Noname manuscript No.
(will be inserted by the editor)

Direct Blue Noise resampling of meshes of arbitrary topology

Jean-Luc Peyrot · Frédéric Payan · Marc Antonini

Received: date / Accepted: date

Abstract We propose in this paper a novel sampling method and an improvement of a spectral analysis tool that both handle complex shapes and sharp features. Starting from an arbitrary triangular mesh, our algorithm generates a new sampling pattern that exhibits blue noise properties. The fidelity to the original surface being essential, our algorithm preserves sharp features. Our sampling is based on a discrete dart throwing applied directly on the surface to get good blue noise sampling patterns. It is also driven by a feature detection tool - to avoid geometric aliasing. Experimental results prove that our sampling scheme is faster than techniques based on brute-force dart throwing, and produces sampling patterns with blue noise properties even for complex surfaces of arbitrary topology. In parallel, we also propose an improvement of a tool initially developed for the spectral analysis of non-uniform sampling patterns, that may generate biased results with complex shapes. The proposed improvement overcomes this problem.

This work is supported by a grant from *Région Provence Alpes Côte d'Azur* (France).

Jean-Luc Peyrot, Frédéric Payan and Marc Antonini
Laboratory I3S - University Nice - Sophia Antipolis - CNRS (UMR 7271)
2000, route des Lucioles
06909 Sophia Antipolis, France
Tel.: +33 4 92 94 27 01 - Fax: +33 4 92 94 26 80

Jean-Luc Peyrot
E-mail: peyrot@i3s.unice.fr

Frédéric Payan
E-mail: fpayan@i3s.unice.fr

Marc Antonini
E-mail: am@i3s.unice.fr

Keywords blue noise · sampling · sampling analysis · feature-preservation

1 Introduction

In recent years, many works focused on the sampling patterns of surfaces. Given its capability to avoid aliasing artifacts [11], Poisson disk distribution has received considerable attention in computer graphics. Indeed, a Poisson disk distribution generates patterns that satisfy a uniform but irregular distribution of the samples within the domain: a minimum distance is ensured between samples, but with the constraint that samples do not lie on a spatial regular lattice. Such sampling patterns exhibit the so-called *blue noise* properties [19].

To analyze the spectral quality of these patterns, we generally study the power spectrum (Figure 1 on the left) that represents the distribution of distances between samples. From the power spectrum, two statistics are usually derived: the radially averaged power spectrum (RAPS) (Figure 1 in the middle) and the anisotropy (Figure 1 on the right). The RAPS assesses the radial distribution of the distances between samples, while the anisotropy evaluates the radial uniformity of the sampling pattern over the domain.

An ideal Poisson disk distribution presents a RAPS similar to a step function (see Figure 1 in the middle). It consists in a wide zero-region at low frequencies and a flat high-frequency region, both connected with a sharp transition at the cut-off frequency, that corresponds to the minimum distance between samples. The ideal anisotropy is constant, and very low. These characteristics are particularly relevant for many applications such as rendering, imaging, texturing, geometry processing and numerical simulations [15], [27], [33].

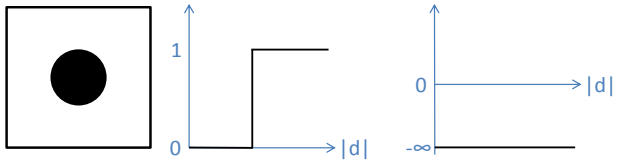


Fig. 1 From left to right: The power spectrum, the radially averaged power spectrum (RAPS) and the anisotropy of an ideal Poisson disk distribution.

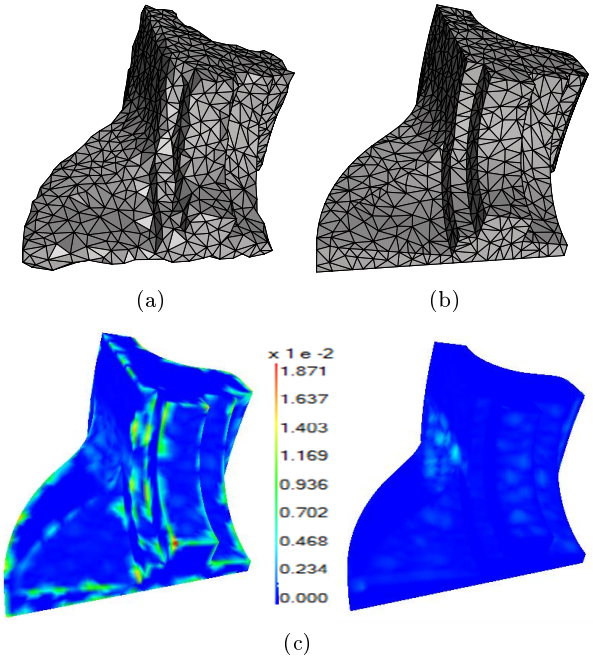


Fig. 2 FANDISK sampled and triangulated respectively, without (a) and with (b) preservation of the features. (c) Geometric errors with respect to the initial surface, computed with MESH [2].

Motivation and contributions

Currently, few blue noise sampling techniques handle surfaces with feature lines [17], [18], [25]. However, sampling surfaces without taking into account the feature lines may generate the so-called *geometric aliasing*: see Figure 2. Moreover, many sampling techniques do not handle complex shapes, for instance high-genus surfaces, because the sampling is not done directly on the input surface, but in a parameter domain.

Our objective consists in resampling surface meshes, by optimizing their number of vertices while having good spectral properties. We propose in this paper a blue noise sampling technique that takes a triangle mesh as input, and produces a new sampling pattern satisfying the blue noise properties. Our sampling is based on direct brute-force dart throwing - to process arbitrary topologies - and uses geodesics [10] - to guarantee the sampling quality even for complex shapes. Our algorithm also includes a segmentation technique that detects sharp features. This technique avoids potential aliasing artifacts, and finally strengthens the fidelity to the original shape geometry.

The core idea of our dart throwing technique is to use a discretization of the input surface combined to a geodesic metric based on Dijkstra's algorithm [12]. The discrete grid created over the input mesh limits the positions of the darts to vertices belonging to this grid, and consequently reduces the time-complexity of geodesics - one major drawback of brute-force dart throwing techniques - without damaging the accuracy of the measures. Experimental results will show that our technique outperforms prior methods that utilize continuous settings to perform brute-force dart throwing [7], [15], [18]. Our sampling technique has been briefly presented in a short paper [26], but more technical details and experimentations are presented in this paper to show the interest of our approach.

We also propose in this paper an improvement of the tool of Wei *et al.* [31], initially developed for analyzing the sampling quality of a given pattern. As this tool is based on the exponential maps technique [29] to compute inter-sample distances, measurement bias may occur in bumpy or sharp areas like corners or feature lines. To overcome this problem, and in order to take advantage of our discrete setting, we introduced in this analysis tool the Dijkstra's algorithm to compute more accurately the power spectra, and thus to analyze more precisely the sampling patterns. In this setting, the discrete grid is generated using a dithered subdivision, to enforce the independence of our algorithm to the quality of the input mesh elements.

The paper is organized as follows: Section 2 reviews the blue noise sampling methods developed for surfaces and existing sampling analysis tools. Section 3 explains in a nutshell the theory of dart throwing to generate Poisson disk distributions on surfaces, and then details our technique. Next, we discuss in Section 4 our improvements to the spectral analysis tool of Wei *et al.* [31]. Experimental results, in-depth assessments and comparisons with prior methods in terms of blue noise properties and timings are provided in Section 5. Finally, Section 6 summarizes our work and presents several perspectives.

2 Prior works

2.1 Prior works on blue noise sampling for surfaces

The techniques of blue noise sampling can be classified according to three categories: the *parameterization-based methods* [22], [23], the *direct methods* [4], [7], [9], [15], [18] and the *relaxation-based methods* [6], [33]. Given our objectives, we also review the *feature preserving methods*.

2.1.1 Parameterization-based methods

The surface is parameterized to a planar domain before applying any 2D sampling technique. Thereby, these methods overcome the issue relative to the computation of geodesics, but the generated patterns may suffer from parameterization distortions, in particular for surfaces with complex topologies.

Li *et al.* [22] present a tiling-based sampling method for surfaces of arbitrary topology. The idea is to first build a dual surface from an input parameterized surface, and then to arrange pre-computed Poisson disk tiles on it. This method produces sampling patterns with good blue noise properties as long as the parameterization is quad-based and has low distortions.

Li *et al.* [23] propose an original approach based on parameterization. The idea consists in performing an anisotropic sampling on a planar domain in order to obtain an isotropic sampling of the input surfaces.

