DOI: 10.1111/cgf.12071
EUROGRAPHICS 2013 / I. Navazo, P. Poulin
(Guest Editors)

Volume 32 (2013), Number 2

Sifted Disks

Mohamed S. Ebeida[1], Ahmed H. Mahmoud[2], Muhammad A. Awad[2], Mohammed A. Mohammed[2],
Scott A. Mitchell[1], Alexander Rand[3], and John D. Owens[4]

[1]Sandia National Laboratories, New Mexico, USA
[2]Alexandria University, Alexandria, Egypt
[3]CD-adapco, Texas, USA
[4]University of California, Davis, USA

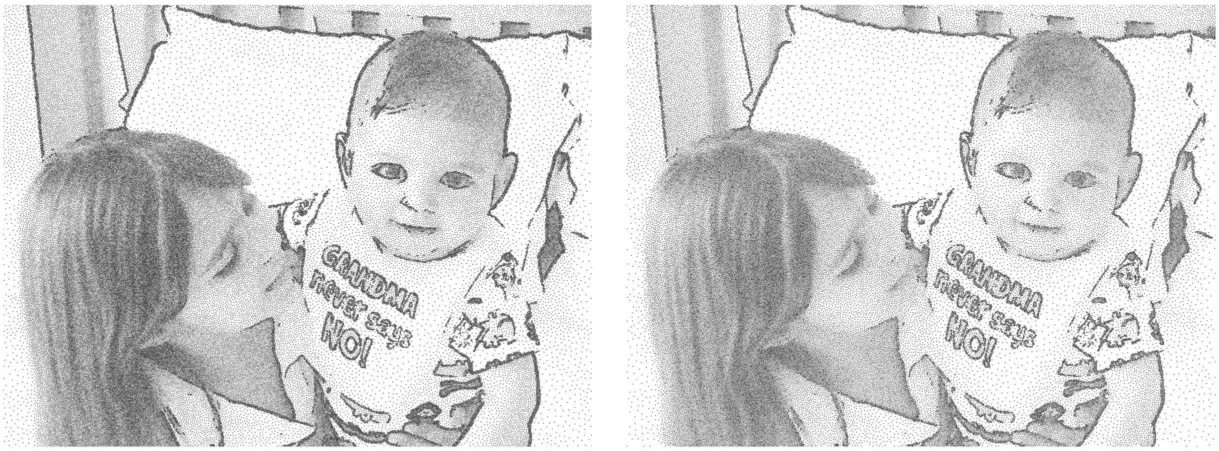


Figure 1: *A sifted point cloud (right) retains much of the visual quality of the original (left), but using fewer points. 113k points were reduced by 16% in 19 seconds. Sifted disks are maximal and satisfy the same sizing function as the original.*

Abstract

We introduce the Sifted Disk technique for locally resampling a point cloud in order to reduce the number of points. Two neighboring points are removed and we attempt to find a single random point that is sufficient to replace them both. The resampling respects the original sizing function; In that sense it is not a coarsening. The angle and edge length guarantees of a Delaunay triangulation of the points are preserved. The sifted point cloud is still suitable for texture synthesis because the Fourier spectrum is largely unchanged. We provide an efficient algorithm, and demonstrate that sifting uniform Maximal Poisson-disk Sampling (MPS) and Delaunay Refinement (DR) points reduces the number of points by about 25%, and achieves a density about 1/3 more than the theoretical minimum. We show two-dimensional stippling and meshing applications to demonstrate the significance of the concept.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computing Methodologies]: Computer Graphics—Computational Geometry and Object Modeling

1. Introduction

Maximal Poisson-disk Sampling (MPS) is a popular technique for generating random point clouds in the plane. Sample points are generated uniformly at random. Each sample point is the center of a disk which precludes the introduction of any other sample within it. For a maximal sample, there is no room to introduce another sample point, leading to the property that every point in the domain is within a sample disk. The sample points are separated-yet-dense: no two points are closer than the disk radius, yet the disks cover the entire domain. The minimum separation distance is a lower bound on the minimum edge length in any triangulation of the points, and the coverage property provides an upper bound on the largest Delaunay circumcircle. Together these bound the minimum (and maximum) angles in a Delaunay triangulation of the points. These properties are also known as well-spaced [Tal97, HMP06].

Delaunay Refinement (DR) is the most popular triangulation method in computational geometry [She96, She02]. Delaunay refinement follows a different construction than MPS, introducing points locally and deterministically based on the quality of an intermediate triangulation. But the end result is also a well-spaced point set.

Being well-spaced is crucial for provable quality guarantees in meshing applications: both the quality and the number of points affects the accuracy and efficiency of numerical simulations. The smallest edge length often determines the largest simulation time-step, e.g. in deformations and particle tracking schemes. The angles of the elements affect the numerical accuracy, which may in turn affect the convergence rate. Within a single step, processing time is typically proportional to the number of points.

In graphics, two-dimensional meshes are ubiquitous in online and offline rendering, animation, geometric algorithms such as collision detection, etc. Real-time meshing often benefits from offline preprocessing, regardless of offline speed; e.g. mesh smoothing [Tau00].

Poisson-disk Sampling (PS) is widely used in graphics. Lagae and Dutre [LD08] describe “sampling patterns for a wide range of applications,” notably high-quality ray tracing, as well as “object distribution, primitive distribution for illustration, and texture basis functions.” Bowers et al. [BWWM10] also note “texturing, remeshing, subsurface scattering, global illumination, non-photorealistic rendering, and point-based rendering.”

Since generating a Maximal PS is difficult, historically most applications have used non-maximal point clouds. Because PS only offers guarantees on the minimum point spacing and not on domain coverage (i.e. it is an undersampling), many applications desiring a sampling would benefit from a maximal one. Even when points come from some other process, maximality is often desired. For example, in DR mesh generation, a maximal packing improves both quality bounds and performance [AB04]. Recently algorithms for generating maximal samplings have become efficient enough for applications [Coo86, BWWM10, EPM*11, EMP*12, Fat11, GM09, JK11, LWSF10, WCE07].

Our contribution. We have found that point sets generated with the above methods have more points than are necessary to achieve the well-spaced property. Because the cost of processing or storing a point set grows with its size, reducing its size without compromising its quality is a clear win.

