Spatial and Spectral Methods for Irregular Sampling in Computer Graphics

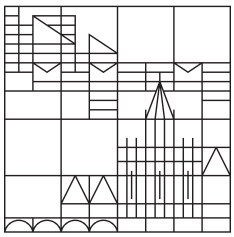
Dissertation zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften

vorgelegt von

Daniel Heck

an der

Universität Konstanz



Mathematisch-Naturwissenschaftliche Sektion

Fachbereich Informatik und Informationswissenschaft

Tag der mündlichen Prüfung: 14. Oktober 2013
1. Referent: Prof. Dr. Oliver Deussen
2. Referent: Prof. Dr. Dietmar Saupe

Abstract

Most general rasterization algorithms in computer graphics are based on point sampling of the image to be rendered. One peculiarity of computer graphics is that *irregular* sampling patterns are widely used, mainly to prevent moiré artifacts in rendered images. The best irregular sampling patterns have a *blue noise* characteristic in the spectral domain: such patterns attain a particularly good tradeoff between moiré prevention and noise-free rendition of low image frequencies.

Despite considerable research on blue noise sampling over the last 30 years, several important questions have not been answered completely so far: What is the most desirable irregular sampling pattern? To what extent can such a sampling pattern be realized in practice? What geometric properties of a sampling pattern are especially desirable? Some progress towards answering these questions has been made during the last few years. This thesis continues this line of research and presents new experimental and theoretical results on irregular sampling patterns. Our focus is on the interaction between geometric and spectral properties of sampling patterns and their impact on the sampling process.

The main contributions of this thesis fall broadly into three different areas. First, we extend previous results on the spectral analysis of irregular sampling to explain in more detail how the shape of the power spectrum of a sampling pattern affects the visual appearance of aliasing. We then study the limiting case of *Poisson disk* sampling, which is the prevalent form of irregular sampling used in computer graphics, and demonstrate that it leads to sampling patterns with certain undesirable properties. Finally, we study the mathematical relationship between spatial statistics and spectral measures to make two important contributions to the theory of blue noise sampling. First, we study two *realizability conditions*, which explain how spatial and spectral characteristics of a point set constrain each other. And second, we show how to derive efficient irregular sampling patterns directly from a specification of their desired spectral properties.

Zusammenfassung

Die meisten allgemeinen Rasterisierungs-Verfahren in der Computergrafik basieren auf der punktweisen Abtastung der darzustellenden Bilder. Eine Besonderheit in der Computergrafik ist dabei, dass häufig *irreguläre* Abtastmuster verwendet werden, in erster Linie um Moiré-Muster in den resultierenden Bildern zu unterdrücken. Die besten irregulären Abtastmuster zeichnen sich durch eine *blue noise*-Charakteristik im Spektralbereich aus: diese Muster erreichen einen besonders guten Kompromiss aus Moiré-Vermeidung und rauschfreier Darstellung niedriger Bildfrequenzen.

Obwohl *blue noise* in der Computergrafik seit 30 Jahren erforscht wird, sind einige grundlegende Fragen bisher nur unvollständig beantwortet: Was ist das günstigste irreguläre Abtastmuster? Wie kann es in der Praxis konstruiert werden? Wie korrespondiert das mit geometrischen Eigenschaften der Punktverteilung? In den vergangenen Jahren wurden deutliche Fortschritte bei der Beantwortung dieser Fragen erzielt. Diese Arbeit führt diese Forschungsansätze fort und präsentiert neue experimentelle und theoretische Ergebnisse zu irregulären Samplingmustern. Der Schwerpunkt liegt dabei auf dem Zusammenhang zwischen geometrischen und spektralen Eigenschaft der Samplingmuster und ihrem Einfluss auf den Abtastvorgang.

Die Hauptbeiträge der Arbeit fallen dabei grob in drei Bereiche. Zunächst verallgemeinern wir frühere Ergebnisse zur spektralen Untersuchung von Samplingmustern, um genauer zu erklären, wie das Leistungsspektrum der Samplingmuster das visuelle Erscheinungsbild von Aliasing beeinflusst. Wir untersuchen dann mit *Poisson disk*-Sampling eine der populärsten Formen von irregulärem Abtastmustern in der Computergrafik, und demonstrieren, dass die resultierenden Samplingmuster in einem wichtigen Grenzfall unerwünschte Eigenschaften annehmen. Zuletzt untersuchen wir den mathematischen Zusammenhang zwischen räumlichen Statistiken und Spektralmaßen, um zwei wichtige Ergebnisse abzuleiten. Zunächst untersuchen wir zwei *Realisierbarkeits-Bedingungen*, welche erklären, wie sich räumliche und spektrale Eigenschaften von Samplingmustern gegenseitig einschränken. Zuletzt untersuchen wir, wie effiziente irreguläre Samplingmuster direkt aus den gewünschten spektralen Eigenschaften abgeleitet werden können.

Acknowledgments

First, I would like to thank my advisor, Prof. Dr. Oliver Deussen, for giving me the opportunity to work in his group, for the freedom to pursue my personal interests in computer graphics, and for his patience when things didn't go as smoothly as we both hoped they would. I am also grateful to my second advisor Prof. Dr. Dietmar Saupe for his thorough and insightful feedback on this thesis.

My colleagues in the computer graphics lab made my stay in Konstanz as enjoyable as it was. I am particularly grateful to Thomas Schlömer, Michael Balzer, Hendrik Strobelt, Boris Neubert, and Sören Pirk—for discussions, coffee breaks, and extracurricular activities, both during and after work.

Few people seem be able to complete their PhD thesis without the need for periodic moral support, and I am no exception. My final thanks therefore go to my family for their continued support during the last five years.

Contents

Chapter 1

Introduction

One of the main challenges in computer graphics is *rasterization*, which is the problem of generating digital images from abstract descriptions of the image content. Almost all general-purpose rasterization algorithms are based on the idea of point sampling: During rasterization of an image I, we evaluate I at a finite set of sample positions and then use these samples to compute the final pixel colors. Since rasterization algorithms based on point sampling only require that I can be evaluated at arbitrary positions, they are very general and widely used, both for real-time and offline rendering.

Since sampling reduces the continuous image I to a finite number of samples, we may lose information in the process. If we want to ensure that the final raster image is a faithful representation of I, we have to answer the following fundamental questions:

- How many samples do we need and where do we place them?
- What image errors result if we sample incorrectly and how can we prevent or reduce such errors?

The famous *sampling theorem* answers these questions in the special case of regular sampling patterns: It relates the required sample density to the bandwidth of the image being sampled and predicts the occurrence of *aliasing* artifacts if the sampling rate is too low.

In still images, aliasing usually manifests as jagged edges or moiré patterns, both of which can be highly distracting to human observers. Preventing such artifacts by *antialiasing* is therefore an integral aspect of all rasterization algorithms. The theoretical solution to antialiasing is *prefiltering*, which removes high frequencies from the signal before sampling: In digital audio recording and digital photography, for example, such a filter can be implemented as an

analog low-pass filter in the microphone or in front of the image sensor. In computer graphics, however, exact prefiltering is only possible for very simple scenes and lighting models. This is a serious limitation because it implies that aliasing is, to a certain extent, unavoidable in graphics. Often, the best we can do is to control the amount and visibility of aliasing through the choice of sampling pattern.

One obvious way to reduce the *amount* of aliasing is to take more samples. As can be seen in the top row of Figure 1.1, increasing the sampling rate reduces aliasing by limiting the range of frequencies that can cause aliasing in the first place. Unfortunately, oversampling can only *guarantee* the absence of aliasing if the bandwidth of the image being sampled is known. Since this is rarely the case in practice, oversampling is often combined with techniques that limit the *visibility* of aliasing.

The easiest way to make aliasing less conspicuous is to use irregular sampling patterns. This replaces strong aliasing patterns with unstructured noise, as illustrated in the bottom row of Figure 1.1. The main challenge with irregular sampling is that we can choose from an uncountable number of possible sampling patterns, which complicates both the theoretical analysis and the practical implementation. As shown in Figure 1.1, the spatial distribution of the samples has a significant impact on the final image quality. The best results are often obtained with so-called *blue noise* sampling patterns (shown in the last column) which are tuned to work especially well for natural images which are dominated by low frequency content.

1.1 Overview of this Thesis

Irregular sampling in general and blue noise sampling in particular have a long history in computer graphics, and countless algorithms for constructing blue noise sampling patterns have been proposed over the years. There are two main approaches to studying blue noise sampling: the *geometric* viewpoint focuses on spatial properties of the sample points, and the *spectral* viewpoint on the properties of sampling patterns in the Fourier domain. The geometric approach is often more intuitive and is the basis of almost all algorithms for constructing blue noise sampling patterns. The spectral approach, however, gives us a direct link to the behavior of a point distribution during sampling and is usually used for evaluating sampling patterns.

In most computer graphics research so far, the two approaches have been complementary. The major goal of this thesis is to study the relationship between the two. This allows us in particular to make progress in the theo-

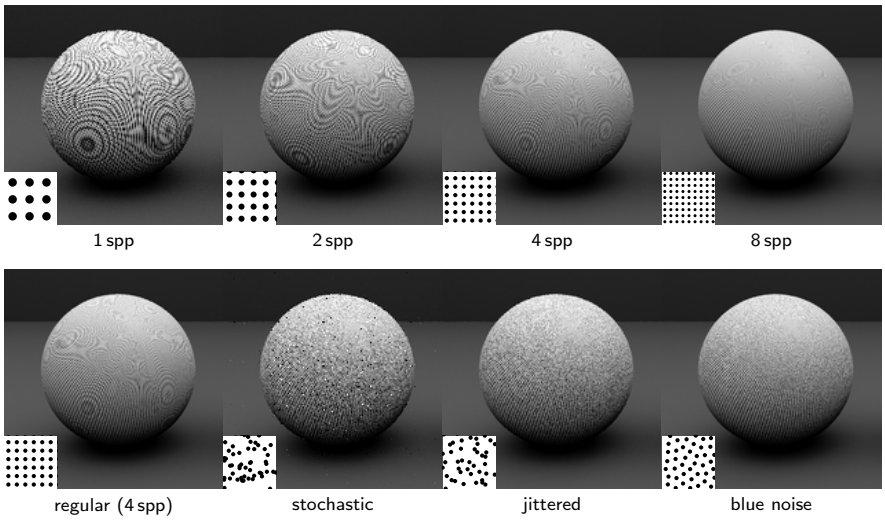


Figure 1.1: Sampling images containing regular high-frequency features often leads to aliasing in the form of moiré patterns. Two ways of dealing with moiré patterns are shown. *(Top)* Increasing the number of samples per pixel (spp) reduces the *amount* of aliasing, but in general it cannot guarantee the absence to moiré artifacts. *(Bottom)* Irregular sampling changes the *appearance* of aliasing by replacing moiré patterns with unstructured noise. All images in the bottom row use 4 spp on average.

retical investigation of blue noise sampling. Our research is motivated by the following fundamental questions that haven't been answered satisfactorily so far:

- What exactly is the effect of blue noise sampling on the frequency content of the sampled image?

- Basically all known blue noise patterns in graphics are *Poisson disk* patterns, i.e., they are constructed by enforcing a constraint on the minimal distance of two samples. Is this a necessary prerequisite for efficient sampling?

- Can we derive irregular sampling patterns from a specification of the desired spectral behavior?

- What is the most desirable blue noise sampling pattern? Under what circumstances can it be realized?

Chapter 2 begins with an introduction to sampling theory and its application to computer graphics. After briefly introducing the necessary notation and terms from signal processing, the remainder of this chapter gives an overview of existing approaches to antialiasing in graphics. The main goal of this overview is to distinguish *easy* problems, in which aliasing can be prevented almost completely, from *hard* problems, in which aliasing can only be suppressed to some extent. Irregular sampling is most useful in these hard aliasing problems, where it reduces the visibility of aliasing that cannot be removed completely by other means. The chapter concludes with a survey of irregular and blue noise sampling.

In Chapter 3 we study the irregular sampling process in the Fourier domain. We first discuss the special case of *periodic oversampling*, in which the same arrangement of samples is replicated periodically over the whole image plane; in this case, the whole image formation process can be described in a simple mathematical form. In the more general case of non-periodic sampling patterns, the analysis becomes more involved. In contrast to classical sampling theory, we cannot assume that the signal being sampled is bandlimited; we therefore put a special focus the visual appearance of aliasing and how it is related to the *power spectrum* of the sampling pattern. We use this analysis to discuss the tradeoffs involved in blue noise sampling and introduce two numerical measures for the shape of a blue noise spectrum.

In Chapter 4 we investigate the most popular paradigm for constructing blue noise patterns: the *Poisson disk* criterion. A Poisson disk pattern is an irregular distribution of points in which the points have a certain minimum separation. Empirically, it was observed that such patterns have a blue noise spectrum, and that a higher separation corresponds to better sampling properties. To test this assumption, we present a new algorithm for constructing Poisson disk patterns with a much higher separation than previously possible. Even though the resulting point sets outperform other Poisson disk patterns in many sampling applications, the requirement of a high separation leads to strong oscillations in the power spectrum. These oscillations can can cause strong aliasing artifacts when sampling certain image frequencies.

These results highlight one fundamental limitation of most previous algorithms for constructing sampling patterns: their inability to control the spectral characteristics of the sampling pattern directly. Chapter 5 therefore studies the problems involved in deriving sampling patterns directly from their spectral properties. The key to this study is the mathematical relationship between the power spectrum and a spatial statistics called the *radial distribution function*. This relationship allows us to formulate necessary conditions on

the realizability of power spectra and formulate an algorithm that constructs point distributions matching a given spectrum. We use these tools to design two new forms of blue noise patterns: *step blue noise*, which has a power spectrum shaped like a step function, and *single-peak blue noise*, which has only a single peak in the spectrum but is otherwise flat. Both blue noise patterns outperform many existing sampling patterns and guarantee the absence of aliasing artifacts for a wide frequency range.

Chapter 6 concludes the thesis with a discussion of open questions. Appendix A summarizes the mathematics behind the energy and power spectral density, and Appendix B reviews the main properties of several important classes of sampling patterns used in graphics.

1.2 Summary of Contributions

The main contributions of this thesis can be summarized as follows:

- We extend previous studies on irregular sampling in the Fourier domain to study how blue noise sampling affects non-constant images and influences the visual appearance of aliasing.

- We introduce two measures to quantify the shape of power spectra, the *effective Nyquist frequency* ν_{eff} and the *oscillation* Ω. In contrast to previous attempts to quantify irregular sampling patterns, the proposed measures are directly related to the sampling behavior and the visual appearance of aliasing.

- We extend the analysis of *Poisson disk* patterns to much higher disk radii than was possible before. We demonstrate that it is possible to achieve such radii without converging towards regular arrangements, which is a common problem with previous optimization algorithms such as Lloyd's method.

- We relate the spatial distribution of a point set to its spectral properties by studying the relationship between the autocorrelation and the power spectrum. This generalizes previous results in a similar direction by Wei and Wang [2011] and puts them on a more solid mathematical foundation.

- This relationship allows us to formulate *realizability conditions*, i.e., necessary conditions for the realizability of a power spectrum by a point

distribution. This is a crucial missing link between spatial and spectral properties of point distributions. We show how these conditions constrain the range of power spectra that are achievable.

- We propose a new iterative algorithm for constructing point distributions from a given power spectrum.
- We use all of the preceding results to *design* new blue noise sampling patterns by specifying their desired spectral properties. We use the realizability conditions to find suitable parameters and construct the associated point distributions.
- Finally, we show that the sampling patterns we obtain in this way in fact have desirable properties when applied to image-plane sampling.

This thesis is primarily based on the following publications:

Thomas Schlömer, Daniel Heck, and Oliver Deussen. **Farthest-point optimized point sets with maximized minimum distance**. In *High Performance Graphics 2011*, pages 135–154. Eurographics Association, 2011.

> For this paper I was the co-author, but the work was divided evenly. I was primarily responsible for the research that also appear in this thesis, namely, the idea of moving points to the farthest point, the analysis of runtime and convergence, and writing the corresponding sections of the paper. Compared to the original publication, this thesis adds a discussion of two additional geometric criteria, namely the *coverage radius* and the *maximality* of the resulting point sets.

Daniel Heck, Thomas Schlömer, and Oliver Deussen. **Blue noise sampling with controlled aliasing**. *ACM Trans. Graph.*, 32(3), 2013.

> For this paper I was the principal researcher and author. Compared to the original publication, this thesis significantly expands on the theoretical analysis of blue noise sampling, discusses numerical issues in more depth, and extends the original evaluation.

Chapter 2

Sampling, Aliasing and Antialiasing

This chapter serves as a brief introduction to sampling, aliasing, and antialiasing in graphics. We first review important definitions and terms from classical sampling theory and discuss how they relate to graphics. We then survey the problem of antialiasing. For a few simple graphical primitives, such as polygons or image textures, antialiasing can be performed accurately and at relatively low cost. These specialized antialiasing methods do not generalize to more complex rendering problems such as ray tracing, however. Most general antialiasing techniques are based on oversampling, either using regular or irregular sampling patterns. We discuss the necessary modifications to the standard signal processing pipeline and survey previous research on *blue noise sampling*, which is one of the standard forms of irregular sampling in graphics.

2.1 Fundamentals of Sampling Theory

Sampling theory studies the problem of representing analog signals by a discrete set of coefficients so that the original signal can be recovered exactly from the coefficients [Shannon, 1949a, Oppenheim and Schafer, 2009]. The classical signal processing pipeline is shown in Figure 2.1: A continuous signal f is reduced to a countable set of point samples taken at a fixed, regular distance T.

To understand under which conditions is it possible to reconstruct f exactly from these samples, the sampling process can be modeled mathematically as follows. The sampling step converts the input signal into a discrete representation by evaluating f at all integer multiples of the sampling interval T.

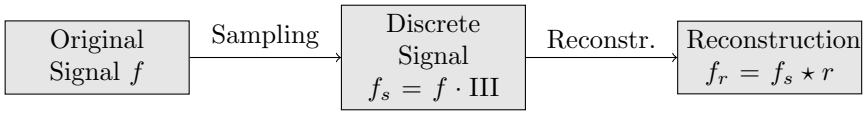


Figure 2.1: Sampling theory considers a processing pipeline in which a continuous signal f is first sampled and then reconstructed into a continuous signal f_r. The goal is to have the reconstructed signal match the original signal $f_r = f$.

This discretized signal can be represented either as the set of discrete samples $\{f[n] = f(nT)\}_{n\in\mathbb{Z}}$ or as a time-continuous signal f_s that is zero everywhere except at the sample locations:

$$f_s(x) = \sum_{n\in\mathbb{Z}} f[n]\delta(x - nT). \tag{2.1}$$

The reconstruction step converts the sampled signal f_s back into a continuous function. The reconstruction amounts to an interpolation of the samples and is performed by convolving f_s with a reconstruction filter $r(x)$

$$f_r(x) = f_s \star r(x). \tag{2.2}$$

The reconstruction is said to be *perfect* if $f_r \equiv f$, i.e., if the original signal can be recovered exactly from the samples. The *sampling theorem* [Shannon, 1949b] states that perfect reconstruction is possible if f contains no frequencies higher than $1/2T$ and reconstruction is performed using a sinc function $r(x) = \mathrm{sinc}(x/T)$. The critical frequency $\nu_c = 1/2T$ is known as the *Nyquist frequency* of the sampling grid. Signal frequencies above the Nyquist frequency cannot be represented accurately by the discrete set of samples and are mapped ("aliased") to low frequencies in the reconstructed signal.

The emergence of aliasing is easy to understand in the Fourier domain. Eq. (2.1) can be written more concisely using the *comb function* $\mathrm{III}_T(x) = \sum_{n\in\mathbb{Z}} \delta(x - nT)$

$$f_s(x) = f(x) \cdot \mathrm{III}_T(x). \tag{2.3}$$

The Fourier transformation of $\mathrm{III}_T(x)$ is another comb function $\mathrm{III}_{1/T}(\nu)/T$, so the Fourier transform of the sampled signal f_s is

$$\hat{f}_s(\nu) = \frac{1}{T}\hat{f} \star \mathrm{III}_{1/T}(\nu) = \frac{1}{T}\sum_{n\in\mathbb{Z}} \hat{f}(\nu + n/T). \tag{2.4}$$

This equation has and interesting interpretation: sampling in the spatial do-

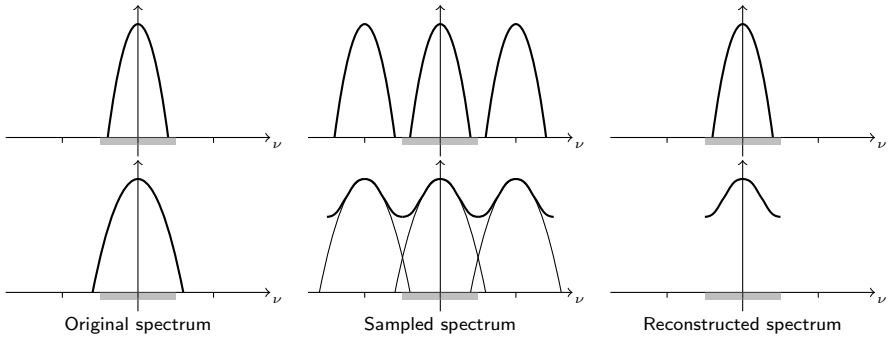


Figure 2.2: Visualization of sampling and reconstruction in the frequency domain. The spectrum of the original signal is replicated in the sampled spectrum, and its bandwidth determines whether perfect reconstruction is possible. *(Top)* If the bandwidth is smaller than the Nyquist frequency (indicated by the gray bar) the replicated spectra do not overlap, and the original signal can be recovered by using a lowpass filter that cuts out the central copy of the spectrum. *(Bottom)* If the bandwidth is too large, the replicated spectra overlap and distort the sampled spectrum. In this case, the original spectrum cannot be recovered by lowpass filtering.

main corresponds to replicating the spectrum $\hat{f}$ across the whole frequency axis and summing the result. If $\hat{f}$ contains frequencies above the Nyquist frequency $\nu_c = 1/2T$, these replicated spectra overlap and cause aliasing: frequencies $|\nu| > \nu_c$ wrap around and show up as spurious low-frequency content. This is visualized in Figure 2.2.

Aliasing can be prevented by using a lowpass filter to remove frequencies above the Nyquist frequency before sampling.

$$f_p(x) = p \star f(x), \tag{2.5}$$

The lowpass filter $p(x) = \mathrm{sinc}(xT) \equiv \mathrm{sinc}_T(x)$ ensures that the bandwidth of f_p matches the Nyquist frequency, so the sampling theorem guarantees that this *prefiltered* signal f_p can be perfectly reconstructed. (The fact that the sinc function is used for both prefiltering and reconstruction is a coincidence; in more general formulations of the sampling theorem the two filters are distinct [Unser, 2000].)

But what if the distortion due to the prefilter is more severe than the distortion due to aliasing would be? In theory this can never happen: It can be shown that prefiltering with an ideal lowpass filter is equivalent to performing an orthogonal projection of f onto the space of bandlimited functions [Unser, 2000]. This means that, among all bandlimited functions, f_p is the best ap-

proximation to f in the L_2 sense. Prefiltering with a perfect lowpass filter therefore guarantees that the reconstructed signal is as close to the original signal as possible.

Sampling and Reconstruction of Images The preceding discussion can be easily generalized from time-continuous signals $f(t)$ to signals in higher dimensions. In the case of images, the signal being sampled is a function $I(x, y)$ of two continuous variables and the samples are arranged in a rectangular grid so that the samples are placed at a distance T both horizontally and vertically. For simplicity, we restrict our discussion to grayscale images and assume that I measures brightness only. With these conventions, the sampled image $I[i, j]$ can be written as

$$I[i, j] = I(iT, jT), \quad \text{or} \quad I[\boldsymbol{n}] = I(\boldsymbol{n}T). \tag{2.6}$$

Again, the representation in terms of scaled Diracs is useful for many calculations

$$I_s(\boldsymbol{x}) = \sum_{\boldsymbol{n}} I[\boldsymbol{n}]\delta(\boldsymbol{x} - \boldsymbol{n}) = I(\boldsymbol{x}) \cdot \mathrm{III}_T(\boldsymbol{x}).$$

The two-dimensional Dirac and comb functions are defined as follows

$$\delta(\boldsymbol{x}) = \delta(x_1)\delta(x_2), \quad \mathrm{III}_T(\boldsymbol{x}) = \mathrm{III}_T(x_1)\,\mathrm{III}_T(x_2). \tag{2.7}$$

Other functions such as $\mathrm{sinc}(\boldsymbol{x})$ or $\mathrm{rect}(\boldsymbol{x})$ are likewise generalized to two dimensions by separation of variables.

The two-dimensional sampling theorem states that the function $I(\boldsymbol{x})$ can be reconstructed from its samples $I[\boldsymbol{n}]$ if the support of its spectrum $\hat{I}(\boldsymbol{\nu})$ is contained in $[-1/2T, 1/2T]^2$. In analogy to the one-dimensional case, this reconstruction is performed by convolving the sampled image I_s with a two-dimensional sinc function:

$$I(\boldsymbol{x}) = I_s \star \mathrm{sinc}_{1/T}(\boldsymbol{x}) = \sum_{\boldsymbol{n} \in \mathbb{Z}^2} I[\boldsymbol{n}]\,\mathrm{sinc}_{1/T}(\boldsymbol{x} - \boldsymbol{n}). \tag{2.8}$$

Regular sampling and reconstruction of images is treated in more detail by Gonzalez and Woods [2008] and the survey paper by Dubois [1985].

2.2 Sampling in Computer Graphics

Even though the image-formation process in computer graphics is governed by the rules of sampling theory, several adjustments are necessary in practice. In particular, the implementation of the prefiltering and reconstruction steps differs significantly from the theoretical idealizations discussed in the previous section.

2.2.1 Image-Plane Sampling

The most common form of sampling in graphics and the main focus of this thesis is image-plane sampling; the term refers to the two-dimensional plane onto which 3D scenes are projected in computer graphics. How are prefiltering, sampling and reconstruction performed in this case?

Prefiltering is synonymous with antialiasing and covered in detail in the remainder of this chapter. In graphics, prefiltering is generally challenging since we only have incomplete knowledge about the image being rendered, especially when the image, or parts of it, are generated procedurally. Common examples of procedural content in images are procedural textures and materials and most non-trivial forms of light simulation. In such cases, sampling is often the only way to gain information about the image being rendered, so filtering *before* sampling is practically impossible.

Sampling is typically performed by computing the color at a position $\boldsymbol{x}$ in the image plane. Exactly *how* this computation is performed depends on the rendering algorithm being used and the scene being rendered and can be as simple as returning the color of the geometric primitive at $\boldsymbol{x}$ or as complex as performing complex light simulation in a path tracer.

In some rendering scenarios, it is possible to combine prefiltering and image-plane sampling with other computations (such as temporal antialiasing and light simulation) into one large multidimensional integral that can be evaluated by efficient numerical methods [Cook et al., 1984, Hachisuka et al., 2008]. While this approach can reduce rendering times significantly by exploiting spatial and temporal coherence in the scene, the influence of the sampling patterns on the final image quality becomes much harder to analyze. For this reason, we will consider the consequences of sampling in the plane independently from all other rendering aspects.

Reconstruction of digital images, finally, is either performed computationally or using physical devices such as projectors, monitors, or printers. Computational reconstruction follows Eq. (2.8) and is most often used for *resampling*

tasks such as resizing or warping [Wolberg, 1990]. For practical computations, ideal reconstruction using the sinc function is unsuitable since it has infinite support and decays slowly; instead, specially designed *reconstruction kernels* with finite support are used. The search for reconstruction kernels that are easy to calculate and yield accurate and good-looking results has received a lot of attention both in image processing [Thévenaz et al., 2000] and computer graphics [Mitchell and Netravali, 1988].

Alternatively, reconstruction of digital images can also be performed by physical output devices such as monitors or projectors which generate an analog image from the pixel values. Most devices can be modeled as a linear process in which each pixel $I[i,j]$ is transformed into a continuous light distribution of the form $I[i,j]h(x-i,y-j)$. The function h is the *point-spread function* (PSF) of the output device and describes the spatial distribution of light for a single pixel. Informally spoken, the PSF models the shape of the pixels displayed by the device. The whole reconstruction process by the output device can be described mathematically by a convolution with h as follows

$$I_r(\boldsymbol{x}) = \sum_{\boldsymbol{i}} I[\boldsymbol{i}]h(\boldsymbol{x}-\boldsymbol{i}) = I_s \star h(\boldsymbol{x}). \tag{2.9}$$

The exact shape of the PSF can vary from device to device and may even depend on the spatial position on the display, but it is often sufficient to assume a fixed, homogeneous PSF. All physically realizable PSFs differ significantly from the sinc function: for CRT displays the PSF can be modeled as a Gaussian and for LCD displays as a box kernel [Foley et al., 1996]. Perfect reconstruction in the signal processing sense therefore is not possible with physical devices, and even perfectly bandlimited functions may not be reconstructed exactly on the display. In principle it would therefore be desirable to take the PSF of the output device into account when preparing an image for display, especially when the goal is to obtain the highest possible image quality. For most applications this isn't done; the only attempt we are aware of is Kajiya and Ullner's paper on font rendering [1981].

The effect of imperfect reconstruction by PDF viewers, printers and projectors should be taken into account when studying most example images in this thesis. PDF viewers perform low-quality resampling, which often leads to additional aliasing and moiré patterns, and printers and projectors tend to blur the output. Images that illustrate aliasing artifacts such as Figure 1.1 are often best viewed on a monitor at 100% magnification.

2.2.2 Temporal Sampling

In addition to image-plane sampling, another important form of sampling occurs when dealing with animated images such as movies or interactive applications, where continuous motion is represented by a discrete set of intermediate images or *frames*. Typical frame rates are 24 Hz for movies and 30–100 Hz for interactive graphics. Rasterizing individual frames can be interpreted as a sampling operation, where each sample in time corresponds to a full 2D image. Like all sampling operations, this can cause aliasing, known as *temporal aliasing*, which maps high frequencies (fast movement) to low frequencies (slow movement). Common examples of temporal aliasing are rotating helicopter blades which appear to rotate very slowly or backwards, and CRT monitors which appear to flicker heavily.

Temporal antialiasing aims to prevent such artifacts by filtering out image changes that are too fast for the chosen frame rate. The predominant effect of this filtering is *motion blur*. Mathematically, it corresponds to filtering the time-dependent image $I(\boldsymbol{x}, t)$ using a temporal low-pass filter $w(t)$:

$$I(\boldsymbol{x}) = \int_{-\infty}^{\infty} I(\boldsymbol{x}, t) w(t) \, \mathrm{d}t \tag{2.10}$$

The most general way to perform temporal antialiasing is to evaluate Eq. (2.10) numerically, for example using Monte Carlo integration [Cook et al., 1984].

In contrast to image-plane sampling, ideal prefiltering in the signal-processing sense is often not necessary or even desirable when performing temporal antialiasing. The reason is that the visual appearance of motion blur is often an artistic decision, for example to simulate the "look" of certain analog film cameras or to exaggerate motion blur. We will not cover temporal antialiasing in this thesis; for an overview and pointers to relevant research see the article by Sung et al. [2002].

2.3 Specialized Antialiasing Techniques

Most images in computer graphics are not bandlimited since hard edges, procedural detail, and detail due to perspective compression can produce arbitrarily high image frequencies. *Antialiasing* is therefore required to prevent aliasing artifacts when rendering or sampling such images. Mathematically, antialiasing corresponds to filtering the image with a lowpass filter h before sampling

$$I_p(\boldsymbol{x}) = I \star h(\boldsymbol{x}). \tag{2.11}$$

To evaluate this convolution analytically, we obviously require an analytical expression for the image I as well. But a simple mathematical description of I is only possible in a few special cases which we discuss in this section.

2.3.1 Edge and Polygon Antialiasing

Much of the early work on antialiasing focused on simple geometric objects such as lines, circles, and polygons. When drawing such shapes on a raster display, aliasing primarily takes the shape of jagged edges; this effect is also referred to as *jaggies* or *staircasing*. For monochrome displays, such jaggies are unavoidable, but grayscale and color displays allow smoother edges by using intermediate color values.

The simplest approach to smoothing hard geometric edges is based on the idea of *pixel coverage* (Figure 2.3 (a)). When rendering a polygon, each pixel is treated as a little square and the fraction of this square covered by the polygon is used as a grayscale value [Catmull, 1978]. Even though this simple idea is intuitive and gives reasonable results for polygon edges, it performs badly for more complex antialiasing problems such as texture filtering [Smith, 1995]. The underlying problem is that pixel coverage corresponds to filtering with a box filter, which is only a crude approximation to the ideal lowpass filter in Eq. (2.11) and produces stronger aliasing *and* blurring than better filters.

A better approximation to Eq. (2.11) is illustrated in Figure 2.3 (b), in which the grayscale value of the pixel is obtained by calculating the weighted average of the polygon with a filter kernel placed at each pixel. An analytic way to compute the necessary integrals was presented by Duff [1989], who considered the convolution integral along scanlines and decomposed it according to the edges of the polygon; these partial integrals can then be solved in closed form if the filter is piecewise polynomial and the polygon is flat-shaded or Gouraud-shaded. Two generalizations of Duff's integration approach based on geometric decomposition of polygons have been proposed by Lin et al. [2005] and Auzinger et al. [2012].

If the polygon is large compared to the pixel, we see in Figure 2.3 (b) that the integral doesn't depend on the shape of the polygon or its orientation, but only on the distance from the pixel center to the edge. If this pixel-to-edge distance can be computed efficiently, edge antialiasing can be performed without 2D filtering. This approach is especially viable for lines [Gupta and Sproull, 1981, McNamara et al., 1998] and fixed shapes such fonts and vector textures for which distance fields can be precalculated [Frisken et al., 2000,

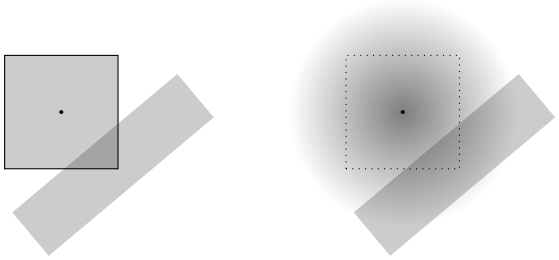


Figure 2.3: Different ways to compute the pixel intensity in edge antialiasing. The square represents a single pixel and the rotated box the edge being drawn. *(a)* Pixel coverage measures the fraction of the pixel covered by the polygon being rendered. *(b)* Convolution-based antialiasing computes the integral of the polygon weighted with the filter kernel.

Green, 2007]. For small geometric features, distance-based antialiasing gives incorrect results, however.

One final approach to edge antialiasing called *morphological antialiasing* has recently become popular in real-time rendering [Jimenez et al., 2011]. The general idea is to remove jagged edges by postprocessing: we first identify edges in the rendered image using morphological operators and then smooth them using directional filters. Combined with texture filtering, this can prevent the two most common sources of aliasing in real-time graphics. The main advantage of this approach is that it works better with complex pixel shaders than other antialiasing methods and has a predictable per-frame cost. Its main disadvantage is that it only reduces the visibility of jagged edges and ignores aliasing due to geometric detail and moiré patterns. Morphological antialiasing is a useful approximation in certain interactive applications such as computer games, but it is generally not suitable for high-quality rasterization.

2.3.2 Texture Filtering

To increase the realism of rendered scenes, surface detail can be simulated by mapping *textures* to the polygons being rendered [Heckbert, 1986b]. Due to perspective projection, textures appear warped on the display, so the frequency content of the final image depends not only on the texture image, but also on the distance from the camera, the viewing angle, and the geometry of the scene. Exact antialiasing of textures is therefore significantly more difficult and computationally demanding than antialiasing of colored polygons [Heckbert,

1986a]. *Texture lookup* refers to the tasks of deriving the antialiased color value at a particular position inside a textured polygon.

If the texture is viewed head-on, the only effect of the camera transform is to change the apparent size of the texture, and texture mapping is equivalent to *magnification* or *minification* of the texture image. Most graphics hardware uses bilinear filtering, but significantly higher quality can be obtained using higher-order filters [Thévenaz et al., 2000].

If the texture is viewed at an angle, perspective compression must be taken into account. A simple way to do this is *mip-mapping*: the strength of perspective compression is measured for each pixel by a single scaling factor, which is then used to resample the texture during texture lookup. To make this more efficient, the texture is stored as a *pyramid* and the scaling factor is only used to select an appropriate pyramid level [Williams, 1983]. Mip-mapping works well for close-up textures that aren't heavily distorted, but in general is a poor approximation to correct antialiasing: Since perspective compression is stronger in one direction than the others, using only a single scaling factor necessarily leads to blurry results in the distance.