Jacobian distances are used to improve the wrapping back phase into 3D, and thus to limit parameterization distortions.

We can also cite the work of Alliez *et al.* [1] that is one of the first parameterization-based remeshing technique for surfaces using a blue noise 2D sampling. The authors use a technique of halftoning initially based for grey-scale images to resample the vertices on the parameterized surface before the triangulation.

2.1.2 Direct methods

"Direct" means that the sampling is done on the surface directly (without parameterization). These methods are inspired by the technique of dart throwing developed for planar domains by Dippé *et al.* [13]. They generate patterns with excellent blue noise properties, but are time-consuming when the geodesics are computed accurately.

For instance, Fu *et al.* [15] extend the 2D dart throwing method of Dunbar *et al.* [14], and choose exact geodesic metrics to measure distances between samples. Once the sampling is done, relaxation is applied on the samples to improve their isotropy and finally produces high quality meshes, to the detriment of the blue noise properties.

Cline *et al.* [7] and then Corsini *et al.* [9] were inspired by the hierarchical dart throwing (HDT) of White *et al.* [32], that uses quad-trees to generate samples in 2D domain. Cline *et al.* [7] propose a dart throwing for surfaces, faster than prior techniques. The main contribution is a new index structure particularly efficient for excluding regions already covered by previous darts. They also show that their technique can be extended to other types of surfaces like NURBS, implicit surfaces, etc.

Corsini *et al.* [9] propose two methods to generate efficient blue noise sampling patterns on meshes. Both are independent of the input connectivity and sampling, and are based on a Monte-Carlo algorithm to pre-generate a pool of samples on the surface meshes. This pool of samples is then shrunk by selecting only the samples verifying the given minimum distance, *via* a cubic space subdivision, or directly with respect to the samples. The first approach is similar to the one proposed by Bowers *et al.* [4], which uses GPU to parallelize the generation of samples with respect to cubic subdivisions. Their two algorithms present good blue noise properties, but the use of Euclidean distances may limit their efficiency with complex shapes, or surfaces varying in narrow spaces, for instance.

2.1.3 Relaxation-based methods

They are all inspired by Lloyd's relaxation [24]. Basically, a given density function is minimized, by updating iteratively the position of the samples. These kinds of methods overcome the problem of controllability of sampling size relative to dart throwing techniques. They are also faster and less complex. The main drawback is that they tend to generate hexagonal lattices, whose regularity does not fulfill blue noise properties.

In 2D, we can cite the work of Chen *et al.* [5] who implement a parallelized local version of the algorithm FPO (farthest point optimization) originally developed by Schlömer *et al.* in [28], and based on constrained farthest point optimization (CFPO). These methods do not present an hexagonal lattice which constitutes one of their main advantages, but their algorithms only handle the plane space by repositioning samples so that they are as far away as possible from each other, and are also restricted to uniform sampling.

Xu *et al.* [33] propose the concept of capacity constrained surface triangulation (CCST), that produces sets of samples exhibiting good blue noise properties. This is an extension of the capacity-constrained De-

launay triangulation (CCDT) of Xu *et al.* [34] developed for planar domains. This work gives one remeshing example to show that the resulting meshes also present well-shaped triangles thanks to the relaxation. Nonetheless, it does not manage sharp features, or surfaces with complex topologies.

In parallel, Chen *et al.* [6] present an extension of the concept of capacity-constrained point distributions (CCPD) developed by Balzer *et al.* [3]. They provide a flexible variational framework for generating blue noise sampling on surfaces and deformable surfaces. Multi-class sampling is also possible.

2.1.4 Feature preserving sampling

Among the sampling techniques described above, some approaches strive to preserve feature lines. For instance, Ge *et al.* [17] propose to combine the Euclidean and Riemannian metrics to compute the minimum distances while taking into account the feature lines.

Recently, Geng *et al.* [18] combine the techniques of Cline *et al.* [7] and Fu *et al.* [15] to perform adaptive sampling while preserving features. This technique outperforms [15] in terms of time-consumption and complexity, but is difficult to handle since the user has to tune several parameters in order to get satisfactory results.

2.2 Tools for analyzing the quality of surface sampling

One popular technique to evaluate the spatial uniformity of a sampling pattern was proposed by Lagae *et al.* in [21]. This technique computes the ratio between the minimum inter-sample distance of the generated patterns, and the average inter-sample distance computed from the maximum packing of a given number of samples. This technique handles only uniform sampling.

Then, several analysis tools extended the Fourier analysis [4], [8], [23], [30], [31]. Bowers *et al.* [4] use a spectral mesh basis that handles only uniform cases and is restrained to a small number of samples due to the expensive numerical complexity. In 2011, Wei *et al.* [31] extend the typical Fourier analysis kernel (*i.e.* cosinus function) to assess various sampling patterns. This method uses a geodesic computation method based on decals [29], that locally computes a parameterization from each sample to its neighbor ones. This parameterization is used to compute the geodesic distances between samples afterwards. We adapted this tool to support general topologies as those presented in Section 5 and to compute the distances between samples more accurately.

3 Discrete Dart Throwing

This section presents the notion of Dart Throwing, and details the different stages of our sampling algorithm.

3.1 Poisson disk distribution

Nowadays, Poisson disk distribution is widespread in computer graphics due to its blue noise properties. It meets the two requirements highlighted in [8]: the uniformity of the sampling patterns, with the irregularity of the sample positions.

Given a set of N samples $S = \{S_i\}$ embedded in an n-dimensional sampling domain Ω, the irregularity and the uniformity can be expressed with the following equation [16]:

$$\forall S_i \in S, \forall \Omega_i \subset \Omega, P(S_i \in \Omega_i) = \int_{\Omega_i} ds, \tag{1}$$

where $P(S_i \in \Omega_i)$ represents the probability of S_i to fall into the sub-domain Ω_i of Ω. It is proportional to the size of Ω_i, such that no region of Ω will be empty of samples. Typical Poisson disk distributions can be done using equation (1), but it is not convenient with surfaces because it may generate some clusters of samples. Therefore it is usual to restrain the sample positions with respect to each other, and thus avoid such clustering artifacts, by imposing a minimum distance $2R$ between any pair of samples:

$$\forall (S_i, S_j) \in S^2, ||S_i - S_j|| \geq 2R. \tag{2}$$

Dart throwing (DT) is performed as explained below. Assume that $2R$ is the minimum distance required between two samples. The DT consists in i) picking out randomly a sample S_i on the domain (irregularity), ii) drawing a disk of radius R around it, and iii) verifying if this disk intersects another disk. If no disk is intersected, the sample S_i is kept since it respects the required distance $2R$ (uniformity). Otherwise, the sample S_i is discarded. This process is iterated until no more sample can be thrown on the domain without violating the minimum distance, or until a user-given number of samples is reached.

To extend DT to surfaces, it seems natural to calculate geodesics instead of Euclidean distances. Indeed, Euclidean metric may introduce disparity when computing the radii across sharp features or in some bumpy regions for instance, as shown in Figure 3. The initial 2D disk around each sample is thus replaced by a *circular patch* on the surface, depending on the geodesic distance R. The computation of geodesics is one issue

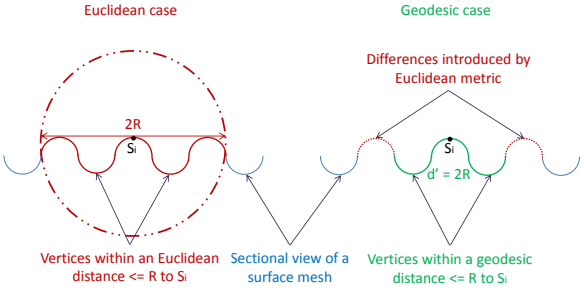


Fig. 3 Disparity of distance measurement between Euclidean and Geodesic metrics on surfaces.

of the DT algorithms, because a trade-off between accuracy and time cost is needed.

DT can be generalized to adaptive (*i.e.* non-uniform) sampling by incorporating a density function. In the context of remeshing, curvature-aware sampling is particularly relevant, since it allows to better approximate the original surface if the density function depends on curvature values. In other words, the higher the curvature values, the smaller the radii of the associated circular patches are. Curvature-aware DT is illustrated in Figure 4 for a planar domain.

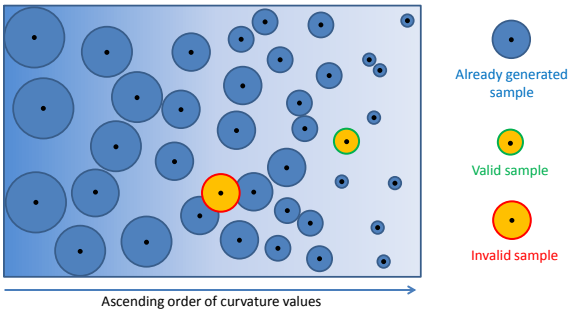


Fig. 4 Curvature-aware dart throwing. Blue disks are associated to samples already positioned. The red and green disks indicate if the given samples respect the required distance.

