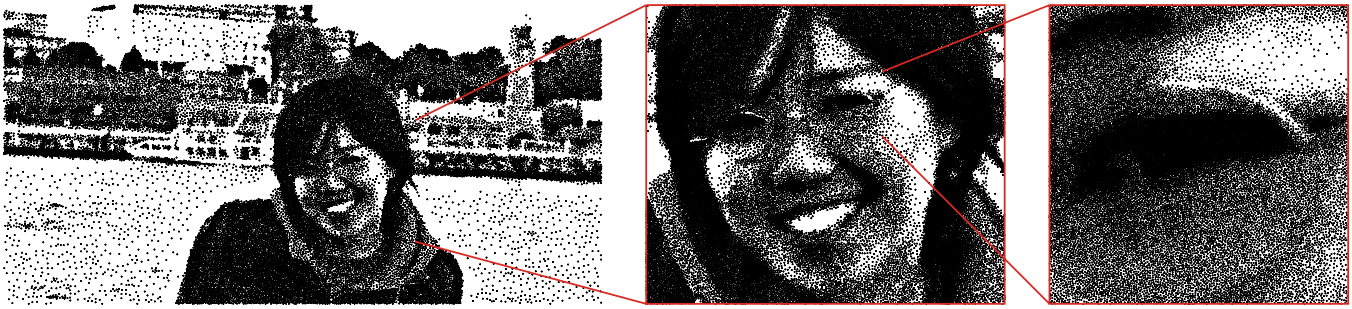
Recursive Wang Tiles for Real-Time Blue Noise

Johannes Kopf
University of Konstanz

Daniel Cohen-Or
Tel Aviv University

Oliver Deussen
University of Konstanz

Dani Lischinski
The Hebrew University



32,965 points, 22.75ms
1,449,011 points per second

34,897 points, 15.7ms
2,222,739 points per second

22,748 points, 11.67ms
1,949,272 points per second

Figure 1: Zooming into a stippled non-photorealistic rendering. Each image shows a subset of the same implicitly infinite point set: while zooming in, more points are shown to maintain the apparent density. Only the local visible area of the point set was evaluated for each image.

Abstract

Well distributed point sets play an important role in a variety of computer graphics contexts, such as anti-aliasing, global illumination, halftoning, non-photorealistic rendering, point-based modeling and rendering, and geometry processing. In this paper, we introduce a novel technique for rapidly generating large point sets possessing a blue noise Fourier spectrum and high visual quality. Our technique generates non-periodic point sets, distributed over arbitrarily large areas. The local density of a point set may be prescribed by an arbitrary target density function, without any preset bound on the maximum density. Our technique is deterministic and tile-based; thus, any local portion of a potentially infinite point set may be consistently regenerated as needed. The memory footprint of the technique is constant, and the cost to generate any local portion of the point set is proportional to the integral over the target density in that area. These properties make our technique highly suitable for a variety of real-time interactive applications, some of which are demonstrated in the paper.

Our technique utilizes a set of carefully constructed *progressive* and *recursive* blue noise Wang tiles. The use of Wang tiles enables the generation of infinite non-periodic tilings. The progressive point sets inside each tile are able to produce spatially varying point densities. Recursion allows our technique to adaptively subdivide tiles only where high density is required, and makes it possible to zoom into point sets by an arbitrary amount, while maintaining a constant apparent density.

Keywords: non-periodic tiling, Poisson disk distribution, blue noise, Wang tiles, anti-aliasing, object positioning, sampling, stippling, texture synthesis.

1 Introduction

Generation of point sets is a fundamental task in computer graphics, as well as many other fields, since they lie in the very foundation of any sampling technique. In computer graphics, point patterns were studied in a variety of contexts, such as anti-aliasing, distribution ray tracing, Monte Carlo path tracing, geometry processing, point-based modeling and rendering, digital halftoning, object positioning, and primitive placement in non-photorealistic rendering (NPR). Thus, it is not surprising that the properties of various distributions have been extensively studied, and a variety of techniques have been proposed for their generation.

When generating a point set, there are various aspects that should be considered. In applications such as object positioning, the main concern is the *visual quality* of the resulting pattern, such as absence of noticeable repetitions. Other applications, such as anti-aliasing, are primarily concerned with the *spectral characteristics* of the distribution, typically preferring distributions with a blue noise Fourier spectrum. Halftoning and non-photorealistic rendering are examples of applications concerned with *dynamic range* (the ability to reproduce high contrasts) and *resolution-independence* (the ability to maintain the same apparent density under varying degrees of magnification).

Beyond the concerns above, space and time efficiency is of utmost importance for any interactive application that requires distributing a large number of points. A common practical approach in such cases is to utilize tiling techniques, where one or more tiles are precomputed and then placed next to each other to form point sets of arbitrary sizes. However, designing a good set of tiles is a very challenging problem, since the goal is to encapsulate the desired *global* characteristics of the distribution into a small set of *local* building blocks.

In this paper we introduce a novel technique for tile-based generation of blue noise point sets. Our technique utilizes Wang tiles, each containing a carefully constructed point set. The use of Wang tiles enables the generation of infinite non-periodic tilings. The points in each tile form a *progressive* sequence, enabling matching arbitrary spatially varying point densities. The tiles are also *recursive*, making it possible to employ adaptive subdivision only in regions where

