A General Deterministic Sequence for Sampling *d*-dimensional Configuration Spaces

Jan Rosell*, Máximo Roa, Alexander Pérez, Fernando García**

Institute of Industrial and Control Engineering (IOC-UPC)
Barcelona, SPAIN, Email: jan.rosell@upc.edu

Received: date / Revised version: date

Abstract Previous works have already demonstrated that deterministic sampling can be competitive with respect to probabilistic sampling in sampling-based path planners. Nevertheless, the definition of a general sampling sequence for any d-dimensional Configuration Space satisfying the requirements needed for path planning is not a trivial issue. This paper makes a proposal of a simple and yet efficient deterministic sampling sequence based on the recursive use, over a multi-grid cell decomposition, of the ordering of the 2^d descendant cells of any parent cell. This ordering is generated by the digital construction method using a $d \times d$ matrix T_d. A general expression of this matrix (i.e. for any d) is introduced and its performance analyzed in terms of the mutual distance. The paper ends with a performance evaluation of the use of the proposed deterministic sampling sequence in different well known path planners.

1 Introduction

Configuration Space has been the planning paradigm that has nurtured the field of robotic path planning since its beginnings. The explicit characterization of the obstacles of the $\mathcal{C}$-space is not possible when the number of degrees of freedom of the robot is high and, therefore, sampling-based methods are usually proposed. These methods only require the collision evaluation of a discrete set of sample configurations and the interconnection of the free ones in either roadmaps or trees. Therefore, the generation

* *This work was partially supported by the CICYT projects DPI2004-03104 and DPI2005-00112.*

** *M. Roa is supported by Alβan (European Union Programme of High Level Scholarships for Latin America) under Scholarship No. E04D039103CO.*

of samples is one of the crucial factors in the performance of sampling-based planners.

Random sampling using a uniform distribution is usually considered, like in the basic Probabilistic Roadmap Method (PRM [3]) or the basic Rapidly-exploring Random Trees approach (RRT [5]); although deterministic sampling sequences have also been proposed [1]. Deterministic sampling sequences have the advantages of classical grid search approaches, i.e. a lattice structure (that allows to easily determine the neighborhood relations) and a good uniform coverage of the $\mathcal{C}$-space. Deterministic sampling sequences applied to PRM-like planners have given good results compared to the basic PRM planner [6].

For difficult path-planning problems, importance sampling methods are required to bias the sampling towards some given areas of the $\mathcal{C}$-space, and reduce the amount of samples needed to find a solution (e.g. [7,2]). These strategies usually rely on a uniform generation of samples that are then filtered by different criteria to select samples in critical regions; therefore, deterministic sampling can also be used for that purpose.

This paper has as objective the proposal of a general deterministic sampling sequence and its performance evaluation. Section 2 and 3 introduce the deterministic sampling sequence. Section 4 makes a performance analysis and Section 5 presents its use on different path planners. Finally, Section 6 concludes the work.

2 Deterministic sampling sequence

The deterministic sampling sequence used is based on a multi-grid cell decomposition, introduced in Section 2.1, and on the low-dispersion ordering of the descendant cells of any given parent cell, introduced in Section 2.2. The expression of the sequence is given in Section 2.3 and Section 2.4 discusses the mapping of samples to configurations of $\mathcal{C}$-space.

2.1 Multi-grid cell decomposition

A multi-grid decomposition of a d-dimensional space of parameters is considered. An initial cell with sides of unitary size covering the entire space composes the first grid. The levels in the multi-grid are called partition levels and are enumerated such that the first one is level 0 and the maximum resolution[1] corresponds to partition level M (partition levels are denoted by super-indices). A cell of a given partition level M is called a sample or an M-cell; and is coded as follows. Let:

[1] The maximum resolution needed is a fixed value determined by the clearance of the path planning problem to be solved.

