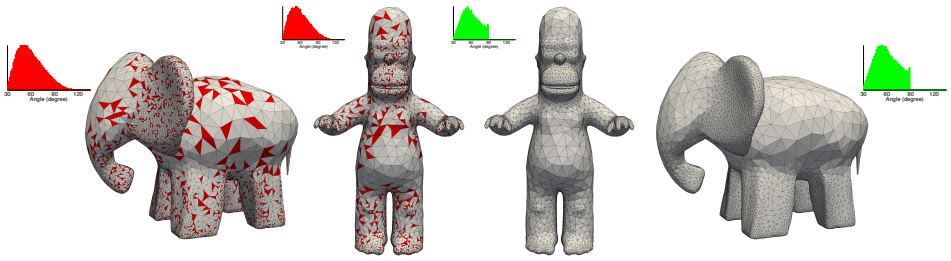
Eurographics Symposium on Geometry Processing 2016
Maks Ovsjanikov and Daniele Panozzo
(Guest Editors)

Volume 35 (2016), Number 5

Disk Density Tuning of a Maximal Random Packing

Mohamed S. Ebeida[1], Ahmad A. Rushdi[1,2], Muhammad A. Awad[3], Ahmed H. Mahmoud[4], Dong-Ming Yan[5], Shawn A. English[1], John D. Owens[4], Chandrajit L. Bajaj[2], and Scott A. Mitchell[1]

[1]Sandia National Laboratories, Albuquerque, NM [2]University of Texas, Austin, TX [3]Alexandria University, Alexandria, Egypt
[4]University of California, Davis, CA [5]NLPR, Institute of Automation, CAS, China



(a) Initial triangulation with obtuse angles. (b) Tuned non-obtuse triangulations.

Figure 1: Tuning nonuniform meshes on curved surface elephant and homer models, successfully eliminating obtuse angles.

Abstract

We introduce an algorithmic framework for tuning the spatial density of disks in a maximal random packing, without changing the sizing function or radii of disks. Starting from any maximal random packing such as a Maximal Poisson-disk Sampling (MPS), we iteratively relocate, inject (add), or eject (remove) disks, using a set of three successively more-aggressive local operations. We may achieve a user-defined density, either more dense or more sparse, almost up to the theoretical structured limits. The tuned samples are conflict-free, retain coverage maximality, and, except in the extremes, retain the blue noise randomness properties of the input. We change the density of the packing one disk at a time, maintaining the minimum disk separation distance and the maximum domain coverage distance required of any maximal packing. These properties are local, and we can handle spatially-varying sizing functions. Using fewer points to satisfy a sizing function improves the efficiency of some applications. We apply the framework to improve the quality of meshes, removing non-obtuse angles; and to more accurately model fiber reinforced polymers for elastic and failure simulations.

1. Introduction

A two-dimensional disk packing of a domain $\mathcal{D}$ is the arrangement of a set of radius-r disks with center points $P = \{p_i\}$ within the domain. Three quality properties are typically desired for graphics applications. Specifically, a packing must be:

1. Conflict-free: A packing is said to be *conflict-free* if no disk contains the center of another disk. That is, $\forall p_i, p_j \in P, i \neq j : \|p_i - p_j\| \geq r$. For constant radii, this is equivalent to the $r/2$-disks (centered at the same points) not overlapping. A violation of this condition is called a *conflict*.

2. Maximal: A packing is *maximal* if adding any new disk would generate a conflict. For constant radii, this is equivalent to the disks covering the entire domain. That is, $\forall p \in \mathcal{D}, \exists p_i \in P : \|p - p_i\| < r$. For many applications, maximality provides accuracy, as any domain point is represented by a nearby disk center. The same conflict-free and maximality conditions apply to non-uniform (spatially varying radii) disk packings. Note that maximal packings are not unique, or necessarily *maximum* (densest/sparsest); it is often possible to move disks around to make room to add another disk without generating a conflict. Conversely, it is often possible to remove a disk and move others to still cover the domain. We define the *density* of a packing as the fraction of the domain area covered by non-overlapping half-radius disks.

3. Random: A packing process is said to be *random* or *bias-free* if it does not favor any specific region of the uncovered part of the

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

domain when adding a new sample. This implies that the likelihood of the next sample p_i lying inside any uncovered subdomain Ω is proportional to the area of that subdomain. This is equivalent to *uniform sampling* from the uncovered part of the domain. That is $\forall p_i, \forall \Omega \in \mathcal{D}_{i-1} : prob(p_i \in \Omega) = \text{area}(\Omega)/\text{area}(\mathcal{D}_{i-1})$, where $\mathcal{D}_{i-1}$ is the union of all the uncovered regions of the domain $\mathcal{D}$ at the i^{th} sampling attempt. The packing itself is said to be random if it is generated by such a process. In practice, we say that a packing is random if its spectral properties are similar to those of a typical sampling generated by uniform sampling.

Quality Metrics: The quality of disk packings is typically measured via the shapes of the triangles in the corresponding Delaunay triangulation. Conflict-free and maximality are often quantified using distance and angle metrics, such as minimum edge length, the Hausdorff distance, the RMS distance, and the minimum and maximum angles in the Delaunay triangulation constructed around the point set P. Randomness can be quantified through the angle distribution, but is typically presented in the frequency domain. A random (yet not maximal or conflict-free) Monte Carlo sampling, for example, is represented by a white-noise spectrum. On the other hand, a random, maximal and conflict-free, Maximal Poisson-disk Sampling (MPS) is represented by a truncated blue-noise spectrum.

Density Tuning Versus Changing the Sizing Function: The sizing function determines the radius of some disk given its center. The standard approach to modify the density of a given packing is to modify its sizing function through refinement or coarsening. However, the problem of disk density tuning is different. Tuning does not change the underlying sizing function, but rather moves, places, or removes points (disks) in order to change the discrete disk density. A packing with a tuned density will strictly satisfy conflict-free and maximality, and maintain as much randomness as possible. Note that coarsening a mesh for numerical simulations usually increases the discretization error, while tuning does not.

Theoretical Density Limits: The tuning approach we present in this paper is inspired by the prior observation that maximal packings are not maximum, and there is in fact a wide range of area densities that are achievable by a maximal packing. For illustration, Figure 2 shows the extreme densities of a maximal packing: (a) densest, density 0.9 and (b) sparsest, density 0.3. In both extremes, point locations are pinned. In the densest packing, moving a point creates a conflict (but does not spoil maximality). In the sparsest, moving a point uncovers part of the domain (but does not spoil conflict-free). Table 1 summarizes the average point density, corresponding area fraction ratio, and Delaunay edge length of different maximal samples with uniform radius r, where: $\triangle(r)$ is the point set 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 $\sqrt{2}r$), $\hexagon(r)$ is the hexagonal lattice with side length r (diagonal length $2r$), MPS is Maximal Poisson-disk Sampling, and DR is Delaunay refinement (Delaunay circumcenter insertion). The case of $\triangle(r)$ corresponds to the densest sampling respecting the conflict-free condition, while $\triangle(\sqrt{3}r)$ represents the least dense sampling that still respects the maximality condition; see Figure 2. The aim of tuning an initial point set is to either move and place points to obtain a density closer to that of $\triangle(r)$, or move and remove points to obtain a density closer to that of $\triangle(\sqrt{3}r)$. The MPS process produces a maximal packing with density in the middle of this range, about 0.55. For packings with an intermediate density, such as MPS, many points are not pinned and their location may be changed without spoiling either condition. Besides movement, tuning includes adding and removing points, all without violating the maximality and disk-free constraints. It may come as a surprise that it is also possible to tune the density without destroying the third property, randomness. We shall see that only at the extremes, as the density approaches 0.3 or 0.9, randomness degrades significantly.