Anisotropic filtering achieves sharper results by allowing axis-specific scaling factors. Efficient implementations of anisotropic filtering have been proposed based on summed area tables [Crow, 1984] and elliptically weighted averaging [Greene and Heckbert, 1986]. Even though these advanced texture filters are still approximations, they are widely used both for real-time and offline rendering. In principle it would be possible to achieve even more exact antialiasing of textures, but at a certain point we lose the performance benefits compared to solutions based on oversampling.

The filtering approaches discussed so far require that the texture is specified as a raster image, so they don't work for *procedural textures*. Antialiasing of procedural textures is a challenging problem and must either be performed manually by the programmer [Ebert et al., 2002] or using general antialiasing algorithms based on oversampling. One important exception is *procedural noise*, which is commonly used in shaders to simulate detail [Lagae et al., 2010]. In recent years, several procedures for procedural noise have been devised that allow the bandwidth of the produced noise to be controlled directly [Cook and DeRose, 2005, Lagae et al., 2009, 2011]. Filtering such noise functions can be performed by adjusting the frequency range.

2.4 Antialiasing by Oversampling

As discussed in the previous section, there are basically only two kinds of image features that can be antialiased by prefiltering: hard edges and image-based textures. This limits analytical antialiasing to scenes consisting of textured polygons with simple lighting. More realistic graphics require non-analytical approaches to antialiasing, most of which are based on *oversampling* combined with numerical evaluation of the prefilter in Eq. (2.11). Antialiasing based on oversampling is extremely general since it doesn't make any assumptions about the scene content, such as whether it is composed of triangles, what the lighting model is, or what the lens or camera parameters are. It only requires that the scene can be sampled by evaluating the image function I at arbitrary positions.

Oversampling implies that we take more samples than there are pixels in the final image. As a final step we therefore need to perform *resampling*, which translates the original samples S to the final pixel grid P. Resampling can be decomposed into three separate steps, shown schematically in Figure 2.4:

1. *Reconstruction* interpolates the initial samples S to form the reconstructed image I_r.

2. *Lowpass filtering* removes frequencies that cannot be represented by final sampling pattern.

3. *Downsampling* evaluates the filtered image at the new sample positions P.

The third step is usually trivial, but reconstruction and lowpass filtering must be chosen carefully to obtain good antialiasing results.

This means that there are now two separate sampling steps, and therefore two possible sources of aliasing. Mitchell and Netravali [1988] proposed the terms *prealiasing* and *postaliasing* for aliasing that is introduced during the oversampling and the downsampling steps respectively. The amount of postaliasing depends primarily on the quality of the reconstruction and lowpass filters and can be controlled relatively well. Prealiasing, however, originates in the oversampling step and can (only!) be influenced by the initial sampling pattern. Since we are primarily interested in the sampling step, we will ignore postaliasing in this thesis and use the terms prealiasing and aliasing interchangeably.

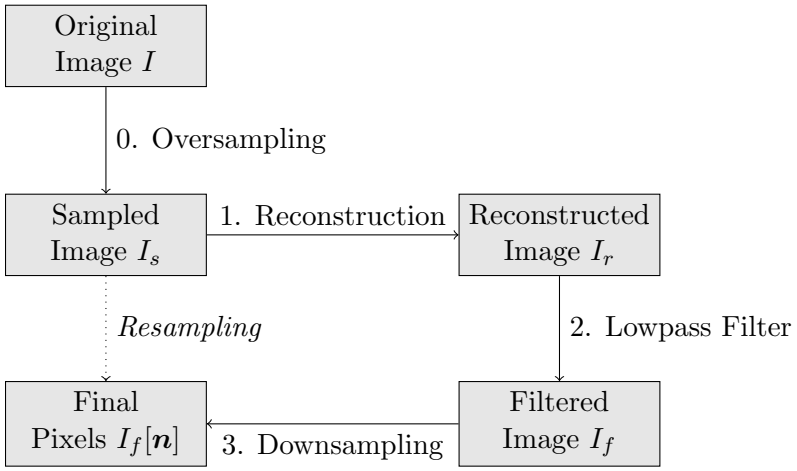


Figure 2.4: Resampling consists of three main steps: reconstruction, filtering, and the actual resampling step.

2.4.1 Regular Oversampling

In the simplest form of oversampling, the samples are positioned on a regular grid. Conceptually, regular oversampling is the same as rendering a high-resolution image without antialiasing first, and then downsampling this image to the output resolution. Regular oversampling directly affects the Nyquist frequency ν_c of the sampling grid. To double ν_c, we also have to double the vertical and horizontal sampling rates, so the cost of oversampling grows quadratically with the bandwidth of the image being sampled. For this reason oversampling is often more expensive than specialized antialiasing techniques.

One advantage of regular oversampling is that resampling to the final pixels is straightforward. Let T_s and T_p denote the sample spacings of the oversampling grid S and the pixel grid P. Both the reconstruction and lowpass filter steps from Figure 2.4 can be performed by convolving with differently scaled sinc functions. Following Eq. (2.8), the sinc for reconstruction is scaled according to the original sample spacing

$$I_r(\boldsymbol{x}) = I_s \star \operatorname{sinc}_{T_s}(\boldsymbol{x}),$$

but the sinc used for lowpass filtering is scaled according to sample distance T_p of the target grid

$$I_f(\boldsymbol{x}) = I_r \star \operatorname{sinc}_{T_p}(\boldsymbol{x})$$

Both filter operations can be combined into a single convolution with the wider

sinc, so we finally obtain

$$\begin{aligned} I_f(\boldsymbol{x}) &= I_s \star (\mathrm{sinc}_{T_s}(\boldsymbol{x}) \star \mathrm{sinc}_{T_p}(\boldsymbol{x})) \\ &= I_s \star \mathrm{sinc}_{\max(T_p,T_p)}(\boldsymbol{x}) \\ &= I_s \star \mathrm{sinc}_{T_p}(\boldsymbol{x}). \end{aligned} \tag{2.12}$$

The last step follows only if we are oversampling, which implies that the original samples are more finely spaced, i.e., $T_p \geq T_s$.

The final pixels are obtained by evaluating Eq. (2.12) at the pixel positions P. In practice, the convolution reduces to a sum since the sampled signal I_s is non-zero only at the positions of the original samples S and the sinc function is replaced by a kernel with finite support. This is the standard resampling process used in signal and image processing applications [Thévenaz et al., 2000].

Regular oversampling is simple and well-understood, but it is rarely used in graphics. Since the frequency content of most images isn't known beforehand, there is always a risk that the chosen sampling rate is too low and the rendered image contains aliasing. The most visually distracting form of aliasing in graphics is the moiré pattern (Figure 2.5). Dealing with moiré patterns is non-trivial due to the way they are perceived by the human visual system, which especially sensitive to low-frequency signals and structured patterns. In practice, even low-contrast moiré patterns are easily visible, as can be seen in Figure 2.5. For this reason, simply increasing the sampling rate is often not the most effective way to combat moiré patterns.

2.4.2 Irregular Oversampling

The standard approach to dealing with moiré patterns in graphics is *irregular sampling*, which exploits the fact that moiré patterns result from the interaction of periodic image features with a periodic sampling pattern. By sampling with an irregular sampling pattern, the structured moiré patterns are replaced with unstructured noise. The details of this process are described in the next chapter.

One direct consequence of irregular sampling is that the reconstruction process must be modified. A simple convolution such as

$$I_f(\boldsymbol{x}) = I_s \star \mathrm{sinc}_{T_p}(\boldsymbol{x}) \tag{2.13}$$

performs badly in the case of irregular samples since the nonuniform sample density leads to a nonuniform intensity distribution in the reconstructed

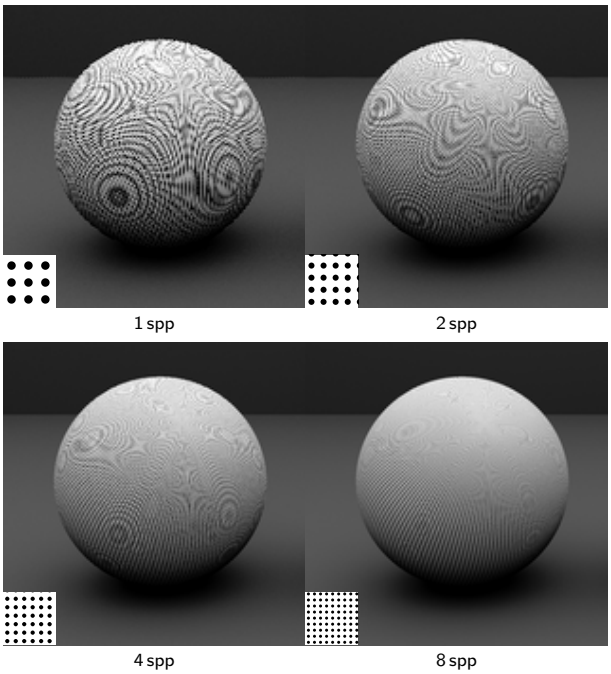


Figure 2.5: Moiré patterns remain distracting even as we increase the sampling rate. On a computer screen, structured patterns are still clearly visible at 8 spp.

image. The standard reconstruction method used in graphics includes an additional normalization term to remove the influence of the non-uniform sample density [Dippé and Wold, 1985]:

$$I_f(\boldsymbol{x}) = \frac{I_s \star \mathrm{sinc}_{T_p}(\boldsymbol{x})}{S \star \mathrm{sinc}_{T_p}(\boldsymbol{x})} \tag{2.14}$$

The denominator is basically a density map of the sample points. It is easy to see that this normalization ensures that if the original signal I is constant, the final image I_f will be constant as well.

Dividing by the normalization term in Eq. (2.14) means that the reconstruction step is no longer equivalent to a simple low-pass filter, and can no longer be analyzed using standard Fourier theory. For theoretical investigations, it is therefore customary in graphics to ignore the normalization step and assume that reconstruction is performed using a convolution as in Eq. (2.13) [Mitchell, 1991]. Although this means that our mathematical model of reconstruction is not entirely correct and will usually make pessimistic predictions,

it is still possible to derive important insights about the influence of sampling patterns.

Resampling from irregular samples is still an active area of research, and better mathematical and algorithmic reconstruction methods have been proposed in the literature [Gröchenig, 1992]. So far, these reconstruction methods have not found their way into computer graphics; this is an important open question, which we discuss in some more detail in the conclusion.

Eq. (2.14) can be rewritten in the following form which is easier to implement

$$I_f(\boldsymbol{x}) = \frac{1}{N(x)} \sum_{\boldsymbol{x}_i \in S} r(\boldsymbol{x} - \boldsymbol{x}_i) I(\boldsymbol{x}_i), \qquad N(x) = \sum_{\boldsymbol{x}_i \in S} r(\boldsymbol{x} - \boldsymbol{x}_i). \tag{2.15}$$

Here, we have replaced the sinc function with a general reconstruction kernel r. This reformulation shows that it is possible to compute $I_f(\boldsymbol{x})$ for fixed $\boldsymbol{x}$ incrementally by accumulating the effect of each sample $\boldsymbol{x}_i$. This is important because it means that it is not necessary to keep all samples $I(\boldsymbol{x}_i)$ in memory.

Irregular sampling is widely used in graphics, primarily because it effectively prevents moiré patterns. There are several disadvantages that have to be kept in mind, however:

- Irregular sampling patterns are less efficient, i.e., a higher sampling rate is required to obtain the same image quality.
- Most irregular sampling patterns are more expensive to compute, which makes them less suitable for real-time applications.
- Theoretical analysis is more difficult since the results from classical sampling theory rely on regular samples and don't generalize easily. Defining how a good irregular sampling pattern should look like is therefore a significant challenge.

2.5 Blue Noise Sampling

The ability of irregular sampling patterns to suppress moiré artifacts was first observed by Yellot [1983], who studied the spatial distribution of receptor cells on the retina of monkeys and humans. Close to the fovea, the receptors are arranged on a closely packed hexagonal grid and aliasing is prevented by optical blurring in the eye [Williams, 1985]. In the periphery of the retina, however, the receptor density decreases, so we would expect aliasing to become

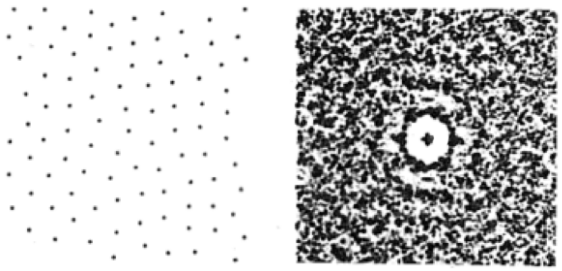


Figure 2.6: Spatial arrangement of extrafoveal receptors in monkey eyes (left) and their spectrum (right). Source: [Yellot, 1983].

an issue. But the receptor distribution in the periphery is not only more sparse but also irregular (Figure 2.6). Yellot argued that this irregularity is responsible for preventing strong aliasing effects in human vision.

The distribution of these outer receptors isn't completely random since the size of the cells imposes a constraint on their minimum separation. Mathematically, this can be modeled as a *Poisson disk* pattern, which is a random arrangement of non-overlapping disks with a prescribed radius R. Obviously, two points in such a distribution cannot be closer than $2R$. Poisson disk patterns have a characteristic shape in the Fourier domain, consisting of a single peak at the origin surrounded by an empty low-frequency region. The remaining energy is smoothly distributed in the high-frequency region, as shown in Figure 2.6

Around 1985, the idea of suppressing aliasing using irregular sampling, and in particular Poisson disk sampling, was picked up in the graphics community [Dippé and Wold, 1985, Cook, 1986]. The original focus was on ray tracing and the ability of irregular sampling patterns to mask aliasing effects, but Ulichney [1988] observed that halftoning benefits from the same kind of irregularity (Figure 2.7). Ulichney's introduced the term *blue noise* for spectra that are zero in the low-frequency region. Today, such blue noise patterns have found application in many other areas of computer graphics such as stippling [Deussen et al., 2000, Secord, 2002], general object distribution [Hiller et al., 2003, Lagae and Dutré, 2005], and improved photon mapping [Spencer and Jones, 2009].

For a long time, Poisson disk patterns were the only known point distributions with a blue noise spectrum, and the two terms were used almost interchangeably. The first algorithm for constructing Poisson disk patterns with a given disk radius R was proposed by Cook [1986]. This "dart throw-

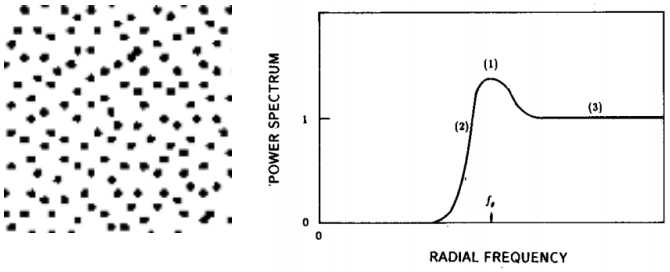


Figure 2.7: Ulichney's original characterization of blue noise [Ulichney, 1988]. Note that this profile is purely empirical: for example no justification for the small peak in the transition region is given.

ing" approach incrementally constructs the point set by generating random candidate points that are accepted if their distance to all existing points is at least R and are rejected otherwise.

The performance of dart throwing depends directly on the rejection rate, so the algorithm becomes significantly slower as more and more points are added. Most attempts to speed up dart throwing do so by employing various data structures to reduce the rejection rate [Jones, 2006, Dunbar and Humphreys, 2006, White et al., 2007, Gamito and Maddock, 2009, Kalantari and Sen, 2011]. Several alternative approaches to speed up dart throwing have been proposed, such as parallelization [Wei, 2008] and dynamically adjusting the Poisson disk radius [McCool and Fiume, 1992].

Even with these improved algorithms, constructing large Poisson-disk patterns often remains too costly for real-time applications. To speed up the creation of large point sets, *tiling methods* can be used to put together large patterns from a set of small patterns. Several different kinds of tilings have been used, such as Wang tilings [Cohen et al., 2003], Penrose tilings [Ostromoukhov et al., 2004] and corner tiles [Lagae and Dutré, 2005, Schlömer, 2012]. In addition, tile-based methods have been extended to non-uniform sample densities [Ostromoukhov et al., 2004, Kopf et al., 2006].

McCool and Fiume [1992] were the first to use Lloyd's method [Lloyd, 1982] to improve the spatial distribution of dart throwing points to distribute them more evenly in the plane. Lloyd's methods iteratively moves each point to the centroid of its associated Voronoi region, which slowly spreads out the points. The main problem with Lloyd's method is that it converges towards regular point sets. Attempts have been made to stop Lloyd's algorithm prematurely before regular patterns are formed (McCool and Fiume used 10 iterations for

their experiments), but so far no reliable stopping criterion has been found that guarantees uniform point distributions but also prevents regular patterns.

Researchers have therefore tried to devise alternatives to Lloyd's algorithm. Balzer et al. [2009] propose a modification of the usual Lloyd iteration that uses *power diagrams* instead of Voronoi diagram to compute the centroids. These diagrams are endowed with the additional property that all cells have the same *capacity* (area). Like Lloyd's methods, the algorithm leads to spatially uniform point distributions, but the constraint on cell areas prevents convergence to hexagonal arrangements. Unfortunately, the underlying algorithm for constructing power diagrams is very slow since it requires discretization of the underlying domain [Balzer and Heck, 2008]. Several algorithmic improvements to the original method have been proposed by Li et al. [2010], and de Goes et al. [2012] present a new mathematical formulation for Balzer's approach that avoids the discretization and thereby allows a much more efficient numerical solution.

The main effect of Lloyd's method is a relaxation: as the points are moved to the centroid of the Voronoi regions, they spread out and move away from each other. This is similar to the way a set of mutually repulsive particles behaves. Two recent papers have used this analogy to generate uniform point distributions. Fattal [2011] takes his inspiration from statistical mechanics and defines an energy that depends on all particle positions and is minimal if the points have maximal separation. This energy is then used to define a Boltzmann distribution parametrized by a temperature parameter that controls the amount of disorder in the system: A temperate $T = 0$ corresponds to a hexagonal grid, and $T = \infty$ to a random distribution of points. The paper then proposes an efficient Monte Carlo algorithm for drawing samples from this distribution, where each sample corresponds to a set of points in the plane. Schmaltz et al. [2010], on the other hand, model a set of points in the plane as charged particles that move under the effect of electrostatic forces. In this model, disorder is not introduced by a temperature parameter but by defining a global force field that jitters the point positions. The algorithm proposed in the paper evolves the point distribution to a stable equilibrium.

2.6 Discussion

Raster graphics and raster displays are one of the corner stones of modern computer graphics, but their discrete nature means that image artifacts due to aliasing can occur [Crow, 1977]. To this day, preventing or reducing aliasing artifacts remains a lively research area.

Research on specialized antialiasing algorithms has subsided in the last years, primarily due to the difficulty of finding problems that allow analytical prefiltering. The rise of highly parallel graphics processors, however, has made several new antialiasing schemes such as *morphological antialiasing* possible that would have been too expensive for normal CPUs.

Research on general antialiasing based on oversampling has primarily focused on *blue noise sampling*, which attempts to marry the advantages of irregular sampling (its resilience to aliasing) and regular sampling (the clean representation of low image frequencies). Most algorithms for constructing blue noise sampling patterns are based on geometrical constraints that lead to a uniform but irregular distribution of points in the plane. The spectral characteristics, which actually underly the definition of blue noise, are usually only considered as an afterthought. We will give a more precise definition of blue noise in the next chapter and then study the relationship between geometrical and spectral properties of sampling patterns in the following chapters.

Chapter 3

Fourier Analysis of Irregular Sampling

The previous chapter motivated irregular sampling simply by noting that it prevents moiré patterns when sampling regular periodic image features. In this chapter, we explain this process in more detail by studying irregular sampling in the frequency domain.

We start with a with the special case of *periodic supersampling* in Section 3.1, which is a form of oversampling that is commonly used in graphics hardware. We show that in this special case the effect of oversampling and resampling can be described as a prefilter operation, and how the low-pass characteristics of this filter depend on the spatial arrangement of the sample points.

If the sampling pattern isn't periodic, we describe the sampling process using a more general formalism, namely by considering the interaction between the *power spectra* of the original image I, the sampling pattern S, and the sampled signal I_s. We are particularly interested in the effect of the sampling pattern S. The analysis is particularly simple if the power spectrum of S can be expressed in closed form; this is the case for regular, stochastic and jittered sampling. For other irregular sampling patterns it is only possible to study how sampling affects individual frequency components. We therefore study test signals with a fixed frequency to understand how other irregular sampling patterns influence the sampled signal and affects the visual appearance of aliasing. We use these insights to explain the effect of blue noise sampling in detail.

Contributions: The analysis of periodic supersampling is a new result, and we aren't aware of any prior attempts to analyze this in the Fourier

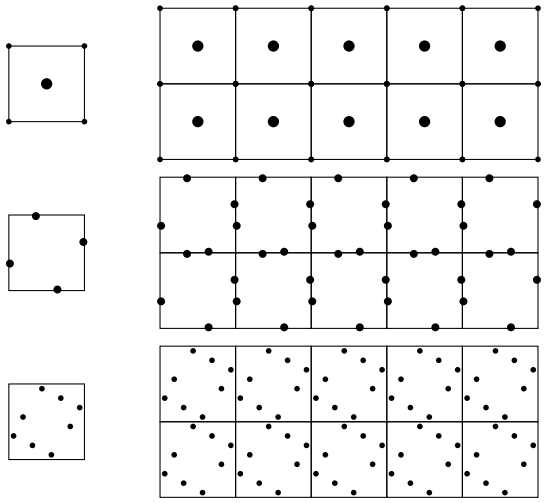


Figure 3.1: Three examples of periodic oversampling. A fixed sampling pattern (shown on the left) is repeated periodically over the whole image plane. Each square on the right-hand side corresponds to one pixel in the final output, and the size of the points is proportional to the weight it is assigned during resampling.

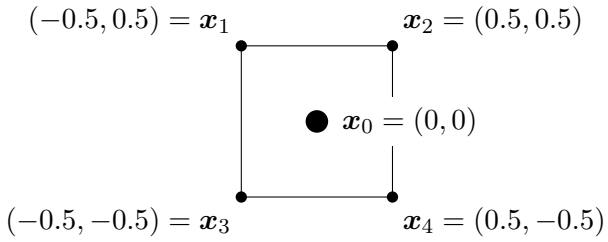
domain. The relationship between the power spectra of the sampling pattern and the sampled image is a standard result in the theory of random signals but doesn't seem to be widely known in the graphics community. We use this relationship to extend the analysis of irregular sampling by Dippé and Wold [1985] and Mitchell [1991] to non-constant images. The main contribution of this chapter is the detailed discussion of the effect of the sampling pattern on the visual appearance of aliasing, which was ignored in previous publications. This chapter is partially based on [Heck et al., 2013].

3.1 Periodic Oversampling

There is one case of irregular oversampling in which the whole processing pipeline including resampling can be analyzed in the Fourier domain. In many applications of oversampling in real-time graphics, a fixed arrangement of samples is used for each pixel and replicated over the whole image plane (see Figure 3.1). We will refer to this as *periodic oversampling*.

In this section we use a slightly different notation than in the remainder of this chapter. Instead of considering the sampling patterns as a single set $S = \{\boldsymbol{x}_i\}$ of points distributed in the image plane, we group all samples that

affect a pixel at position $\boldsymbol{n}$ together into one set $\{\boldsymbol{x}_{k,\boldsymbol{n}}\}$. The coordinates $\boldsymbol{x}_{k,\boldsymbol{n}}$ measure the sample positions *relative* to the pixel center. For example, the following image shows the relative coordinates of the *Quincunx* sampling pattern that consists of five samples for each pixel.



For the moment, we still allow arbitrary irregular sampling patterns, so the relative sample positions $\boldsymbol{x}_{k,\boldsymbol{n}}$ can depends on the pixel $\boldsymbol{n}$.

With this notation for the samples, we can write each pixel in the final image $I_f[\boldsymbol{n}]$ as a *weighted sum* of the samples in its neighborhood

$$I[\boldsymbol{n}] = I_f(\boldsymbol{n}) = \sum_{k \in \mathbb{N}} w_{k,\boldsymbol{n}} I(\boldsymbol{n} + \boldsymbol{x}_{k,\boldsymbol{n}}). \tag{3.1}$$

It can be seen that this is simply a rearrangement of Eq. (2.15) in which we have subsumed the effect of the reconstruction filter $r(\boldsymbol{x})$ and the normalization $N(\boldsymbol{x})$ into a set of *reconstruction weights* $w_{k,\boldsymbol{n}}$. For convenience, the sum extends over all $k \in \mathbb{N}$; we assume that the weights $w_{k,\boldsymbol{n}}$ are non-zero only for a finite number of samples. Again, the subscript $\boldsymbol{n}$ emphasizes that weights and sample positions can change from pixel to pixel. In real-time graphics, the combination of sample positions and weights is sometimes referred to as a *supersampling pattern*.

For each pixel $\boldsymbol{n}$, Eq. (3.1) averages close-by samples—its local action is therefore comparable to that of a low-pass filter. In fact, it is possible to write $I[\boldsymbol{n}]$ as a convolution of the original signal I with a suitable filter $h_{\boldsymbol{n}}$ followed by point-sampling at the pixel center:

$$I[\boldsymbol{n}] = h_{\boldsymbol{n}} \star I(\boldsymbol{n}) = \left[h_{\boldsymbol{n}} \star I(\boldsymbol{x})\right]_{\boldsymbol{x}=\boldsymbol{n}}. \tag{3.2}$$

As we will prove below, the impulse response $h_{\boldsymbol{n}}$ of the filter is a superposition of scaled Diracs

$$h_{\boldsymbol{n}}(\boldsymbol{x}) = \sum_{k \in \mathbb{Z}} w_{k,\boldsymbol{n}} \delta(\boldsymbol{x} + \boldsymbol{x}_{k,\boldsymbol{n}}). \tag{3.3}$$

This formulation of resampling is valid for all possible sample weights and sample positions, even if the points $\boldsymbol{x}_{k,\boldsymbol{n}}$ are distributed irregularly. We can interpret each $h_{\boldsymbol{n}}$ as a *local* prefilter in the neighborhood of pixel $\boldsymbol{n}$.

The proof that Eqs. (3.2) and (3.3) are equivalent to Eq. (3.1) is a straightforward calculation. We write out the convolution to obtain

$$\begin{aligned} h_{\boldsymbol{n}} \star I(\boldsymbol{n}) &= \left[\int_{\mathbb{R}^2} h_{\boldsymbol{n}}(\boldsymbol{u}) I(\boldsymbol{x}-\boldsymbol{u})\,\mathrm{d}\boldsymbol{u}\right]_{\boldsymbol{x}=\boldsymbol{n}} \\ &= \left[\sum_{k\in\mathbb{Z}} w_{k,\boldsymbol{n}} \int_{\mathbb{R}^2} \delta(\boldsymbol{u}+\boldsymbol{x}_{k,\boldsymbol{n}}) I(\boldsymbol{x}-\boldsymbol{u})\,\mathrm{d}\boldsymbol{u}\right]_{\boldsymbol{x}=\boldsymbol{n}} \\ &= \left[\sum_{k\in\mathbb{Z}} w_{k,\boldsymbol{n}} I(\boldsymbol{x}+\boldsymbol{x}_{k,\boldsymbol{n}})\right]_{\boldsymbol{x}=\boldsymbol{n}} = I[\boldsymbol{n}]. \end{aligned} \tag{3.4}$$

The last line elucidates why the combination of sampling and weighted averaging of the samples is shift-invariant even though point sampling in general is not: in the last line the positions $\boldsymbol{x}_{k,\boldsymbol{n}}$ no longer determine the positions at which I is evaluated but the distance by which copies of I are shifted in the weighted superposition. The only sampling operation takes place afterwards and evaluates this superposition at the pixel center $\boldsymbol{n}$.

In principle, each pixel $\boldsymbol{n}$ can have a different filter $h_{\boldsymbol{n}}$. This is the case, for example, when stochastic sampling is used. Each filter tells us how the image function I is filtered, but only in the neighborhood of the corresponding pixel. To study the *global* influence of oversampling we would like to describe its effect on the image as a whole. In the general case of arbitrary irregular sampling patterns this doesn't seem to be possible, but for *periodic* sampling patterns there is a simple, intuitive relationship between the input image, the supersampling pattern, and the final image.

Periodic supersampling uses the same set of sample positions $\boldsymbol{x}_k$ and weights w_k for each pixel. In this case we can drop the index $\boldsymbol{n}$ from all formulas and obtain

$$I_f[\boldsymbol{n}] = h \star I(\boldsymbol{n}), \quad \text{with} \quad h(\boldsymbol{x}) = \sum_k w_k \delta(\boldsymbol{x}+\boldsymbol{x}_k). \tag{3.5}$$

What this means is that the total effect of oversampling and weighted averaging can be described as a global prefilter h that is applied to the input image $I_f(\boldsymbol{x}) = h \star I(\boldsymbol{x})$, followed by sampling at the pixel coordinates $\boldsymbol{n}$. In the

Fourier domain, Eq. (3.5) can be written as

$$\hat{I}_f(\boldsymbol{\nu}) = \hat{h}(\boldsymbol{\nu})\hat{I}(\boldsymbol{\nu}), \qquad \hat{h}(\boldsymbol{\nu}) = \sum_k w_k e^{+2\pi i \boldsymbol{\nu}\cdot\boldsymbol{x}_k}.$$

When dealing with images, we are primarily interested in the attenuation of the signal, so we consider the modulus of $\hat{I}_f$

$$|\hat{I}_f(\boldsymbol{\nu})| = M(\boldsymbol{\nu}) \cdot |\hat{I}(\boldsymbol{\nu})|, \qquad M(\boldsymbol{\nu}) = |\hat{h}(\boldsymbol{\nu})|$$

The function $M(\boldsymbol{\nu})$ is the *modulation transfer function* (MTF) and depends only on the sample positions and sample weights. Ideally, we would like M to attenuate all frequencies above the Nyquist frequency ν_c of the pixel grid.

3.1.1 Visualization of Supersampling Patterns

Over the years, a large number of fixed supersampling patterns have been proposed, especially for antialiasing in real-time rendering [Akenine-Möller et al., 2008]. The results from the previous section can be used to visualize the effect of those sampling patterns in the Fourier domain. In previous studies on periodic oversampling, the behavior of different sampling patterns could only be described in terms of their geometric arrangement [Goss and Wu, 1999, Beets and Barron, 2000], so the frequency viewpoint is a welcome alternative.

The main results are shown in Figures 3.2 and 3.3. The first column shows the impulse response, the pattern's name, as well as the average number of samples per pixel n and the number of samples used in the weighted sum m. In most cases, increasing n is more costly than increasing m since it determines how often the image function I must be evaluated. For this reason, hardware-based oversampling schemes often place samples on the edge between pixels so that they can be efficiently shared; the best-known example of this strategy is the Quincunx pattern [NVIDIA, 2001].

The second column shows a 2D representation of the filter's MTF. The displayed frequency range is $[-4\nu_c, 4\nu_c]$ along both axes. The red square in the middle marks the Nyquist region of the pixel grid, the other squares are those regions of the frequency domain that are folded into the Nyquist region by the sampling operation.

In the third column we show the result of resampling the well-known Barbara image from its native resolution of 512×512 pixels to 128×128 pixels. The original image and its Fourier transform are shown in the first row. This is a common test image in image processing since the patterns on trousers, headscarf, and tablecloth easily lead to strong moiré artifacts. Since we shrink

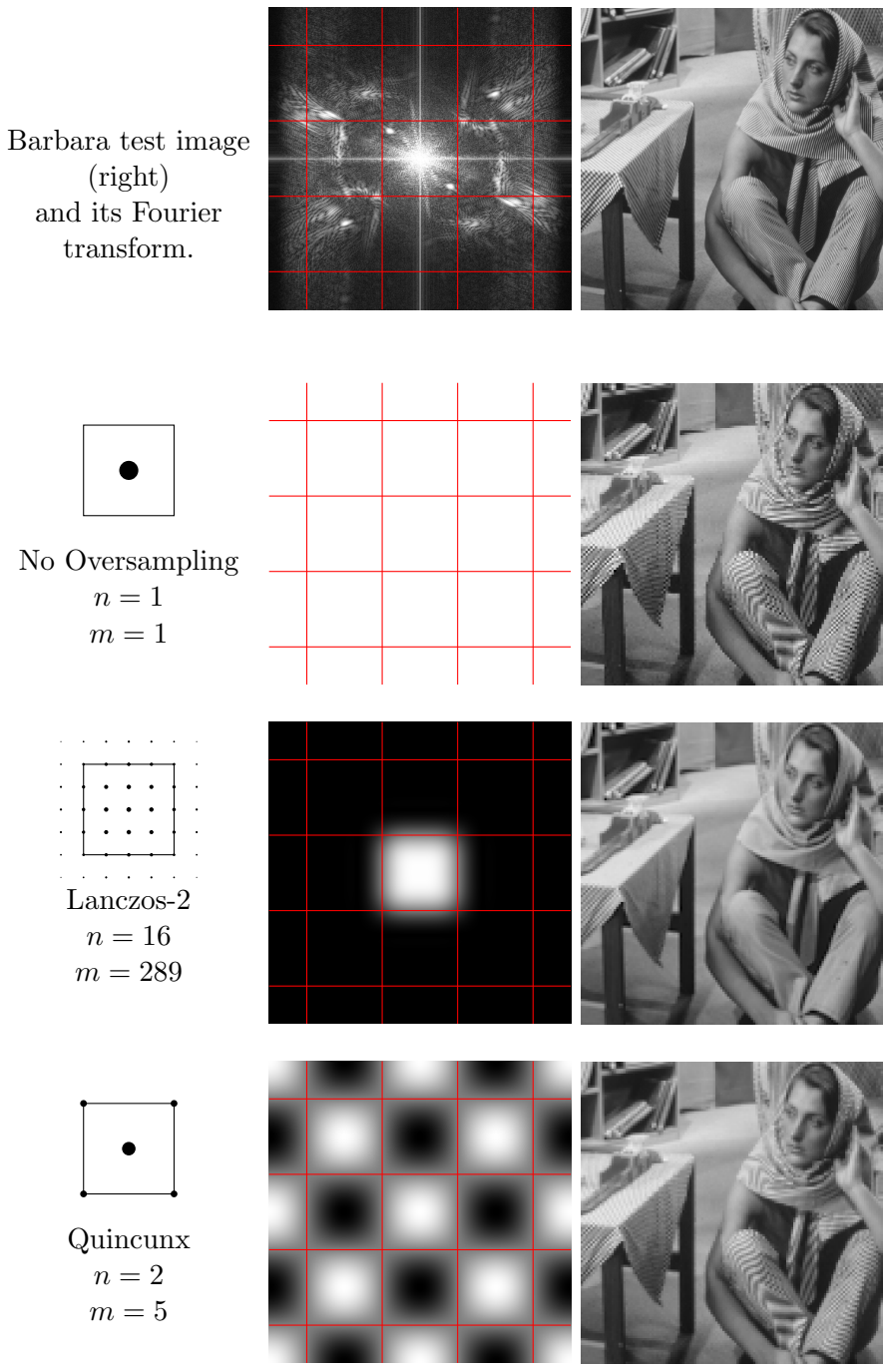


Figure 3.2: Visualization of the frequency behavior of several common 2D supersampling patterns. Continued in Fig. 3.3.