– The index matrix V^M be the binary $d \times M$ matrix whose rows are the binary representation of the indices $v_j^M \ \ \forall j \in 1 \dots d$ of an M-cell on the regular grid of partition level M:

$$V^M = \begin{pmatrix} v_1^M \\ \vdots \\ v_j^M \\ \vdots \\ v_d^M \end{pmatrix} = \begin{pmatrix} a_{M1} \dots a_{i1} \dots a_{11} \\ \vdots \quad\quad \vdots \quad\quad \vdots \\ a_{Mj} \dots a_{ij} \dots a_{1j} \\ \vdots \quad\quad \vdots \quad\quad \vdots \\ a_{Md} \dots a_{id} \dots a_{1d} \end{pmatrix} \tag{1}$$

being a_{Mj} and a_{1j} the most and the least significant bit, respectively, of the binary representation of v_j^M.

– W^M be a $d \times M$ matrix of weights:

$$W^M = \begin{pmatrix} w_{11} \dots w_{1j} \dots w_{1M} \\ \vdots \quad\quad \vdots \quad\quad \vdots \\ w_{i1} \dots w_{ij} \dots w_{iM} \\ \vdots \quad\quad \vdots \quad\quad \vdots \\ w_{d1} \dots w_{dj} \dots w_{dM} \end{pmatrix} \tag{2}$$

with:

$$w_{ij} = 2^{(M-j)d+i-1} \text{ for } i \in 1 \dots d, \ \ j \in 1 \dots M \tag{3}$$

Then, the sample code C^M and its index matrix V^M are related as follows:

$$C^M = V^M \cdot W^M \tag{4}$$

$$V^M = C^M \& W^M \tag{5}$$

where the operation $A \cdot B$ represents the scalar product of matrices A and B, and the operation $a \& B$ between a scalar a and a matrix B computes the bit-AND operation between a and all the components b_{ij} of B. As an example the conversion operations of cell code 22 with indices (6,1) on the grid of partition level $M = 3$ (Fig. 1a) are:

$$C^3 = V^3 \cdot W^3 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 16 & 4 & 1 \\ 32 & 8 & 2 \end{pmatrix} = 22 \tag{6}$$

$$V^3 = C^3 \& W^3 = 22 \& \begin{pmatrix} 16 & 4 & 1 \\ 32 & 8 & 2 \end{pmatrix} =$$

$$= 010110 \& \begin{pmatrix} 010000 & 000100 & 000001 \\ 100000 & 001000 & 000010 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{7}$$

For any m-cell, with $m < M$, the cell code coincides with that of the first M-cell it contains (i.e. the descendant M-cell with lower cell code), as illustrated in Fig. 1b.

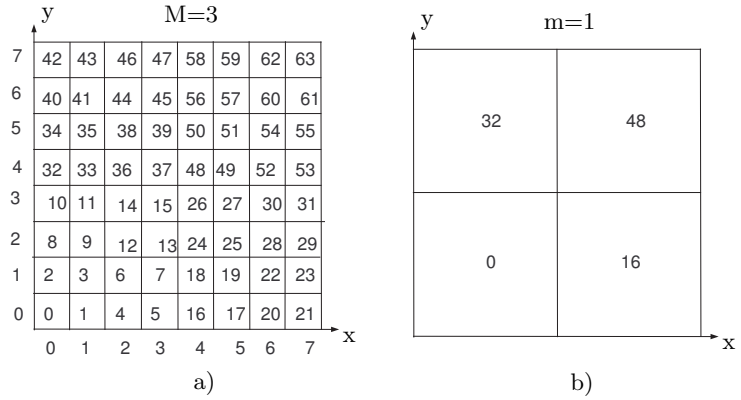

Fig. 1 Coding for $M = 3$ of: a) M-cells b) 1-cells.

2.2 Ordering of descendant cells

The position of a cell with respect to its parent cell can be defined by a binary word, i, with d bits, each one determining the index (0 or 1) over the corresponding axis, i.e. $i = \sum_{j=1}^{d} n_j 2^{j-1}$ (Fig. 2).

Finding a low-dispersion ordering of the 2^d descendant cells of a parent cell is then equivalent to find a sequence, L_d, of 2^d binary words such that each element of the sequence maximizes the mutual distance, i.e. the minimum distance to the previous elements of the sequence. This criterion is further discussed in Section 4.1.