In this paper we show that not only is this reduction possible, it can be done efficiently and profitably. The sample count is reduced by about 25%. The density is about halfway between the original point cloud and the theoretical limit, or about 1/3 more points than the theoretical limit. The theoretical limit is the vertices of a tiling of equilateral triangles, with the longest possible edges for the disks to still cover the domain. The equilateral tiling has no random structure and poor spectra for graphics. We do not know of a reliable way to obtain sparser point clouds than ours that still satisfy the coverage requirement. We can sift a one-million-point input sample in about 40 seconds. Having fewer points is significant because each is a placeholder for some long calculation, such as finite-element simulations for meshes or interpolation or integration points for graphics. Since most commonly used graphics algorithms are about linear-time in the input size, a 25% reduction in points while maintaining MPS quality typically results in a 25% speedup for the application, essentially for free. High-quality renderings can take hours; the time it takes to sift a sample is much less. The same is true for meshes for finite element simulations.

Our method works on both constant-density and variable-density distributions. We respect the original spacing function, but pick points that are more efficient in satisfying that function. Thus we characterize our method as a resampling rather than a coarsening. Unlike the more predictable techniques of optimization-based point movement, this resampling process, and its outcome, are random; There is little impact on the Fourier spectrum of an MPS point cloud, which is important for texture synthesis. We demonstrate two-dimensional stippling and meshing applications.

Our sifting algorithm preserves the maximality property regardless of the source, and makes the point density adhere more closely to the sizing function. Off-centers is a variant of Delaunay refinement that reduces the point density. Sifting a uniform off-center point cloud doubles the reduction in point density, and improves the spectrum.

We call our method “sifting.” We restrict our attention to two-dimensions. Sifting in high dimensions would be subject to the same challenges as sampling in high dimensions.

2. Previous work

The computer graphics meshing literature has multiple approaches that reduce the size of meshes through various approximations. For instance, progressive meshes [Hop96] offers a continuous sequence of level-of-detail approximations of a source mesh, allowing tradeoffs between mesh size and accuracy. Alternatively, multiresolution mesh representations [SG05] efficiently manipulate meshes exceeding the size of main memory. A third option preserves appearance while reducing mesh size [COM98]. These methods focus

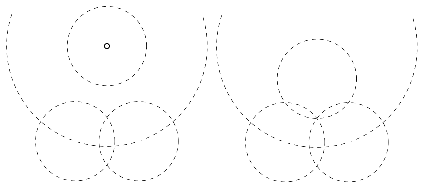


Figure 2: *Left: DR circumcenter insertion left a gap between the new (white) and old (black) points. Later this gap is filled with another point, increasing the density. Right: ODR avoids small gaps. The black points are co-circular (dashed) with a third point outside the figure, the vertices of a Delaunay triangle. The small solid circles are the r coverage disks.*

on reducing the mesh by approximating it, whereas we reduce the size without compromising the quality of the point distribution.

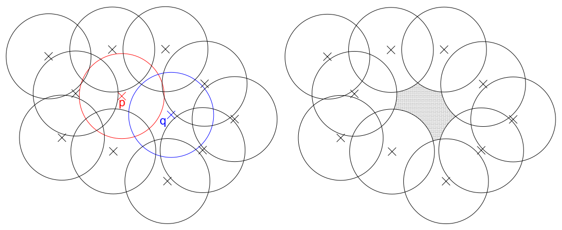
Point cloud manipulation shares several similarities with sifting. Simplification of point clouds can either remove selected points, leaving a true subset of the original surface [ABCO*01], or resample the surface to create new points [PGK02]. Both methods minimize error metrics, but neither maintains the exact original properties of the surface. These and many other methods leverage Delaunay triangulations (DT) of the point set. Floater and Reimers [FR01] performs 3d surface reconstruction from the local DT.

In mesh generation, Delaunay Refinement (DR) algorithms incrementally insert points until output conditions are achieved. Variants can be tailored to slightly different output properties: e.g. minimum angle [Rup95] or Voronoi aspect ratio [HMP06]. Chew's first DR algorithm [Che89] targets a particular mesh size; It is natural for comparing with uniform MPS since their outputs have identical maximality and coverage properties [EMD*11].

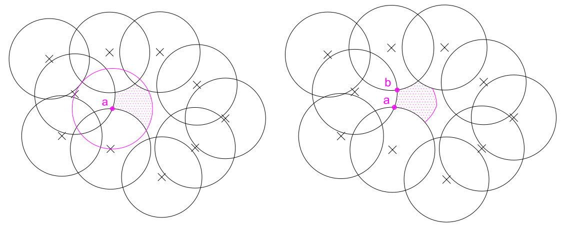
DR is typically based on circumcenter insertion. Circumcenters are vertices in the Voronoi diagram, the points furthest from the current sample points. Inserting centers is a greedy strategy for increasing coverage; It can paint itself into a corner and be forced to insert points in poor positions later. A point could be placed in some neighborhood of a circumcenter instead [CC09,FCC10]. Off-center point selection [Üng09] takes advantage of this freedom to place points more strategically. The exact position is calculated from the minimum angle of a triangle, with the goal of not leaving small gaps; see Figure 2. The final result is a coarser point cloud. More aggressive optimization upon insertion [EÜ09] can yield further improvements in some targeted quantity such as minimum Delaunay angle or Voronoi cell shape.

The bubble mesh [SG98] approach couples the adjustment of the location of points with dynamically adding and removing them.

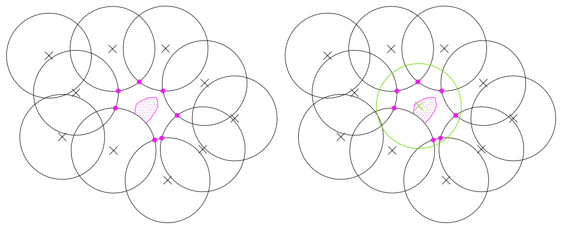
Rather than optimize *during* the point sampling, we follow a *post-process* sparsification, as e.g. [JÜ08]. Other mesh



(a) *Our algorithm tries to replace two neighboring sample points, p and q, with a single random point (left). The removal of candidates p and q creates the uncovered void (right).*



(b) *The replacement point must cover the whole void. Only a subregion of the void is good for covering void corner a (left). The subregion where a replacement sample covers both corners a and b is even smaller (right).*



(c) *The resampling subregion is the intersection of the subregions for all corners (left). Any random point from it would cover the whole void (right).*

Figure 3: *Sifting two disks p and q from a maximal sample.*

smoothing and optimization algorithms [TAD08, EÜZ09] are typically focused on quality metrics and lose original properties of the sample/mesh.

The authors are unaware of previous work that addresses the problem of reducing the size of a maximal and/or well-spaced distribution of points without affecting the maximality/well-spacedness of that distribution. One obvious approach is to just sample with a larger radius, i.e. a lower density. While simple, this increases the maximum distance between a domain point and a sample point, i.e. it is worse for interpolation and accuracy.