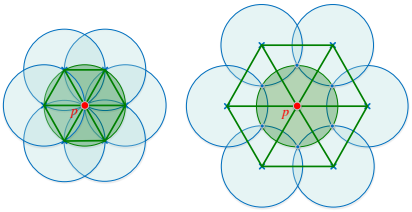


Figure 2: Two extreme maximal packings, both equilateral triangle tilings. In the densest case (left), p is pinned by the conflict-free condition. Moving p creates a conflict, but it is possible to restore conflict-free and keep coverage by ejecting it and moving its neighbors to cover its void. In the sparsest case (right), p is pinned by the maximality condition. Moving p uncovers part of the domain, but it is possible to restore coverage keeping conflict-free by injecting a point or more in p's unique void.

Sample type	Area fraction	Delaunay edge lengths
$\triangle(r)$	0.93	$\{r\}$
$\square(r)$	0.81	$\{r, \sqrt{2}r\}$
$\hexagon(r)$	0.62	$\{r, \sqrt{3}r, 2r\}$
DR(r)	0.60	$[r, 2r)$
MPS(r)	0.55	$[r, 2r)$
$\square(\sqrt{2}r)$	0.40	$\{\sqrt{2}r, r\}$
$\triangle(\sqrt{3}r)$	0.31	$\{\sqrt{3}r\}$

Table 1: Area fraction ratio and Delaunay edge lengths of different maximal samples with uniform radius r.

1.1. Related Work

In many computer graphics applications, the three packing properties are required. Conflict-free provides sampling efficiency, as nearby points are often redundant, maximality ensures full coverage of the graphical domain, and randomness avoids visual artifacts including aliasing [PH04]. Furthermore, in many sampling and integration applications, blue noise sampling [JZW*15] is desired as randomness avoids bias [SK13]. Historically, many random packing algorithms have sacrificed maximality in order to be simple or efficient. Today, true random maximal Poisson-disk sampling is simple and efficient [EMP*12]. In addition, an MPS provides well-shaped Voronoi [EM12], Delaunay [EMD*11], and tetrahe-

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.

dral [GYC*16] meshes for both uniform and nonuniform sizing functions [MREB12]. Several improved versions of MPS have been introduced targeting more efficient [GYJZ15, Yuk15] and parallel [IYLV13] implementations. However, one undesirable property of MPS is that many disks are at distance nearly r from one another, spoiling it from being pure truncated blue-noise. Heck et al. [HSD13] have recently shown that it is possible to change a random point set to tune its spectra to different profiles, including better blue noise. We focus here on the property that MPS produces a particular *area fraction density*, typically 0.55, and we can tune that distribution to attain a different density while maintaining all other properties. A key feature of our proposed algorithm is its powerful capability to inject, eject, or relocate points without sacrificing the sizing function or creating conflicts. Classical geometric methods either inject, eject, or relocate points to achieve a certain objective. For example, different versions of Delaunay refinement (DR) [She98, Si08] aim at creating quality Delaunay meshes by inserting points at the circumcenters of poor-quality triangles, increasing the density of the final triangulation. However, DR's quality-based strategy does not directly respect a user-prescribed sizing function, and indirectly follows the local feature size. The DR primitive can only increase an initial density, but not decrease it. On the other hand, Centroidal Voronoi Tessellation (CVT) [DFG99, DGJ10] relocates a point to enforce the seed of a Voronoi cell to be at its center of mass. This approach might use some sizing function information, but does not incorporate injection or ejection capabilities. It is therefore more constrained for density tuning purposes. Similarly, mesh simplification [CMS98] eliminates mesh points resulting from oversampled data. This can only work for reducing the density of a mesh, but would not work when injecting points is needed.

Contribution Summary. We introduce a novel framework for tuning the discrete density of a maximal disk packing. Using planar and curved surface models, we demonstrate the tuning capability towards different user-desired objectives, e.g., non-obtuse triangulation and accurate fiber simulations.

2. Injection of Uniform Disks

In this section, we illustrate the three successively more-aggressive phases for injecting points. We start with a maximal disk packing. Common to all phases, we iterate over randomly selected sample points. For the selected sample point, we remove or move the associated disk or its neighbors to create a void or space in the domain that is not covered by any of the remaining disks. If the void is large enough, we inject one or more new disks in order to increase density. If not, we search for another candidate void for injection.

Phase I. Void Diameter Injection. In the first phase, we uniformly randomly select a disk with center p, remove it to create a void, and identify the intersection corners of the void. Then, we iteratively try to cover that void with as many disks as possible, in order to increase the disk density. We first find the pair of void corners that are farthest apart, the diameter pair (a,b), and compute the Euclidean distance between them; $\mathbf{d}(a,b)$. We create a new disk with a center repositioned at either a or b, randomly selected. If $\mathbf{d}(a,b) < r$, then the entire void is covered, and no other point/disk can be injected. In this case, one disk was replaced with one disk, so the disk density did not change. Otherwise, if $\mathbf{d}(a,b) > r$, a part of the original void is still uncovered, leaving room for more points to be placed. We repeat the previous steps for the new partial void: we find the new longest diameter pair (a,b) of the partial void, and inject a new disk centered again at one of them, randomly selected. We stop when the original void of p is completely covered with the new disks. An illustration of these steps is shown in Figure 3, where disk p was replaced with 3 disks. Each of the new disks is centered at one end of the longest diameter of p's void or partial void.

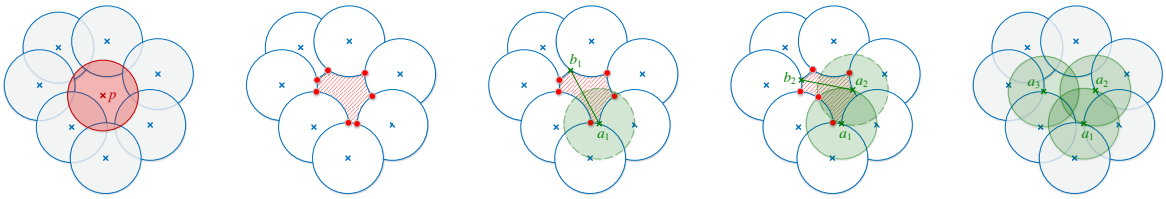
Phase II. Neighbor Repeller Injection. In the second phase, Repeller Injection, we consider a local neighborhood of a randomly selected disk with center p. The goal here is to create coverable gaps by moving the neighbors of p away from it. To do that, we first identify all the neighbors of p. Leaving p in place, we iterate over its neighbors in a random order. For a neighbor q, we create its unique void, and move q to its void corner farthest from p to possibly create a gap. If this leaves an uncovered gap we inject a point at a location randomly selected within this gap, just as in Void Injection, and declare success. Otherwise, if no gap is created after q is moved, we move the next neighbor of p. If all neighbors have been moved and no point has been injected, we roll back to phase I and apply Void Injection on p, moving p to one of its diameter void corners, and injecting new points, if possible.