L_d can be obtained using a digital construction method [8] that finds the sequence multiplying a $d \times d$ binary matrix, T_d, by the binary representation of the indices of the sequence:

$$L_d(i) = T_d \;\; i = T_d \begin{pmatrix} n_1 \\ \vdots \\ n_d \end{pmatrix} \tag{8}$$

An example of T_d for $d = 2$ and $d = 3$ is:

$$T_2 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \qquad T_3 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{9}$$

and the corresponding orderings obtained are shown in Table 1. A proposal for T_d is introduced in Section 3.

Table 1 Ordering L_d for $d = 2$ and $d = 3$.

i	$L_2(i) = T_2\ i$
0	$\begin{pmatrix}1&0\\1&1\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix} = \begin{pmatrix}0\\0\end{pmatrix} = 0$
1	$\begin{pmatrix}1&0\\1&1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\1\end{pmatrix} = 3$
2	$\begin{pmatrix}1&0\\1&1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix} = 2$
3	$\begin{pmatrix}1&0\\1&1\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix} = 1$

i	$L_3(i)$
0	0
1	5
2	3
3	6
4	4
5	1
6	7
7	2

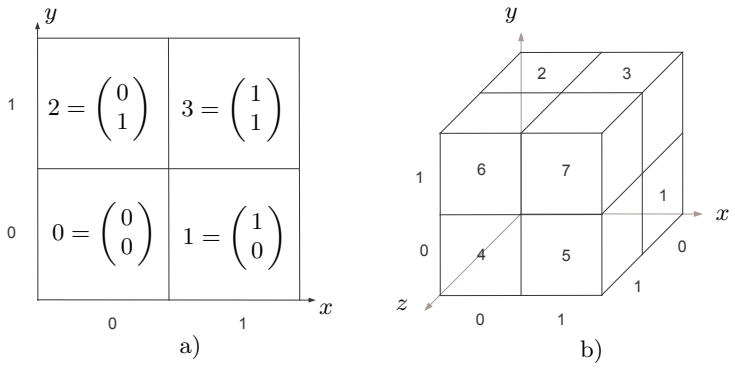

Fig. 2 Position i of a cell with respect to its parent cell for two and three dimensional spaces.

2.3 The sampling sequence

The sampling sequence, $s_d(k)$, is a sequence of sample codes that specifies the ordering in which the d-dimensional space is to be explored. The sequence $s_d(k)$ is based on the recursive use of L_d.

Let $k \geq 0$ be the index of the sequence and T_d be the matrix that determines the cell ordering of the descendant cells as introduced in the previous section. Then:

$$s_d(k) = (T_d V_k^M) \cdot W'^M \tag{10}$$

where V_k^M is the index matrix corresponding to k, the product $T_d V_k^M$ is the standard binary matrix multiplication between matrices T_d and V_k^M, and W'^M is a $d \times M$ matrix of weights, with:

$$w'_{ij} = 2^{(j-1)d+i-1} \text{ for } i \in 1 \ldots d \;\; j \in 1 \ldots M \tag{11}$$

(Note that matrix W'^M coincides with W^M if the order of its columns is exchanged).

Table 2 First 20 samples of sequence s_2.

k	0	1	2	3	4	5	6	7	8	9
$s_2[k]$	0	48	32	16	12	60	44	28	8	56
k	10	11	12	13	14	15	16	17	18	19
$s_2[k]$	40	24	4	52	36	20	3	51	35	19

Table 3 First 10 samples of r_2^{48}.

k	0	1	2	3	4	5	6	7	8	9
$r_2^{48}[k]$	48	60	56	52	51	63	59	55	50	62

As an example, considering $M = 3$ and the expression of T_2 proposed in (9), the sample for $k = 6$ is:

$$\begin{aligned} s_2(6) &= \left[\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\right] \cdot \begin{pmatrix} 1 & 4 & 16 \\ 2 & 8 & 32 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 16 \\ 2 & 8 & 32 \end{pmatrix} = 44 \end{aligned} \tag{12}$$

The first 20 samples generated by $s_2(k)$ are shown in Table 2. Following the sequence over Fig. 1a gives a good understanding of how it incrementally and uniformly covers the space.