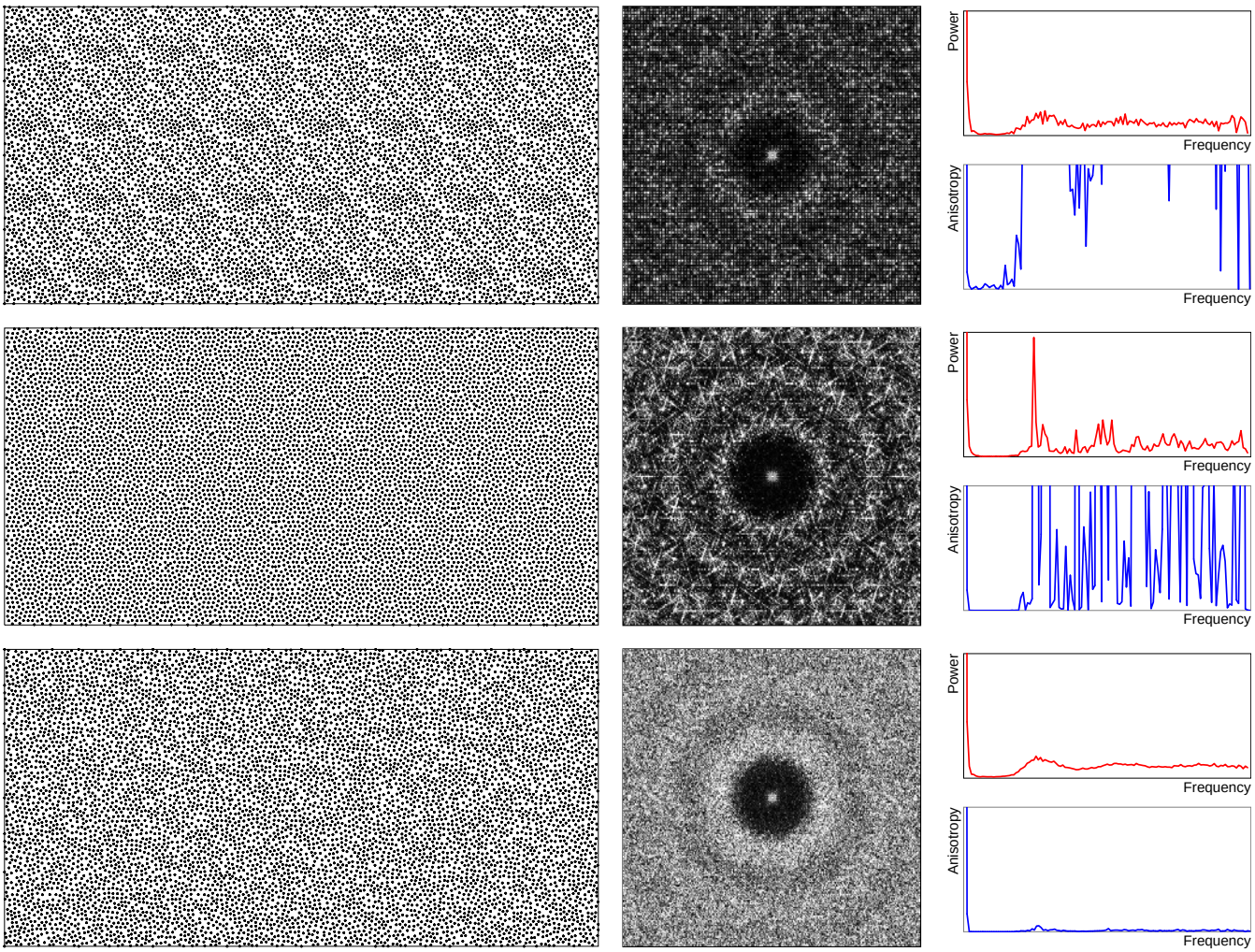


Figure 2: Uniform density blue noise point sets created with periodic tiling (top), Penrose tiling (middle), and our method (bottom). Each row shows an extract of the point set, the Fourier power spectrum, and the mean radial power and radial anisotropy plots. The repetitions in the periodic tiling are easily noticed, revealing the underlying lattice, and are also manifested by a grid of spikes in the spectrum. While the energy of the spectrum is distributed quite evenly, the anisotropy is quite strong. The energy of the Penrose tiling spectrum is concentrated into isolated spikes of energy, resulting in extreme anisotropy values. The spectrum reveals the rotational ten-fold symmetry of the Penrose tiling. Our method, in contrast, produces few repetition artifacts both spatially and in the spectrum. The energy of the power spectrum is well distributed and the anisotropy is very low, as desired for a blue noise spectrum.

high density is desired. Recursion also enables zooming into point sets by an arbitrary amount while maintaining a constant apparent density. Constructing a recursive set of tiles while maintaining progressivity and blue noise properties is challenging, as it requires the point set in each tile to become a proper subset of the point set after the subdivision.

Consequently, our method is capable of generating very large non-periodic and non-uniform density point sets possessing a blue noise Fourier spectrum and featuring high visual quality. The points may be distributed over arbitrarily large areas. The local point density may be prescribed by an arbitrary target density function, without any preset bound on the maximum density. Since our technique is deterministic and tile-based, any local portion of a potentially infinite point set may be consistently regenerated upon need. The memory footprint of the technique is constant, and the cost to generate any local portion of the point set is proportional to the integral over the target density in the area of interest. In practice, point generation speed reaches a few million points per second. These properties make our technique an ideal candidate for a variety of real-time interactive applications. As we shall show in the remainder of this paper, no other single method to date features all of these characteristics combined.

We demonstrate our method in the context of three applications: anti-aliasing, stippled non-photorealistic rendering, and interactive texture synthesis by painting texton densities.

2 Background

As mentioned earlier, the analysis of various point distributions and the development of efficient algorithms for generating point sets with various desirable characteristics has been the subject of much previous and current research over the past twenty years. Below we survey only a handful of the methods that are most relevant to this work.

Many previous authors have pointed out that for sampling in 2D, isotropic point distributions with a blue noise Fourier spectrum — minimal energy in low frequencies and lack of concentrated energy spikes — are desirable in many applications (see, e.g., [Dippé and Wold 1985; Cook 1986; Mitchell 1987; Ulichney 1988; Shirley 1991; Mitchell 1991; McCool and Fiume 1992; Glassner 1994; Hiller et al. 2001; Kollig and Keller 2002; Kollig and Keller 2003; Keller 2004; Ostromoukhov et al. 2004]). For example, when sampling a non-bandlimited function with a blue noise point set, aliasing manifests itself as high-frequency uncorrelated noise, rather than the more objectionable low frequency structures.

The *dart throwing* algorithm [Cook 1986] is one of the simplest, but also slowest, techniques for generating Poisson disk distributed point sets, which possess blue noise spectral characteristics. Random point locations are generated sequentially, and each new point is discarded if another point already exist within a certain radius around it. McCool and Fiume [1992] describe a more practical variant of dart throwing, where the dart radius is gradually decreased as more samples are placed. The order in which samples are added to the set is recorded. The result is a *progressive* sequence of points, which has the desirable property that any prefix subsequence is also Poisson disk distributed (no two points are closer to each other than the dart radius of the last point in the subsequence). This property enables generation of point sets according to a spatially varying target density function by using only those points in the set whose rank in the progressive sequence does not exceed the desired density at the corresponding location. This approach is still too slow for directly generating large point sets, but it may be used to generate a tile containing a well distributed set of points with toroidal boundary conditions, which may then be used to tile arbitrarily large portions of the plane periodically. Unfortunately, such *periodic tilings* typically suffer from obvious repetition artifacts and high anisotropy in the Fourier spectrum, as shown in Figure 2 (top row).

