Signals and Sampling

CMPT 461/761
Image Synthesis
Torsten Möller
Reading

- Chapter 7 of “Physically Based Rendering” by Pharr & Humphreys
- Chapter 14.10 of “CG: Principles & Practice” by Foley, van Dam et al.
- Chapter 4, 5, 8, 9, 10 in “Principles of Digital Image Synthesis,” by A. Glassner
- Chapter 4, 5, 6 of “Digital Image Warping” by Wolberg
- Chapter 2, 4 of “Discrete-Time Signal Processing” by Oppenheim, Shafer
Motivation

• We live in a continuous world
• Computer can only offer finite, discrete rep.
• To discretize a continuous phenomenon
  – Take a finite number of samples – *sampling*
  – Use these samples to *reconstruct* an approximation of the continuous phenomenon
• To get the best approximation, need to be intelligent with sampling and reconstruction
If not careful …

• Artifacts can be caused by both sampling \textit{(pre-)} and reconstruction \textit{(post-aliasing)}:
  – Jaggies
  – Moire
  – Flickering small objects
  – Sparkling highlights
  – Temporal strobing

• Preventing these artifacts - Antialiasing
Signal processing and sampling

- Signal transform in a black-box

- Sampling or discretization:

  - Multiplication with “shah” function
Reconstruction (examples)

- nearest neighbor

- linear filter:
  - Convolution with box filter
  - Convolution with tent filter
Main issues/questions

• Can one ever perfectly reconstruct a continuous signal? – related to how many samples to take – the ideal case

• In practice, need for antialiasing techniques
  – Take more samples – *supersampling then resampling*
  – Modify signal (*prefiltering*) so that no need to take so many samples
  – Vary sampling patterns – *nonuniform sampling*
Motivation- Graphics

Original (continuous) signal → "Graphics" → "manipulated" (continuous) signal

Reconstruction filter

sampled signal

sampling
Basic concept 1: Convolution

• How can we characterize our “black box”?  
• We assume to have a “nice” box/algorithm: 
  – linear
  – time-invariant
• then it can be characterized through the response to an “impulse”:
Convolution (2)

• Impulse: \( \delta(x) = 0, \text{ if } x \neq 0 \)
  \[ \int_{-\infty}^{\infty} \delta(x) dx = 1 \]

• discrete impulse: \( \delta[k] = 0, \text{ if } k \neq 0 \)
  \( \delta[0] = 1 \)

• Finite Impulse Response (FIR) vs.
• Infinite Impulse Response (IIR)
Convolution (3)

- Continuous convolution …
- Discrete: an signal $x[k]$ can be written as:
  \[ x[k] = \ldots + x[-1]\delta[k + 1] + x[0]\delta[k] + x[1]\delta[k - 1] + \ldots \]
- Let the impulse response be $h[k]$:

\[
\begin{array}{c}
\delta[k] \\
\text{“System” or Algorithm} \\
h[k]
\end{array}
\]
Convolution (4)

• for a linear time-invariant system $h$, $h[k-n]$ would be the impulse response to a delayed impulse $\delta[k-n]$

• hence, if $y[k]$ is the response of our system to the input $x[k]$ (and we assume a linear system):

$$y[k] = \sum_{n=-N}^{N} x[n]h[k-n]$$

IIR - $N=\text{inf}$.
FIR - $N<\text{inf}$. 

```
x[k]   →  "System" or Algorithm  →  y[k]
```
Basic concept 2: Fourier Transforms

• Let’s look at a special input sequence:
  \[ x[k] = e^{i\omega k} \]

• Then applying to a linear, time-invariant h:
  \[
  y[k] = \sum_{n=-N}^{N} e^{i\omega (k-n)} h[n] \\
  = e^{i\omega k} \sum_{n=-N}^{N} e^{-i\omega n} h[n] \\
  = H(\omega) e^{i\omega k}
  \]
Fourier Transforms (2)