If only the samples of a given cell are necessary, they can be obtained with the following (re)sampling sequence. Let m_K be the partition level of that cell and K be its code. Then:

$$r_d^K(j) = K + (T_d V_j^{(M-m_K)}) \cdot W'^{(M-m_K)} \text{ with } j \geq 0 \tag{13}$$

As an example, the sixth sample generated by $r_2(k)$ over the 1-cell 48 (Fig. 1b) is:

$$\begin{aligned} r_2^{48}(6) &= 48 + \left[\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right] \cdot \begin{pmatrix} 1 & 4 \\ 2 & 8 \end{pmatrix} \\ &= 48 + \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 \\ 2 & 8 \end{pmatrix} = 48 + 11 = 59 \end{aligned} \tag{14}$$

The first 10 samples generated by $r_2^{48}(k)$ are shown in Table 3. Following the sequence over Fig. 1a shows the same incremental and uniform coverage feature.

2.4 Mapping to $\mathcal{C}$-space

The deterministic sampling sequence proposed obtains samples of a d-dimensional unit cube of parameters, that must be appropriately mapped to configurations of a particular $\mathcal{C}$-space.

For the path planning of robot manipulators of d d.o.f., the $\mathcal{C}$-space is $\Re^d$. Taking into account the maximum displacement of each joint, the configuration coordinates can be scaled to the unit cube $[0,1]^d \subset \Re^d$. Then, the indices $(v_1^M, \ldots, v_d^M)$ on the grid of partition level M of the samples obtained by $s_d(k)$ are mapped to coordinates of the $\mathcal{C}$-space as follows:

$$x_j = v_j^M s_M + \frac{s_M}{2} \qquad \forall j \in 1 \ldots d \tag{15}$$

being $s_M = \frac{1}{2^M}$ the size of the sides of the M-cells.

For the path planning of 3D rigid-bodies that can both translate and rotate the mapping must be carefully handled, due to the difficulty to uniformly distribute samples over the rotation group, as discussed in [4].

3 Proposal for T_d

In the digital construction method introduced in [8] the matrix T_d is proposed to be built by columns in an incremental way, but no general expression or procedure is given except for the first column that must be composed of ones in order to place the second sample as far as possible from the first one.

A first attempt to obtain a general expression was done in [9], where the proposed T_d (called T_d^A) gives an ever non-increasing mutual distance for the elements of L_d, although the maximization of the mutual distance is not guaranteed. In this paper a new proposal for T_d (called T_d^C) is introduced and its performance is evaluated and compared to that of [9].

3.1 Matrix T_d^A

This matrix is proposed in [9] and is constructed as follows. Each column $j \in 1 \ldots d$ is composed of $(j-1)$ zeros followed by a 1 (that corresponds to the diagonal). The rest of the column is filled by alternating $(j-1)$ zeros with $(j-1)$ ones until the column is completed. The resulting T_d is a lower diagonal matrix with non-zero diagonal elements, and therefore is full rank. As a consequence, T_d is able to generate all the 2^d elements of the sequence.

As an example, for $d = 9$ the matrix is:

$$T_9^A = \begin{pmatrix} 1&0&0&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&0&0 \\ 1&1&0&1&0&0&0&0&0 \\ 1&0&0&0&1&0&0&0&0 \\ 1&1&1&0&0&1&0&0&0 \\ 1&0&1&0&0&0&1&0&0 \\ 1&1&0&1&0&0&0&1&0 \\ 1&0&0&1&0&0&0&0&1 \end{pmatrix} \tag{16}$$