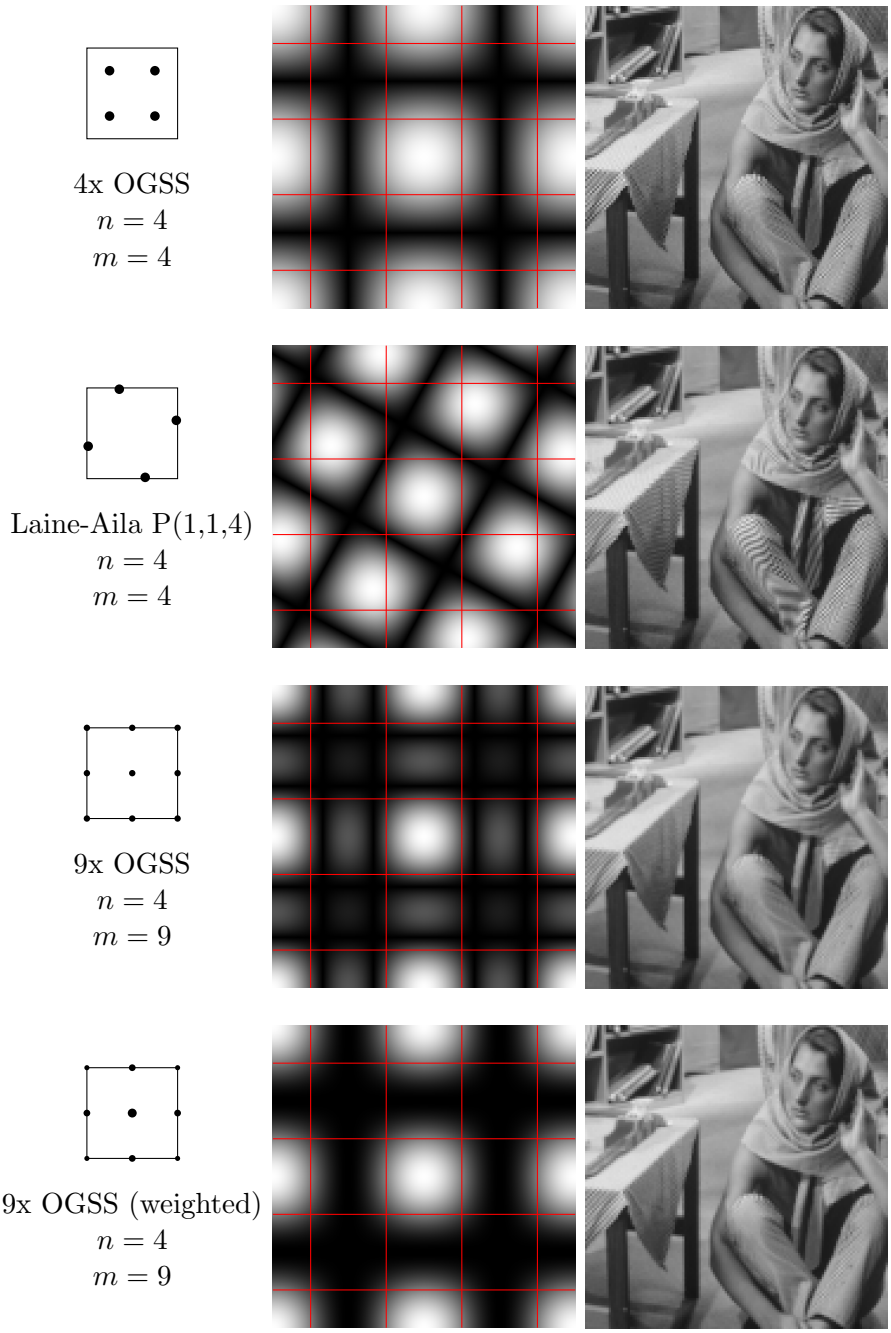


Figure 3.3: Visualization of the frequency behavior of several common 2D supersampling patterns.

the image by a factor of 4 in both directions, we know that there are no frequencies outside $[-4\nu_c, 4\nu_c]$. Since the stripes are predominantly vertical and diagonal, the image's Fourier transform contains a lot of energy in the blocks to the left and right and diagonal from the Nyquist region. We therefore expect supersampling patterns that don't attenuate these regions to cause visible moiré artifacts.

The worst result is obtained when using no supersampling at all, as shown in the second row. As expected, this results in strong moiré artifacts and jaggies. In contrast, almost optimal results are obtained by using a large number of samples per pixel and weights derived from a Lanczos-2 filter (row 3). Since we know that the highest frequency in the Barbara image is $4\nu_c$ we have used a sampling rate of $n = 16$ samples per pixel. The resulting supersampling pattern suppresses aliasing in the Barbara image almost completely.

The last row in Figure 3.2 shows the so-called Quincunx pattern mentioned earlier [NVIDIA, 2001]. It can be seen that it causes heavy moiré patterns. The main selling point of the Quincunx pattern is that only two samples have to be taken per pixel, since the three others can be shared with neighboring pixels. This is an important aspect for hardware implementations.

The first two rows in Figure 3.3 compare two sampling patterns with 4 samples per pixel. The first is a simple regular grid with $n = 2$. Even though the MTF shows a fair amount of energy outside of the Nyquist region, the results are acceptable for the resampling task since the filter's energy is concentrated in areas where the spectral energy of the test image is low. The second sampling pattern is one of the "optimal" sampling patterns determined by Laine and Aila's optimization approach [Laine and Aila, 2006]. The four samples form a rotated square. Since the MTF is high in the squares diagonal from the Nyquist region, heavy aliasing results for diagonal patterns, especially those in the trousers.

In the last two rows we show two regular grid patterns; in real-time graphics these are commonly referred to as *ordered-grid supersampling* (OGSS). The two sampling patterns differ only in the weights w_k: The first pattern uses constant weights whereas the second pattern derives its weights from the Lanczos-2 filter. The differences in the two examples are subtle, but the second pattern yields sharper result and less aliasing for horizontal and vertical patterns.

3.2 Fourier Analysis of Irregular Sampling

We now turn to the more difficult problem of describing the sampling process in the case of non-periodic, irregular sampling patterns. The mathematical analysis of irregular sampling is more challenging since individual sampling patterns cannot be expressed in closed form. It is possible, however, to to study the *overall* effect of irregular sampling by averaging over whole classes of sampling patterns. In graphics, this kind of analysis was pioneered by Yellot [1983] and Mitchell [1991]. This section collects and explains the relevant mathematical concepts. Since we are not aware of a clear exposition of these topics in the computer graphics literature, and since we will use the results extensively in the remainder of this thesis, we have chosen a relatively detailed exposition.

Similar to the description of regular sampling in Section 2.1, we can model irregular sampling as multiplication of the image $I(\boldsymbol{x})$ with the sampling pattern $S(\boldsymbol{x})$. To simplify the notation in the following, we will denote the sampled image by $J(\boldsymbol{x})$ instead of $I_s(\boldsymbol{x})$:

$$J(\boldsymbol{x}) = \sum_i I(\boldsymbol{x}_i)\delta(\boldsymbol{x} - \boldsymbol{x}_i) = I(\boldsymbol{x}) \cdot S(\boldsymbol{x}). \tag{3.6}$$

The sampling function S consists of Dirac peaks at the sample positions

$$S(\boldsymbol{x}) = \sum_i \delta(\boldsymbol{x} - \boldsymbol{x}_i) \tag{3.7}$$

and replaces the sampling comb $\mathrm{III}(\boldsymbol{x})$ from regular sampling.

3.2.1 Power Spectrum of Sampled Signals

To understand the effect of irregular sampling on the frequency distribution, we need to study the spectrum of the sampled image J. We model both the image being sampled and the sampling pattern as infinite signals; in this case, we can characterize the frequency content using the *power spectra* of the involved signals. The mathematical definition of the power spectrum and its most important properties are reviewed in Appendix A. Figure 3.4 shows three common sampling patterns and their power spectra.

The power spectrum of the sampled image P_J is obtained by convolving the power spectra of the sampling pattern P_S and the original image P_I:

$$P_J(\boldsymbol{\nu}) = P_I \star P_S(\boldsymbol{\nu}), \tag{3.8}$$

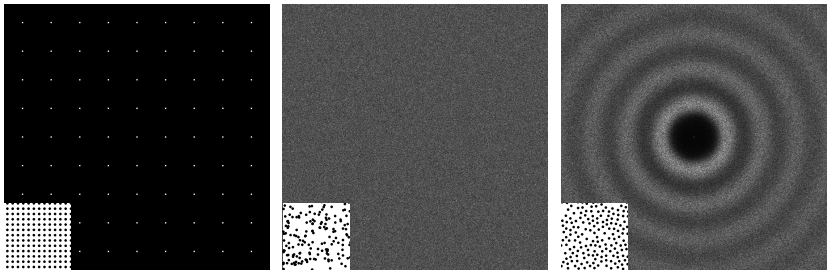


Figure 3.4: *(Top)* Empirical power spectra for several common sampling patterns. The random and dart throwing spectra were estimated from ten *periodograms* each.

provided the sampling pattern is statistically independent of the image I (see Eq. (A.5)). This relationship is visualized in Figure 3.5. As we have indicated in this figure, P_J always contains one copy of original spectrum P_I at the origin. The goal of the reconstruction process is to recover this copy by lowpass filtering.

Mathematically, the central copy of P_I is due to the non-zero mean of P_S: According to Eq. (A.6), the power spectrum of a signal with non-zero mean can be written as

$$P_X(\boldsymbol{\nu}) = |m_X|^2\delta(\boldsymbol{\nu}) + \check{P}_X(\boldsymbol{\nu}).$$

For a sampling pattern S, the mean m_S is the expected number of samples per unit area, i.e., the sampling density n. In this case, we therefore have $P_S(\boldsymbol{\nu}) = n^2\delta(\boldsymbol{\nu}) + \check{P}_S(\boldsymbol{\nu})$. (Conversely, $\check{P}_S$ is the standard power spectrum without the DC peak.)

This yields the main expression for the spectrum of a sampled signal as a sum of two components:

$$\boxed{P_J(\boldsymbol{\nu}) = n^2 P_I(\boldsymbol{\nu}) + P_I \star \check{P}_S(\boldsymbol{\nu})} \tag{3.9}$$

The first term, $n^2 P_I(\boldsymbol{\nu})$, is simply a scaled copy of the original spectrum. This term is independent of the actual sampling pattern and depends only on the sample density n. The spatial arrangement of the sample points affects only the second term $P_I \star \check{P}_S(\boldsymbol{\nu})$. Since this term reflects the distortion introduced by sampling, we refer to it as the *sampling error* $\epsilon(\boldsymbol{\nu})$

$$\boxed{\epsilon(\boldsymbol{\nu}) = \frac{1}{n^2} P_I \star \check{P}_S(\boldsymbol{\nu}).} \tag{3.10}$$

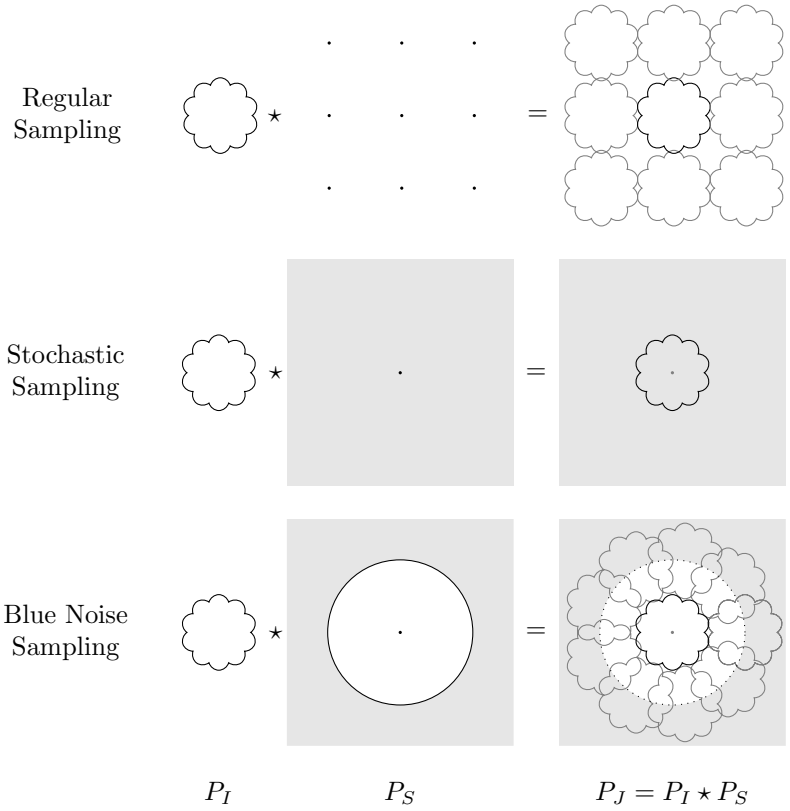


Figure 3.5: The spectrum of the sampled signal P_J is obtained by convolving the original spectrum P_I with the spectrum of the sampling pattern P_S. One copy of P_I is always placed at the origin, which is due to the first term in Eq. (3.9) Aliasing occurs when the surrounding copies of P_I overlap the central spectrum. The illustrated power spectra are the same as in Figure 3.4.

(The $1/n^2$ term is a normalization factor which ensures that the magnitude of ϵ is comparable to the magnitude of P_I.) Note that $\epsilon(\boldsymbol{\nu})$ is not an absolute error but a function of frequency. We are primarily interested in the low-frequency region of $\epsilon(\boldsymbol{\nu})$ which measures the amount of aliasing that reaches into the central copy of the image spectrum. Recovering P_I from P_J by lowpass filtering is only possible if P_I and ϵ do not overlap; otherwise the resulting image will contain aliasing. The functional shape of $\epsilon(\boldsymbol{\nu})$ determines the spectral composition of this aliasing. We will illustrate this in the following sections.

3.2.2 Simple Sampling Patterns

We can derive and study the sampling error $\epsilon(\boldsymbol{\nu})$ analytically if either the sampled signal or the sampling pattern has a simple mathematical form. In this section we discuss three cases in which P_S can be expressed in closed form.

Regular Sampling In the case of regular sampling with spacing T, the sampling pattern is $S(\boldsymbol{x}) = \mathrm{III}_T(\boldsymbol{x})$ and the corresponding power spectrum is also a comb function

$$P_{\text{regular}}(\boldsymbol{\nu}) = \frac{1}{T^2 T^2}\,\mathrm{III}_{1/T}(\boldsymbol{\nu}) = n^2\,\mathrm{III}_{1/T}(\boldsymbol{\nu}). \tag{3.11}$$

This follows from the fact that the Fourier transform of III_T is $\mathrm{III}_{1/T}$ (see [Mallat, 2009, Theorem 2.4] for a proof), and the relation $n = 1/T^2$ for the sample density of a regular grid.

The sampling error therefore consists of replicates of P_I placed at the regular grid $\mathrm{III}_{1/T}$ except at the origin

$$\epsilon_{\text{regular}}(\boldsymbol{\nu}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^2\setminus\{0\}} P_I(\boldsymbol{\nu} - \boldsymbol{k}/T) \tag{3.12}$$

This is the first case illustrated in Figure 3.5: as long as the copies of P_I do not overlap, no aliasing can occur. This is exactly the case if the bandwidth of X is less than $1/2T$, which, of course, is simply the classical sampling theorem.

Note that all copies of P_I have the same strength as the central copy, so the strength of aliasing is comparable to the original signal. This is the reason regular sampling is primarily useful in situations in which aliasing is impossible or can be prevented by other means.

Stochastic Sampling If the sample positions S are chosen completely at random, we are dealing with *stochastic sampling*, which was proposed for sampling in graphics by Dippé and Wold [1985] and Cook [1986]. Mathematically, stochastic sampling can be modeled as a *Poisson process*, whose autocorrelation and power spectrum are given by

$$C_{\text{Poisson}}(\tau) = n\delta(\tau) + n^2, \qquad P_{\text{Poisson}}(\nu) = n^2\delta(\nu) + n. \tag{3.13}$$

(See [Daley and Vere-Jones, 2002].) Combining this with Eq. (3.10) we obtain

$$\epsilon(\boldsymbol{\nu}) = \frac{1}{n^2} n \star P_I(\boldsymbol{\nu}) = \frac{1}{n}\int_{\mathbb{R}^2} P_I(\boldsymbol{\nu})\,d\boldsymbol{\nu} = \frac{P_I}{n}.$$

The symbol P_I denotes the *total power* of the image I. This tells us that the sampling error is constant for all image frequencies, so the noise has white noise characteristics—in other words, stochastic sampling adds a constant "noise floor" to the sampled image. This is the main advantage of stochastic sampling: since the sampling pattern has no structure at all, aliasing always has the appearance of broadband noise, and moiré patterns cannot occur.

The primary disadvantage of stochastic sampling is that the sampling error is non-zero for all frequencies, so even low image frequencies are contaminated with noise. Perfect reconstruction therefore isn't possible for *any* input signal with reconstruction methods based on lowpass filtering.

A second disadvantage is the slow decrease of the sampling error as we increase the sampling rate. Since $\epsilon(\boldsymbol{\nu})$ measures the squared error, the absolute error only decreases as $1/\sqrt{n}$. (This is to be expected, since this is also the convergence rate of Monte Carlo integration.) We therefore need four times the number of samples to halve the error; in practice, this means that we require an inordinate number of samples to get rid of all noise.

Jittered Sampling Jittered sampling derives its name from the process used to construct the sampling pattern. The starting point is a regular sampling pattern $S = \{\boldsymbol{x}_i\}$, the points of which are "jittered" by displacing them by a random offset

$$\boldsymbol{x}_i' = \boldsymbol{x}_i + \boldsymbol{U},$$

where $\boldsymbol{U}$ is a random variable with a certain distribution. In the most common form of jittered sampling, $\boldsymbol{U}$ is chosen randomly from the square $[-T/2, T/2)^2$. This form of jittered sampling is also called *stratified sampling*, since it is equivalent to subdividing the image plane into square "strata" of size $T \times T$ and placing one random sampling point into each of them.

The power spectrum of a jittered sampling pattern can be written in closed form [Dippé and Wold, 1985, Brémaud et al., 2005]

$$P_{\text{jitter}}(\boldsymbol{\nu}) = |\hat{\mathbb{P}}_U(\boldsymbol{\nu})|^2 P_S(\boldsymbol{\nu}) + n\big(1 - |\hat{\mathbb{P}}_U(\boldsymbol{\nu})|^2\big), \tag{3.14}$$

where $\mathbb{P}_U$ is the probability density of U and $\hat{\mathbb{P}}_U$ its Fourier transform. P_S denotes the spectrum of the original unjittered sampling pattern.

In the case of stratified sampling, the probability density of U is

$$\mathbb{P}_U(\boldsymbol{x}) = n \cdot \text{rect}_T(\boldsymbol{x}), \qquad \hat{\mathbb{P}}_U(\boldsymbol{\nu}) = \text{sinc}_{1/T}(\boldsymbol{\nu}),$$

and the first term in Eq. (3.14) simplifies to $|\hat{\mathbb{P}}_U(\boldsymbol{\nu})|^2 P_S(\boldsymbol{\nu}) = n^2\delta(\boldsymbol{\nu})$ since for

all $\boldsymbol{\nu} \neq 0$ one of the factors is always zero. We therefore have

$$P_{\text{stratified}}(\boldsymbol{\nu}) = n^2\delta(\boldsymbol{\nu}) + n\big(1 - |\operatorname{sinc}_{1/T}(\boldsymbol{\nu})|^2\big). \tag{3.15}$$

For all $\boldsymbol{\nu} \neq 0$ we have $P_{\text{stratified}} \leq n = P_{\text{stochastic}}$, which means that stratified sampling is always at least as good as stochastic sampling. (Again, a similar result is also known in the theory of Monte Carlo integration [Veach, 1997].) This guarantee, combined with the ease of constructing jittered sampling patterns on the fly, is one of the main reasons for their continued popularity in graphics.

3.3 Blue Noise Sampling

The previous section focused on examples of irregular sampling in which the power spectrum of the sampling pattern P_S could be expressed analytically. In this section we study the converse situation in which the spectrum of the image P_I has a closed-form expression. This helps us to get a better understanding of blue noise sampling—in particular how the shape of the power spectrum affects the visual appearance of aliasing. This section generalizes results by Dippé and Wold [1985] who primarily studied the effect of irregular sampling on constant signals.

3.3.1 Irregular Sampling of Constants

In some cases the image spectrum can be written in closed form. The simplest example is a constant image with average power α:

$$I_{\text{const}}(\boldsymbol{x}) = \alpha, \qquad P_{\text{const}}(\boldsymbol{\nu}) = \alpha^2\delta(\boldsymbol{\nu})$$

In this case, the sampling error evaluates to

$$\epsilon(\boldsymbol{\nu}) = \frac{\alpha^2}{n^2}\check{P}_S \star \delta(\boldsymbol{\nu}) = \frac{\alpha^2}{n^2}\check{P}_S(\boldsymbol{\nu}). \tag{3.16}$$

In other words: For constant functions, the sampling error is proportional to the power spectrum of the sampling pattern. This is the "flat field response" studied by Dippé and Wold [1985].

If the sampling pattern has a blue noise spectrum, like the dart throwing pattern in Figure 3.4, the sampling error is concentrated in high frequencies, whereas the low-frequency region of the sampled signal is clean. This observation is the source of the widely held misconception that "blue noise sampling

shifts aliasing into the high frequencies". While this is true in the case of constant images, we will see in the next section that low-frequency aliasing is very well possible for certain signals, even when using an irregular sampling pattern.

3.3.2 Irregular Sampling of Sinusoidals

A second, more general class of signals that can be expressed in closed form are plane waves of the form

$$I_{\boldsymbol{f}}(\boldsymbol{x}) = \alpha \cdot \cos(2\pi \boldsymbol{f} \cdot \boldsymbol{x}). \tag{3.17}$$

(For $\boldsymbol{f} = 0$ we obtain the constant signals from the previous example.) Studying signals of this shape is especially interesting since they are periodic, so they help us understand how moiré patterns occur and how they are influenced by the power spectrum of the sampling pattern.

The power spectrum corresponding to such an image $I_{\boldsymbol{f}}$ is

$$P_{\boldsymbol{f}}(\boldsymbol{\nu}) = \frac{\alpha^2}{4}\big(\delta(\boldsymbol{\nu} - \boldsymbol{f}) + \delta(\boldsymbol{\nu} + \boldsymbol{f})\big) \tag{3.18}$$

and the sampling error ϵ follows directly from Eqs. (3.10) and (3.17)

$$\epsilon_{\boldsymbol{f}}(\boldsymbol{\nu}) = \frac{\alpha^2}{4n^2}\big(\check{P}_S(\boldsymbol{\nu} - \boldsymbol{f}) + \check{P}_S(\boldsymbol{\nu} + \boldsymbol{f})\big). \tag{3.19}$$

This equation tells us that the distortion due to sampling is proportional to the sum of two shifted copies of $\check{P}_S$, the power spectrum of the sampling pattern. The spectral distribution of the sampling error ϵ depends on the way the two copies of $\check{P}_S$ overlap.

3.3.3 Blue Noise Sampling

What this means in practice is illustrated in Figure 3.6. The first column shows the spectrum of the signal $P_{\boldsymbol{f}}$ for different frequencies $\boldsymbol{f}$, which consists of two Dirac peaks at distance $2f$. We focus on the results shown in the last column, which illustrates how the sampling error of a blue noise sampling pattern varies with the signal frequency $\boldsymbol{f}$. We have assumed an idealized blue noise spectrum as in Figure 3.5 with a zero-region, a broad peak at the transition (the black ring), and a constant level in high frequencies.

For $f = 0$ and $f = 0.1$, the low-frequency region remains empty, so the sampled signal can be reconstructed exactly using a lowpass filter. For $f = 0.3$

the non-zero parts of P_S start overlapping the original spectrum, which will manifest as high-frequency noise in the resampled image. The least favorable results are obtained for $f = 0.5$: The peaks in the power spectrum P_S overlap close to the origin, which results in low-frequency noise. Above $f = 0.7$, the sampling error has the appearance of broadband noise.

To visualize the appearance of low-frequency noise at certain image frequencies, we sample the so-called *zone plate* using different blue noise sampling patterns. The zone plate is a synthetic image $I_{\text{zone}}(\boldsymbol{x})$ which oscillates more rapidly as $|\boldsymbol{x}|$ increases

$$I_{\text{zone}}(\boldsymbol{x}) = \frac{1}{2}\big(\cos(|\boldsymbol{x}|^2) + 1\big). \tag{3.20}$$

In fact, the local frequency of I_{zone} is proportional to $\boldsymbol{x}$, so aliasing artifacts in the zone plate can be easily related to spectral properties of the sampling pattern.

If we could prefilter this signal correctly before sampling, the high-frequency region would appear as flat gray. Without perfect filtering, the best we can hope for is white, unstructured noise in the high frequencies. Instead, as can be seen in Figure 3.7, peaks in the power spectrum P_S result in low-frequency noise with a random, "blotchy" appearance. Even though this low-frequency noise does not form regular geometric patterns as moiré artifacts do, its low frequency and high contrast make it undesirable for the same reasons.

The above observations about the influence of P_S when sampling a periodic signal with frequency $\boldsymbol{f}$ can be summarized as follows:

- If $P_S(\boldsymbol{\nu})$ is flat in the neighborhood of $\boldsymbol{f}$, the error $\epsilon_{\boldsymbol{f}}$ is also flat around the origin, i.e., the sampling error is rendered as broadband or white noise.
- If $P_S(\boldsymbol{\nu})$ has a peak around $\boldsymbol{f}$, copies of this peak overlap at the origin of $\epsilon_{\boldsymbol{f}}$. The width of this peak then determines the appearance of this aliasing: A broad peak leads to broadband noise, whereas a narrow peak produces low-frequency aliasing that appears as structured noise. The height of the peak determines the variance and therefore the visibility of the aliasing.

3.3.4 Measuring Blue Noise

We have discussed in the previous section how the shape of the power spectrum influences the spectral distribution of the sampled image. Blue noise patterns therefore aim to achieve a good tradeoff between the following demands:

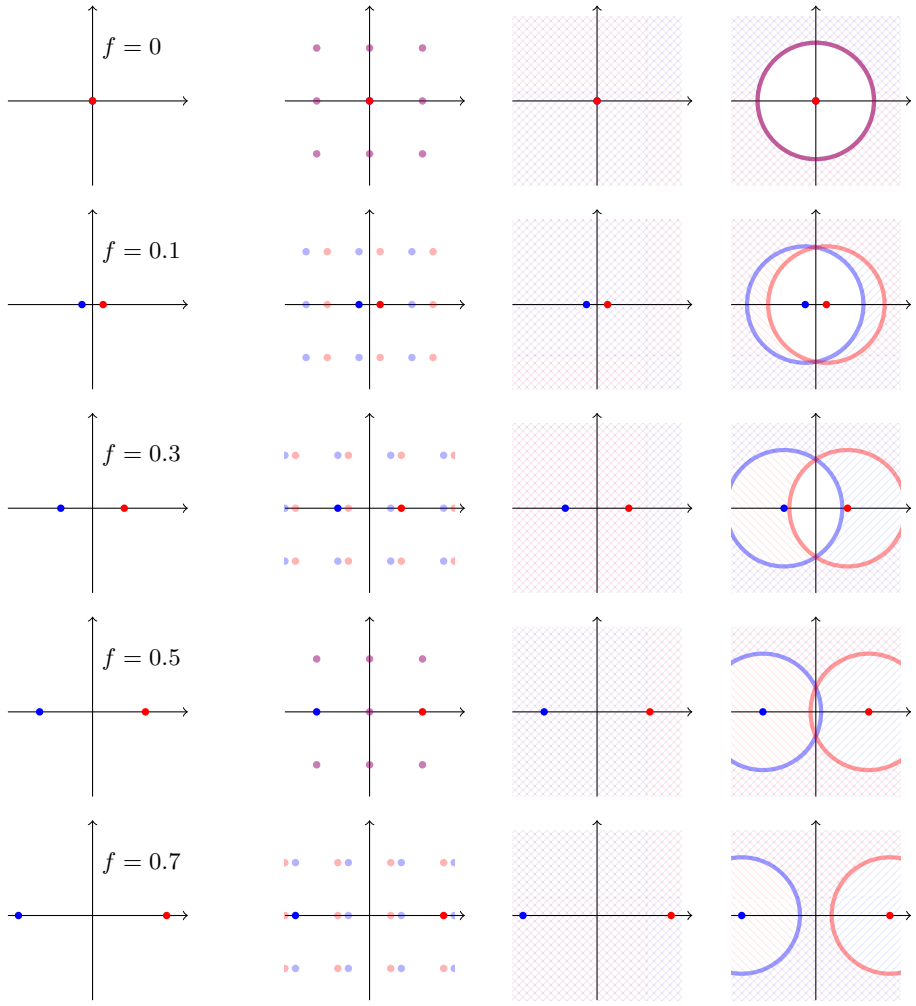


Figure 3.6: Illustration of the sampling error $\epsilon(\boldsymbol{\nu})$ when sampling sinusoidals of different frequencies f. The first column shows the spectrum of the signal which consists of two Dirac peaks (red and blue) at distance $2f$. The remaining columns illustrate the sampling error for different sampling patterns. *(Regular Sampling)* The sampling error consists of Dirac peaks distributed in the frequency plane; aliasing appears at discrete frequencies, which can lead to moiré patterns. *(Stochastic Sampling)* The sampling error is independent of the signal frequency and always appears as a constant layer of white noise. *(Blue Noise Sampling)* For low signal frequencies, an aliasing-free low frequency region remains. At $f = 0.5$, aliasing takes the form of low-frequency noise since the rings of the spectrum overlap near the origin.

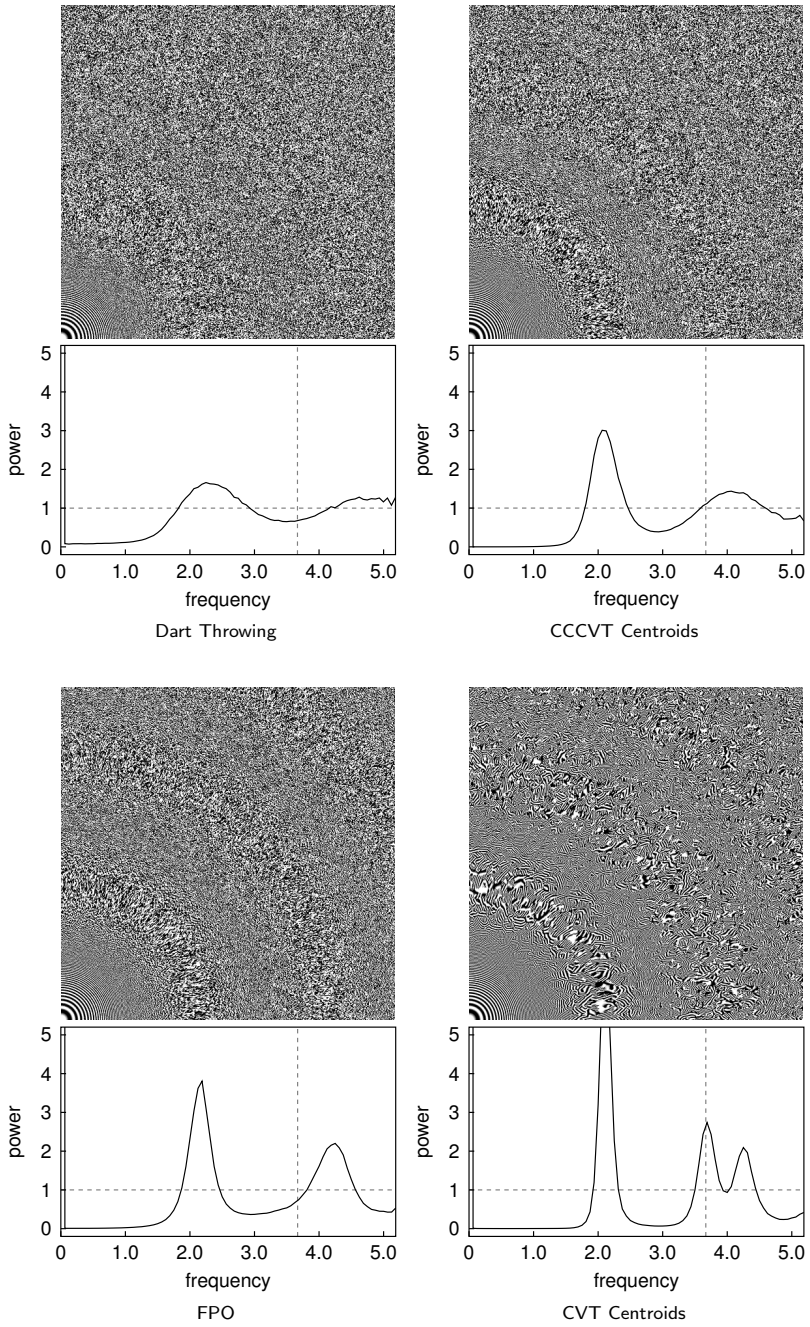


Figure 3.7: A bird's eye view of a sampled zone plate reveals low-frequency aliasing for blue noise patterns that have peaks in their power spectrum. Ideally, we would like to turn all aliasing into featureless white noise. Instead, high and narrow peaks in the spectrum lead to high-contrast, structured noise, not unlike the moiré patterns of regular sampling.

- The *low-frequency region* of P_S should contain as little energy as possible so that low image frequencies can be sampled without aliasing. Obviously, this region should also be as wide as possible to allow error-free reconstruction for wide range of frequencies.

- The *high-frequency region* of P_S should be flat so that high image frequencies are mapped to broadband noise. Peaks in the high-frequency region should be as flat and as wide as possible to prevent high-contrast, low-frequency aliasing in the sampled image.

In this section we introduce two numerical measures that characterize the shape of the power spectrum: the *effective Nyquist frequency* ν_{eff} measures the size of the zero region and indicates the range of frequencies that can be represented with almost no aliasing and the *oscillation* Ω measures the amount of oscillation in the power spectrum, and therefore the risk of low-frequency, structured aliasing.

In the theory of uniform sampling, the range of frequencies that can be reconstructed without aliasing is given by the Nyquist frequency $\nu_c = 1/2T$, where T is the sample spacing. This direct relationship between sample distances and frequencies that can be reconstructed unfortunately breaks down for non-uniform sampling. As an alternative, we define the equivalent of the Nyquist frequency directly in the frequency domain. Figure 3.8 illustrates the definition of the effective Nyquist frequency ν_{eff}. The visualization of blue noise sampling in Figure 3.8 (a) suggests that the range of aliasing-free frequencies roughly equals half the radius of the zero-region in the power spectrum.

Actually measuring this radius is not quite as easy as the illustration suggests because we are dealing with stochastic sampling patterns, and the spectrum is therefore never exactly zero. We therefore define the effective Nyquist frequency as follows. We consider the average energy in the power spectrum up to a certain frequency ν

$$P_{\text{avg}}(\nu) = \frac{1}{\pi\nu^2} \int_{|\boldsymbol{\nu}'|<\nu} P(\boldsymbol{\nu}') \, \mathrm{d}\boldsymbol{\nu}'$$

and define the *effective Nyquist frequency* ν_{eff} as the largest frequency so that $P_{\text{avg}}(2\nu_{\text{eff}})$ stays below a given threshold

$$\nu_{\text{eff}} = \max\{\nu : P_{\text{avg}}(2\nu) \leq E_\tau\}. \tag{3.21}$$

Ideally, we would like to set $E_\tau = 0$ to determine which frequencies can be represented without *any* aliasing, but this is impractical since stochastic sampling

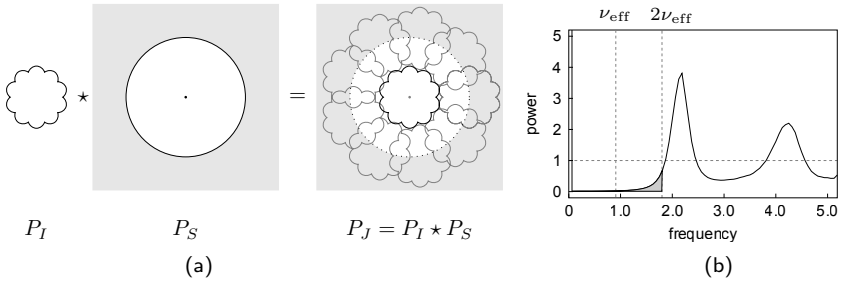


Figure 3.8: *(a)* The spectrum of the sampled signal P_J is obtained by convolving the original spectrum with the spectrum of the sampling pattern. Aliasing occurs when the replicated spectra P_I overlap the central spectrum on the right-hand side. *(b)* The effective Nyquist frequency is a measure for the size of the zero region in the power spectrum. It is chosen so that $2\nu_{\text{eff}}$ roughly corresponds to the radius of the circle representing P_S in (a).

patterns always introduce some noise into the low-frequency region. Our reference for choosing E_τ is dart throwing, which has more noise in low-frequency region than other blue noise patterns. We chose $E_\tau = 0.1$, which is the lowest threshold for which dart throwing with a dart throwing radius of $d_{\min} = 0.75$ (see Section 4.1.1) consistently yields a non-zero value for ν_{eff}.

