial lattice structure. Finally, the sequence is incremental because at any given time, the sequence can be stopped while still yielding low discrepancy and low dispersion. If the sequence is stopped at $i = 2^k$ for any integer k, then all samples are equally spaced, much as in a classical grid. More importantly, if the sequence is stopped elsewhere, the distribution of points is still good, which would not be the case of the resolution was simply improved by scanning.

4 Higher Dimensional Sampling

For use in motion planning, straightforward extensions of the van der Corput sequence to $[0,1]^d$ would be very useful; unfortunately, such sequences have not been found. Simply making a vector-valued sequence will only generate samples along a diagonal line. Halton used the bit-reversal technique to extend the sequence, using a different base for each dimension [7]. His method is as follows: choose d distinct primes $p_1, p_2, \ldots, p_d$ (usually the first d primes, $p_1 = 2$, $p_2 = 3$, ...). To construct the ith sample, consider the digits of the base p representation for i in the reverse order: $i = a_0 + pa_1 + p^2 a_2 + p^3 a_3 + \ldots$, in which $a_j \in \{0, 1, \ldots, p-1\}$. Define the following element of $[0,1]$:

$$r_p(i) = \frac{a_0}{p} + \frac{a_1}{p^2} + \frac{a_2}{p^3} + \frac{a_3}{p^4} + \cdots .$$

The ith sample in the Halton sequence is

$$(r_{p_1}(i), r_{p_2}(i), \ldots, r_{p_d}(i)), \qquad i = 0, 1, 2, \ldots .$$

This sequence is known to produce asymptotically-optimal discrepancy. It satisfies the first and last criteria from Section 1; therefore, it is a useful sequence. For virtually all randomized motion planning algorithms, one can replace a pseudo-random sequence with the deterministic Halton sequence because of the satisfaction of these properties. Recent experimental results in [6] show Halton points performing well versus other sampling techniques in the context of the PRM.

It is possible, though, to construct an alternative generalization of the van der Corput sequence, which is able to satisfy all three criteria? The neighborhood structure offered by a lattice is particularly useful in the context of motion planning. For example, in the probabilistic roadmap method, substantial time is invested in performing nearest-neighbor queries to build the roadmap. In a lattice, this information is already implicitly defined.

In addition to having grid structure, the van der Corput sequence naturally creates a *multiresolution* grid as it progressively fills in gaps in the unit interval. A generalization of the van der Corput sequence which has grid structure should have this property as well, for several reasons. First, for many problems it is impossible to know ahead of time what an appropriate resolution might be. In addition to this, it is intuitive that a multiresolution approach yields more incremental quality than a grid of fixed resolution. Finally, any open sequence (such as the van der Corput sequence) which also has grid structure *must* be multiresolution, because if the resolution is fixed, then the number of samples is fixed as well, which is a contradiction for an open sequence.

In summary, what is required is a multi-dimensional generalization of the van der Corput sequence: an open sequence generating a multiresolution grid and satisfying the criteria given at the beginning of this paper. Below, we present such a sequence.

5 A New Sequence

Before proceeding to describe our new sequence, several definitions will prove useful. Consider a classical grid in the d-dimensional unit cube, $[0,1]^d \subset \Re^d$; we define a multiresolution classical grid of resolution level l to be a grid with 2^{dl} points (i.e., 2^l points per axis). From this definition, it is apparent that a grid of resolution level l contains all the points from resolution level $(l-1)$, and that of all grids having this property, it is the one with the fewest points (assuming that all dimensions are required to have the same number of points per axis).

In addition to considering classical grids, it is worthwhile to examine Sukharev grids as well [11, 16]. Consider a grid in the d-dimensional unit cube, with k points per axis; the unit cube may then be divided into k^d regions. While the classical grid places a vertex at the origin of each region, the Sukharev grid places a vertex at the center of each region. This has the advantage of optimizing ℓ^∞ dispersion, defined in Section 2. While this difference may not seem very large, it is significant when the grids are taken to be multiresolution; while a multiresolution classical grid has 2^{dl} points for resolution level l, a multiresolution Sukharev grid has 3^{dl} points. Most of our ideas apply equally to both classical and Sukharev grids; while we will deal primarily with classical grids for the sake of brevity, we will note applications to the Sukharev case as well.