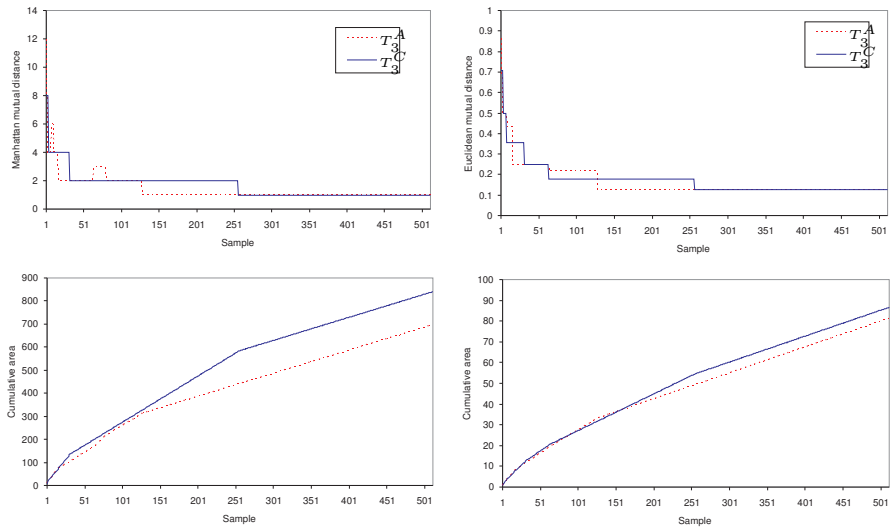

Fig. 3 Top: Mutual distances (Manhattan (left) and Euclidean (right)) in the sampling of $\Re^3$ with $M = 3$; Bottom: Cumulative area in the sampling of $\Re^3$ (Manhattan (left) and Euclidean (right)).

3.2 Matrix T_d^C

This matrix is based on a prime decomposition. For each prime dimension, i.e. $d = 2, 3, 5, 7, \ldots$, T_d is first defined:

$$T_2^C = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \tag{17}$$

$$T_3^C = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{18}$$

$$T_d^C = \text{Trunc}_d(T_{(d+1)}^C) \;\; \forall d \text{ prime s.t. } d \geq 5 \tag{19}$$

where the function $\text{Trunc}_d(M)$ returns the submatrix of M composed of the first d columns and the first d rows.

Then, for any d, a recursive construction is done based on the prime decomposition of d, e.g.:

$$T_6^C = \begin{pmatrix} T_3^C & 0 \\ T_3^C & T_3^C \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \tag{20}$$

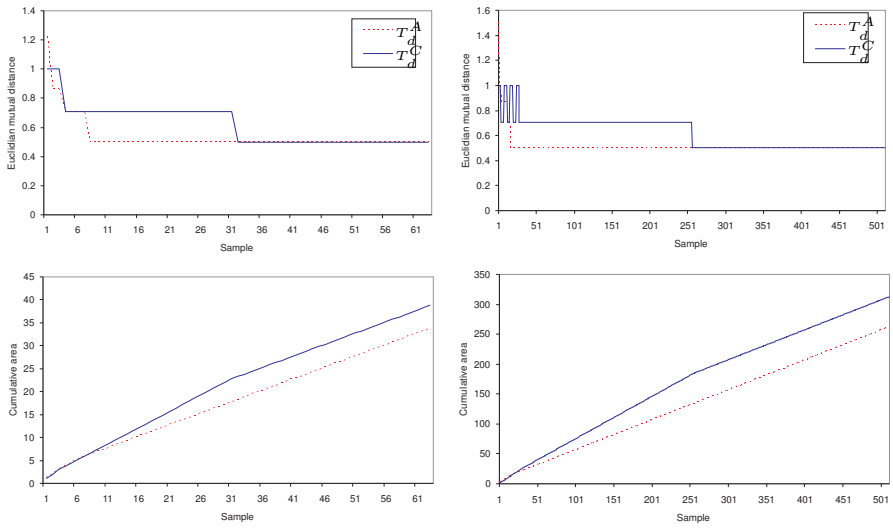

Fig. 4 Top: Mutual Euclidean distances in the sampling of $\Re^6$ (left) and $\Re^9$ (right) with $M = 1$; Bottom: Corresponding cumulative areas.

$$T_9^C = \begin{pmatrix} T_3^C & T_3^C & 0 \\ 0 & T_3^C & 0 \\ T_3^C & 0 & T_3^C \end{pmatrix} \tag{21}$$

It can be seen that this approach results with a first column not composed exclusively of ones. This makes that the second sample is not as far as possible from the first one, i.e. the second sample does not maximize the mutual distance. Nevertheless, as shown in the next section, in the long run it results in a better performance, i.e. in a slower decreasing of the mutual distance.