- View $h$ as a *linear operator (circulant matrix)*
- Then $e^{i\omega k}$ is an eigen-function of $h$ and $H(\omega)$ its eigenvalue
- $H(\omega)$ is the Fourier-Transform of the $h[n]$ and hence characterizes the underlying system in terms of frequencies
- $H(\omega)$ is periodic with period $2\pi$
- $H(\omega)$ is decomposed into
  - phase (angle) response $\angle H(\omega)$
  - magnitude response $|H(\omega)|$
Fourier transform pairs

\[ F(\omega) = \int_{-\infty}^{+\infty} f(x)e^{-i2\pi \omega x} \, dx \]

\[ f(x) = \int_{-\infty}^{+\infty} F(\omega)e^{i2\pi \omega x} \, d\omega \]
Properties

• Linear \[ af(x) + bg(x) \Leftrightarrow aF(\omega) + bG(\omega) \]
• Scaling \[ f(ax) \Leftrightarrow 1/a F(\omega/a) \]
• Convolution \[ f(x) \otimes g(x) \Leftrightarrow F(\omega) \otimes G(\omega) \]
• Multiplication \[ f(x) \times g(x) \Leftrightarrow F(\omega) \otimes G(\omega) \]
• Differentiation \[ \frac{d^n}{dx^n} f(x) \Leftrightarrow (i\omega)^n F(\omega) \]
• Delay/Shift \[ f(x - \tau) \Leftrightarrow e^{-i\tau} F(\omega) \]
Properties (2)

- Parseval’s Theorem
  \[ \int_{-\infty}^{\infty} f^2(x) dx \Leftrightarrow \int_{-\infty}^{\infty} F^2(\omega) d\omega \]

- Preserves “Energy” - overall signal content
- Characteristic of orthogonal transforms
Proof of convolution theorem

\[
\int_{-\infty}^{+\infty} \left[ \int_{-\infty}^{+\infty} f(y) g(x - y) \, dy \right] e^{-i2\pi\omega x} \, dx \\
= \int_{-\infty}^{+\infty} f(y) \left[ \int_{-\infty}^{+\infty} g(x - y) e^{-i2\pi\omega x} \, dx \right] \, dy \\
= \int_{-\infty}^{+\infty} f(y) \left[ \int_{-\infty}^{+\infty} g(z) e^{-i2\pi\omega (y+z)} \, dz \right] \, dy \quad z = x - y \\
= \int_{-\infty}^{+\infty} f(y) e^{-i2\pi\omega y} G(\omega) \, dy = F(\omega)G(\omega)
\]
Transforms Pairs

Fourier Transform

Average Filter

Box/Sinc Filter

© Machiraju/Möller
Transforms Pairs (2)
Transform Pairs - Shah

- Sampling = Multiplication with a Shah function:

  \[ \text{multiplication in spatial domain} = \text{convolution in the frequency domain} \]

- frequency replica of primary spectrum (also called aliased spectra)
General Process of Sampling and Reconstruction

Original function → Acquisition → Sampled function → Reconstruction → Re-sampled function → Resampling → Reconstructed function

- e.g., supersampling
- e.g., resample at screen resolution

© Machiraju/Möller
How so? - Convolution Theorem

Spatial Domain:

Convolution:
\[ \int_{-\infty}^{\infty} f(t) \times g(x - t) \, dt \]

Frequency Domain:

Multiplication:
\[ F(\omega) \times G(\omega) \]
Sampling Theorem

• A signal can be reconstructed from its samples without loss of information if the original signal has no frequencies above 1/2 of the sampling frequency

• For a given bandlimited function, the rate at which it must be sampled (to have perfect reconstruction) is called the **Nyquist frequency**

• Due to Claude Shannon (1949)
Example

2D

Given

Needed

1D

Given

Needed

© Machiraju/Möller
From last week ... 

- linear time-invariant system
- impulse response
- convolution
- putting in one frequency - eigenvector / eigenvalue
- analysing spatial domain things by what happens in the Frequency domain
- able to reason about sampling errors --> aliasing and how to avoid it!
Once Again ...