We describe the points of a classical grid of resolution l as follows:

$$P_l^n = \left\{ \left(\frac{i_1}{2^l}, \cdots, \frac{i_n}{2^l} \right) : i \in \mathbb{Z}, 0 \le i \le 2^l - 1 \right\}.$$

One may also define the *grid region* associated with point j at resolution level l as:

$$G_{j,l} = \left[j_1, j_1 + \frac{1}{2^l} \right) \times \cdots \times \left[j_n, j_n + \frac{1}{2^l} \right).$$

Similar definitions may be made for the Sukharev case.

With these definitions in mind, we may proceed to consider the sequence itself. To motivate the way our sequence is generated, consider a d-dimensional classical grid of resolution level 1, having 2^d total points. A first question is raised immediately: in what order should these points be placed? Two possible criteria for making a decision are dispersion and discrepancy. As seen above, these measure the uniformity of coverage of the space; therefore, they are natural criteria for choosing the optimal placement order. However, using dispersion as the decision criterion for a multiresolution grid often results in ties. In fact, in the case of a Sukharev grid or a classical grid on a toroidal manifold, the ℓ^∞ dispersion remains constant between complete resolution levels. For example, a Sukharev grid with i points, $3^{dl} \le i < 3^{d(l+1)}$ will have the same ℓ^∞ dispersion as the grid with 3^{dl} points (for a more-detailed explanation of the relationship between ℓ^∞ dispersions of classical and Sukharev grids, see [11]). Given this fact, it seems best to use discrepancy as the decision criterion.

From the discussion of discrepancy in Section 2, a range space $\mathcal{R}$ must be chosen over which to calculate the discrepancy. Preferably, we should choose one which is suitable for grids and grid regions. Hence, we define the set of *canonical rectangles*, similar to the b-ary canonical boxes in [12]: given positive integers n and m, let $\mathcal{Q}_m^n$ be the following family of n-dimensional canonical rectangles:

$$\mathcal{Q}_m^n = \left\{ \left[\frac{i_1}{2^m}, \frac{i_1 + j_1}{2^m} \right) \times \cdots \times \left[\frac{i_n}{2^m}, \frac{i_n + j_n}{2^m} \right) : i, j \in \mathbb{Z}, 0 \le i \le 2^m - 1, 1 \le j \le \min(2^m - i, 2) \right\}.$$

This closely relates to the previous definitions regarding the points of a multiresolution grid and their associated grid regions. In fact, the rectangles of $\mathcal{Q}_m^n$ may be as wide in any dimension as a single grid region at resolution level $m-1$, or two grid regions at resolution level m. For the case of $m = 1$, visualize $\mathcal{Q}_1^n$ as the set of all convex unions of the 2^d grid regions of a unit cube. Finally, define $\widetilde{\mathcal{Q}}_m^n = \bigcup_{i=0}^m \mathcal{Q}_i^n$. Again, these definitions apply to the case of classical grids, but analogous formulations may be made for Sukharev grids.

Now, let discrepancy be taken over the set $\mathcal{Q}_1^d$ be the criterion for determining the optimal order of the points of resolution level 1. Using this criterion, an optimal ordering list L of the first 2^d grid points (the first resolution level) may be explicitly computed (or, if we are dealing with a Sukharev grid and use a suitable modified range space, we may compute the optimal ordering list L_s for the first 3^d points). Hence, from this point on assume that we have such an ordering. Note that this only yields the correct ordering for the first resolution level. How should we fill in the next resolution level?