4 Comparative study

4.1 Performance index

Different uniformity measures have been proposed in the literature [6]. Perhaps the most useful in path planning is dispersion, a metrics-based criterion. Let $X = [0,1]^d \subset \Re^d$ be the space where to generate samples. Let P be the set of samples taken from X. Then, dispersion for P is defined as:

$$\delta(P,\rho) = \sup_{q \in X} \min_{p \in P} \rho(q,p) \tag{22}$$

being ρ any metrics on X. Intuitively, it corresponds to the largest empty ball inscribed in X and with the center in X. A criterion that emphasizes

dispersion requires that points be as far away from each other as possible, e.g. the mutual distance of the samples must be maximized [8]. Mutual distance on P is defined as:

$$\rho_m(P) = \min_{x,y \in P} \rho(x, y) \tag{23}$$

The mutual distance can be considered with different metrics:

- Manhattan distance between the cells codes in the parameters space.
- Euclidean metrics between the samples in the $\mathcal{C}$-space, in the case of $\Re^d$.

In this paper the cumulative area of the mutual distance is used as the basic performance index, since it shows how far the samples are being placed from each other as a progression in the sample number. This allows the visualization of the gain obtained by one method compared to another.

4.2 Sampling in $\Re^3$

The first experiment is the comparison between the use of T_d^C and T_d^A in the sampling of $\Re^3$. Using $M = 3$, Fig. 3 (top) shows the Manhattan mutual distance (left) and the mutual distance measured with Euclidean metrics (right) computed for all the 512 samples of the sequence. The continuous blue line corresponds to T_3^C and the red point line to T_3^A. It can be seen in both figures that the continuous line decreases more slowly showing that the approach that uses T_3^C gets a better uniform coverage of the space as samples are placed. This is better shown in Fig. 3 (bottom) where the cumulative area of the mutual distance is represented.

4.3 Sampling in $\Re^6$ *and* $\Re^9$

The last experiments are the comparisons between the use of T_d^C and T_d^A in the sampling of $\Re^6$ and $\Re^9$. Using $M = 1$, Fig. 4 (top) shows the Euclidean mutual distances for all the 64 and 512 samples of the corresponding sequences. Fig. 4 (bottom) shows the cumulative areas. It can be seen that in both cases, like in $\Re^3$, T_d^C also provides the best uniform coverage of $\mathcal{C}$-space.

5 Examples

In this Section the proposed deterministic sequence is applied to different path planners. Although an exhaustive study has not been done, the results show a better performance when the proposed deterministic sequence is used.

Table 4 Comparison using the PRM of the Motion Strategy Library.

	Halton	Hammersley	Random	$s_6(k)$
PRM Vertices	141	144	139 ± 10	126
PRM Edges	278	284	270 ± 23	250
Num components	2	2	2.4 ± 1.4	1
Col. Detection	13,637	14,027	$12,864 \pm 1,577$	6,314
Construct Time	1.41s	1.65s	$1.52 \pm 0.35s$	0.64s

Table 5 Comparison using the SBL planner.

	Random	$s_9(k)$
Num. Vertices	219.63 ± 70.70	118.71 ± 28.24
Construct Time	$0.38 \pm 0.10s$	$0.25 \pm 0.04s$

5.1 Comparison with other deterministic sequences

The Motion Strategy Library is an Open Source code for the testing of motion planning algorithms, available at http://msl.cs.uiuc.edu/msl/. The library includes planners based on Rapidly-exploring Random Trees (RRTs) and on Probabilistic Roadmaps (PRMs). It has the utility of selecting random sampling or deterministic sampling based on the Halton or Hammersley sequences. It has already been used in [6] to evaluate deterministic sampling vs. random sampling using the basic PRM.

The library has been extended to include the deterministic sampling proposed in the present paper. The results using 300 samples of a basic PRM to solve a simple path planning problem for a 6 d.o.f. manipulator are shown in Table 4 and Fig. 5. It can be seen that using the same number of samples, the proposed sequence outperforms other sampling methods, allowing the PRM to be constructed more quickly using less edges and vertices, and being all of them connected on a single component.

5.2 Use in local sampling

The Motion Planning Kit (MPK) is a C++ library and toolkit for developing single and multi-robot motion planners, available at http://ai.stanford.edu/~mitul/mpk/home.html. It includes a Single-Query Bi-Directional Probabilistic Roadmap Planner (SBL [10]). This planner expands two trees rooted at the queried configurations, using for the node expansion a local sampling procedure over a hypercube centered at the node to be expanded and covering the 30% of the total $\mathcal{C}$-space.