3.2 Discrete dart throwing: proposed approach

This section explains our proposed approach to sample surface meshes and guarantee the preservation of features, using an efficient discrete dart throwing algorithm. Let us denote M as an input 2-manifold triangular mesh, of any genus, closed or not.

3.2.1 Vertex classification

It consists in separating the vertices of M with respect to three classes: *corners*, *sharp features* and *smooth regions*. The sampling will then be driven by this classification, to preserve original features of M.

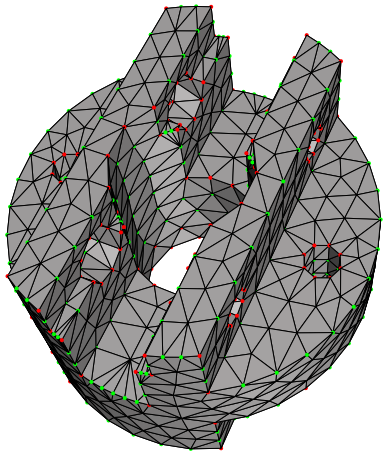


Fig. 5 Green and red points are respectively the vertices considered as *sharp features* and *corners*.

Our classification is based on normal tensor voting theory [20]. It computes for each vertex v of M a 3×3 weighted covariance matrix. Its three sorted eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$ (λ_2 and λ_3 are respectively the maximum and the minimum curvature values) are used to classify v:

- If λ_1 is dominant, and λ_2, λ_3 are close to 0, v is classified into *smooth regions*;
- If λ_1 and λ_2 are dominant, and λ_3 is close to 0, v is classified into *sharp features*;
- If the three eigenvalues are approximately equal, v is classified into *corners*.

Figure 5 shows the efficiency of this classification on SOCKET: green and red points are respectively the vertices considered as *sharp features* and *corners*.

3.2.2 Curvature-aware sampling

This section describes our curvature-aware DT, inspired by [7]. To guarantee that the features are processed, the sampling is done in order of priority. More precisely, the first samples are thrown on the *corners*, then among the vertices of the class *sharp features*, and finally among the vertices of the class *smooth regions*.

Radius formulation

One specificity of our algorithm is to propose a solution for computing the radius of the circular patch relative to a candidate sample. We started from the uniform case, where the radius R_{min} is constant. The total area $|M|$ of the surface is approximatively equal to the area of all the circular patches: $|M| \approx (N_t.\pi.R_{min}^2)$, where

N_t represents the number of samples. According to this formulation, we consider that R_{min} is given by

$$R_{min} = \alpha.\sqrt{\frac{|M|}{\pi.N_t}} \; with \; \alpha \; \leq \; 1, \tag{3}$$

where α compensates the regions of the initial mesh that are not covered by the patches (empirically set to 0.65).

We also want our sampling to be curvature-aware, in order to better approximate the shape geometry. A popular way is to use the curvature values. During this work we propose a decreasing exponential function depending on the extremal curvature values, in order to focus on curved areas during sampling:

$$R = R_{min}.(1 + e^{C.\lambda_2} + e^{C.\lambda_3}), \tag{4}$$

where C is a negative parameter. We observed that $C = -8.0$ for *sharp features*, and $C = -6.0$ for *smooth regions* generate sampling patterns that exhibit good blue noise properties, while ensuring that the number of samples is close to the user-given parameter N_t. Also, the radius relative to samples that belong to the class *corners* is always set to R_{min}. It makes our algorithm more convenient than some prior works like [18], since only one parameter, the number of samples N_t, is required as input.

Low-complexity Dart Throwing

One of our objectives is to reduce the well-known computational complexity of typical brute-force DT. We prove in this paper that having a discrete approach instead of a continuous one (as the prior techniques) can be effective, even when geodesics are used. There are two major issues: i) the computation of geodesic distances to define the circular patches, and ii) the update of the available sampling domain for the next throwing.

The computation of geodesic distances is expensive when accurate measures are needed. It is the case with shapes of complex topologies. To reduce the runtime, we estimate the geodesic distances using Dijkstra's algorithm [12], one of the fastest method. Basically, it computes the shortest path between two points by following the mesh edges. Consequently, the accuracy is relative to the mesh density: the denser the mesh M, the more accurate the measures of the geodesic distances will be. Therefore, before the classification several midpoint subdivisions are performed on M to generate a denser mesh, called M_{sub}. The midpoint subdivision splits each triangle into four ones, by simply adding a new vertex in the middle of each edge, and connecting them.

Experimentations showed that this step increases the accuracy of our algorithm, but also the runtime if too many subdivisions are performed. In order to address this issue, our algorithm assesses the density of the input mesh M, and then computes how many subdivisions are needed. Our technique relies on the ratio between the average area of the triangles and the area of M: if this ratio is greater than a threshold (we set it to 2.5×10^{-5}), our algorithm considers that M is not dense enough, and computes the number of iterations needed to reach this "resolution". This technique automatically adapts the density of the mesh, and makes the process faster if M is already dense enough.

The update of the available sampling domain consists in removing, from the mesh M_{sub}, the region corresponding to the circular patch of a newly accepted sample. It will prevent us from throwing a dart on an unallowable region, and it makes the DT faster. To achieve this goal, Geng *et al.* [18] propose to extract an iso-line, which corresponds to the boundary of the newly added circular patch on the surface. In [7], Cline *et al.* propose to recursively split the triangles crossed by the boundary, and then to update the mesh connectivity. These techniques are efficient but time-consuming.

Our approach is quite different and takes advantage of the grid offered by M_{sub}. The idea is to limit the candidate samples to the set of vertices of M_{sub}. Section 5 will show that if M_{sub} is dense, the resulting set of samples will exhibit blue noise properties as good as "triangle-based interpolating" sampling techniques, while being less complex.

Limiting the candidate samples to the set of vertices of M_{sub} makes the generation of available boundaries much easier. The structure of our algorithm is based on a list, containing all the vertices initially, and in which each candidate sample is randomly chosen. When a candidate is valid, the list is shrunk by eliminating all the vertices met during the research of its associated circular patch over the surface of M_{sub}. Finally, validating a candidate sample only consists in checking that each vertex of M_{sub} within the associated circular patch is still part of the list, which significantly simplifies the DT process, while generating patterns with good blue noise properties, as shown in Section 5.

4 Dithered spectral analysis tool

To analyze the quality of the generated sampling patterns, we selected the tool of Wei *et al.* [31], as in [33]. This tool performs a differential analysis of sampling patterns on surfaces, and is well suited to analyze non-uniform sampling. For the computation of

the power spectrum, this method extends the classical Fourier analysis expressed by

$$P(\mathbf{f}) = \frac{1}{N} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \cos(2\pi \mathbf{f}.(s_i - s_j)), \tag{5}$$

where $\mathbf{f}$ is the frequency vector, to a more general analysis given by

$$P(\mathbf{q}) = N \times \int_{\Omega_d} K(\mathbf{q}, \chi(s, s', \mathbf{d}) p(\mathbf{d})) \delta \mathbf{d}. \tag{6}$$

In this equation, $\mathbf{q}$ is a set of parameters relative to the generic kernel K that weights the power spectrum computation in order to assess different properties of the sampling distribution. As an example, for the Fourier analysis, K is the cosine function, and $\mathbf{q}$ represents the frequencies. $\mathbf{d}(.)$ is the differential function of inter-sample distances. Ω_d is the sampling domain for the function $\mathbf{d}$. $p(\mathbf{d})$ is the probability density function of $\mathbf{d}$ in Ω_d and $\chi(s, s', \mathbf{d})$ is the function that locally warps a distance in a non-uniform domain to a uniform one.

This approach enables to focus on different properties of the distribution, like the spatial density instead of the traditional frequency, for instance.

During our experimentations, and after several tests and an in-depth study of this tool, we observed that the discrete exponential maps [29] used to compute the local parameterization around each sample and consequently the geodesic distances between samples may be not convenient for complex shapes, in particular with sharp features and/or of high genus.

To make the computation of the geodesic distances more accurate, we included in this tool the Dijkstra's algorithm already presented in Section 3.2.2. To highlight the interest of using this algorithm during the analysis, Figure 6 shows the RAPS of a blue noise sampling pattern generated on Axle (a complex model shown in Table 2), obtained either with the original tool of Wei *et al.* [31] or with our improved method. We observe that with our tool, the low frequencies really present a zero-band region, meaning that the minimum distance is preserved, and that our approach respects the sharp features of the input mesh. Furthermore it allows to better "capture" the sharp transition between the low and high frequencies.

