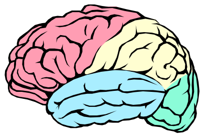
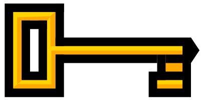
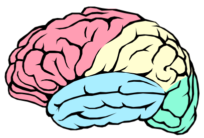
Signals and Sampling

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

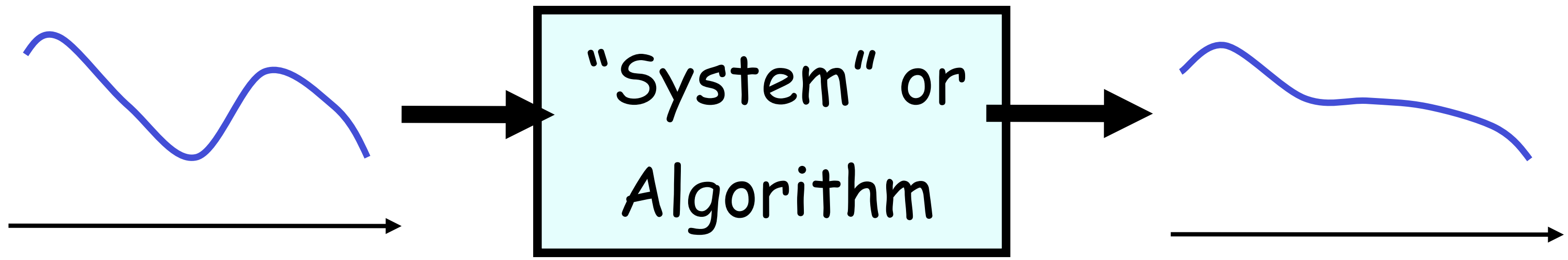
- We live in a continuous world
- Computer can only offer finite, discrete rep.
- To discretize a continuous phenomenon
  - Take a finite number of samples – *sampling*
  - Use these samples to *reconstruct* an approximation of the continuous phenomenon
- To get the best approximation, need to be intelligent with sampling and reconstruction

If not careful …

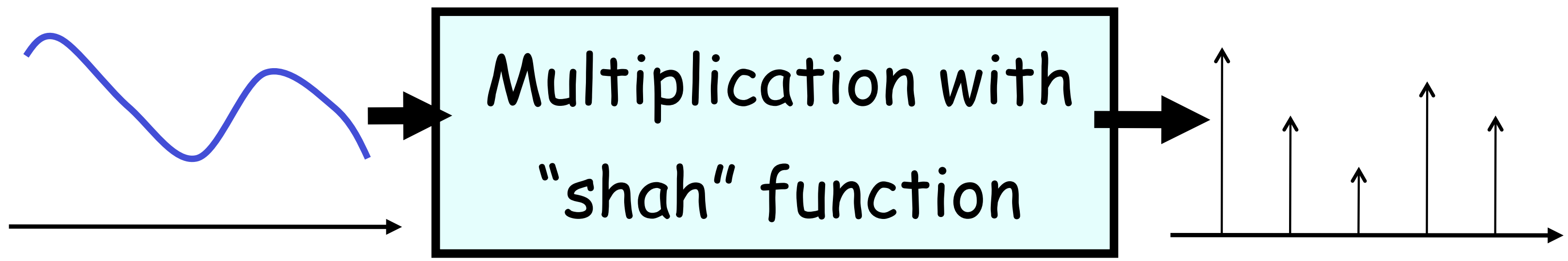
- Artifacts can be caused by both sampling (*pre*-) and reconstruction (*post-aliasing*):
  - Jaggies
  - Moire
  - Flickering small objects
  - Sparkling highlights
  - Temporal strobing
- Preventing these artifacts - Antialiasing

Signal processing and sampling

- Signal transform in a black-box

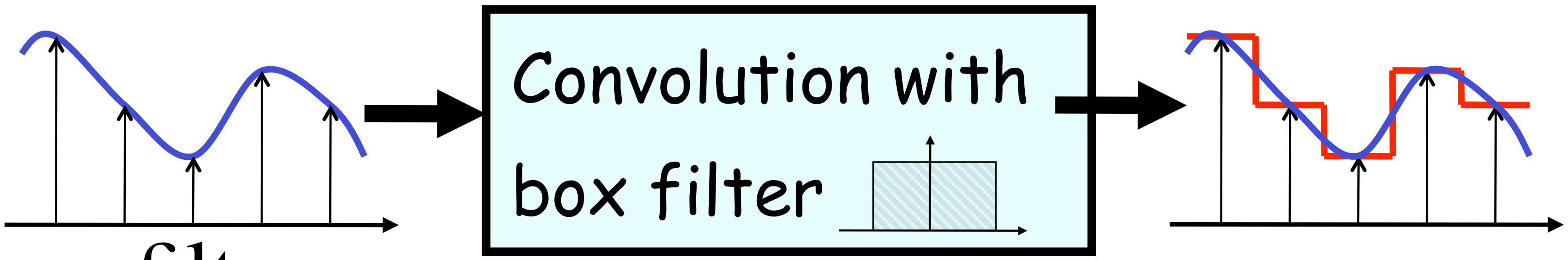


- Sampling or discretization:

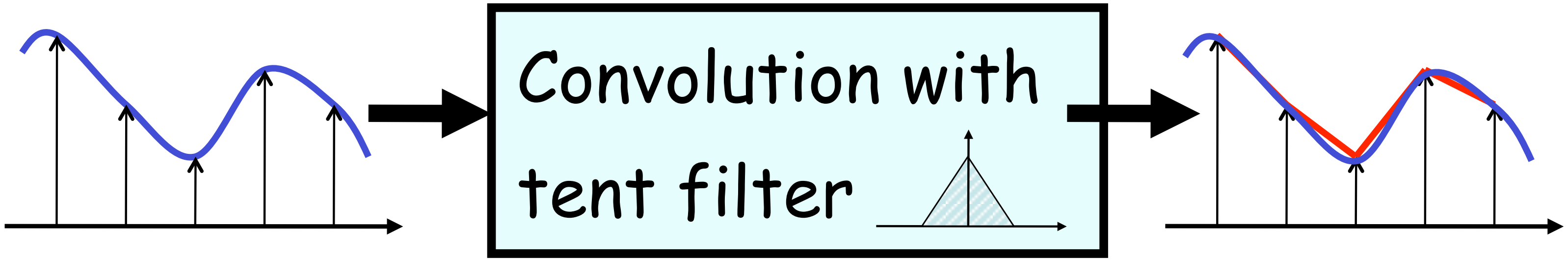


Reconstruction (examples)

- nearest neighbor



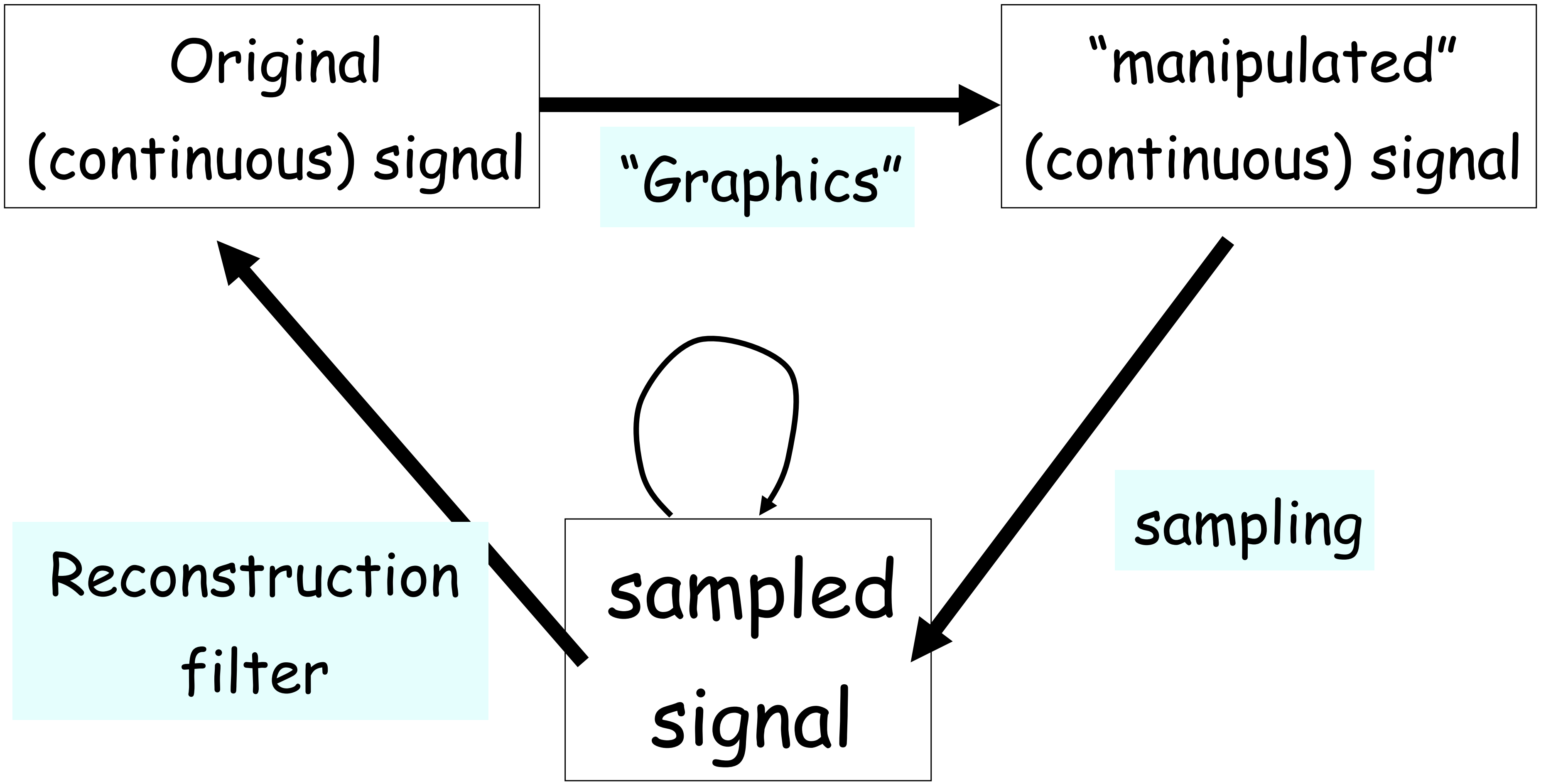
- linear filter:



Main issues/questions

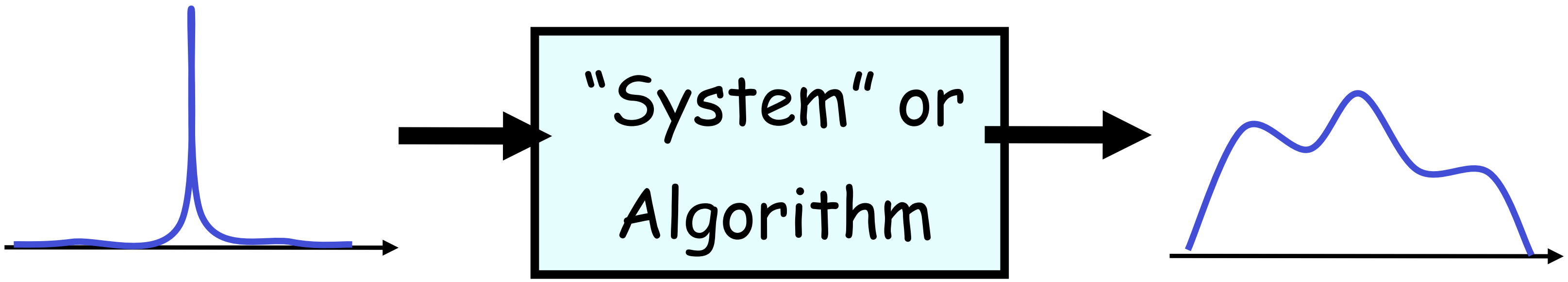
- Can one ever perfectly reconstruct a continuous signal? – related to how many samples to take – the ideal case
- In practice, need for antialiasing techniques
  - Take more samples – *supersampling then resampling*
  - Modify signal (*prefiltering*) so that no need to take so many samples
  - Vary sampling patterns – *nonuniform sampling*

Motivation- Graphics



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: $\delta(x) = 0, \text{if } x \neq 0$

$$\int_{-\infty}^{\infty} \delta(x)dx = 1$$

Not a math function

- discrete impulse: $\delta[k] = 0, \text{if } k \neq 0$

$$\delta[0] = 1$$

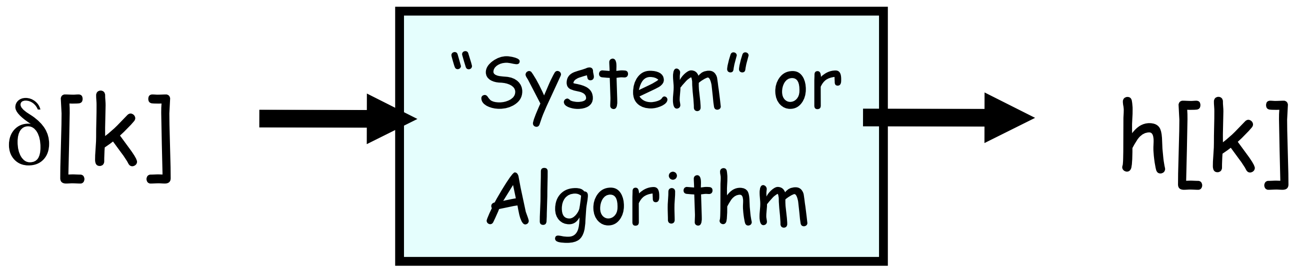
- Finite Impulse Response (FIR) vs.
- Infinite Impulse Response (IIR)

Convolution (3)

- Continuous convolution …
- Discrete: an signal x[k] can be written as:

$$x[k] = ... + x[-1]\delta[k+1] + x[0]\delta[k] + x[1]\delta[k-1] + ...$$

- Let the impulse response be h[k]:

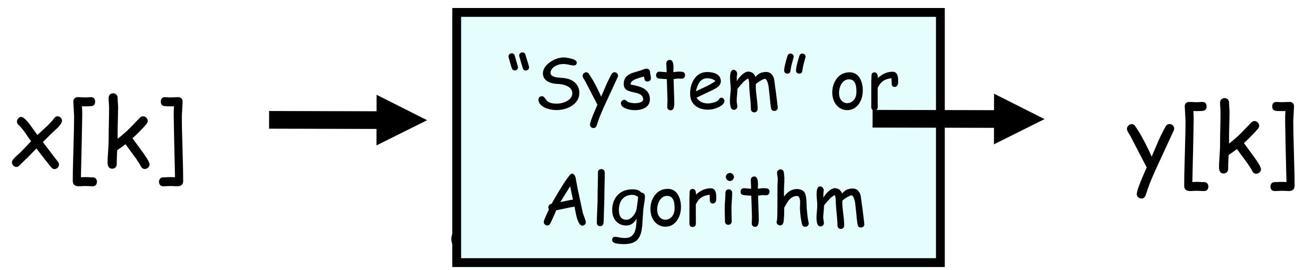


Convolution (4)

- for a linear time-invariant system h, h[k-n] would be the impulse response to a delayed impulse δ[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=inf.
FIR - N<inf.