To answer this question, recognize that L may be viewed as an ordering not only of the first 2^d samples, but of the grid regions of the unit cube. This identification can be made because the discrepancy is calculated over a range space consisting of unions of these grid regions. Since this is the case, to maintain optimality over $\mathcal{Q}_1^d$, any future samples must follow this ordering: each future group of 2^d points must iterate through the ordering L in the same way as the first 2^d points. Hence, sample i must fall into the region of the unit cube specified by $L[i \mod 2^d]$. However, this solves only part of our problem. Suppose that we know that point i of the second resolution level must fall into $G_{j,1}$ for some j; however, where within $G_{j,1}$ should it be placed? Recognize that during the transition from resolution level l to $(l+1)$, each grid region is subdivided into 2^d subregions. Since our initial ordering scheme determined the optimal ordering of placement of 2^d points within a region, we may recursively apply it to each subregion (with an

```
GET_SAMPLE(n, L, origin, factor)
1  Point sample ← origin;
2  index ← n % |L|; //remainder of integer division
3  nextN ← n/|L|; //quotient of integer division
4  sample ← sample + (currentFactor × L[index]);
5  if (nextN = 0)
6      return sample;
7  else
8      f ← factor/2;
9      return GET_SAMPLE(nextN, L, sample, f);
```

Figure 1: Recursively generate a new sample from the vertices of a classical grid.

appropriate scaling factor, of course). An algorithm that implements this approach is given in Figure 1.

Therefore, define the infinite sequence for a classical grid as $S_d = \{s_0, s_1, \ldots : s_i = GET\_SAMPLE(i, L, \vec{0}, 1.0)\}$, in which the zero vector denotes the origin and L is the ordering list appropriate for dimension d. A brief examination of GET_SAMPLE will yield insight into the behavior of the sample sequence. At each recursion level, the integer remainder (which is less than 2^d) tells the function which grid region of the current resolution level it should go to. The function then updates the sample to be the origin of that grid region. The integer quotient tells the function how many times that grid region has previously been visited, which in turn specifies how the function behaves when it is called recursively on that region. Finally, the function returns when it determines that it has found the exact location of the sample.

It is important to note one particular feature. Suppose that point i of resolution level l is added to grid region $G_{j,(l-1)}$. Then, by the nature of the recursion, a corresponding point will have to be added to every other grid region $G_{k,(l-1)}, k \neq j$ before another point is added to $G_{j,(l-1)}$. This feature contributes to the quality of uniformity discussed in the introduction, and will contribute to the following proof, which shows that the sequence retains optimality under recursion.

Theorem 1 *Take the first i elements of the sampling sequence S_d.*

1. *This sequence is a multiresolution grid sampling sequence of length i.*
2. *From the set of multiresolution grid sampling sequences, it is discrepancy-optimal over $\widetilde{\mathcal{Q}}_l^n$, in which $l = \lceil \log_{2^d} i \rceil$, i.e., the current resolution level.*

Proof: (1) For this to be the case, the sequence must form a classical grid for every $i = 2^{dl}, l \in \mathbb{Z}$. We show this to be the case by induction on l. First, take the base case $l = 0$. The first point of the sequence is the origin, which is a classical grid of size 1. Now, assume that $i = 2^{dl}$, and that the sequence formed a classical grid for every $j = 2^{dm}, m \in \mathbb{Z}, 0 \leq m \leq l-1$. Now, $2^{dl} - 2^{d(l-1)} = 2^{d(l-1)}(2^d - 1)$; by the observation preceding this theorem, this implies that each grid region $G_{j,(l-1)}$ had $2^d - 1$ points added to the point already placed in it at the previous resolution level. Moreover, these were added according to the specification of the ordering list L, which places points on the 2^d grid vertices of a certain region. Therefore, each region contains the 2^d points of a classical grid. Since the union of two classical grids of uniform resolution results in a classical grid of the same resolution, at $i = 2^{dl}$ the samples form a classical grid of resolution l. Therefore, our inductive hypothesis is shown to be true, and part (1) is proven.