As our spectral analysis tool now uses Dijkstra's algorithm, one might wonder if a regular input mesh influences the quality of our analyses. Indeed, if the input mesh that we want to resample is regular, the subdivided mesh used to make Dijkstra's algorithm more accurate will be also regular. Consequently, when computing the geodesic distances, the well-known *quantization bias* of Dijkstra's algorithm may appear.

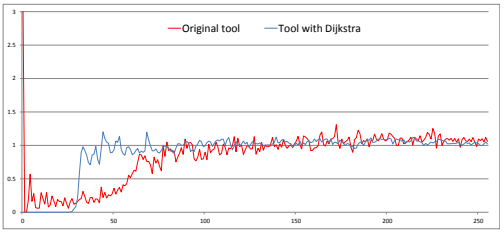
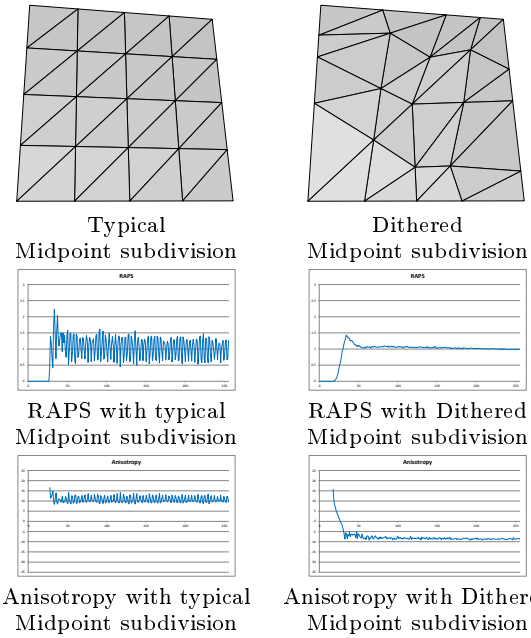


Fig. 6 Comparison of the RAPS of Axle (approximatively 1k samples) computed with the tool of Wei *et al.* [31] and with our modified version.

Table 1 Subdivided version of a planar mesh, RAPS and anisotropy with typical Midpoint subdivision (left) and with Dithered Midpoint subdivision (right).



To overcome this potential problem, a random shift is applied during the midpoint subdivision. We indeed observed that this technique improves the computation of the geodesic distances during the analysis, by limiting the measurement bias in some directions. Notice that the shift is done along the edges, in order to take into account feature lines, again. This technique is similar to the famous dithering [8], that aims at introducing a noise willingly, in order to randomize the bias (in our context, the length and the orientation of the edges in the subdivided input mesh).

To show the interest of this technique, we first generated a blue noise sampling pattern from a planar regular mesh. Second, we analyzed the sampling quality with our tool, when the random shift is enabled, or not.

Results are given by Table 1. As expected, we observe that combining randomness and subdivision deletes the *quantization bias*: the RAPS presents typical blue noise properties (at right) which is not the case if a classical midpoint subdivision is used (at left). Additional results are presented in Appendix A to prove that the proposed analysis tool also works with other sampling patterns, such as white noise.

5 Experimental results and discussions

This section presents experimental results to assess the efficiency of our algorithm. We first give several visual results to prove that our algorithm well preserves the feature lines. We then compare the quality of the patterns generated with our algorithm and with several state-of-the-art techniques. Next, we evaluate the robustness of our algorithm, and compare our runtime with [15] and [18], that combine geodesic distances and continuous settings to perform brute-force DT on surface meshes.

5.1 Preservation of the features

Table 2 shows four input shapes, and the output sampling patterns generated with our algorithm, when the features are preserved or not. Each model has different genus (from 0 to 10), and contains around one thousand samples. We observe that, if the features are not taken into account during the sampling, few samples lie on them. *A contrario*, a feature-preserving sampling increases the number of samples in such regions, and finally improves significantly the fidelity to the initial surface.

5.2 Analysis of the sampling quality

We first study the influence of the preservation of the features on the sampling quality. As in [33], each spectral analysis is based on eight sets of generated samples (*i.e.* eight sampling outputs). Table 3 shows the power spectrum, the RAPS and the anisotropy associated to each model presented in Table 2, with or without the vertex classification step. We observe that the preservation of the features does not influence significantly the sampling properties (see Figure 1 for ideal properties), even for the shapes with a lot of sharp features, Axle for instance. Nevertheless, we notice a difference of anisotropy for Socket and Axle: feature lines are considered as preferential directions, which inevitably increases the sampling anisotropy, when they are taken into account by our method.

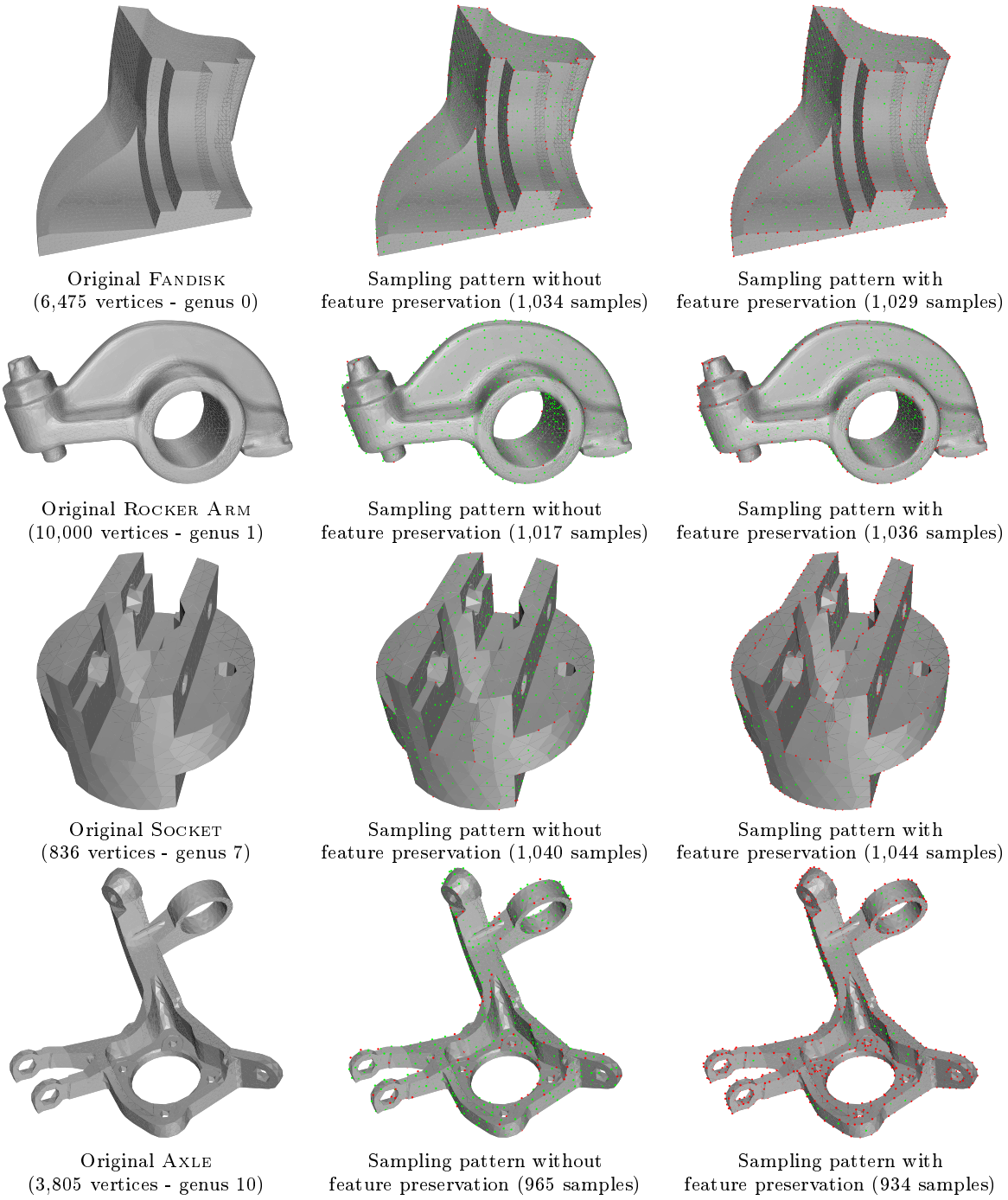
We now compare our results with those of two prior works, [9] and [31]: see Table 4. We observe that our method generates distributions with higher blue noise properties. Indeed, the RAPS relative to the prior methods oscillate significantly around the cut-off frequency, meaning that too many samples are thrown around the minimum distance. We also remark that the minimum distance is always respected with our method or [31]. This is not the case with [9], even when the geometric features are taken into account. We also notice that our method produces distributions with an higher anisotropy than [31]. It was expected since this latter does not preserve the sharp features. On the other hand, our anisotropy is lower than the one of [9] for each mesh.