To measure the amount of oscillation of the power spectrum, we use the standard deviation of $P(\boldsymbol{\nu})$ from the 1-level

$$\Omega = 10 \left(\frac{1}{|R|} \int_R |P(\boldsymbol{\nu}) - 1|^2 \, \mathrm{d}\boldsymbol{\nu} \right)^{1/2}.$$

Here, R is the integration domain and $|R|$ its area. We exclude the zero-region of the power spectrum by integrating over the ring $R = \{\boldsymbol{\nu} : \nu_0 \leq |\boldsymbol{\nu}| \leq \nu_1\}$. The inner radius ν_0 is the lowest frequency for which $P(\nu_0) = 1$; the outer radius $\nu_1 = \nu_0 + 10\nu_{\text{hex}}$ is chosen to cover approximately the first 10 peaks in the power spectrum.

The effective Nyquist frequency is similar to the conventional Nyquist frequency in the sense that frequencies below ν_{eff} can be sampled and reconstructed with little error. Frequencies above ν_{eff}, on the other hand, are replaced by aliasing. In this case, the magnitude of Ω determines whether this aliasing is, on average, closer to white noise (Ω small) or colored noise (Ω large).

Table 3.1 lists the values of ν_{eff} and Ω for several classes of sampling patterns used in graphics. It is obvious that a larger zero region (high ν_{eff})

Method	ν_{eff}	Ω	d_{avg}
Random	0	0.05	0.47
Jittered Grid	0.24	0.06	0.59
Dart Throwing	0.58	1.52	0.80
Kernel Density	0.88	2.14	0.86
CCCVT Centroids	0.89	2.34	0.88
El. Halftoning	0.89	2.49	0.88
Regular grid	0.95	14.77	0.93
CVT Centroids	0.98	5.35	0.94
Hexagonal grid	1.01	12.38	0.99

Table 3.1: The effective Nyquist frequency and oscillation for a selection of common sampling patterns in graphics. For reference, we have also included d_{avg}, the average distance from a point to its nearest neighbor. Compare also the detailed information in Appendix B.

correlates with more oscillation in the high-frequency region. This interdependence was sometimes referred to as the *noise-aliasing tradeoff* Dippé and Wold [1985], Glassner [1995]. Note that practical values for ν_{eff} are slightly larger than theoretical values due to the tolerance introduced by E_τ. The practical limit for irregular sampling patterns seems to be around $\nu_{\text{eff}} \approx 0.9$. Note also that for the patterns listed in this table, the effective Nyquist frequencies increases as we increase the spacing between the points, which we measure using d_{avg}, the average nearest-neighbor distance. This observation is the starting point for our study of Poisson disk sampling in the following chapter.

3.4 Discussion

The way in which the sampling pattern affects the visual appearance of aliasing is easy to describe using the power spectrum, but the details of the analysis aren't widely understood in graphics—despite the fact that the main formulas, namely Eqs. (3.8) and (3.9), were already used in one of the earliest publications on irregular sampling in graphics [Dippé and Wold, 1985]. The most common misunderstanding, which is repeated by almost every paper on blue noise sampling published during the last years, is that blue noise sam-

pling replaces aliasing with high-frequency noise.[1] In reality, the goal of blue noise sampling is to replace aliasing with *broadband* noise while keeping low frequencies as clean as possible. This improves the perceived image quality in undersampled images since broadband noise is visually more neutral than low-frequency artifacts. Compared to stochastic sampling, blue noise sampling guarantees the correct reconstruction of low frequencies. This is particular relevant for natural images, which are dominated by low-frequency content [Burton and Moorhead, 1987].

Obviously, the *perfect* blue noise sampling pattern would have an extremely wide zero region and be completely flat in the high-frequency region. This would guarantee that the sampling error is zero for low frequencies and featureless white noise for high frequencies. How close can we get to this ideal? We will discuss this problem in detail in Chapter 5, where we will see that such step-like power spectra are in fact realizable, but their zero region is noticeably smaller than that of conventional blue noise patterns. So some oscillation in the power spectrum seems to be unavoidable if we want to increase the effective Nyquist frequency above a certain limit, but the details aren't understood so far.

.

[1] If this were the case, then removing *all* aliasing would be a simple matter of low-pass filtering the sampled signal!

Chapter 4

Irregular Sampling with Maximized Spacing

As we have seen in Table 3.1, a wide separation of the sample points correlates with better blue noise properties in the sense that the effective Nyquist frequency becomes higher. But what happens if we spread out the points even further? Is this even possible?

For a long time, this question could not be answered due to the practical difficulty of generating irregular point distributions with a high separation. The few methods that were capable of spreading out the points efficiently, like Lloyd's method, also converged towards regular patterns. This is not a coincidence: the most efficient packing of disks in the plane is a hexagonal grid, and it is difficult to formulate optimization schemes that consistently converge towards the "suboptimal" results that are desirable for irregular sampling. All the methods that achieve $d_{\text{avg}} > 0.8$ in Table 3.1 are recent optimization methods that designed to retain some irregularity in the final point set [Balzer et al., 2009, Fattal, 2011, Schmaltz et al., 2010].

Contributions: In this chapter we present a new optimization scheme that spreads out point in the plane without converging towards regular arrangements. The resulting point sets achieve extremely high nearest-neighbor distances, but in contrast to other methods the results remain irregular and isotropic. We analyze the spectral behavior of the resulting point sets to study their suitability for sampling applications. Even though they are isotropic and therefore show no rotatinoal order, it turns out that the strong constraint on the minimal distance leads to a noticeable *translational order*, which can lead to visible aliasing artifacts. This chapter is based on [Schlömer et al., 2011].

4.1 Geometric Measures of Uniformity

The key to constructing blue noise patterns geometrically is finding geometric properties of point distributions that correlate with a blue noise spectrum. Most research has focused on the empirical observation that the following two properties usually yield a blue noise spectrum:

1. *Uniformity.* The points are distributed uniformly in the sense that sample density is approximately equal everywhere.

2. *Irregularity.* The point set has little translational or rotational symmetries. This also implies *isotropy*.

These conditions also make intuitive sense for sampling applications: Uniformity implies a good coverage of the sampling domain, and irregularity prevents moiré patterns in the sampled image. It is important note that there is no one-to-one correspondence between the above geometrical criteria and good sampling properties, as we will see in this chapter and the next. In most algorithms for constructing sampling patterns, uniformity is the primary constraint, and irregularity is introduced by some form of randomization.

As a simple characterization of uniform, well-distributed points, we demand that the point set contains neither clusters (regions where points clump up) nor holes (regions that are empty). This section discusses two simple but widespread geometric measures to quantify this idea of uniformity.

4.1.1 Nearest-Neighbor Distance

The simplest and most common measures for the uniformity of point sets are based on the pairwise distance between points. Given a set of points S, the *nearest-neighbor distance* (NND) of any point $\boldsymbol{x} \in S$ is the Euclidean distance to its nearest neighbor in S. We denote this distance by $d_{\min}(\boldsymbol{x})$:

$$d_{\min}(\boldsymbol{x}) = \min_{\boldsymbol{x}' \in S} |\boldsymbol{x} - \boldsymbol{x}'| \tag{4.1}$$

We can derive two summary statistics from the NND:

$$d_{\min} = \min_{\boldsymbol{x} \in S} d_{\min}(\boldsymbol{x}), \qquad d_{\text{avg}} = \mathbb{E}[d_{\min}(\boldsymbol{x})]. \tag{4.2}$$

We call $d_{\min}$ the *global NND* because it measures the minimal separation between any two points in S and d_{avg} the *average NND* because it measures the expected distance from a point to its nearest neighbor.

Since the nearest-neighbor distance depends on the density of points n, it is customary to normalize it as follows

$$d_{\min}(\boldsymbol{x}) = \frac{\min_{\boldsymbol{x}' \in S} |\boldsymbol{x} - \boldsymbol{x}'|}{d_{\text{hex}}}, \qquad d_{\text{hex}} = \left(\sqrt{3}/2n\right)^{1/2}. \tag{4.3}$$

The normalization factor d_{hex} denotes the spacing in a hexagonal lattice at point density n, so we always have $0 \leq d_{\min} \leq d_{\text{avg}} \leq 1$.

The NND has a long history as a measure for characterizing point distributions, both in graphics and outside [Clark and Evans, 1954]; it is also a common measure for point distributions used in quasi-Monte-Carlo integration [Grünschloß et al., 2008]. In graphics, an irregular point set with NND $d_{\min}$ corresponds to a Poisson disk pattern with radius $R = d_{\min}/2$: if any two points are at least $d_{\min}$ apart, disks of radius R placed at the points cannot overlap and vice versa.

By prescribing a minimum distance $d_{\min}$ on a set of points, we impose a certain level of uniformity on the point set since we prevent groups of points from clustering up. Point sets with $d_{\min} < 0.8$ are relatively easy to generate, and even rejection sampling (the well-known "dart throwing" approach) works very well. Higher values of $d_{\min}$ are considerably more difficult to achieve and often require optimization approaches or iterative algorithms. For a long time, the only known point sets with $d_{\min} > 0.85$ suffered from regularity artifacts since they were based on optimization methods that converge to hexagonal structures. Lagae and Dutré [2008] even conjectured that this is a necessary consequence of a high NND and recommended the range $0.65 \leq d_{\min} \leq 0.8$ for obtaining a good tradeoff between irregularity and uniformity. We will return to this conjecture later in this chapter.

For comparing different point distributions, the average NND d_{avg} is a more useful quantity than $d_{\min}$ since it isn't affected by individual outliers. In general, d_{avg} measures how well the points in S are spread out. If d_{avg} is low, the points are clustered and the local point density fluctuates; if d_{avg} is high, the points are spread out and the point density is more homogeneous.

4.1.2 Coverage Radius

In addition to the absence of point clusters, we also expect uniform distributions to have no "holes". One measure to quantify the largest hole in a point distribution is the *coverage radius* [Mitchell et al., 2012].

If we are given a point set S and place a disk of radius r on every point $x \in S$, the coverage radius is the smallest r so that the disks cover the whole

domain, i.e.,

$$R_c = \min\{r : \mathbb{R}^2 \subset \cup_{\boldsymbol{x}\in S} D_r(\boldsymbol{x})\}, \tag{4.4}$$

where $D_r(\boldsymbol{x})$ denotes a disk of radius r centered at $\boldsymbol{x}$. Obviously, the coverage radius grows with the largest hole in the point set. Like the NND, we normalize R_c to the point density by dividing by d_{hex}.

The coverage radius is easy to calculate using the Voronoi diagram of S. For each point $\boldsymbol{x} \in S$, the Voronoi cell $V_{\boldsymbol{x}}$ consists of all points that are closer to $\boldsymbol{x}$ than to any other point. To cover $V_{\boldsymbol{x}}$ completely with the smallest possible disk $D_r(\boldsymbol{x})$ centered at $\boldsymbol{x}$, the radius r must be set to the largest distance from $\boldsymbol{x}$ to the corners of $V_{\boldsymbol{x}}$. This is the local coverage radius. Since the Voronoi partitions the plane, the global coverage radius R_c is the maximum of all local radii.

There is an interesting alternative interpretation of the coverage radius that we will return to later in this chapter. The *largest empty circle* of S is the largest circle that does not contain any point from S. The center of this largest empty circle is also called the *farthest point* of S, and its radius is equal to the coverage radius R_c, since any set of disks with a smaller radius would, by definition, not cover the farthest point.

A few algorithms for generating blue noise patterns have been proposed that take the coverage radius into account.

The *farthest point strategy* by Eldar et al. [1997] is an efficient, non-iterative technique for generating Poisson disk patterns. Given a few randomly distributed seed points, Eldar's algorithm deterministically adds new points to a given set by inserting them at the farthest point of the current points. The algorithm we will propose in the next section extends this idea.

Gamito and Maddock [2009] and Ebeida et al. [2011, 2012] consider a related problem. Their goal is to construct *maximal Poisson disk* patterns. The constructed point set is a Poisson disk pattern with a prescribed radius $R = d_{\min}/2$, and it is maximal in the sense that no additional point can be added to the set without violating the Poisson disk criterion. In other words, in a maximal Poisson disk pattern the largest empty circle must have a diameter smaller than R, or $R_c < d_{\min}$. (As can be seen in Table 4.2, several sampling patterns proposed in graphics are maximal in this sense, although this is rarely formally proven.) The relationship between the coverage radius, the nearest-neighbor distance and the farthest point is illustrated in Figure 4.1.

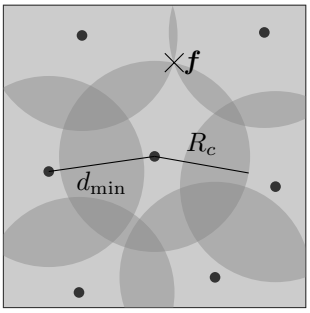


Figure 4.1: Illustration of the coverage radius. The square cannot be covered with smaller disks because this would leave the farthest point $\boldsymbol{f}$ uncovered. In this case, it is obvious that $R_c < d_{\min}$, so the point distribution is *maximal*.

4.2 Farthest-Point Optimization

This section presents a novel algorithm for generating Poisson disk patterns which we call *farthest point optimization* (FPO). Our algorithms builds on the *farthest point strategy* mentioned in the previous section, but instead of inserting *new* points at the position of the farthest point, we iteratively move *existing* points to the farthest point. We will see that this produces irregular point sets with an extremely high NND that are also maximal.

4.2.1 Main Algorithm

The main idea behind our algorithm is to iteratively increase the spacing of a given point set. Each step takes a single point from a set of points X and attempts to move it to a new position that is as far away from the remaining points as possible, i.e., the *farthest point*. One full iteration consists of moving each point in X once. As we will see, this iteration scheme converges, and each full iteration increases d_{avg}.

Geometrically, the farthest point f_Y of a set of points Y is the center of the largest circle that can be placed in the domain under consideration without covering any of the points in Y. This largest empty circle can be computed efficiently using the Delaunay triangulation $\mathcal{D}(Y)$: it corresponds to the largest circumcircle of the triangles in $\mathcal{D}(Y)$. An equivalent formulation in terms of the Voronoi diagram of Y was used by Eldar et al. [1997].

In our case, to move a point x, we need to inspect the Delaunay triangulation (DT) of the remaining points $X\backslash\{x\}$. Instead of calculating the full DT for each point x, we build a full DT once and update it dynamically during the iteration: before we move x, we remove it from the DT, inspect the remaining

```
FARTHEST-POINT-OPTIMIZATION(X)
 1  D = DELAUNAY(X)
 2  repeat
 3       foreach vertex x in D
 4            (f, r_max) = (x, d_min(x))
 5            DELAUNAY-REMOVE(D, x)
 6            foreach t in D
 7                 (c, r) = center and radius of t's circumcircle
 8                 if r > r_max
 9                      (f, r_max) = (c, r)
10            DELAUNAY-INSERT(D, f)
11  until converged
12  return vertices of D
```

Figure 4.2: A naive implementation of *farthest point optimization*.

triangles to find the farthest point f, and finally reinsert f as a new point into the DT. The full algorithm can be formulated as shown in Figure 4.2.

We make sure that a point is only moved to a new position if its new local NND, namely $r_{\max}$, would be larger than its old local NND $d_{\min}(x)$; otherwise, we simply reinsert it at its old position.

Figure 4.3 illustrates how the method successively distributes five points $X = \{x_1, \ldots, x_5\}$ in the unit torus. Panels 1a and 1b show how the target position for the first point x_1 is chosen: we search for the triangle in $\mathcal{D}(X \backslash \{x_1\})$ that has the largest circumcircle and move x_1 to the circle's center. The distance map in the background indicates that this is indeed the farthest point. We proceed in the same way for $x_2, \ldots, x_5$, as shown in the remaining panels. A few intermediate steps during the optimization of a larger set consisting of 1024 points are shown in Figure 4.4.

4.2.2 Runtime Complexity

To derive the runtime complexity of FARTHEST-POINT-OPTIMIZATION, we consider one full iteration through the inner loop. We denote the average *degree* of a point (i.e., its average number of neighbors in the Delaunay triangulation) by g and the number of points by $n := |X|$. The runtime of lines 4–10 can now be broken down as follows:

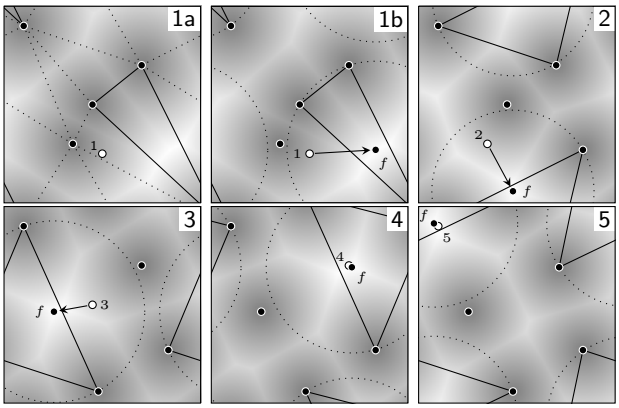


Figure 4.3: Geometrical illustration of one full iteration applied to 5 points in the unit torus. Each point is successively moved to the center of the largest empty circle of the remaining points. The grayscale image in the background represents the distance map of $X\backslash\{x_i\}$ and reflects the toroidal metric. The dotted circle is the largest empty circle, and the highlighted triangle the corresponding face in the Delaunay triangulation of $X\backslash\{x_i\}$.

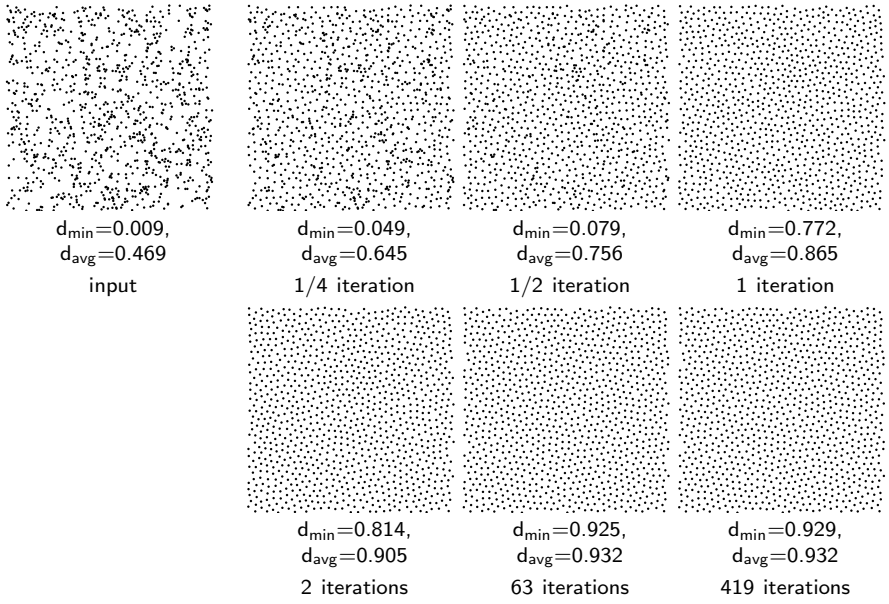


Figure 4.4: Farthest-point optimization of a random point set with 1024 points. Both the global minimum distance $d_{\min}$ and the average minimum distance d_{avg} increase rapidly using our optimization technique. After only one iteration the point set is already well-distributed.

4: $\mathcal{O}(g)$ since we have to inspect the Delaunay neighbors of x to determine $d_{\min}(x)$.

5: between $\mathcal{O}(g)$ and $\mathcal{O}(g^2)$, depending on the algorithm used [Devillers, 2002].

6–9: $\mathcal{O}(n)$ since there are $\mathcal{O}(n)$ triangles in $\mathcal{D}(X)$.

10: $\mathcal{O}(g)$ if we already know the triangle that contains the point; otherwise, between $\mathcal{O}(\sqrt{n})$ and $\mathcal{O}(\log n)$, depending on the algorithm used to locate the triangle [Devroye et al., 2004].

We assume that $g = \mathcal{O}(1)$ which is true or conjectured to be true for large classes of well-distributed point sets [Erickson, 2005]. In this case, the overall runtime is $\mathcal{O}(n)$ for a single movement and $\mathcal{O}(n^2)$ for a full iteration of the inner loop.

Two algorithmic improvements allow us to reduce the runtime to approximately $\mathcal{O}(n \log n)$ per full iteration.

First, we can speed up the process of inserting the farthest point f into the triangulation (line 10). The farthest point f is the circumcenter of the triangle t corresponding to the largest empty circle. In practice, f almost always lies *inside* t, and since we already know t from lines 6–9, the farthest point can usually be inserted in constant time. The fact that f tends to lie inside t follows from a general property of Delaunay triangulations: they maximize the number of acute triangles in the triangulation [Fortune, 1995], and the circumcenter of acute triangles always lies inside the triangle itself.

Second, we can speed up the search for the farthest point in lines 6–9 by using a binary search tree to keep track of the largest empty circle. This lets us find the farthest point in $\mathcal{O}(\log n)$, but increases the time required for lines 5 and 10 also to $\mathcal{O}(\log n)$ since structural changes to the Delaunay triangulation must be reflected in the tree. Taken together, this means that the running time is dominated by the tree operations, and the time required for a full iteration is $\mathcal{O}(n \log n)$.

4.2.3 Convergence

It is easy to see that this farthest-point optimization always converges and yields arrangements with a high average NND. The key observation is that moving a point x to the farthest point of $X \backslash \{x\}$ never decreases $d_{\min}(x)$. In the worst case, no better position can be found and x remains at its old position. Because $d_{\text{avg}} \propto \sum_x d_{\min}(x)$, the average NND must increase during a full

iteration, so the optimization can never return to a previous point distribution or get stuck in cyclic configurations. We stop the iteration as soon as the increase of d_{avg} falls below a threshold ϵ, i.e., as soon as $d_{\text{avg}}^{\text{new}} - d_{\text{avg}}^{\text{old}} < \epsilon$; this must happen eventually since d_{avg} is bounded for points in the unit torus. Convergence is fast enough that we can use the machine precision for ϵ.

Only the *average* nearest-neighbor distance is guaranteed to increase—for $d_{\min}$ we can only be sure that it is non-decreasing. In fact, it is possible to construct point sets that remain mostly stable for several iterations (a regular grid with a few defects, for example). In this case, $d_{\min}$ may also remain constant. But $d_{\min}$ is strictly increasing as long as all points are still moving. For randomly distributed point sets we found this to be always the case.

4.2.4 Variants

The algorithm presented in the previous section works well, but it has a few disadvantages:

- The final $\mathcal{O}(n \log n)$ algorithm from the previous section is efficient, but since two interacting data structures (the Delaunay triangulation and the tree for the empty circles) must be updated dynamically and kept in sync, its implementation is non-trivial.
- If the points are already well distributed, the algorithm works harder than necessary to find the farthest point; we will see that the search can often be simplified radically.
- The algorithm relies on the Delaunay triangulation as its data structure. Since the Delaunay triangulation depends on all point positions, its use is problematic when attempting to parallelize the algorithm.

In the following, we will address these three problems with two different variants of the main algorithm.

Local FPO

The variant discussed in this section builds on the naive $O(n^2)$ algorithm shown in Figure 4.2, but uses a simple heuristic to speed up the search for the farthest point to $O(1)$. Instead of searching for the *largest* empty circle when moving a point x, we contend ourselves with finding a *large* empty circle in the neighborhood of x. In other words, instead of checking the circumcircle of all triangles in $\mathcal{D}(X \backslash \{x\})$, we restrict the search to a neighborhood of triangles

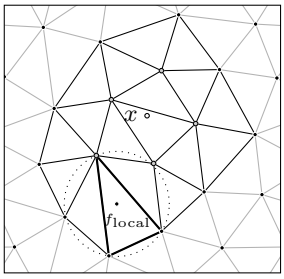


Figure 4.5: A local variant of the FPO can be obtained by moving a point x to *local* instead of the *global* farthest point. This can be done by searching for the largest empty circle in the neighborhood of x.

$T \subset \mathcal{D}(X \backslash \{x\})$ that is in some sense "close" to x. If the expected size of T is independent of n, each point can be moved in $\mathcal{O}(1)$.

There are many possible strategies for choosing the neighborhood T. In our experience, the choice of T does not influence the quality of the resulting point sets, only the number of iterations required to distribute the points. Here, we discuss only one strategy that has proven to be a good compromise between iteration and convergence speed: we include in T all triangles that are incident with the neighbors of x in $\mathcal{D}(X)$ (see Figure 4.5). Since there are $\mathcal{O}(g^2)$ such triangles, moving a single point can indeed be done in constant time. We will refer to this variant as *local FPO*, in contrast to the *global FPO* from Section 4.2.1.

Since our discussion of convergence from Section 4.2.3 only relied on the fact that the $d_{\min}(x)$ doesn't decrease for any point x, it remains valid in the case of the local FPO. But since points aren't moved to the *largest* empty circle anymore, d_{avg} increases more slowly. Nevertheless, both methods converge towards point sets that are indistinguishable. In fact, once the points are sufficiently well distributed, local and global FPO are equivalent, since the farthest point of $X \backslash \{x\}$ is almost always located inside the hole that resulted from removing x.

This suggests a hybrid algorithm that uses the global $\mathcal{O}(n \log n)$ algorithm for the first few iterations and then switches to the more efficient $\mathcal{O}(n)$ algorithm for the remainder of the optimization. In practice, this has turned out to be the fastest variant of farthest point optimization.

Constrained FPO

The second variant of the original farthest-point optimization tackles the problem of parallelization. We only briefly sketch the main idea and refer to the original paper [Chen and Gotsman, 2012] for details.

Chen and Gotsman propose an algorithm they call *constrained farthest point optimization* (CFPO). Similar to the local FPO discussed in the previous section, it speeds up the search for the farthest point. In the case of CFPO, the plane is subdivided into small squares, and the search for the farthest point of $X\backslash\{x\}$ is restricted to blocks of 3×3 squares in the neighborhood of x. The authors prove that if certain conditions on the initial distribution of the points X are fulfilled, the farthest point of $X\backslash\{x\}$ always lies inside this 3×3 block, and that this procedure will converge similar to the original FPO and the local FPO. No global Delaunay triangulation is needed since a local triangulation can be computed on-the-fly for each block.

The main advantage of this approach is that non-overlapping blocks can be processed in parallel, so substantial performance improvements over the original sequential algorithm are possible. Since it only affects the performance but not the final results, we won't discuss constrained FPO further in this thesis.

4.3 Evaluation

In this section we empirically study the main properties of the proposed optimization scheme and the point sets it generates. We discuss two different aspects: first, the overall runtime and speed of convergence; and second, the geometrical and spectral properties of the resulting point sets.

4.3.1 Convergence and Runtime

We implemented the global and local FPO using the dynamic Delaunay triangulations from CGAL [CGAL], which we extended to handle toroidal boundary conditions. Despite their iterative nature, both algorithms are reasonably fast. For 4096 points, one iteration takes an average of 39 ms for the global FPO and 25 ms for the local FPO. Performance measurements were obtained using a single core of a Xeon processor with 2.8 GHz. Starting with a random point distribution, the full optimization until $d_{\min} \geq 0.925$ takes on average 4.7 s (122 iterations) using the global FPO and 8.8 s using the local FPO (348 iterations).

Method $d_{\min} =$	0.75	0.775	0.8	0.825	0.85	0.875	0.9	0.925
[Lloyd, 1982]	70	113	425*	-	-	-	-	-
[Balzer et al., 2009]	111	357*	-	-	-	-	-	-
Local FPO	3	4	6	8	14	27	64	352
Global FPO	1	2	2	3	4	6	13	118
$d_{\text{avg}} =$	0.75	0.775	0.8	0.825	0.85	0.875	0.9	0.925
[Lloyd, 1982]	2	3	4	5	8	13	29	122
[Balzer et al., 2009]	2	2	2	4	10	50	414*	-
Local FPO	1	1	1	1	1	2	3	10
Global FPO	1	1	1	1	1	2	2	6

Table 4.1: Number of iterations needed to achieve a certain NND $d_{\min}$ *(top)* and average NND d_{avg} *(bottom)*. The results are averaged values from optimizing 10 sets of 4096 random points. (*) indicates that the prescribed NND could not always be achieved.

If the original points are randomly distributed, farthest-point optimization converges towards distributions with $d_{\min} \approx 0.93$. Since convergence becomes slower as we approach the maximum, we have found it useful to stop the iteration earlier. In our experience, a threshold of $d_{\min} = 0.925$ is a good compromise between high-quality results and reasonable computation times.

The convergence speed of the global and local FPO variants is compared in Figure 4.6. Both $d_{\min}$ and d_{avg} increase rapidly at first and then converge more slowly towards a maximum around 0.932. The achieved maximum isn't the same for each point set but consistently lies between 0.93 and 0.933. For both algorithms, the three curves for d_{avg} (dashed lines) lie almost on top of each other. This means that convergence of d_{avg} is mostly independent of the number of points, which underlines how effectively FPO distributes the points. The convergence of the $d_{\min}$ (solid lines) depends more strongly on the input size, especially for the local FPO.

Finally, Table 4.1 compares the number of iterations required to obtain well-distributed point sets with Lloyd's method and the algorithm by Balzer et al. It is obvious that both FPO variants are far more effective than the other methods at spreading out the points: a handful of iterations are typically sufficient to obtain point sets with very high Poisson disk radii. These improvements are even more significant considering that state-of-the-art techniques [Balzer et al., 2009, Schmaltz et al., 2010] require $\mathcal{O}(n^2)$ per iteration.

Influence of Initialization In all of our experiments, we started with a random distribution of points. Of course, one obvious question is whether we

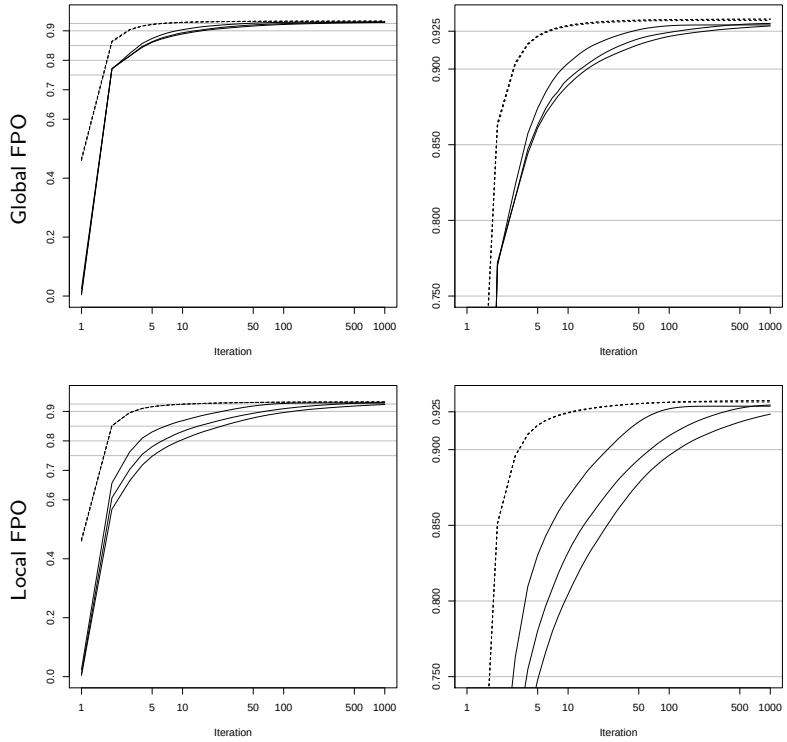


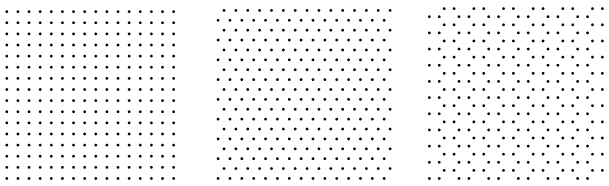
Figure 4.6: Average convergence of $d_{\min}$ (solid lines) and d_{avg} (dashed lines) for random sets of 512, 4096, and 32768 points, from left to right. The right column magnifies the region $0.75 \leq d_{\min} \leq 0.95$.

can improve the convergence rate by choosing a better initialization such as a Poisson disk or jittered grid pattern. Overall, we have not found this to noticeably improve the runtime. Even though a better initialization would allow us to start at a higher average NND, say $d_{\text{avg}} \approx 0.9$, we see in Figure 4.6 that this would shave off only a handful of iterations from the total runtime, since most of the time is spent at $d_{\text{avg}} > 0.9$. Since the FPO is so efficient at redistributing the points, there is hardly a penalty for starting with a random initialization.

Even though most input point sets converge towards irregular arrangements, some stable configurations are regular, such as the following three:

Sampling Pattern	$d_{\min}$	d_{avg}	R_c	Notes	Page
Stochastic Sampling	0.01	0.47	1.73		p. 107
Jittered Grid	0.05	0.59	1.08		p. 108
Dart throwing	0.77	0.81	1.07		p. 109
Best Candidate / Farthest Point Strat.	0.75	0.84	0.76		p. 110
Kernel Density	0.43	0.86	0.78	I	p. 111
Electrostatic Halftoning ($\varrho = 0.002$)	0.74	0.88	0.77	I	p. 112
CCCVT Centroids	0.75	0.90	0.74	I	p. 113
Farthest-Point Optimization	0.93	0.93	0.86	I	p. 118
Low discrepancy	0.90	0.92	0.66	R	p. 115
Lloyd's method	0.80	0.94	0.67	I, R	p. 114
Rectangular grid	0.93	0.93	0.66	R	p. 116
Hexagonal grid	1.00	1.00	0.58	R	p. 117

Table 4.2: Geometric measures of uniformity for common irregular sampling patterns. The patterns are sorted by increasing values of d_{avg}, the expected nearest-neighbor distance. The sampling patterns are presented in more detail in Appendix B.



When removing a point x from any of these patterns, the largest empty circle in the remaining point set has its center precisely at x, so moving single points leaves the point set unchanged. If there are defects in the regular arrangements, however, FPO quickly breaks up the regularity. In this sense, FPO doesn't actively randomize its input, but it amplifies irregularities; this intuitively explains why the algorithm doesn't converge towards regular arrangements.

4.3.2 Geometric Properties

How do point distributions constructed with FPO behave with respect to the two measures of uniformity we introduced in Section 4.1, the nearest-neighbor distance and the coverage radius?

Nearest Neighbor Distance The most obvious feature of point sets generated using FPO is that they have a very high nearest-neighbor distance: as shown in Table 4.2, both $d_{\min}$ and d_{avg} are often significantly higher than with previous algorithms for generating Poisson disk patterns.

The classical non-iterative method for generating Poisson-disk patterns is the *dart throwing* algorithm proposed by Cook [1986]. With dart throwing and related methods, the best NND we can achieve is around $d_{\min} \approx 0.75$. The *farthest point strategy* [Eldar et al., 1997], which is the non-iterative algorithms on which FPO is based, also produces results around $d_{\min} \approx 0.75$. The main limitation of non-iterative algorithms seems to be their inability to undo previous suboptimal decisions.

Do iterative methods fare better? Several iterative methods for constructing blue noise patterns have been proposed in recent years, in particular Fattal's kernel density method [2011], electrostatic halftoning by Schmaltz et al. [2010] and CCCVT by Balzer et al. [2009]. It can be seen in Table 4.2 that these methods consistently produce higher d_{avg} than non-iterative methods for generating blue noise patterns. The global NND $d_{\min}$, however, is comparable to dart throwing.

The only sampling patterns that consistently achieve higher values for $d_{\min}$ and d_{avg} than FPO are regular grid or semi-regular point distributions like low-discrepancy sequences or the result of Lloyd's method. But due the regular patterns that occur in the final distributions, these are not as resilient to aliasing.

Coverage Radius The second spatial statistic we are interested in is the coverage radius R_c. As can be seen in Table 4.2, for FPO we obtain an average value of $R_c = 0.86$. We always have $R_c < d_{\min}$, so FPO point sets are *maximal*. Nevertheless, compared to other iterative construction algorithms, the coverage radius of FPO is relatively high. The effect can actually be seen when visualizing the resulting point distributions: Figure 4.7 demonstrates that FPO has a tendency to arrange points in such a way that small holes form that are surrounded by a ring of points. The appearance of these holes is somewhat surprising: Since the FPO algorithm moves points to the largest empty region, we would expect the coverage radius to decrease during the optimization.

