Incremental Grid Sampling Strategies in Robotics

Stephen R. Lindemann, Anna Yershova, and Steven M. LaValle

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

Summary. We present algorithms for generating deterministic sample sequences using incremental grid-based sampling. Our algorithms are designed to generate dense sample sequences over spaces common in robotics, such as the unit cube, $SO(3)$, and $SE(3)$. Our sampling techniques provide the advantageous properties of uniformity, lattice structure, and incremental quality. In addition, the inherent structure of grid-based sequences not only enables them to be used in the place of other sampling techniques in existing algorithms, but also permits the development of new algorithms aimed at exploiting this structure.

1 Introduction

Numerous algorithms in robotics rely heavily on the generation of samples over a continuous state space. Motion planning algorithms, for example, use sampling along with efficient collision detection to find collision-free paths in the configuration space. Hence, it is of great importance that the underlying sample sequences be as good as possible. Previous work has argued that randomization is not the key to the success of modern motion planning algorithms; rather, there are deterministic sampling schemes which can be expected to perform as good or better than sampling uniformly at random [9, 10]. Furthermore, it was speculated that integrating sampling methods more closely into motion planning algorithms could be advantageous compared to viewing sampling as a "black box". One way to do this is by using a sample sequence with lattice structure, and developing algorithms which exploit that structure. In order to make this possible, it is of first importance to develop sample sequences which have lattice structure as well as high incremental quality, which is of great importance in many situations. Incremental quality is the property that if the sequence is stopped after any number of samples, the samples taken up to that point uniformly cover the space. Previous work addressed this by the introduction of incremental grid-based sequences which optimized discrepancy [11]. These sequences have high incremental quality; however, the

methods described require exponential space to store the optimal ordering. This renders these methods unsuitable for high-dimensional problems. In this paper, we introduce new grid-based sequences which optimize different uniformity measures. The orderings we find are simple and compact to represent, making them suitable for high-dimensional problems. In addition to this, we show how to use these sample sequences to sample other topological spaces common in robotics, such as $SO(3)$ and $SE(3)$. The methods we present use Platonic solids as a means to apply sampling techniques originally developed for the unit cube to these spaces, and were originally formulated in [23].

We begin by briefly overviewing important concepts in the area of uniformity measures and uniform sampling techniques.

2 Uniformity Measures and Uniform Sampling Techniques

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 $X = [0,1]^d \subset \mathbb{R}^d$ 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* [22] to measure how far from ideal the point set P is:

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

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:

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

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 $[0,1]^d$).

Discrepancy can be thought of as enforcing two criteria: first, that no region of the space is left uncovered; and second, that no region is too full. Dispersion eliminates the second criterion, leaving only the first. It can be shown that low discrepancy implies low dispersion [13]. It is easy to conceive

of a uniformity criteria that emphasizes the second criterion, rather than the first. A criterion such as this would require that points be spread as far away from each other as possible; we call this *mutual distance*, and introduce it in Section 3. As in the case of dispersion, a bound on discrepancy implies one on mutual distance.

Motivated by integration and optimization problems, sampling issues have been studied extensively in the applied mathematics community. Sample sets and sequences were developed to replace the random sequences traditionally used for these applications; they received the name *quasi-Monte Carlo* to denote this connection. Due to the fundamental importance of numerical integration, and the close connection between discrepancy and integration error, most of the quasi-Monte Carlo literature focuses on the discrepancy measure. Low-discrepancy sampling methods can be divided into three categories: Halton/Hammersley sampling [3, 4, 20], lattices [5, 7, 12, 18, 21], and (t, s)-sequences and (t, m, s)-nets [13, 14].

3 Optimal Grid Sampling Techniques

In contrast to the sampling techniques discussed above, we introduce methods based on incremental sampling of grid points. In [11], grid sample sequences which optimized discrepancy were introduced. In that work, it was shown how to construct discrepancy-optimal grid sampling sequences, and several interesting properties of such sequences were proven. In this section, we examine optimal grid sampling sequences with respect to different quality measures. The new sequences will retain the good properties of the discrepancy-optimal sequences and will add key new advantages.

Before proceeding to describe the sequences, several definitions will be useful. Consider a classical grid in the d-dimensional unit cube, $I^d = [0,1]^d \subset \mathbb{R}^d$; we define a multiresolution classical grid of resolution level l, P_l^n to be a grid with 2^{dl} points (i.e., 2^l points per axis). More formally, $P_l^n = \left\{\left(i_1/2^l, \cdots, i_n/2^l\right) : i \in \mathbb{Z}, 0 \leq i \leq 2^l - 1\right\}$. One may also define the *grid region* associated with point j at resolution level l: $G_{j,l} = \left[j_1, j_1 + 1/2^l\right) \times \cdots \times \left[j_n, j_n + 1/2^l\right)$.