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(ω) its eigenvalue
- H(ω) is the Fourier-Transform of the h[n] and hence characterizes the underlying system in terms of frequencies
- H(ω) is periodic with period 2π
- H(ω) is decomposed into
  - phase (angle) response $\triangleleft H(\omega)$
  - magnitude response $\left|H(\omega)\right|$

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 $af(x)+bg(x) \Leftrightarrow aF(\omega)+bG(\omega)$
- Expansion $f(ax) \Leftrightarrow 1/a\, F(\omega/a)$
- Convolution $f(x) \otimes g(x) \Leftrightarrow F(\omega) \times G(\omega)$
- 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

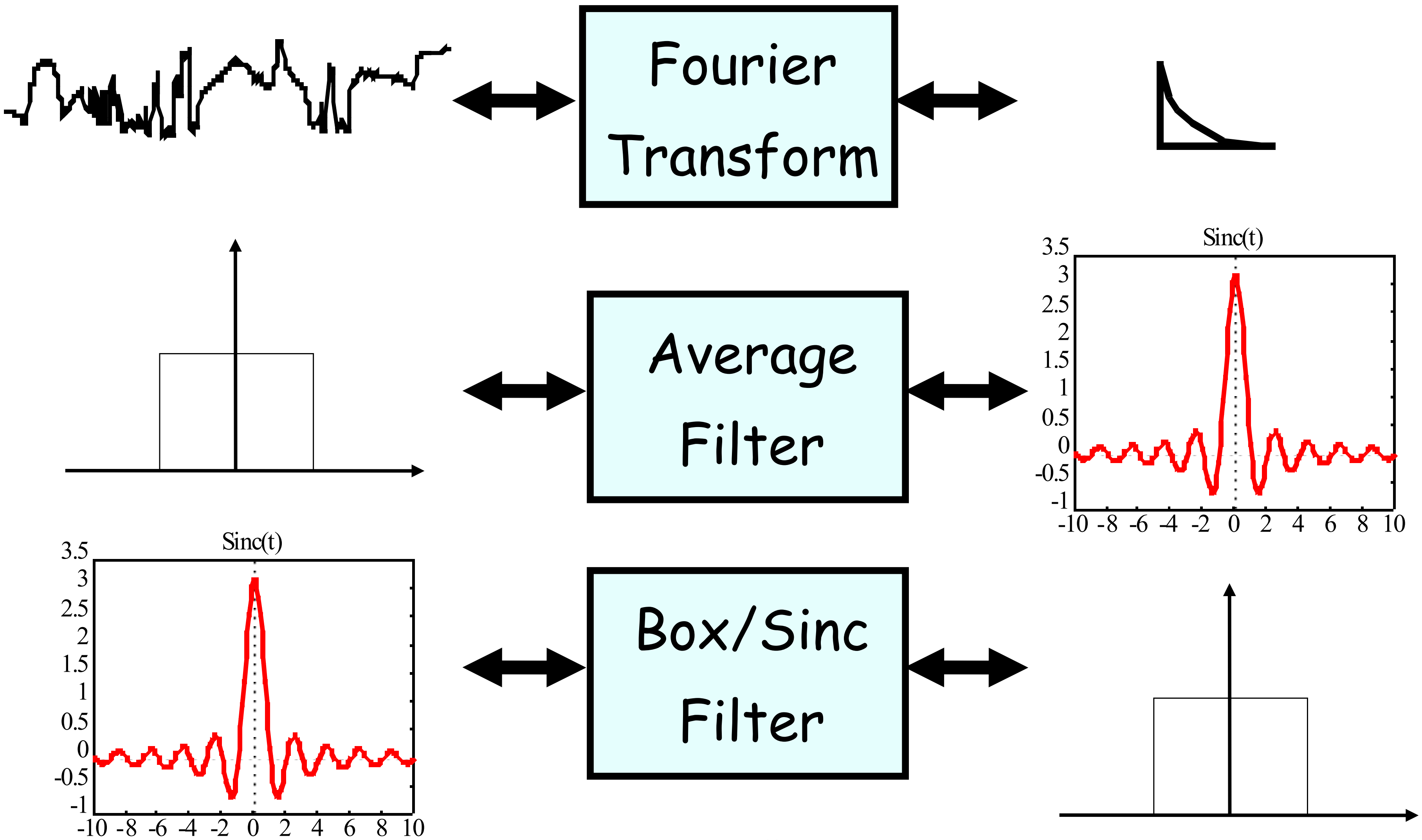
$$\int_{-\infty}^{+\infty}\left[\int_{-\infty}^{+\infty} f(y)g(x-y)dy\right]e^{-i2\pi\omega x}dx$$

$$=\int_{-\infty}^{+\infty} f(y)\left[\int_{-\infty}^{+\infty} g(x-y)e^{-i2\pi\omega x}dx\right]dy$$

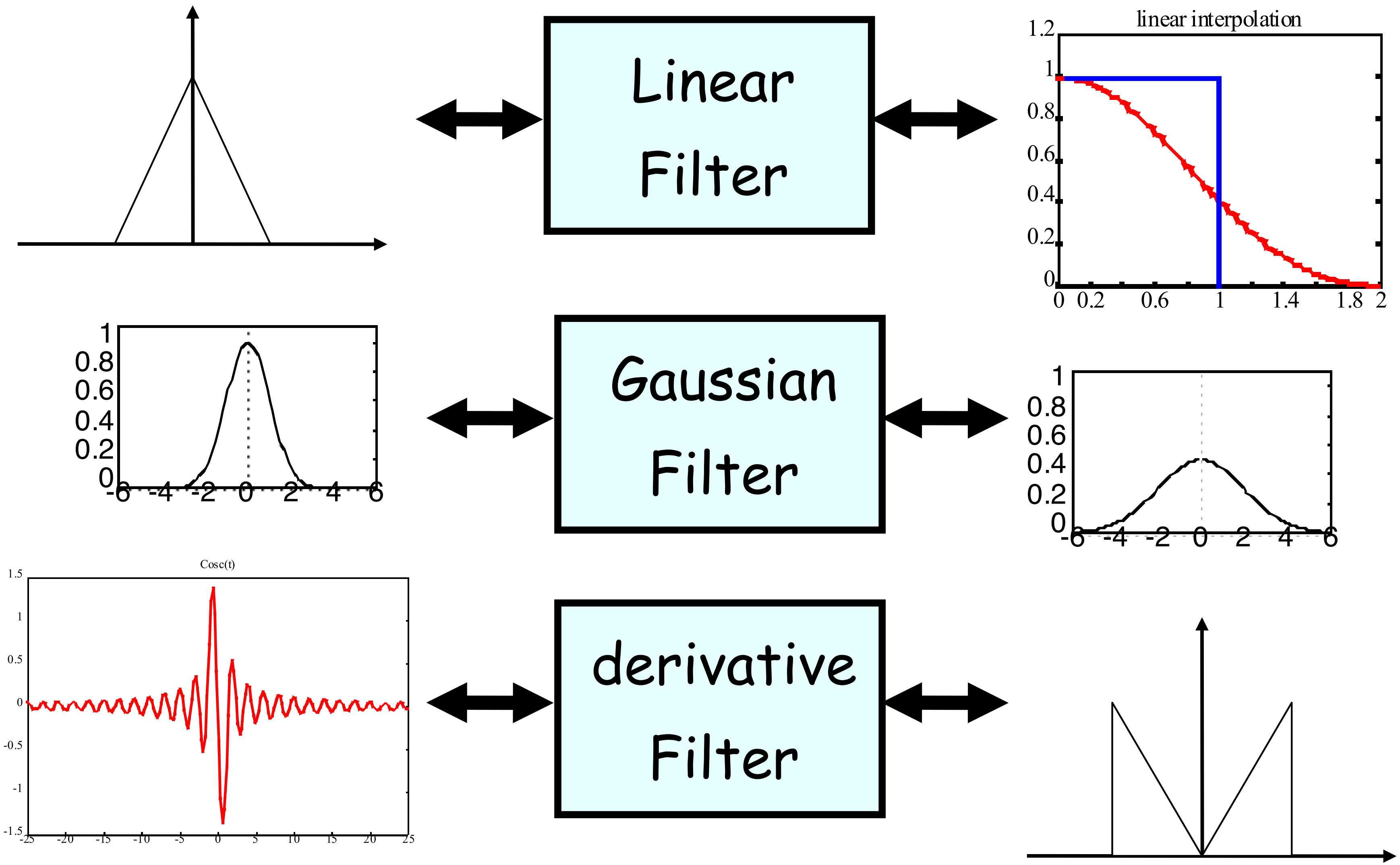
$$=\int_{-\infty}^{+\infty} f(y)\left[\int_{-\infty}^{+\infty} g(z)e^{-i2\pi\omega(y+z)}dz\right]dy \qquad z = x - y$$

$$=\int_{-\infty}^{+\infty} f(y)e^{-i2\pi\omega y}G(\omega)dy = F(\omega)G(\omega)$$

Transforms Pairs



Transforms Pairs (2)

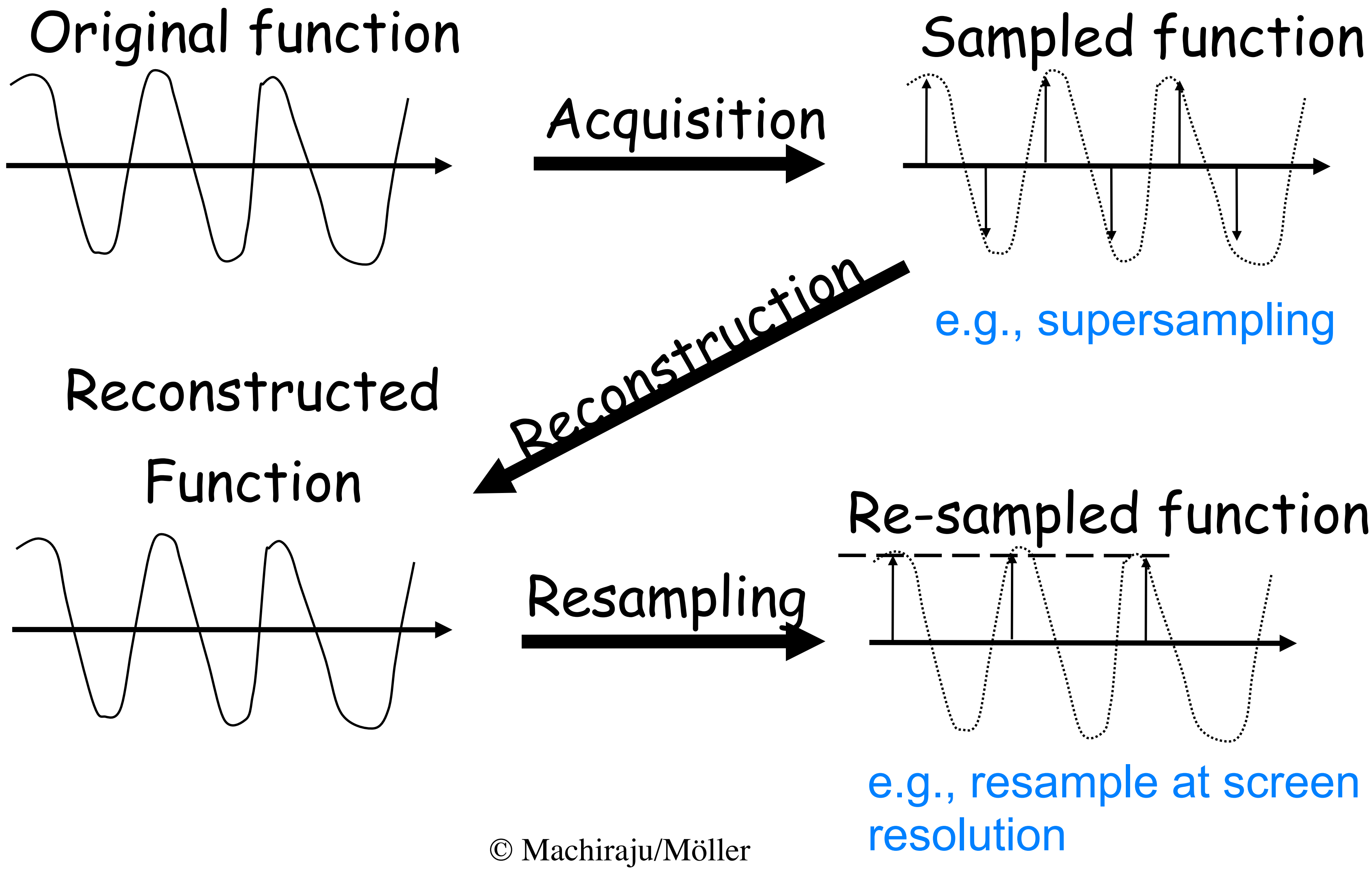


Transform Pairs - Shah

- Sampling = Multiplication with a Shah function:

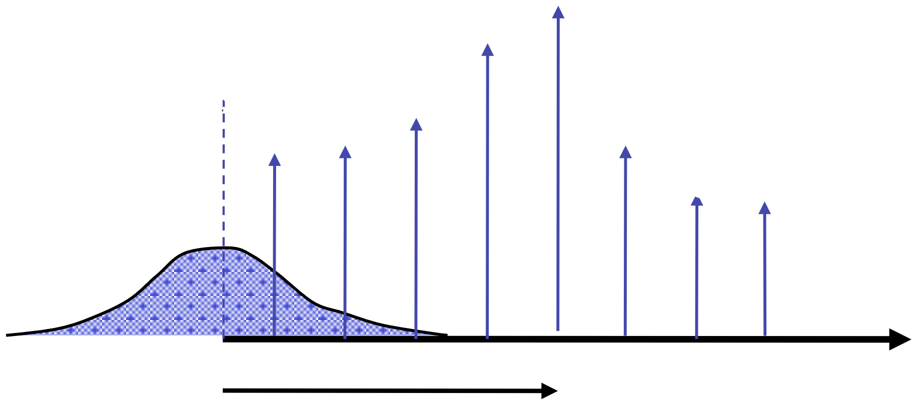
- multiplication in spatial domain = convolution in the frequency domain
- frequency replica of primary spectrum (also called aliased spectra)

General Process of Sampling and Reconstruction



How so? - Convolution Theorem

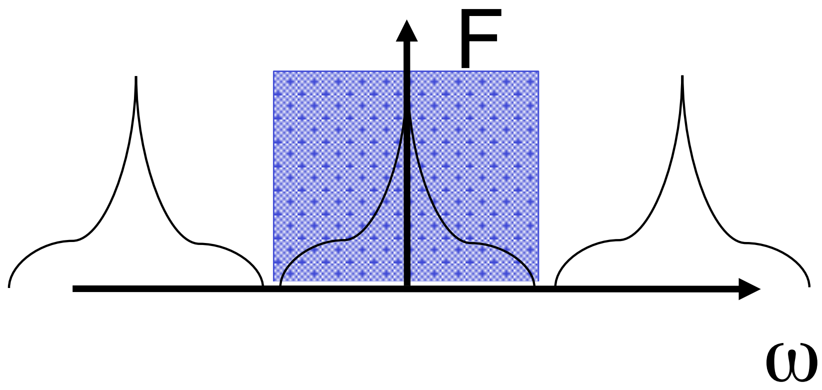
Spatial Domain:



Convolution:

$$\int_{-\infty}^{\infty} f(t) \times g(x-t)\,dt$$

Frequency Domain:



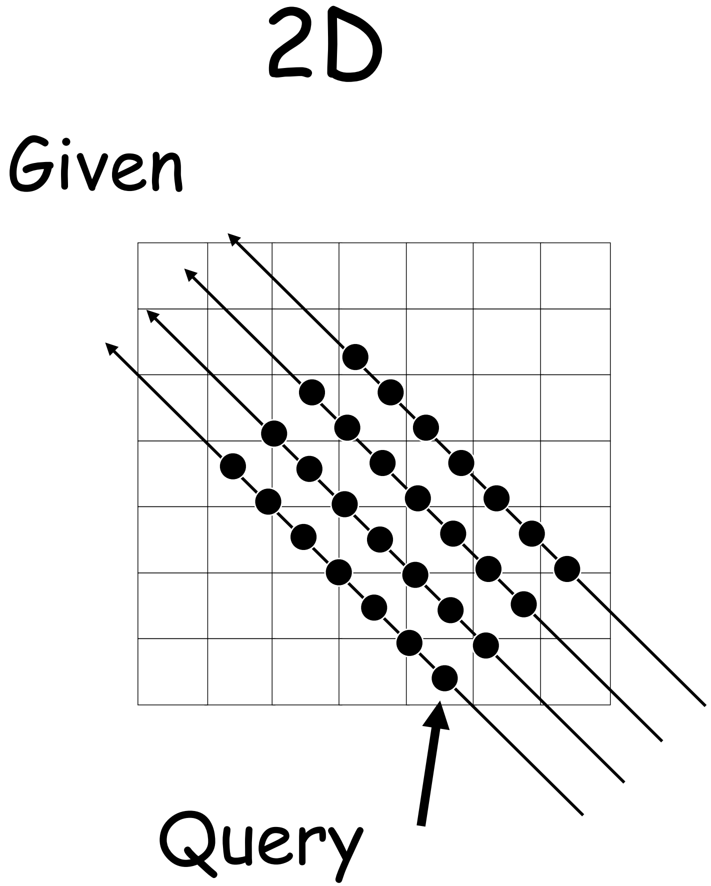
Multiplication:

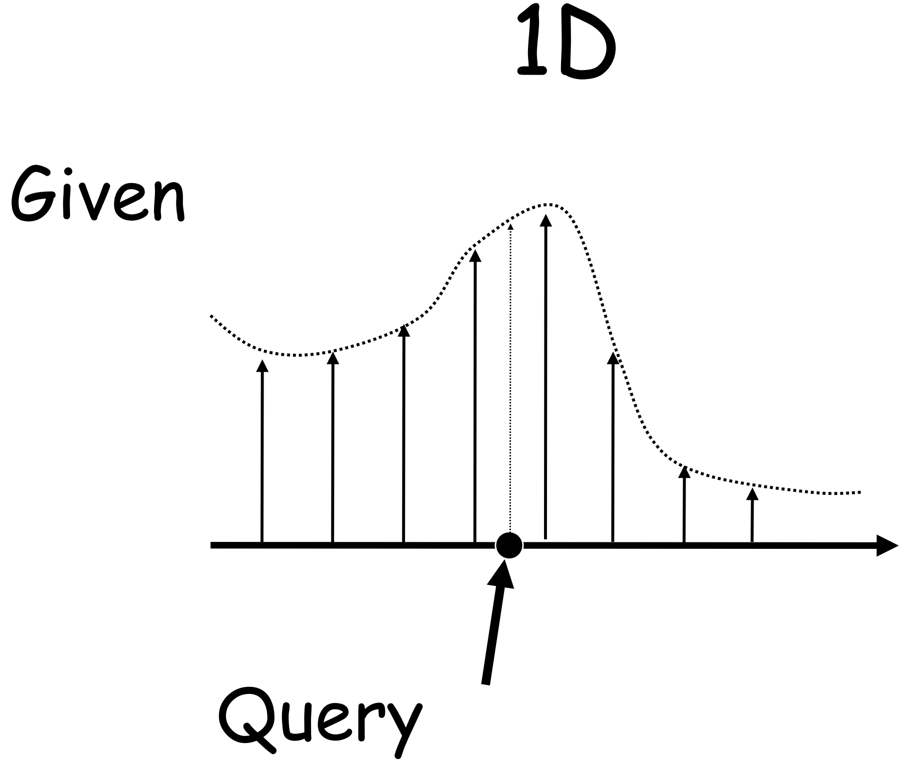
$$F(\omega) \times G(\omega)$$

Sampling Theorem

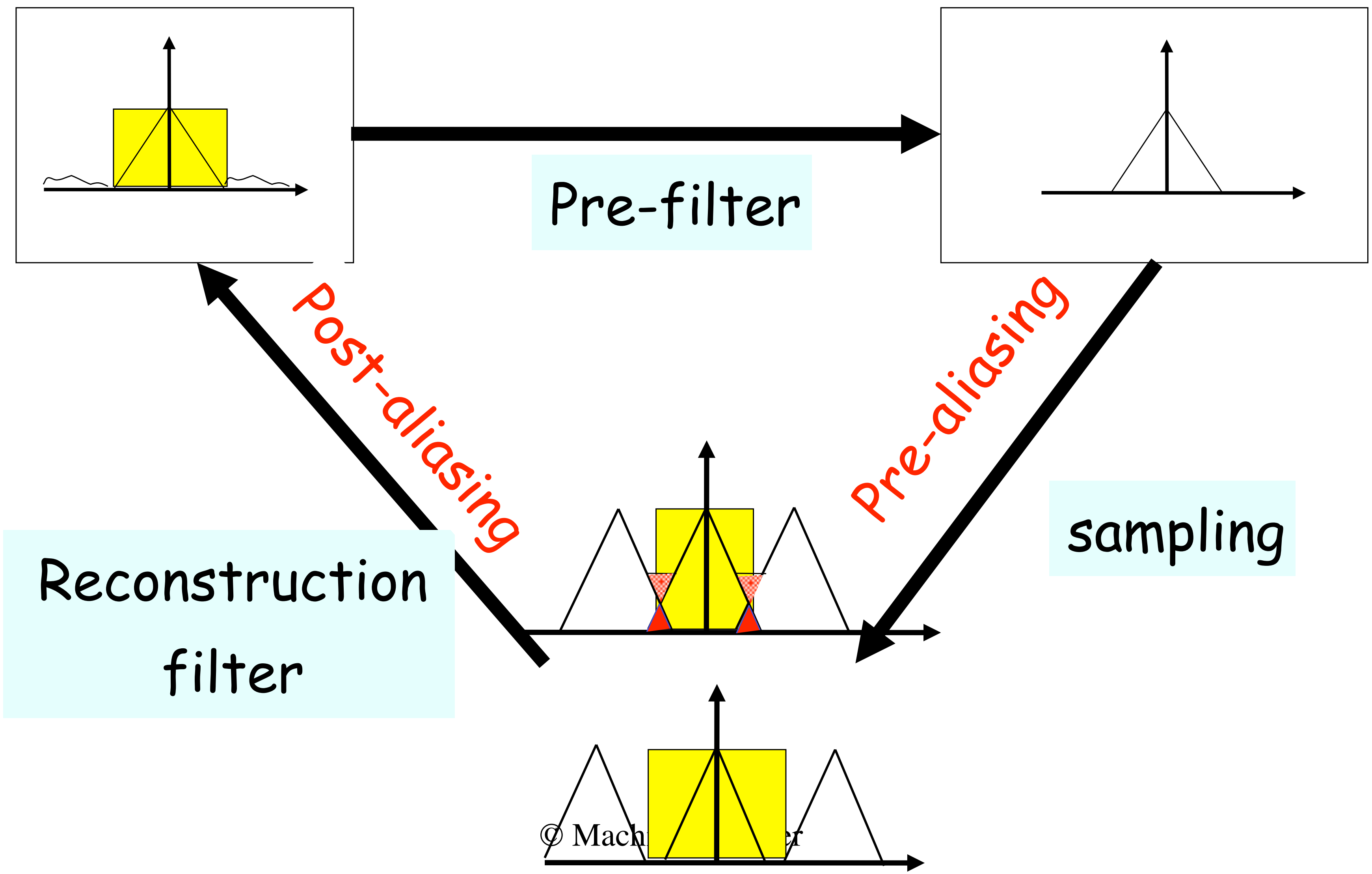
- A signal can be reconstructed from its samples without loss of information if the original signal has no frequencies above 1/2 of the sampling frequency
- For a given bandlimited function, the rate at which it must be sampled (to have perfect reconstruction) is called the *Nyquist frequency*
- Due to Claude Shannon (1949)

Example



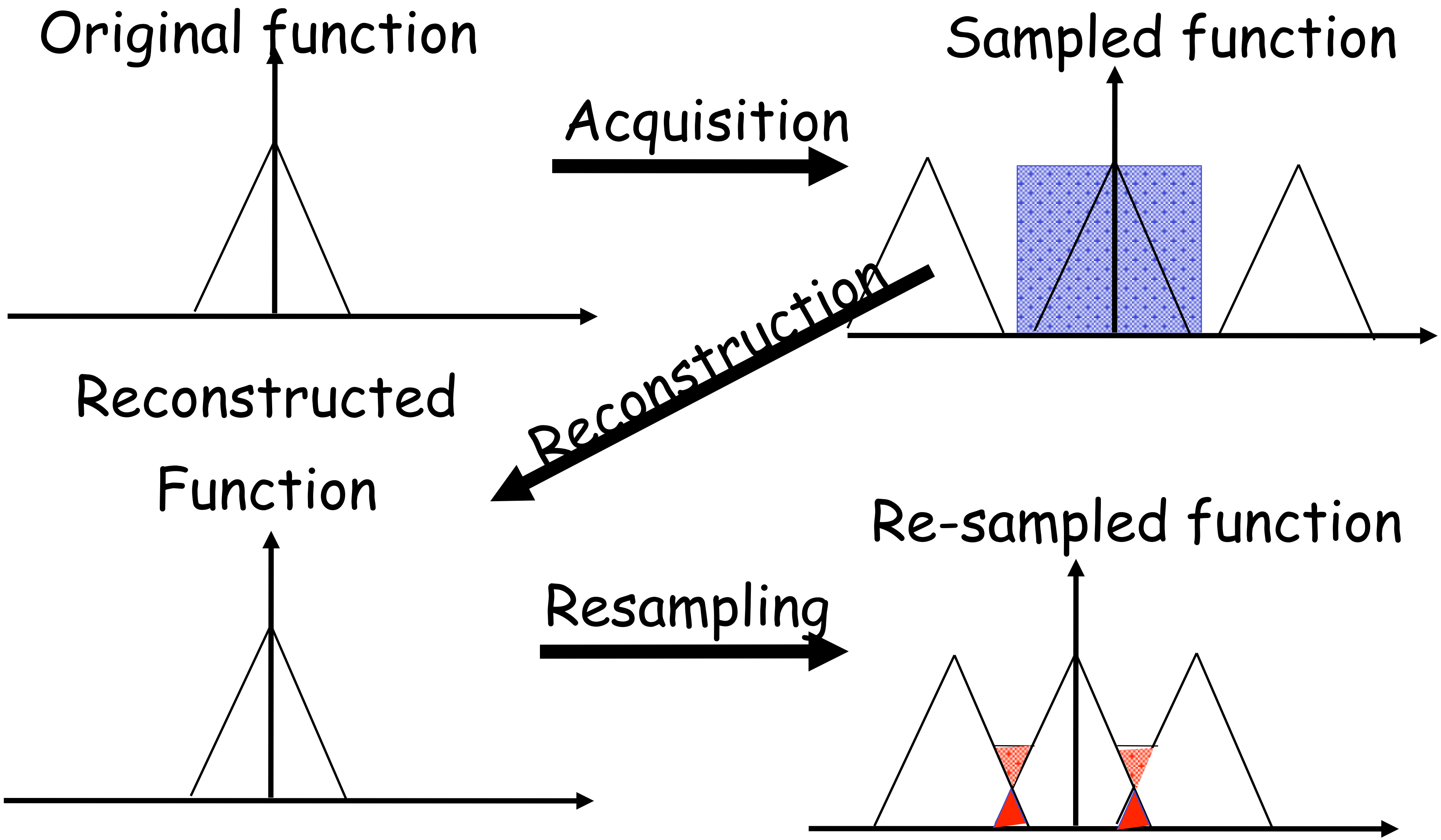


Once Again ...