Ostromoukhov *et al.* [2004] introduced a much faster technique for generating blue noise patterns. The idea is to hierarchically subdivide a Penrose tiling of the plane and apply pre-computed correction vectors to improve the resulting pattern. Through a clever use of the Fibonacci number system, this technique is well suited for generation of non-uniform sampling patterns. However, uniform density patterns generated by this technique reveal some visual artifacts, which also manifest themselves in the Fourier spectrum, as demonstrated in Figure 2 (middle row).

The method presented in this paper avoids the repetition artifacts present in periodic and Penrose tilings, and exhibits higher spectral quality, as evidenced by the bottom row of Figure 2.

Another difficulty with utilizing Penrose tilings for generating very large point sets over arbitrarily large areas is that they are strictly aperiodic. Thus, it does not seem possible to quickly generate only a small local portion of the point set without starting from a very large base tiling and using recursive subdivision around the region of interest. In contrast, our method supports rapid "random access" to local portions of arbitrarily large point sets.

Non-periodic tilings of the infinite Euclidean plane may also be generated using a small set of Wang tiles [Wang 1961; Wang 1965], unit square tiles with color-coded edges. Wang tilings of the plane are obtained by placing tiles such that all adjoining edges have matching colors. Cohen *et al.* [2003] used a set of eight Wang tiles to stochastically create infinite non-periodic seamless 2D textures. Note, that although this eight-tile set is not strictly aperiodic in the sense that it can never produce periodic tilings, the stochastic process used when laying down tiles guarantees non-periodicity. Although repetitions cannot be avoided altogether when using a finite tile set, non-periodicity makes them difficult to detect visually. Hiller *et al.* [2001] and Cohen *et al.* [2003] used Wang tiles to generate non-periodic point sets with blue noise properties, utilizing Lloyd's relaxation [Lloyd 1983] to optimize an initial set of positions.

Recently, Lagae and Dutré [2005] introduced Poisson disk tiles, an interesting extension of Wang tiles aimed at rapid generation of Poisson disk distributed point sets. They construct a set of 4096 Poisson disk tiles, and introduce a "direct stochastic tiling" algorithm that uses a hashing function to support real-time placement of their tiles at arbitrary locations on the plane, without generating a complete tiling. They also describe a variant of the direct stochastic algorithm that works with the eight-tile set of Cohen *et al.* [2003], which we also use in our method.

The above Wang tiling based methods are able to generate high-quality point sets of uniform density. However, they have not been designed for directly generating the non-uniform density point sets needed by many applications. It should be noted that non-uniform density point sets may be generated by warping a uniform density one, as described by Secord *et al.* [2002] (a technique known as *inversion* or *transformation method* in the Monte Carlo literature). Indeed, this approach was used by Lagae and Dutré [2005] for stippling and for environment map importance sampling. However, warping point sets in this manner is inferior to direct generation of non-uniform density Poisson disk point sets, as demonstrated in Figure 3. This was also pointed out by Ostromoukhov *et al.* [2004].

In a contemporaneous work, Dunbar and Humphreys [2006] describe two modified dart throwing algorithms that run in $O(n \log n)$ and $O(n)$ time, respectively, and are guaranteed to terminate. The core of their method is a data structure, which allows sampling only the regions where it is legal to place a dart. The spectral quality of the resulting patterns is comparable to true dart throwing; in the $O(n \log n)$ case the results are equivalent. However, their method is currently limited to uniform point sets.

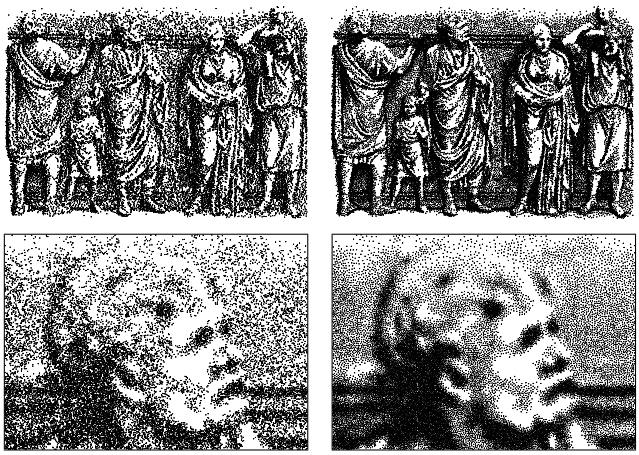


Figure 3: Non-uniform point sets generated by warping a uniform blue noise point set (left) vs. our method (right). Top: 28,493 points, bottom: 20,220 points. The images on the left are noisier and exhibit lower dynamic range.

3 Overview

The following steps provide a high-level summary of our method for generating a set of recursive Wang tiles containing progressive blue noise point sets:

1. Generate a set of *source tiles* (progressive toroidal blue noise tiles) using dart throwing with gradually decreasing dart radius [McCool and Fiume 1992].
2. Construct a set of *progressive Wang tiles*. This is done using a new algorithm for merging source tiles together, while preserving a progressive ordering and maintaining the blue noise properties of the original tiles.
3. Make the Wang tile set recursive, by establishing a *subdivision rule* for each tile in the set. A subdivision rule is a recipe for replacing a *base tile* with an $n \times n$ grid of scaled-down versions of base tiles.
4. Apply a relaxation process to make the points of each base tile a proper subset of the point set after the subdivision. This step is necessary for generation of continuous progressive transitions from the base point set to the subdivided one.

The technique outlined above is used once and for all to generate a set of tiles. Once available, these tiles may be used to rapidly generate non-uniform density blue noise point sets. We begin by constructing a coarse non-periodic base tiling locally using the direct stochastic tiling algorithm of Lagae and Dutré [2005]. The progressive ordering of the points in each tile is then used to reproduce the target density; we sample the density function for each point of each visible tile and decide whether to include the point based on its rank. Simultaneously, we check for density values exceeding the maximum density of the current subdivision level. Next, we subdivide the tiling and process only those sub-tiles covering areas where the target density has not yet been matched. We repeat the process recursively, until the target density is achieved. It is easy to see that the computational cost of the above method is roughly proportional to the integral of the target density function over the visible region of interest.

4 Progressive Blue Noise Wang Tiles

In this section we introduce our algorithm for construction of *progressive blue noise Wang tiles*, a tile set that enables us to rapidly distribute points over arbitrarily large areas on the plane according to an arbitrary target density function, such that the resulting point sets possess a blue noise spectrum.

As explained in Section 2, a progressive toroidal blue noise tile may be generated using a dart throwing algorithm, where the radius of the darts is slowly decreased [McCool and Fiume 1992]. The algorithm assigns each point a rank, so that the resulting ranking defines a progressive sequence of points with increasing density. Using such a tile it is possible to quickly generate large point sets with arbitrary non-uniform density. However, since a single tile is used to generate the point set, the tiling is periodic and repetition artifacts are apparent (Figure2).