Phase III. Crystal Growth Injection. In the third phase, Crystal Injection, we grow a set of void corner points that are attractors that pull nearby disk centers towards them. We freeze disks as their centers attach to attractor points. This is analogous to crystal growth. We start with two disks with centers at distance r apart; the disk positions are frozen and their intersection points are the first attractor points. We iterate over the mobile (not frozen) disks; see Figure 4 for an illustration. If a mobile disk does not cover any attractor point, then we move it as in Void Injection. For a mobile disk $D(p)$ covering attractor point a, we seek to move its disk center p to a. However, a might be strictly inside some other disks, leading to a conflict, so we first delete any such disks. If moving and/or deleting disks leaves an uncovered void, we immediately inject points to recover maximality.

2.1. Transition Between Phases

The three aforementioned phases have increasing injection capabilities. However, they are complementary and are all needed to tune to a high density close to that of $\triangle(r)$. The order of the phases is a trade-off between their tuning capabilities, complexity, and maintaining maximality, conflict-avoidance, and randomness. Void Diameter Injection is the least complex as it involves only injections and no movements, and can therefore be used for fast fine density tuning. Based on their tuning saturation levels, our injection framework transitions between its different phases according to the values in Table 2. Specifically, the injection framework starts in its first phase, and once it hits its saturation area fraction value (0.75), the framework makes a transition to the second phase. Once its saturation area fraction value (0.79) is hit, a transition is made into the third injection phase. The same idea applies to transitions between ejection phases when the saturation area fraction values of the first and second phases are reached, as further explained in the next section.

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.



(a) A disk with center p is selected (randomly) as the candidate region for injection.

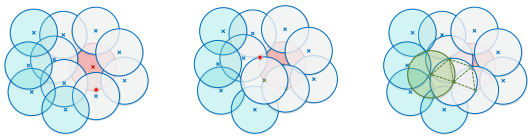
(b) Disk p is removed, temporarily spoiling coverage. Void corners identified.

(c) A new disk is injected at either of the two corners that are farthest apart (a_1, b_1).

(d) Update the void. If a partial void remains uncovered, repeat step (c).

(e) Stop when the void of p is fully covered again. Here, 3 new disks were injected.

Figure 3: Void diameter injection of a uniform packing: a disk is removed and replaced with as many disks as possible. Injection stops when added disks cover the void of the removed disk completely.



(a) Initial packing. (b) Crystals forming. (c) Crystal injection.

Figure 4: Crystal growth injection of a uniform packing. Disks move to form "crystals" of equilateral triangles. Once a void is created, a new disk in inserted there.

Method	Phase	Area Fraction
Injection	Crystal Growth	0.89
	Neighbor Repeller	0.79
	Void Diameter	0.75
Start	Simple MPS Packing	0.55
Ejection	Void Diameter (Sifted Disks)	0.41
	Neighbor Attractor	0.35
	Crystal Growth	0.31

Table 2: Area fractions achieved by different injection and ejection phases and maximal Poisson-disk packing.

3. Ejection of Uniform Radii Disks

We illustrate the three successively aggressive phases of the ejection process. Again, start with a maximal disk packing. Our goal is to remove disks until the required density is reached, constantly maintaining the conflict-free and coverage properties. Similar to the injection case, we eject disks using a less-aggressive strategy until the density achieves a threshold near the best density it typically achieves (Table 2). Termination occurs as soon as the user-desired density is achieved, or when no more points can be relocated.

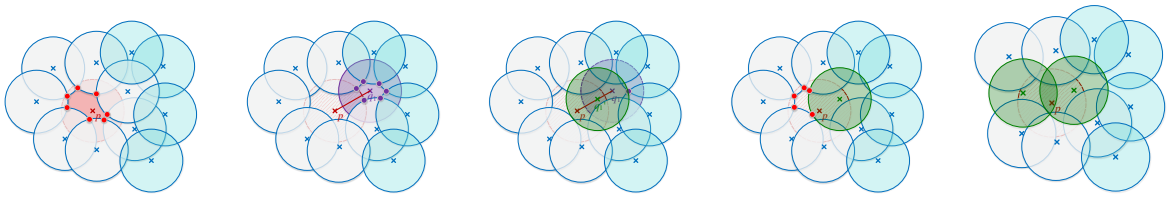
Phase I. Void Diameter Ejection. In the first ejection phase, we pick two (or more) neighboring disks a, b as candidates for removal. We find their combined void corners. We find the subregion of the void where a disk center covers all the corners, and randomly select the new center p from it. This is the same as the Sifted Disks [EMA*13] operation.

Phase II. Neighbor Attractor Ejection. The second ejection phase is a modification of Repeller Injection but moving disks towards the candidate instead of away; see Figure 5. We start with a randomly selected disk with center p, and consider each of its neighboring disk centers q in sequence. We move q to q', the point on segment $\overline{pq}$ as close as possible to p while still covering q's void, and keeping it outside all other disks except perhaps p's. The next paragraph explains how to calculate q'. Moving q will not uncover some part of the domain, and will likely cover some of p's void. Remove p as soon as its remaining void is covered. Otherwise, we attempt to at least make some geometric progress by finding a new position for p. Move p to the center of the minimum-radius disk that covers its remaining-void corners. If this position is inside some disk (which can happen because the void is not convex), then project it to the remaining void. In rare cases there may be no new position for p that covers its remaining void; in that case we put p and its neighbors back in their original positions.

To find q', we compute q's void corners. For each corner we find the interval of $\overline{pq}$ within r of it. We intersect all these intervals. For each neighboring disk, we subtract the segment of $\overline{pq}$ it covers. The endpoint of the remaining interval closest to p is q'.

Phase III. Crystal Growth Ejection. In the third phase, we find two disks with center distance $\sqrt{3}r$, and move other points to also be at distance $\sqrt{3}r$ from them. To accomplish this, we may re-use our prior Attractor Ejection algorithm if we enlarge these disks to have a dilated radius of $\sqrt{3}r$. (In the final output these disks still have a radius of only r.) The two intersection points of the big $\sqrt{3}r$ disks are attractor points. Placing a disk center at an attractor point would create equilateral triangles of the maximum possible edge length, while still having the center of the triangle covered by r-disks. We pick any other sample q at random. If its disk does not cover an attractor point, we simply move it as in Attractor Ejection. Otherwise, we move q to the (closest) attractor point that its disk covers. If this creates a conflict, then we remove the other conflicted disks (not q) and resample any void to regain maximality, then update the attractor points.

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.



(a) A disk with center p is selected (randomly). Identify its unique void/corners.

(b) Pick a neighboring disk center q_1, and find the corners of its void.

(c) Move q_1 along segment $\overline{pq_1}$ as far as possible without uncovering q_1's unique void.

(d) If any void remains, find a neighbor q_2 at a different corner, and move it as in (c).

(e) Repeat (c)–(d) until p's whole void is covered and we can eject p.

Figure 5: Neighbor attractor ejection of a uniform packing: ejection succeeds if attracted neighbors manage to cover the void of p completely. Otherwise, if all neighbors have moved and p's void is still uncovered, p is moved back to the center of its remaining void.

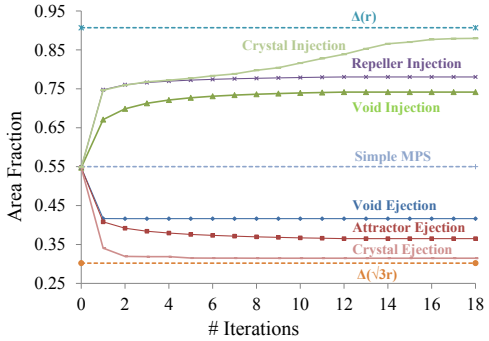
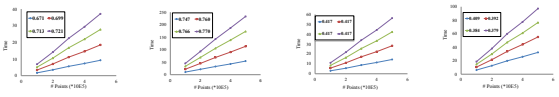


Figure 6: Convergence of different tuning phases.