(2) We also show this by induction on the current resolution l. First, we note that resolution level 0 consists of only one point, which is placed on the origin, and it is trivially optimal over $\widetilde{\mathcal{Q}}_0^n$, which consists simply of the unit cube. Now, assume that i is such that the current resolution level is l, and that all sequences up to length $2^{d(l-1)}$ are optimal over $\widetilde{\mathcal{Q}}_{l-1}^n$. We will show that the sequence is optimal over $\widetilde{\mathcal{Q}}_l^n$, and the proof will be complete.

From the definition of $\widetilde{\mathcal{Q}}_l^n$, $\widetilde{\mathcal{Q}}_l^n = \widetilde{\mathcal{Q}}_{(l-1)}^n \cup \mathcal{Q}_l^n$. Also, we know that the first $2^{d(l-1)}$ points were added in the optimal order with respect to $\widetilde{\mathcal{Q}}_{(l-1)}^n$, by assumption; denote the grid region associated with the j-th sample of that complete grid as $G_{j,(l-1)}$ (the j-th sample is located at the origin of $G_{j,(l-1)}$). Each of the points of the current resolution level fall into one of the $G_{j,(l-1)}$; moreover, by the nature of the recursion, the order in which they fall into the $G_{j,(l-1)}$ is the same order that the original $2^{d(l-1)}$ points did. Consequently, all points of the current resolution level are optimal with respect to the set of rectangles $\widetilde{\mathcal{Q}}_{(l-1)}^n$.

Now, examine the rectangles which are part of $\mathcal{Q}_l^n$, which can be partitioned into the set of all rectangles which are completely contained within one of the $G_{j,(l-1)}$ above, and those which are not. For those which are completely contained within one of the $G_{j,(l-1)}$, optimality is clearly seen. Denote as p_j the subset of the sample sequence contained in $G_{j,(l-1)}$; by the definition of the recursion, the points $G_{j,(l-1)}$ are added to $G_{j,(l-1)}$ in precisely the optimal order defined in L; since this is the case for all $G_{j,(l-1)}$, the point sequence is optimal for all rectangles completely enclosed in some $G_{j,(l-1)}$.

Finally, we must consider the set of all rectangles in $\mathcal{Q}_l^n$ which are not enclosed in any $G_{j,(l-1)}$. First, note that for $\mathcal{Q}_l^n$ the maximum width of any rectangle in a single dimension is $1/2^{(l-1)}$, the size of each block $G_{j,(l-1)}$. Let $r \in \mathcal{Q}_l^n$ be a rectangle partially enclosed in gr_j; then, there are three possibilities for each dimension of r: first, it is entirely in $G_{j,(l-1)}$; second, it covers the top half of $G_{j,(l-1)}$ and the bottom half of some other block; or

third, it covers the bottom half of $G_{j,(l-1)}$ and the top half of some other block. Denote by r_o the portion of r which is outside of $G_{j,(l-1)}$, and by r_i the portion which is inside. Now, one can take the reflection of r about r_i; then, each part of r_o is mapped to a place inside $G_{j,(l-1)}$. Define q to be a rectangle resulting from such a reflection, and note that $q \subseteq G_{j,(l-1)}$ and $q \in \mathcal{Q}_l^n$. Hence, we know that q is part of a set of rectangles for which optimal discrepancy has already been shown.

Let P_c be the set of points in the sequence for which analogous points can be found in every grid region $G_{j,(l-1)}$, and let P_i be the remaining points. Note that all points in P_i are analogous to each other by the observation immediately preceding this theorem; this implies that no rectangle in $\mathcal{Q}_l^n$ can contain more than one of these points. Now, take some rectangle r as described above. If this rectangle contains a point $p \in P_i$, then denote $G_{j,(l-1)}$ as the grid region containing this point; else, choose it to be any grid region containing a portion of r. Define r_o, r_i as above; then, we may once again take the reflection of r about r_i. The rectangle q obtained from this reflection has measure identical to rectangle r, and it contains the same number of points. We know this to be the case, because by assumption all points in r_o have analogues in $G_{j,(l-1)}$. This is the case since $p \in P_i$ is contained in r_i, by our choice of $G_{j,(l-1)}$; thus, all points in r_o are part of P_c and consequently have analogues in every grid region of resolution $(l-1)$. Thus, since r and q have identical measures and numbers of enclosed points, and q is part of the set for which optimal discrepancy has been shown, it is impossible for r to hurt the total discrepancy.