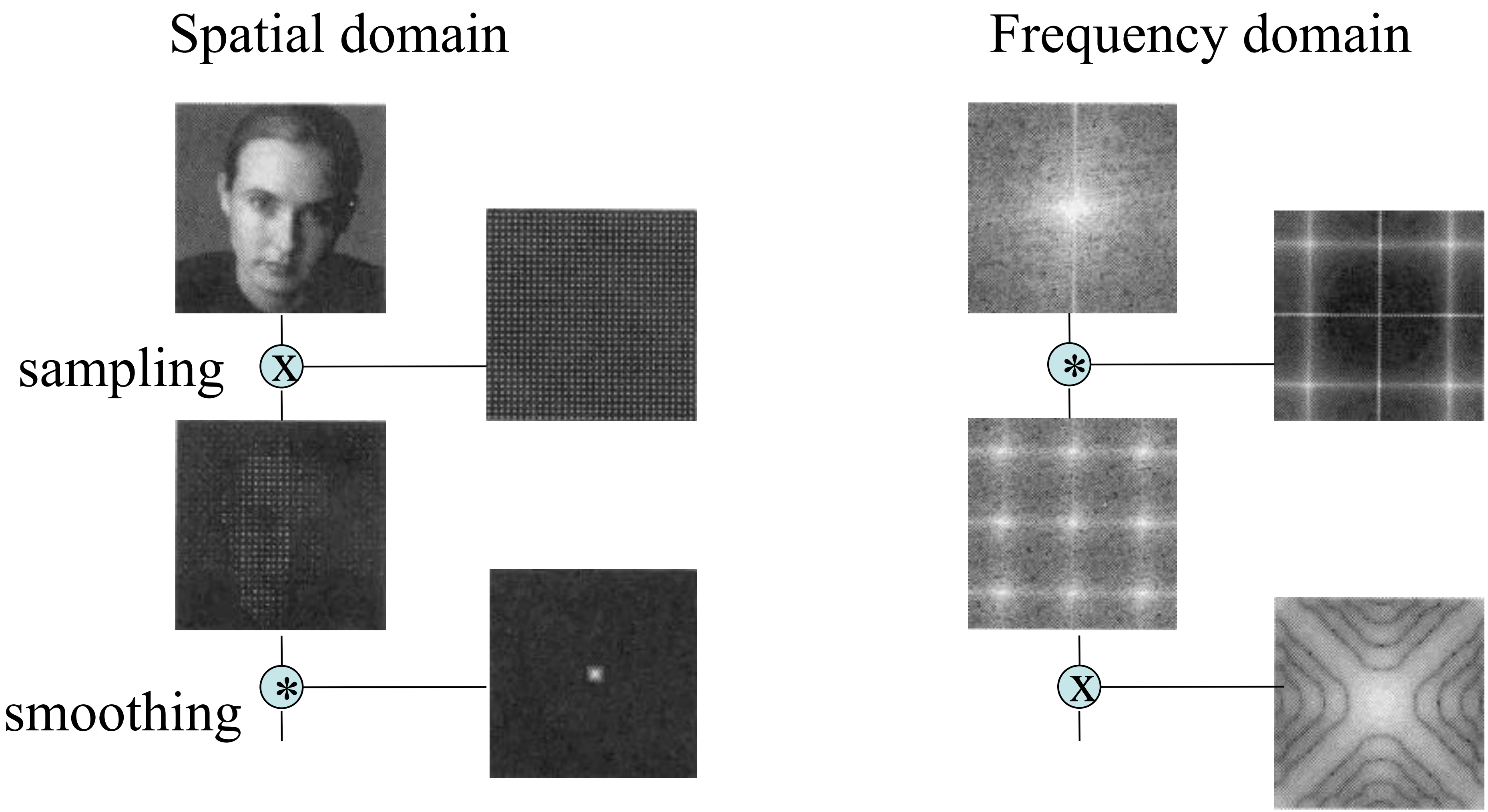


Pre-filter
Post-aliasing
Pre-aliasing
sampling
Reconstruction filter

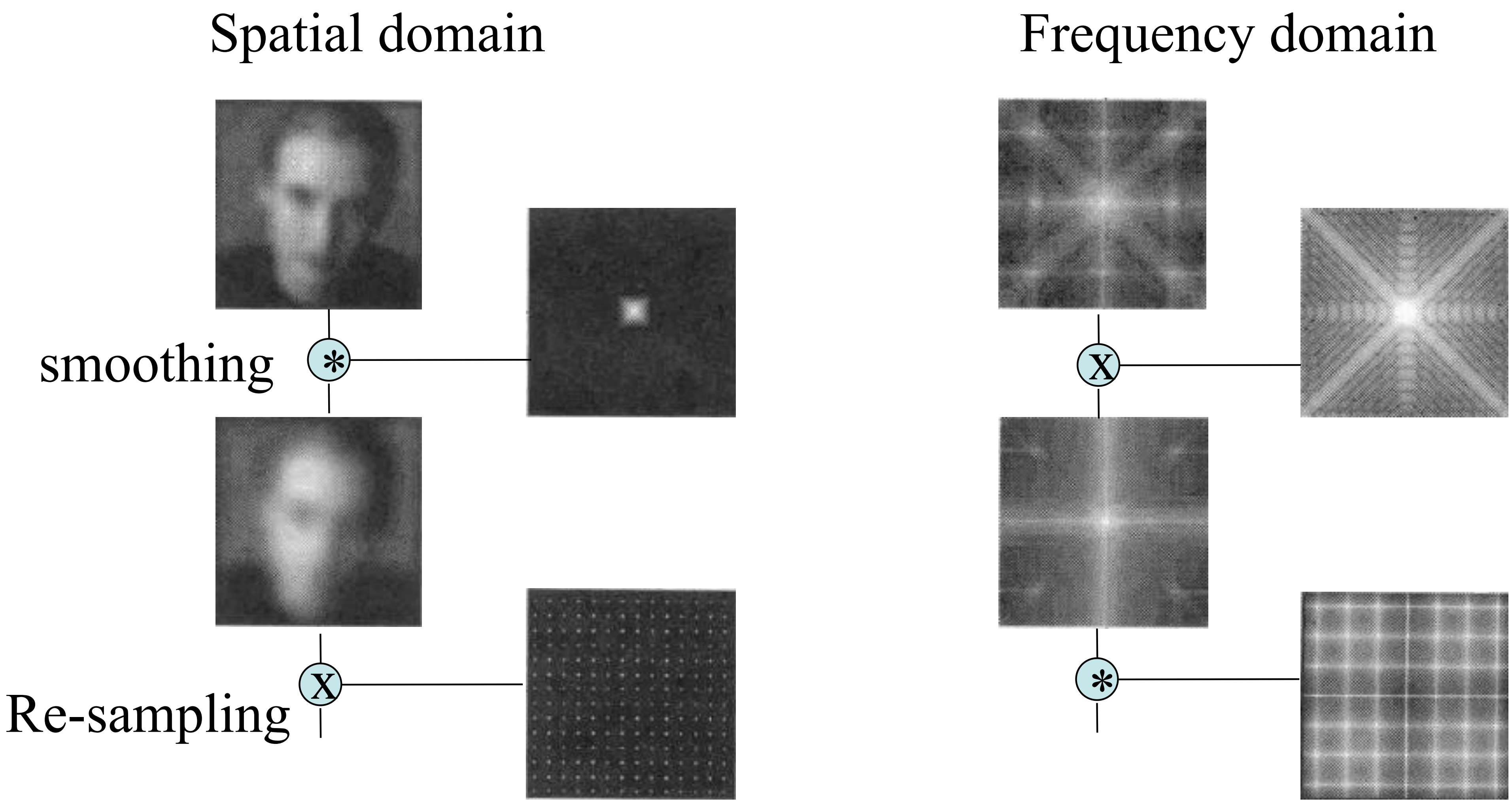
In the frequency domain



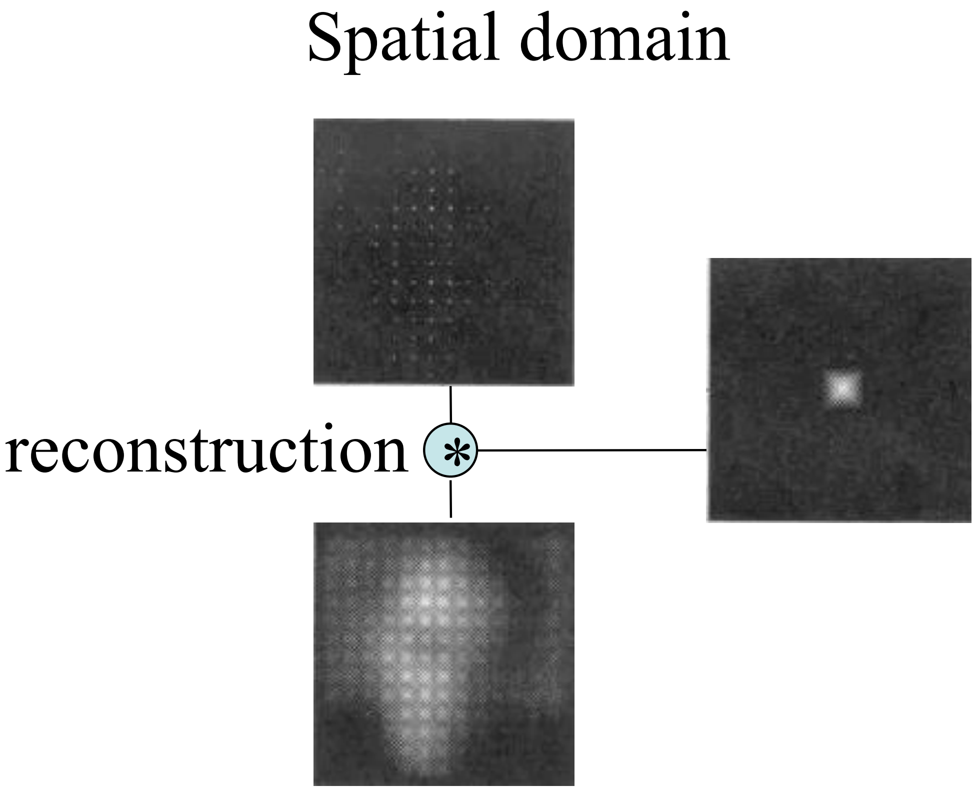
Pipeline - Example

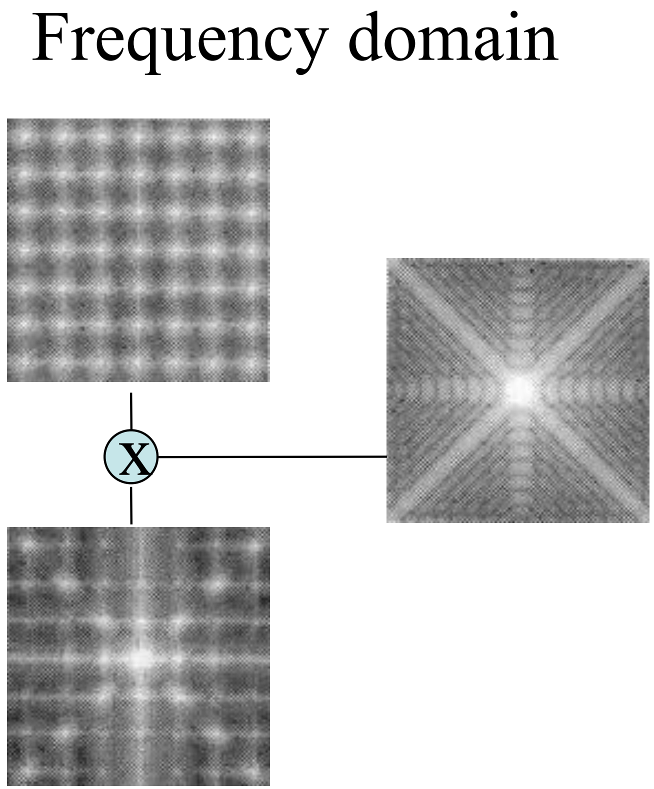


Pipeline - Example (2)



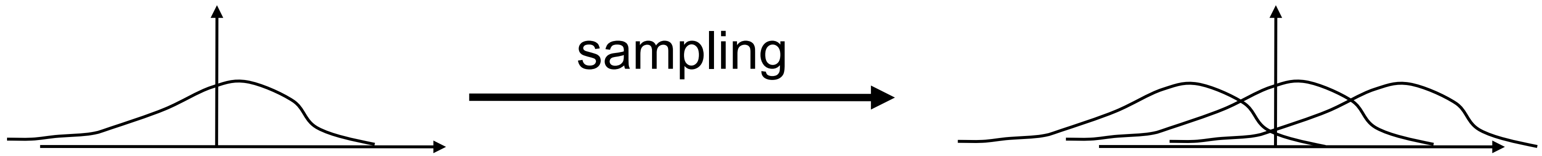
Pipeline - Example (3)





Cause of Aliasing

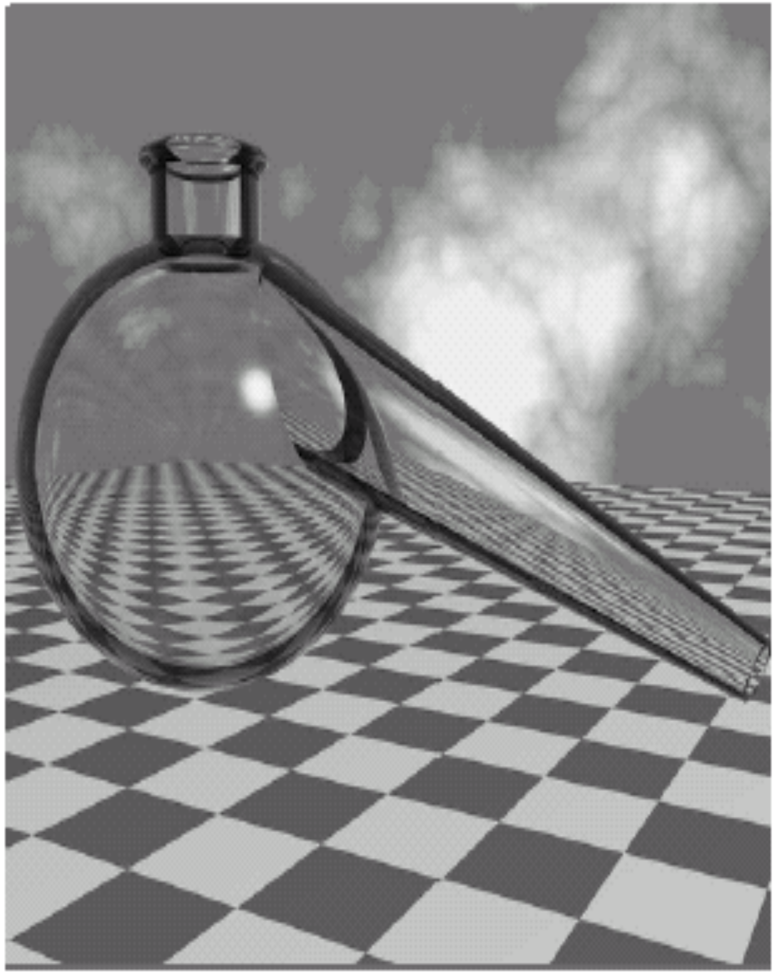
- Non-bandlimited signal – *prealiasing*

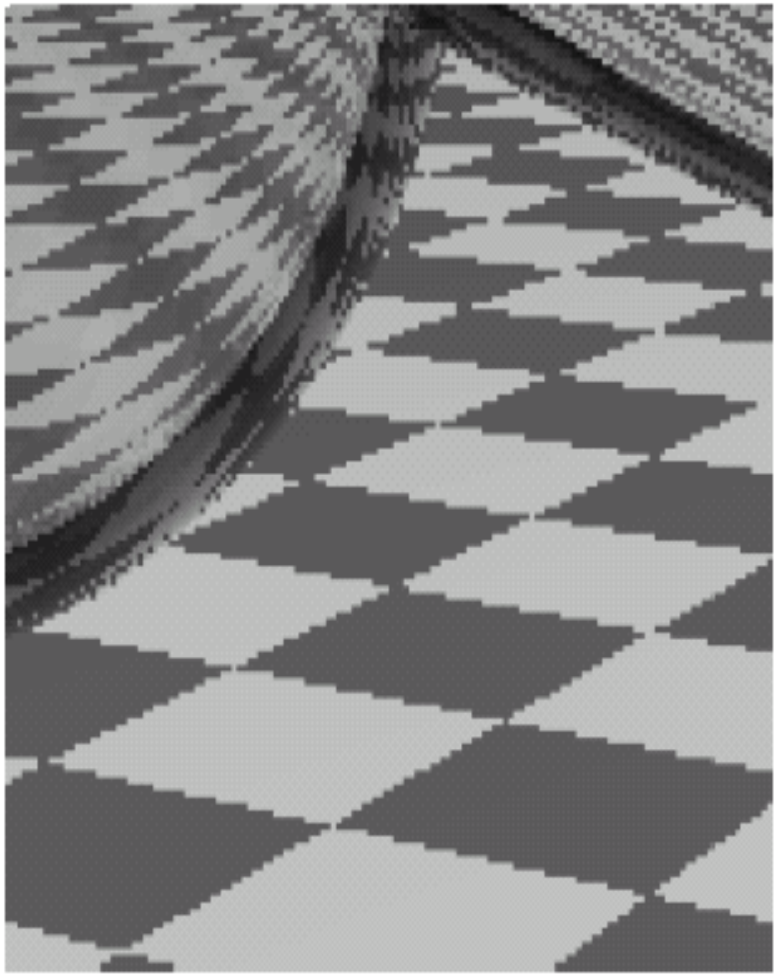


- Low sampling rate (<= Nyquist) – *prealiasing*

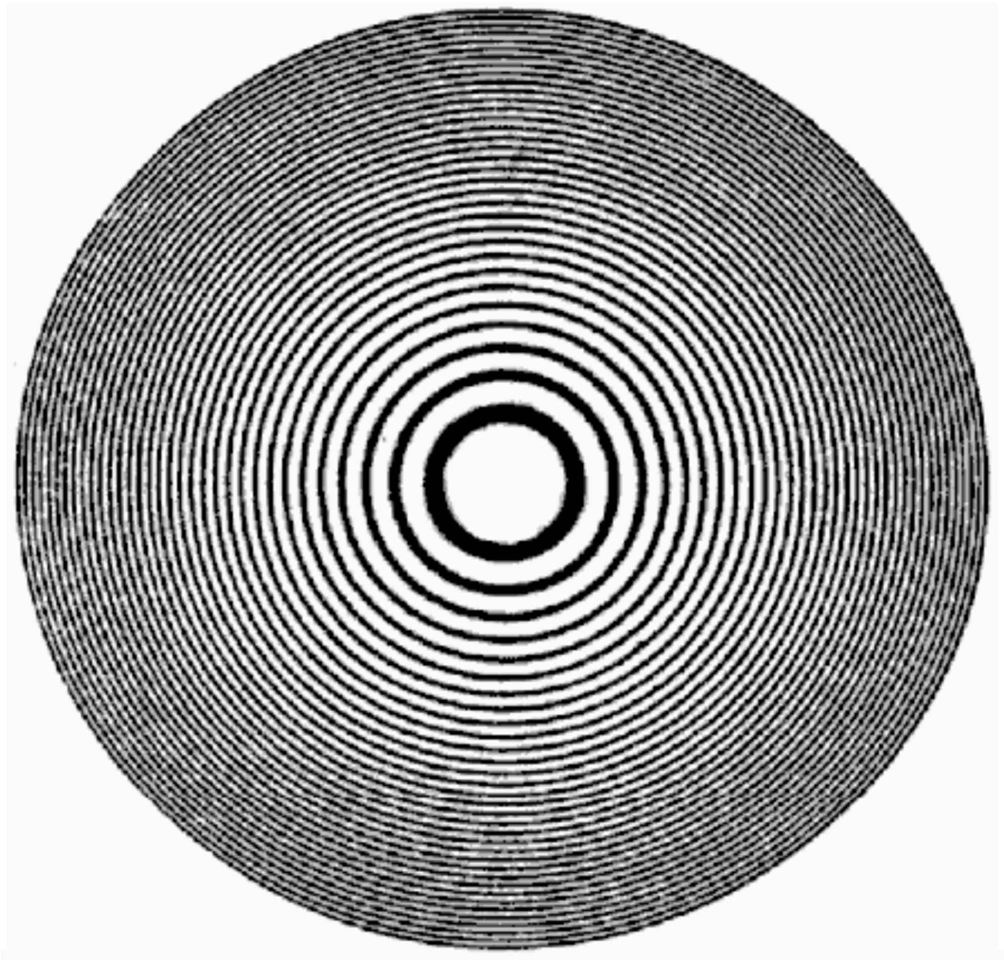
- Non perfect reconstruction – *post-aliasing*