4. Performance Measures

Convergence and Runtime. We study the density tuning progress if one phase is continued ad infinitum, in terms of area fraction achieved as the number of attempts to inject/eject a point increases. Starting at the simple MPS reference point (0.55), different phases have rapid area fraction variation initially, and increase/decrease until saturated at the values listed earlier in Table 2. Injection (increasing area fraction) is upper-limited by the densest hex packing $\triangle(r)$, while ejection (decreasing area fraction) is lower-limited by the sparsest hex packing $\triangle(\sqrt{3}r)$; where $\triangle(r)$ and $\triangle(\sqrt{3}r)$ were defined in Table 1. Each phase individually can achieve better area fraction performance, whether in injection or ejection, when compared to simple MPS. However, the combined three phases of injection and ejection approach the densest/sparsest packing limits. See Figure 6 and Figure 7 for convergence and runtime comparisons.



(a) Injection I. (b) Injection II. (c) Ejection I. (d) Ejection II.

Figure 7: Linear runtime performance of different tuning phases.

Spectral Analysis. We now take our analysis to the frequency domain, which is suitable for studying the tuning impact on the "randomness" and blue noise properties of the tuned meshes. Although MPS does not have perfect blue noise behavior (inter-sample distances), it gets very close and hence we use it as our test starting point. We apply our tuning (injection and ejection) steps to a 2D planar *unit box* with periodic boundary conditions, and characterize the resulting meshes at the saturation points of each phase. Experimental results are shown in Figure 8. The middle row represents the performance of simple MPS packing, with density 0.55, which is the starting point of the tuning framework. Going up one row at a time, we see the performance of different injection phases levels: 0.75, 0.79, 0.84 and 0.89, respectively. Similarly, going down one row at a time shows the performance of different ejection phases levels 0.41, 0.35, 0.33 and 0.31, respectively. In all rows, the first column is an arrangement of the point sets, visually showing randomness up to the saturation level of either injection or ejection where structured artifacts show up, indicating loss of randomness, upon approaching the theoretical limits. The second and third columns show a 2D Fourier transform and a sliced cross section versus frequency variation. The MPS Fourier transform shows clear circular behavior which indicates that no direction is favored over the other in the frequency domain. This circular behavior is similarly observed in the first two stages on injection and ejection. Directionality is introduced as the third phase of either injection or ejection is approached, indicating gradual loss of randomness. The mixed circular-dotted Fourier behavior at densities 0.84 and 0.33 turns into almost an all-dotted one in the saturation cases of injection and ejection at densities 0.89 and 0.31. The fourth column presents a measure of anisotropy, which is the property of being directionally dependent. Despite noisy, the MPS anisotropy metric is close to flat along frequency. The same observation is valid for the first two phases of both injection and ejection. High amplitudes and variabilities against frequency are introduced in the third injection and ejection phases as they approach their practical saturation limits, close to the theoretical limits, indicating directionality (and hence: structure) is introduced.

Angle and Edge Lengths Distributions. Using the same unit box experiment deployed for spectral analysis, the fifth and sixth

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.

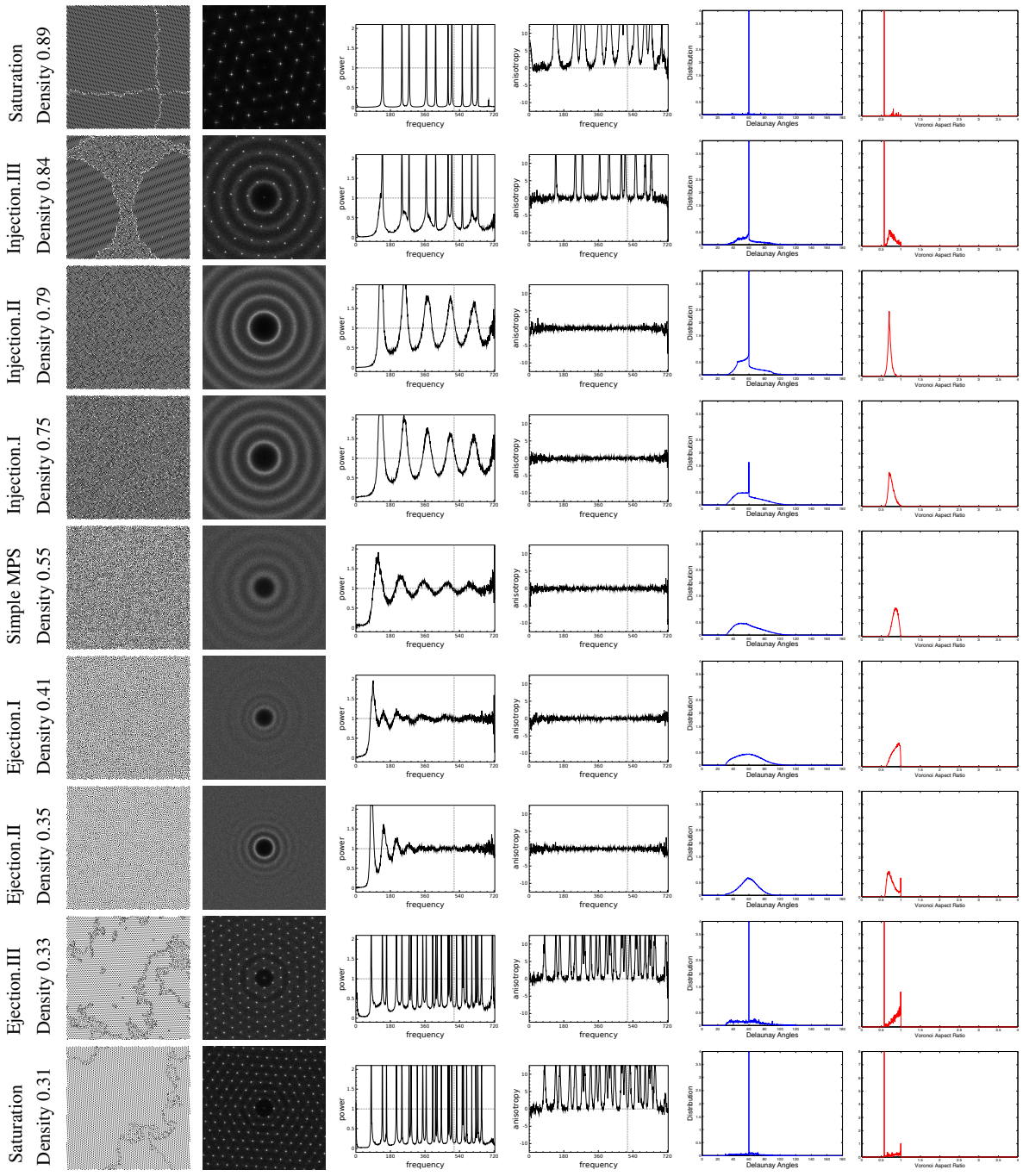


Figure 8: Spectral analysis comparisons of the Simple MPS case, and the different phases of injection and ejection, when applied to a unit box with periodic boundary conditions. From left to right: the sample point set, 2D Fourier spectrum, power cross section, anisotropy, Delaunay angle distribution, and Voronoi aspect ratio.