3. Algorithm overview

Figure 3 shows an overview of the sifted disk algorithm. We start with a point cloud sample. The sample could come from Poisson-disk sampling, Delaunay refinement, or any other method. We try to reduce the sample size by replacing a pair

(p,q) of samples with a single point o. The new sample preserves the maximal and disk-free conditions, and hence preserves the quality. For the non-uniform case, we perform a conservative resampling by enforcing both conditions even if the original did not locally satisfy the disk-free condition.

We iterate over all neighboring pairs of samples. We create the void that is uncovered when p and q are removed. We identifying a subregion of the void such that any sample point in it covers the whole void: equivalently, covers the void corners. If this subregion exists, we replace p and q with a random point o from it. Otherwise, we return p and q back to the sample and try the next candidate pair. We terminate when no candidate pair can be replaced. (A more aggressive strategy would be to then try to replace three points with two, etc.)

4. Algorithm details

We discuss the algorithm for the uniform case, constant r, then extend to the nonuniform case, varying $r(x)$. Given a spacing constant r and domain $\mathcal{D} \in \mathbb{R}^2$, a finite point set $X \subset \mathcal{D}$ is a maximal sampling if it satisfies two conditions:

$$\forall x_i, x_j \in X, i \neq j : \|x_i - x_j\| \geq r \quad (1a)$$
$$\forall p \in \mathcal{D}, \exists x_i \in X : \|p - x_i\| < r \quad (1b)$$

The first condition ensures that no two sample points are closer than r to each other. The second condition ensures every domain point is within r of some sample, and implies maximality [EPM*11].

Two points are *neighbors* iff their disks overlap. When replacing p and q with o, we define subsample X' as the neighbors of p or q, and $\mathcal{D}'$ as the subdomain of $\mathcal{D}$ inside the disks of p, q or X'. The *void* $\mathcal{V}$ is the subdomain of $\mathcal{D}'$ that is no longer covered, where (1b) does not hold, if p and q are removed.

The replacement steps for a pair of neighbors follow. As a preprocess we generate the Delaunay triangulation of the sample, $\mathcal{T}$. We also sort each sample's neighbors by counterclockwise angular order. We enforce (1a) and (1b) for $X' \cup o$ and $\mathcal{D}'$.

1. Retrieve the sorted neighbor lists of p and q, $\mathcal{L}_p$ and $\mathcal{L}_q$.
2. Splice the two lists to form a periodic sorted list, $\mathcal{L}_{pq}$, in order around $\mathcal{V}$.
3. Filter $\mathcal{L}_{pq}$ by discarding disks not on the boundary of $\mathcal{V}$.
4. Generate the *corners* of $\mathcal{V}$, the intersection point a of a disk with its successor in $\mathcal{L}_{pq}$ touching $\mathcal{V}$.
5. Choose a uniform random point o from $\mathcal{V}$ whose disk covers the whole void, all of its corners. Replace p and q in X by o and update $\mathcal{T}$. If o does not exist, retain p and q in X and try another pair.

4.1. Uniform sampling

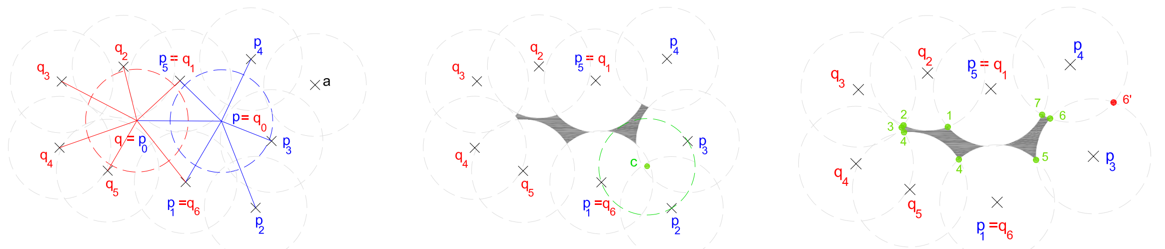
Splicing the lists. Given $\mathcal{L}_p$ and $\mathcal{L}_q$, the periodic sorted neighbors of each of p and q, the goal is to obtain one list $\mathcal{L}_{pq}$ in order around $\mathcal{V}$; see the shaded region in Figure 4c. In p's list, $\mathcal{L}_p$, simply replace q by q's list (sans p). Remove redundant points, i.e. each of the two common neighbors of p and q should appear in the list just once.

Filtering $\mathcal{L}_{pq}$. At this point, some disks in $\mathcal{L}_{pq}$ might not contribute a corner to the void because they are blocked by adjacent disks: e.g. p_2's disk in Figure 4b. Let l_1, l_2, l_3 denote any three consecutive points in $\mathcal{L}_{pq}$, and d_1, d_2, d_3 their disks. Let a be the intersection of d_1 and d_2 to the left of $\overrightarrow{l_1 l_2}$. The case we need to detect is that a is not a corner because it is actually inside the next disk, d_3. Point a is inside d_3 only if d_1 and d_3 overlap, and l_3 is also to the left of $\overrightarrow{l_1 l_2}$. Finding a and checking if it is in d_3 can be computed directly, but a faster check ($\times 3$ speedup) is to check if the point c equidistant from l_1, l_2, l_3 lies inside d_2. See "Correctness" below for why these checks are equivalent.

Void corners. Find the void corners a by computing the intersection of consecutive disks d_1, d_2, of points l_1, l_2 in $\mathcal{L}_{pq}$, and retaining the one to the left of $\overrightarrow{l_1 l_2}$; see Figure 4c.

Resampling. We find a point, o, whose disk covers all the void corners while preserving the minimum separation distance. Actually computing the geometry of the pink subregion to sample from, Figure 3c, would be relatively complex and expensive. Instead we use Ebeida et al.'s [EMP*12] flat quadtree method, which was developed for exactly this task. The idea is to keep a set of squares that form an outer geometric approximation to the pink region, try sampling from the squares, and refine and discard squares if sampling fails. We start with a set of squares containing the void, the bounding box of the void corners. When a square is refined, we check if it is too far away from a corner for a sample point to cover the corner; these squares are completely outside the pink region in Figure 3c and are discarded. Squares inside a disk of some l_i are also discarded. We sample o uniformly from the remaining squares. We accept the first sample whose disk covers all the corners; since a disk is convex, it will also cover the entire void. If, after a constant number of attempts, no acceptable o was found, we refine all the squares to better approximate the region and repeat. If no squares remain, the pink region is empty and no such point o exists; we return p and q to the sample and try another pair. Otherwise, we replace (p,q) with o.

Retriangulation See Figure 5. The local triangulation is performed by removing any edge between p or q and a point in $\mathcal{L}_{pq}$. We then calculate the Delaunay edges between o and $\mathcal{L}_{pq}$, possibly removing edges between points in $\mathcal{L}_{pq}$. Computing the local Delaunay triangulation for maximal disk-free points has been described in detail [EMD*11]. Because of the angle bounds, the changes are local and do not propagate beyond the void, and can be done in constant time.