Pre-aliasing

Pre-filter

Post-aliasing

Reconstruction filter

Sampling
In the frequency domain

Original function

Reconstructed function

Sampled function

Reconstructed function

Re-sampled function

Acquisition

Reconstruction

Resampling

© Machiraju/Möller
Pipeline - Example

Spatial domain

sampling

smoothing

Frequency domain

© Machiraju/Möller
Pipeline - Example (2)

Spatial domain

Frequency domain

smoothing

Re-sampling

© Machiraju/Möller
Pipeline - Example (3)

Spatial domain

Frequency domain

© Machiraju/Möller
Cause of Aliasing

• Non-bandlimited signal – \textit{prealiasing}

• Low sampling rate (\leq \text{Nyquist}) – \textit{prealiasing}

• Non perfect reconstruction – \textit{post-aliasing}
Aliasing example
Aliasing: Sampling a Zone Plate

\[ \sin(x^2 + y^2) \]
Aliasing: Sampling a Zone Plate

\[
\sin(x^2 + y^2)
\]

Sampled at 128 x 128 and reconstructed to 512 x 512 using windowed sinc

Left rings: part of the signal
Right rings: aliasing due to undersampling

© Machiraju/Möller
Antialiasing 1: Pre-Filtering

Original function  →  Band-limited function

Sampled Function  ←  Acquisision

Reconstruction

© Machiraju/Möller

36
Antialiasing 2: Uniform Supersampling

- Increasing the sampling rate moves each copy of the spectra further apart, potentially reducing the overlap and thus aliasing.
- Low-pass filter and then the resulting signal is re-sampled at image resolution.

\[ \text{Pixel} = \sum_{k} w_k \times \text{Sample}_k \]
Point vs. supersampling

Point vs. 4x4 Supersampled

Checkerboard sequence by Tom Duff
Summary: Antialiasing

- Antialiasing = Preventing aliasing

1. Analytically pre-filter the signal
   - Solvable for points, lines and polygons
   - Not solvable in general (e.g. procedurally defined images)

2. Uniform supersampling and resample

3. Nonuniform or stochastic sampling – later!
Reconstruction = Interpolation

**Spatial Domain:**
- convolution is exact

\[ f_r(x) - f(x) = 0 \]

**Frequency Domain:**
- cut off freq. replica

\[ \text{Sinc}(x) = \frac{\sin(\pi x)}{\pi x} \]
Example: Derivatives

**Spatial Domain:**
- convolution is exact

\[ f_r^d(x) - f'(x) = 0 \]

**Frequency Domain:**
- cut off freq. replica

\[ \text{Cosc}(x) = \frac{\cos(\pi x)}{x} - \frac{\sin(\pi x)}{\pi x^2} \]

© Machiraju/Möller
Reconstruction Kernels

- Nearest Neighbor (Box)
- Linear
- Sinc
- Gaussian
- Many others

Spatial d.  Frequency d.
Interpolation example

Nearest neighbor  Linear Interpolation

© Machiraju/Möller
Ideal Reconstruction

• Box filter in frequency domain =
• Sinc Filter in spatial domain
• Sinc has *infinite* extent – not practical
Ideal Reconstruction

• Use the sinc function – to bandlimit the sampled signal and remove all copies of the spectra introduced by sampling

• But:
  – The sinc has infinite extent and we must use simpler filters with finite extents.
  – The windowed versions of sinc may introduce ringing artifacts which are perceptually objectionable.
Reconstructing with Sinc: Ringing
Ideal filters

- Also have ringing in pass/stop bands
- Realizable filters do not have sharp transitions
Summary: possible errors

- **Post-aliasing**
  - reconstruction filter passes frequencies beyond the Nyquist frequency (of duplicated frequency spectrum)
  => frequency components of the original signal appear in the reconstructed signal at different frequencies

- **Smoothing due to prefiltering**
  - frequencies below the Nyquist frequency are attenuated

- **Ringing (overshoot)**
  - occurs when trying to sample/reconstruct discontinuity

- **Anisotropy**
  - caused by not spherically symmetric filters
Higher Dimensions?

• Design typically in 1D
• Extensions to higher dimensions (typically):
  – Separable filters
  – Radially symmetric filters
  – Limited results
• Research topic
Aliasing vs. Noise
Distribution of Extrafoveal Cones

• Yellot theory (1983)
  – Structured aliases replaced by noise
  – Visual system less sensitive to high freq noise

Monkey eye cone distribution

Fourier Transform

© Machiraju/Möller
Non-Uniform Sampling - Intuition

• Uniform sampling
  – The spectrum of uniformly spaced samples is also a set of uniformly spaced spikes
  – Multiplying the signal by the sampling pattern corresponds to placing a copy of the spectrum at each spike (in freq. space)
  – Aliases are coherent, and very noticeable