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.

columns of Figure 8 examine the distributions of Delaunay angles and Voronoi aspect ratios, respectively, in the tuned meshes. Delaunay angles are well distributed between 30° and 120° in the simple MPS case. As we deviate away from it, in both injection and ejection cases, angles start to pile up, which shows up in the corresponding histograms in the form of peaks. As we approach the practical limits, more equilateral triangles get formed. Therefore, large peaks at 60° are introduced at the practical limits of the last phases of injection and ejection.

5. Tuning Non-uniform Disks

We extend our methods to disks with radii that vary across the domain (non-uniform). The average number of neighbors per disk tends to be larger in the case of non-uniform disks, compared to the uniform case [MREB12]. This has a direct impact on our algorithm. Intuitively, non-uniform disks can be weighted according to their varying radii, and the relative distances between centers need to take into consideration the corresponding weights of their disks when finding void corners. Therefore, we consider power diagrams and power vertices for the non-uniform analysis, instead of Voronoi diagrams and Voronoi vertices [EMA*13] as in uniform packing. In specific, we define the *power distance* $\mathbf{w}(a,b)$ from a to b to be $\mathbf{d}^2(a,b) - r^2(a)$, as in power diagrams or additively weighted Voronoi diagrams. This distance depends on the radius associated with the starting measurement point and is therefore asymmetric; $\mathbf{w}(a,b) \neq \mathbf{w}(b,a)$. A key choice is in the definition of conflict. We choose a variant related to the *prior-disk* and *smaller-disk* criteria [MREB12]. In particular, when we move or inject a disk we consider it to be the latest arrival, and allow its center to be placed anywhere not already covered by the other disks. That is, we accept any disk center if the weighted distances of the other centers to it are all positive. It is possible for the new disk to be large enough that it covers the center of some other (prior) disk. This changes the arrival order of the disks, but, as in *smaller-disk*, it ensures that no pair of disks cover each others' centers. Our solutions for non-uniform injection and ejection have two phases, based on the first and second phases of our uniform injection/ejection procedures. There are no Crystal Injection or Crystal Ejection phases here.

Non-Uniform Disk Injection. In the first phase of injection, we consider removing one disk and replacing it by two (or more). However, the disk radius is potentially different at each void corner. For each corner c, its *weighted diameter* is its maximum weighted distance to another corner, $\max_i \mathbf{w}(c,c_i)$. We remove p, and replace it with a disk at the corner with the maximum weighted diameter. If the diameter is negative, then this disk covers all other corners and injection failed. Otherwise, there are some uncovered corners, and we recursively compute the new void(s) and inject the new corner with the maximum weighted diameter. We stop when maximality is recovered. In the second phase, we consider moving the neighbors q of disk with center p, in sequence. We move q to its void corner c with maximum $\mathbf{w}(c,p)$. If this leaves a void, then we recursively insert a new disk at c' with maximum $\mathbf{w}(c',p)$.

Non-Uniform Disk Ejection. For the first phase, we proceed with Void Diameter ejection. For the second phase (Attractor Ejection), we start by assuming that the disk radius at q is invariant, and estimate q's new position q' exactly as before. The problem is that its new radius might be too small to cover its original void. This would destroy maximality and require *adding* points. So, if that happens, we do a simple heuristic search for an acceptable position. We try $q'' = q + 0.9\vec{qq'}$. We try this scaling up to three times total, stopping if the original void is covered. While this works reasonably well in practice, many other strategies for searching for good positions are possible, such as moving q' farther if it covers the void by a wide margin, or moving q' to the corner closest to p.

6. Applications

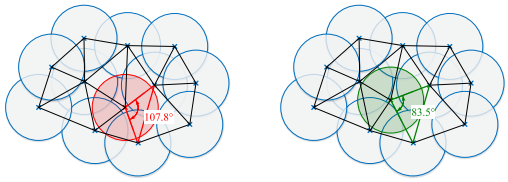
6.1. Non-Obtuse Triangulation

A non-obtuse triangle mesh is composed of a set of triangles in which every angle is less than or equal to 90°. These triangles are called *non-obtuse triangles* and are generally considered more desirable than triangles with obtuse angles; non-obtuseness is a classical measure of mesh quality. Several methods attempt this problem without respecting the sizing function or the noise properties of the mesh [EÜ07, LZ06]. Here we apply the tools we developed in the previous sections to successfully modify an existing Delaunay mesh constructed over a well-spaced point set (e.g., generated by MPS), and includes obtuse triangles. We tune the mesh vertices using relocation, injection, and ejection to yield a non-obtuse triangulation. Intuitively, we pursue the following strategy. Consider an obtuse triangle in a triangulation. The circumcenter of an obtuse triangle lies outside it. The circumcenter of this circumcircle, v, is a Voronoi vertex. Thus, an obtuse triangle indicates that the region near v has no nearby sample points. Moving sample points even farther away from v, using v as a repeller in Repeller Injection, is likely to allow the injection of a new sample point at v or nearby. We can then locally re-triangulate using this new sample point and eliminate the non-obtuse triangle. In Section 2 and Section 3, we chose the point to move uniformly at random. In this application, we instead target vertices associated with obtuse triangles, apply the following steps in order, and re-trangulate:

1. Relocation: Moving the point associated with the obtuse angle outside its void edge-circles (circles centered at edge's midpoint with diameter equals to edge length), while maintaining maximality and conflict-free conditions.
2. Ejection: If relocation fails at a point, we eject that point and move its neighbors to reclaim coverage as described in Section 3. If achieved, we try to further relocate them for maximality and to eliminate conflicts, e.g., moving each point toward the furthest gap corner without re-creating an obtuse angle.
3. Injection: if both relocation and ejection fail, we remove that point and apply our injection algorithm as described in Section 2. In specific, we insert a point in the circumcenter of the obtuse triangle and relocate points on its unique void boundary while relocating its neighbors to maintain maximality and conflict-free conditions.

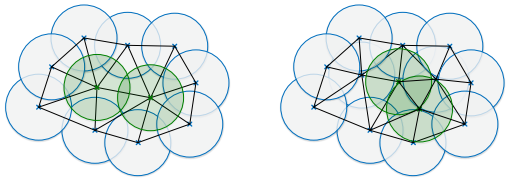
This three-step algorithm guarantees visiting all obtuse angles in the mesh. An example obtuse angle in a Delaunay triangulation is shown in Figure 9, along with how either one of relocation, ejection, and injection can successfully eliminate it. However, all three steps are necessary in our algorithm to cover every possible obtuse

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.



(a) A Delaunay triangulation with a problematic obtuse angle.

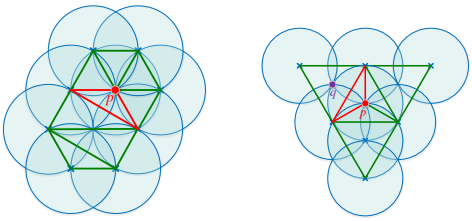
(b) Relocating the point at the obtuse angle.



(c) Ejecting a point at obtuse angle reclaimed maximality.

(d) Injecting 2 disks at obtuse angle point.

Figure 9: The elimination of obtuse angles in a triangulation, achieved by either relocation, ejection, or injection.



(a) Injection needed at circumcenter of obtuse triangle. Ejection creates new obtuse angle.

(b) Ejection is needed because p is trivalent. Injecting q would create new obtuse angles.