Since we have now shown that the sequence is optimal over $\widetilde{\mathcal{Q}}_{(l-1)}^n$ and $\mathcal{Q}_l^n$, we know that the sequence is optimal over $\widetilde{\mathcal{Q}}_l^n = \widetilde{\mathcal{Q}}_{(l-1)}^n \cup \mathcal{Q}_l^n$. Therefore, our inductive hypothesis is shown to be true and part (2) of the theorem is proven. ■

6 Useful Properties for Motion Planning

Thus far we have defined a sample sequence which incrementally builds a multiresolution grid in an order which is discrepancy-optimal over an appropriately chosen range space. While this is of value on its own, we are particularly interested in using this sequence for motion planning applications, especially those which depend heavily on having a good sample set (e.g., the PRM). Hence, we now examine several properties of this sample sequence, to demonstrate the potential benefits of this sequence in motion planning applications.

A first consideration is the amount of time required to generate each sample. If it is computationally expensive to generate the sample sequence, this may offset time gained through the quality of the sequence. Hence, we give bounds on the time required to generate a particular sample. (In this and all future considerations, all scalar mathematical operations are considered to be constant time, since they depend on internal representation only. Vector operations are considered to be $O(d)$ time.)

Property 1 *The position of the i-th sample in the d-dimensional sampling sequence S_d can be generated in $O(\log i)$ time.*

Proof: The recursive function specified in Figure 1 may be written as $GL(i) = GL(i/2^d) + O(d)$. The solution to this recursion is $O(d \log_{2^d} i)$. Since $\log_{2^d} i = (\log i)/d$, the final result is $O(d(\log i)/d) = O(\log i)$. ■

For purposes of comparison, pseudo-random samples usually require $O(d)$ time and Halton samples require $O(d \log i)$.

In the introduction, we stated that lattice structure is desirable because the location of neighbors can easily be determined. It is well-known that all points in a lattice can be specified in terms of a collection of d linearly-independent basis vectors $b_1, \ldots, b_d$. In the case of a grid, the basis vectors are simply the columns of the $d \times d$ identity matrix. By adding (or subtracting) these basis vectors, the neighbors of a point can be found immediately. We define the *i-neighbors* of a point p as those points which may be reached by adding or subtracting i distinct basis vectors, $1 \leq i \leq n$.

However, our points are specified in terms of their index in the sequence; based on this index alone, it is unclear how to calculate the index of a neighbor in the sample space. It is possible to do so, however. The algorithm is too long to present here in its entirety; thus, we will sketch its operation.

For any element in the ordering list L, it is possible to store the order indices of all of the i-neighbors of each element. Since any i-neighbor may be found through a sequence of 1-neighbors, it suffices to store the order indices of each elements 1-neighbors, of which there are $2d$. The space required to do this is consequently $O(d2^d)$, since $|L| = 2^d$.

Now, suppose we wish to find a particular 1-neighbor of sample i. As in the GET_SAMPLE function, we may execute a recursion, storing the sample index of each "ancestor" and the order index between ancestors. We then use this information along with the neighborhood information stored with each element in the ordering list L to find the sample index of the desired neighbor. Then, we perform a simple query to see if the sample corresponding to this index exists.