5.3 Robustness of our algorithm

We now study the influence of the input mesh density and the influence of the output sampling density on the blue noise properties of sampling patterns generated with our method on three models: Sharp Sphere, Mask and Fertility. Table 5 displays the resulting sampling patterns, and Table 6 gives the corresponding power spectra, RAPS and anisotropy curves. In terms of RAPS, we observe that the generated sample sets globally exhibit satisfactory blue noise properties, whatever the size of the input meshes (ranging from 10,443 to 241,607 vertices), and the number of generated samples. This is also the case for the anisotropy, and we observe, as expected, that the sampling density influences the anisotropy, which decreases as the number of samples rises.

Finally, we assess the sampling quality of the generated patterns in function of the quality of the input mesh elements. First we consider highly regular and uniform input meshes. Indeed, it is wise to verify if limiting the sampling domain to a midpoint subdivided mesh (*i.e.* a uniform and regular grid), influences the quality of the sampling patterns. We thus applied our algorithm to the model Eight that is defined by a regular and uniform grid (see Table 7), and generated an output sampling pattern with approximatively the same number of samples than the number of original vertices (around 3k vertices). We observe that the resulting sampling pattern is uniform but highly irregular, leading to very good blue noise properties, as depicted in Table 7, on the right. The random dart throwing and the circular patches used around samples are sufficient to produce an irregular sampling pattern.

Table 2 Four surfaces and our resulting samplings. From left to right: original shapes, sampling patterns respectively without and with preservation of features. To provide visual assessment of the feature-preserving solution, samples that lie on sharp features are depicted in red, while others are depicted in green.



Original Fandisk (6,475 vertices - genus 0) | Sampling pattern without feature preservation (1,034 samples) | Sampling pattern with feature preservation (1,029 samples)

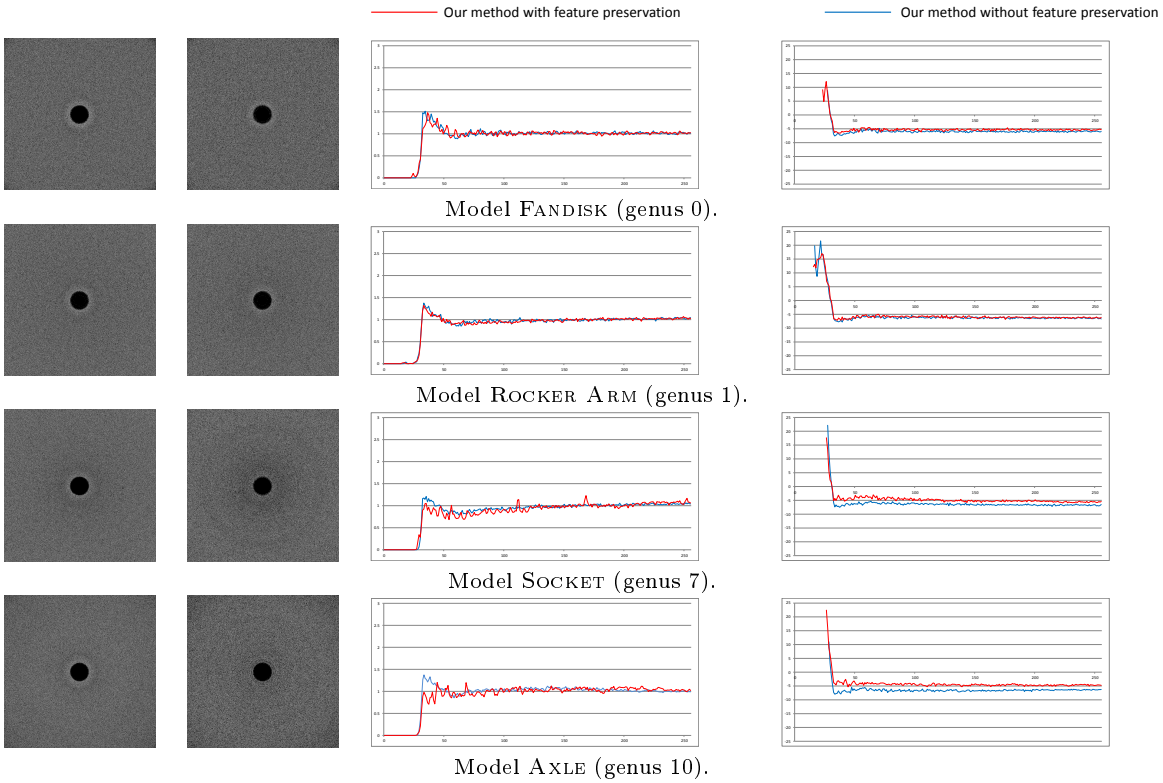
Original Rocker Arm (10,000 vertices - genus 1) | Sampling pattern without feature preservation (1,017 samples) | Sampling pattern with feature preservation (1,036 samples)

Original Socket (836 vertices - genus 7) | Sampling pattern without feature preservation (1,040 samples) | Sampling pattern with feature preservation (1,044 samples)

Original Axle (3,805 vertices - genus 10) | Sampling pattern without feature preservation (965 samples) | Sampling pattern with feature preservation (934 samples)

Then, we apply our algorithm on a highly irregular input grid, for instance the Bunny model: see Table 8. We observe that our algorithm generates a sampling pattern with poorer blue noise properties: the transition at the cut-off frequency is slower for the RAPS. The problem comes from our discrete approach. More precisely, the density of the input vertices and the size of the triangles vary significantly. Even with subdivisions the problem remains the same. After DT, inevitably the regions with large triangles are less sampled than the ones with narrow triangles, even if the circular patches around the samples limit the density of output samples in the regions with small triangles. Applying numerous subdivisions on such meshes is a naive solution to

Table 3 Influence of the feature preservation on the sampling quality for several models. From left to right: power spectra without (blue curves) and with (red curves) feature preservation, followed by their respective RAPS and anisotropy curves.



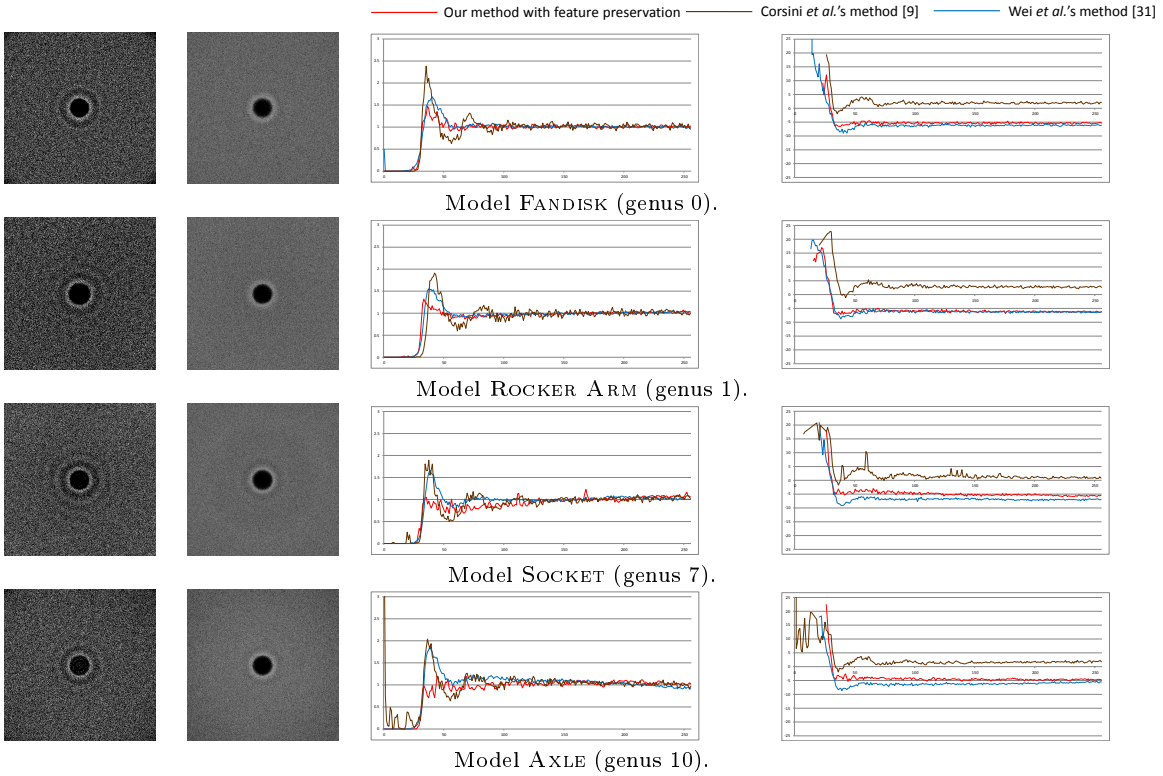
Model FANDISK (genus 0).