To avoid repetition, we turn to Wang tiles, which make it possible to tile the plane non-periodically using a small set of square tiles. Cohen *et al.* [2003] describe how to generate Wang tiles that produce non-periodic point sets with blue noise properties. However, the method does not generate a *progressive* ranking of the point in each tile, and therefore their tiles are not suitable for matching arbitrary non-uniform target density functions.

It should be noted that creating a set of progressive Wang tiles is much harder than creating a single progressive tile. Recall that the edges of Wang tiles are color coded, and that any two tiles are allowed to be adjacent in a tiling if the colors of their adjoining edges match. Thus, each tile must be constructed so as to maintain the Poisson disk property across each of its edges with respect to several possible neighboring tiles. As explained by Cohen *et al.* [2003], dart throwing does not work in this case. Whenever a dart lands near a tile boundary, its Poisson disk must be checked against points in all the possible neighbor tiles. This typically results in fewer points being inserted in the vicinity of tile edges.

Therefore, we developed a new method to create a progressive blue noise Wang tile set. Our idea is to create a set of Wang tiles by merging together several progressive toroidal *source tiles*, similarly to the merging of texture tiles in [Efros and Freeman 2001; Cohen et al. 2003]. Our goal is to generate a set of $2K^2$ Wang tiles, where K is the number of different edge colors. We begin by computing a unique source tile for each of the K edge colors. To create each one of the $2K^2$ Wang tiles, we begin with a new source tile, and merge it with the source tiles corresponding to the colors of its edges, as shown in Figure 4. After merging the source tiles we define a sequential ordering of the resulting points to make the tile progressive. The tiles generated in this manner fit together seamlessly, because the points in the vicinity of each edge of a particular color always originate in the same source tile. In the remainder of this section we describe the two steps (merging and reordering) in more detail.

As outlined above, to create each Wang tile we start with a fresh unique source tile T and merge in four other source tiles N, E, S, W corresponding to the colors of the edges, one at a time. Consider the situation shown in Figure 4b, where T has been merged with a tile E corresponding to the color of its east edge (pink). To merge these two tiles we compute a *seam* connecting the endpoints of the right tile edge (shown in orange). Points to the left of this seam come from tile T, while points to the right of the seam come from tile E. Since both T and E are blue noise tiles, points away from the seam are well separated from each other, so our goal is to find a seam that avoids short distances between points across the seam.

We begin by computing a Voronoi diagram of the union of the point sets $T \cup E$. Each edge in the Voronoi diagram separates two neighboring points in the unified point set. Thus, our idea is to construct the seam from a sequence of Voronoi edges which separate pairs of points that are as far from each other as possible. More precisely, we construct a planar graph whose edges are the edges of the Voronoi diagram, clipped by the boundary of the tile. The corners of the tile are added as vertices to this graph. Each Voronoi edge is assigned a cost $c = (1 - d/d_{max})^r$, where d is the distance between the two points separated by the edge and d_{max} is the maximum among these distances. Thus, the cost penalizes short distances. The exact penalty is determined by the exponent r (we used $r = 10$ in our experiments). Let s and t denote the vertices corresponding to the endpoints of the tile's east edge. We compute the min-cost path between s and t using Dijkstra's shortest path algorithm [Cormen et al. 2001], and use the result as our seam. Using the shortest path has the nice property that it prevents the seam from venturing too deep into the interior of the tile, so that each tile in the resulting set has a unique interior part (the black points in Figure 4c). This reduces repetitions and improves the spectral characteristics of the resulting tiled point sets.

Merging in the four edge source tiles yields the full density point set for the Wang tile. In order to make this tile progressive, we must find an appropriate ranking (sequential ordering) of the points in the set. We initialize the ranking by sorting the points according to their ranks in the original source tiles, resolving conflicts (points

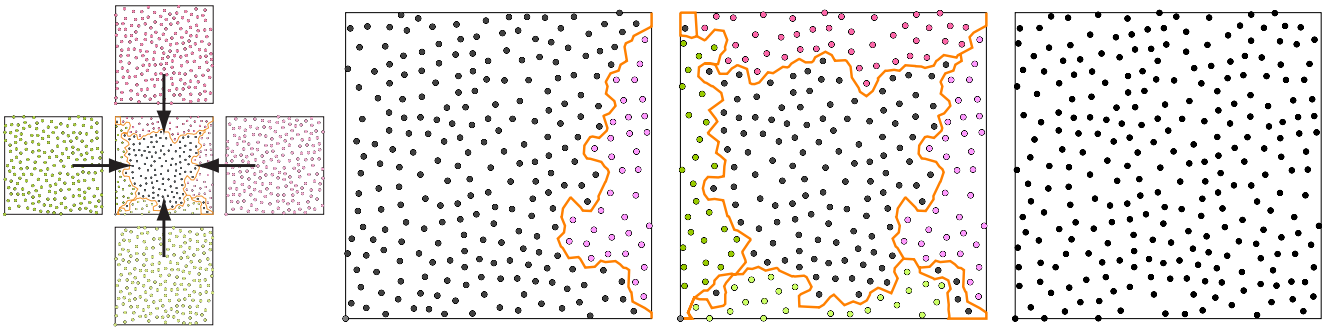


Figure 4: Merging source tiles to create a blue noise Wang tile (a) Each edge has an associated source tile. The Wang tile is initialized to a *unique* source tile (black points). (b) After merging with the source tile associated with the east edge. (c) After merging with all four edge tiles. Note that a large portion of the resulting point set comes from the initial unique tile. (d) The same point set with seams removed and all points colored black. The resulting set is well distributed and the seams are impossible to detect.

from different source tiles having the same rank) in a random fashion. This effectively "interleaves" the original sequences to yield a single new sequence.

Since the resulting ranking is consistent with the original ranking, it works fine inside the contiguous regions originating in the different source tiles (and in particular across tile boundaries). However, this ranking fails to account for the adjacencies across the seams. Thus, it is possible for two points with a low rank to be too close to each other. Our goal is now to prevent this from happening, while perturbing the ranking as little as possible.