Property 2 *Let the number of samples taken so far be N. Then, a 1-neighbor of any of these samples can be found in $O((\log N)/d)$ time.*

Proof: Apply the algorithm described above. The function will recurse at most $O((\log N)/d)$ times and generate as many ancestors, similar to the GET_SAMPLE function. Note that this requires only $O((\log N)/d)$ rather than $O(\log N)$ time as in the analy-

sis of GET_SAMPLE because the actual sample location is not being remembered; only indices are being calculated, which are dimension-independent. Now, to find the desired neighbor, the entire ancestor chain may have to be traversed (this is similar to binary addition, in which adding 1 may result in each bit needing to be changed). However, only a constant amount of work is done each time (applying some simple formulas to obtain new indices). After doing this, the sample index of the desired neighbor has been calculated; if this value is larger than N, then the neighbor does not exist. If a vector of pointers to previous samples is kept, simply indexing into this vector will allow one to determine the previously-calculated position of this sample. Therefore, the total time required is simply $O((\log N)/d)$. ■

The scheme described in the proof above can easily be adapted to the case of motion planning, in which some samples of index less than N may not exist, due to being in collision with some obstacle. In this case, the corresponding entry in the vector is nil, and the query returns that the vertex does not exist. Also, the fact that we can calculate neighbors in this way suggests the potential for developing "lazy" planners that can search the space without allocating huge amounts of space for storing edge connections and neighborhood information.

This method is an improvement over naïve search (in which the proximity of every sample to the initial point is checked), and can be used to find 2-,...,d-neighbors in addition to 1-neighbors; however, there may be situations in which it is desired to determine the radius necessary to connect to the i-neighbors of a particular point at resolution level l, for use in naïve search. At resolution level l, the distance between 1-neighbors is $1/2^l$; hence, the distance between i-neighbors is $\sqrt{i(1/2^l)^2} = \sqrt{i}/2^l$. Thus, by setting the connection radius appropriately, one may use some other technique to connect neighboring grid points. In passing, it should be noted that to prevent a point from connecting with non-neighboring points (e.g., those of distance $2/2^l$ in a single direction), i must be 3 or less.

Finally, since the grid is multiresolution, it may be possible to reduce collision checks in resolution level l by remembering some results from resolution level $(l-1)$. We give a bound on the number of collision checks that may be saved in this way.

Property 3 *If samples are connected only to their 1-neighbors, then at most a fraction of* $\frac{1}{2^{d-1}}(1 - \frac{1}{2^{l-1}})$ *of the collision checks required at resolution level* l *may be saved.*

Proof: The total number of points in a grid of resolution level l is 2^{dl}. Assume for a moment that we are on a toroidal manifold, so that each point has $2d$ 1-neighbors. Then the total number of edges is $d2^{dl}$, since each edge is shared by two points. Now, to correct for being in $\Re^d$, we must remove some edges. For each dimension, a fraction of 2^l of the edges cross the boundary (since we may recall that the number of points per axis is 2^l). Therefore, we must remove $d2^{dl}/2^l = d2^{(d-1)l}$ edges, leading to a total of $d2^{dl} - d2^{(d-1)l}$ edges in resolution level l. Since the fraction of collision checks saved is the same as the fraction of new edges covered by edges of the previous resolution level, we find that we may save:

$$\frac{2(2^{d(l-1)} - 2^{(d-1)(l-1)})}{2^{dl} - 2^{(d-1)}} = 2 \cdot 2^{-d} \frac{2^{dl} - 2 \cdot 2^{(d-1)l}}{2^{dl} - 2^{(d-1)}} =$$
$$\frac{1}{2^{d-1}} \left(1 - \frac{2^{(d-1)l}}{2^{dl} - 2^{(d-1)}}\right) = \frac{1}{2^{d-1}} \left(1 - \frac{1}{2^l - 1}\right)$$

■

From the equation above, it can be seen that while there may be some savings for low-dimensional applications, there will be only slight savings for higher-dimensional problems. Consequently, we expect the primary benefits of our sequence's lattice structure to be in its implicitly-defined neighbors, rather than in collision check savings.