Figure 10: Both injection and ejection are needed to achieve a non-obtuse triangulation: (a) a case when only injection works, (b) a case when only ejection works. In both cases, relocation would not work.

angle. The need for relocation and ejection might be obvious, but there are cases where only injection would work for getting rid of an obtuse angle; see Figure 10 for an example where only one of injection and ejection succeeds. Note that the extension of our algorithm from 2D to curved surfaces is straightforward: a local "void" is restricted to the input triangulation.

6.1.1. Experimental Examples

Our tuning framework successfully eliminates all obtuse angles in uniform and non-uniform 2D domains, as well as curved surfaces. To illustrate the capability of the non-obtuse tuning algorithm, we apply it to a few meshes of some standard domains in different cases. We plot the angle histogram before and after the tuning process to highlight the elimination of the angle histogram tail beyond 90°. Figure 11 shows the successful elimination of obtuse triangles in non-convex 2D domains including some sharp corners (cat, wedge, bat), for the case of a uniform sizing function (equal-radius packing), and a dolphin-shaped domain for the case of non-uniform (varying radii) packing. In Figure 12, we show the successful elimination of obtuse triangles when our tuning algorithm is applied to a set of curved surfaces (bimba, Ramesses, elk, and fertility), for the case of a uniform sizing function (equal-radius packing). For comparison with other non-obtuse triangulation methods, Figure 13 illustrates the capability of our algorithm in comparison with the gap processing algorithm of Yan and Wonka [YW13]. Yan and Wonka's algorithm did not eliminate all obtuse angles; it rather reduced the maximum angle in a uniform curved bunny-shaped mesh from $\sim 116^\circ$ to $\sim 110^\circ$. In contrast, our tuning framework eliminated all obtuse angles, making the maximum angle equal to 90°. Starting with an MPS mesh with non-uniform sizing functions, Figure 1 shows the successful elimination of obtuse triangles when our tuning algorithm is applied to curved surfaces (elephant and homer), for the case of a non-uniform sizing function (varying-radius packing). The resulting tuned meshes are all Delaunay meshes with no angles greater than 90°. Their vertices form a point set that maintains maximality, conflict-free, and randomness properties. All tuned triangulations are less dense than their initial versions, indicating most obtuse angles are resolved via ejection. Table 3 presents numerical comparisons between initial and tuned triangulations of some of the uniform and non-uniform models listed above, using standard mesh quality metrics including the number of vertices v, the minimum and maximum angles ($\theta_{\min}$ and $\theta_{\max}$), the minimum edge length as a fraction of the maximum edge length ($L_{\min}$), the Hausdorff distance (d_H); the maximum distance from a point set (initial/tuned) to the nearest point in a reference point set, and the root mean square distance d_{RMS} between a point set (initial/tuned) to the nearest point in a reference point set, as a % of the diagonal length of the bounding box.

6.2. Modeling Fiber Materials

In this application, we use tuning to improve the fidelity of modeling the micro-structure of a unidirectional E-glass fiber reinforced epoxy material. The composite consists of a random arrangement of fibers embedded in a matrix material. The fibers have circular cross sections of about the same diameter. For this unidirectional material, the fibers are aligned in roughly the same direction. Figure 14a shows a cross-sectional image of the actual material, showing that they resemble a packing of non-overlapping disks. Taking any maximal packing of overlapping disks and halving the radii of its disks results in a maximal packing of non-overlapping disks; see Figure 14b. To accurately model the stress-strain and failure of the bulk fiber material, a model must match the fiber in terms of the density of the fiber—the fraction of the domain area covered by fiber disks—and the randomness of fiber disks. Both factors are critical for fidelity. The density is critical because the fiber is much stronger than the epoxy, so the material strength is proportional to the cross-sectional area of the fibers, to first order. The randomness provides a second level of accuracy, and is especially critical for the failure strength. Many have tried to characterize material strength and stress response using deterministic structures, such as hexagonal or square lattices [LBM00, BAC05, Mal08].

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.

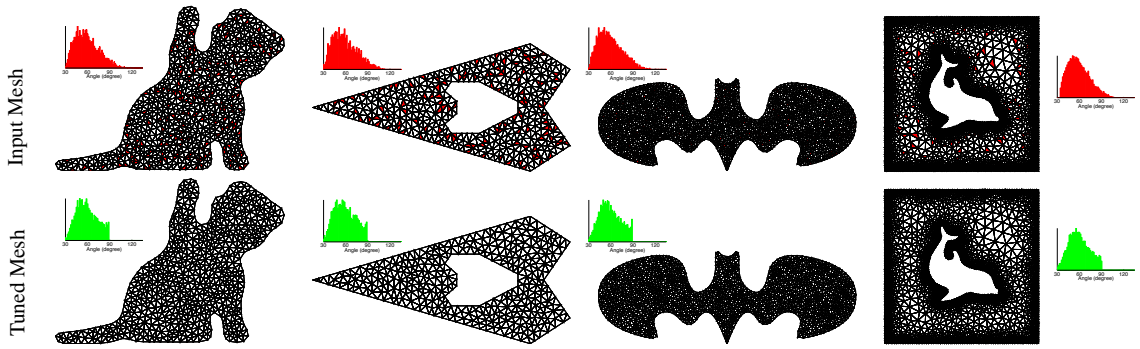


Figure 11: Initial (top) and tuned (down) uniform meshes of cat-, wedge-, and bat-shaped domains, along with nonuniform meshes of a dolphin-shaped domain. Meshes and corresponding histograms clearly show the elimination of obtuse (red) triangles.

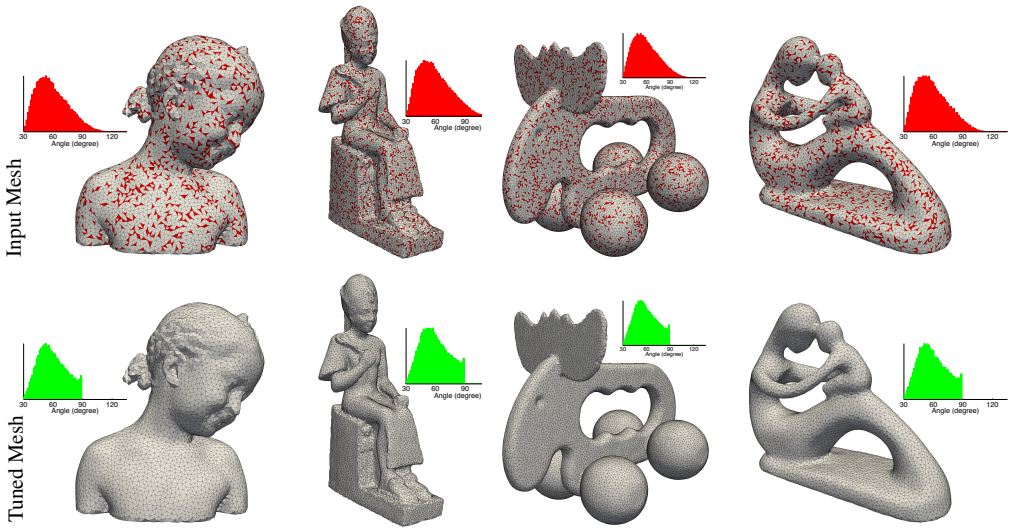
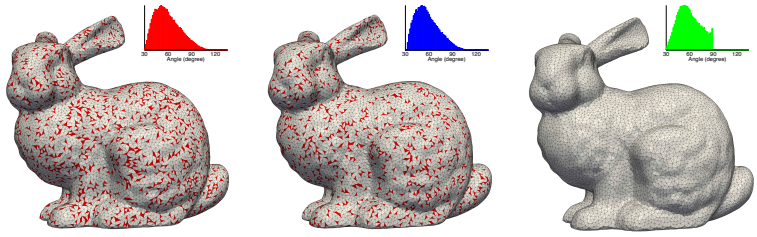


Figure 12: Initial (up) and tuned (down) uniform meshes of curved surface models (bimba, Ramesses, and Moai). Meshes and corresponding histograms clearly show the elimination of obtuse (red) triangles.