Our previous work focused on optimizing discrepancy; however, discrepancy is only one of several interesting and useful uniformity criteria. In addition to this, discrepancy-optimal sequences currently suffer from the requirements of both exponential time to compute and exponential space to store. By optimizing different uniformity criteria, we will derive sequences which eliminate the exponential space requirement, leading to sequences which have very compact, efficient representations. Our techniques are capable of optimizing either dispersion or another criterion we call *mutual distance*; the mutual distance of a set S is

$$\rho_m(S) = \min_{x,y \in S} \rho(x, y)$$

Intuitively, mutual distance corresponds to forcing sequence points to be as far from each other as possible. Both dispersion and mutual distance are metric-based criteria, in contrast to discrepancy, which is measure-based. The techniques we develop in this section will exploit this fact to generate sequences with compact representations.

The rest of this section proceeds as follows. First, we will describe the representation that our ordering takes and how it is used to generate samples. Second, we prove the existence of optimal orderings admitting the representation given. Third, we show how we compute optimal orderings and approximations of the optimal orderings. Primary focus will be given to mutual distance; however, we will address dispersion-optimal orderings as well.

3.1 The digital construction method

The most widely-used quasi-Monte Carlo sampling techniques are (t, m, s)-nets and (t, s)-sequences. Typically, these sample sets and sequences are generated by a method known as the *digital construction method*. The digital construction of a (t, m, s)-net (in base 2) takes a set of generator matrices and computes sample locations based on multiplying these matrices by a vector which whose components are the bit representation of the sample index. Our approach for computing grid orderings which optimize dispersion or mutual distance uses this same idea.

Consider the integers $[0, 1, \ldots, 2^d - 1]$. Each integer can be written in the following form:

$$x = \sum_{i=0}^{d-1} a_i 2^i, \ a_i \in \mathbb{Z}_2.$$

Hence, there is a straightforward bijection between this set of integers and the group $\mathbb{Z}_2^d$. Now consider a $d \times d$ matrix, with each element $e_{ij} \in \mathbb{Z}_2$. If this matrix is full rank, it defines a linear transformation for the vector space $\mathbb{Z}_2^d$ over $\mathbb{Z}_2$ (using standard matrix/vector multiplication). Via this transformation, and the previously-mentioned bijection, the bit representations of the integers $[0, 1, \ldots, 2^d - 1]$ are mapped to d-dimensional vectors corresponding to grid points. Each transformation represents a unique ordering of the set of all grid points in the grid of resolution 1. See Figure 1 for an example. It is clear that the space of all grid orderings is much larger than the orderings that can be generated by matrices of this type. It is desirable that our orderings take this form, due to its compact representation and the ease of computation it affords. Fortunately, there exist orderings which both optimize either dispersion or mutual distance and which admit this representation; we proceed to show that this is the case.

3.2 Optimal orderings using the digital method

Our approach for constructing an optimal generator matrix is to build it one column at a time, rather than computing it as a unit. This is possible because

The following matrix defines an ordering in four dimensions.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

An integer $x = a_0 + a_1 2^1 + a_2 2^2 + a_3 2^3 \in [0, 15]$ is written in vector form as $(a_0\ a_1\ a_2\ a_3)'$. Multiplying this vector on the left by the given matrix yields another vector, whose elements are interpreted as the coordinates of a 4-dimensional grid. Applying the transformation to the sequence $\{0, 1, \ldots, 15\}$ yields an ordering which optimizes mutual distance for the grid of resolution level 1. Points in higher resolution levels are obtained by recursively applying this ordering.

Fig. 1. An example of the digital construction method in 4 dimensions.

only the first i columns affect the location of the first 2^i samples (in the bit representation of the first 2^i integers, the final $d - i$ elements are zero and hence the final $d - i$ columns of the matrix do not affect the product). Hence, we will select d generator vectors, one at a time, and use these as the columns of our generator matrix.