Beginning with the initial ranking described above, we generate a new ranking iteratively, by fixing the ranks of the points one by one, until all ranks have been fixed. In each step, we assign the next available rank to the *first* point from the initial ranking, whose minimal distance to all previously fixed points is greater than a threshold function. In the densest possible packing of the unit square with k non-overlapping disks (hexagonal lattice), the radius of each disk is approximately $0.54/\sqrt{k}$. We thus choose $\alpha/\sqrt{k}$ with $\alpha = 0.5$ as our threshold function. The entire process is summarized in pseudocode in Figure 5.

```
foreach tile in the Wang tile set do
    // create a blue noise tile
    create a unique initial blue noise source tile T
    foreach edge tile in (N,E,S,W) do
        T ← Merge(T, edge tile)
    end
    // create a progressive ranking
    interleave the original rankings
    for i = 2 to n−1 do
        for j = i to n−1 do
            d = min_{k<i} ||p_j − p_k||
            if d > α/√i then break
        end
        swap p_i and p_j
    end
end
```

Figure 5: Progressive blue noise Wang tile set creation pseudocode. p_k denotes the position of the k-th point in the current ordering.

5 Recursive Point Sets

Our progressive Wang tiles enable generation of point sets with superior spectral quality at the same speed as periodic progressive tilings; specifically, the anistropy is much lower and the mean radial power is closer to the desired blue noise profile (see Figure 2). The maximal density of these point sets is determined by the number of points in each tile, and by the area of the tile. This causes some visual artifacts when the target density function has high dynamic range: in order to match the highest target density the tiles must be scaled down, but then in regions with low target density only a few points in each tile are selected, sometimes revealing the underlying tile grid.

Furthermore, although arbitrarily high densities may be generated by scaling down the tiles, the maximum density must be known in advance. In certain scenarios, such as interactive object placement or adaptive sampling, this is a limiting requirement.

In this section we introduce *recursive Wang tiles*, an extension of our progressive Wang tiles that makes it possible to generate arbitrarily high densities in a fully adaptive manner using recursive subdivision. The idea is to associate one or more *subdivision rules* with each *base tile* in the Wang tile set. A subdivision rule replaces a base tile with a grid of $n \times n$ scaled down versions of tiles from the same set.

Using recursive Wang tiles, we start by tiling the plane using a coarse grid of large base tiles. In areas where the desired density exceeds the maximum density of the base tile, the tile is recursively subdivided until the desired density is met, as was described in Section 3. Our method for deriving the subdivision rules is described in Section 5.1. In order to obtain a smooth progressive transition from a base tile to its subdivision we carefully design the tiles such that the point set of each base tile is a *proper subset* of the point set after the subdivision. This is achieved by relaxation, as described in Section 5.2.

5.1 Subdivision Rules

When subdividing a base tile, the resulting arrangement of the $n \times n$ sub-tiles must respect the color matching constraints of a valid Wang tiling. These constraints must be enforced both (i) across the subdivided boundaries of adjacent base tiles, and (ii) across the edges between the sub-tiles in the interior of the base tile. We satisfy the first constraint by associating each base tile edge color with

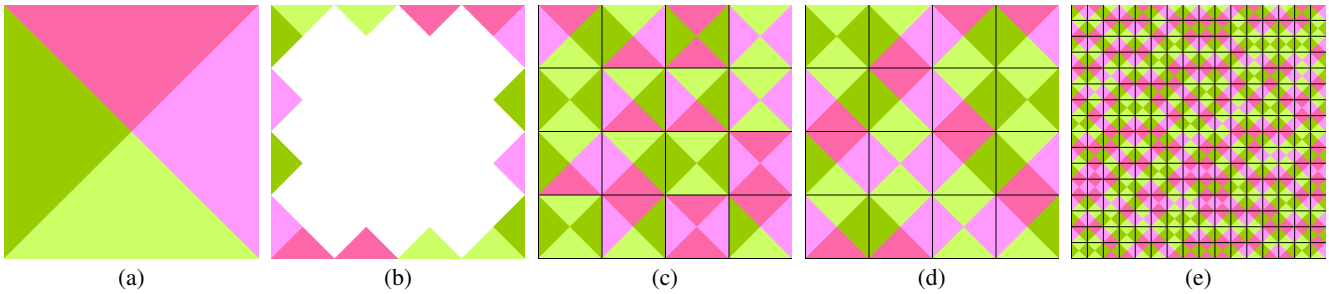


Figure 6: Recursive Wang Tiling: (a) base tile; (b) exterior edge constraints; (c) initial random subdivision tiling; (d) repaired subdivision; (e) recursive subdivision of (d). Note that steps b–d are done in pre-processing once and for all.

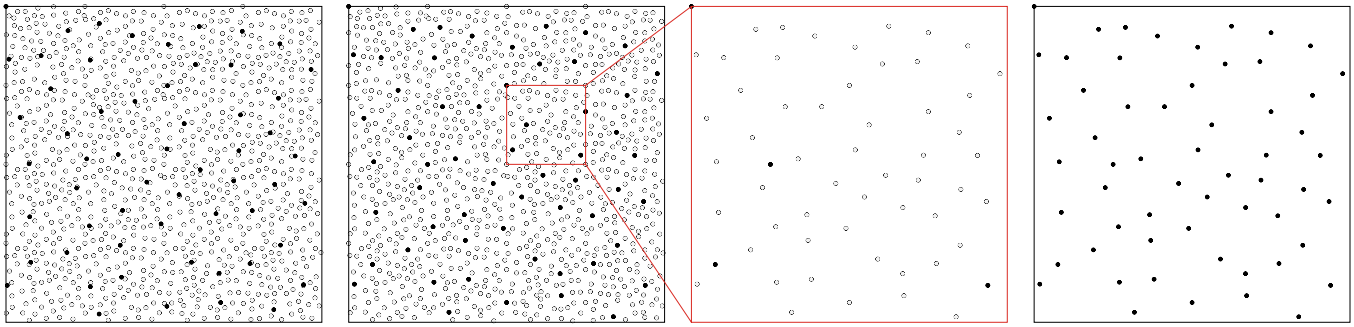


Figure 7: Generation of a recursive progressive Wang tile: (a) Original base (black) and subdivision points (outlined); (b)–(c) Point set after the relaxation procedure, the base points are now a subset of the subdivision points: the tile is self-similar. (d) Base points of the tile only. Note that the points in the zoomed window correspond to the base points exactly.

a *unique* n-color sequence (see Figure 6a–b). Now we must find an arrangement of sub-tiles that agrees with these color sequences, while also satisfying the second constraint.

With complete Wang tile sets (K^4 tiles for K edge colors), we can simply use the scanline algorithm of Cohen *et al.* [2003] to determine the interior sub-tiles. However, this is not possible with the *compact* tile sets having $2K^2$ tiles, which cannot guarantee satisfying more than two simultaneous edge constraints, as is necessary at the borders of the base tile.