Aliasing example



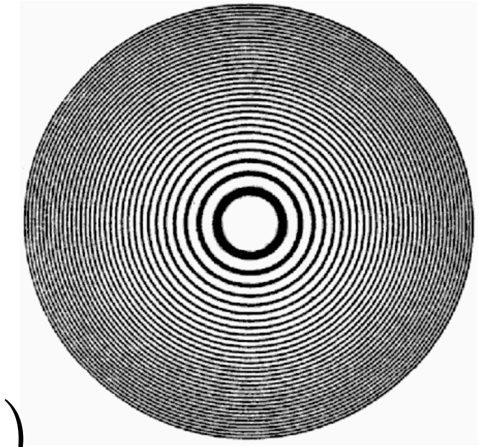


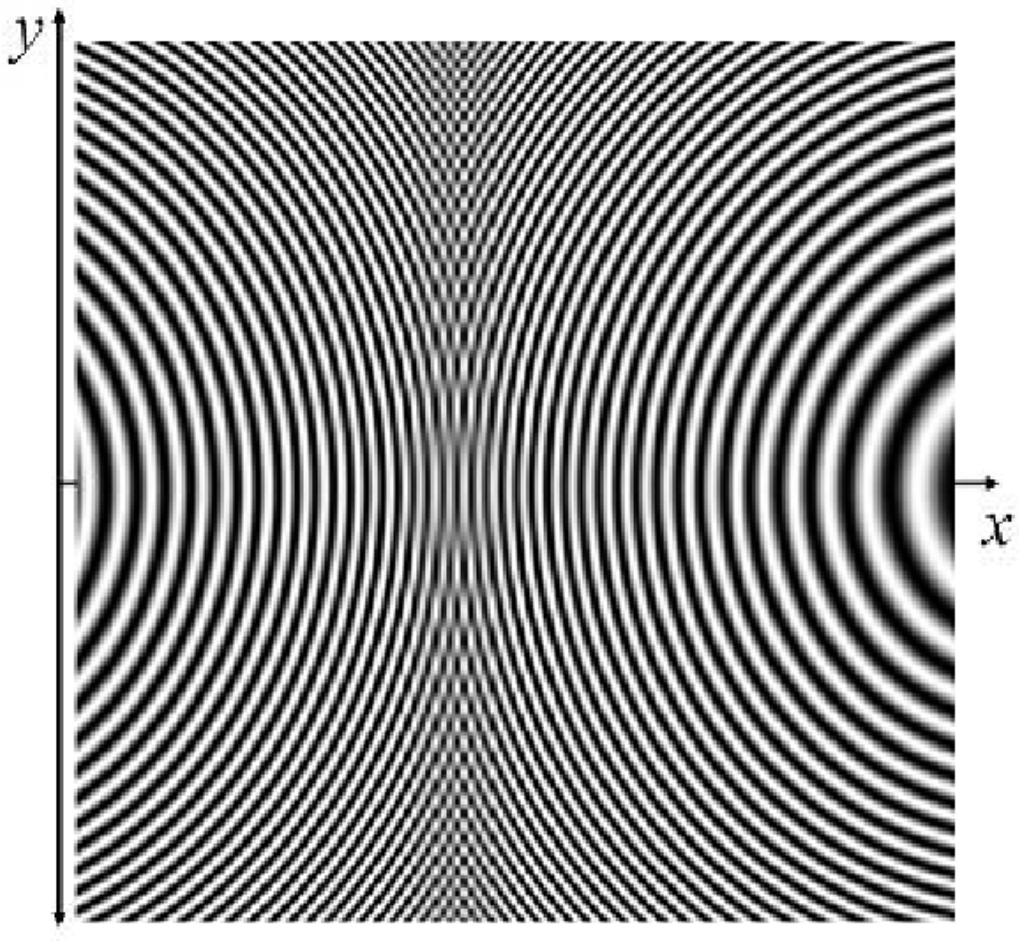
Aliasing: Sampling a Zone Plate



$$\sin(x^2 + y^2)$$

Aliasing: Sampling a Zone Plate



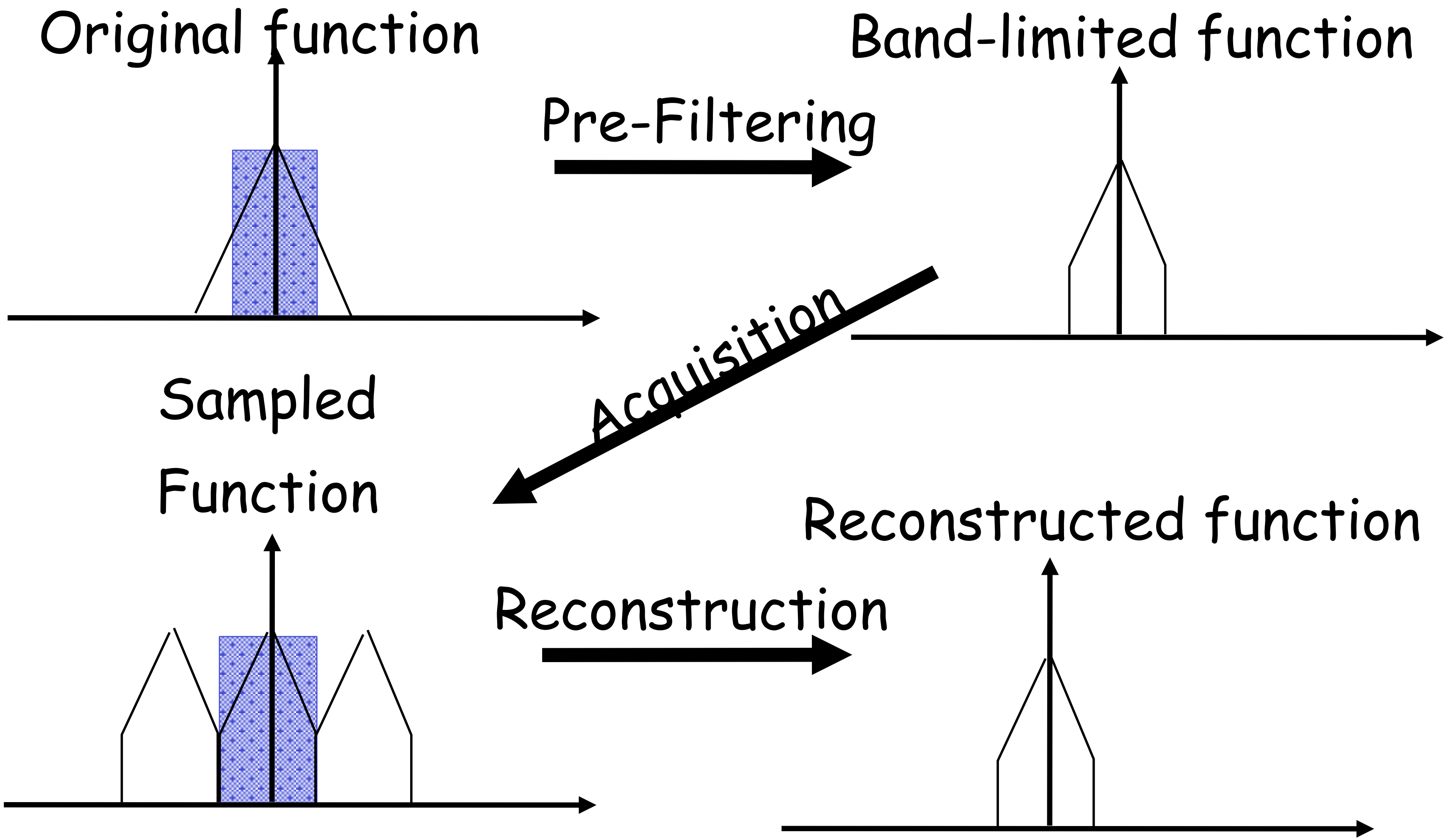


$\sin(x^2 + y^2)$

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

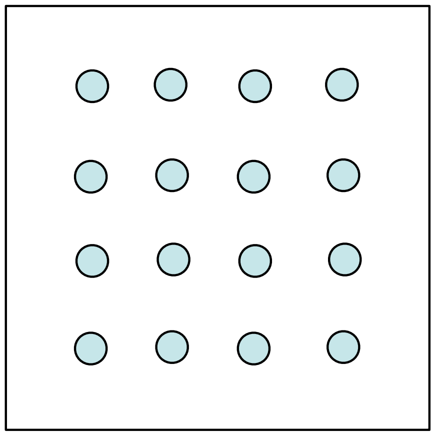
Left rings: part of the signal
Right rings: aliasing due to undersampling

Antialiasing 1: Pre-Filtering

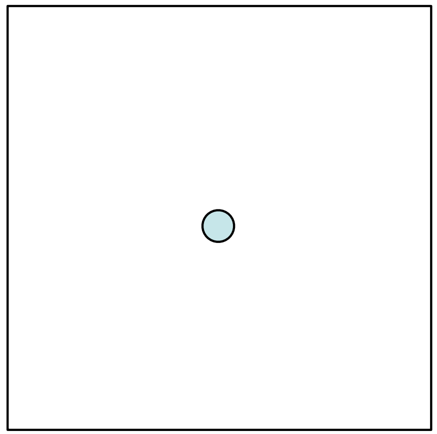


Antialiasing 2: Uniform Supersampling

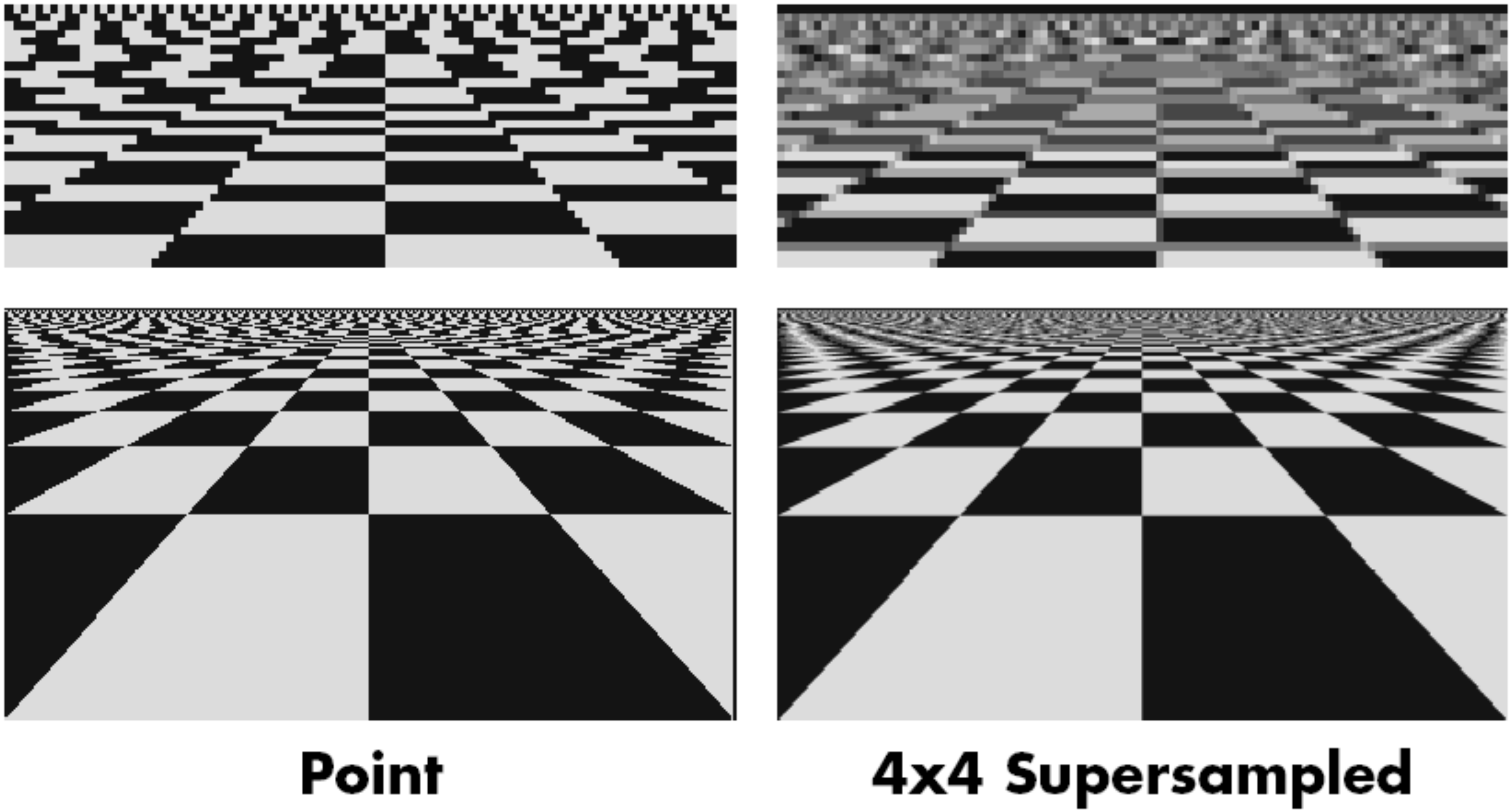
- Increasing the sampling rate moves each copy of the spectra further apart, potentially reducing the overlap and thus aliasing
- Low-pass filter and then the resulting signal is re-sampled at image resolution



$$Pixel = \sum_k w_k \times Sample_k$$



Point vs. supersampling



Checkerboard sequence by Tom Duff

Summary: Antialiasing

- Antialiasing = Preventing aliasing

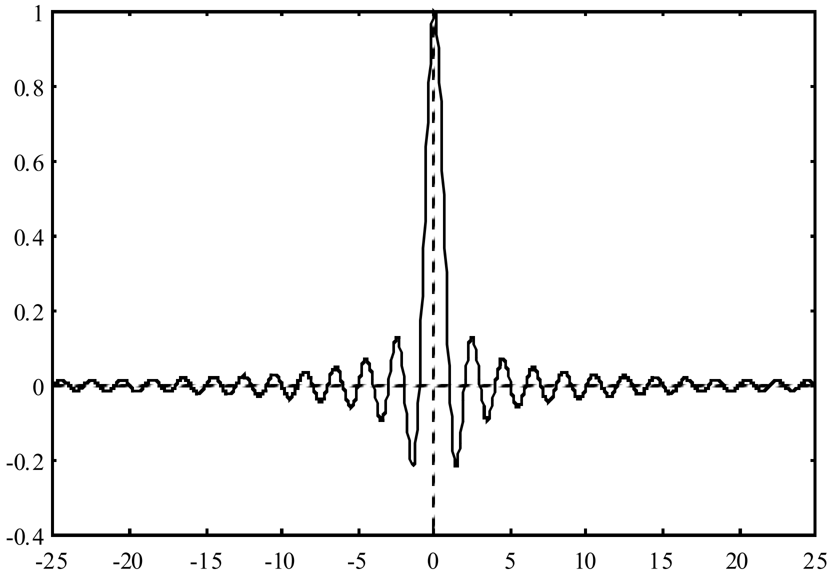
1. Analytically pre-filter the signal
   - Solvable for points, lines and polygons
   - Not solvable in general (e.g. procedurally defined images)
2. Uniform supersampling and resample
3. Nonuniform or stochastic sampling – later!

Reconstruction = Interpolation

Spatial Domain:

- convolution is exact

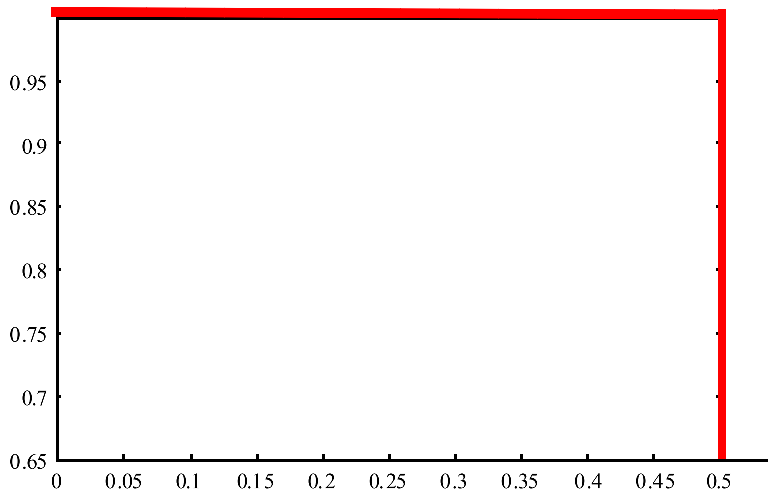
$$f_r(x) - f(x) = 0$$



Frequency Domain:

- cut off freq. replica

$$\mathrm{Sinc}(x) = \frac{\sin(\pi x)}{\pi x}$$

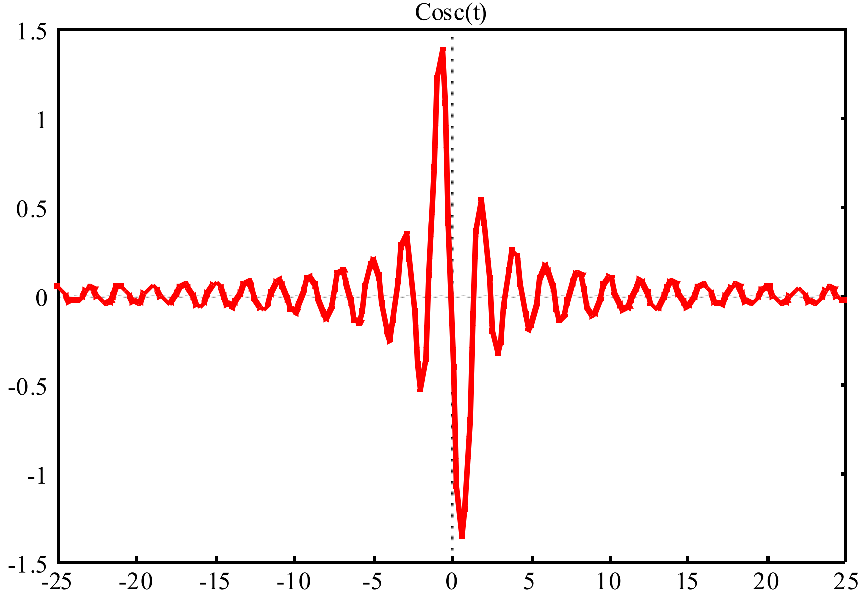


Example: Derivatives

Spatial Domain:

- convolution is exact

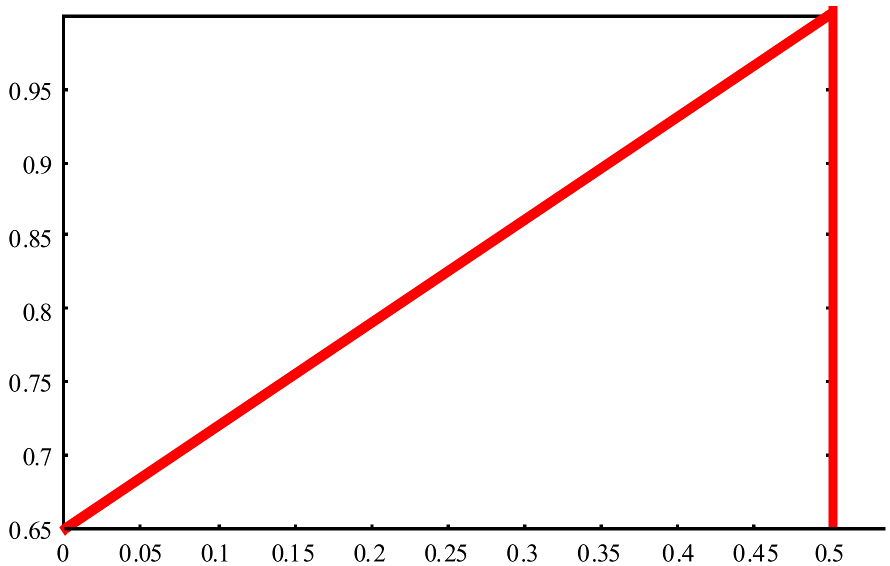
$$f_r^d(x) - f'(x) = 0$$



Frequency Domain:

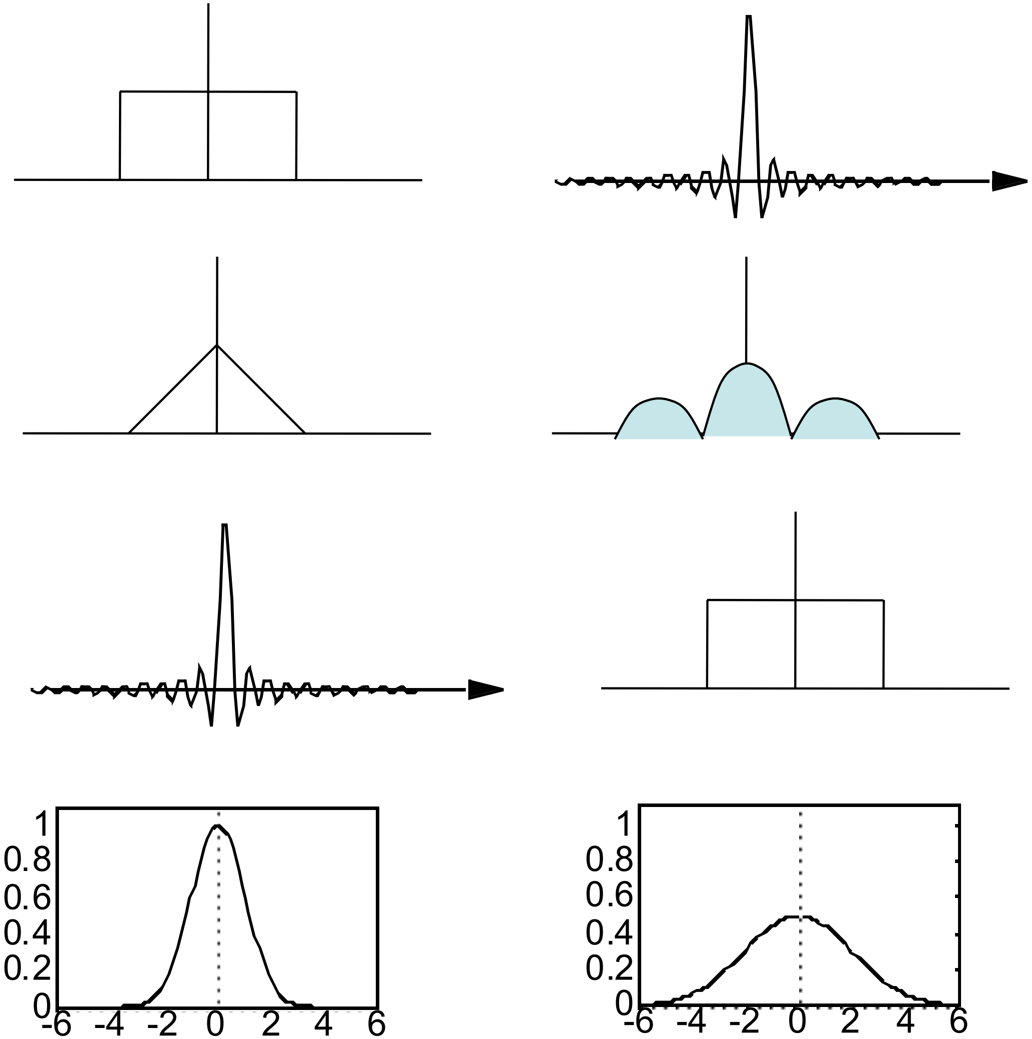
- cut off freq. replica

$$\mathrm{Cosc}(x) = \frac{\cos(\pi x)}{x} - \frac{\sin(\pi x)}{\pi x^2}$$



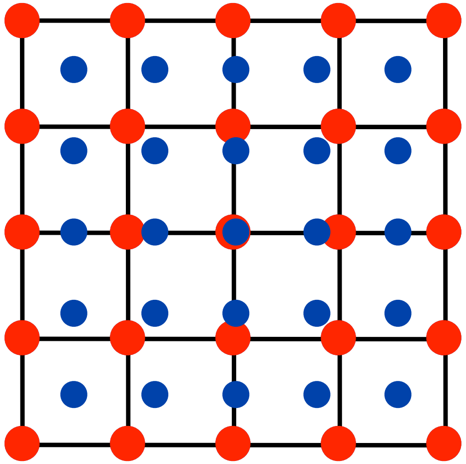
Reconstruction Kernels

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



Spatial d. Frequency d.