Before proceeding to show how to construct matrices to generate optimal orderings, we must first define our optimality criteria more precisely. A grid ordering which optimizes mutual distance is defined to be a sequence of points such that, given the first k elements of the sequence, the $(k+1)$-th element is chosen to maximize the mutual distance of the set $\{x_1, x_2, \ldots, x_{k+1}\}$. First, we will show how to construct generator matrices which optimize mutual distance; then, we will comment briefly on the analogous dispersion-optimal constructions. We begin with a useful lemma:

Lemma 1. *Let $G_i = \{g_1, g_2, \ldots, g_i\}$ be i linearly independent vectors in $\mathbb{Z}_2^d$, and let $S_i = Span(G_i) = \{x : x = \sum_j a_j g_j, a_j \in \mathbb{Z}_2\}$. Take some vector $u \in \mathbb{Z}_2^d \setminus S_i$, and construct the sets $G_{i+1} = G_i \cup u$ and S_{i+1}. If the distance from u to its nearest neighbor in S_i is d, then the distance from every element of $S_{i+1} \setminus S_i$ to its nearest neighbor in S_i is also d.*

Proof. The set $S_{i+1} \setminus S_i$ is simply the left coset of S_i generated by u. It is obvious that the difference between any element of the coset and any element of S_i is an element of the coset; its magnitude is the same as the distance between u and some element of S_i. Hence the distance from every element of $S_{i+1} \setminus S_i$ to its nearest neighbor in S_i is d.

While simple at first glance, this lemma is important for understanding grid orderings constructed using the digital method. After choosing i generator vectors and generating the points corresponding to them, the remaining $2^d - 2^i$ grid points are divided into equivalence classes (each of size 2^i), corresponding

to different choices of the next generator vector g_{i+1}. The above lemma shows that all vectors in a given equivalence class have the same distance to the set of points generated by the first i generator vectors.

Proposition 1. *The first generator vector is* $(1\ 1\ \cdots\ 1)'$.

Proof. The first point in the sequence (for any choice of initial generating vector) is the zero vector. Hence, the first generator vector g_1 should be chosen to be maximally distant from the zero vector. That vector is clearly $(1\ 1\ \cdots\ 1)'$.

Proposition 2. *Let* $G_i = \{g_1, g_2, \ldots, g_i\}$ *be* i *linearly independent vectors in* $\mathbb{Z}_2^d$ *with* $g_1 = (1\ 1\ \cdots\ 1)'$*, and let* $S_i = Span(G_i) = \{x : x = \sum_j a_j g_j, a_j \in \mathbb{Z}_2\}$*. If the next generator vector* g_{i+1} *is chosen to be some vector* $u \in \mathbb{Z}_2^d$ *which is maximally distant to* S_i*, then mutual distance is maximized for all samples in the sequence with indices in the range* $(2^i, 2^{i+1}]$.

Proof. In order that the mutual distance be maximized for sample index 2^d+1, the vector chosen must be maximally distant from the set S_i, as assumed. Since selecting this vector as the next generating vector will result in the generation of the next 2^i elements of the sequence, we must show that making such a selection will maximize mutual distance for the next 2^i elements. Assume that the distance from u to S_i is r. Then, the mutual distance of the resulting set is $\rho_m(S_i \cup u) = \min(r, \rho_m(S_i))$. Taking $g_{i+1} = u$, we know by Lemma 1 that each element in the set $S_{i+1} \setminus S_i$ (that is, each of the next 2^i elements in the sequence) is also of distance r to S_i. Also, it is the case that $\rho_m(S_{i+1} \setminus S_i) = \rho_m(S_i)$. So, the addition of the next 2^i elements does not change the mutual distance of the set, which after the addition of the vector u was already $\min(r, \rho_m(S_i))$. Thus at each point of the sequence with index in the range $(2^i, 2^{i+1}]$, the mutual distance remains maximized.

Hence, by choosing the $(i+1)$-th generator vector to maximize the distance from the set generated by the previous i generator vectors, we maximize the mutual distance for the next 2^i elements of the sequence. By constructing our generator matrix from d such vectors, then, we obtain an ordering for the first 2^d points of the sequence which optimizes our criterion of mutual distance. To obtain the entire infinite sequence, then, it remains only to specify how to apply to generator matrix to samples whose index is greater than or equal to 2^d (note that the first sample has index 0, so the first 2^d elements have index up to $2^d - 1$). As in the case of the discrepancy-optimal sequence [11], we apply the ordering recursively to each of the 2^d grid regions corresponding to the first 2^d points. After the top-level grid region is selected by applying the generator matrix to the first d bits of the sample index, the position within that region is found by applying the generator matrix to the next d bits of the sample index. This process is repeated until the final sample location is found.