We employ a stochastic search in order to find a valid interior arrangement of sub-tiles. We begin by tiling the base tile with sub-tiles chosen at random, without accounting for the color matching constraints (Figure 6c). Next, we iteratively "repair" the tiling: in each step we randomly select a sub-tile and count the number of constraints it violates. The sub-tile is then replaced with another one that has a fewer or equal number of violations. If a valid arrangement exists, the search converges rather quickly, e.g., in the order of milliseconds for $1 \rightarrow 4\times4$ or seconds for $1 \rightarrow 64\times64$. If the process fails to converge, we simply start over with a different assignment of n-color sequences to the external base tile edges. This process is illustrated in Figures 6c–d. Figure 6e shows the result of applying two levels of recursive subdivision to the base tile in Figure 6a. Recall that this procedure is merely a pre-processing step, computed once for all the tiles in a particular Wang tile set. Valid recursive Wang tile sets are included in the supplementary material.

We use the direct stochastic tiling algorithm by Lagae and Dutré [2005] to generate an infinite non-periodic base tiling. The algorithm allows randomly accessing arbitrary parts of the tiling. To further increase the variety of the recursive tiling, it is possible to use a set of l different subdivision rules for each tile. During runtime, a deterministic hashing function is evaluated at the integer lattice position of the tile to select one of the l subdivisions. However, for the recursive point sets described in the next section, we used only one subdivision rule per tile.

5.2 Progressive Recursive Point Sets

As mentioned earlier, our goal is to produce a set of base tiles and subdivision rules, such that the point set of each base tile (*base points*) is a proper subset of the subdivision point set. We achieve this property with a relaxation procedure. At first, after subdividing a base tile, the base points do not coincide with points in the subdivision set (Figure 7a). In each relaxation step, each base point is moved slightly towards its nearest point in the subdivision set. This adjustment is performed simultaneously for all the base tiles. Note that moving the base points affects all subdivision sets as well, since they are formed from scaled down versions of the base tiles. The relaxation process converges after a small number of iterations (typically 30–40). The result is illustrated in Figure 7b–c.

Finally, we must determine a new ranking for the points of the subdivision set. The first N ranks are reserved to the points coinciding with points of the base set. The ranking of the remaining points is established using a modified dart throwing algorithm (with a decreasing dart radius), where the dart locations are generated by randomly selecting points from the subdivision set. The order in which the dart throwing algorithm accepts the points becomes their new ranking. For more details we refer to the pseudocode presented in Figure 8 and the source code in the supplementary material.

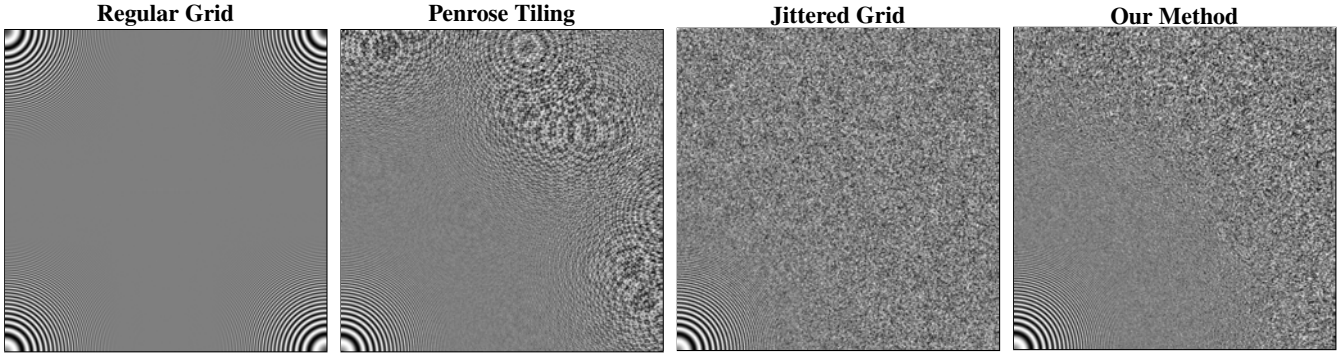


Figure 9: The zone plate test pattern sampled with one sample per pixel using various patterns. The center is in the lower left corner. A Gaussian filter was used to reconstruct the shown images. Regular sampling yields strong aliasing artifacts. Penrose tiling improves over this, but artifacts may still be observed. The jittering removes the aliasing almost completely, but the reconstructed image shows a considerable amount of noise. Our method similarly reduces the aliasing, but at a significantly lower noise level. Note that the pattern generation time with our method was even lower than jittering with a fast random number generator.

```
// Relax until base points ⊂ subdivision sets
repeat
    foreach base point p in any tile do
        s ← closest point to p in subdivision set
        d_p ← s − p
    end
    foreach base point p in any tile do
        p ← p + α d_p // α = 0.1
    end
until Σ_p d_p < ε // ε = 0.000001

// Create progressive ranking
candidates ← subdivision point set − base points
ranking ← base points
radius ← 1
repeat
    c ← random candidate
    if min_{p∈ranking} ||c − p|| > radius then
        remove c from candidates
        append c to ranking
    end
    if some number of consecutive failures then
        radius ← γ · radius // γ = 0.99
    end
until no candidates left
```

Figure 8: Progressive and recursive point sets creation pseudocode. The algorithm is executed simultaneously for all tiles. See the source code in the supplementary materials for full details.

6 Results and Applications

Using the techniques described above, we have constructed a set of eight progressive and recursive Wang tiles with 2048 points per tile. The entire construction process took about 20 minutes. Once the tile set has been constructed, point sets are generated as described in Section 3. We found point generation to be extremely fast on the order of several millions of points per second (see Figures 1 and 11). Almost all of the time is spent sampling the density function, while the time spent on tile subdivision is negligible in comparison. The actual time to generate the points in a particular region of interest is proportional to the integral over the density in that region.

As discussed earlier, our technique features several desirable properties that make it well suited for a large variety of applications. In the remainder of this section we demonstrate its use in three different contexts: blue noise sampling, non-photorealistic rendering, and object distribution.