7 Experimental Results

While extensive empirical testing is needed to conclusively determine the practical utility of our sequences, we have conducted several experiments using the classical grid-based sequence in a PRM-like planner. Our experiments indicate that dispersion and discrepancy are good measures of sample sequence quality and that grid-based sequences with high incremental quality can be acceptable replacements for random samples in roadmap planners. For comparison purposes, we tried three different sampling schemes: random sampling, multiresolution grid sampling in scanning order, and the discrepancy-optimal order discussed in this paper. The four experimental setups can be seen in Figures 2 and 3, and results in Figures 4 and 5. In these experiments, all grid sampling methods were configured to connect only to their neighbors in the current grid resolution level (although the algorithm described in Section 6 was not used). We used two different connection rules with the random samples: one uses a fixed radius, and the other attempts to connect to the k nearest neighbors. Since our grid sampling methods attempted at most $2d$ connections per new node, we set $k = 2d$.

In each experiment, we found that grid-sampling methods performed well when compared to random sampling. Discrepancy-optimal grid sampling outperformed random sampling in each experiment, and scanning-order grid sampling performed reasonably well. However, we believe that in general, scanning-order grid sampling will not perform as well as random sampling, because its incremental quality is so low. On the other hand, discrepancy-optimal grid sampling sequences have good incremental quality and we would therefore expect

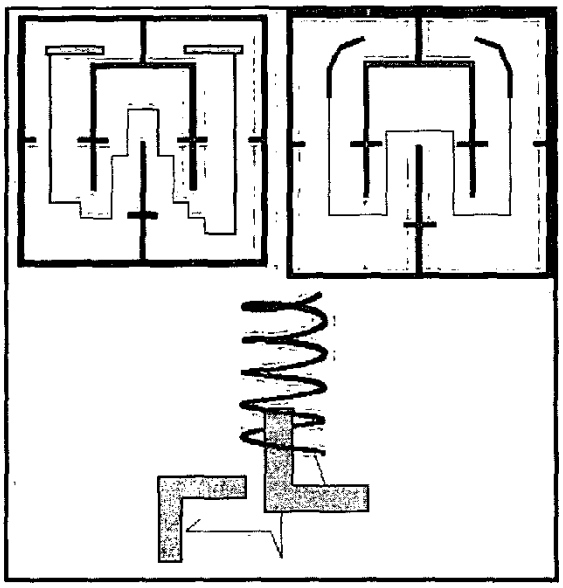

Figure 2: Preliminary experiments: top row, from the left: moving a rigid bar through a maze ($d = 3$), moving a rigid chain through a maze ($d = 5$); bottom row, removing an L-shaped robot from a spring ($d = 6$).

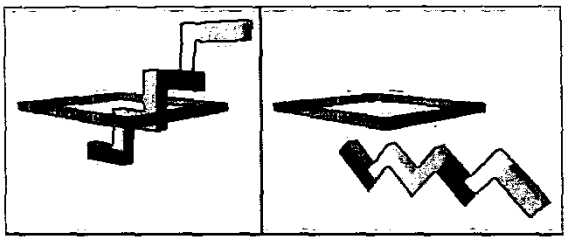

Figure 3: Moving a robot arm with a fixed base through a rectangle ($d = 6$). On the left is the initial state, on the right the goal state.

Prob.	Dim	R_Rad	R_KNear	Scan	Opt
Bar	3	440.52	922.44	684	877
Links	5	6674.04	2404.84	6785	4854
Elbow	6	4399.58	4518.04	3168	2413
Arm	6	148.58	1040.13	2704	1652

Figure 4: Comparisons of the number of nodes used in the experiments. R_Rad uses random samples and a fixed connection radius, R_KNear uses random samples and attempts k-nearest connections, Scan uses multiresolution grid samples in scanning order, and Opt uses the sequences introduced in this paper. Random sampling sequences are averaged over 50 trials.

them to perform well across a broad range of tests. In our experiments, we did not see the running time correspond precisely to the number of nodes in the roadmap; there are several reasons for this. Most importantly, the performance of the roadmap planners can depend heavily on the connection method chosen and its implementation. In particular, a fixed-radius connection technique with a large radius may result in few nodes but a large number of connection attempts, while a k-nearest approach or grid-based connection technique may have more nodes yet attempt connections more conservatively. After a connection method has been chosen, there are still several ways one can tune parameters or optimize performance for certain problems. Other researchers have recognized the difficulty of making good experimental comparisons for PRM-style planners, and work has been done to develop a more complete experimental analysis of different techniques [6]. While this degree of thoroughness is outside the scope of this paper, we believe that our experiments give a good estimate of expected performance.