The SBL planner has been extended to incorporate (in the local sampling expansion procedure) the deterministic sampling sequence proposed, instead of using pure random sampling. Table 5 compares the results obtained for the motion planning of a 6 d.o.f. Puma robot mounted over a 3 d.o.f. mobile

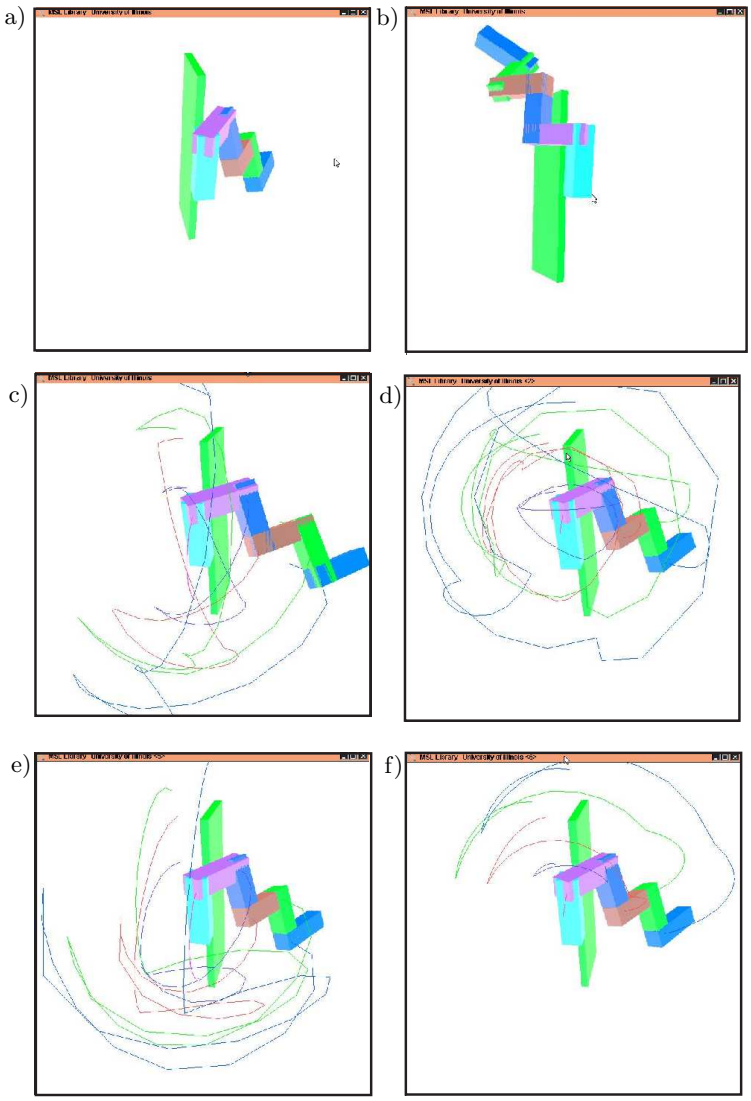

Fig. 5 a) Initial configuration; b) final configuration; c) path using the Halton sequence; d) path using the Hammersley sequence; e) path using a random sequence; f) path using the proposed sequence $s_6(k)$ with T_6^C.

platform in a particular environment[2]. Fig. 6a shows a typical path obtained

[2] Since the SBL planner randomly selects some directions to project the sampled configurations, the results using deterministic sampling also vary between executions. The results shown in Table 5 are the 95% confidence intervals obtained from 50 executions.