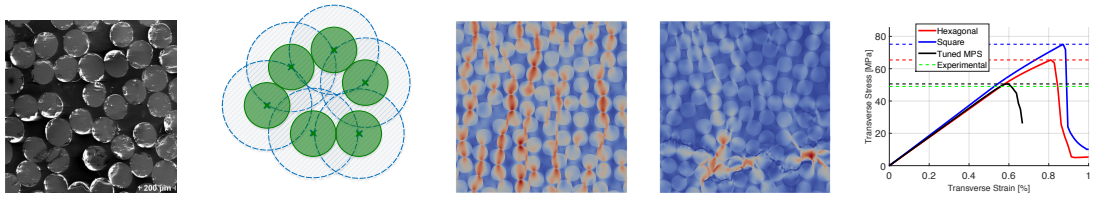
(a) Initial. (b) Gap-processed. (c) Tuned (ours).

Figure 13: A bunny model: (a) initial MPS packing surface, $\theta_{max} = 116.17^\circ$; (b) mesh generated using the gap processing algorithm in [YW13], max angle only reduced to $\theta_{max} = 109.61^\circ$; and (c) tuned non-obtuse triangulation, $\theta_{max} = 90^\circ$.

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.

Mesh	Model	v			$\theta_{\min}$		$\theta_{\max}$		$L_{\min}$		d_H		d_{RMS}	
		Initial	Tuned	%change	Initial	Tuned	Initial	Tuned	Initial	Tuned	Initial	Tuned	Initial	Tuned
Uniform	Bunny	11.5K	10.5K	-8.7	30.56	30.06	116.18	90.00	0.501	0.501	0.5008	0.5695	0.0429	0.0449
	Loop	10.7K	9.9K	-7.48	30.14	30.05	117.41	90.00	0.502	0.500	0.4388	0.4388	0.0612	0.063
	Moai	12K	10.9K	-9.17	30.36	30.06	118.80	90.00	0.500	0.500	0.4718	0.5266	0.0736	0.0772
	Bimba	6.5K	5.9K	-9.23	30.03	30.16	114.80	89.99	0.501	0.500	0.6566	0.6566	0.0926	0.0972
	Ramesses	11.5K	10.5K	-8.7	30.16	30.05	117.39	90.00	0.500	0.500	0.518	0.518	0.0754	0.0789
	Fertility	8.4K	7.7K	-8.33	30.26	30.07	116.00	90.00	0.500	0.500	0.3704	0.3864	0.0503	0.0531
	Elk	20.4K	18.7K	-8.32	30.20	30.02	118.04	90.00	0.500	0.500	0.3311	0.328	0.0518	0.0541
Non-uniform	Bunny	6.6K	5.8K	-12.12	19.05	19.89	126.99	90.00	0.162	0.149	0.4091	0.4091	0.0636	0.0689
	Homer	4.4K	3.9K	-11.36	21.39	21.33	126.14	90.00	0.092	0.087	0.467	0.54	0.1058	0.1146
	Elephant	12.7K	11.2K	-11.81	21.92	18.35	123.43	90.00	0.053	0.05	0.5349	0.5789	0.0819	0.0906

Table 3: Mesh quality measures of initial and tuned non-obtuse triangulations: v is the number of vertices, $\theta_{\min}$ and $\theta_{\max}$ are the minimum and maximum angles, $L_{\min}$ is the minimum edge length (as fraction of maximum edge length), d_H is the Hausdorff distance, and d_{RMS} is the root mean square distance (as a % of the diagonal length of the bounding box).



(a) A cross-section of a fiber micrograph. (b) Nonoverlapping fiber disks model. (c) Tuned MPS model at peak load. (d) Tuned MPS model post fracture. (e) simulated tensile responses of different packings.

Figure 14: Modeling fiber structures: (a) Scanning electron micrograph of a fiber composite cross-section; (b) $r/2$ nonoverlapping fibers are modeled using MPS disks of radius r, representing a composite fiber material with an area fraction of 67%, tuned MPS performance at (c) peak load, and (d) post fracture, and (e) simulated tensile responses of tuned MPS, hexagonal, and square packings. Solid lines show the simulated stress under increasing strain, and horizontal dashed lines show the expected peak stress before failure. The green dashed line is the experimental peak stress. Our tuned MPS simulation matches the experimental one closely.

While these produce plausible results under elastic loading, they significantly over-predict strength and damage resistance. On the other hand, several random packings have been tried. While these tend to give a more plausible response curve *shape*, in absolute terms they are inaccurate because they do not get the fiber density right [WCS98, TGL08, SGP06, SG06, TTL06]. This is an active materials science challenge problem, and unfortunately, the data available from physical experiments are inadequate to validate any stress/strain or failure predictions. Physical experiments include complex phenomena not represented in the models, and the models include local quantities that are currently impossible to measure in experiments. The state-of-the-art in this field is to judge models by first principles, how accurately they represent the material. By this standard, our new predictions are preferred by subject matter experts. A clear solution for this modeling problem is a random packing with its density adjusted to the exact density of the fiber material. Recall that MPS and its near-maximal variants usually produce volume fractions around 55%. In fiber materials, the volume fraction is typically larger, in the range of 60–70%. These values are purely a result of the process used to create the packing; recall maximal packings can vary in area fraction by a factor of 3, from 0.3 to 0.9, *without changing the coverage and conflict radii*.

We use tuning of an MPS to increase its density (by injection) to the correct fiber density, while maintaining randomness. We compare our tuned MPS model to hexagonal and square packings. The extent of the fiber material is orders of magnitude larger than the fiber diameter, so the material is modeled well by a periodic arrangement of disks in a square. We start with an MPS over this square. We use disk injection to increase the area fraction to the area fraction of the material at hand, in our case 67%. We load the model transverse to the fibers, to its peak value before fracture and post fracture, as illustrated in Figure 14c and Figure 14d. We calculate the boundary conditions using a multi-scale approach, solving for the relative velocities at the periodic nodes, ensuring the homogenized response is uniaxial. We ran four examples, each with a different random MPS as input and injected disks as output. Figure 14e shows the four simulated responses, plus their average. Note that the simulations span a small range in the elastic range, but at fracture (peak load) there is a wider variation in the responses. Obviously, methods that do not capture the fiber noise properties fail to present a good peak stress model when compared to experimental results. Poisson-disk sampling, on the other hand, preserves the noise property but does not represent the correct discrete density and hence needs to be tuned using our method for a reliable physical modeling.

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.

7. Conclusions

We have introduced a method to tune the discrete density of a random disk packing to a user-specified target. In contrast to prior methods, we are able to get much closer to the densest-possible and sparsest-possible packings. Disks are added, moved, or removed one by one, giving very fine control. Blue noise is retained for much of the control range, but is lost as we approach a structured tiling at the extremes of the allowed densities. We have demonstrated the efficiency of our method using uniform sizing functions over planar domains as well as the curved domains typical of graphics models. We also show the usefulness of our method for modeling: we can match the density of physical materials to generate more realistic fracture simulations. For future work, we consider methods to reintroduce randomness. Injected disks are exactly one disk radius away from neighbors, while ejected disks tend to be equidistant to several nearby disks. Both of these can be updated. We will also explore the limits of how fast the distribution may be graded.