Theorem 1. *The sequence generated by the above procedure optimizes mutual distance at every point in the sequence.*

Proof. We show this by induction on the resolution level l. By Propositions 1 and 2, the sequence optimizes mutual distance for the first 2^d samples, which corresponds to $l = 1$. Now, assume that the sequence optimizes mutual distance for resolution level l, and examine the sequence during resolution level $l + 1$. It is clear that within a particular grid region, the points are added in an order which maximizes the mutual distance (this follows directly from the fact that the generator matrix maximizes mutual distance, and the matrix is applied recursively to each grid region). Now, it is also the case that if some point is added to a particular grid region (chosen recursively as described above), a corresponding point is added to every other grid region in the space before any another point is added to the initial grid region (this is because the grid region is determined by the lowest dl bits of the sample index, which cycle through all possibilities before a higher bit changes). This implies that if some point x of resolution level $l + 1$ occurs in the sequence and decreases the mutual distance, the next 2^{dl} points to be added will be points corresponding to x, but in different grid regions. This implies that they will not reduce the mutual distance, because they will have the same distance from their neighbors that x had from its neighbors. Note that points from neighboring grid regions do not interact in such a way as to decrease the mutual distance, because there will always be an analogous point in the same grid region with identical distance, and we know that all intra-grid region distances follow a mutual distance-maximizing ordering. Since each point x which reduces the mutual distance is added according to the optimal ordering given by the generator matrix, and all other points do not reduce the mutual distance, we see that each point added in resolution level $l + 1$ is added in a way that maximizes the mutual distance. Hence the inductive hypothesis is seen to be true and the theorem proved.

The construction of dispersion-optimal generator matrices proceeds similarly. The most significant difference is that the optimality criteria must be slightly modified. Lemma 1 implies that given some G_i and corresponding S_i, for *any* choice of g_{i+1}, the dispersion of the sequence will remain constant until all samples of set S_{i+1} have been taken. This means that the optimality criteria used for mutual distance is inadequate for dispersion; we must make a slight modification. Assume that the set G_i is given. Then the optimal choice for the next generator vector g_{i+1} is defined to be that which minimizes the dispersion of the point set S_{i+1}. Note that a sequence optimizing this criteria will also be optimal in the point-wise sense (as in the case of mutual distance), but the reverse is not necessarily true. For dispersion-optimal sequences, the first generator vector can also be seen to be $(1\ 1\ \cdots\ 1)'$. Each successive generator vector is chosen to optimize the above criterion. After the entire generator matrix has been constructed, we apply it recursively (as in the case of mutual distance). Under the appropriate definitions, it can be shown that an optimal sequence is obtained.

3.3 Useful Properties for Motion Planning

In [11], several properties of discrepancy-optimal grid sequences were shown. The same properties hold for sequences optimizing mutual distance or dispersion; we present the two main properties here. For each of these properties, dimension is considered to be constant. However, at various points we note dependence on dimension; in all cases, dimension has only a limited effect on practical performance.

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.

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 bit representation of the i-th sample contains $O(\log i)$ bits. Thus the vector representing the sample will have this length. There will be $O(\log i)$ recursions (the number of recursions is the number of bits divided by the dimension). Each recursion requires $O(d^2)$ operations, which is constant time since d is constant. Hence the total time is $O(\log i)$.

Property 2. Let the number of samples taken so far be N. Then, the existence of a neighbor (in the current resolution level) of any of these samples can be found in $O(\log N)$ time.

Proof. Given the index of sample i, one can find its position in $O(\log i)$ time. Then, depending on the total number of samples, N, taken, add the appropriate quantity to the calculated position, reflecting the desired neighbor. To calculate the index of this sample, one can easily use the inverse of the procedure used to generate the position (the inverse of the generator matrix is easily computed, since it is full rank). Then a hash table can be queried to see if a sample of that index exists, in constant time. The cost to compute the inverse index is at most $O(\log N)$, so the total cost is $O(\log N)$.

These properties indicate the potential of the sequences we have introduced. These samples can be generated very quickly (especially given the fact that all mathematical operations can be done using simple bit operations). In fact, samples such as these are likely to cost much less than "good" random points, such as those generated by sophisticated nonlinear congruential techniques. In addition to this, sampling from a set of grid points gives implicit structure to the sample set. This structure can be exploited efficiently, which could be of value when using the sample sequence for applications such as motion planning.

4 Deterministic Sample Sequences for Spheres and SO(3)

Thus far, we have considered sampling in the unit cube $I^d = [0,1]^d \subset \mathbb{R}^d$. Problems in robotics, however, typically result in much more complex topological spaces. Algorithms for tasks such as motion planning typically take samples from the unit cube and transform them to the appropriate space. However, it may be possible to design sample sequences explicitly with particular topological spaces in mind. In this section, we consider the case of spheres and $SO(3)$. We will then discuss how to sample $SE(3)$ using these methods. We begin with some brief definitions.