• Non-uniform sampling
  – Samples at non-uniform locations have a different spectrum; a single spike plus noise
  – Sampling a signal in this way converts structured aliases into broadband noise
  – Noise is incoherent, and much less objectionable
Uniform vs. non-uniform point sampling

Uniformly sampled 40x40

Uniformly jittered 40x40

© Machiraju/Möller
Non-Uniform Sampling Patterns

- **Poisson**
  - Pick \( n \) random points in sample space

- **Uniform Jitter**
  - Subdivide sample space into \( n \) regions

- **Poisson Disk**
  - Pick \( n \) random points, but not too close
Poisson Disk Sampling

Spatial Domain

Fourier Domain

© Machiraju/Möller
Uniform Jittered Sampling

Spatial Domain  
Fourier Domain

© Machiraju/Möller
Non-Uniform Sampling - Patterns

- Spectral characteristics of these distributions:
  - Poisson: completely uniform (white noise). High and low frequencies equally present.
  - Poisson disc: Pulse at origin (DC component of image), surrounded by empty ring (no low frequencies), surrounded by white noise.
  - Jitter: Approximates Poisson disc spectrum, but with a smaller empty disc.
Stratified Sampling

• Divide sample space into stratas
• Put at least one sample in each strata
• Also have samples far away from each other
  – samples too close to each other often
    provide no new information

• Example: uniform jittering
Jitter

- Place samples in the grid
- Perturb the samples up to 1/2 width or height
Texture Example

“ideal” – 256 samples/pixel

Jitter with 1 sample/pixel

1 sample/pixel uniform and unjittered

Jitter with 4 samples/pixel
Multiple Dimensions

- Too many samples
- 1D
- 2D

© Machiraju/Möller
Jitter Problems

• How to deal with higher dimensions?
  – Curse of dimensionality
  – D dimensions means $N^D$ “cells” (if we use a separable extension)

• Solutions:
  – We can look at each dimension independently and stratify, after which randomly associate samples from each dimension
  – Latin Hypercube (or N-Rook) sampling
Multiple Dimensions

- Make (separate) strata for each dimension
- Randomly associate strata among each other
- Ensure good sample "distribution"
  - Example: 2D screen position; 2D lense position; 1D time
Aside: alternative sampling lattices

- Dividing space up into equal cells doesn’t have to be on a Cartesian lattice.
- In fact - Cartesian is NOT the optimal way how to divide up space uniformly.

![Cartesian](image1.png) ![Hexagonal](image2.png) Hexagonal is optimal in 2D
Aside: optimal sampling lattices

- We have to deal with different geometry
- 2D - hexagon
- 3D - truncated octahedron
Latin Hypercubes (LHS) or N-Rooks in 2D

- Generate a jittered sample in each of the diagonal entries
- Random shuffle in each dimension
- Projection to each dimension corresponds to a uniform jittered sampling

© Machiraju/Möller
LHS or N-Rooks in \( k \)-D

Generate \( n \) samples \((s^i_1, s^i_2, \ldots, s^i_k)\) in \( k \) dimensions

- Divide each dimension into \( n \) cells
- Assign a random permutation of \( n \) to each dimension
- Sample coordinates are jittered in corresponding cells according to indices from the permutations

\[
\begin{array}{cccccccccc}
7 & 5 & 8, & 1 & 4 & 10 & 3 & 9 & 2 & 6 \\
3 & 5 & 1 & 6 & 9 & 4 & 8 & 2 & 7 & 10 \\
7 & 10 & 3 & 9 & 1 & 8 & 2 & 5 & 6 & 4 \\
\end{array}
\]

\( k = 3 \)

\( s^3_1 \) is from the 8-th cell from dimension 1

\( n = 10 \)

© Machiraju/Möller
Stratification - problems

- **Clumping** and holes due to randomness and independence between strata
- LHS can help but no quality assurance due to random permutations, e.g., diagonal

Other geometries, e.g. stratify circles or spheres?
How good are the samples?