8 Conclusions and Future Work

In conclusion, we have presented a new sample sequence, which satisfies all of the desirable criteria explained in Section 1 (uniformity, lattice structure, and incremental quality), and which is an arbitrary-dimensional generalization of the van der Corput sequence. As a low-discrepancy, low-dispersion sequence, it provides good coverage of the sample space; as a lattice, it has implicit neighborhood structure, which can be exploited in planning algorithms; and having incremental quality, it provides good coverage if terminated after any sample and can be easily interchanged with other incremental sampling techniques. These properties suggest that this sequence will be of benefit to the motion planning community. In particular, we believe that this sequence is a useful replacement for random sampling in PRM-style planners.

There are several directions for future work. In the previous section, we expressed the desire to do compre-

Prob.	Dim	R_Rad	R_KNear	Scan	Opt
Bar	3	1.98	1.81	1.08	1.48
Links	5	373.26	844.58	438.00	219.10
Elbow	6	71.11	777.08	63.20	29.56
Arm	6	44.28	69.93	33.91	13.94

Figure 5: Comparisons of the construction times (in seconds) corresponding to the results of the previous figure. The experiments were implemented in Gnu C++ on a 2.0GHz PC running Linux.

hensive experimental analysis; as part of this, we would like to determine limits on the dimensions for which this sequence is useful, and to gain insight into the relative merits of different types of grid sampling methods. Second, we would like to discover a more elegant way to describe and generate the sequence (such as the bit-reversal description appropriate for the van der Corput and Halton sequences); currently, we use a less-appealing recursive scheme based on an explicitly-calculated ordering for the first resolution level. Third, we would like to investigate the possibility of using other sets of rectangles for discrepancy calculations. Fourth, we have already mentioned that the extension to the Sukharev grid is fairly straightforward; similarly, an extension to an arbitrary lattice is not difficult. We would like to implement and test both of these extensions. Finally, we plan to continue to develop software generating and utilizing this sample sequence for use in our own planners, and to make the software available for use by the community (the latest versions of this software are available at http://msl.cs.uiuc.edu/).

Acknowledgments This work was funded in part by NSF Awards 9875304, 0118146, and 0208891. We also thank Robert Bohlin, Michael Branicky, Bruce Donald, Mike Erdmann, and Lydia Kavraki for helpful recent discussions.

References

[1] 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.

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

[3] R. Bohlin and L. Kavraki. Path planning using lazy prm. In *IEEE Int. Conf. Robot. & Autom.*, 2000.

[4] 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.

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

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

[7] J. H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numer. Math.*, 2:84–90, 1960.

[8] C. Holleman and L. E. Kavraki. A framework for using the workspace medial axis in PRM planners. In *IEEE Int. Conf. Robot. & Autom.*, pages 1408–1413, 2000.

[9] Y. K. Hwang and N. Ahuja. Gross motion planning–A survey. *ACM Computing Surveys*, 24(3):219–291, September 1992.

[10] 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.

[11] S. M. LaValle and M. S. Branicky. On the relationship between classical grid search and probabilistic roadmaps. In *Proc. Workshop on the Algorithmic Foundations of Robotics (to appear)*, December 2002.

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

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

[14] G. Sánchez and J.-C. Latombe. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In *Int. Symp. Robotics Research*, 2001.

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

[16] 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.

[17] J. G. van der Corput. Verteilungsfunktionen I. *Akad. Wetensch.*, 38:813–821, 1935.

[18] H. Weyl. Über die Gleichverteilung von Zahlen mod Eins. *Math. Ann.*, 77:313–352, 1916.

[19] 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.

[20] Y. Yu and K. Gupta. On sensor-based roadmap: A framework for motion planning for a manipulator arm in unknown environments. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, pages 1919–1924, 1998.