The existence of these holes does not seem to have a significant impact on other spatial and spectral properties of the point sets, however. We experimented with different ways to avoid the holes during construction or plugging them afterwards. This makes it possible to reduce the coverage radius to

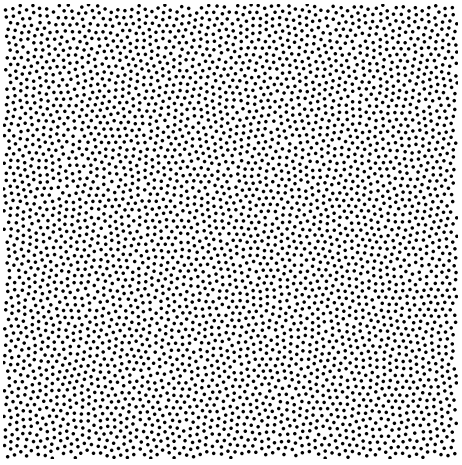


Figure 4.7: Point sets generated by FPO have a slightly non-uniform appearance due to the existence of small "holes" in the point set. These holes have almost no influence on the geometric and spectral properties of the point set, however.

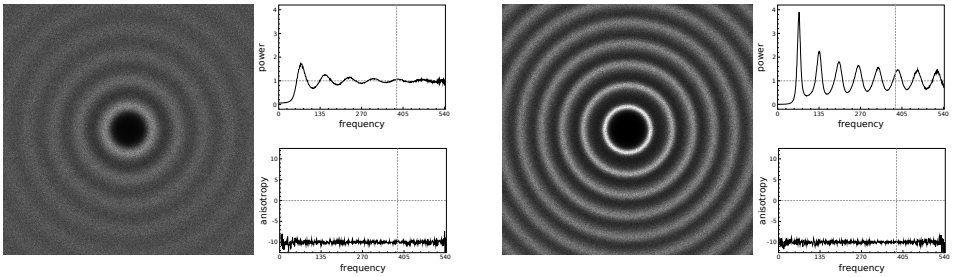
$R_c \approx 0.76$ without affecting other statistics except for $d_{\min}$, which goes slightly down to 0.9. The small holes in FPO point sets may be visually conspicuous, but they have little impact on the statistical and spectral behavior of the point set.

4.3.3 Spectral Properties

Figure 4.8 shows the standard spectral measures—power spectrum, radially averaged power spectrum, and anisotropy—based on ten FPO point sets with 4096 points (for each set $d_{\min} \approx 0.93$). We compare the results to pure dart throwing ($d_{\min} \approx 0.77$).

We see in Figure 4.8 that there is almost no energy around the origin and no discernible anisotropy for FPO points. In the introduction to this chapter we mentioned the conjecture by Lagae and Dutré [2008] that for $d_{\min} > 0.85$ point sets become anisotropic. The point sets generated by FPO demonstrate that this is not the case, and that isotropic point sets are possible at least up to $d_{\min} \approx 0.932$. It is not clear whether this bound can be improved even further.

The effective Nyquist frequency of FPO is approximately $\nu_{\text{eff}} \approx 0.9$, which is slightly higher than that of other irregular sampling patterns (cf. Table 3.1. This wide zero-region around the origin is what we hoped for, since prior experience suggested that a high Poisson disk radius leads to a wider and



Dart Throwing, $d_{min} \approx 0.75$ *Our Method*, $d_{min} \approx 0.93$

Figure 4.8: Common spectral measures for point sets generated by pure dart throwing *(left)* and our optimization method *(right)*. Both of our algorithms consistently converge towards anisotropic point sets almost no energy around the origin of the power spectrum.

cleaner zero region.

What is remarkable, however, is the strong increase of the oscillation of the power spectrum: for FPO points we have $\Omega \approx 4.6$, which is almost twice as high as other blue noise patterns with a high ν_{eff}. This is evident in Figure 4.8: the peaks in the power spectrum of FPO are more pronounced than for dart throwing and fall off much more slowly. In fact, the power spectrum falls off more slowly than any other blue noise pattern we are aware of: traditional blue noise patterns such as the dart throwing patterns have a single wide peak after the zero region and a few small oscillations afterwards. For sampling these peaks in the power spectrum are of course undesirable: As explained in Chapter 3, each peak in the spectrum can introduce low-frequency noise into the sampled image.

4.4 Discussion

This chapter discussed a new iterative algorithm for constructing Poisson disk patterns which we call *farthest point optimization*. The algorithm is geometrically intuitive and fairly efficient, in particular compared to previous methods for constructing Poisson disk patterns. The resulting point distributions are, in a sense, the "perfect" Poisson disk patterns: they are completely isotropic, but have a much higher Poisson disk radius than point sets achievable with other algorithms. In the original publication on FPO [Schlömer et al., 2011] we concluded as follows:

> The main feature of the resulting point sets is that they are prac-

> tically optimal blue noise samples under the assumption that such point sets should be both irregular and of high minimum distance. This re-raises the question of ideal image plane sample points as we suspect that it will be hard to increase the minimum distance further without introducing regular structures.

In retrospect, it is easy to see the fallacy behind this statement: for sampling applications the spectral and not the geometric properties are of primary importance. As we have seen in this chapter, good Poisson disk properties do not automatically translate into good blue noise properties. This was a silent assumptions in previous research on blue noise which focused on increasing the Poisson disk radius.

It is therefore important to clearly distinguish between "blue noise" and "Poisson disk" patterns. This seems obvious, since blue noise refers to frequency characteristics whereas Poisson disk is a spatial property, but for a long time, it made sense to use the two terms interchangeably, primarily because Poisson disk patterns were the only way to construct point distributions with a blue noise spectrum.

In a sense, FPO may mark the end of the long-going quest for higher and higher Poisson disk radii. Enforcing a wide separation of sample points was a heuristic that has served the graphics community well for many years, but it is no guarantee for good sampling patterns. Better sampling patterns in the future will therefore require either additional geometric constraints, such as Balzer's *capacity constraints* [Balzer et al., 2009], or a focus on the spectral properties during construction. We will cover the latter approach in the following chapter.

Chapter 5

Spectral Construction of Blue Noise

The FPO algorithm presented in the previous chapter follows the standard approach for constructing sampling patterns that has been pursued in graphics over the last 30 years: some geometric insight or constraint is used to arrange points in the plane, and afterwards, Fourier analysis is used to evaluate the spectral properties of the point set. This approach to constructing blue noise sampling patterns has three major disadvantages.

1. Finding geometric properties that correlate with a good blue noise spectrum *and* can be turned into efficient construction algorithms is highly nontrivial.

2. The range of sampling patterns that can be studied in the first place is limited by the geometric constraints we can come up with. For example, basically all blue noise sampling patterns that have been constructed oscillate in the high-frequency region, but geometric considerations cannot tell us whether this is necessary or merely a side effect of the construction algorithm.

3. For image-plane sampling we are particularly interested in the power spectrum of the resulting point set, but geometric methods give us no direct control over this spectrum.

It would therefore be desirable to be able to "design" sampling patterns by specifying their spectral properties and then deriving their spatial distribution automatically. This is the topic of this chapter.

Contributions. The key to linking the spectral and geometrical views of sampling patterns lies in the mathematical relationship between the power spectrum, which is defined in the Fourier domain, and the autocorrelation, which is defined in the spatial domain. From a theoretical perspective, this allows us to answer the question which power spectra can actually be realized by point distributions. From a practical perspective, this leads to an iterative algorithm for constructing point distributions matching a given power spectrum.

We apply this algorithm to two different applications. First, we demonstrate that it is possible to simulate other algorithms for generating blue noise patterns simply by imitating their power spectrum, but without knowing anything about the underlying geometric constraints. The second application is to design new blue noise patterns with particular spectral behavior. To this end we consider two classes of functions which are idealizations of typical blue noise spectra. The spectrum of *step blue noise* is a step function, and we determine the highest possible position of the step. The spectrum of *single-peak blue noise* is similar to a step function, but has one single configurable peak in the transition region. We show that both types of blue noise have favorable sampling properties. This chapter is based on [Heck et al., 2013].

5.1 Autocorrelation and Pair Correlation

We explained the importance of the power spectrum for analyzing irregular sampling in Chapter 3. Since the power spectrum is defined in the Fourier domain, there is no simple geometric relationship between the point positions and the spectrum. As a consequence, there is no way to influence the spectrum directly, and no way to actually design sampling patterns with particular spectral properties. This restriction has held back research on blue noise for a long time.

The first step towards tackling this problem was made by Wei and Wang [2011], who were looking for a geometric alternative to the power spectrum that is more intuitive and easier to analyze. They observed that for a sufficiently large sampling pattern $S(\boldsymbol{x}) = \sum_i \delta(\boldsymbol{x} - \boldsymbol{x}_i)$ in the unit square, the power spectrum can be approximated by the so-called *periodogram*

$$\begin{aligned} P_S(\boldsymbol{\nu}) \approx \left|\mathcal{F}\big[S(\boldsymbol{x})\big]\right|^2 &= \sum_i e^{-2\pi i \boldsymbol{x}_i \cdot \boldsymbol{\nu}} \sum_j e^{2\pi i \boldsymbol{x}_j \cdot \boldsymbol{\nu}} \\ &= \sum_{i,j} e^{-2\pi i \boldsymbol{r}_{ij} \cdot \boldsymbol{\nu}} \end{aligned}$$

Here, $\boldsymbol{r}_{ij} = \boldsymbol{x}_i - \boldsymbol{x}_i$ denotes the distance vectors between two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. We can replace the sum with an integral by introducing a density function $\rho(\boldsymbol{r})$

$$P_S(\boldsymbol{\nu}) = \int_{\mathbb{R}^2} \rho(\boldsymbol{r}) e^{-2\pi i \boldsymbol{r}\cdot\boldsymbol{\nu}} \,\mathrm{d}\boldsymbol{r}$$

Written in this way, the power spectrum is basically the Fourier transform of the function $\rho(\boldsymbol{r})$, which measures distribution of distance vectors and was therefore called the "differential distribution function" [Wei and Wang, 2011].

It turns out that this differential distribution function is actually a well-known quantity, namely the *autocorrelation function* of the sampling pattern $C_S(\boldsymbol{r})$. For general signals, the autocorrelation measures the self-similarity of a signal under translation and is defined as

$$C_X(\boldsymbol{r}) = \mathbb{E}\big[X(\boldsymbol{x})X(\boldsymbol{x}+\boldsymbol{r})\big].$$

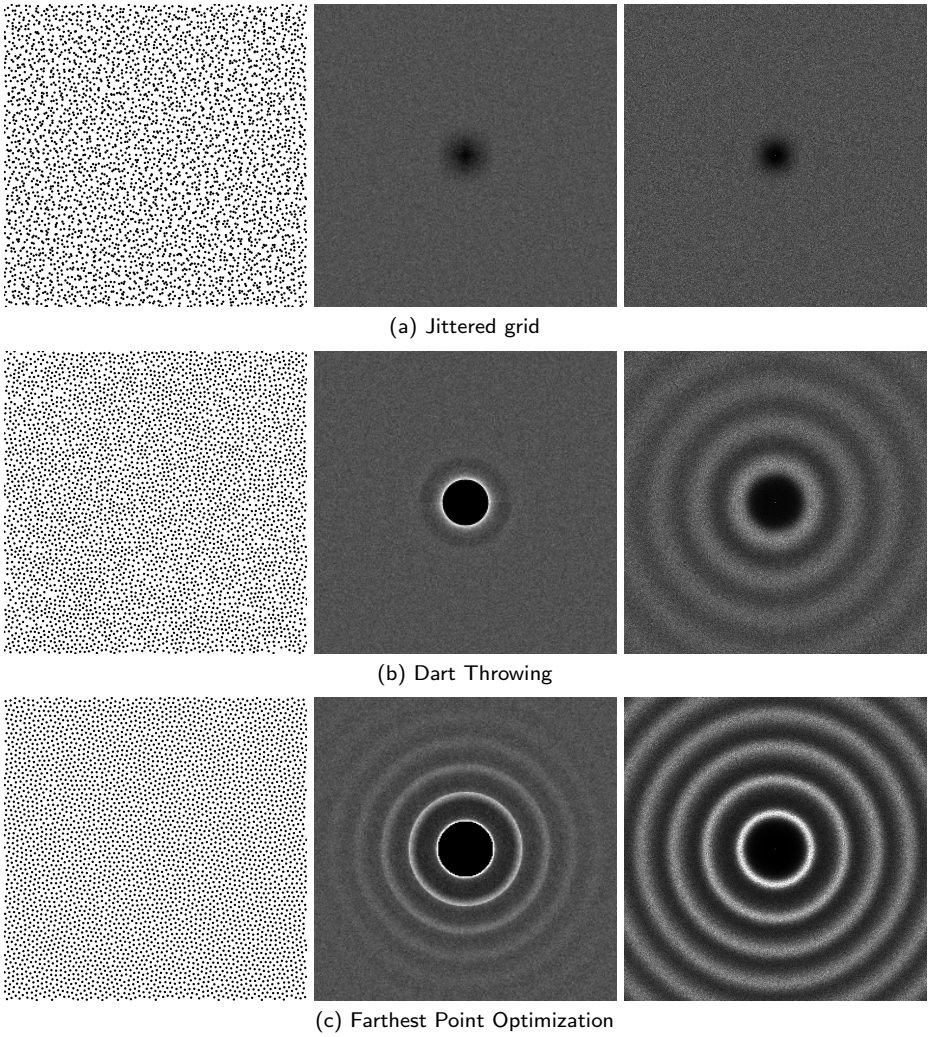
The most important property of the autocorrelation is that its Fourier transform is equal to the power spectrum

$$P_X(\boldsymbol{\nu}) = \mathcal{F}\big[C_X(\boldsymbol{r})\big]. \tag{5.1}$$

The autocorrelation and its relation to the power spectrum are explained in more detail in Appendix A. Figure 5.1 shows the autocorrelation compared to the power spectrum for a few common sampling patterns; more examples can be found in Appendix B.

Wei and Wang [2011] were primarily interested in using the autocorrelation as an analysis tool, since $C_S(\boldsymbol{r})$ reflects both the radial and the translational symmetry of a point set. Traditionally, the power spectrum has been used to measure these two aspects of a point set [Ulichney, 1988], but using the autocorrelation instead has several immediate advantages:

- It is easy to calculate, since it can be approximated by a 2D histogram of pairwise distances. In particular, no Fourier transforms are necessary.
- It is easy to interpret, since it is a function of spatial distances, not frequency.
- It can be generalized to non-Euclidean spaces, as long as a suitable metric can be defined, so it allows the analysis of point distributions on surfaces, for example.
- It is easy to influence by adjusting the relative positioning of points. For the purposes of this chapter, this is the main advantage.



(a) Jittered grid

(b) Dart Throwing

(c) Farthest Point Optimization

Figure 5.1: A few examples of autocorrelation functions of sampling patterns, compared to their power spectra. *(Left)* Point set. *(Center)* Autocorrelation. *(Right)* Power spectrum. A high (bright) value in the autocorrelation indicates a strong correlation between two points at that distance. For dart throwing and FPO, for example, there is a very strong, clean peak at the Poisson disk radius.

Of course, the power spectrum remains crucial for interpreting the sampling process, since the autocorrelation does not directly predict the amount and appearance of aliasing.

For point distributions, it is common to use a quantity known as the *pair correlation function* (PCF) instead of the autocorrelation [Illian et al., 2008]. It is closely related and defined as

$$g(\boldsymbol{r}) = \frac{1}{n^2} C_X(\boldsymbol{r}) - \frac{1}{n}\delta(\boldsymbol{r}). \tag{5.2}$$

Basically, the PCF is a renormalized version of the autocorrelation with the Dirac peak at the origin removed. The magnitude of $g(\boldsymbol{r})$ measures the amount of correlation between points at distance $\boldsymbol{r}$: $g(\boldsymbol{r}) < 1$ indicates a negative correlation, $g(\boldsymbol{r}) > 1$ a positive correlation and for $g(\boldsymbol{r}) = 1$ there is no correlation. The radius beyond which $g(\boldsymbol{r}) \approx 1$ measures the *correlation length*, which can be thought of as the distance below which the points "interact". For irregular point sets $g(\boldsymbol{r}) \to 1$ as $r \to \infty$ since the points are uncorrelated at long distances.

5.1.1 Radial Distribution Function

We are primarily interested in *isotropic* point distributions to avoid directional bias when sampling. For isotropic point distributions, the power spectrum, the autocorrelation, and the pair correlation are radially symmetric, so

$$P(\boldsymbol{\nu}) = P(\nu), \qquad C_S(\boldsymbol{r}) = C_S(r), \qquad g(\boldsymbol{r}) = g(r).$$

$P(\nu)$ is known as the *radial power spectrum* and $g(r)$ the *radial distribution function*. For isotropic point sets they obviously contain the same information as their two-dimensional counterparts, but since they are one-dimensional functions, they are often easier to handle mathematically. Lau et al. [2003] used RDF diagrams to qualitatively illustrate the spatial distribution of points, but we aren't aware of other applications in computer graphics.

Just as the autocorrelation is related to the power spectrum by a Fourier transform, the RDF $g(r)$ is related to the *radial power spectrum* $P(\nu)$ by a Hankel transform

$$\boxed{P(\nu) = 1 + n\mathcal{H}[g(r)], \qquad g(r) = \frac{1}{n}\mathcal{H}[P(\nu) - 1].} \tag{5.3}$$

(Compared to Chapter 3, this definition of the power spectrum differs by a factor of $1/n$.) The Hankel transform is the special case of a Fourier transform

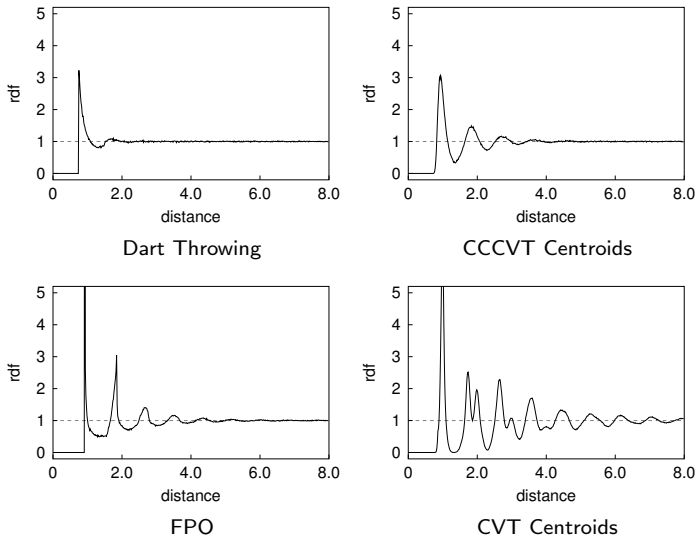


Figure 5.2: Radial distribution functions for several blue noise patterns. The plots have been obtained by averaging the RDFs of 10 realizations of each point set.

for radially symmetric functions; we will briefly discuss its main properties in the next section. The RDF and its relation to the radial power spectrum will be used extensively in the remainder of this chapter.

Figure 5.2 shows the RDFs of several different point distributions. Several important properties can be read off directly from the RDF. The nearest-neighbor distance, for example, is the radius at which the RDF becomes non-zero, and the average NND corresponds roughly to the first peak in the RDF.

5.1.2 Hankel Transform

In Eq. (5.3) we used the *Hankel transform* $\mathcal{H}$ to describe the relationship between the radial power spectrum and the radial distribution function. In principle, the Hankel transform is the equivalent of a two-dimensional Fourier transform of a spherically symmetric function $f(r) = f(r, \theta)$ [Bracewell, 1999]. If we rewrite the standard Fourier transform in polar coordinates, we obtain

$$F(u, \phi) = \int_0^\infty r f(r) \int_0^{2\pi} e^{-2\pi i r u \cos(\theta - \phi)} \, \mathrm{d}\theta \, \mathrm{d}r.$$

Function	Hankel Transform	Notes
$af(r) + bg(r)$	$aF(u) + bG(u)$	Linearity
$f(ar)$	$a^{-2}F(u/a)$	Scaling
$\text{disk}(r)$	$2\pi\,\text{jinc}(2u)$	Unit disk
$H(r-a)$	$\delta(u)/2\pi u - 2\pi a^2\,\text{jinc}(2ua)$	Step function
1	$\delta(u)/2\pi u$	Constant

Table 5.1: Common Hankel transform pairs used in the text.

The inner integral can be rewritten using the definition of the Bessel function

$$J_0(a) = \frac{1}{2\pi}\int_0^{2\pi} e^{-ia\cos\theta}\,\mathrm{d}\theta, \tag{5.4}$$

which yields

$$F(u) = \mathcal{H}[f(r)] = 2\pi\int_0^{\infty} rf(r)J_0(2\pi ru)\,\mathrm{d}r. \tag{5.5}$$

The right-hand side does not depend on ϕ, which implies that the Fourier transform of a circularly symmetric function is also circularly symmetric. This integral transform is the *Hankel transform*. The inverse Hankel transform is identical to the forward transform, so

$$f(r) = \mathcal{H}[F(u)] = 2\pi\int_0^{\infty} uF(u)J_0(2\pi ru)\,\mathrm{d}u.$$

The main properties of the Hankel transform follow from the properties of the two-dimensional Fourier transform; we list the most important ones in Table 5.1. The unit disk $\text{disk}(r)$, the Heaviside step function $H(r)$, and the jinc function are defined as follows

$$\text{disk}(r) = \begin{cases} 1 & r \le 1 \\ 0 & r > 1 \end{cases}, \qquad H(r) = \begin{cases} 0 & r \le 0 \\ 1 & r > 0 \end{cases}, \qquad \text{jinc}(x) = \frac{J_1(x)}{x}.$$

5.2 Spectrum Matching Algorithm

The standard approach to constructing sampling patterns is to specify the desired geometric properties first, then construct a matching point distribution, and finally analyze the spectral behavior of the resulting point set. In

this section we study how to reverse this process by specifying the desired *target spectrum* P_t, derive from this spectrum the required geometric constraints and use these to construct a suitable point distribution. We do this by using the relationship between the radial power spectrum and the radial distribution function from the previous section to translate the problem of matching a target spectrum P_t to the problem of matching a target RDF g_t. This is significantly easier since the RDF is defined in the spatial domain and can therefore be influenced directly by adjusting the relative positioning of the points.

Similar algorithms have also been proposed by other researchers. Rintoul and Torquato [1997] studied a similar problem to simulate the microstructure of disordered materials in physics. Their algorithm is based on simulated annealing, but we have found the convergence to be very slow and unreliable for reasonably large point sets. Two recent papers by Zhou et al. [2012] and Öztireli and Gross [2012] propose spectrum matching algorithms for computer graphics. Because these papers were published while our work was still under review, we haven't been able to evaluate them in detail. We perform a preliminary comparison with Zhou et al.'s algorithm in Section 5.4.4.

5.2.1 Main Algorithm

From an algorithmic point of view, our method is very similar to that of Zhou et al.: Both algorithms synthesize point sets with a specified spectral behavior by reformulating the problem in the spatial domain, and both iteratively update the positions of all points by applying a force to each point

$$\boldsymbol{x}_i' = \boldsymbol{x}_i + h \cdot \boldsymbol{F}_i, \tag{5.6}$$

where h is a step size parameter and $\boldsymbol{F}_i$ is a force that depends on the current point positions. The main difference between both algorithms is how they calculate the forces in Eq. (5.6): Zhou et al. propose a force based on gradient descent, whereas ours is motivated geometrically.

Our algorithm works as follows. First, the "target" power spectrum $P_t(\nu)$ is transformed into an equivalent RDF $g_t(r)$ by numerically evaluating the Hankel transform; we will discuss this numerical integration in more detail in the following section. Since both functions are one-dimensional, our approach can only synthesize isotropic point sets.

The current point set is initialized with a random distribution of points. To evolve the point set towards the target distribution, we let all points attract

or repel each other using forces of the form

$$\boldsymbol{F}_i = \sum_{j \neq i} f(|x_i - x_j|) \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}. \tag{5.7}$$

The function $f(r)$ determines the strength of attraction or repulsion and is defined as follows:

$$f(r) = \underbrace{\int_0^r g(x)\,\mathrm{d}x}_{G(r)} - \underbrace{\int_0^r g_t(x)\,\mathrm{d}x}_{G_t(r)}. \tag{5.8}$$

This choice of f can be motivated as follows: Since the RDF measures the density of points at a certain distance, $G(r)$ measures the average point density in disks of radius r. For some fixed radius r_0, $G(r_0) > G_t(r_0)$ indicates that the current point distribution contains more point pairs that are closer than r_0 than the target distribution. In this case the point set as a whole has to spread out and $f(r_0)$ should be repulsive to make room. Conversely, if $G(r_0) < G_t(r_0)$, too many pairs have a distance greater than r_0 and the points have to move closer together; in this case, $f(r_0)$ should be attractive. In both cases, the choice $f(r) = G(r) - G_t(r)$ fulfills this condition.

In each iteration, we first calculate all the forces $\boldsymbol{F}_i$ and then update the positions according to Eq. (5.6). Moving the points can be easily parallelized since we use a global force and because the new position of each point only depends on the old position of all other points.

The main parameter during each iteration is the step size h, which is chosen adaptively depending on the largest force as in Zhou et al. We use an additional temperature parameter T to reduce the step size whenever the optimization gets stuck

$$h = T\frac{F_{\max}}{\sqrt{n}}, \qquad F_{\max} = \max_i \|\boldsymbol{F}_i\|.$$

The $\sqrt{n}$ term ensures that the step size is independent of the number of points. To track the progress of the optimization, we use an energy

$$E = \|g(r) - g_t(r)\|_2$$

that measures the difference between the current and the target RDF. The optimization is considered stuck if E hasn't decreased during the last 20 iterations; in this case, the temperature is reduced by a constant factor α. The

choice of α is a tradeoff between accuracy (high value) or faster termination (low value); we used a factor of $\alpha = 0.9$ for our experiments. The algorithm starts with $T = 1$ and terminates once the temperature has fallen below 10^{-3}.

We assume that the points lie in the unit torus and calculate the RDF for absolute distances in the interval $[0, 0.5)$. Using smaller intervals can speed up some of the computations by reducing the number of point pairs that must be considered. This optimization was proposed by Wei and Wang [2011], but we have found that it reduces the quality of the results for RDFs that decay slowly. RDFs are approximated using histograms of pairwise distances, and since the force $f(r)$ is derived from the RDFs, we discretized it using the same bin width. The number of bins is a tradeoff: large bins lower the resolution of $f(r)$ which limits the precision with which points can be moved, but small bins lead to noisy RDF estimates which also leads to inaccurate results. In our experiments, using as many bins as there are points $n_{\text{bins}} = n$ has proven to be a good compromise.

Since a certain amount of noise is inevitable when estimating RDFs from finite point sets, we optionally smooth the histograms using a Gaussian kernel. There is no simple rule for choosing the optimal width σ of the Gaussian, since this involves a tradeoff between reducing noise vs. keeping relevant information in the RDF. In our experiments, good values for σ were between 0 and $16/n_{\text{bins}}$. All of our results have been generated with $\sigma = 8/n_{\text{bins}}$, and no parameters had to be adjusted manually.

5.2.2 Numerical Hankel Transform

The original formulation of the relationship between power spectrum and RDF in Eq. (5.3) aimed for mathematical simplicity

$$P(\nu) = 1 + n\mathcal{H}[g(r)], \tag{5.9}$$

but for numerical computations, a different notation is preferable. Since $g(r) \to 1$ as $r \to \infty$, the Hankel transform becomes easier to evaluate if we rewrite Eq. (5.3) as

$$P(\nu) = 1 + n\mathcal{H}\big[g(r) - 1 + 1\big] = 1 + n\mathcal{H}\big[g(r) - 1\big] + n\frac{\delta(\nu)}{2\pi\nu}. \tag{5.10}$$

The main advantage of this notation is that $g(r) - 1$ vanishes as $r \to \infty$, so the integral underlying the Hankel transform can be evaluated more easily. Incidentally, this representation also emphasizes the main components of a typical power spectrum, namely the DC peak at $\nu = 0$ and the oscillation

around 1, which is due to the Hankel transform of $g(r) - 1$.

We can compute the radial power spectrum numerically by integrating $g(r) - 1$ over a finite interval $[0, r_1)$

$$P(\nu) = 1 + 2\pi n \int_0^{r_1} r\big[g(r) - 1\big] J_0(2\pi r\nu)\,\mathrm{d}r + n\frac{\delta(\nu)}{2\pi\nu}.$$

There are two remaining problems with this formulation. The first is that we are effectively cutting off $g(r) - 1$ at the upper limit of the integral r_1, which can lead to Gibbs artifacts in the calculated $P(\nu)$. To solve this problem, we use a window function to smoothly reduce $g(r) - 1$ to zero:

$$P(\nu) = 1 + 2\pi \int_0^{r_1} \big[g(r) - 1\big] J_0(2\pi r\nu) w(r)\,\mathrm{d}r + n\frac{\delta(\nu)}{2\pi\nu}.$$

We have used a standard Blackman window of width r_1 for $w(r)$.

The final issue is that $g(r)$ is usually obtained from a histogram of pairwise point distances, which is necessarily noisy. For high values of ν this is not problematic: The Bessel function J_0 oscillates quickly, so the effect of noise in the data averages out. For low frequencies ν, however, the noise leads to systematic errors. We solve this problem by integrating over a smoothed version of $g(r)$ for low frequencies.

The preceding observations applied to calculating the power spectrum from the RDF, but the same ideas are also valid for the inverse transformation which computes $g(r)$ from $P(\nu)$.

5.2.3 Simulating Blue Noise Construction Methods

The simplest way to obtain realizable power spectra is to derive them from known point distributions. Even though this is a trivial solution, it is an interesting one, since there is a wide range of blue noise distributions we can use as a model and simulate. It also allows us to compare the results of our algorithm to the original point sets.

A few examples from this approach are shown in Figures 5.3 and 5.4. In all examples we compare the original point distribution to the simulated result. It is obvious that the RDFs and power spectra of the simulated distributions are very similar. The main difference is that sharp features in the RDFs are smoothed out during the optimization process, which is most obvious in the case of dart throwing and Lloyd's method.

The general shape of the power spectra is retained, however. This is interesting and somewhat unexpected, especially in the case of CCCVT and

Lloyd's method. As we explained in Section 2.5, both methods involve a fairly sophisticated geometric construction, so it is not at all obvious that the resulting point distributions can also be constructed—at least to a reasonable accuracy—from summary statistics like the RDF or power spectrum alone. This illustrates that $P(\nu)$ and $g(r)$ contain a lot of information about the overall distribution of points.

It is worth noting that specifying $P(\nu)$ or $g(r)$ generally does not uniquely determine the resulting point distributions. This is illustrated by the result for jittering in Figure 5.3, in which the simulated point sets are spectrally indistinguishable but are visually distinct: the simulated points show a tendency to arrange in short strings, whereas the original points are distributed more uniformly. In this case, the RDF/spectrum puts only a mild constraint on the overall distribution of points. The fact that the RDF doesn't always uniquely characterize the distribution of points was already observed by Rintoul and Torquato [1997].

5.3 Designing Low-Oscillation Blue Noise

The algorithm discussed in the previous section assumes that we already know the target spectrum $P_t(\nu)$. It turns out, however, that finding suitable spectra itself is a challenging problem. The main difficulty is that not all spectra are realizable by point distributions.

In this section we first discuss two necessary conditions which describe when a power spectrum is realizable. We then consider two ways to design "new" power spectra for blue noise sampling. We specifically aim to design power spectra with a low amount of oscillation in the high-frequency region while keeping the zero region as wide as possible. We do this by explicitly constructing the desired power spectra in functional form and then constructing point sets matching these spectra. We consider two classes of blue noise spectra: *step blue noise* has the shape of a step function, whereas *single-peak blue noise* contains an additional peak at in the transition region but is otherwise flat. We will see how the realizability conditions lead to a tradeoff between the size of the zero-region in the spectrum and the amount of oscillation in higher frequencies.

5.3.1 Realizability Conditions

There are two necessary conditions a power spectrum $P(\nu)$ must fulfill to be realizable by a point process [Crawford et al., 2003, Uche et al., 2006]. The first necessary condition is $P(\nu) \geq 0$, which follows directly from the definition

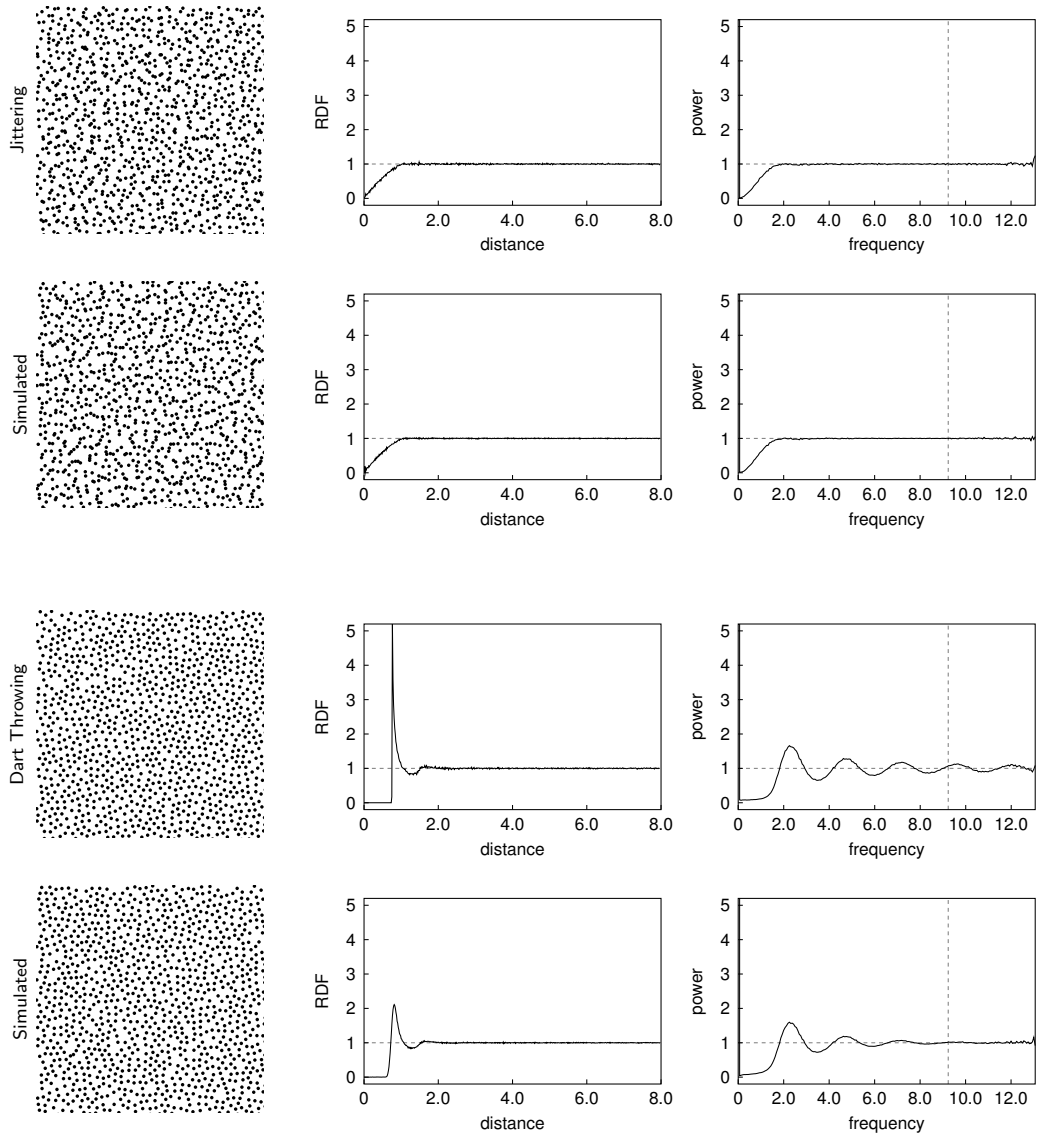


Figure 5.3: Simulating jittered grid and dart throwing.