(a) *Retrieval of disks bounding the void. Here $\mathcal{L}_p$ is $\{p_0, \dots p_5\}$ and $\mathcal{L}_q$ is $\{q_0, \dots q_6\}$. Using $p_0 = q$, $q_0 = p$, $p_1 = q_6$, and $p_5 = q_1$, the spliced list $\mathcal{L}_{pq}$ is $\{q_1, \dots q_6, p_2, \dots p_4\}$*

(b) *We filter $\mathcal{L}_{pq}$ by removing any disk that does not bound the void. Here p_2 is removed because the center of the circumcircle of p_1, p_2, p_3 is covered and p_2 lies on the outside of $\overrightarrow{p_1 p_3}$.*

(c) *The void corners are the intersection of consecutive disks in $\mathcal{L}_{pq}$. The corners are the vertices of a polygon containing the void. We retained 6 and discarded 6′ because 6′ lies to the right (outside) of $\overrightarrow{p_3 p_4}$.*

Figure 4: *Local sifting example. After generating void corners, we sample a replacement point from the void with the constraint that its disk covers all the void corners.*

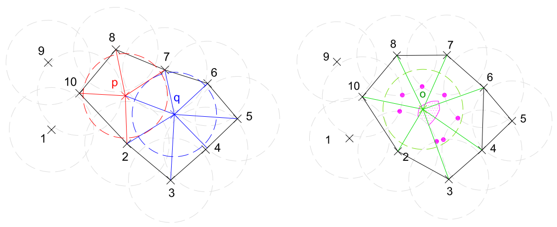
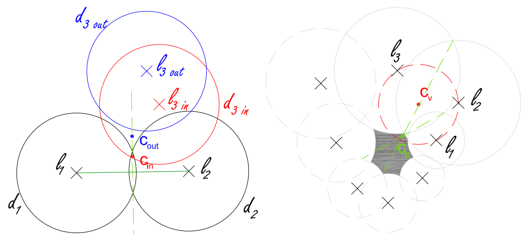


Figure 5: *Resampling requires local re-triangulation.*

Correctness. For correctness we must prove that we found at least all the disks bounding the void, then filtered only those that did not contribute to the void. The first condition is trivial: an uncovered corner $\in d_1 \cap d_2$ is a point inside d_p or d_q, so both l_1 and l_2 are neighbors of p or q and gathered initially. (It can also be shown that these samples are vertices of a Delaunay triangle.)

It remains to show that no filtered disk touches the void $\mathcal{V}$. A Voronoi cell V_2 is defined as the set of points closer to l_2 than any other sample point. A Voronoi cell is bounded by separators. A separator is the perpendicular bisector of the segments between two sample points. These perpendicular bisectors contain the Voronoi edges, and also the chord between the intersections of two circles.

In particular, we show d_2 does not contribute a corner iff the point c equidistant from l_1, l_2, l_3 lies inside d_2. The left intersection a of d_1 and d_2 might possibly bound the void. See Figure 6a. The point c equidistant from points l_1, l_2, l_3 is a Voronoi vertex, the center of the Delaunay circumcircle of the three points. The perpendicular bisector of $\overline{l_1 l_2}$ contains a, c, and the Voronoi edge e_{12} separating the Voronoi cell V_1 for l_1 from V_2 for l_2. If c lies outside d_2, then $r = \|a - l_2\| < \|c - l_2\| = \|c - l_3\|$. Moreover the segment $\overline{ca}$ lies inside e_{12}; specifically a lies outside V_3 and hence $\|a - l_3\| > \|c - l_3\|$; therefore $\|a - l_3\| > r$ and a is outside d_3 and is a void corner.



(a) *Uniform, Voronoi vertex $a \in d_3 \Leftrightarrow c \in d_2$.* **(b)** *Nonuniform, power vertex $a \in d_3 \Leftrightarrow c_p \in d_2$.*

Figure 6: *Corner coverage detection.*

Conversely, if c lies inside d_2, then $r = \|a - l_2\| > \|c - l_2\| = \|c - l_3\|$. Moreover, a lies in V_3, so is closer to l_3 than to l_2: $\|a - l_3\| < \|a - l_2\| = r$. Hence a lies inside d_3.

4.2. Nonuniform sampling

Unfortunately, given a non-constant spacing function $r(x)$ over domain $\mathcal{D}$, it is not in general possible to simultaneous guarantee conflict-free disks (1a) and complete domain coverage (1b). Mitchell et al. [MREB12b] describes several generalizations of (1a) and (1b). We choose the *smaller-disk* conflict criteria MPS as our preferred input, and resample using a form of the *smaller-disk* and *prior-disks* criteria, because they provide coverage and a simple definition of the void. Specifically, point o is a valid resampled point if it satisfies two conditions:

$$\forall x_i \in X' : \|x_i - o\| \geq r(x_i) \qquad (2a)$$

$$\forall p \in \mathcal{D}', \exists x_i \in \{X \cup o\} : \|p - x_i\| < r(x_i) \qquad (2b)$$

As in the uniform case, we enforce the conflict and coverage conditions locally. The original sample might not have satisfied (2a), but we require it to hold when we resample anyway. In that sense our resampling is a strict improvement. Nonuniform sampling requires slightly different handling in the Filtering and Resampling steps. (By non-uniform we mean that the sampling density varies spatially, but the probability of selecting the next random point within a given subregion is still proportional to the subregion's area.)

The key change for the nonuniform case is to use a weighted power diagram and its dual Regular triangulation [AE84]. The standard construction uses weights equal to the disk radii (sizing function). There is a nice geometric correspondence between power diagrams and our disks and points [YW12]. As before, two points are neighbors if their disks overlap. The line separating the Voronoi cells of neighbors l_1 and l_2 is straight and perpendicular to $\overline{l_1 l_2}$ but shifted if the disks d_1 and d_2 have unequal radii: it is the line passing through the intersection points of d_1 and d_2. As in the Voronoi diagram, for three weighted points the three separating lines meet at a common point, the power vertex c_P.

Filtering. For the equidistant-point-in-d_3 check we use the power vertex instead of the Voronoi vertex. The correctness of this follows directly from the redefinition of distance and separators. Figure 6b shows an example where using the power vertex c_P gives the correct answer, but the unweighted Voronoi vertex c_V does not.

Resampling. In the uniform case, the disk of any sample in a flat quadtree square cannot cover a corner if the square is farther than r from the corner. Such squares are discarded. For the nonuniform case, the radius is changing. Since we cannot check the radius at every point in the square, we adopt a conservative test based on the sizing function (radius) at the square's center, the size of the square, and the maximum rate of change of the sizing function [MREB12a]. The test is a sufficient condition for discarding the cell, and becomes more accurate (closer to necessary) as the square is refined.