Interpolation example

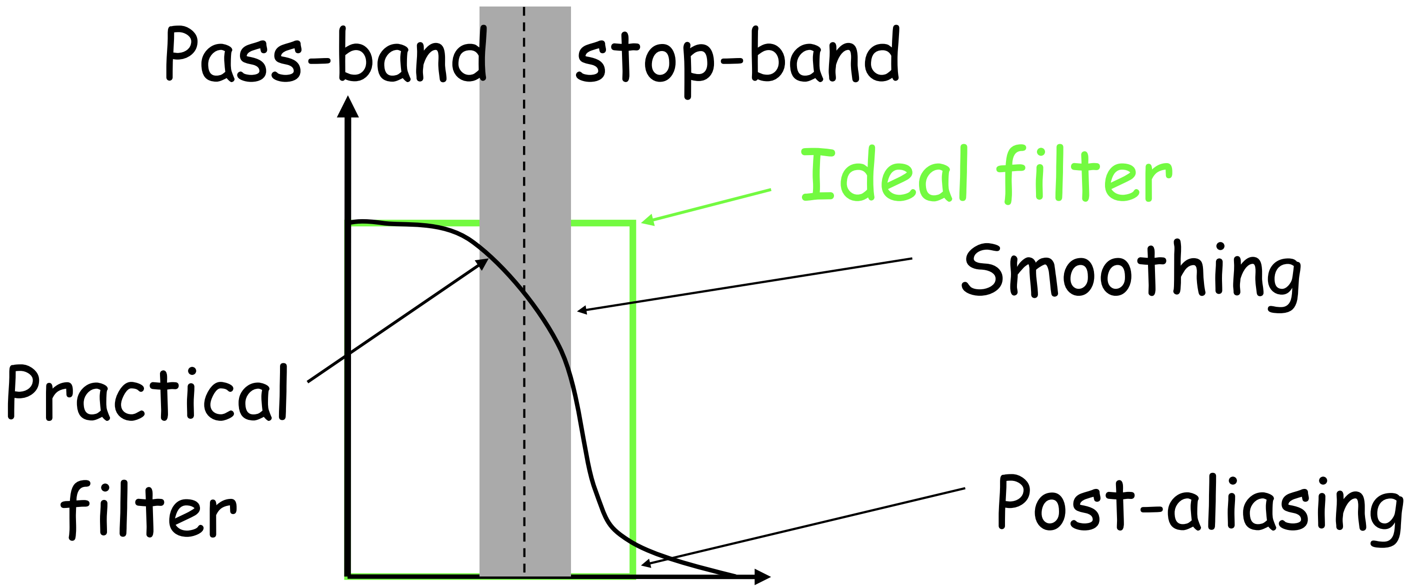


Nearest neighbor

Linear Interpolation

Ideal Reconstruction

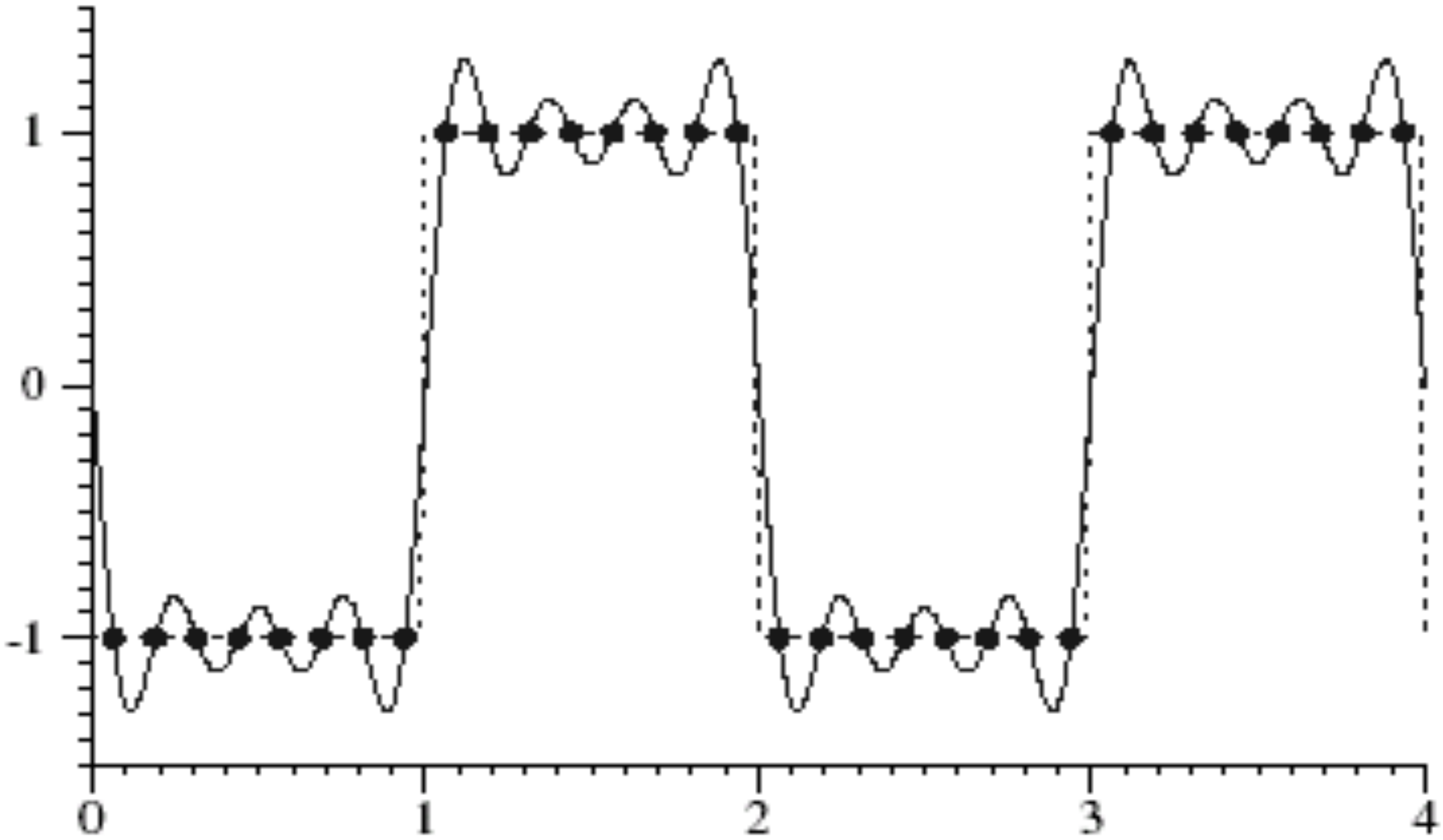
- Box filter in frequency domain =
- Sinc Filter in spatial domain
- Sinc has *infinite* extent – not practical



Ideal Reconstruction

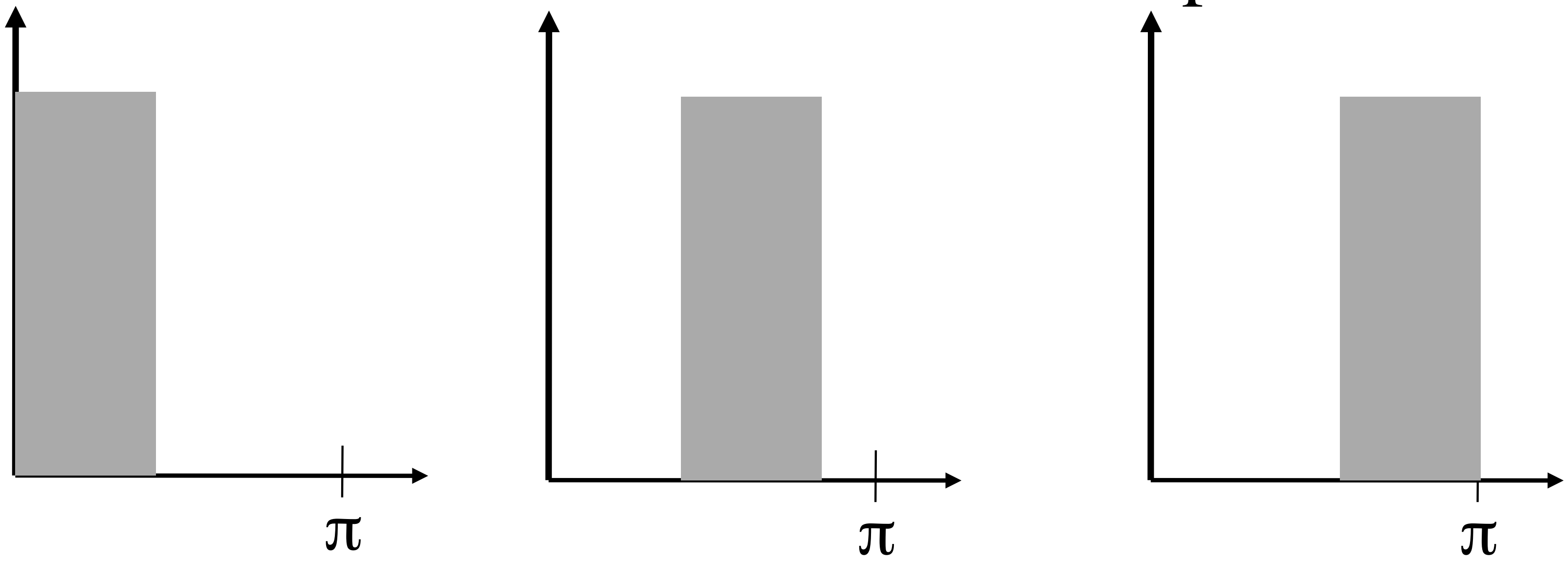
- Use the sinc function – to bandlimit the sampled signal and remove all copies of the spectra introduced by sampling
- But:
  - The sinc has infinite extent and we must use simpler filters with finite extents.
  - The windowed versions of sinc may introduce ringing artifacts which are perceptually objectionable.

Reconstructing with Sinc: Ringing



Ideal filters

– Also have ringing in pass/stop bands
– Realizable filters do not have sharp transitions



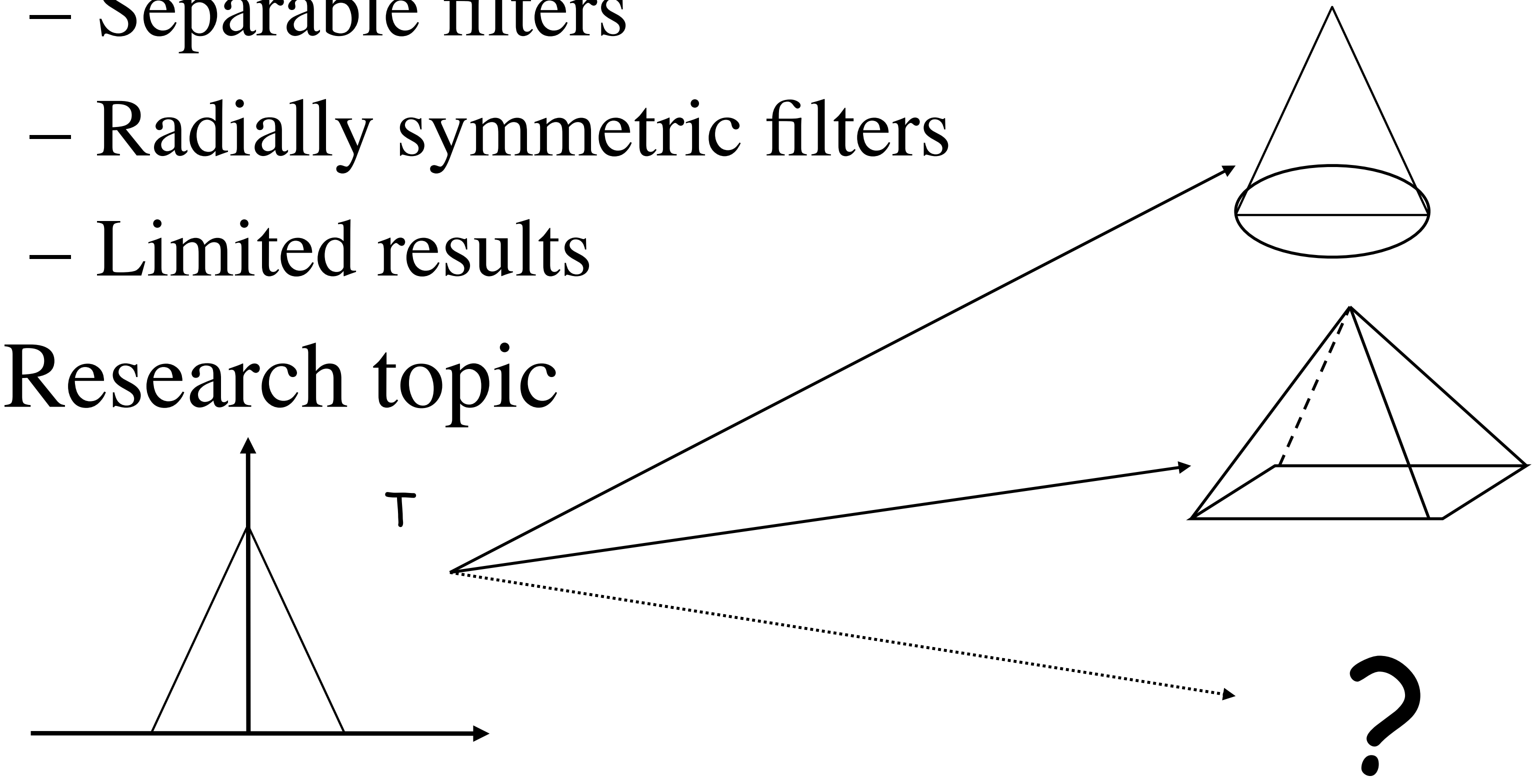
Low-pass filter   band-pass filter   high-pass filter

Summary: possible errors

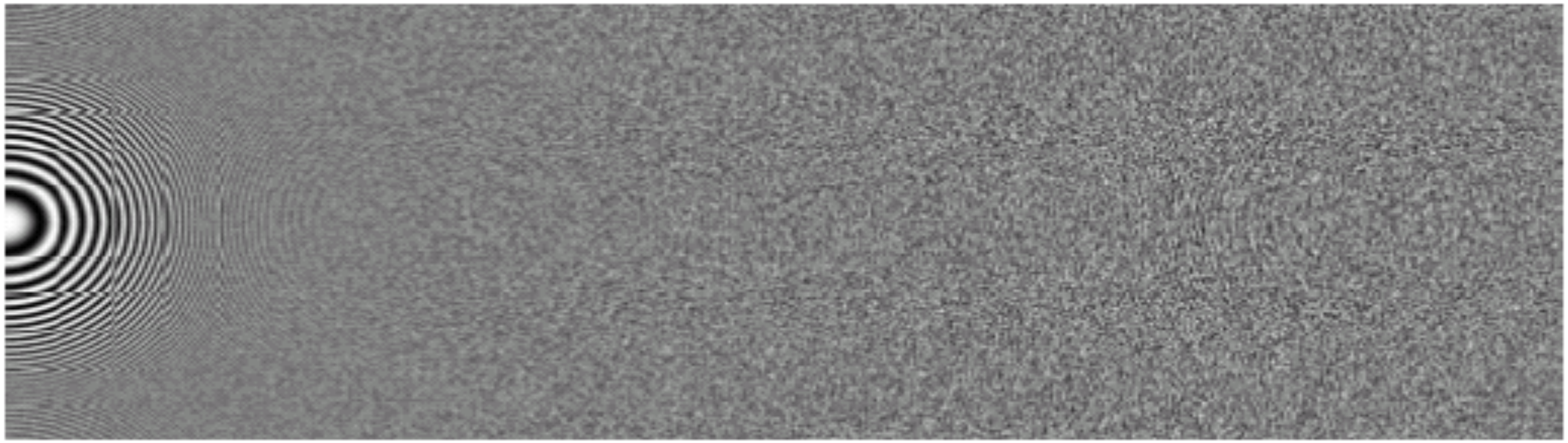
- Post-aliasing
  - reconstruction filter passes frequencies beyond the Nyquist frequency (of duplicated frequency spectrum) => frequency components of the original signal appear in the reconstructed signal at different frequencies
- Smoothing due to prefiltering
  - frequencies below the Nyquist frequency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