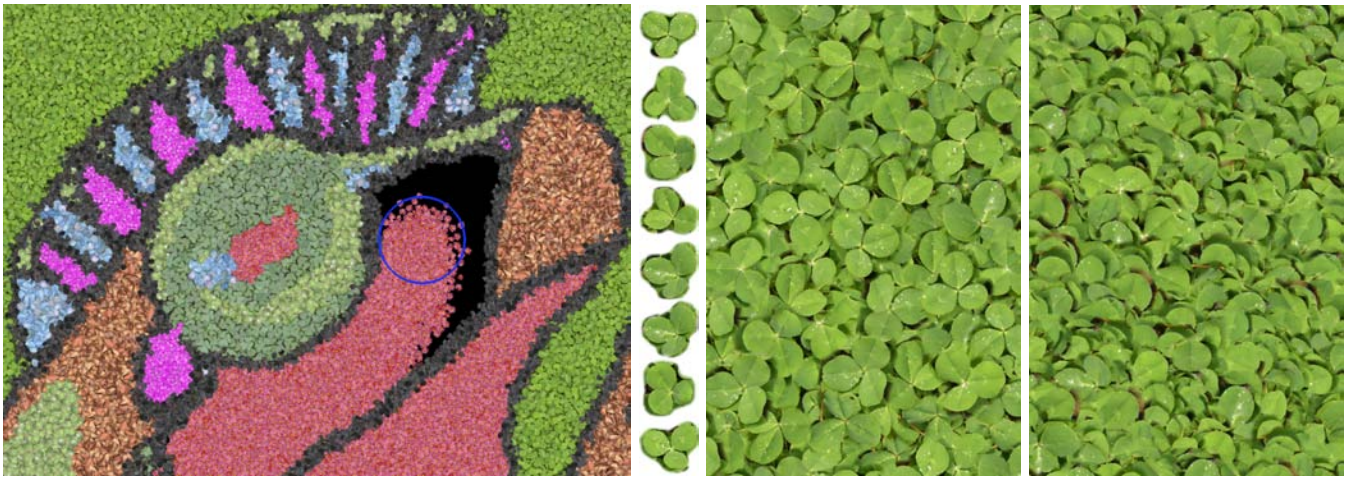
6.1 Blue Noise Sampling

The spectral quality of the point sets generated by our technique has already been demonstrated in Figure 2. Next, we test the effectiveness of such point sets for anti-aliasing. Figure 9 shows several reconstructions of the "zone plate" test pattern ($\sin(x^2+y^2)$). Each 512×512 image shows the result of sampling the pattern with one sample per pixel and filtering with a three pixel wide Gaussian kernel.

As expected, sampling using a regular grid causes severe aliasing: the upper and right circular patterns are aliasing artifacts. With a Penrose tiling pattern [Ostromoukhov et al. 2004] the situation is improved, however significant aliasing may still be observed, particularly at higher frequencies (note how the aliasing corresponds to the locations of energy spikes in Figure 2). Jittered regular grids are commonly used for anti-aliasing. The aliasing is almost completely removed; however, a considerable amount of noise is present in the reconstructed image. Our method similarly reduces the aliasing, but features a significantly lower noise level. Note that the generation of uniform density point sets with our method is just as fast as generating jittered regular grids. Results for additional test patterns are included in the supplementary material.

6.2 Non-photorealistic rendering

Many non-photorealistic rendering techniques require distributing primitives, such as pen or brush strokes or stipples (see, e.g., [Winkenbach and Salesin 1994; Salisbury et al. 1994; Hertzmann 1998; Deussen et al. 2000; Durand et al. 2001; Praun et al. 2001; Kalnins et al. 2002; Secord et al. 2002; Hiller et al. 2003]). Different gray tones are often reproduced by varying the local density of primitives: in dark regions they are placed closer together than in light regions.



(a) (b) (c) (d)

Figure 10: Our interactive texture painting application. (a) Snapshot from a painting session. (b) One of the texton classes. The full set contains 6 classes. A blue noise placement of textons using our method results in superior visual quality (c), compared to random texton placement (d).

The visual quality of the result depends directly on the quality of the primitive distribution. Human viewers easily detect unwanted regularities or clumping of primitives. In particular, the stippling technique [Deussen et al. 2000; Secord 2002; Secord et al. 2002; Hiller et al. 2003] is sensitive to these artifacts, because the simple primitives (pure circular dots) are unable to hide flaws in the distribution. Another important aspect is the *frame-to-frame coherence* in NPR animations: the positions of primitives in each new frame should be consistent with their positions in the previous frame.

We should note that progressive stroke textures may also be used to create coherent hatched images [Salisbury et al. 1994; Praun et al. 2001]. Instead of computing positions of independent strokes, a set of textures captures hatchings at various tones and scales. These textures are applied to objects at runtime. However, due to the discrete nature of texture, these methods provide only limited zooming abilities.

Our method is well suited for real-time stippling. The non-periodicity of our point sets reduces detectable repetitions, and the even spacing of stipples reduces overlaps, and allows producing a high dynamic range using a low number of stipples, as demonstrated in Figure 3. Furthermore, our technique provides superior frame-to-frame coherence: when panning and zooming over stippled illustrations points move consistently with the camera motion and new points are inserted in-between, when necessary, to maintain a constant apparent tone. Generating additional stipples while zooming in comes without a performance penalty due to the local nature of our technique. This is demonstrated in Figure 1 and in the accompanying video.

6.3 Object positioning

Many scenes in computer graphics consist of extensive distributions of objects, e.g., ecosystems or crowds. In most cases, only a small set of representative objects are created, and *instances* (possibly with transformations) are distributed throughout the scene [Sutherland 1964]. Avoiding very small distances between adjacent instances is often a desirable property. For example, biological and environmental forces in nature often do not result in random plant locations. Instead, plants prevent others from growing in their direct vicinity and, thus, Poisson disk distributions arise.

Large scenes might contain many millions of instances; thus, it might not be feasible to even store their positions explicitly. In contrast, a density map is a much more compact representation, and it may be compressed even further using standard image coding techniques. The high speed of our technique allows computing the instance positions on-the-fly from the density maps only where needed at any given time.

We demonstrate our technique for instance positioning in an interactive texture painting application (see Figure 10). The user provides an ordered set of "texture classes", each consisting of a few representative textons (small cut-out images). During runtime, the textons are simply rendered with alpha blending. We allow the user to control the placement of textons by directly painting individual density maps for the texton classes using various brushes.

As the user paints, the texton positions are generated in real-time using our technique. First, we generate a point set according to a combined density map, which is the sum of the individual density maps. Then, we determine a texton class for each point based on the values of the individual density maps at the corresponding location. Each texton is deterministically assigned a random-looking rotation angle, based on its coordinates. Using our method for the placement of textons results in a much higher visual quality compared to simpler distributions, such as randomly positioned textons (Figure 10c–d). This application is also demonstrated in the accompanying video.

7 Conclusions

We have presented a technique for the rapid generation of blue noise point sets with non-uniform density. The global characteristics of blue noise distributions are encapsulated into a small set of tiles. The point set in each tile is carefully precalculated so that the tiles can fit together spatially as well as recursively among themselves in scale space. The strength of the recursiveness of our tiles is that it provides unlimited dynamic range of the point sets.