4.3. Other conflict criteria

Mitchell et al. [MREB12a] defines other conflict conditions such as the larger disk containing the center of the smaller disk, or considering the order in which the samples were drawn. For these, revising the condition of when to discard a flat quadtree square is fairly straightforward. In addition, the mean-radius conflict condition is interesting because it produces a (half-radius) disk packing. The main challenges are that the definitions of corners and the sample region may depend on the unknown replacement disk. We leave these variations, and higher-dimensions, for future work.

5. Maximal point cloud densities

Delaunay refinement is the worst; off-centers is better; and sifting is the best, in terms of density. Table 1 summarizes the average point density and Delaunay edge lengths (separation

sample type	point density	relative density	Delaunay edge lengths
$\triangle(r)$	$\frac{2}{\sqrt{3}}r^{-2}$	3	$\{r\}$
$\square(r)$	r^{-2}	2.60	$\{r, \sqrt{2}r\}$
⬡(r)	$\frac{4}{3\sqrt{3}}r^{-2}$	2	$\{r, \sqrt{3}r, 2r\}$
DR(r)	$0.75\,r^{-2}$	1.95	$[r, 2r)$
MPS(r)	$0.70\,r^{-2}$	1.82	$[r, 2r)$
ODR(r)	$0.64\,r^{-2}$	1.66	$[r, 2r)$
sDR(r)	$0.57\,r^{-2}$	1.48	$[r, 2r)$
sMPS(r)	$0.51\,r^{-2}$	1.33	$[r, 2r)$
sODR(r)	$0.51\,r^{-2}$	1.33	$[r, 2r)$
$\square(\sqrt{2}r)$	$\frac{1}{2}r^{-2}$	1.30	$\{\sqrt{2}r, 2r\}$
$\triangle(\sqrt{3}r)$	$\frac{2}{3\sqrt{3}}r^{-2}$	1	$\{\sqrt{3}r\}$

Table 1: *(Average) point density and edge length ranges of different maximal samples with uniform radius r.*

distances) before and after sifting. Figure 8 shows quality plots before and after sifting. We include the average output of some common algorithms. We also include some extremal distributions, in order to provide a theoretical upper bound on how much a point cloud could be improved.

- $\triangle(r)$ is the points at the corners of the lattice of equilateral triangles with side length r.
- $\square(r)$ is the square lattice with side length r, diagonal length $\sqrt{2}r$.
- ⬡(r) is the hexagonal lattice with side length r, diagonal length $2r$.
- MPS is Maximal Poisson-disk sampling.
- DR is Delaunay refinement, Delaunay circumcenter insertion.
- ODR is off-center Delaunay refinement.
- sMPS, sDR, sODR are sifted MPS, DR, and ODR.

$\triangle(r)$ is the densest sampling respecting the minimum separation distance, whereas $\triangle(\sqrt{3}r)$ is the least dense sampling that still respects the maximality criteria. $\square(\sqrt{2}r)$ and ⬡(r) have the longest Delaunay edge lengths possible, $2r$, while still respecting the maximality criteria. (The densest packings by dimension is a popular research topic [NS12].)

All these distributions satisfy the same inhibition/coverage criteria, but the variations are significant: $\triangle(r)$ has three times as many points as $\triangle(\sqrt{3}r)$, and half the maximum edge length of $\square(\sqrt{2}r)$ and ⬡(r).

The aim of sifting is to obtain random points with density close to that of $\triangle(\sqrt{3}r)$. In the case of MPS, we maintain the inherent (and desirable) randomness of the sample. In the case of DR and ODR, we *introduce* desirable randomness. Sifting MPS results provides two main advantages over ODR: a larger reduction in the size of the sample as well as a clean Fourier spectrum, discussed in Section 5.1.

For the algorithmically generated clouds, each sifted cloud has fewer points than its original cloud; indeed, every sifted cloud has fewer points than *any* of the original clouds! Quantitatively, the relative density is reduced to about that of the sparse square lattice. Off-centers [Üng09] was lauded as a signal achievement in reducing the point density of DR. In the uniform case it reduces relative density from 1.95 to 1.66, whereas sifting reduces DR about 50% more, to 1.48. Furthermore, sifting reduces ODR from 1.66 to 1.33, twice the improvement of ODR alone. Sifting also improves the spectrum. (Off-centers reportedly produces a larger benefit over DR in the non-uniform case.) sMPS and sODR produce about the same number of points, but MPS starts with 7% fewer points than ODR. The completely structured mesh $\triangle(\sqrt{3}r)$ has the theoretical minimum density, 1, but has a terrible spectrum. Sifting transforms point clouds closer to the theoretically best density, while still providing a good spectrum for graphics.

Variance in the MPS and sMPS densities decreases as the sample size grows. For DR, ODR, and their sifted variants, sample sizes are dependent on the precise specification of the algorithm (e.g. queue ordering); the results are given for the most common settings.

5.1. Output spectrum, triangle angles and edge lengths

We analyze quality with a variety of sample distribution properties. The Fourier spectrum is a useful measure for demonstrating "blue noise" properties of the output (particularly desirable in computer graphics). The Point Set Analysis (PSA) [Sch11] tool allows standardized comparison plots. Additionally, we consider histograms of Delaunay-Voronoi diagram properties: Voronoi aspect ratio, angles of Delaunay triangles, and Delaunay edge lengths. We compare results in Figures 7 and 8. Sifting tends to smooth the distributions, and shift them towards larger Voronoi cells, longer mesh edges, and smaller point density.

In addition to an artifact-free Fourier spectrum, MPS produces smooth histograms for each of the Delaunay metrics. This makes intuitive sense: random sampling should ensure nearby quantities have similar probability. DR with strict circumcenter insertion typically gives a clean Fourier spectrum. On the other hand, the ODR spectrum has a different nature than the regular diminishing oscillations seen with MPS and DR. This change is likely due to the way the algorithm inserts points much more systematically, thus promoting locally regular patterns. DR and ODR angle histograms are not smooth but instead have a noticeable peak around 60 degrees. By design, ODR produces longer Delaunay edge lengths than DR (or MPS). While the most probably MPS and DR Delaunay edge lengths are near the disk radius, ODR produces the most edges near 1.7 times the disk radius. This reflects the attempt by ODR to space points in a similar fashion to the sparsest maximal equilateral lattice.