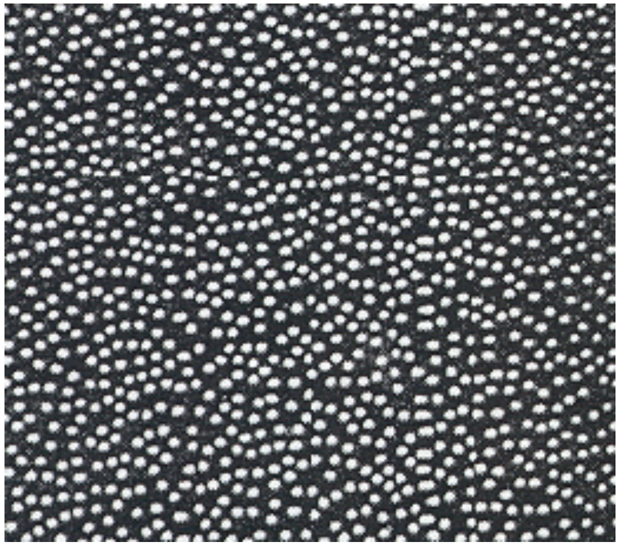
Aliasing vs. Noise



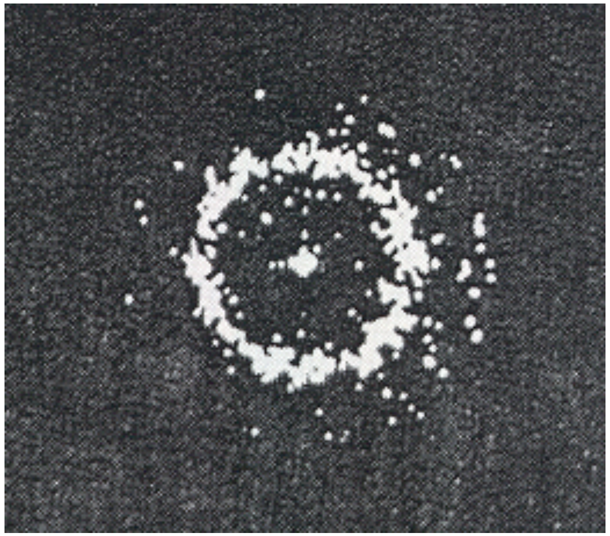
Distribution of Extrafoveal Cones

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

Monkey eye cone distribution



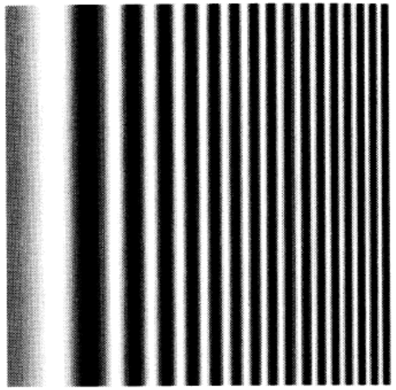
Fourier Transform



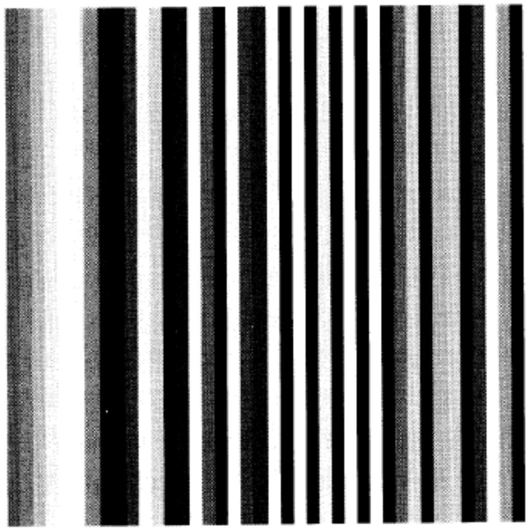
Non-Uniform Sampling - Intuition

- 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

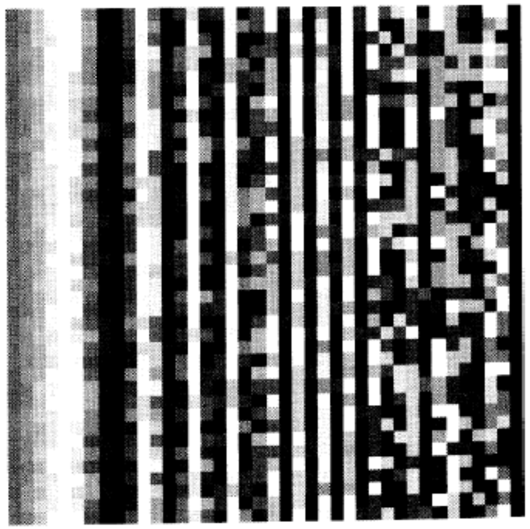


(b)



(c)

Uniformly sampled
40x40

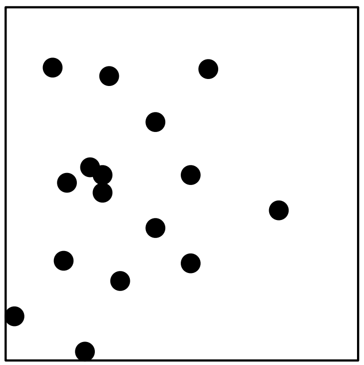


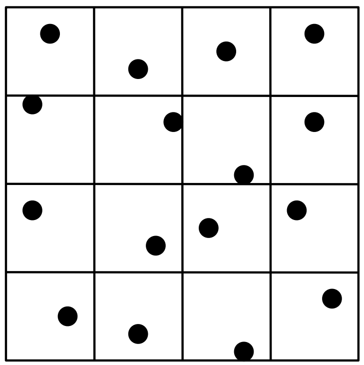
(d)

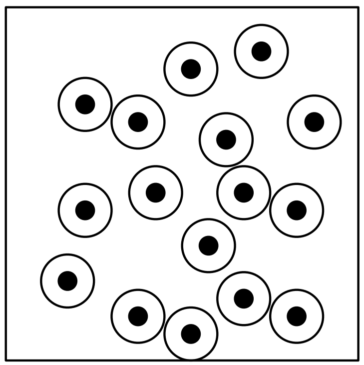
Uniformly jittered
40x40

Non-Uniform Sampling Patterns

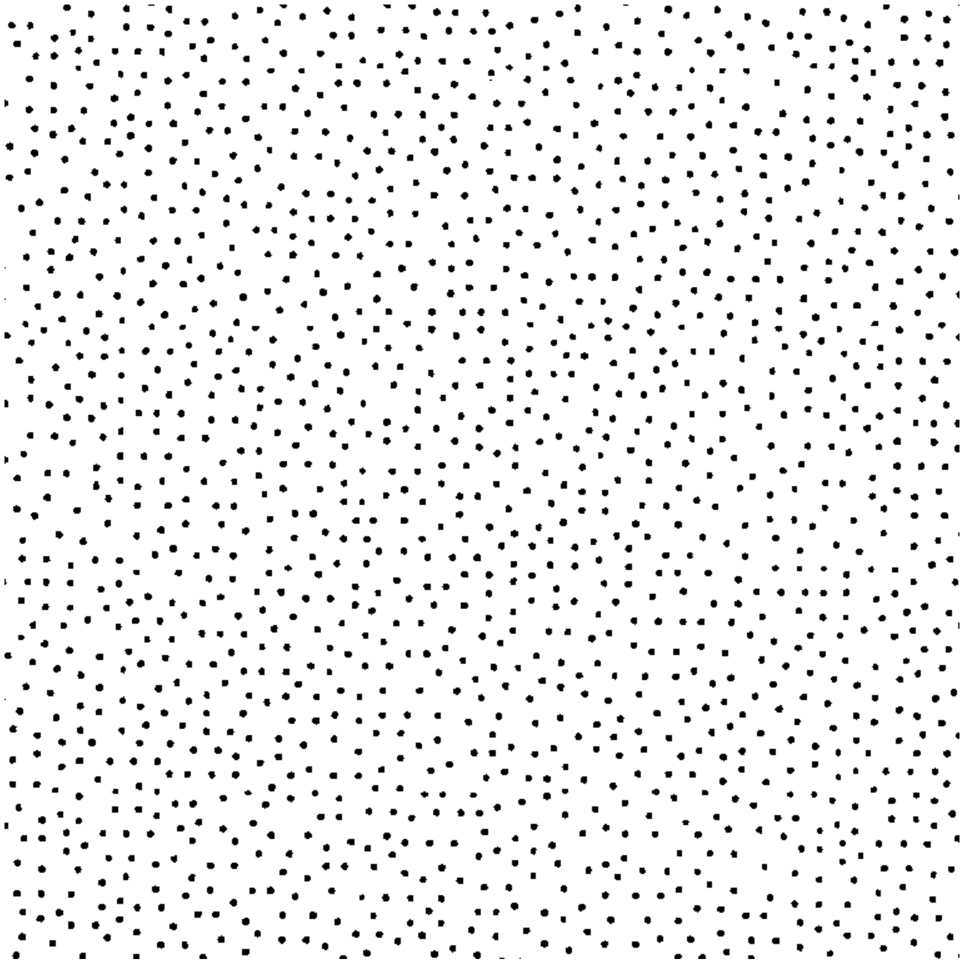
- Poisson
  - Pick n random points in sample space
- Uniform Jitter
  - Subdivide sample space into n regions
- Poisson Disk
  - Pick n random points, but not too close



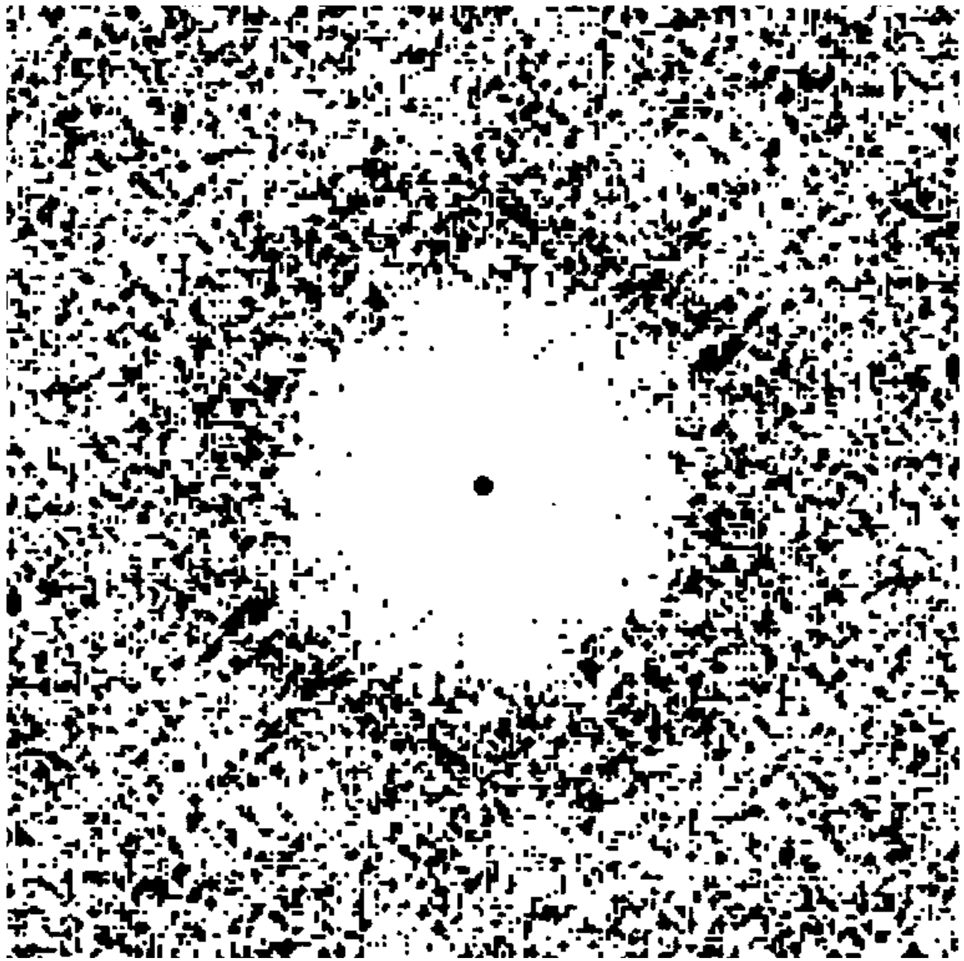




Poisson Disk Sampling

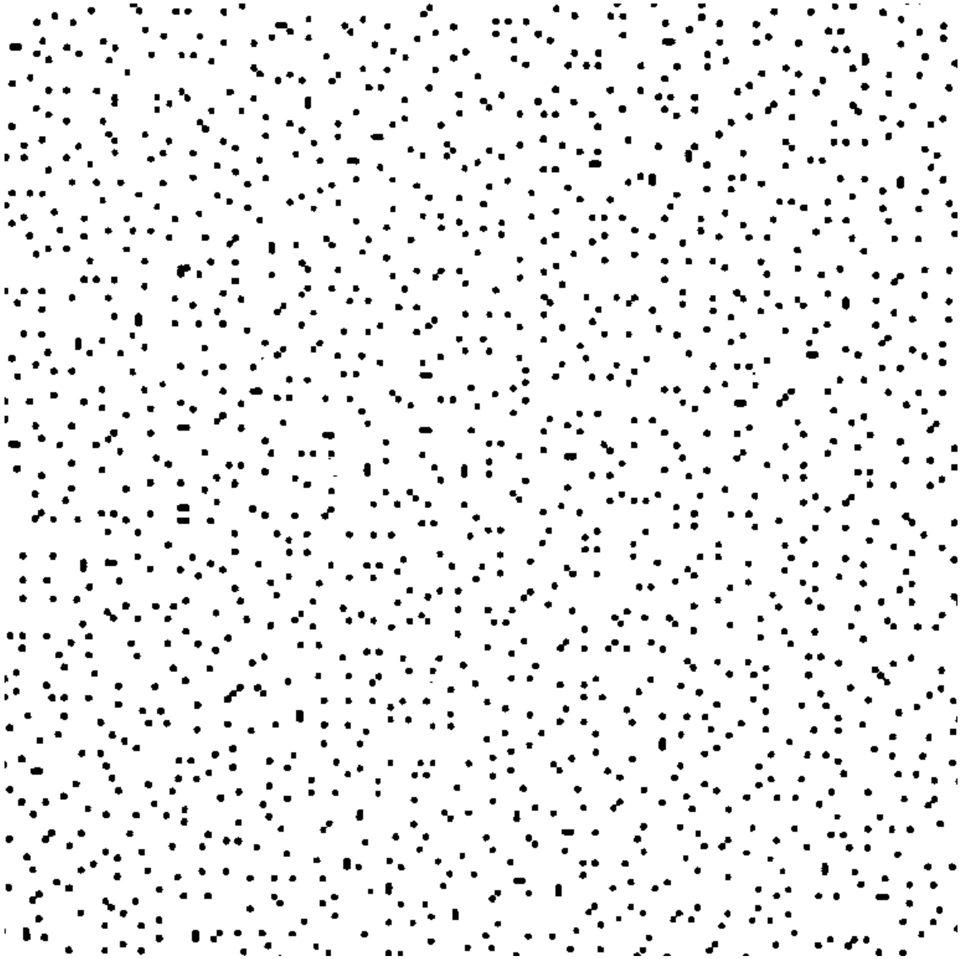


Spatial Domain

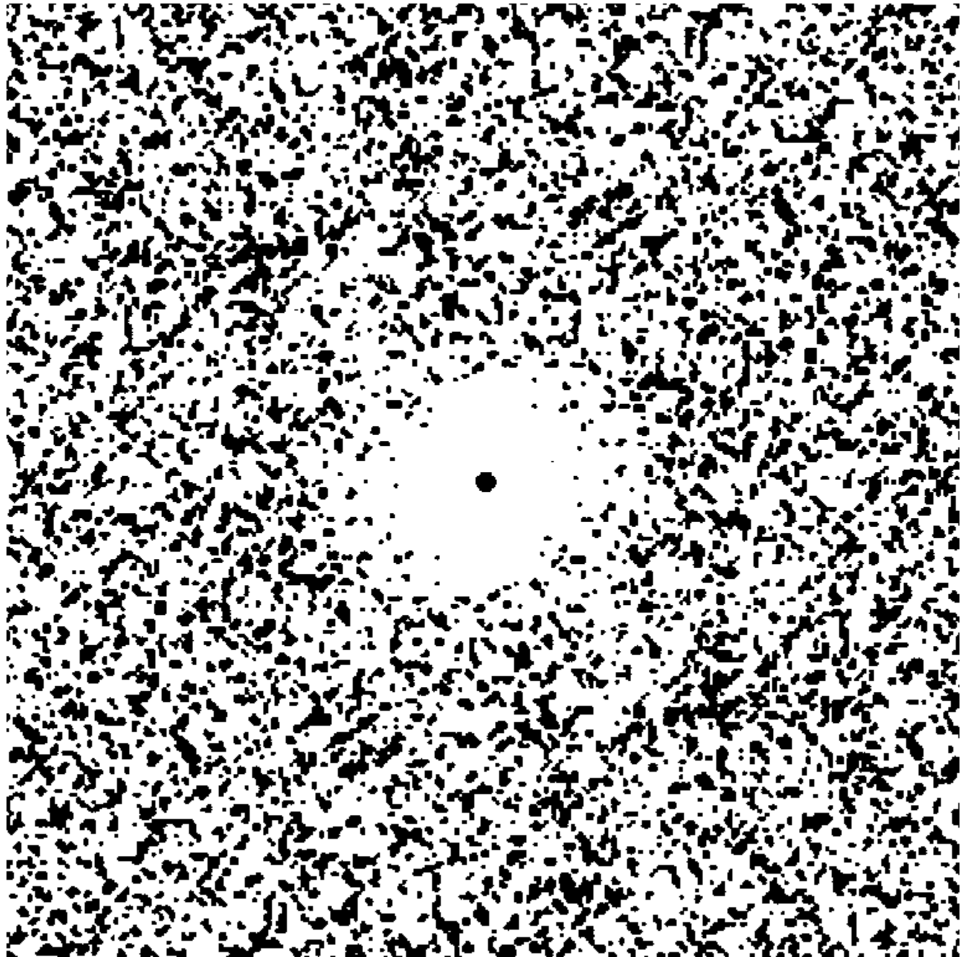


Fourier Domain

Uniform Jittered Sampling



Spatial Domain



Fourier Domain

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.