• How can we evaluate how well our samples are distributed in a more global manner?
  – No “holes”
  – No clumping

• Well distributed patterns are low-discrepancy
  – more evenly distributed

• Want to construct low-discrepancy sequence

• Most of these are deterministic!
Discrepancy

• Intuition: for a well distributed set of samples in 
  \([0,1]^n\), the relative volume of any sub-region should 
  be close to the relative percentage of points therein

• For a particular set \(B\) of sub-volumes of \([0,1]^d\) and a 
  sequence \(P\) of \(N\) sample points in \([0,1]^d\)

\[
D_N(B,P) = \sup_{b \in B} \left| \frac{\# \{ x_i \in b \}}{N} - Vol(b) \right|
\]

• E.g., for the marked sub-volume, 
  we have \(|7/22 - 1/4| \leq D_{22}(B, P)\)
Discrepancy

• Examples of sub-volume sets $B$ of $[0,1]^d$:
  – All axis-aligned
  – All those sharing a corner at the origin (called \textit{star discrepancy} $D_N^*(P)$)

• \textbf{Asymptotically lowest} discrepancy that has been obtained in $d$ dimensions:

$$D_N^*(P) = O\left( \frac{(\log N)^d}{N^{\frac{1}{d}}} \right)$$
Discrepancy

• How to create low-discrepancy sequences?
  – *Deterministic sequences!* Not random anymore
  – Also called pseudo-random
  – Advantage: easy to compute

• 1D:

\[
x_i = \frac{i}{N} \quad \Rightarrow \quad D_N^*(x_1, \ldots, x_N) = \frac{1}{N}
\]

Optimal yet uniform:

\[
x_i = \frac{i - 0.5}{N} \quad \Rightarrow \quad D_N^*(x_1, \ldots, x_N) = \frac{1}{2N}
\]

In general, 

\[
D_N^*(x_1, \ldots, x_N) = \frac{1}{2N} + \max_{1 \leq i \leq N} \left| x_i - \frac{2i - 1}{2N} \right|
\]
Pseudo-Random Sequences

• Radical inverse
  – Building block for high dimensional sequences
  – “inverts” an integer given in base $b$

$$n = a_k ... a_2 a_1 = a_1 b^0 + a_2 b^1 + a_3 b^2 + ...$$

$$\Phi_b(n) = 0.a_1 a_2 ... a_k = a_1 b^{-1} + a_2 b^{-2} + a_3 b^{-3} + ...$$
Van Der Corput Sequence

- One of the simplest 1D sequence: $x_i = \Phi_2(i)$
- Uses radical inverse of base 2
- Asymptotically optimal discrepancy

$$D_N^*(P) = O\left(\frac{\log N}{N}\right)$$

<table>
<thead>
<tr>
<th>$i$</th>
<th>binary form of $i$</th>
<th>radical inverse</th>
<th>$x_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>0.01</td>
<td>0.25</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>0.11</td>
<td>0.75</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>0.001</td>
<td>0.125</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>0.101</td>
<td>0.625</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
<td>0.011</td>
<td>0.375</td>
</tr>
</tbody>
</table>
• Use a *prime number basis* for each dimension
• Achieves best possible discrepancy asymptotically

\[ x_i = (\Phi_2(i), \Phi_3(i), \Phi_5(i), ..., \Phi_{p_d}(i)) \]
\[ D_N^*(P) = O\left( \frac{(\log N)^d}{N} \right) \]

• Can be used if \( N \), the number of samples, is not known in advance — all prefixes of a Halton sequence are well distributed
Hammersley Sequences

- Similar to Halton
- But need to know $N$, the total number of samples, in advance
- Slightly lower discrepancy than Halton

$$x_i = \left( \frac{i}{N}, \Phi_{p_1}(i), \Phi_{p_2}(i), \ldots, \Phi_{p_{d-1}}(i) \right)$$

Prime numbers
Halton vs. Hammersley

First 100 samples in $[0, 1]^2$
Hammersley Sequences

In 2D, $x_i = (i/N, \Phi_{p_1}(i))$

As $p_1$ increases, the pattern becomes regular, resulting in aliasing problems
Hammersley Sequences

Similar behavior on the sphere.

Samples on the sphere are obtained by wrapping the square into a cylinder and then doing a radial projection
Folded Radical Inverse

- Modulate each digit in the radical inverse by an offset than modulo with the base
- Hammersley-Zaremba or Halton-Zaremba
- Improves discrepancy

\[ \Phi_b(n) = \sum_{i=1}^{\infty} \frac{1}{b^i} \]

\[ \Phi_b(n) = \sum_{i=1}^{\infty} ((a_i + i - 1) \mod b) \frac{1}{b^i} \]
Halton and Hammersley folded
(t,m,d) nets

• Most successful constructions of low-discrepancy sequences are (t,m,d)-nets and (t,d)-sequences.