The sifting algorithm promotes some similar changes in

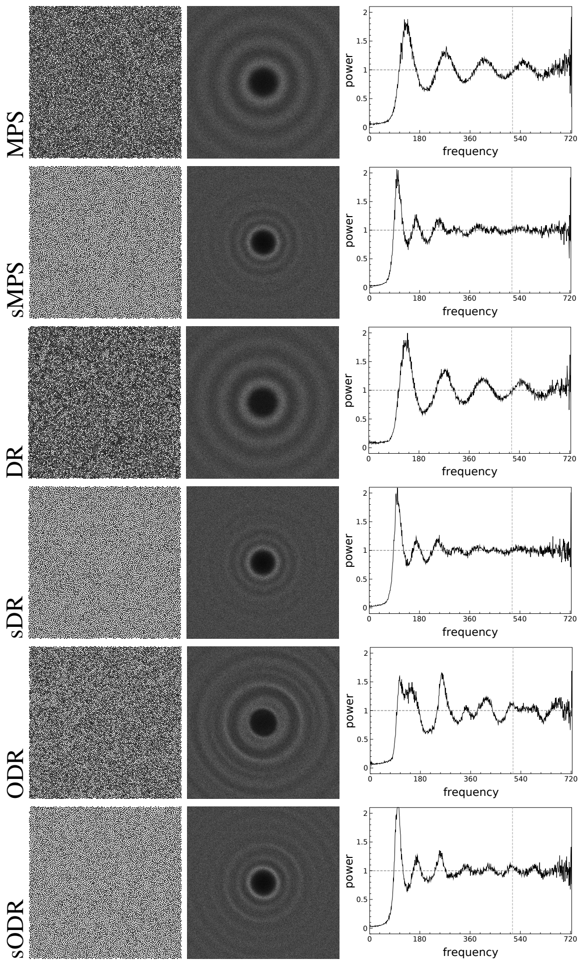


Figure 7: *Spectrums before and after sifting. The left is sample points; middle their Fourier spectrum; and right their radial power. Computed with PSA.*

these various metrics independent of the input sample. Compared to non-sifted algorithms, oscillations in the Fourier spectrum are compressed and the first peak is higher. The resulting spectrum retains the desirable characteristics of minimal low frequency energy and no radial artifacts. Sifting of DR and MPS alters the Delaunay edge length histograms, replacing triangulations with predominantly short edges near the disk length with ones with mostly longer edges between 1.6 and 2.0 times the disk length. This represents significant progress towards the smallest possible sampling: $\triangle(\sqrt{3})$ with all edge lengths $\sqrt{3} \approx 1.7$. Sifted ODR has more long edges than its unsifted counterpart, but the impact is less dramatic since the ODR samples already have a larger share of long edges. Sifting alters the DR and ODR histograms but generally does not eliminate their non-smooth nature.

Finally, we note that sifting does not introduce a directional bias in the Delaunay edges; see Figure 9a. Uniformly-

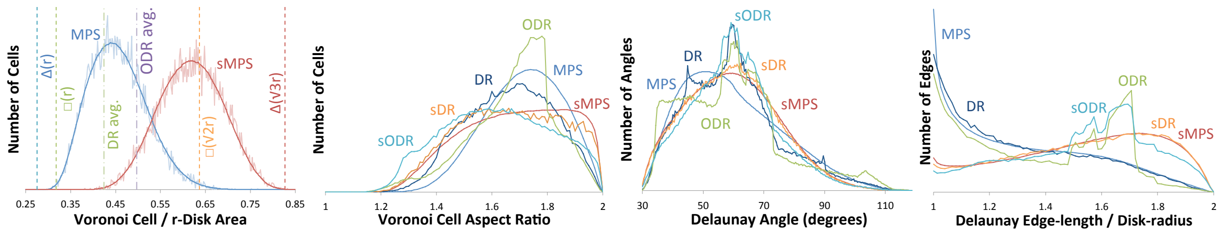
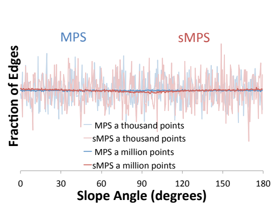
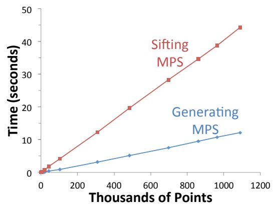


Figure 8: *Impact of sifting on meshing quality measures, for different input.*



(a) *Sifting does not introduce an orientation bias to Delaunay edges: slope is uniform random.*



(b) *Sifting time is linear in the number of input points, and roughly 4× longer than generating an MPS input.*

Figure 9: *Sifted edge orientations and sifting time.*

	MPS	sMPS	DR	sDR
interior points	580	419	580	417
...reduction	-	27%	-	28%
min angle	30.6	30.5	31.7	30.6
max angle	115.5	114.7	110.7	115.5
min Vor ratio	1.30	1.23	1.26	1.24
max Vor ratio	1.994	1.998	1.982	1.998

Table 2: *Mesh sifting savings and quality.*

oriented edges is an important property for certain meshing and fracture simulation applications [EKL*11].

5.2. Performance

On average, sifting an MPS point cloud requires 4 times as long as generating the MPS sample. We sift one million points in about 40 seconds, and the sifting process usually reduces the size of the sample by about 25%. Empirically our sifting code takes linear time as demonstrated in Figure 9b.

There are also some theoretical reasons to expect linear run-time: there are a linear number of pairs and each takes constant time to consider. For the uniform case, the number of neighbors of a single point is a constant [EPM*11, EMD*11]. This means that there are a linear number of pairs before replacement. Empirically a linear number of pairs (half of them) are replaced, so a linear number of pairs total are considered. Moreover, all of the local data structures such as neighbor lists are of constant size. The flat quadtree method is empirically linear in the number of samples produced [EMP*12]. So the steps for a single pair take constant expected time. This will also hold in the non-uniform case if the sizing function (disk radii) does not vary too quickly, but with worse constants [MREB12a]. In any case, run-time is not *worst-case* linear because the location of a resample candidate is random, and its location could be persistently perverse, albeit with low probability.

6. Applications

6.1. Stippling

We describe the process used to produce our stippling images, such as Figure 1. Given a grayscale image, we perform edge detection.