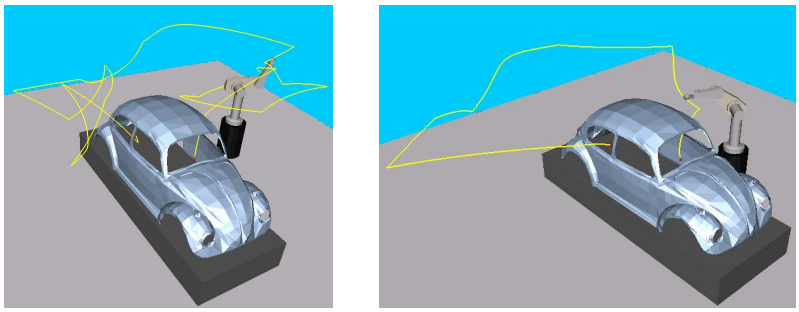

Fig. 6 Sample solution paths of a 9 d.o.f. path planning problem found using random sampling (left) and deterministic sampling (right).

with the original random SBL and Fig. 6b shows a path followed when using the deterministic sequence $s_9(k)$ with T_9^C. The deterministic sequence enhances the planner's performance, building the path with less milestones and in a shorter time than with the original expansion procedure using random sampling.

5.3 Use in $\mathcal{C}$-space modeling

The Kautham planner is a path planner, being developed by the authors, that uses deterministic sampling to generate a hierarchical cell decomposition model of $\mathcal{C}$-space where harmonic functions are computed and used to find a solution. Fig. 7a shows a 2D-example that illustrates the quadtree generated from the sampling and classification of configurations using the proposed deterministic sequence. Fig. 7b shows the same number of samples obtained by random sampling. It is obvious that this latter set could not generate a proper quadtree of the $\mathcal{C}$-space.

6 Conclusions

The performance of sampling-based path planners relies on the set of samples used. For uniform sampling, deterministic sampling sequences are a good alternative since they outperform probabilistic methods in terms of the dispersion obtained.

This paper proposes a simple and yet efficient deterministic sampling sequence computed on a multi-grid decomposition of a parameter space and the corresponding mapping to the $\mathcal{C}$-space. The expression given is general for any d-dimensional Configuration Space.

The deterministic sampling sequence introduced has been validated by the good results obtained when applied to different path planners.

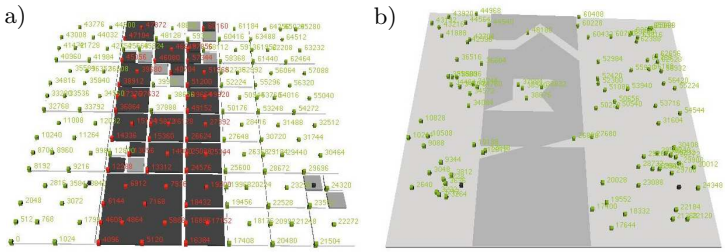

Fig. 7 a) Quadtree obtained from 250 samples of the proposed deterministic sampling sequence; b) Distribution of the same number of samples obtained randomly that shows that the quadtree could not have been properly build using this set of samples.

References

1. M. S. Branicky, S. M. LaValle, K. Olson, and L. Yang. Quasi-randomized path planning. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 1481–1487, 2001.
2. D. Hsu, T. Jiang, J. Reif, and Z. Sun. The bridge test for sampling narrow passages with probabilistic roadmap planners. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 4420–4426, 2003.
3. L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. K. Overmars. Probabilistic roadmaps for path planning in high - dimensional configuration spaces. *IEEE Trans. on Robotics and Automation*, 12(4):566–580, August 1996.
4. J. Kuffner. Effective sampling and distance metrics for 3D rigid body path planning. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 3993–3998, 2004.
5. J. J. Kuffner and S. M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 995–1001, 2000.
6. S. M. LaValle, M. S. Branicky, and S. R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *Int. J. of Robotics Res.*, 23(7-8):673–692, 2004.
7. P. Leven and S. Hutchinson. Using manipulability to bias sampling during the construction of probabilistic roadmaps. *IEEE Trans. on Robotics and Automation*, 19(6):1020–1026, 2003.
8. S. R. Lindemann, A. Yershova, and S. M. LaValle. Incremental grid sampling strategies in robotics. In *Proc. of the Sixth Int. Workshop on the Algorithmic Foundations of Robotics*, 2004.
9. J. Rosell and M. Heisse. An efficient deterministic sequence for sampling-based motion planners. In *Proc. of the IEEE Int. Symp. on Assembly and Task Planning*, 2005.
10. G. Sanchez and J.C. Latombe. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In *Proc. of the 9th Int. Symposium on Robotics Research*, 2001.