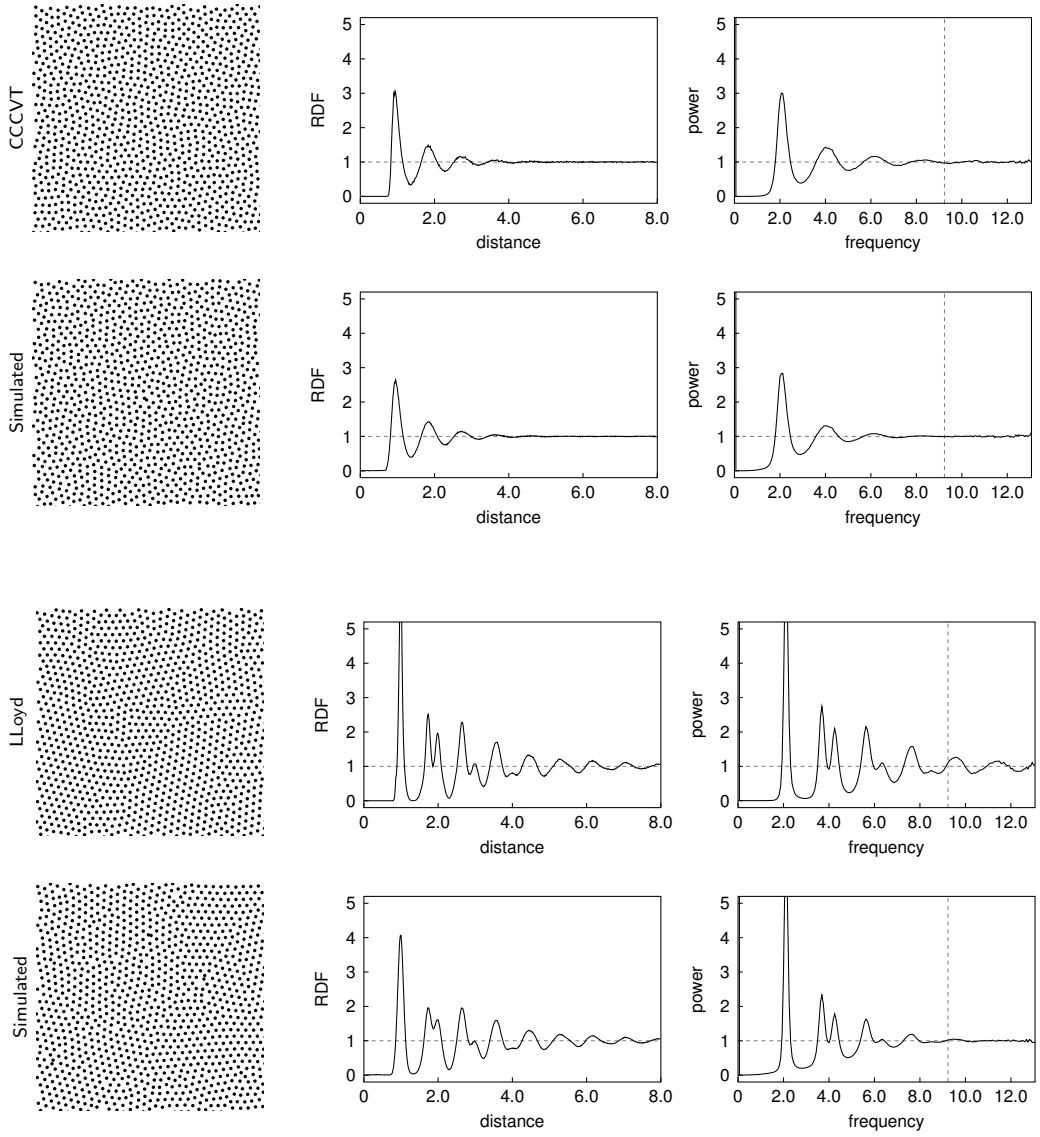


Figure 5.4: Simulating CCCVT and Lloyd's method.

of the power spectrum as an energy average. Likewise, we must have $g(r) \geq 0$, since the RDF can be interpreted as the probability of finding another point at a certain distance r from a reference point. We therefore have the following necessary conditions for the realizability of a power spectrum or RDF:

$$\boxed{\begin{aligned} g(r) &\geq 0, \\ P(\nu) &\geq 0. \end{aligned}} \tag{5.11}$$

Because the two functions are linked via a Hankel transform, these two conditions severely limit the range of realizable power spectra. Whether these two conditions are not only necessary but also sufficient is still an open question, but no counterexamples are known [Torquato and Stillinger, 2002]. We refer to Eq. (5.11) as the *realizability conditions*.

Because the space of functions that obey the realizability conditions is not easy to parametrize [Giraud and Peschanski, 2006, Uche et al., 2006], constructing realizable power spectra is a nontrivial problem. In the following sections we discuss two experiments that show how to find realizable blue noise power spectra with a simple mathematical form.

5.3.2 Step Blue Noise

The effect of the realizability conditions is best visualized using a concrete example. We therefore study an idealization of blue noise which we call *step blue noise*: The power spectrum of step blue noise is zero in low frequencies and constant in high frequencies, i.e.,

$$P_{\text{step}}(\nu; \nu_0) = n\frac{\delta(\nu)}{2\pi\nu} + H(\nu - \nu_0). \tag{5.12}$$

Here we have included the DC peak at the origin and used the Heaviside step function H (cf. Figure 5.5(a)). A point distribution with a step-like power spectrum was already constructed by Zhou et al. [2012], but it seems their result was obtained by trial and error. In contrast, the realizability conditions allow us to derive the largest possible frequency for the step position $\nu_{\max}$.

The RDF associated with the step blue noise spectrum (5.12) can be derived using Eq. (5.10) and the Hankel transforms in Table 5.1:

$$g_{\text{step}}(r) = 1 - \frac{2\pi\nu_0^2}{n}\,\text{jinc}(2\nu_0 r).$$

The realizability conditions require that $g_{\text{step}} \geq 0$. Using the fact that the jinc

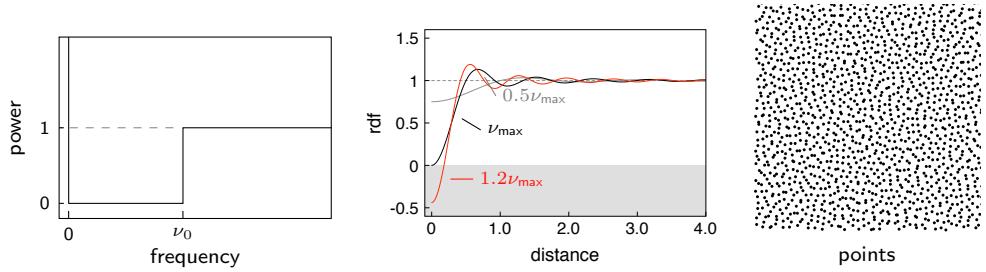


Figure 5.5: Radial power spectra, corresponding RDFs, and cutouts from corresponding points sets for step blue noise. A radial power spectrum that is a perfect step is only realizable up to a maximum frequency $\nu_{\max}$. For frequencies $\nu_0 > \nu_{\max}$ the corresponding RDF becomes negative.

function has a maximum of $\text{jinc}(0) = 1/2$, we can solve this inequality for ν_0 to obtain $\nu_{\max}$

$$g_{\text{step}} \geq 0 \quad \Leftrightarrow \quad \nu_0 \leq \nu_{\max} = \sqrt{n/\pi}.$$

Figure 5.5(a) demonstrates that for $\nu_0 > \nu_{\max}$, the RDF g_{step} becomes negative. The only way to move the position of the step further to the right is to increase the sample density n.

Figure 5.5 also shows one resulting point set for $\nu_0 = \nu_{\max}$. Note that the point distribution differs significantly from usual Poisson disk patterns in that it contains many closely spaced point pairs. This is noteworthy because for basically every other known point distribution with a blue noise spectrum, the points have a certain minimal separation. It also goes against the conventional wisdom that efficient sampling patterns should be spread out as much as possible. But in this case, the primary goal isn't efficiency but aliasing prevention. In this sense, the irregular separation of the samples seems to help decorrelate the sampling pattern as a whole.

5.3.3 Single-Peak Blue Noise

The step blue noise patterns from the previous section prevent coherent aliasing by keeping the power spectrum flat, but this comes at a cost: The effective Nyquist frequency of these patterns cannot be higher than $\sqrt{n/4\pi}$, which is 56.4% of the maximum Nyquist frequency of the hexagonal lattice ν_{hex}. In this section, we discuss a class of blue noise patterns that offer a much higher Nyquist frequency (up to 86 % of ν_{hex}) by introducing a single peak into an otherwise flat power spectrum.

The functional form we choose for this *single-peak blue noise* is a general-

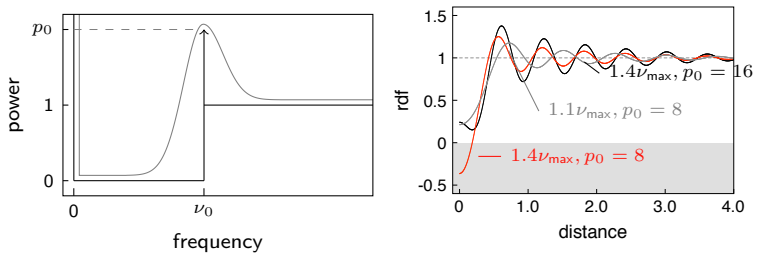


Figure 5.6: To achieve a higher effective Nyquist frequency, we have to allow the power spectrum to go above 1 by introducing a (potentially smoothed) peak around $\nu_0 > \nu_{\max}$. To keep the corresponding RDF positive, we have to increase the height p_0 of the peak as we increase ν_0.

ization of the step blue noise spectrum

$$P_{\text{peak}}(\nu; \nu_0; p_0) = n\frac{\delta(\nu)}{2\pi\nu} + G_\sigma \star (p_0\delta(\nu - \nu_0) + H(\nu - \nu_0)) .$$

Compared to P_{step}, we add a Dirac peak of power p_0 at the step frequency ν_0 and convolve this peak and the step function with a Gaussian kernel G_σ with standard deviation σ. Figure 5.6 illustrates the shape of this power spectrum.

This family of blue noise spectra has three interesting properties:

1. Aside from the single peak at ν_0, there are no oscillations.
2. The width and height of the peak can be controlled by adjusting the smoothing radius σ and the peak energy p_0.
3. The step spectrum P_{step} is included as a special case.

Not all spectra in this family are realizable, however, and due to the realizability conditions, the parameters cannot be adjusted independently. The main challenge therefore is to find combinations of the three parameters ν_0, p_0, and σ that are realizable and yield good sampling patterns.

The convolution with a Gaussian makes it harder to analyze P_{peak} analytically. We have therefore restricted ourselves to exploring this family of blue noise patterns empirically by searching for configurations for which ν_0 is as high as possible, the power spectrum is flat above and below ν_0, and structured aliasing is kept at an acceptable level by appropriate choice of p_0 and σ. Figure 5.7 shows some of the results we generated. We show the result for two different peak positions: $\nu_0 = 1.33\nu_{\max}$ in subfigures (a, b) and $\nu_0 = 1.66\nu_{\max}$ in subfigures (c, d). For each peak position, we show one result with a narrow

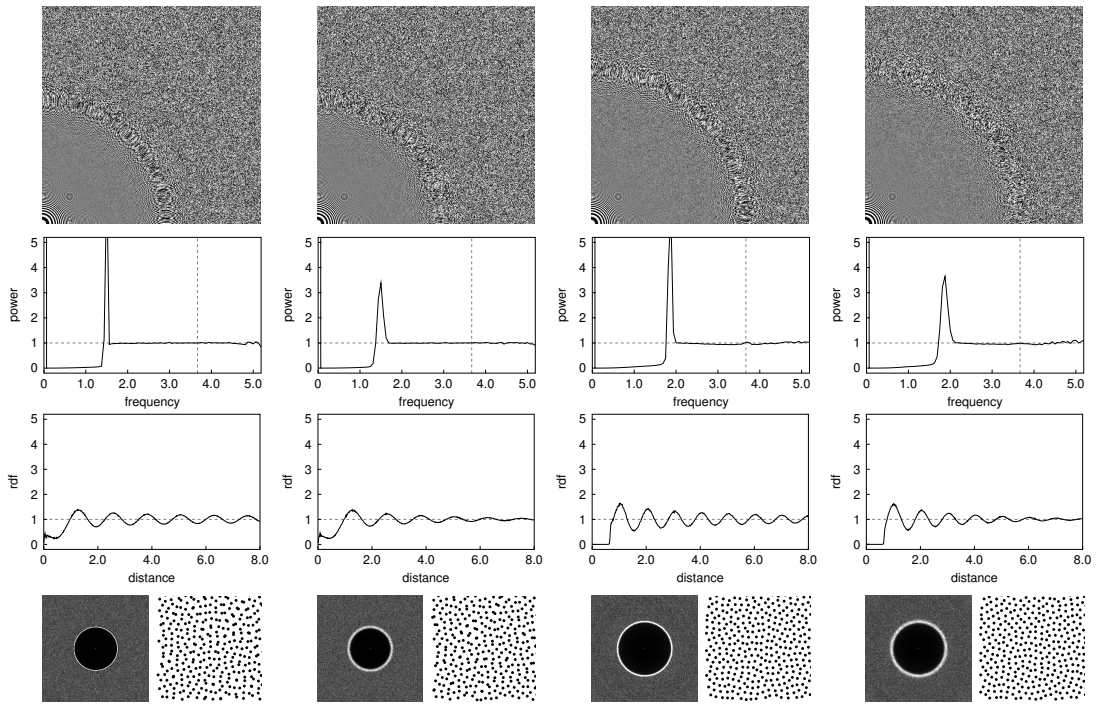


Figure 5.7: A selection of realizable blue noise patterns with a single peak. *(a)* We can increase ν_{eff} by moving energy to higher frequencies—in this case a peak of height $P(\nu_0) = 16$—but only at the cost of structured aliasing (visible patterns in the form of rings around ν_0). *(b)* Smoothing the peak to $P(\nu_0) \approx 4$ yields acceptable aliasing at the cost of a slightly reduced ν_{eff}. *(c)* The highest effective Nyquist frequency we can produce without deviating strongly from a flat spectrum in frequencies below or above ν_0. *(d)* The best compromise we found between a high ν_{eff} and structured aliasing. The stated parameters are for 4096 points; for a different number of points n, the parameters p_0 and σ must be scaled by $\sqrt{n/4096}$.

peak (small σ) and one with a wide peak (large σ). As in all other experiments in this paper, we generated sets of 4096 points in the unit torus and averaged the results over ten such sets.

The more we increase the peak height, the further we are able to push ν_0 and thus the effective Nyquist frequency. If we take this too far, however, strong aliasing can show up in the sampled image. The zone plate renderings at

the top of Figure 5.7 demonstrate that visible patterns emerge above $P(\nu_0) \approx 4$. We can reduce these artifacts by increasing the amount of smoothing σ. This has two effects: It decreases the height of the peak and increases its width, which means that aliasing is scattered over a wider range of frequencies. The highest effective Nyquist frequency we could produce while keeping $P(\nu)$ sufficiently flat above and below ν_0 is shown in column (c).

The best compromise we found between a high effective Nyquist frequency and structured aliasing is shown in (d); this is the spectrum we will use as the reference for *single-peak blue noise*. This configuration yields an effective Nyquist frequency that is comparable to classic blue noise patterns but gets rid of most of the high-frequency oscillation.

5.4 Evaluation

Our original motivation for blue noise spectra with low oscillation was to reduce the risk of low-frequency, structured aliasing, as shown in Figure 3.7.

Table 5.2 compares the effective Nyquist frequency ν_{eff}, the oscillation Ω and several spatial statistics of step blue noise and single-peak blue noise with other sampling patterns used in computer graphics. We have divided the sampling patterns in the table in three categories

- *Low Oscillation* for point sets whose power spectrum is relatively flat.
- *High Effective Nyquist* for point sets with a large zero region and large values of ν_{eff}.
- *Regular* for regular or semi-regular point distributions.

The effective Nyquist and oscillation measures from Section 3.3.4 are shown in the first two columns. We already mentioned in Section 3.3.4 that for traditional blue noise patterns, a high ν_{eff} comes at the cost of a high oscillation. The point sets constructed in this paper demonstrate that it is possible to achieve high values of ν_{eff} with little oscillation, so this "noise-aliasing tradeoff" [Dippé and Wold, 1985, Glassner, 1995] is not strict.

In the following sections we evaluate the performance of step and single-peak blue noise in three different sampling scenarios. First we consider *undersampling*, i.e., sampling signals with a bandwidth significantly higher than the effective Nyquist frequency of the sampling pattern. The other case we consider is *oversampling*: If we increase the sampling rate, all unbiased sampling patterns should converge to the true image function, but the rate of convergence depends crucially on the spectral behavior of the sampling pattern. To

	Method	ν_{eff}	Ω	$d_{\min}$	d_{avg}	R_c
Low Ω	Stochastic Sampling	0	0.05	0.01	0.47	1.73
	Jittered Grid	0.24	0.06	0.05	0.59	1.08
	Dart Throwing	0.58	1.52	0.76	0.80	1.07
	Step Blue Noise	0.58	0.01	0.09	0.64	0.91
High ν_{eff}	*Single-Peak Blue Noise*	0.86	1.44	0.55	0.80	0.77
	Kernel Density	0.88	2.14	0.43	0.86	0.78
	CCCVT Centroids	0.89	2.34	0.75	0.88	0.74
	El. Halftoning	0.89	2.49	0.74	0.88	0.77
	Farthest Point Optim.	0.90	4.64	0.93	0.93	0.86
Reg.	CVT Centroids	0.98	5.35	0.80	0.94	0.67
	Regular grid	0.95	14.77	0.93	0.93	0.66
	Hexagonal grid	1.01	12.38	0.99	0.99	0.58

Table 5.2: Comparison of several frequency and spatial statistics of sampling patterns. The last column marks methods that are largely (F)ree of structured aliasing and methods that are either (R)egular or show strong (O)scillations in their power spectrum.

measure this difference in residual noise, we study the performance of different sampling patterns at high sampling rates. Finally, we consider a standard test scene containing an infinite checkerboard.

5.4.1 Low Sampling Rate

We first evaluate the performance of different sampling patterns when *undersampling*. As a test image, we render a zone plate image; this is preferable to more realistic test images since it shows the response for a wide range of frequencies and aliasing effects are not masked by image features.

As a replacement for the standard zone plate function $z(r) = [1+\cos(\alpha r^2)]/2$, we use the following generalized form

$$z'(r) = \frac{1}{2}\big(1 + \cos[\nu_{\text{cut}}(\alpha r + \phi)]\big),$$

which allows us to limit the highest frequency ν_{cut} that can occur and configure the position r_{cut} of this cutoff frequency in the final image, by defining the

parameter α and the phase ϕ as follows

$$\alpha := \min(r/r_{\text{cut}}, 1), \qquad \phi := \max(r - r_{\text{cut}}, 0).$$

Figure 5.8 shows the result of rendering a zone plate image using step blue noise. For all zone plate renderings in this chapter, we have tiled toroidal sets of 4096 points over the image-plane and used a Lanczos filter with a support of width 4 for resampling. We focus on the low-frequency region since this is where step blue noise differs most from other sampling patterns with a low amount of oscillation. It is obvious that dart throwing leads to a lot more noise in the low-frequency region; images sampled with stochastic sampling or jittered grid would be even noisier. We also see that dart throwing actually yields better results for the higher image frequencies shown in Figure 5.8. This is also due to the shape of the power spectrum, which rises only slowly before the first peak.

Figure 5.9 compare the behavior of single-peak blue noise to FPO as a representative of conventional blue noise patterns. We chose FPO to emphasize the effect of heavy oscillations in the power spectrum; the effect is less severe but still visible for other blue noise methods with a relatively high amount of oscillation [Balzer et al., 2009, Schmaltz et al., 2010, Fattal, 2011]. Single-peak blue noise shows no structured aliasing beyond the narrow range of frequencies around $2\nu_{\text{eff}}$, and even low-frequency image content appears slightly cleaner.

5.4.2 High Sampling Rate

In most of this thesis we focused on undersampling, because the risk of undersampling is the primary reason for using blue noise sampling in the first place. It is still important to consider the limiting case, i.e., the image quality as we increase the sampling rate. We expect all sampling pattern to improve as the sampling rate increases, but there are measurable differences.

For this evaluation we use a more realistic test image of a 3D scene containing a textured sphere, edges, and smoothly shaded areas [Pharr and Humphreys, 2010], and used different sampling patterns to downsample a high-resolution input to a smaller output (see Figure 5.10). This is equivalent to image-plane sampling in a ray tracer, but easier to control since other error sources (texture filtering, secondary rays) can be ignored. As a quantitative error measure we use the *peak signal-to-noise ratio* (PSNR)

$$\text{PSNR} = 10 \cdot \log_{10} \frac{255^2}{\text{MSE}}, \qquad \text{MSE} = \text{mean square error}.$$

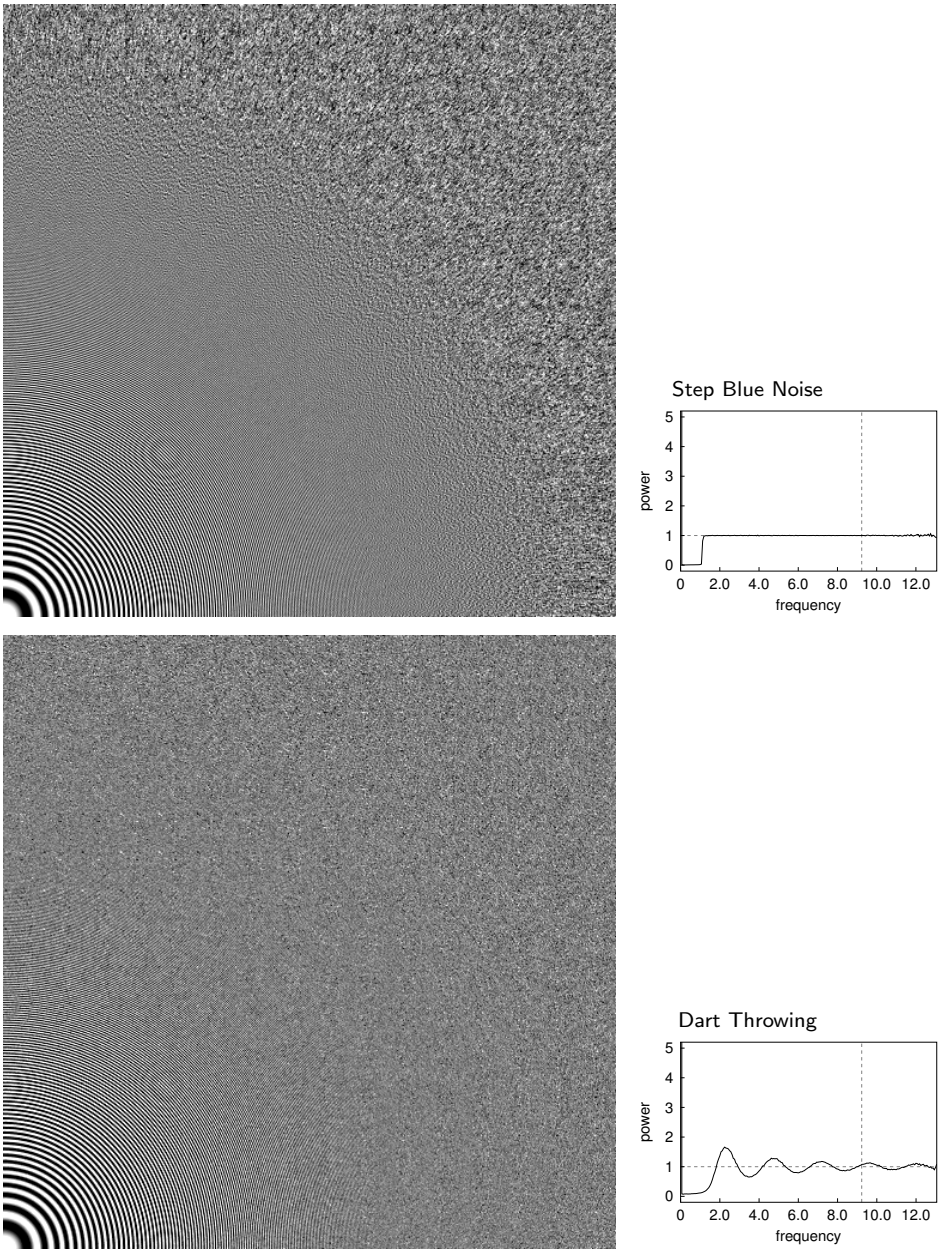


Figure 5.8: Step blue noise achieves a much cleaner rendition of low frequencies than dart throwing. Since the power spectrum of dart throwing rises slowly, it gives better results at intermediate frequencies.

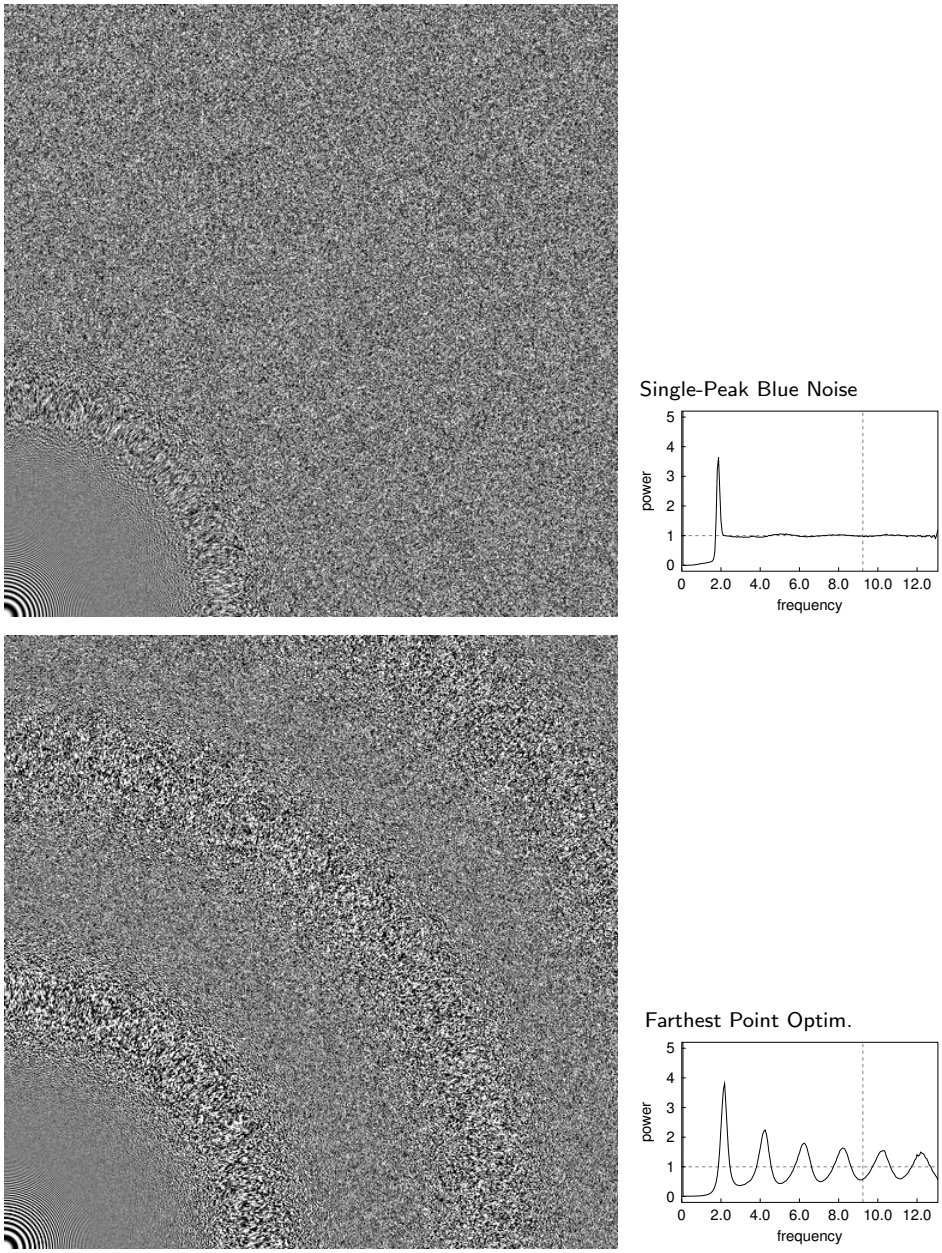


Figure 5.9: The difference between single-peak blue noise and conventional blue noise patterns is most obvious in the high-frequency region. Peaks in the power spectrum lead to moiré-like artifacts at certain image frequencies which can be avoided by flattening the spectrum. In the case of single-peak blue noise, these artifacts are therefore restricted to a small range of image frequencies.

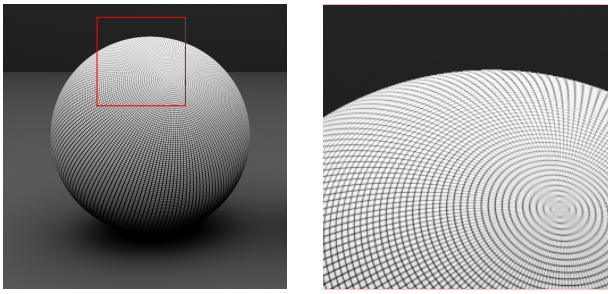


Figure 5.10: The scene used to evaluate the performance of sampling patterns at high sampling rates. *(Left)* Original high-resolution image. *(Right)* Detail. Compare also with Figure 1.1, which shows the moiré artifacts that occur when subsampling this image at an insufficient sampling rate.

A PSNR of 50 is comparable to high-quality JPEG compression. At low sampling rates the PSNR would not be an appropriate error measure since it only quantifies the total error, not whether it takes the form of moiré, structured aliasing, or unstructured noise; but in this experiment, we are intentionally oversampling, and the PSNR is only used to quantify the residual noise due to irregular sampling.

The main results are shown in Table 5.3. As expected, the best results are achieved by hexagonal grids, but the price for this performance are severe moiré artifacts at lower sampling rates, as we already saw in Figure 1.1. The worst results are obtained by stochastic sampling, which yields visible noise even at 128 samples per pixel.

In the *low oscillation* category, jittered grid initially performs worse than dart throwing but catches up at high sampling rates. Both are outperformed by step blue noise, however, which produces a significantly higher image quality at all sampling rates. In the *high effective Nyquist* category, the best results are obtained by CCCVT [Balzer et al., 2009] and Electrostatic Halftoning [Schmaltz et al., 2010], followed by our single-peak blue noise pattern. Both Kernel Density [Fattal, 2011] and FPO (Chapter 4) improve only slowly above 32 spp and eventually perform even worse than step blue noise. We have reproduced this behavior for different test scenes but have been unable to find a good explanation so far.

5.4.3 Checkerboard Sampling

As a final example, Figures 5.11 and 5.12 show the results of sampling an infinite checkerboard with several different sampling patterns. In each case,

	Method	*16 spp*	*32 spp*	*64 spp*	*128 spp*
Low Osci.	Stochastic Sampling	32.2	35.4	38.5	41.7
	Jittered Grid	38.2	43.5	48.9	53.9
	Dart Throwing	42.4	45.9	48.9	51.6
	Step Blue Noise	44.4	52.2	56.6	58.3
High ν_{eff}	*Single-Peak Blue Noise*	48.4	55.9	60.2	64.2
	Kernel Density	50.3	55.9	57	57.7
	CCCVT Centroids	51.9	60.8	64.8	69.9
	El. Halftoning	51.6	59.6	63	66.2
	Farthest Point Optim.	49.2	54.1	56.5	58.3
Reg.	CVT Centroids	52.6	57.2	58.3	59.3
	Hexagonal Grid	55.2	64.8	66.1	75.1

Table 5.3: PSNR as a function of the number of samples per pixel (spp). The values are averages over 10 images downsampled to 160×160 using the respective class of sampling pattern. The reference image is downsampled using a hexagonal grid at 512 spp.

the left column shows the rendered result and the right column an error image. The scene was sampled at 8 samples per pixel for all sampling patterns.

Figure 5.11 shows three sampling patterns that don't perform well for this test scene. Both the hexagonal and the rectangular grid result in strong moiré patterns as we get closer to the horizon. The stochastic sampling pattern in the last row naturally doesn't result in moiré patterns, but the whole image is very noisy.

In Figure 5.12 we compare the two sampling patterns constructed in this chapter to a CCCVT pattern, which is a state-of-the-art blue noise pattern. The performance of all three irregular sampling patterns is relatively similar for this scene and at this sampling rate; this is also reflected in the similar PSNR values. For CCCVT and single-peak blue noise, the effect of the peak in the power spectrum is again visible in the sampled image, as a fine horizontal line close to the horizon; the location is marked with a small arrow. Step blue noise prevents this artifact at the cost of a slightly higher overall noise level.

5.4.4 Comparison with Related Algorithms

As mentioned in Section 5.2, algorithms for constructing point sets matching a given spectrum have already been proposed by other researchers. In our tests,

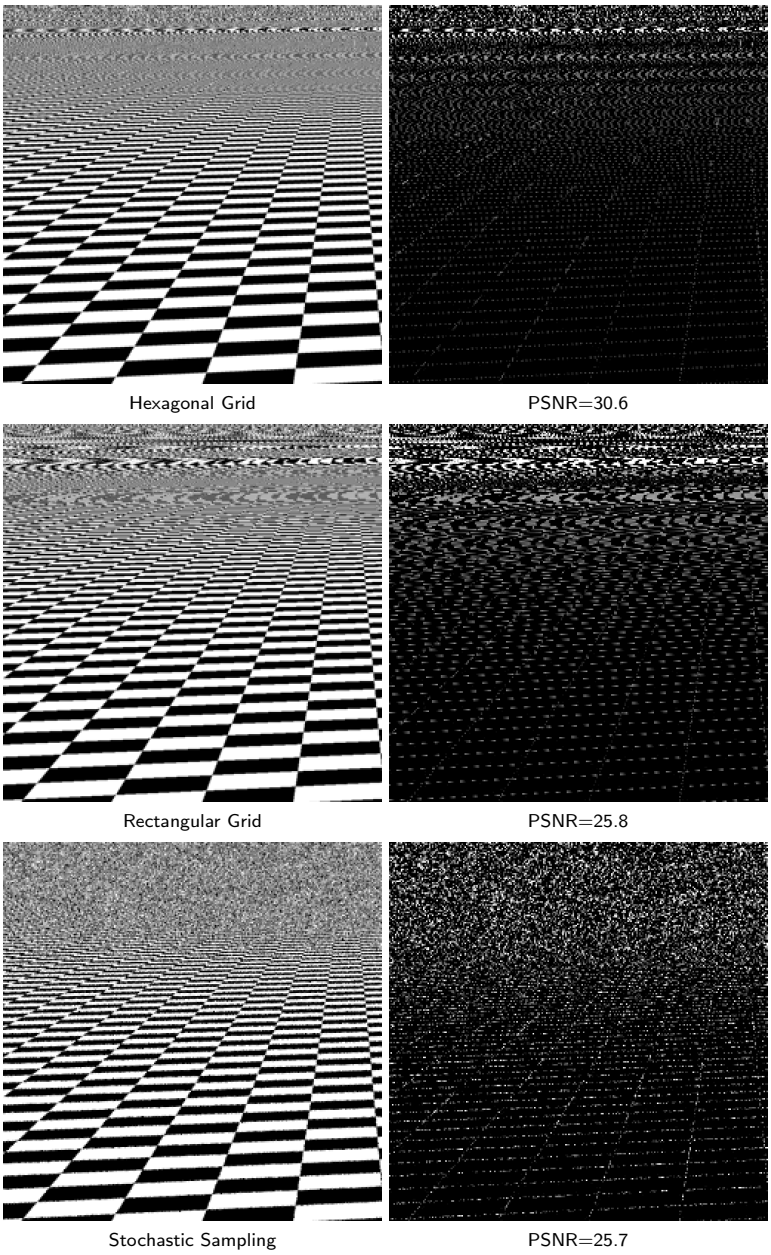


Figure 5.11: Rendering of a checkerboard image (left) and error image (right). The images are rendered at 8 samples per pixel.