We define a sizing function:

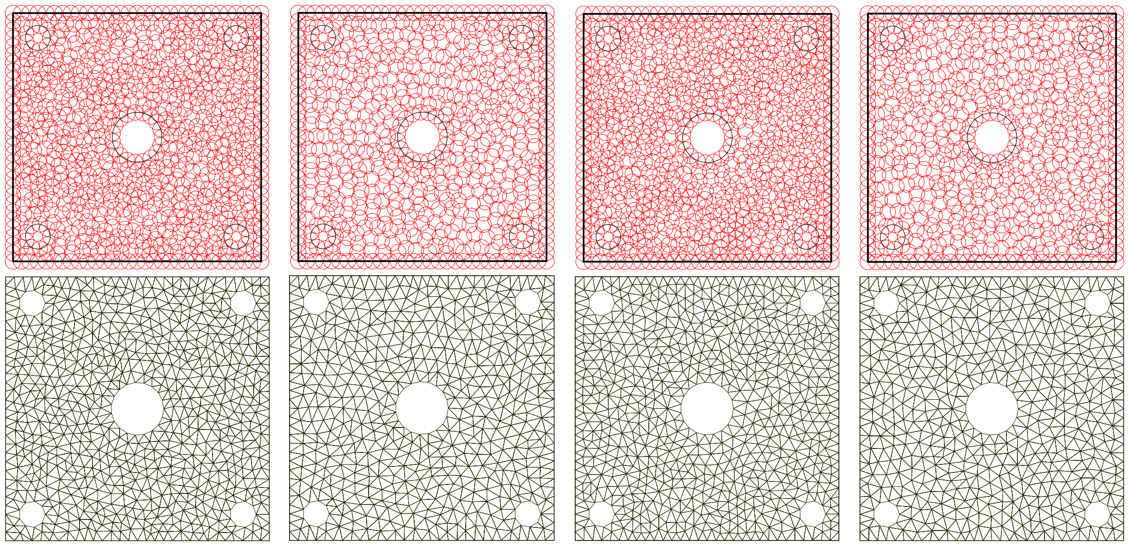
$$r(x) = r_{\min} + (r_{\max} - r_{\min})g^2(x) = r_{\min}(1 + Ag^2(x))$$

Here $g(x) \in [0,1]$ is the grayscale value of the pixel containing x, and is constant over the pixel. Here $r_{\max} \Leftrightarrow A$ scales g and controls the contrast in the stippling density; we chose $A = 9$. Here $r_{\min}$ is the length of the diagonal of a pixel. A lower bound of $r_{\min}$ on the sizing function ensures that each pixel accepts at most one sample point. This has the affect of hiding the jumps in $g(x)$ at pixel boundaries.

We generate an MPS using the minimum-disk conflict criteria: $\|x_i - x_j\| \geq \min(r(x_i), r(x_j))$. We find a Regular triangulation using $r(x)$ for the weight of x. Finally, we sift the point set as described in Section 4. To produce the image we overlay the sifted points with the detected edges.

6.2. Meshing

Figure 10 shows uniform point clouds and Delaunay triangulations before and after sifting. Table 2 shows the fraction of interior points saved, and quality measures. In the table, "Vor ratio" means the aspect ratio of a Voronoi cell: the ratio of the distances from the farthest Voronoi vertex to the sample, and the closest Voronoi edge to the sample.



(a) *MPS.* **(b)** *sMPS.* **(c)** *DR.* **(d)** *sDR.*

Figure 10: *Mesh sifting application.*

7. Conclusions

We have shown that it is possible to resample a point cloud to use fewer points, while still respecting the input sizing function. We can still produce well shaped meshes and random samples for stippling. An important extension is to curved surfaces, resampling based on the curvature function and adaptively by viewpoint. In the future we wish to explore higher dimensional sampling problems.

We appear to be able to resample over rapidly changing sizing functions. We would like to explore the theoretical limits.

Acknowledgements

The UC Davis author thanks the SciDAC Institute for Ultrascale Visualization and the National Science Foundation (grant # CCF-1017399) for supporting this work.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

[AB04] ATTALI D., BOISSONNAT J.-D.: A linear bound on the complexity of the Delaunay triangulation of points on polyhedral surfaces. *Discrete & Computational Geometry 31*, 3 (Feb. 2004), 369–384. doi:10.1007/s00454-003-2870-4. 2

[ABCO*01] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Point set surfaces. In *IEEE Visualization 2001* (Oct. 2001), pp. 21–28. doi:10.1109/VISUAL.2001.964489. 3

[AE84] AURENHAMMER F., EDELSBRUNNER H.: An optimal algorithm for constructing the weighted Voronoi diagram in the plane. *Pattern Recognition 17*, 2 (1984), 251 – 257. doi:10.1016/0031-3203(84)90064-5. 6

[BWWM10] BOWERS J., WANG R., WEI L.-Y., MALETZ D.: Parallel Poisson disk sampling with spectrum analysis on surfaces. *ACM Transactions on Graphics 29* (Dec. 2010), 166:1–166:10. doi:10.1145/1882261.1866188. 2

[CC09] CHERNIKOV A., CHRISOCHOIDES N.: Generalized two-dimensional Delaunay mesh refinement. *SIAM J. Sci. Comput. 31* (2009), 3387–3403. doi:10.1137/080723028. 3

[Che89] CHEW L. P.: *Guaranteed-Quality Triangular Meshes*. Tech. Rep. 89-983, Department of Computer Science, Cornell University, 1989. 3

[COM98] COHEN J., OLANO M., MANOCHA D.: Appearance-preserving simplification. In *Proceedings of SIGGRAPH 98* (July 1998), Computer Graphics Proceedings, Annual Conference Series, pp. 115–122. doi:10.1145/280814.280832. 2

[Coo86] COOK R. L.: Stochastic sampling in computer graphics. *ACM Transactions on Graphics 5*, 1 (Jan. 1986), 51–72. 2

[EKL*11] EBEIDA M. S., KNUPP P. M., LEUNG V. J., BISHOP J. E., MARTINEZ M. J.: Mesh generation for modeling and simulation of carbon sequestration process. In *DOE Scientific Discovery through Advanced Computing (SciDAC)* (2011). 8

[EMD*11] EBEIDA M. S., MITCHELL S. A., DAVIDSON A. A., PATNEY A., KNUPP P. M., OWENS J. D.: Efficient and good Delaunay meshes from random points. *Computer-Aided Design 43*, 11 (2011), 1506 – 1515. Solid and Physical Modeling 2011. URL: http://www.sciencedirect.com/science/article/pii/S0010448511002119, doi:10.1016/j.cad.2011.08.012. 3, 4, 8

[EMP*12] EBEIDA M. S., MITCHELL S. A., PATNEY A., DAVIDSON A. A., OWENS J. D.: A simple algorithm for maximal Poisson-disk sampling in high dimensions. *Computer Graphics Forum 31*, 2 (May 2012), 785–794. doi:10.1111/j.1467-8659.2012.03059.x. 2, 4, 8

[EPM*11] EBEIDA M. S., PATNEY A., MITCHELL S. A., DAVIDSON A., KNUPP P. M., OWENS J. D.: Efficient maximal Poisson-disk sampling. *ACM Transactions on Graphics 30*, 4 (2011), 49:1–49:12. doi:10.1145/1964921.1964944. 2, 4, 8