References

[BAC05] BARBERO E. J., ABDELAL G. F., CACERES A.: A micromechanics approach for damage modeling of polymer matrix composites. *Composite Structures 67*, 4 (2005), 427–436. doi:10.1016/j.compstruct.2004.02.001. 8

[CMS98] CIGNONI P., MONTANI C., SCOPIGNO R.: A comparison of mesh simplification algorithms. *Computers & Graphics 22*, 1 (Feb. 1998), 37–54. doi:10.1016/S0097-8493(97)00082-4. 3

[DFG99] DU Q., FABER V., GUNZBURGER M.: Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review 41*, 4 (1999), 637–676. doi:10.1137/S0036144599352836. 3

[DGJ10] DU Q., GUNZBURGER M., JU L.: Advances in studies and applications of centroidal Voronoi tessellations. *Numerical Mathematics: Theory, Methods and Applications 3*, 2 (Mar. 2010), 119–142. doi:10.4208/nmtma.2010.32s.1. 3

[EM12] EBEIDA M. S., MITCHELL S. A.: Uniform random Voronoi meshes. In *Int. Meshing Roundtable* (2012), pp. 258–275. doi:10.1007/978-3-642-24734-7_15. 2

[EMA*13] EBEIDA M. S., MAHMOUD A. H., AWAD M. A., MOHAMMED M. A., MITCHELL S. A., RAND A., OWENS J. D.: Sifted disks. *Computer Graphics Forum 32*, 2 (May 2013), 509–518. doi:10.1111/cgf.12071. 4, 7

[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. *Comput. Aided Des. 43*, 11 (2011), 1506–1515. doi:10.1016/j.cad.2011.08.012. 2

[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 (2012), 785–794. doi:10.1111/j.1467-8659.2012.03059.x. 2

[EÜ07] ERTEN H., ÜNGÖR A.: Computing acute and non-obtuse triangulations. In *Proceedings of the 19th Canadian Conference on Computational Geometry* (Aug. 2007), CCCG2007, pp. 205–208. 7

[GYC*16] GUO J., YAN D.-M., CHEN L., ZHANG X., DEUSSEN O., WONKA P.: Tetrahedral meshing via maximal Poisson-disk sampling. *Computer Aided Geometric Design 43* (Mar. 2016), 186–199. doi:10.1016/j.cagd.2016.02.004. 2

[GYJZ15] GUO J., YAN D.-M., JIA X., ZHANG X.: Efficient maximal Poisson-disk sampling and remeshing on surfaces. *Computers & Graphics 46* (Feb. 2015), 72–79. doi:10.1016/j.cag.2014.09.015. 3

[HSD13] HECK D., SCHLÖMER T., DEUSSEN O.: Blue noise sampling with controlled aliasing. *ACM Transactions on Graphics 32*, 3 (June 2013), 25:1–25:12. doi:10.1145/2487228.2487233. 3

[IYLV13] IP C. Y., YALÇIN M. A., LUEBKE D., VARSHNEY A.: PixelPie: Maximal Poisson-disk sampling with rasterization. In *Proceedings of the 5th High-Performance Graphics Conference* (July 2013), pp. 17–26. doi:10.1145/2492045.2492047. 3

[JZW*15] JIANG M., ZHOU Y., WANG R., SOUTHERN R., ZHANG J. J.: Blue noise sampling using an SPH-based method. *ACM Transactions on Graphics 34*, 6 (Nov. 2015), 211:1–211:11. doi:10.1145/2816795.2818102. 2

[LBM00] LANDIS C. M., BEYERLEIN I. J., MCMEEKING R. M.: Micromechanical simulation of the failure of fiber reinforced composites. *J Mechanics and Physics of Solids 48*, 3 (2000), 621–648. doi:10.1016/S0022-5096(99)00051-4. 8

[LZ06] LI J. Y. S., ZHANG H.: Nonobtuse remeshing and mesh decimation. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing* (June 2006), SGP '06, pp. 235–238. doi:10.2312/SGP/SGP06/235-238. 7

[Mal08] MALIGNO A. R.: *Finite element investigations on the microstructure of composite materials*. PhD thesis, University of Nottingham, 2008. 8

[MREB12] MITCHELL S. A., RAND A., EBEIDA M. S., BAJAJ C. L.: Variable radii Poisson-disk sampling. In *Proceedings of the 24th Canadian Conference on Computational Geometry* (Aug. 2012), CCCG '12, pp. 185–190. 3, 7

[PH04] PHARR M., HUMPHREYS G.: *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann, 2004. 2

[SG06] SWAMINATHAN S., GHOSH S.: Statistically equivalent representative volume elements for unidirectional composite microstructures: Part II—With interfacial debonding. *Journal of Composite Materials 40*, 7 (Apr. 2006), 605–621. doi:10.1177/0021998305055274. 10

[SGP06] SWAMINATHAN S., GHOSH S., PAGANO N. J.: Statistically equivalent representative volume elements for unidirectional composite microstructures: Part I—without damage. *Journal of Composite Materials 40*, 7 (Apr. 2006), 583–604. doi:10.1177/0021998305055273. 10

[She98] SHEWCHUK J. R.: Tetrahedral mesh generation by Delaunay refinement. In *Proceedings of the Fourteenth Annual Symposium on Computational Geometry* (1998), SCG '98, ACM, pp. 86–95. doi:10.1145/276884.276894. 3

[Si08] SI H.: Adaptive tetrahedral mesh generation by constrained Delaunay refinement. *International Journal for Numerical Methods in Engineering 75*, 7 (2008), 856–880. doi:10.1002/nme.2318. 3

[SK13] SUBR K., KAUTZ J.: Fourier analysis of stochastic sampling strategies for assessing bias and variance in integration. *ACM Transactions on Graphics 32*, 4 (July 2013), 128:1–128:11. doi:10.1145/2461912.2462013. 2

[TGL08] TOTRY E., GONZÁLEZ C., LLORCA J.: Failure locus of fiber-reinforced composites under transverse compression and out-of-plane shear. *Composites Science and Technology 68*, 3–4 (Mar. 2008), 829–839. doi:10.1016/j.compscitech.2007.08.023. 10

[TTL06] TAY T.-E., TAN V. B. C., LIU G.: A new integrated micro–macro approach to damage and fracture of composites. *Materials Science and Engineering B 132*, 1 (July 2006), 138–142. doi:10.1016/j.mseb.2006.02.023. 10

[WCS98] WERWER M., CORNEC A., SCHWALBE K.-H.: Local strain fields and global plastic response of continuous fiber reinforced metal-matrix composites under transverse loading. *Computational Materials Science 12*, 2 (1998), 124–136. doi:10.1016/S0927-0256(98)00037-8. 10

[Yuk15] YUKSEL C.: Sample elimination for generating Poisson disk sample sets. *Computer Graphics Forum 34*, 2 (2015), 25–32. doi:10.1111/cgf12538. 3

[YW13] YAN D.-M., WONKA P.: Gap processing for adaptive maximal Poisson-disk sampling. *ACM Transactions on Graphics 32*, 5 (Sept. 2013), 148:1–148:15. doi:10.1145/2516971.2516973. 8, 9

© 2016 The Author(s)
Computer Graphics Forum © 2016 The Eurographics Association and John Wiley & Sons Ltd.