• Basis b: a prime or prime power

• 0 =< t =< m

• A (t,m,d)-net in base b is a point set in $[0,1]^d$ consisting of $b^m$ points, such that every box

$$E = \prod_{i=1}^{d} \left[ a_i b^{-c_i}, (a_i + 1) b^{-c_i} \right]$$

where $\sum_{i=1}^{d} c_i = m - t$

of volume $b^{t-m}$ contains $b^t$ points

Reference: www.mathdirect.com/products/qrn/resources/Links/QRDemonstration_Ink_4.html

Optimal in absolute terms

© Machiraju/Möller
(t,d) Sequences

- (t,m,d)-nets ensures that samples are well distributed for particular integer subdivisions of the space.
- A (t,d)-sequence in base b is a sequence $x_i$ of points in $[0,1]^d$ such that for all integers $k \geq 0$ and $m > t$, the point set
  \[ \left\{ x_i \mid kb^m \leq i < (k+1)b^m \right\} \]
  is a (t,m,d)-net in base b.
- The number $t$ is the quality parameter.
  - Smaller $t$ yield more uniform nets and sequences because b-ary boxes of smaller volume still contain points.

Reference: www.mathdirect.com/products/qrn/resources/Links/QRDemonstration_lnk_4.html
(t,d) = (0,2) sequences

- Used in pbrt for the Low-discrepancy sampler
- First and succeeding block of 16 = 2^4 samples in the sequence give a (0,4,2) net
- First and succeeding block of 8 = 2^3 samples in the sequence give a (0,3,2) net
- etc.

All possible uniform divisions into 16 rectangles:

One sample in each of 16 rectangles

N-rook property
Practical Issues

• Create one sequence
• Create new ones from the first sequence by “scrambling” rows and columns
• This is only possible for (0,2) sequences, since they have such a nice property (the “n-rook” property)
Texture

Jitter with 1 sample/pixel

Hammersley Sequence with 1 sample/pixel
Best-Candidate Sampling

- Jittered stratification
  - Randomness (inefficient)
  - Clustering problems between adjacent strata
  - Undersampling ("holes")

- Low Discrepancy Sequences
  - No explicit preventing two samples from coming too close

- "Ideal": Poisson disk distribution
  - too computationally expensive

- Best Sampling - approximation to Poisson disk – a form of farthest point sampling
Poisson Disk

- Comes from structure of eye – rods and cones
- Dart Throwing
- No two points are closer than a threshold
- Very expensive
- Compromise – Best Candidate Sampling
  - Every new sample is to be farthest from previous samples amongst a set of randomly chosen candidates
  - Compute pattern which is reused by tiling the image plane (translating and scaling).
  - Toroidal topology
Best-Candidate Sampling

Jittered

Poisson Disk

Best Candidate
Best-Candidate Sampling

Jittered
Best-Candidate Sampling

Poisson Disk
Best-Candidate Sampling

Best Candidate
Dart throwing

\[
i \leftarrow 0
\]

\[
\text{while } i < N \\
\quad x_i \leftarrow \text{unit()} \\
\quad y_i \leftarrow \text{unit()} \\
\quad reject \leftarrow \text{false}
\]

\[
\text{for } k \leftarrow 0 \text{ to } i - 1 \\
\quad d \leftarrow (x_i - x_k)^2 + (y_i - y_k)^2
\]

\[
\text{if } d < (2r_p)^2 \text{ then} \\
\quad reject \leftarrow \text{true} \\
\quad \text{break}
\]

\[
\text{endif}
\]

\[
\text{endfor}
\]

\[
\text{if not reject then} \\
\quad i \leftarrow i + 1
\]

\[
\text{endif}
\]

\[
\text{endwhile}
\]

- Throw a dart.
- Check the distance to all other samples.
- This one is too close—forget it.
- Append this one to the pattern.
Texture

Jitter with 1 sample/pixel

Best Candidate with 1 sample/pixel

Jitter with 4 sample/pixel

Best Candidate with 4 sample/pixel

© Machiraju/Möller
Next

• Rendering Equation
• Probability Theory
• Monte Carlo Techniques