4.1 Definitions and Quality Measures for Points on Spheres and SO(3)

We consider generating samples over spheres and $SO(3)$. Let S^d represent a d-dimensional sphere, embedded in $\mathbb{R}^{d+1}$ as

$$S^d = \{x \in \mathbb{R}^{d+1} \mid \|x\| = 1\}.$$

The set of all rotations in $\mathbb{R}^3$ is denoted as $SO(3)$, which is defined as the set of all 3×3 orthonormal matrices. It will be helpful to sometimes represent $SO(3)$ as the set, H, of unit quaternions, each of which is expressed as $h = a + bi + cj + dk$, with the identification $h \sim -h$ [8]. Note that it appears that $H = S^3$, except that antipodal points on S^3 are identified in the definition of H. This leads to a close relationship between sampling on sphere and sampling on $SO(3)$.

Discrepancy and dispersion may be defined for these spaces, analogous to the case of $\mathbb{R}^d$ given in Section 2. The definition of discrepancy is identical; a rotation-invariant measure μ is used (the Haar measure over the set of all rotation matrices in $SO(3)$), and typical range spaces are the set of all spherical caps (intersections of the sphere with half spaces) or the set of all spherical slices (intersections of two half-spheres) [2, 15]. Likewise, the definition of dispersion is identical; the metric ρ is required to be rotation-invariant.

We now proceed to outline a general method for generating samples on spheres and $SO(3)$. For more detail, see [23].

4.2 Generating Samples on Spheres and SO(3)

Our general approach to sampling is based on Platonic solids. In $\mathbb{R}^3$, a Platonic solid or *regular polyhedron*, is a polyhedron for which every face is a copy of a regular polygon, fixed over all faces, and the degree of every vertex is fixed. Let (v, e, f) denote the numbers of vertices, edges, and faces of a regular polyhedron. Although there are an infinite number of regular polygons,

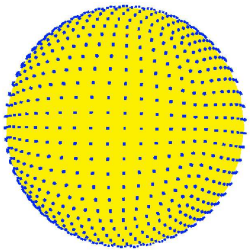


Fig. 2. Distribution of points on the sphere S^2 generated by a grid (Sukharev [19]) on each spherical face.

there are only five regular polyhedra: tetrahedron (4,6,4), cube (8,12,6), octahedron (6,12,8), icosahedron (12,30,20), and dodecahedron (20,30,12). The notion of regular polyhedron can be generalized to higher dimensions to obtain a *regular polytope*. In $\mathbb{R}^4$, it turns out that there are six regular polytopes: simplex (5,10,10,5), cube (16,32,24,8), cross polytope (8,24,32,16), 24 cell (24,96,96,24), 120 cell (600,1200,720,120), 600 cell (120,720,1200,600). The forth element in each sequence denotes the number of 3D cells (which are regular polyhedra). Finally, in $\mathbb{R}^d$ for any $d > 4$, there are only three regular polytopes: simplex, cube, and cross polytope.

We first address the problem of generating a uniformly distributed set of points over S^d. Consider inscribing any $(d+1)$-dimensional regular polytope inside of S^d, so that all of its n vertices lie in S^d. The set of vertices are beautifully arranged around S^d so that the points are evenly spaced. Furthermore, the edges of the polytope yield a regular lattice structure that is natural for building roadmaps in planning problems. For the case of sampling $SO(3)$, we simply use a set of vertices that lie in one hemisphere (making sure that no antipodal pairs of points appear in the set). The edges can be obtained directly from the polytope by making the appropriate identification of antipodal pairs.

Unfortunately, there are only a few combinations of n and d, for which these ideal samples may be constructed for S^d and $SO(3)$. This might be suitable for some applications, such as picking a set of candidate directions from S^d for gradient descent of a potential function; however, in general, we would like to a have a nice distribution of points for any value of n.

To the best of our knowledge, it is impossible to perfectly space n points around S^d, for any n and for $d > 1$. One simple idea that increases the number of samples is place one point in the center of each of the c d-cells of some regular polytope, and lift it to S^d. If we take the union of these points with the set of v polytope vertices, a nice point set of size $c + v$ may be obtained. If more points are placed; however, the problem becomes more complicated. Therefore, we are willing to tolerate some distortion in the distribution of points. It still seems useful, however, to borrow some of the properties of

the regular polytopes to generate good samples. The general idea pursued in this paper is to sample uniformly on the surface of the regular polytope, and then transform generated distribution on the surface of the sphere. We next describe this general method and discuss the induced distortion.