[EÜ09] ERTEN H., ÜNGÖR A.: Quality triangulations with locally optimal Steiner points. *SIAM Journal on Scientific Computing 31* (Feb. 2009), 2103–2130. doi:10.1137/080716748. 3

[EÜZ09] ERTEN H., ÜNGÖR A., ZHAO C.: Mesh smoothing algorithms for complex geometric domains. *Proceedings of the 18th International Meshing Roundtable* (2009), 175–193. doi:10.1007/978-3-642-04319-2_11. 3

[Fat11] FATTAL R.: Blue-noise point sampling using kernel density model. *ACM Transactions on Graphics 30*, 4 (July 2011), 48:1–48:12. doi:10.1145/2010324.1964943. 2

[FCC10] FOTEINOS P., CHERNIKOV A., CHRISOCHOIDES N.: Fully generalized two-dimensional constrained Delaunay mesh refinement. *SIAM Journal on Scientific Computing 32* (2010), 2659–2686. doi:10.1137/090763226. 3

[FR01] FLOATER M. S., REIMERS M.: Meshless parameterization and surface reconstruction. *Computer Aided Geometric Design 18*, 2 (2001), 77–92. doi:10.1016/S0167-8396(01)00013-9. 3

[GM09] GAMITO M. N., MADDOCK S. C.: Accurate multidimensional Poisson-disk sampling. *ACM Transactions on Graphics 29*, 1 (Dec. 2009), 8:1–8:19. doi:10.1145/1640443.1640451. 2

[HMP06] HUDSON B., MILLER G., PHILLIPS T.: Sparse Voronoi refinement. In *Proceedings of the 15th International Meshing Roundtable* (Sept. 2006), Pébay P. P., (Ed.), pp. 339–356. Long version available as Carnegie Mellon University Technical Report CMU-CS-06-132. doi:10.1007/978-3-540-34958-7_20. 2, 3

[Hop96] HOPPE H.: Progressive meshes. In *Proceedings of SIGGRAPH 96* (Aug. 1996), Computer Graphics Proceedings, Annual Conference Series, pp. 99–108. doi:10.1145/237170.237216. 2

[JK11] JONES T. R., KARGER D. R.: Linear-time Poisson-disk patterns. *Journal of Graphics, GPU, and Game Tools 15*, 3 (Oct. 2011), 177–182. doi:10.1080/2151237X.2011.617173. 2

[JÜ08] JAMPANI R., ÜNGÖR A.: Construction of sparse well-spaced point sets for quality tetrahedralizations. In *Proceedings of the 16th International Meshing Roundtable*, Brewer M. L., Marcum D., (Eds.). Springer Berlin Heidelberg, Oct. 2008, pp. 63–80. doi:10.1007/978-3-540-75103-8_4. 3

[LD08] LAGAE A., DUTRÉ P.: A comparison of methods for generating Poisson disk distributions. *Computer Graphics Forum 27*, 1 (2008), 114–129. 2

[LWSF10] LI H., WEI L.-Y., SANDER P. V., FU C.-W.: Anisotropic blue noise sampling. *ACM Transactions on Graphics 29*, 6 (Dec. 2010), 167:1–167:12. doi:10.1145/1882261.1866189. 2

[MREB12a] MITCHELL S. A., RAND A., EBEIDA M. S., BAJAJ C.: Variable radii Poisson-disk sampling. In *Proceedings of the 24th Canadian Conference on Computational Geometry* (2012), pp. 185–190. 6, 8

[MREB12b] MITCHELL S. A., RAND A., EBEIDA M. S., BAJAJ C.: Variable radii Poisson-disk sampling, extended version. In *Proceedings of the 24th Canadian Conference on Computational Geometry* (2012). 5

[NS12] NEBE G., SLOANE N.: A catalogue of lattices. http://www.math.rwth-aachen.de/~Gabriele.Nebe/LATTICES/index.html, 2012. 6

[PGK02] PAULY M., GROSS M., KOBBELT L. P.: Efficient simplification of point-sampled surfaces. In *IEEE Visualization 2002* (Oct. 2002), pp. 163–170. doi:10.1109/VISUAL.2002.1183771. 3

[Rup95] RUPPERT J.: A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms 18*, 3 (May 1995), 548–585. doi:10.1006/jagm.1995.1021. 3

[Sch11] SCHLÖMER T.: PSA point set analysis. Version 0.2.2, http://code.google.com/p/psa/, 2011. 7

[SG98] SHIMADA K., GOSSARD D.: Automatic triangular mesh generation of trimmed parametric surfaces for finite element analysis. *Computer Aided Geometric Design 15*, 3 (1998), 199–222. 3

[SG05] SHAFFER E., GARLAND M.: A multiresolution representation for massive meshes. *IEEE Transactions on Visualization and Computer Graphics 11*, 2 (Mar./Apr. 2005), 139–148. doi:10.1109/TVCG.2005.18. 2

[She96] SHEWCHUK J.: Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. *Applied computational geometry towards geometric engineering* (1996), 203–222. 2

[She02] SHEWCHUK J. R.: Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry 22*, 1–3 (May 2002), 86–95. doi:10.1016/S0925-7721(01)00047-5. 2

[TAD08] TOURNOIS J., ALLIEZ P., DEVILLERS O.: Interleaving Delaunay refinement and optimization for 2D triangle mesh generation. In *Proceedings of the 16th International Meshing Roundtable* (2008), Springer, pp. 83–101. doi:10.1007/978-3-540-75103-8_5. 3

[Tal97] TALMOR D.: *Well-Spaced Points for Numerical Methods*. PhD thesis, Carnegie Mellon University, Pittsburgh, Aug. 1997. CMU CS Tech Report CMU-CS-97-164. 2

[Tau00] TAUBIN G.: Geometric signal processing on polygonal meshes. In *Eurographics 2000, State of the Art Reports* (Aug. 2000), Coquillart S., Duke D., (Eds.), Eurographics Association. 2

[Üng09] ÜNGÖR A.: Off-centers: A new type of Steiner points for computing size-optimal quality-guaranteed Delaunay triangulations. *Compututational Geometry: Theory and Applications 42* (Feb. 2009), 109–118. doi:10.1016/j.comgeo.2008.06.002. 3, 7

[WCE07] WHITE K. B., CLINE D., EGBERT P. K.: Poisson disk point sets by hierarchical dart throwing. In *RT '07: Proceedings of the 2007 IEEE Symposium on Interactive Ray Tracing* (Sept. 2007), pp. 129–132. doi:10.1109/RT.2007.4342600. 2

[YW12] YAN D.-M., WONKA P.: Gap processing for adaptive maximal Poisson-disk sampling. *CoRR abs/1211.3297* (2012). 6