Model ROCKER ARM (genus 1).

Model SOCKET (genus 7).

Model AXLE (genus 10).

Table 4 Comparison with two state-of-the-art algorithms. From left to right: power spectra relative to [9] (brown curves), power spectra relative to [31] (blue curves), followed by their respective RAPS and anisotropy curves. The RAPS and the anisotropy curves of our method are also depicted in red.



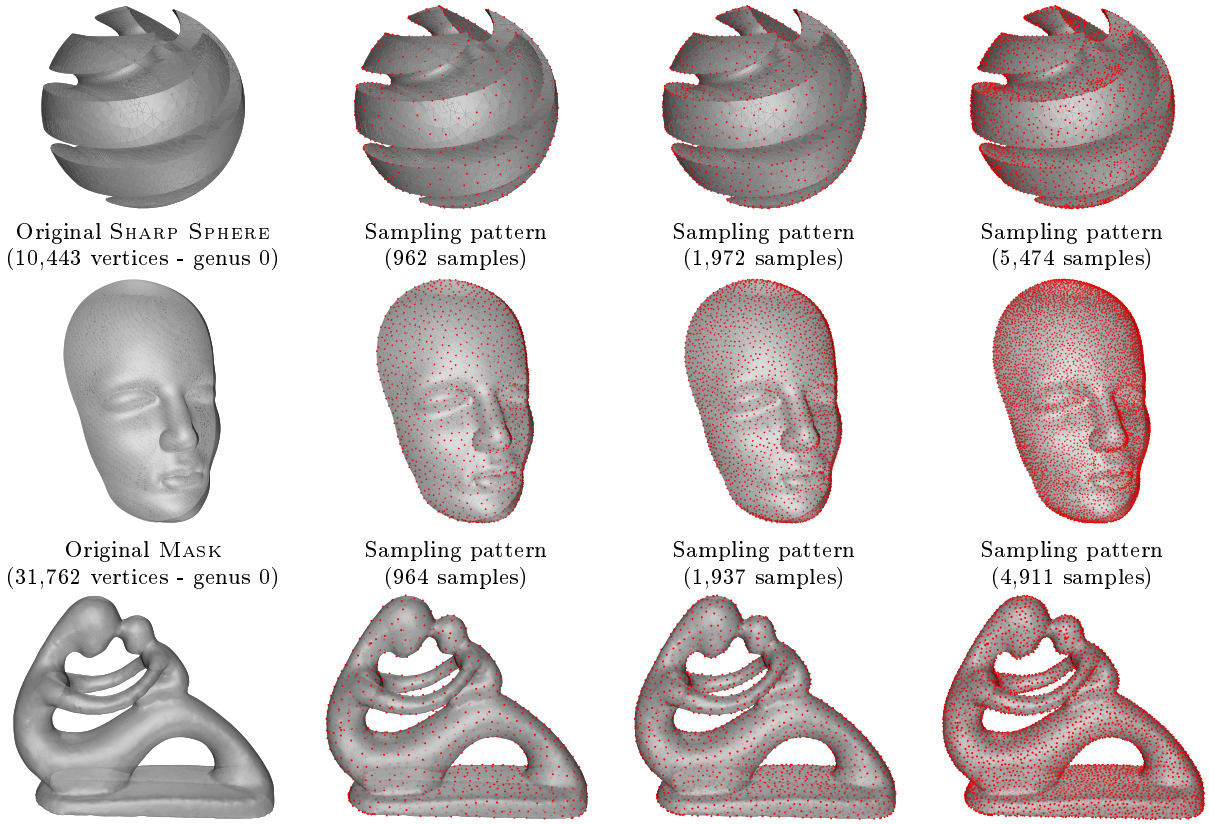
Model FANDISK (genus 0).

Model ROCKER ARM (genus 1).

Model SOCKET (genus 7).

Model AXLE (genus 10).

Table 5 Original shapes and sampling patterns generated with our algorithm at different sampling densities. From left to right: input shapes, and Poisson-disk sample sets with respectively 1k, 2k and 5k samples.

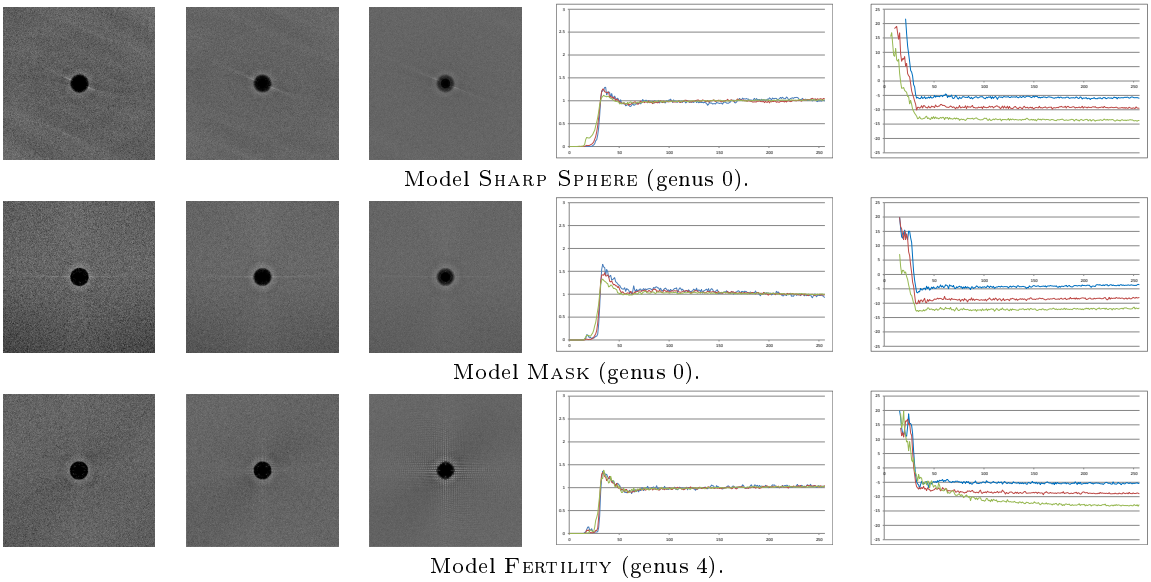


Original Sharp Sphere (10,443 vertices - genus 0) | Sampling pattern (962 samples) | Sampling pattern (1,972 samples) | Sampling pattern (5,474 samples)

Original Mask (31,762 vertices - genus 0) | Sampling pattern (964 samples) | Sampling pattern (1,937 samples) | Sampling pattern (4,911 samples)

Original Fertility (241,607 vertices - genus 4) | Sampling pattern (1,008 samples) | Sampling pattern (1,960 samples) | Sampling pattern (5,045 samples)

Table 6 Comparison of the blue noise properties for different input meshes and output sampling densities. From left to right: power spectra for the 1k-sampling (blue curves), the 2k-sampling (red curves) and the 5k-sampling (green curves), followed by their respective RAPS and anisotropy curves.



Model Sharp Sphere (genus 0).

Model Mask (genus 0).

Model Fertility (genus 4).

Table 7 EIGHT before (on the left) and after sampling (in the middle) with approximatively the same number of vertices. The power spectrum (on the right) proves that our algorithm is not sensitive to a regular and uniform input grid.