Consider a $(d+1)$-dimensional regular polytope inscribed in the sphere S^d. Suppose there exists a good method of sampling the surface of this polytope. The faces (d-dimensional cells) of the polytope, if projected outward to the surface of the sphere, form a tiling of the surface with the d-dimensional *spherical polytopes*. Consider some particular face, F, and its corresponding spherical face, F'. Each point inside F can be described by the barycentric coordinate systems induced by vertices of F after its triangulation. Now imagine that a distribution of points is generated inside F. Each of the points in this distribution can be obtained through several steps of linear interpolation between the vertices of the barycentric coordinate systems. The distribution on F' can be obtained then through similar steps of interpolating between the vertices of F', except that the interpolation should be done on the surface of the sphere [16]. This idea is similar to the one proposed in [1] for stratified sampling of spherical triangles. As an example, consider a cube inscribed in the sphere S^2, and sample the surface of the cube by placing a Sukharev grid [10, 19] on each face of the cube. Using the proposed method we get a distribution of samples on S^2 as shown on Figure 2.

The distribution of points on the sphere S^d obtained by this method will introduce distortion since spherical arcs corresponding to the intervals inside F with the same length may have different lengths in F'. The amount of the distortion, and therefore bounds on the dispersion and discrepancy, can be obtained through the analysis of the maximal arc differences.

This idea can also be adapted to $SO(3)$ (and in general to the projective space of any dimension). Take a four-dimensional regular polytope inscribed in S^3 and use only half of the faces to generate the distribution on the surface. We pick the faces so that in the set of used faces, there must not exist a pair of antipodal points, one from each of two different faces. This way the obtained samples will cover exactly half of the sphere, which forms $SO(3)$ surface.

While this approach can incorporate any uniform sampling method, we use the grid sequences described above and in [11]. In what follows we introduce the concept of a *layered Sukharev grid sequence*.

4.3 Layered Sukharev Grid Sequences for Spherical Cubes

In the construction of our basic grid sequences, we considered only classical multi-resolution grids. We can use these grid sequences to construct Sukharev grids; a Sukharev grid of k points per axis is identical to a classical grid, with the exception that the samples are placed in the center of the k^d grid regions rather than in the bottommost corner. We will be using a layered version of this sequence; a layered Sukharev grid of resolution l is the union of all Sukharev grids with 2^i points per axis, for $0 \leq i \leq l$. A *layered Sukharev grid*

sequence constructs the Sukharev grids one resolution at a time. Since each individual Sukharev grid is a shifted version of a classical grid, we can use the sequences previously described to generate the samples. In what follows we generalize layered Sukharev grid sequence to the sphere S^d. We first show how the points should be generated in each of the spherical cubes, and then how all these points can be combined into one sequence on the sphere.

Consider a face, F, of a $(d+1)$-cube inscribed in a sphere S^d. F is a d-dimensional cube, which in each of its corners has d edges. If we project all of these edges onto the surface of the sphere they form arcs, which delineate a spherical d-cube, F'. The lengths, α, of these arcs are equal for all edges of F. If we consider those equatorial angles that correspond to the edges coming from a common vertex of F, we can define an *angular coordinate system* for the spherical face F'. Indeed, the coordinates $(x_1, x_2, ...x_{n-1})$ with all possible values $x_i \in [0, \alpha]$ specify all possible points of F'.

The construction of the sequence, T, essentially follows the construction of the layered Sukharev grid sequence for the unit cube, except that instead of the Euclidean coordinate system we use the angular coordinate system defined above. We call this a *Sukharev spherical grid*. The dispersion of the resulting sequence can be calculated, as can the discrepancy, which is especially appropriate when using a discrepancy-optimal grid ordering to generate the Sukharev spherical grid. These results can be found in [23]. It should be fairly obvious that the time complexity of generating samples on a Sukharev spherical grid is the same as for generating ordinary grid samples. Consequently, Properties 1 and 2 from Section 3.3 hold.

4.4 Layered Sukharev Grid Sequences for S^d

Now that we have defined a sequence for each of the spherical cubes separately, we need to define an ordering by which these will be combined to form a sample sequence over the entire surface of the sphere. A straightforward way to do this is to place one point from each of the faces' sequences at a time. The order in which the faces should be considered can be explicitly computed using the same uniformity criteria used in computing the underlying grid sequence. This ordering, combined with the grid orderings of each individual spherical cube, yields the entire sphere sampling sequence.

5 Sample Sequences for Cross Products of $\mathbb{R}^n$ and S^d

We have defined the multiresolution grid sequences for the unit cube (Section 3), and the sphere S^d (Section 4). The spaces that arise in robotics are often the cross products of these. For example, the set of all rotations and translations of a 3d rigid body, denoted as $SE(3)$, can be represented by $\mathbb{R}^3 \times \mathbb{R}P^3$. The rotations and translations of m multiple rigid bodies is represented by $(\mathbb{R}^3 \times \mathbb{R}P^3)^m$.