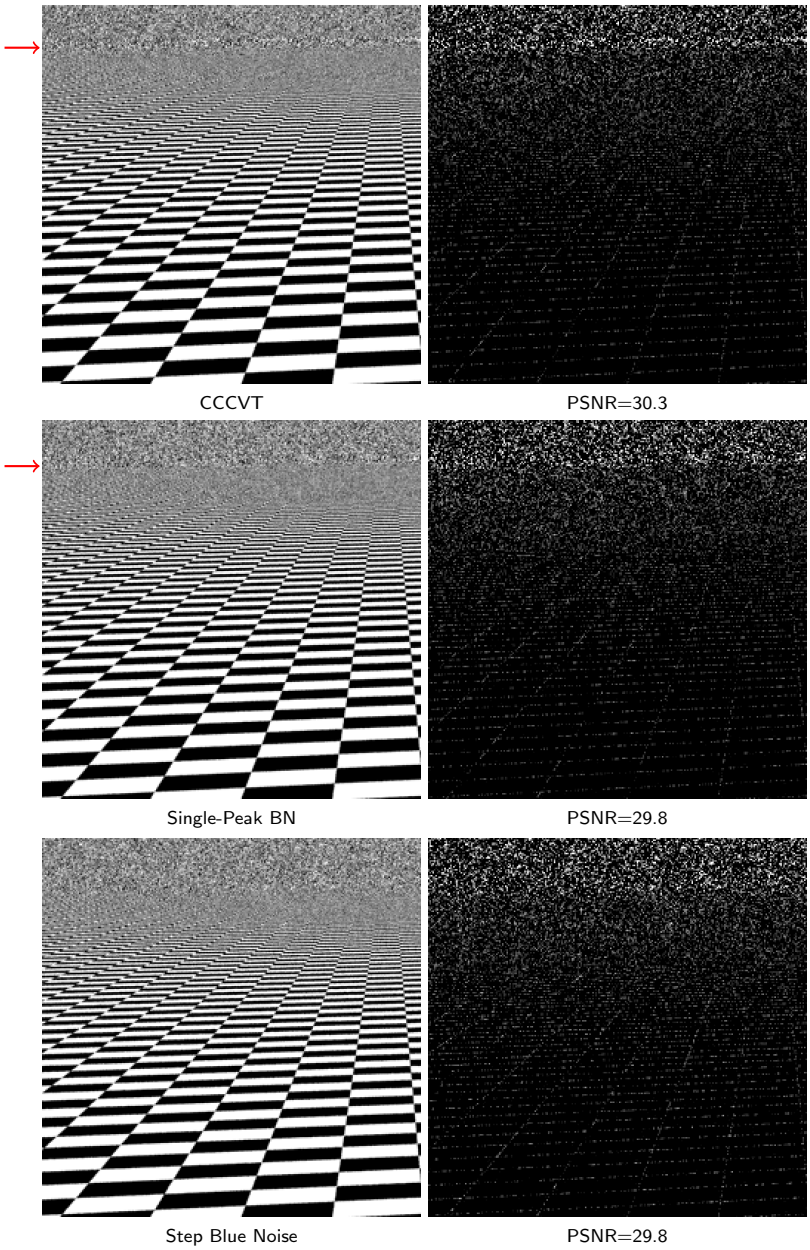


Figure 5.12: Rendering of a checkerboard image (left) and error image (right). The images are rendered at 8 samples per pixel.

the annealing approach by Rintoul and Torquato [1997] converged only slowly and didn't give good results for reasonably large point sets. In this section, we compare our results to those achieved using the algorithm by Zhou et al. [2012]. But since the primary goal of this chapter is to show that certain types of blue noise are realizable, not to find the best possible way to construct them, we restrict ourselves to a brief, qualitative comparison.

Figure 5.13 shows the matching result of our algorithm for the step and the single-peak blue noise profile in comparison to the result by Zhou et al.'s algorithm. The step power spectrum is matched slightly better by our algorithm ($\Omega = 0.01$) than by the method by Zhou et al. ($\Omega = 0.05$), which shows a certain amount of ripple in the high-frequency region.

The single-peak spectrum is more challenging and was matched accurately by our algorithm whereas the method by Zhou et al. had problems adapting to the target spectrum. This is easy to see in the radial power spectra in Figure 5.13: The result by Zhou et al. is non-zero in the low frequencies and oscillates more strongly in the high frequencies.

The average computation time using the CUDA-based implementation by Zhou et al. was 15.7 s on an nvidia Quadro 4000, while our CPU implementation took an average 69.7 s on a quad-core 2.8 GHz CPU.

Judging from the anisotropy plots, it seems as if both methods produce slightly anisotropic point sets, most noticeably at low frequencies and at the frequency corresponding to the step in the power spectrum. This behavior can at least partly be attributed to the standard definition of anisotropy [Ulichney, 1993, Lagae and Dutré, 2008] which measures the *relative* variance and therefore includes a division by the radial power. The anisotropy is therefore ill-defined if the power spectrum is very close to zero, which is the case for our point sets.

5.5 Discussion

In this chapter we have discussed several contributions to the theory of blue noise sampling.

The first contribution is a new algorithm for constructing point distributions matching a given power spectrum or radial distribution function. Similar algorithms have been proposed while our paper was under review [Zhou et al., 2012, Öztireli and Gross, 2012], and a detailed comparison is an interesting open question. We performed a limited comparison to the approach by Zhou et al. in Section 5.4.4 and were able to demonstrate noticeably better results

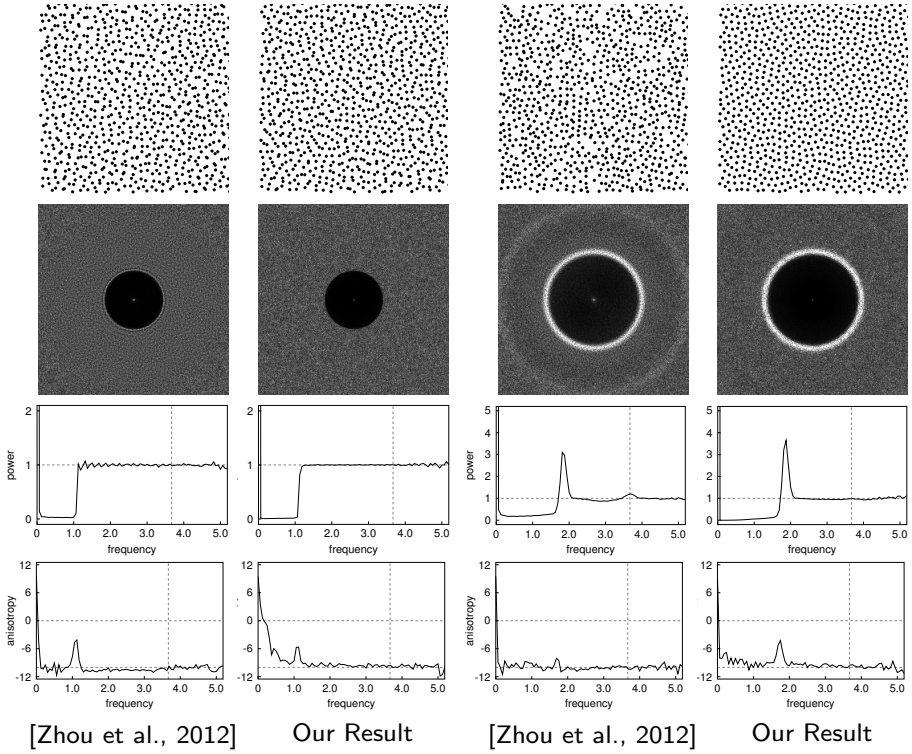


Figure 5.13: Comparison of matching P_{step} *(left)* and P_{peak} *(right)* using the method by Zhou et al. [2012] and our approach. Our method matches achieves a slightly cleaner *step* spectrum and a significantly better *single-peak* spectrum.

with our algorithm, at least for generating step blue noise and single-peak blue noise distributions.

The second contribution is a theoretical investigation of blue noise sampling, in particular the question: to what extent can we influence the spectral properties of a sampling patterns to our advantage? The relationship between the power spectrum and the radial distribution function leads to a strong constraint on the *realizability* of power spectra and limits the range of spectra that can be achieved by point distributions.

Finally, we demonstrated as a third contribution how to design blue noise patterns by prescribing the shape of the power spectrum. The main difficulty with this approach is to find functional models for blue noise that are easy to parametrize and can be handled analytically. As two simple examples we studied step blue noise and single-peak blue noise, and demonstrated that the resulting sampling patterns outperform many traditional sampling patterns

used in graphics. What is remarkable about these results isn't so much the performance of the new sampling patterns but the fact that they were obtained not by placing elaborate geometric constraints on the point set. Instead, we used a generic construction algorithm to derive them from a specification of their spectral behavior.

Most research on blue noise sampling has focused on geometric properties of the sampling patterns: Poisson disk sampling, Lloyd relaxation, jittered sampling, or more recently *maximal* Poisson disk sampling are all based on geometric notions of the sample distributions. We have shown in this chapter that if we are only interested in image-plane sampling, we can achieve comparable, or even better, results by focusing on the *spectral* properties of the sampling patterns. We are confident that we have only scratched the surface of what is possible using this approach, and that further research will lead to even better results.

Chapter 6

Conclusion and Outlook

For the last 30 years, research on irregular sampling in computer graphics has focused almost exclusively on geometric methods for constructing sampling patterns. This approach has been very successful: a variety of efficient algorithms for constructing sampling patterns are now available, some of them producing sampling patterns with excellent blue noise properties.

One major limitation of this geometric approach remains, however: it doesn't allow us to influence the spectral characteristics of sampling patterns directly. These spectral properties are crucial because they predict the performance of a sampling pattern and allow us to make guarantees about the visual appearance of aliasing in the sampled image.

In the introduction, we listed open questions regarding irregular sampling. To what extent did the preceding chapters answer these questions?

What exactly is the effect of blue noise sampling on the sampled image? Classical sampling theory typically simplifies this question by assuming that the signal being sampled is bandlimited or can be bandlimited by prefiltering the signal. In graphics, we often have to live with aliasing and therefore want to control the visual appearance of this aliasing. We discussed this question in detail in Chapter 3, in particular the relationship between the shape of the power spectrum and the spectral characteristics and appearance of aliasing.

Are all efficient sampling patterns also Poisson disk patterns? We have studied this question from two different angles. In Chapter 4 we studied the question whether all Poisson disk patterns are also efficient sampling patterns, and found that this is true only to a limited extent: for very high Poisson disk radii, as achieved by farthest point optimization, the sample points are coerced into an arrangement with a relatively high amount of trans-

lational order, which leads to strong oscillations in the power spectrum. In Chapter 5 we demonstrated that dropping the minimum distance requirement can be useful to decorrelate the point set: this is how the single-step blue noise pattern achieves its resilience to aliasing artifacts.

Can we derive irregular sampling patterns from a specification of their spectral behavior? As we demonstrated in Chapter 5, it is in fact possible to derive efficient sampling patterns from first principles, by demanding isotropy and a power spectrum with a blue noise spectrum. In particular, no assumptions about geometric properties of the final sampling pattern are necessary.

What is the most desirable blue noise sampling pattern? Under what circumstances can it be realized? We discussed the question of realizability in Chapter 5 and have made some progress in the search for the most desirable sampling pattern. But the final answer to this question remains elusive. The main difficulty at the moment is the problem of mathematically modeling suitable power spectra. We have approached this problem mostly empirically, but even though the "step" and "single-peak" models discussed in the previous chapter are insightful, they only cover a tiny range of possible power spectra. Better models of blue noise spectra are one obvious area for extending the research of this thesis.

We discuss two other interesting research directions in the following paragraphs.

Adaptive Sampling In many applications, the process of taking samples is very expensive; in ray tracing, for example, computing the value of a single sample involves the numerical evaluation of complex integral equation. In such cases, the goal is often to maximize the final image quality given a certain computational budget. One way of approaching this problem is to vary the sampling rate adaptively with the local frequency content, which can improve the image quality by focusing the sampling effort on difficult parts of the image. Throughout this thesis we assumed a constant sampling density n, but it is possible to generalize most of the results to spatially varying densities $n(\boldsymbol{x})$ by warping the distance metric according to the desired density [Zhou et al., 2012]. But this is only the first and easiest step; two harder problems still await a satisfactory solution.

The desired point density is generally not known a priori, except for a few special applications such as stippling or halftoning, so adaptive sampling usually requires that the sampling pattern can be refined on-the-fly. Ideally we would like to adaptively add new samples to an existing sampling pattern

while preserving its spectral properties. A few methods for generating irregular point distributions incrementally are known, for example low-discrepancy sequences [Niederreiter, 1992] and hierarchical blue noise sampling [McCool and Fiume, 1992, Ostromoukhov et al., 2004, Kopf et al., 2006]. But how to construct sampling patterns with arbitrary spectra incrementally is still unsolved question.

The second problem is specific to image-plane sampling, where the required sampling rate isn't known in advance. To prevent aliasing, we would like to adjust the sampling rate adaptively, based on the local frequency content of the image. The standard approach to adaptive image-plane sampling is to start with a low-resolution sampling pattern, which is used to estimate the local variance of the image, and take additional samples in regions of high variance. The choice of this initial sampling pattern faces the same tradeoffs as other sampling tasks: a completely random distribution is bias-free but leads to unnecessarily high variance estimates; a regular distribution is more efficient but sensitive to aliasing.

Using blue noise samples for more reliable aliasing detection might be possible, but hasn't been studied so far. The main idea behind this approach would be to exploit the fact that blue noise sampling affects the spectral distribution of aliasing in a predictable way: since high frequencies are mapped to broadband noise, local frequency analysis of blue noise sampled images might give a reliable way to detect aliasing in oversampled images and control the sampling rate. To our knowledge, no study in this direction has been performed yet.

Reconstruction As explained in Section 2.4, we assumed that the resampling process that computes pixels from the irregular samples is based on convolution with a low-pass filter. This is a fundamental assumption that underlies the whole notion of blue noise sampling: we motivated the particular shape of the blue noise spectrum using the idea of lowpass filtering in Chapter 3. In this sense, blue noise sampling patterns are especially well-tuned for convolution-based reconstruction.

Over the years, several alternative types of reconstruction and interpolation algorithms have been proposed, most notably the POCS algorithm [Combettes, 1993, Stasiński and Konrad, 2002] and variants of the frame algorithm [Gröchenig, 1992, Feichtinger and Gröchenig, 1994]. For the frame algorithm it can be shown that perfect reconstruction of bandlimited functions is possible from *any* set of samples that has a sufficiently small coverage radius—so even random sampling patterns with a white noise spectrum can be used for per-

fect reconstruction, with the maximum reconstructible frequency depending on the largest hole in the sampling pattern.

The use of reconstruction algorithms and their interaction with blue noise sampling is almost completely unexplored in computer graphics. Initial experiments with these reconstruction methods suggests that they perform better than normalized convolution at low sampling rates and for very irregular sampling patterns, but at higher sampling rates, they are slower and the image quality is actually worse since their iterative nature easily lead to ringing artifacts. A critical evaluation of these algorithms for graphics applications is an important open problem.

Appendix A

Energy and Power Spectrum

There are two common measures to characterize the strength of a signal as a function of the frequency: the *energy spectrum* for signals with finite extent and the *power spectrum* for signals with infinite extent [Oppenheim and Verghese, 2010]. Since sampling problems involve infinite signals (sampling combs, sine waves, etc.), the power spectrum is the natural measure for studying such problems.

The *total energy* of a signal $x(t)$ is defined as the integral

$$E_x = \int_{\infty}^{\infty} |x(t)|^2 \, \mathrm{d}t. \tag{A.1}$$

If $E_x < \infty$, Parseval's theorem tells us that the Fourier transform $\hat{x}(\nu)$ of x has the same total energy, i.e.,

$$E_x = \int_{-\infty}^{\infty} |x(t)|^2 \, \mathrm{d}t = \int_{-\infty}^{\infty} |\hat{x}(\nu)|^2 \, \mathrm{d}\nu.$$

It therefore makes sense to call $|\hat{x}(\nu)|^2$ the *energy spectrum* or the *energy spectral density* of x

$$E_x(\nu) = |\hat{x}(\nu)|^2. \tag{A.2}$$

For many infinite signals such as sine waves and periodic functions the integral in Eq. (A.1) does not converge, so neither the total energy nor the energy spectrum are defined. This motivates the definition of the *power spectrum* which is defined for a much wider range of signals. To define the power

spectrum, we consider the truncated signal

$$x_T(t) = \begin{cases} x(t) & \text{if } |t| \leq T \\ 0 & \text{otherwise} \end{cases}$$

and its Fourier transform

$$\hat{x}_T(\nu) = \int_{-T}^{T} x(t) e^{-2\pi i \nu t} \, \mathrm{d}t.$$

The *power spectrum* or *power spectral density* (PSD) is obtained by taking the limit

$$P_x(\nu) = \lim_{T \to \infty} \frac{1}{2T} |\hat{x}_T(\nu)|^2 \tag{A.3}$$

provided it actually exists. (The notation S_{xx} is also common for the power spectrum of x.) This definition makes it clear that the power spectrum measures the energy per unit time.

Spectrum of Random Signals and Processes In the analysis of sampling, we model both the image being sampled and the sampling pattern as random signals. In this case, the definition of the power spectrum includes an additional average over all realizations of the signal

$$P_x(\nu) = \lim_{T \to \infty} \mathbb{E}\left[\frac{1}{2T} |\hat{x}_T(\nu)|^2 \right]. \tag{A.4}$$

For the signals we are interested in, this limit always exists. This follows from the *Wiener-Kinchin theorem*, which guarantees convergence for *wide-sense stationary* (WSS) signals. A signal is WSS if its mean m_x and autocorrelation C_x are stationary, i.e., if

$$m_x = \mathbb{E}[x(t)], \qquad \text{and} \qquad C_x(r) \equiv \mathbb{E}\big[x(t)x(t+r)\big].$$

are independent of t. In this case, the Wiener-Kinchin theorem states that P_x exists and equals the Fourier transform of the autocorrelation

$$P_x(\nu) = \lim_{T \to \infty} \mathbb{E}\left[\frac{1}{2T} |\hat{x}_T(\nu)|^2 \right] = \mathcal{F}\big[C_x(r)\big]$$

This relationship between the autocorrelation and the power spectrum is often the easiest way to compute P_x analytically.

Spectrum of Products If we consider the product of two arbitrary signals $Z(t) = X(t) \cdot Y(t)$, the autocorrelation of Z is

$$C_Z(\tau) = \mathbb{E}[Z(t) \cdot Z(t+\tau)].$$

This can be simplified if X and Y are statistically independent

$$C_Z(\tau) = \mathbb{E}[X(t)X(t+\tau] \cdot \mathbb{E}[Y(t)Y(t+\tau)] = C_X(\tau)C_Y(\tau),$$

and since the power spectrum and the autocorrelation form a Fourier transform pair we have

$$P_Z(\nu) = P_X \star P_Y(\nu). \tag{A.5}$$

Spectrum of Uncentered Signals For any signal X with mean m_X, the autocorrelation and power spectrum can be decomposed as follows

$$C_X(\boldsymbol{r}) = |m_X|^2 + \check{C}_X(\boldsymbol{r}), \qquad P_X(\boldsymbol{\nu}) = |m_X|^2\delta(\boldsymbol{\nu}) + \check{P}_X(\boldsymbol{\nu}), \tag{A.6}$$

where $\check{C}_X$ and $\check{P}_X$ denote the autocorrelation and spectrum of the centered signal $\check{X}(\boldsymbol{x}) = X(\boldsymbol{x}) - m_X$.

Appendix B

Overview of Sampling Patterns

This appendix gives an overview and compares several classes of point distributions that have been proposed for sampling in computer graphics. The purpose of this comparison is to collect in one place several qualitative and quantitative measures. Geometrically, the main property of most sampling patterns is that they are *uniform* and *irregular*. Uniformity ensures that the whole domain is covered with samples, and irregularity reduces the visibility of aliasing artifacts.

We already introduced three quantities that measure the uniformity of a point distributions in Chapter 4.1: the *nearest-neighbor distance* $d_{\min}$ and d_{avg} and the *coverage radius* R_c. A point distribution can be considered uniform if $d_{\min}$ and d_{avg} are large, which implies that there are no clusters, and R_c is small, which implies that there are no "holes".

Irregularity, on the other hand, remains a rather vague term, and no numerical measure for irregularity is widely used in graphics. In the following, we use an irregularity measure called the *bond-orientational order* Q^n which measures the local rotational symmetry of a point set. The measure was originally proposed in the context of disk packings [Kansal et al., 2000], but we have found this to be a useful geometric way to assess the irregularity of sampling patterns.

B.1 Bond-Orientational Order

In general, the bond-orientational order Q^n measures how close the *angles* between neighboring points are to a perfect n-fold symmetry. In the case of

n-fold symmetry, each point has n neighbors arranged around the point in $360°/n$ increments. In two dimensions, the most common choices for n are $n = 3, 4, 6$, in which case Q^n measures how similar a point set is to a triangular, rectangular, or hexagonal lattice.

We first define the *local* bond-orientational order q_j^n for a single point $\boldsymbol{x}_j \in S$ as follows:

$$q_j^n = \frac{1}{|\mathcal{N}_j|} \left| \sum_{k \in \mathcal{N}_j} e^{in\theta_{jk}} \right|. \tag{B.1}$$

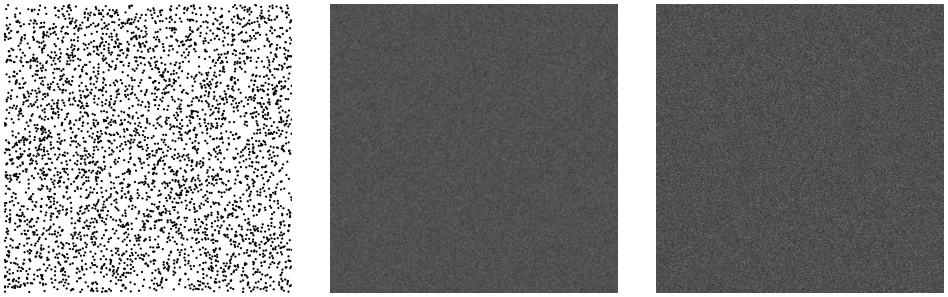
The set $\mathcal{N}_j \subset S$ denotes a suitable neighborhood of $\boldsymbol{x}_j$ and $\theta_{jk} = \angle(\boldsymbol{x}_k - \boldsymbol{x}_j)$ is the angle in the direction of $\boldsymbol{x}_k$. The intuition behind the definition of q_j^n is as follows: If the neighbors $\boldsymbol{x}_k \in \mathcal{N}_j$ are arranged on a regular n-gon around $\boldsymbol{x}_j$ the complex numbers $e^{in\theta_{jk}}$ add constructively, and q_j^n attains its maximum value of 1. Otherwise, they partially cancel each other out and we have $q_j^n < 1$. The global bond-orientational order can then be defined as a simple average over all N points

$$Q^n = \frac{1}{N} \sum_{j=1}^{N} q_j^n. \tag{B.2}$$

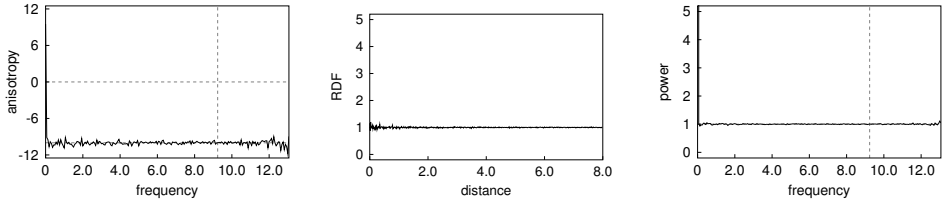
Since hexagonal symmetry often occurs when optimization methods are used to generate point distributions, we use Q^6 as our irregularity measure. A natural choice for the neighborhood $\mathcal{N}_j$ are the Delaunay neighbors of $\boldsymbol{x}_j$. As shown on the following pages, Q^6 generally increases as the point distributions become more uniform, and high values of Q^6 indicate the existence of regions in the point set with hexagonal symmetry. We have found a threshold of 0.5 useful for detecting disordered point sets; for $Q^6 > 0.5$, a more detailed symmetry analysis using either the power spectrum or the autocorrelation is useful.

B.2 Stochastic Sampling

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

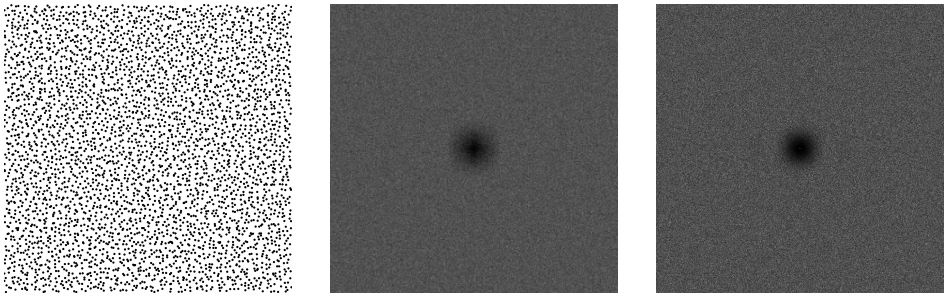
$Q^6 = 0.36$	$d_{\min} = 0.01$	$\nu_{\text{eff}} = 0$
$R_c = 1.73$	$d_{\text{avg}} = 0.47$	$\Omega = 0.05$

Notes

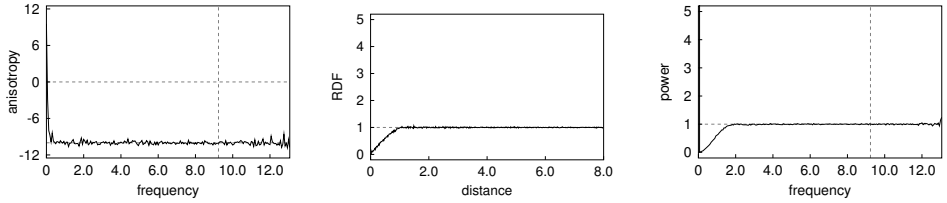
See [Cook, 1986, Dippé and Wold, 1985]. Also known as a *random* or *Poisson* distribution. Because the point positions are completely uncorrelated, the spectrum and autocorrelation are constant. The resulting point set contains many clusters of points (d_{avg} is low) and large holes (R_c is high). A low bond-orientational order Q^6 reflects the irregularity of the point set.

B.3 Jittered Grid

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

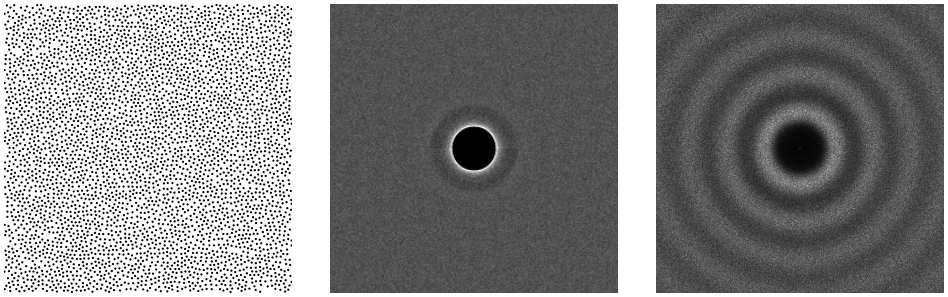
$Q^6 = 0.37$	$d_{\min} = 0.05$	$\nu_{\text{eff}} = 0.24$
$R_c = 1.08$	$d_{\text{avg}} = 0.59$	$\Omega = 0.06$

Notes

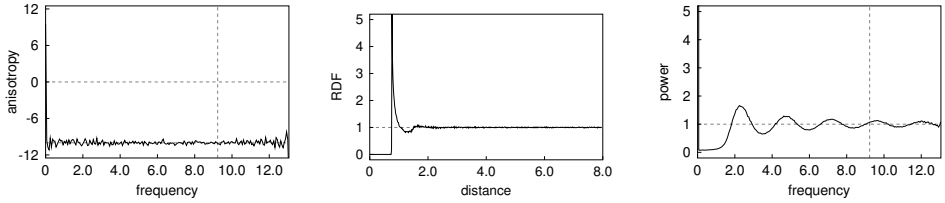
See [Cook, 1986, Dippé and Wold, 1985]. The point set is obtained by randomly perturbing a regular grid. This jitter process prevents large clusters, which leads to a more uniform point distribution, so d_{avg} is higher than for stochastic sampling. Relatively large empty regions are still possible, so R_c is large compared to other methods. The orientational order is as low as for stochastic sampling.

B.4 Dart Throwing

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

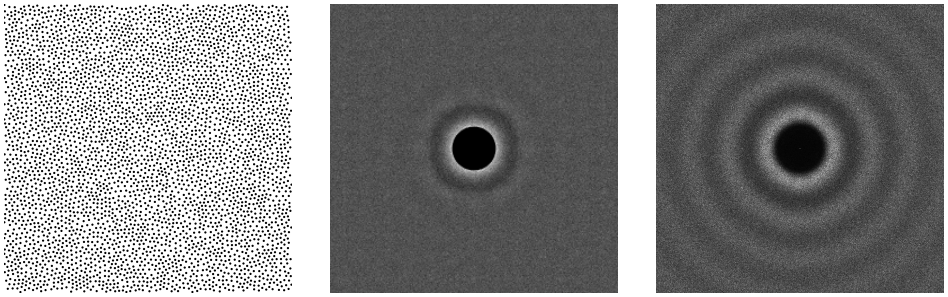
$Q^6 = 0.43$	$d_{\min} = 0.77$	$\nu_{\text{eff}} = 0.59$
$R_c = 1.07$	$d_{\text{avg}} = 0.81$	$\Omega = 1.52$

Notes

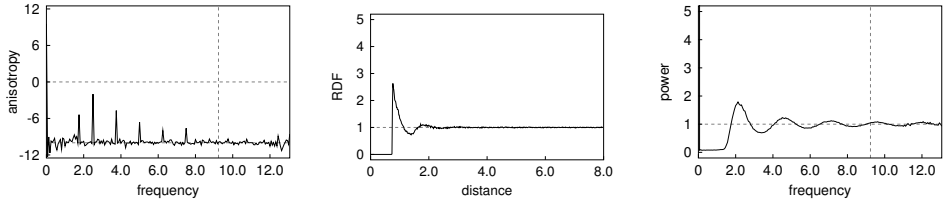
See [Cook, 1986, Lagae et al., 2010]. Also known as a *Poisson disk* pattern, since small disks placed at the points do not overlap. This constraint on the nearest-neighbor distance leads to a higher value for d_{avg}, but large holes are still possible (R_c high). The orientational order has increased noticeably compared to stochastic and jittered sampling.

B.5 Best Candidate/FPS

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

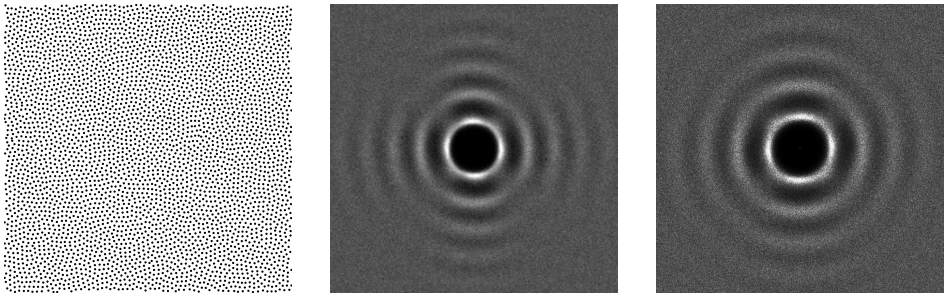
$Q^6 = 0.47$	$d_{\min} = 0.75$	$\nu_{\text{eff}} = 0.77$
$R_c = 0.76$	$d_{\text{avg}} = 0.84$	$\Omega = 1.32$

Notes

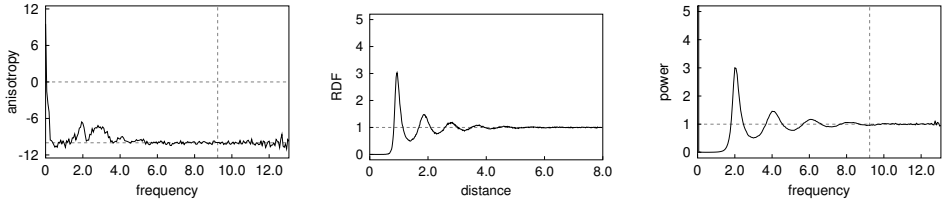
See [Mitchell, 1991], but almost identical results are obtained by the *farthest point strategy* [Eldar et al., 1997, Kanamori et al., 2011]. Similar to dart throwing, but new points are incrementally added to the largest empty region, which significantly reduces the coverage radius R_c. Small regular regions can result, which is reflected by a relatively high value for Q^6 and small peaks in the anisotropy.

B.6 Kernel Density Blue Noise

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

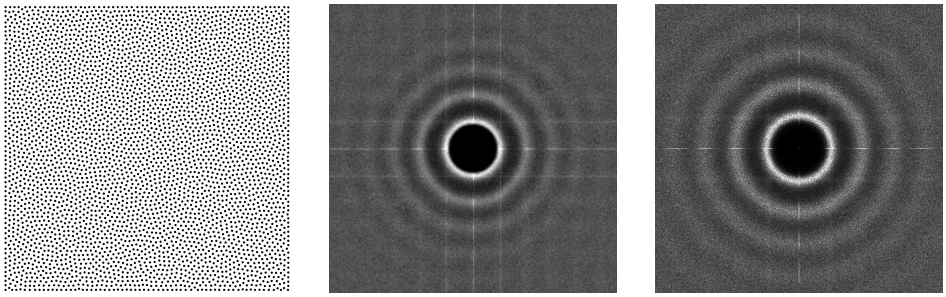
$Q^6 = 0.50$	$d_{\min} = 0.43$	$\nu_{\text{eff}} = 0.88$
$R_c = 0.78$	$d_{\text{avg}} = 0.86$	$\Omega = 2.14$

Notes

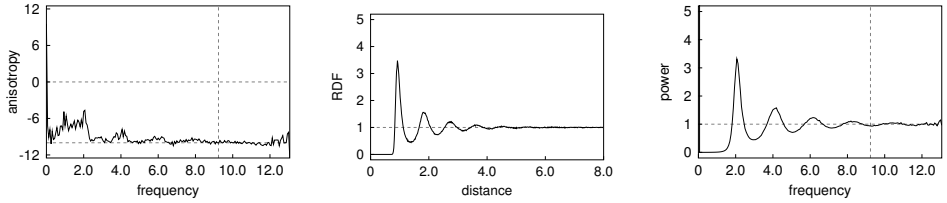
See [Fattal, 2011]. A recent particle-based method for efficiently generating blue noise patterns. The generated point sets exhibit a slight anisotropy which is not visible in point distribution but evident in power spectrum and autocorrelation. This seems to be a side-effect of the hierarchical grid used to generate the point distributions.

B.7 Electrostatic Halftoning

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

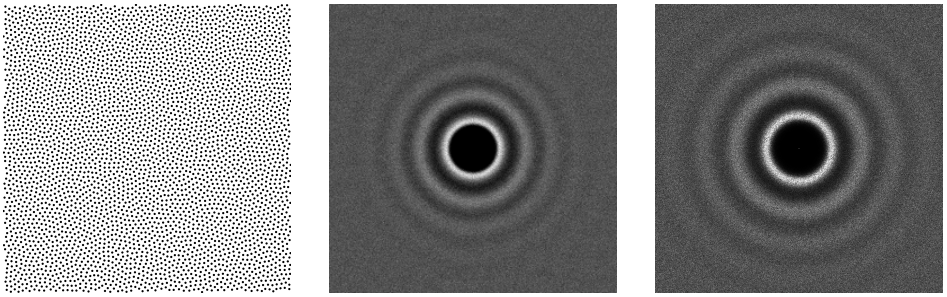
$Q^6 = 0.52$	$d_{\min} = 0.74$	$\nu_{\text{eff}} = 0.89$
$R_c = 0.77$	$d_{\text{avg}} = 0.88$	$\Omega = 2.49$

Notes

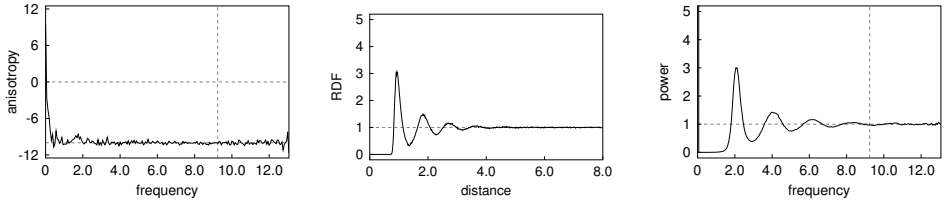
See [Schmaltz et al., 2010]. Models points as interacting particles and prevents regularity by adding a random force field that favors irregular point distributions. The resulting point sets achieve a very good compromise between uniformity and irregularity and excellent blue noise properties. The slight anisotropy is due to non-toroidal boundary conditions.