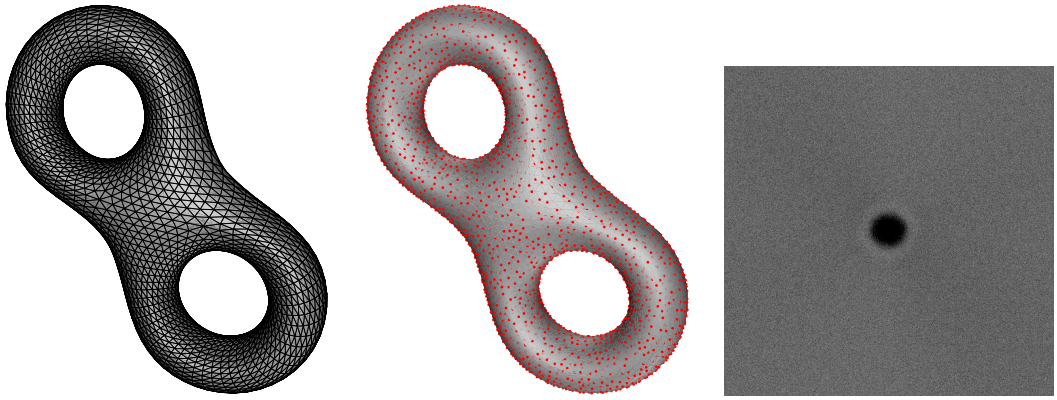
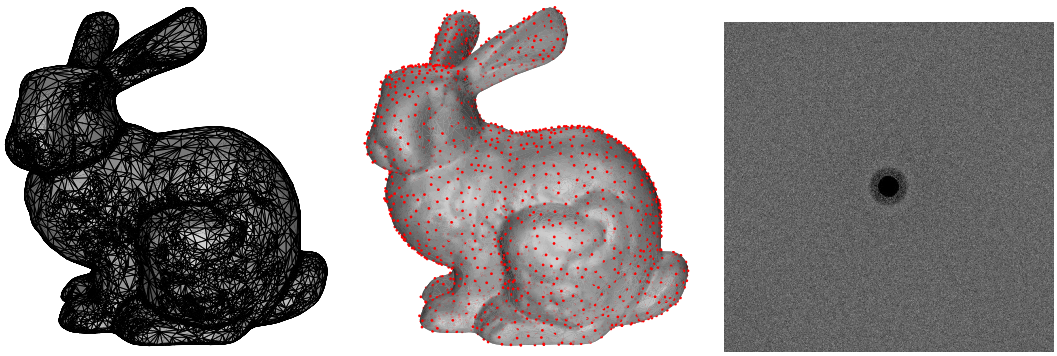


Table 8 BUNNY before (on the left) and after sampling (in the middle) with approximatively 2k samples. A slower transition around the cut-off frequency appears on the power spectrum (on the right). It proves that our algorithm may be sensitive to a highly irregular and non-uniform input grid, because of our discrete approach.



overcome this issue because the runtime should increase drastically. One possible solution may be to further subdivide the large triangles only, in order to attenuate the variability of the triangle areas. This is one of our future works.

5.4 Runtime

This section compares the runtime of our sampling method with similar dart throwing methods (brute-force, and using geodesics). We first compare our runtime with the recent sampling method of Geng *et al.* [18]. Our algorithm ran on a Intel Core i3 CPU 2.30 GHz, 4 GB RAM processor, while the algorithm of [18] ran on an Intel Core Duo CPU 2.67 GHz, 2 GB RAM processor. This algorithm is based on the method of Cline *et al.* [7]: it extends the latter by using a mesh clipping method to ensure the efficiency of the index structure, and collision detection frame. Moreover, the algorithm of [18] is faster than the algorithm of Fu *et al.* [15] that uses exact geodesic to search available boundaries over the surface meshes.

Table 9 proves that our algorithm is globally faster than [18], up to 3 times faster for CASTING. It confirms that our algorithm overcomes the main drawbacks of typical brute-force DT methods, time-consumption and complexity, since our method is also relatively straightforward to implement.

In Table 10, we give runtime for meshes of different sizes, and varying output sampling densities. For SHARP SPHERE and MASK, the asymptotic runtime is rapidly reached. Also the runtime does not increase severely, as the number of output samples rises. This is due to the fact that the higher the sampling density, the smaller the geodesics are. So, even if the number of samples increases, the runtime is not affected significantly since Dijkstra's algorithm requires less time per sample.

For bigger models such as FERTILITY (around 480k vertices for the original and the intermediary subdivided mesh), the algorithm becomes slower, in particular when 10,000 samples are demanded (around 2 minutes). This result highlights one drawback of using Dijkstra's algorithm for computing the geodesics. Never-

Table 9 Runtime comparison between our sampling method and geodesic-based brute-force dart throwing methods [15] and [18].

Models	Sampling method	Nb. facets Input	Nb. facets Subdivided	Nb. vertices Output	Sampling time (ms)
Casting	[15]	10,204	—	1,156	31,843
	[18]	10,204	—	1,834	5,031
	Ours	10,224	40,896	1,931	**1,529**
Fandisk	[15]	12,946	—	382	36,125
	[18]	12,946	—	1,469	7,172
	Ours	12,946	51,784	1,622	**2,605**
Block	[15]	4,208	—	551	8,469
	[18]	4,208	—	2,026	**2,500**
	Ours	4,272	68,352	2,292	3,869
Sharp Sphere	[15]	18,864	—	358	229,547
	[18]	18,864	—	3,559	11,656
	Ours	20,882	83,528	3,534	**7,223**
Hand	[15]	17,290	—	240	89,750
	[18]	17,290	—	2,162	7,594
	Ours	23,186	92,744	2,225	**5,794**

Table 10 Runtimes in function of the input mesh size and output sampling density.

Models	Nb. facets Input	Nb. facets Subdivided	Nb. vertices Output	Sampling time (ms)
Sharp Sphere	20,882	83,528	1,051	6,474
			2,097	7,425
			5,042	7,501
			10,854	7,519
Mask	62,467	62,467	928	2,527
			2,083	3,151
			4,811	3,322
			9,582	3,307
Fertility	483,224	483,224	988	24,325
			1,964	41,487
			5,047	88,060
			9,757	126,859

theless, we believe that the runtime remains acceptable even for this example, since *a priori* sampling does not need to be done in real time.

6 Conclusion and perspectives

We presented a resampling method for discrete surfaces. Our main objectives were: i) to generate sampling patterns exhibiting high blue noise properties, while approximating well the input shapes (preservation of feature lines); ii) to develop a method for meshes of arbitrary topology; iii) to reduce the complexity of the previous brute-force dart throwing techniques while preserving features and using geodesics.

Therefore, we proposed a feature sensitive DT technique for surfaces. One originality of our technique is to limit the sampling domain to a discrete grid defined by a subdivided version of the original mesh. We also use this subdivided mesh as graph for the computation of the geodesics (with Dijkstra's algorithm) during sampling. Finally, our approach is robust, easy to implement, avoids geometric aliasing, and generates patterns with good blue noise properties, whatever the topology of the input meshes.

In parallel, we also presented an improvement of the tool of Wei *et al.* [31], developed for analyzing sampling quality. Our improvement allows the users to analyze more finely the characteristics of sampling patterns generated on surfaces of arbitrary topology.

Experimental results also highlight some limitations or open questions. For example, we observe on some RAPS a small peak before the cut-off frequency. This artifact might come from the radius formulation given by equation (4), but this assumption still needs to be confirmed. Another promising way is the use of our dart throwing for high quality remeshing.

A last promising way is the development of an accurate spectral analysis tool for surface sampling. Even though the tool presented in [31] is satisfactory, we showed some limitations on complex topologies. Our modified version overcomes this problem but, in the same time, may introduce a bias due to Dijkstra's al-

gorithm. Using a recent approach such as [10] for computing accurate geodesic distances could significantly improve this tool (but also our sampling technique).

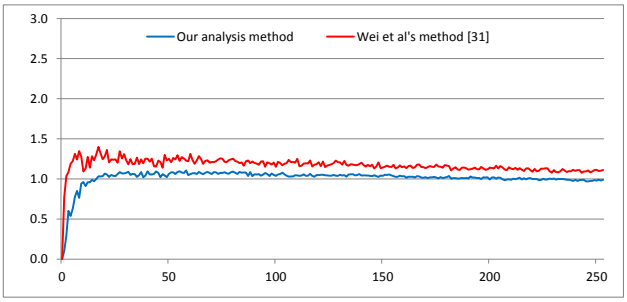
Acknowledgments

We are particularly grateful to Mr. Paolo Cignoni and Miss Ruizhen Hu for answering our questions and sending us several data for our comparisons. We also thank Mr. Li-Yi Wei and Mr. Rui Wang for providing us with their executable. We also would like to thank Leonardo Hidd Fonteles for his help on Dijkstra's implementation.