When designing uniform sequences for such spaces the parameterization of the space together with the choice of measure and metric on the space should be defined carefully. For $SE(3)$ the Haar measure should be taken. However, since there is no natural metric on $SE(3)$, the weighted sum of the metrics on $\mathbb{R}^3$ and $SO(3)$ is usually used. While the weighted metric can be defined on general cross products of the spaces (which is assumed in the construction below), in some cases particular techniques for designing sequences might be advantageous.

In what follows we construct the multiresolution grid sequence for the space that is a cross product of multiple copies of $\mathbb{R}^n$ and S^d. We define it inductively, starting with any tuple of multiresolution grid sequences.

Let T_1 and T_2 be two multiresolution grid sequences. Let T_1 be defined over the space X_1, and T_2 be defined over the space X_2. Either of X_1 or X_2 may be $\mathbb{R}^n$ or S^d. Let $\dim(T_1) = d_1$ and $\dim(T_2) = d_2$ be the dimensions of these sequences. Let m_1 (m_2) be the number of points at the resolution level 0 of sequence T_1 (T_2) respectively. When weighted metric is defined on $X_1 \times X_2$ the values for m_1 and m_2 can be chosen so that they respect the appropriate weights of X_1 and X_2. Then the number of points at the resolution level l is $m_1 \cdot 2^{ld_1}$ and $m_2 \cdot 2^{ld_2}$ for sequence T_1 and T_2 respectively.

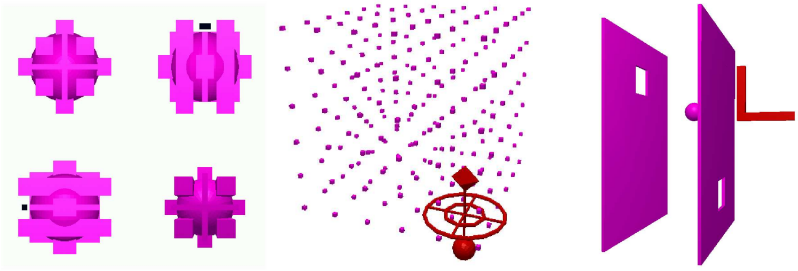
With each point $p = (p_1, ...p_d)$ at the resolution level l one may define the grid region associated with this point as $G_{p,l} = \left[p_1, p_1 + a/2^l\right) \times ... \times \left[p_d, p_d + a/2^l\right)$, in which $a = 1$ for the unit cube sequence and $a = \alpha$ (Section 4) for the sphere sequence.

Next consider the space $X = X_1 \times X_2$. The multiresolution grid sequence that we define for this space has $m_1 \cdot m_2 \cdot 2^{l(d_1+d_2)}$ points at the resolution level l. Each of these points can be expressed as $p = (p_1, p_2) = (p_{1,1}, ...p_{1,d_1}, p_{2,1}, ...p_{2,d_2})$, where $p_1 \in T_1$ and $p_2 \in T_2$. Each of these points has an associated grid region: $G_{p,l} = \left[p_{1,1}, p_{1,1} + a/2^l\right) \times ... \times \left[p_{1,d_1}, p_{1,d_1} + a/2^l\right) \times \left[p_{2,1}, p_{2,1} + b/2^l\right) \times ... \times \left[p_{2,d_2}, p_{2,d_2} + b/2^l\right)$.

The sequence for X is constructed one resolution level at a time. The order in which the points from each resolution level are placed in the sequence can be described as follows. The ordering, $L_X()$, of the first $m_1 \cdot m_2 \cdot 2^{(d_1+d_2)}$ points determine the order of the grid regions whithin X and should be precomputed in advance. Every successive $m_1 \cdot m_2 \cdot 2^{(d_1+d_2)}$ points in the sequence should be placed in these grid regions in the same order. Where exactly each point should be placed within each of the grid regions should be determined by the recursion procedure defined for $[0, 1]^{(d_1+d_2)}$.

6 Experiments

We have implemented our algorithm in C++ and applied to implementations of PRM-based planner [6] in the Motion Strategy Library. The experiments reported here were performed on a 2 Ghz Pentium IV running Linux and compiled under GNU C++.



	Random Sequence	Layered Sukharev Grid Sequence
a)	1088	1067
b)	3460	3285
c)	3481	3202

Fig. 3. Problems involving: a) moving a robot (black) from the north pole to the south pole. Multiple views of the geometry of the problem are shown (obstacles are drawn in lighter shades); b) moving a robot from one corner of a 3d grid to the opposite corner; c) moving an L-shaped object through the holes in the obstacles. The comparisons of the number of nodes generated by different sampling strategies are shown in the table.