With our technique, the online distribution of points is extremely fast since all expensive calculations are applied in a preprocessing

stage. Any local window of the point set is generated at a cost proportional to the integral over density in that window with a constant memory footprint.

In future work, we plan to explore additional applications which require a high speed generation of high quality point sets. An interesting research direction is the extension of our technique for generation of Poisson hyper-ball distributions in higher dimensions. Another important direction for further research is the creation of point sets with characteristics other than blue noise.

Acknowledgments

This work was supported in part by a grant of the LION foundation, by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities, by the Israeli Ministry of Science and Technology, and by DFG Graduiertenkolleg/1042 "Explorative Analysis and Visualization of Large Information Spaces" at University of Konstanz, Germany.

References

COHEN, M. F., SHADE, J., HILLER, S., AND DEUSSEN, O. 2003. Wang tiles for image and texture generation. *ACM Transactions on Graphics 22*, 3 (Proc. SIGGRAPH 2003), 287–294.

COOK, R. L. 1986. Stochastic sampling in computer graphics. *ACM Transactions on Graphics 5*, 1, 51–72.

CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L., AND STEIN, C. 2001. *Introduction to Algorithms*, second ed. The MIT Press, Cambridge, MA, USA.

DEUSSEN, O., HILLER, S., VAN OVERVELD, C. W. A. M., AND STROTHOTTE, T. 2000. Floating points: A method for computing stipple drawings. *Computer Graphics Forum 19*, 3 (Proc. Eurographics 2000), 40–51.

DIPPÉ, M. A. Z., AND WOLD, E. H. 1985. Antialiasing through stochastic sampling. *Computer Graphics 19*, 3 (Proc. SIGGRAPH 85), 69–78.

DUNBAR, D., AND HUMPHREYS, G. 2006. A spatial data structure for fast Poisson-disk sample generation. *ACM Transactions on Graphics 25*, 3 (Proc. SIGGRAPH 2006).

DURAND, F., OSTROMOUKHOV, V., MILLER, M., DURANLEAU, F., AND DORSEY, J. 2001. Decoupling strokes and high-level attributes for interactive traditional drawing. In *Rendering Techniques 2001*, 71–82.

EFROS, A. A., AND FREEMAN, W. T. 2001. Image quilting for texture synthesis and transfer. In *Proc. SIGGRAPH 2001*, 341–346.

GLASSNER, A. S. 1994. *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

HERTZMANN, A. 1998. Painterly rendering with curved brush strokes of multiple sizes. In *Proc. SIGGRAPH 98*, 453–460.

HILLER, S., DEUSSEN, O., AND KELLER, A. 2001. Tiled blue noise samples. In *Proc. Vison Modeling and Visualization 2001*, 265–272.

HILLER, S., HELLWIG, H., AND DEUSSEN, O. 2003. Beyond stippling — methods for distributing objects on the plane. *Computer Graphics Forum 22*, 3 (Proc. Eurographics 2003), 515–522.

KALNINS, R. D., MARKOSIAN, L., MEIER, B. J., KOWALSKI, M. A., LEE, J. C., DAVIDSON, P. L., WEBB, M., HUGHES, J. F., AND FINKELSTEIN, A. 2002. WYSIWYG NPR: drawing strokes directly on 3D models. *ACM Transactions on Graphics 21*, 3 (Proc. SIGGRAPH 2002), 755–762.

KELLER, A. 2004. Myths of computer graphics. In *Monte Carlo and Quasi-Monte Carlo Methods 2004*, Springer-Verlag, Berlin, Germany.

KOLLIG, T., AND KELLER, A. 2002. Efficient multidimensional sampling. *Computer Graphics Forum 21*, 3 (Proc. Eurographics 2002), 557–563.

KOLLIG, T., AND KELLER, A. 2003. Efficient illumination by high dynamic range images. In *Rendering Techniques 2003*, 45–50.

LAGAE, A., AND DUTRÉ, P. 2005. A procedural object distribution function. *ACM Transactions on Graphics 24*, 4, 1442–1461.

LLOYD, S. 1983. An optimization approach to relaxation labeling algorithms. *Image and Vision Computing 1*, 2, 85–91.

MCCOOL, M., AND FIUME, E. 1992. Hierarchical poisson disk sampling distributions. In *Proc. Graphics interface '92*, 94–105.

MITCHELL, D. P. 1987. Generating antialiased images at low sampling densities. *Computer Graphics 21*, 4 (Proc. SIGGRAPH 87), 65–72.

MITCHELL, D. P. 1991. Spectrally optimal sampling for distribution ray tracing. *Computer Graphics 25*, 4 (Proc. SIGGRAPH 91), 157–164.

OSTROMOUKHOV, V., DONOHUE, C., AND JODOIN, P.-M. 2004. Fast hierarchical importance sampling with blue noise properties. *ACM Transactions on Graphics 23*, 3 (Proc. SIGGRAPH 2004), 488–495.

PRAUN, E., HOPPE, H., WEBB, M., AND FINKELSTEIN, A. 2001. Real-time hatching. In *Proc. SIGGRAPH 2001*, 579–584.

SALISBURY, M. P., ANDERSON, S. E., BARZEL, R., AND SALESIN, D. H. 1994. Interactive pen-and-ink illustration. In *Proc. SIGGRAPH 94*, 101–108.

SECORD, A., HEIDRICH, W., AND STREIT, L. 2002. Fast primitive distribution for illustration. In *Rendering Techniques 2002*, 215–226.

SECORD, A. 2002. Weighted voronoi stippling. In *Proc. NPAR 2002*, 37–43.

SHIRLEY, P. 1991. Discrepancy as a quality measure for sample distributions. In *Proc. Eurographics '91*, 183–193.

SUTHERLAND, I. E. 1964. Sketch pad a man-machine graphical communication system. In *Proc. SHARE design automation workshop '64*, 6.329–6.346.

ULICHNEY, R. A. 1988. Dithering with blue noise. *Proc. IEEE 76*, 1, 56–79.

WANG, H. 1961. Proving theorems by pattern recognition II. *Bell Systems Technical Journal 40*, 1–42.

WANG, H. 1965. Games, logic and computers. *Scientific American 213*, 5, 98–106.

WINKENBACH, G., AND SALESIN, D. H. 1994. Computer-generated pen-and-ink illustration. In *Proc. SIGGRAPH 94*, 91–100.

Figure 11: Stippling images created with our system. Large Image: 250,869 points, generated in 99.2ms (2,528,921 points per second). Small Image: 53,042 points, generated in 19.3ms (2,748,290 points per second).