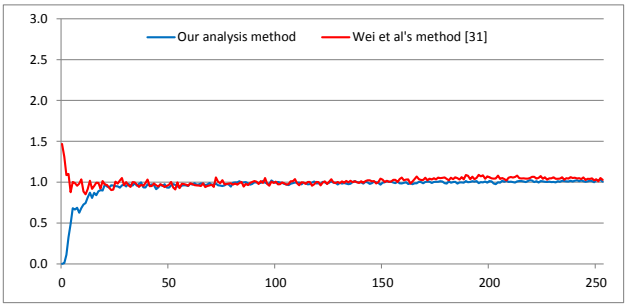
Appendix A: Analysis of white noise sampling

To show that our spectral analysis tool (presented in Section 4) is also efficient for other patterns, we tested it on white noise samples generated on surfaces, and compared with the results produced by the original tool of Wei *et al.* [31]. To be as fair as possible, we generated the patterns on Eight and Hand, two models shown in [31], and each sampling has been generated with the own code of Wei *et al.* [31]. Figure 7 compares the RAPS estimated with our tool and with the original tool of Wei *et al.* [31]. We obtained satisfactory results, since the RAPS produced by our tool are globally flat and equal to 1 at each frequency (typical features of white noise sampling). Note that there is a slope at very low frequencies that can be explained by our discrete approach that inevitably introduces a *quantization bias* during the sampling (computed on the subdivided meshes), and by the well-known bias of Dijkstra's algorithm (the geodesics always lie on the graph edges).

This experimentation confirms the interest of our tool, since the RAPS provided by our spectral analysis algorithm match well the typical white noise characteristics, which is not always the case of [31]: see the RAPS generated by the original tool which are sometimes significantly different from the value 1 even at higher frequencies, on Eight for instance. Moreover their tool also draws a slope at very low frequencies for the two models. So we consider that our adapted tool is valid for all our experimentations.



(a) Eight.



(b) Hand.

Fig. 7 Comparison of the RAPS for two models sampled with white noise patterns, estimated with our tool and with the tool of Wei *et al.* [31].

References

1. Alliez, P., Meyer, M., Desbrun, M.: Interactive geometry remeshing. In: Proceedings of the annual conference on Computer graphics and Interactive Techniques, p. 347–354. ACM, New York, NY, USA (2002)
2. Aspert, N., Santa-Cruz, D., Ebrahimi, T.: Mesh: Measuring errors between surfaces using the hausdorff distance. In: Proceedings of the IEEE International Conference on Multimedia and Exposition, vol. 1, p. 705–708 (2002)
3. Balzer, M., Schlömer, T., Deussen, O.: Capacity-constrained point distributions: A variant of lloyd's method. ACM Transactions on Graphics **28**(3), 86:1–8 (2009)
4. Bowers, J., Wang, R., Wei, L.Y., Maletz, D.: Parallel poisson disk sampling with spectrum analysis on surfaces. ACM Transactions on Graphics **29**(6), 166:1–166:10 (2010)
5. Chen, R., Gotsman, C.: Parallel blue-noise sampling by constrained farthest point optimization. Comput. Graph. Forum **31**(5), 1775–1785 (2012)
6. Chen, Z., Yuan, Z., Choi, Y.K., Liu, L., Wang, W.: Variational blue noise sampling. IEEE Transactions on Visualization and Computer Graphics **18**(10), 1784–1796 (2012)
7. Cline, D., Jeschke, S., Razdan, A., White, K., Wonka, P.: Dart throwing on surfaces. Computer Graphics Forum **28**(4), 1217–1226 (2009)
8. Cook, R.L.: Stochastic sampling in computer graphics. ACM Transactions on Graphics **5**(1), 51–72 (1986). DOI 10.1145/7529.8927
9. Corsini, M., Cignoni, P., Scopigno, R.: Efficient and flexible sampling with blue noise properties of triangular meshes. IEEE Transactions on Visualization and Computer Graphics **18**(6), 914–924 (2012)
10. Crane, K., Weischedel, C., Wardetzky, M.: Geodesics in heat: A new approach to computing distance based on heat flow. ACM Transactions on Graphics (2013)
11. Crow, F.C.: The aliasing problem in computer-generated shaded images. Commun. ACM **20**(11), 799–805 (1977)
12. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik **1**, 269–271 (1959)
13. Dippé, M.A.Z., Wold, E.H.: Antialiasing through stochastic sampling. In: Proceedings of the annual conference on Computer graphics and Interactive Techniques, p. 69–78. ACM, New York, NY, USA (1985)
14. Dunbar, D., Humphreys, G.: A spatial data structure for fast poisson-disk sample generation. In: ACM SIGGRAPH papers, p. 503–508. ACM (2006)
15. Fu, Y., Zhou, B.: Direct sampling on surfaces for high quality remeshing. In: Proceedings of the ACM Symposium on Solid and Physical Modeling, p. 115–124. ACM, New York, NY, USA (2008)

16. Gamito, M.N., Maddock, S.C.: Accurate multidimensional poisson-disk sampling. ACM Trans. Graph. **29**(1), 8:1–8:19 (2009). DOI 10.1145/1640443.1640451. URL http://doi.acm.org/10.1145/1640443.1640451
17. Ge, X., Wei, L.Y., Wang, Y.: Bilateral blue noise sampling. Tech. rep., The Ohio State University, Ohio, USA (2013)
18. Geng, B., Zhang, H., Wang, H., Wang, G.: Approximate poisson disk sampling on mesh. Journal of Computer-Aided Design and Computer Graphics **23**(1), 62–69 (2011)
19. Heck, D., Schlömer, T., Deussen, O.: Blue noise sampling with controlled aliasing. ACM Transactions on Graphics **32**(3), 25:1–25:12 (2013)
20. Kim, H.S., Choi, H.K., Lee, K.H.: Feature detection of triangular meshes based on tensor voting theory. Journal of Computer-Aided Design and Computer Graphics **41**(1), 47–58 (2009)
21. Lagae, A., Dutré, P.: A comparison of methods for generating Poisson disk distributions. Computer Graphics Forum **27**(1), 114–129 (2008). DOI 10.1111/j.1467-8659.2007.01100.x. URL http://www.blackwell-synergy.com/doi/abs/10.1111/j.1467-8659.2007.01100.x
22. Li, H., Lo, K.Y., Leung, M.K., Fu, C.W.: Dual poisson-disk tiling: An efficient method for distributing features on arbitrary surfaces. IEEE Transactions on Visualization and Computer Graphics **14**(5), 982–998 (2008)
23. Li, H., Wei, L.Y., Sander, P.V., Fu, C.W.: Anisotropic blue noise sampling. In: ACM SIGGRAPH ASIA papers, p. 167:1–167:12. ACM (2010)
24. Lloyd, S.P.: Least squares quantization in pcm. IEEE Transactions on Information Theory **28**, 129–137 (1982)
25. Öztireli, A.C., Alexa, M., Gross, M.: Spectral sampling of manifolds. In: ACM SIGGRAPH Asia 2010 papers, SIGGRAPH ASIA '10, pp. 1–8. ACM, New York, NY, USA (2010)
26. Peyrot, J.L., Payan, F., Antonini, M.: Feature-preserving Direct Blue Noise Sampling for Surface Meshes. In: Eurographics (Short Papers), pp. 9–12. DOI 10.2312/conf/EG2013/short/009-912. URL http://diglib.eg.org/EG/DL/conf/EG2013/short/009-912.pdf
27. Pharr, M., Humphreys, G.: Physically Based Rendering: From Theory to Implementation. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2004)
28. Schlömer, T., Heck, D., Deussen, O.: Farthest-point optimized point sets with maximized minimum distance. In: Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics, HPG '11, pp. 135–142. ACM, New York, NY, USA (2011). DOI 10.1145/2018323.2018345. URL http://doi.acm.org/10.1145/2018323.2018345
29. Schmidt, R., Grimm, C., Wyvill, B.: Interactive decal compositing with discrete exponential maps. In: ACM SIGGRAPH papers, p. 605–613. ACM, New York, NY, USA (2006)
30. Wei, L.Y.: Multi-class blue noise sampling. ACM Trans. Graph. **29**(4), 79:1–79:8 (2010). DOI 10.1145/1778765.1778816. URL http://doi.acm.org/10.1145/1778765.1778816
31. Wei, L.Y., Wang, R.: Differential domain analysis for non-uniform sampling. ACM Transactions on Graphics **30**(4), 50:1–50:10 (2011). DOI 10.1145/2010324.1964945
32. White, K.B., Cline, D., Egbert, P.K.: Poisson disk point sets by hierarchical dart throwing. Symposium on Interactive Ray Tracing **0**, 129–132 (2007)
33. Xu, Y., Hu, R., Gotsman, C., Liu, L.: Blue noise sampling of surfaces. Computer and Graphics **36**, 232–240 (2012)
34. Xu, Y., Liu, L., Gotsman, C., Gortler, S.J.: Capacity-constrained delaunay triangulation for point distributions. Computers & Graphics **35**(3), 510–516 (2011)