B.8 CCCVT

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

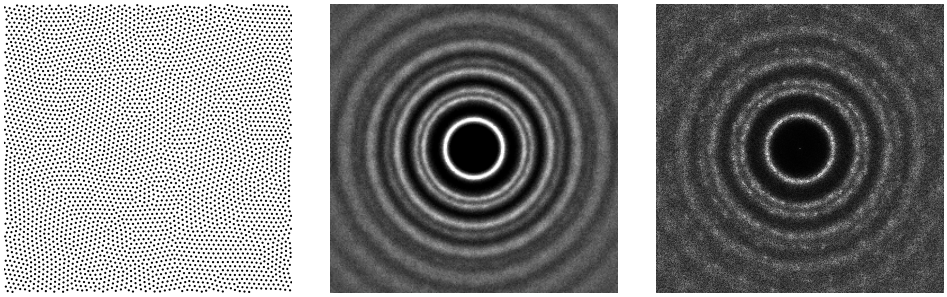
$Q^6 = 0.54$	$d_{\min} = 0.75$	$\nu_{\text{eff}} = 0.89$
$R_c = 0.74$	$d_{\text{avg}} = 0.88$	$\Omega = 2.34$

Notes

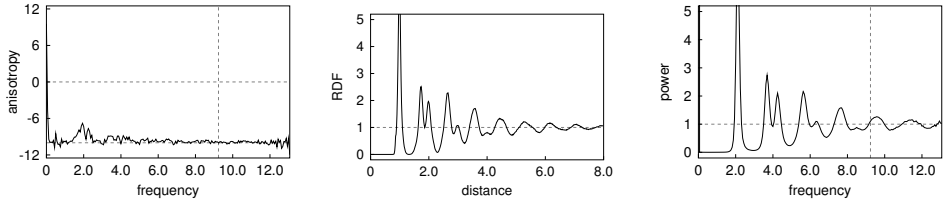
See [Balzer et al., 2009]. The point distribution has the special property that the points form a centroidal Voronoi tessellation in which all Voronoi regions have the same area. The results are very similar to Electrostatic Halftoning: the points achieve a very good compromise between uniformity and irregularity.

B.9 Centroidal Voronoi Tessellation

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

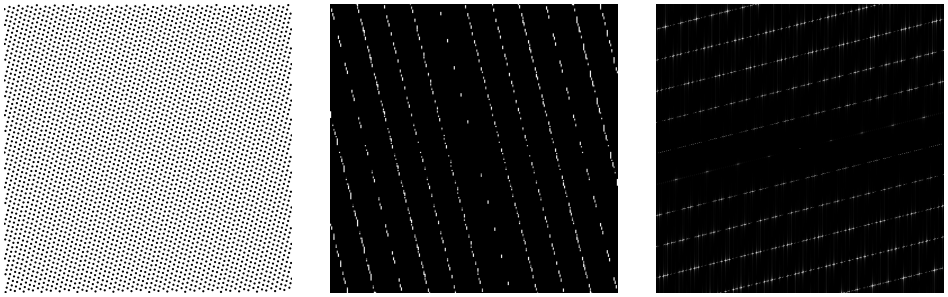
$Q^6 = 0.85$	$d_{\min} = 0.80$	$\nu_{\text{eff}} = 0.98$
$R_c = 0.67$	$d_{\text{avg}} = 0.94$	$\Omega = 5.35$

Notes

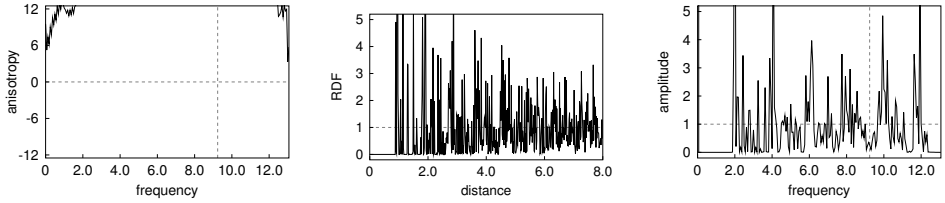
See [Lloyd, 1982, McCool and Fiume, 1992, Du et al., 1999]. An iterative method that "relaxes" a given input point set. The algorithm converges towards partially regular arrangements consisting of hexagonal patches. Since the orientation of these patches is random, the regularity is not immediately in the power spectrum or autocorrelation which perform global averaging, but is reflected in the raised level of the anisotropy and the high bond-orientational order.

B.10 Low Discrepancy

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

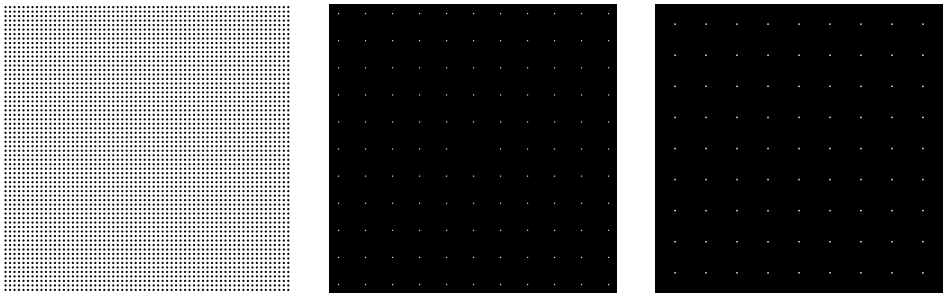
$Q^6 = 0.66$	$d_{\min} = 0.90$	$\nu_{\text{eff}} = 0.96$
$R_c = 0.66$	$d_{\text{avg}} = 0.92$	$\Omega = 6.83$

Notes

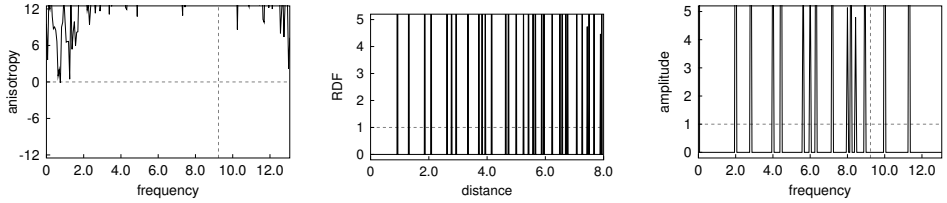
See [Grünschloß and Keller, 2009]. One example of a *low-discrepancy* sequence, which achieves a very low value for a geometrical uniformity measure called the *star discrepancy*. The points are arranged on parallel, slanted lines, so they are very regular; this regularity is reflected in the power spectrum and autocorrelation and means that strong aliasing would occur when sampling regular patterns with a similar angle. Optimizing purely for low discrepancy values generally cannot prevent such regularities, which is the reason why this measure is considered unsuitable for measuring the quality of point sets for image-plane sampling [Dobkin and Mitchell, 1993].

B.11 Regular Grid

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

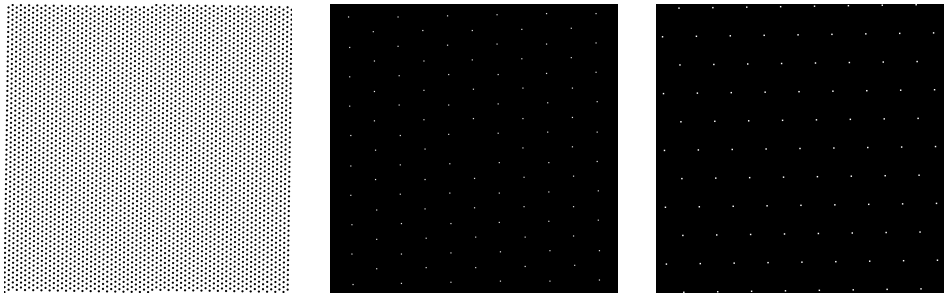
$Q^6 = 0.33$	$d_{\min} = 0.93$	$\nu_{\text{eff}} = 0.95$
$R_c = 0.66$	$d_{\text{avg}} = 0.93$	$\Omega = 14.77$

Notes

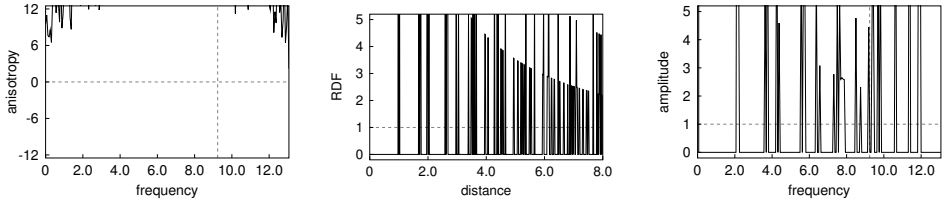
Q^6 detects hexagonal symmetry only, so it doesn't correctly detect the regularity of this rectangular grid. Since none of the algorithms for constructing point sets in graphics converge towards rectangular grids, we have not found this to be a disadvantage.

B.12 Hexagonal Grid

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

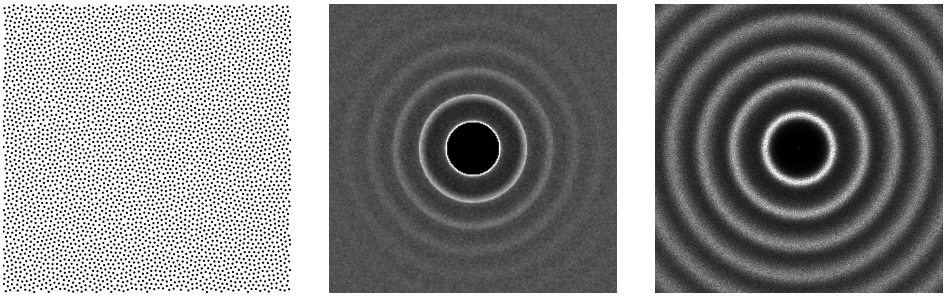
$Q^6 = 1.00$	$d_{\min} = 1.00$	$\nu_{\text{eff}} = 1.02$
$R_c = 0.58$	$d_{\text{avg}} = 1.00$	$\Omega = 12.38$

Notes

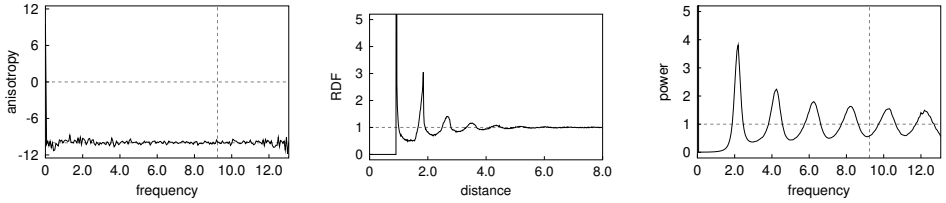
Since a regular hexagonal grid cannot completely fill the unit square, we show what is believed to be the best approximation [Dammertz et al., 2009]. In general, the hexagonal grid is the most efficient 2D sampling pattern, which achieves the highest Nyquist frequency for a given sampling density.

B.13 Farthest Point Optimization

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

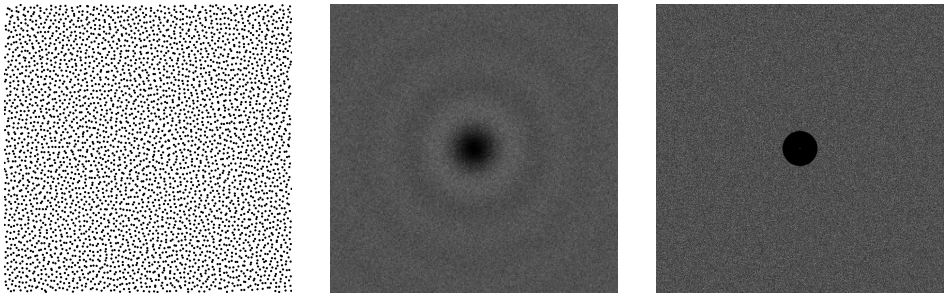
$Q^6 = 0.48$	$d_{\min} = 0.93$	$\nu_{\text{eff}} = 0.90$
$R_c = 0.86$	$d_{\text{avg}} = 0.93$	$\Omega = 4.64$

Notes

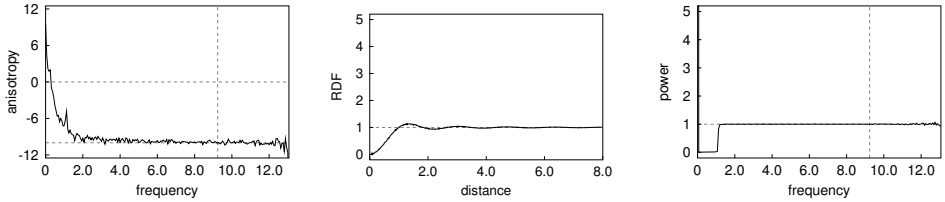
See Chapter 4. Note the strong peak in the autocorrelation and RDF and $r = d_{\min}$. This peak causes the slow decay of the power spectrum.

B.14 Step Blue Noise

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

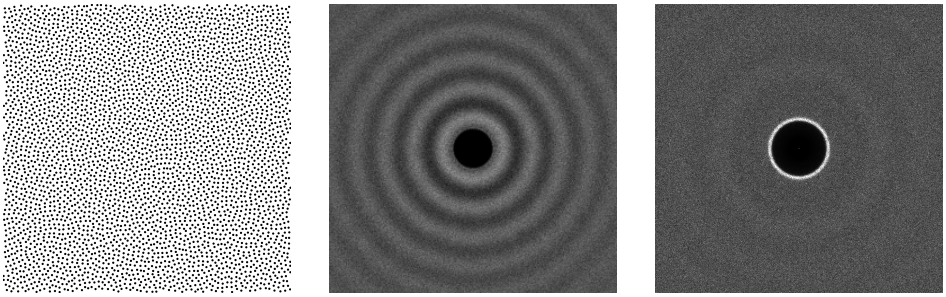
$Q^6 = 0.37$	$d_{\min} = 0.09$	$\nu_{\text{eff}} = 0.59$
$R_c = 0.91$	$d_{\text{avg}} = 0.64$	$\Omega = 0.01$

Notes

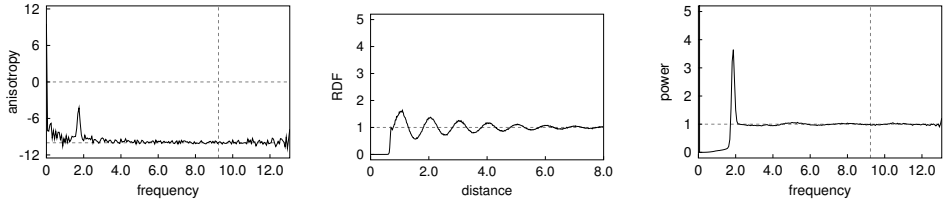
See Chapter 5.

B.15 Single-Peak Blue Noise

Point Distribution, Autocorrelation, Power Spectrum



Anisotropy, RDF, Radial Power Spectrum



Statistics

$Q^6 = 0.41$	$d_{\min} = 0.51$	$\nu_{\text{eff}} = 0.86$
$R_c = 0.77$	$d_{\text{avg}} = 0.80$	$\Omega = 1.44$

Notes

See Chapter 5.

Bibliography

T. Akenine-Möller, E. Haines, and N. Hoffman. *Real-Time Rendering*. A. K. Peters, Natick, MA, USA, third edition, 2008. (Cited on page 31.)

T. Auzinger, M. Guthe, and S. Jeschke. Analytic anti-aliasing of linear functions on polytypes. *Computer Graphics Forum (Proc. Eurographics 2012)*, 31(2):335–344, May 2012. (Cited on page 14.)

M. Balzer and D. Heck. Capacity-constrained Voronoi diagrams in finite spaces. In K. Sugihara and D.-S. Kim, editors, *Proceedings of the 5th Annual International Symposium on Voronoi Diagrams in Science and Engineering (ISVD 2008)*, pages 44–56, Kiev, Ukraine, Sept. 2008. (Cited on page 24.)

M. Balzer, T. Schlömer, and O. Deussen. Capacity-constrained point distributions: A variant of Lloyd's method. *ACM Trans. Graph.*, 28(3):86:1–86:8, 2009. (Cited on pages 24, 49, 60, 63, 66, 87, 90, and 113.)

K. Beets and D. Barron. Super-sampling anti-aliasing analyzed. http://www.x86-secret.com/articles/divers/v5-6000/datasheets/FSAA.pdf, Apr. 2000. (Cited on page 31.)

R. N. Bracewell. *The Fourier Transform and Its Applications*. McGraw-Hill, New York, third edition, 1999. (Cited on page 72.)

P. Brémaud, L. Massoulié, and A. Ridolfi. Power spectra of random spike fields and related processes. *Adv. Appl. Prob.*, 37(4):1116–1146, Dec. 2005. (Cited on page 39.)

G. J. Burton and I. R. Moorhead. Color and spatial structure in natural scenes. *Applied Optics*, 26(1):157–170, Jan. 1987. (Cited on page 48.)

E. C. Catmull. A hidden-surface algorithm with anti-aliasing. *Computer Graphics (Proc. SIGGRAPH 78)*, 12(3):6–11, Aug. 1978. (Cited on page 14.)

CGAL. CGAL, Computational Geometry Algorithms Library, CGAL. http://www.cgal.org. (Cited on page 59.)

R. Chen and C. Gotsman. Parallel blue-noise sampling by constrained farthest point optimization. *Computer Graphics Forum*, 32(5):1775–1785, Aug. 2012. (Cited on page 59.)

P. J. Clark and F. C. Evans. Distance to nearest neighbor as a measure of spatial relationships in populations. *Ecology*, 35(4):445–453, Oct. 1954. (Cited on page 51.)

M. F. Cohen, J. Shade, S. Hiller, and O. Deussen. Wang tiles for image and texture generation. *ACM Trans. Graph.*, 22(3):287–294, July 2003. (Cited on page 23.)

P. L. Combettes. The foundations of set theoretic estimation. *Proc. IEEE*, 81 (2):182–208, Feb. 1993. (Cited on page 99.)

R. L. Cook. Stochastic sampling in computer graphics. *Computer Graphics (Proc. of SIGGRAPH 86)*, 5(1):51–72, 1986. (Cited on pages 22, 38, 63, 107, 108, and 109.)

R. L. Cook and T. DeRose. Wavelet noise. *ACM Trans. Graph.*, 24(3):803–811, July 2005. (Cited on page 16.)

R. L. Cook, T. Porter, and L. Carpenter. Distributed ray tracing. *Computer Graphics (Proc. of SIGGRAPH 84)*, 18(3):137–145, July 1984. (Cited on pages 11 and 13.)

J. Crawford, S. Torquato, and F. H. Stillinger. Aspects of correlation function realizability. *J. Chem. Phys.*, 119(14):7065–7074, Oct. 2003. (Cited on page 78.)

F. C. Crow. The aliasing problem in computer-generated shaded images. *Commun. ACM*, 20(11):799–805, Nov. 1977. (Cited on page 24.)

F. C. Crow. Summed-area tables for texture mapping. *Computer Graphics (Proc. of SIGGRAPH 84)*, 18(3):207–212, July 1984. (Cited on page 16.)

D. J. Daley and D. Vere-Jones. *An Introduction to the Theory of Point Processes, Volume I.* Springer, second edition, 2002. (Cited on page 38.)

S. Dammertz, H. Dammertz, A. Keller, and H. Lensch. Textures on rank-1 lattices. *Computer Graphics Forum*, 28(7):1945–1954, Dec. 2009. (Cited on page 117.)

F. de Goes, K. Breeden, V. Ostromoukhov, and M. Desbrun. Blue noise through optimal transport. *ACM Trans. Graph.*, 31(6):171:1–171:11, Nov. 2012. (Cited on page 24.)

O. Deussen, S. Hiller, C. van Overveld, and T. Strothotte. Floating points: A method for computing stipple drawings. *Computer Graphics Forum*, 19(3): 41–50, Dec. 2000. (Cited on page 22.)

O. Devillers. On deletion in Delaunay triangulations. *Int. J. Comput. Geometry Appl.*, 12(3):193–205, 2002. (Cited on page 56.)

L. Devroye, C. Lemaire, and J.-M. Moreau. Expected time analysis for Delaunay point location. *Comput. Geom.*, 29(2):61–89, Oct. 2004. (Cited on page 56.)

M. A. Z. Dippé and E. H. Wold. Antialiasing through stochastic sampling. *Computer Graphics (Proc. of SIGGRAPH 85)*, 19(3):69–78, 1985. (Cited on pages 20, 22, 28, 38, 39, 40, 47, 85, 107, and 108.)

D. P. Dobkin and D. P. Mitchell. Random-edge discrepancy of supersampling patterns. In *Graphics Interface '93*, pages 62–69, May 1993. (Cited on page 115.)

Q. Du, V. Faber, and M. Gunzburger. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review*, 41(4):637–676, 1999. (Cited on page 114.)

E. Dubois. The sampling and reconstruction of time-varying imagery with application in video systems. *Proc. IEEE*, 73(4):502–522, Apr. 1985. (Cited on page 10.)

T. Duff. Polygon scan conversion by exact convolution. In J. André and R. D. Hersch, editors, *Proc. International Conference on Raster Imaging and Digital Typography*, pages 154–168. Cambridge University Press, Oct. 1989. (Cited on page 14.)

D. Dunbar and G. Humphreys. A spatial data structure for fast Poisson-disk sample generation. *ACM Trans. Graph.*, 25:503–508, July 2006. (Cited on page 23.)

M. S. Ebeida, A. Patney, S. A. Mitchell, A. Davidson, P. M. Knupp, and J. D. Owens. Efficient maximal Poisson-disk sampling. *ACM Trans. Graph. (Proceedings of SIGGRAPH 2011)*, 30(4), 2011. (Cited on page 52.)

M. S. Ebeida, S. A. Mitchell, A. Patney, A. A. Davidson, and J. D. Owens. A simple algorithm for maximal Poisson-disk sampling in high dimensions. *Computer Graphics Forum*, 31(2):785–794, May 2012. (Cited on page 52.)

D. S. Ebert, F. K. Musgrave, D. Peachey, K. Perlin, and S. Worley. *Texturing and Modeling: A Procedural Approach*. Morgan Kaufmann, third edition, 2002. (Cited on page 16.)

Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi. The farthest point strategy for progressive image sampling. *IEEE Trans. Image Process.*, 6(9): 1305–1315, Sept. 1997. (Cited on pages 52, 53, 63, and 110.)

J. Erickson. Dense point sets have sparse Delaunay triangulations. *Discrete Comput. Geom.*, 33:83–115, Jan. 2005. (Cited on page 56.)

R. Fattal. Blue-noise point sampling using kernel density model. *ACM Trans. Graph.*, 30(4):481:1–481:12, July 2011. (Cited on pages 24, 49, 63, 87, 90, and 111.)

H. Feichtinger and K. Gröchenig. Theory and practice of irregular sampling. In J. Benedetto and M. Frazier, editors, *Wavelet: Mathematics and Applications*, chapter 8, pages 305–363. CRC Press, 1994. (Cited on page 99.)

J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice*. Addison Wesley, second edition, 1996. (Cited on page 12.)

S. Fortune. Voronoi diagrams and Delaunay triangulations. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 377–388. CRC Press, New York, 1995. (Cited on page 56.)

S. F. Frisken, R. N. Perry, A. P. Rockwood, and T. R. Jones. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Proceedings of SIGGRAPH 2000*, pages 249–254. ACM, 2000. (Cited on page 14.)

M. N. Gamito and S. C. Maddock. Accurate multidimensional Poisson-disk sampling. *ACM Trans. Graph.*, 29(1):8:1–8:19, Dec. 2009. (Cited on pages 23 and 52.)

B. G. Giraud and R. Peschanski. On positive functions with positive Fourier transforms. *Acta Physica Polonica B*, 37(2):331–346, Feb. 2006. (Cited on page 81.)

A. S. Glassner. *Principles of Digital Image Synthesis*, volume 1. Morgan Kaufmann, first edition, Mar. 1995. (Cited on pages 47 and 85.)

R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Pearson Prentice Hall, Upper Saddle River, NJ, USA, third edition, 2008. (Cited on page 10.)

M. E. Goss and K. Wu. Study of supersampling methods for computer graphics hardware antialiasing. Technical Report HPL-1999-121R1, Hewlett-Packard Laboratories, 1999. (Cited on page 31.)

C. Green. Improved alpha-tested magnification for vector textures and special effects. In *ACM SIGGRAPH 2007 Courses*, pages 9–18. ACM, 2007. (Cited on page 15.)

N. Greene and P. S. Heckbert. Creating raster Omnimax images from multiple perspective views using the elliptical weighted average filter. *IEEE Comput. Graph. Appl.*, 6(6):21–27, June 1986. (Cited on page 16.)

K. Gröchenig. Reconstruction algorithms in irregular sampling. *Mathematics of Computation*, 59(199):181–194, July 1992. (Cited on pages 21 and 99.)

L. Grünschloß and A. Keller. (t, m, s)-nets and maximized minimum distance, part II. *Monte Carlo and Quasi-Monte Carlo Methods 2008*, pages 395–409, 2009. (Cited on page 115.)

L. Grünschloß, J. Hanika, R. Schwede, and A. Keller. (t, m, s)-nets and maximized minimum distance. *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 397–412, 2008. (Cited on page 51.)

S. Gupta and R. F. Sproull. Filtering edges for gray-scale displays. *Computer Graphics (Proc. of SIGGRAPH 81)*, 15(3):1–5, Aug. 1981. (Cited on page 14.)

T. Hachisuka, W. Jarosz, R. P. Weistroffer, K. Dale, G. Humphreys, M. Zwicker, and H. W. Jensen. Multidimensional adaptive sampling and reconstruction for ray tracing. *ACM Trans. Graph.*, 27(3):33:1–33:10, Aug. 2008. (Cited on page 11.)

D. Heck, T. Schlömer, and O. Deussen. Blue noise sampling with controlled aliasing. *ACM Trans. Graph.*, 32(3), 2013. (Cited on pages 28 and 68.)

P. Heckbert. Survey of texture mapping. *IEEE Comput. Graph. Appl.*, Nov. 1986a. (Cited on page 15.)

P. S. Heckbert. Survey of texture mapping. *IEEE Comput. Graph. Appl.*, 6 (11):56–67, Nov. 1986b. (Cited on page 15.)

S. Hiller, H. Hellwig, and O. Deussen. Beyond stippling - methods for distributing objects on the plane. *Computer Graphics Forum*, 22(3):515–522, 2003. (Cited on page 22.)

J. Illian, A. Penttinen, H. Stoyan, and D. Stoyan. *Statistical Analysis and Modelling of Spatial Point Patterns*. John Wiley & Sons, 2008. (Cited on page 71.)

J. Jimenez, D. Gutierrez, J. Yang, A. Reshetov, P. Demoreuille, T. Berghoff, C. Perthuis, H. Yu, M. McGuire, T. Lottes, H. Malan, E. Persson, D. Andreev, and T. Sousa. Filtering approaches for real-time anti-aliasing. In *ACM SIGGRAPH Courses*, 2011. (Cited on page 15.)

T. R. Jones. Efficient generation of Poisson disk sampling patterns. *Graphics, FPU and Game Tools*, 11(2):27–36, 2006. (Cited on page 23.)

J. Kajiya and M. Ullner. Filtering high quality text for display on raster scan devices. *Computer Graphics*, 15(3):7–15, 1981. (Cited on page 12.)

N. K. Kalantari and P. Sen. Efficient computation of blue noise point sets through importance sampling. *Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering (EGSR) 2011)*, 30(4):1215–1221, 2011. (Cited on page 23.)

Y. Kanamori, Z. Szego, and T. Nishita. Deterministic blue noise sampling by solving largest empty circle problems. *Journal of IIEEJ*, 40(210), 2011. (Cited on page 110.)

A. R. Kansal, T. M. Truskett, and S. Torquato. Nonequilibrium hard-disk packings with controlled orientational order. *Journal of Chemical Physics*, 113(12):4844, 2000. (Cited on page 105.)

J. Kopf, D. Cohen-Or, O. Deussen, and D. Lischinski. Recursive Wang tiles for real-time blue noise. *ACM Trans. Graph.*, 25(3):509–518, 2006. (Cited on pages 23 and 99.)

A. Lagae and P. Dutré. A procedural object distribution function. *ACM Trans. Graph.*, 24(4):1442–1461, Oct. 2005. (Cited on pages 22 and 23.)

A. Lagae and P. Dutré. A comparison of methods for generating Poisson disk distributions. *Computer Graphics Forum*, 27(1):114–129, 2008. (Cited on pages 51, 64, and 94.)

A. Lagae, S. Lefebvre, G. Drettakis, and P. Dutré. Procedural noise using sparse Gabor convolution. *ACM Trans. Graph.*, 28(3):54:1–54:10, 2009. (Cited on page 16.)

A. Lagae, S. Lefebvre, R. Cook, T. DeRose, G. Drettakis, D. S. Ebert, J. P. Lewis, K. Perlin, and M. Zwicker. A survey of procedural noise functions. *Computer Graphics Forum*, 29(8):2579–2600, Dec. 2010. (Cited on pages 16 and 109.)

A. Lagae, S. Lefebvre, and P. Dutré. Improving Gabor noise. *IEEE Trans. Vis. Comput. Graphics*, 17(8):1096–1107, 2011. (Cited on page 16.)

S. Laine and T. Aila. A weighted error metric and optimization method for antialiasing patterns. *Computer Graphics Forum*, 25(1):83–94, 2006. (Cited on page 34.)

D. L. Lau, R. Ulichney, and G. R. Arce. Blue- and green-noise halftoning models. *IEEE Signal Processing Magazine*, 20(4):28–38, July 2003. (Cited on page 71.)

H. Li, D. Nehab, L.-Y. Wei, P. V. Sander, and C.-W. Fu. Fast capacity constrained Voronoi tessellation. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, I3D '10, pages 13:1–13:1, New York, NY, USA, 2010. ACM. (Cited on page 24.)

Z. Lin, H.-T. Chen, H.-Y. Shum, and J. Wang. Pre-filtering 2D polygons without clipping. *Journal of Graphics Tools*, 10(1):17–26, 2005. (Cited on page 14.)

S. P. Lloyd. Least square quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. (Cited on pages 23, 60, and 114.)

S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, third edition, 2009. (Cited on page 38.)

M. McCool and E. Fiume. Hierarchical Poisson disk sampling distributions. In *Proc. Graphics interface '92*, pages 94–105. Morgan Kaufmann, 1992. (Cited on pages 23, 99, and 114.)

R. McNamara, J. McCormack, and N. P. Jouppi. Prefiltered antialiased lines using half-plane distance functions. Technical Report WRL 98/2, Compaq Western Research Laboratory, 1998. (Cited on page 14.)

D. P. Mitchell. Spectrally optimal sampling for distribution ray tracing. *Computer Graphics (Proc. of SIGGRAPH 91)*, 25(4):157–164, July 1991. (Cited on pages 20, 28, 35, and 110.)

D. P. Mitchell and A. N. Netravali. Reconstruction filters in computer graphics. *Computer Graphics (Proc. of SIGGRAPH 88)*, 22(4):221–228, June 1988. (Cited on pages 12 and 17.)

S. A. Mitchell, A. Rand, M. S. Ebeida, and C. Bajaj. Variable radii Poisson-disk sampling. In *Proceedings of the 24th Canadian Conference on Computational Geometry*, pages 185–190, 2012. (Cited on page 51.)

H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM, 1992. (Cited on page 99.)

NVIDIA. HRAA: High-resolution antialiasing through multisampling. Technical Brief, 2001. www.nvidia.com/attach/151. (Cited on pages 31 and 34.)

A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall, Upper Saddle River, NJ, USA, third edition, Aug. 2009. (Cited on page 7.)

A. V. Oppenheim and G. Verghese. 6.011 Introduction to communication, control, and signal processing. Massachusetts Institute of Technology: MIT OpenCourseWare, Spring 2010. URL http://ocw.mit.edu. (Cited on page 101.)

V. Ostromoukhov, C. Donohue, and P.-M. Jodoin. Fast hierarchical importance sampling with blue noise properties. *ACM Trans. Graph.*, 23(3): 488–495, 2004. (Cited on pages 23 and 99.)

C. Öztireli and M. Gross. Analysis and synthesis of point distributions based on pair correlation. *ACM Trans. Graph.*, 31(6), 2012. (Cited on pages 74 and 94.)

M. Pharr and G. Humphreys. *Physically Based Rendering: From Theory To Implementation*. Morgan Kaufmann, second edition, 2010. (Cited on page 87.)

M. D. Rintoul and S. Torquato. Reconstruction of the structure of dispersions. *Journal of Colloid and Interface Science*, 186:467–476, 1997. (Cited on pages 74, 78, and 94.)

T. Schlömer. *Non-Periodic Corner Tilings in Computer Graphics*. PhD thesis, University of Konstanz, 2012. (Cited on page 23.)

T. Schlömer, D. Heck, and O. Deussen. Farthest-point optimized point sets with maximized minimum distance. In *High Performance Graphics 2011*, pages 135–154. Eurographics Association, 2011. (Cited on pages 49 and 65.)

C. Schmaltz, P. Gwosdek, A. Bruhn, and J. Weickert. Electrostatic halftoning. *Computer Graphics Forum*, 29(8):2313–2327, Dec. 2010. (Cited on pages 24, 49, 60, 63, 87, 90, and 112.)

A. Secord. Weighted Voronoi stippling. In *Proceedings of the second international symposium on Non-photorealistic animation and rendering*, pages 37–43, 2002. (Cited on page 22.)

C. E. Shannon. Communications in the presence of noise. *Proc. IRE*, 37: 10–21, Jan. 1949a. (Cited on page 7.)

C. E. Shannon. *A Mathematical Theory of Communication*. University of Illinois Press, 1949b. (Cited on page 8.)

A. R. Smith. A pixel is not a little square, a pixel is not a little square, a pixel is not a little square, (and a voxel is not a little cube). Microsoft Technical Memo 6, July 1995. (Cited on page 14.)

B. Spencer and M. W. Jones. Into the blue: Better caustics through photon relaxation. *Comput. Graph. Forum*, 28(2):319–328, 2009. (Cited on page 22.)

R. Stasiński and J. Konrad. Improved POCS-based image reconstruction from irregularly-spaced samples. In *Proc. XI European Signal Proc. Conf.*, 2002. (Cited on page 99.)

K. Sung, A. Pearce, and C. Wang. Spatial-temporal antialiasing. *IEEE Trans. Vis. Comput. Graphics*, 8(2):144–153, 2002. (Cited on page 13.)

P. Thévenaz, T. Blu, and M. Unser. Image interpolation and resampling. In I. Bankman, editor, *Handbook of Medical Imaging, Processing and Analysis*, chapter 25, pages 393–420. Academic Press, San Diego CA, USA, 2000. (Cited on pages 12, 16, and 19.)

S. Torquato and F. H. Stillinger. Controlling the short-range order and packing densities of many-particle systems. *J. Phys. Chem. B*, 106(33):8354–8359, July 2002. (Cited on page 81.)

O. U. Uche, F. H. Stillinger, and S. Torquato. On the realizability of pair correlation functions. *Physica A*, 360(21):21–36, 2006. (Cited on pages 78 and 81.)

R. A. Ulichney. Dithering with blue noise. *Proc. IEEE*, 76(1):56–79, Jan. 1988. (Cited on pages 22, 23, and 69.)

R. A. Ulichney. *Digital Halftoning*. MIT Press, 1993. (Cited on page 94.)

M. Unser. Sampling—50 years after Shannon. *Proc. IEEE*, 88(4):569–587, Apr. 2000. (Cited on page 9.)

E. Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, Dec. 1997. (Cited on page 40.)

L.-Y. Wei. Parallel Poisson disk sampling. *ACM Trans. Graph.*, 27:20:1–20:9, Aug. 2008. (Cited on page 23.)

L.-Y. Wei and R. Wang. Differential domain analysis for non-uniform sampling. *ACM Trans. Graph.*, 30(4):50:1–50:10, July 2011. (Cited on pages 5, 68, 69, and 76.)

K. White, D. Cline, and P. Egbert. Poisson disk point sets by hierarchical dart throwing. In *Symposium on Interactive Ray Tracing*, pages 129–132, 2007. (Cited on page 23.)

D. R. Williams. Aliasing in human foveal vision. *Vision Research*, 25(2): 195–205, 1985. (Cited on page 21.)

L. Williams. Pyramidal parametrics. *Computer Graphics (Proc. of SIGGRAPH 83)*, 17(3):1–11, July 1983. (Cited on page 16.)

G. Wolberg. *Digital Image Warping*. IEEE Computer Society Press, 1990. (Cited on page 12.)

J. I. Yellot, Jr. Spectral consequences of photoreceptor sampling in the rhesus retina. *Science*, 221:382–385, 1983. (Cited on pages 21, 22, and 35.)

Y. Zhou, H. Huang, L.-Y. Wei, and R. Wang. Point sampling with general noise spectrum. *ACM Trans. Graph.*, 31(4), 2012. (Cited on pages 74, 81, 94, and 98.)