Performance results are shown in Figure 3. The robot in the model a) is allowed only to rotate; therefore, the configuration space is $\mathbb{R}P^3$. The robots in the models b),c) are allowed to translate and rotate; therefore the configuration space is $\mathbb{R}^3 \times \mathbb{R}P^3$. We compared the number of nodes generated by the basic PRM planner using a pseudo-random sequence (with quaternion components [17]), and the layered Sukharev grid sequences. The results for pseudo-random sequences were averaged over 50 trials. When we tested the deterministic sequences, we made sure that each particular problem does not have any advantage due to coincidental alignment with the grid directions of the sequence. Therefore, in each trial a fixed, random quaternion rotation was premultiplied to each sample, to displace the entire sequence. The results obtained were averaged over 50 trials (a different random rotation was used in each).

Based on our experiments we have observed that the performance of the deterministic sequences is equivalent to the performance of the random sequences for the PRM-based planner, which makes it an alternative approach to random sampling. It is important to note, however, that for some applications, such as verification problem, only deterministic guarantees are acceptable, making random sequences inappropriate.

7 Conclusions and Future Work

In conclusion, we have presented a family of new grid-based sample sequences. These sequences have the advantageous criteria of uniformity, lattice structure, and incremental quality. Our sample sequences, based on optimizing dispersion or mutual distance measures, have very compact representations and hence are applicable to high-dimensional problems, unlike the grid-based sequences of [11]. In addition to this, we have introduced methods for applying sampling methods designed for the unit cube to such topological spaces as $SO(3)$ and $SE(3)$. The sequences presented are widely applicable to algorithms in robotics and motion planning. Like traditional Monte Carlo or quasi-Monte Carlo sampling techniques, these sequences are uniform and have high incremental quality; hence, they may be easily substituted into existing algorithms in place of other sampling methods. In addition, these sequences have implicit lattice structure, which permits the development of new algorithms designed to exploit that structure.

There are several directions for future work. First, we plan to actually compute these orderings for very high dimensions. We believe that using approximation and heuristic search techniques, we will be able to find good (though not optimal) orderings for up to several thousand dimensions. Second, the sampling method presented for $SE(3)$ hints at a more general approach to forming products of multi-resolution sample sequences. We hope to study this more carefully and formally characterize how to find such sequence products. Finally, we plan to develop algorithms which specifically exploit the regularity implicit in grid sample sequences. By integrating the sampling method more tightly with planning rather than viewing it simply as a black box, it may be possible to see significant advantages not otherwise available.

Acknowledgments

This work was funded in part by NSF Awards 9875304, 0118146, and 0208891.

References

1. J. Arvo. Stratified sampling of spherical triangles. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 437–438, 1995.
2. M. Blümlinger. Slice discrepancy and irregularities of distribution on spheres. *Mathematika*, 38:105–116, 1991.
3. J. H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numer. Math.*, 2:84–90, 1960.
4. J. M. Hammersley. Monte-Carlo methods for solving multivariable problems. *Ann. New York Acad. Sci.*, 86:844–874, 1960.
5. 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.

6. 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.
7. N.M. Korobov. The approximate computation of multiple integrals. *Dold. Akad. Nauk SSR*, 124:1207–1210, 1959.
8. S. M. LaValle. *Planning Algorithms*. [Online], 1999-2003. Available at http://msl.cs.uiuc.edu/planning/.
9. 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*, December 2002.
10. 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.
11. 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.
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. 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.
15. G. Rote and R. F. Tichy. Spherical dispersion with applications to polygonal approximation of the curves. *Anz. Österreich. Akad. Wiss. Math.-Natur, Kl. Abt. II*, 132:3–10, 1995.
16. Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254. ACM Press, 1985.
17. Ken Shoemake. Uniform random rotations. In D. Kirk, editor, *Graphics Gems III*, pages 124–132. Academic Press, 1992.
18. I. H. Sloan and S. Joe. *Lattice Methods for Multiple Integration*. Oxford Science Publications, Englewood Cliffs, NJ, 1990.
19. 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.
20. J. G. van der Corput. Verteilungsfunktionen I. *Akad. Wetensch.*, 38:813–821, 1935.
21. X. Wang and F. J. Hickernell. An historical overview of lattice point sets. In K.-T. Fang, F. J. Hickernell, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2000*, pages 158–167. Springer-Verlag, Berlin, 2002.
22. H. Weyl. Über die Gleichverteilung von Zahlen mod Eins. *Math. Ann.*, 77:313–352, 1916.
23. A. Yershova and S. M. LaValle. Deterministic sampling methods for spheres and SO(3). In *Proc. IEEE International Conference on Robotics and Automation*, 2004.