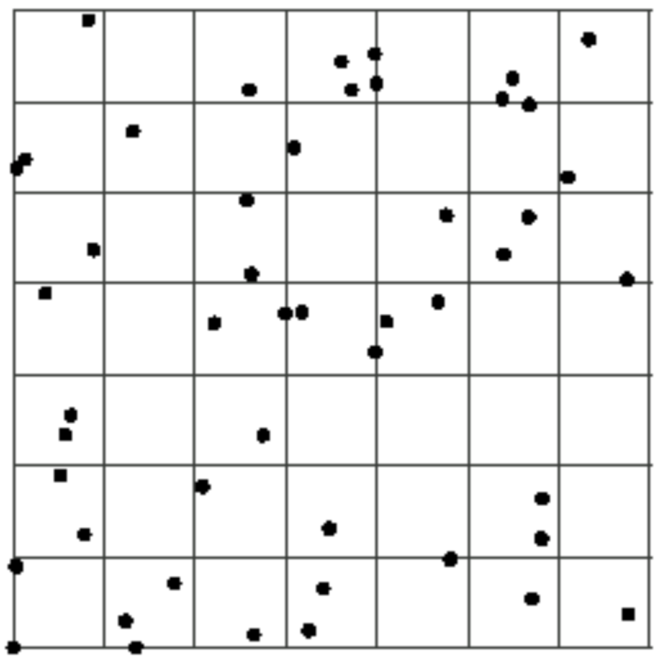
Stratified Sampling

- 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

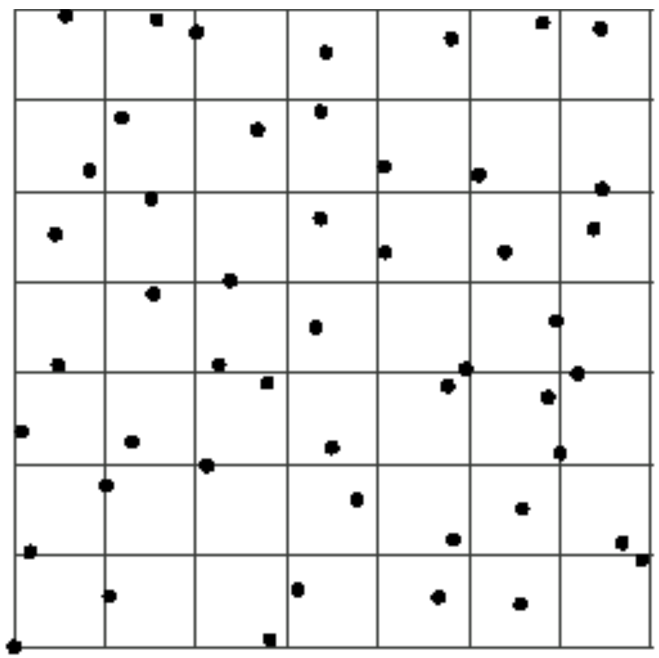
- Example: uniform jittering

Jitter

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



Random



Jittered

Texture Example

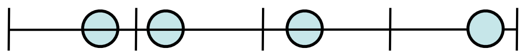
“ideal” – 256 samples/pixel

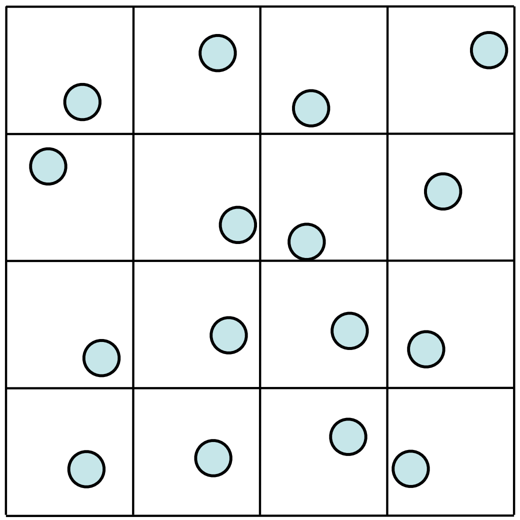
Jitter with 1 sample/pixel

1 sample/pixel uniform and unjittered

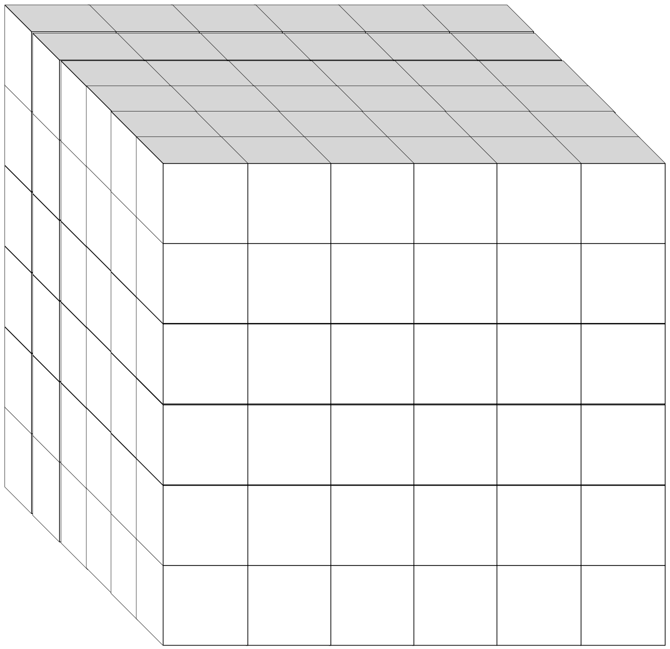
Jitter with 4 samples/pixel

Multiple Dimensions

- Too many samples
- 1D 
- 2D



3D

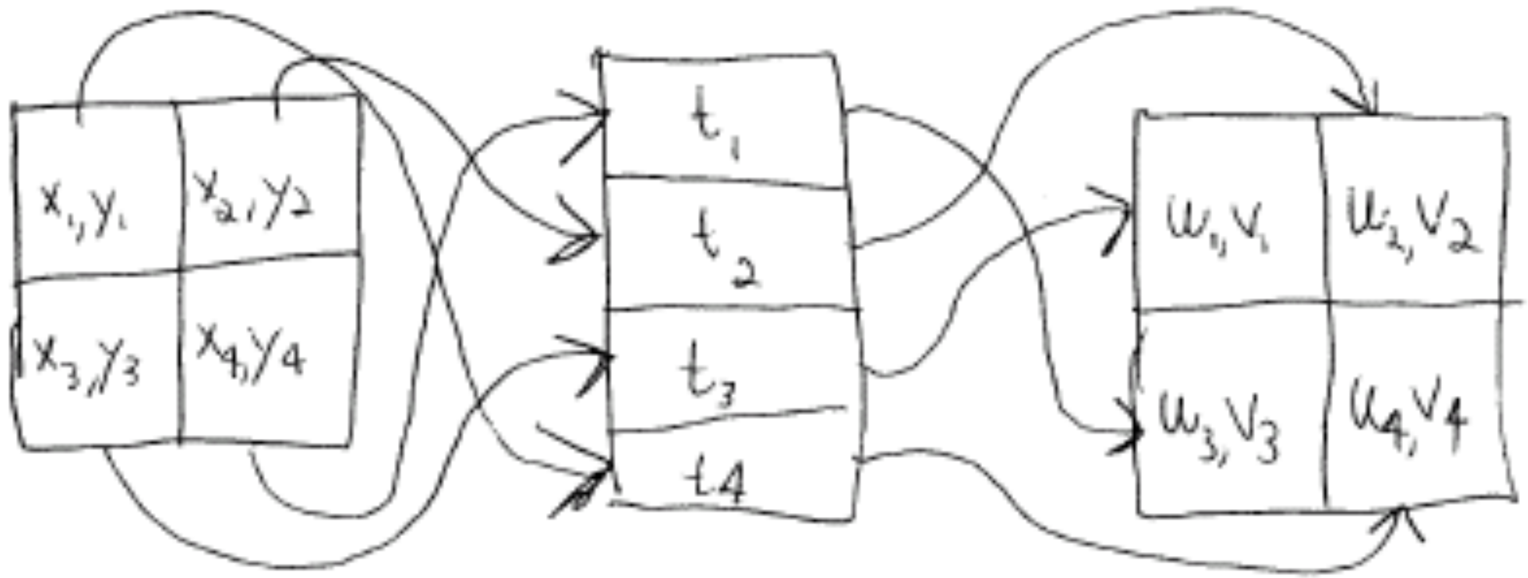


Jitter Problems

- 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

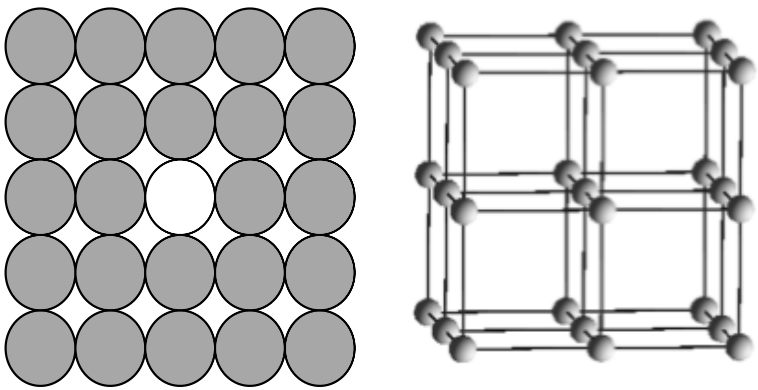
Multiple Dimensions

- 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

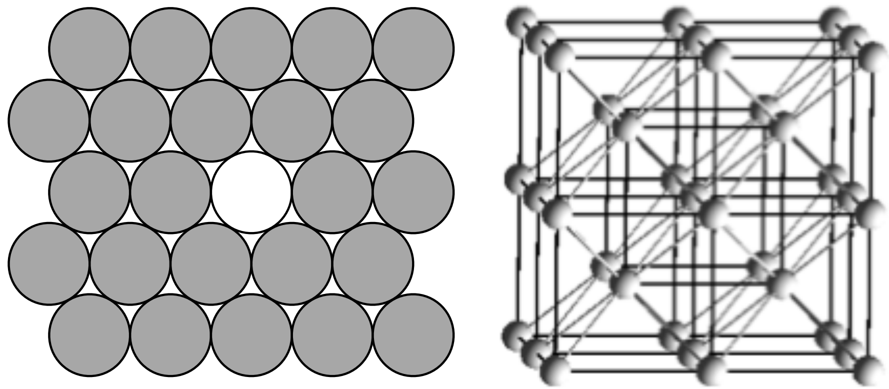


Aside: alternative sampling lattices

- 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



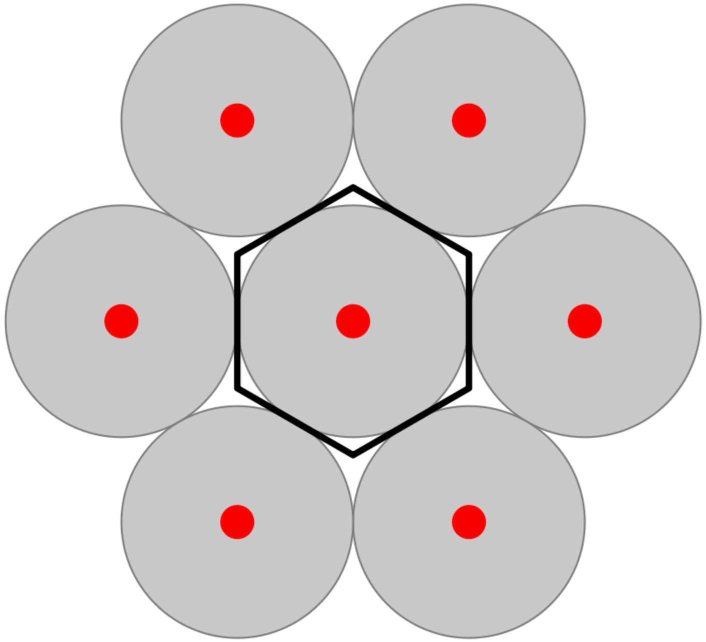
Cartesian

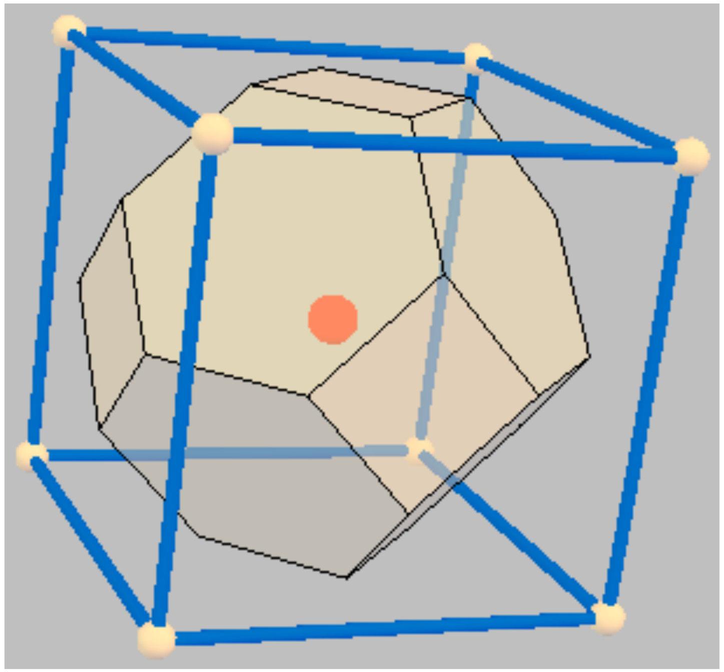


Hexagonal is optimal in 2D

Aside: optimal sampling lattices

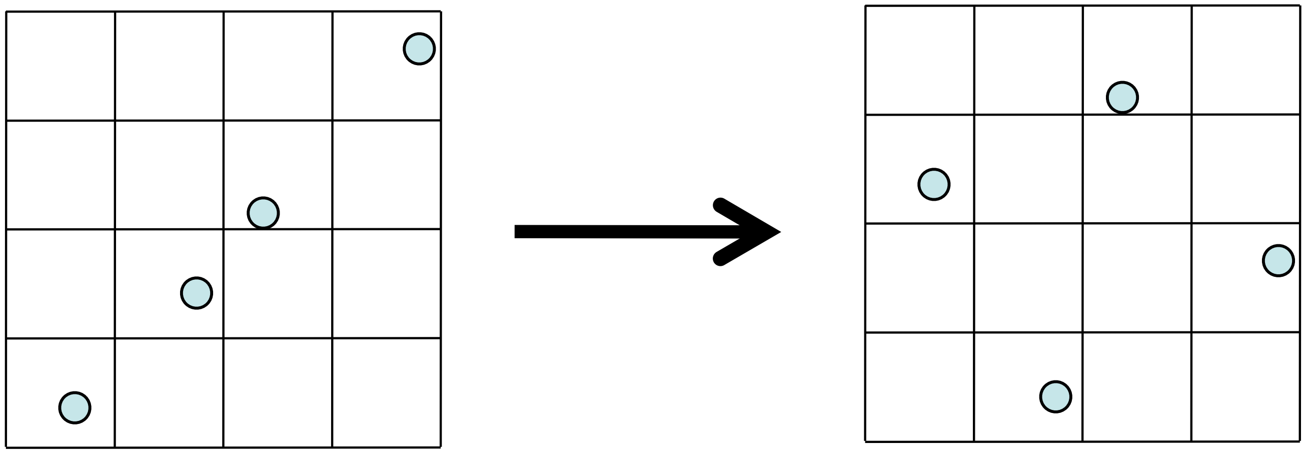
- We have to deal with different geometry
- 2D - hexagon
- 3D - truncated octahedron





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



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

$k = 3$

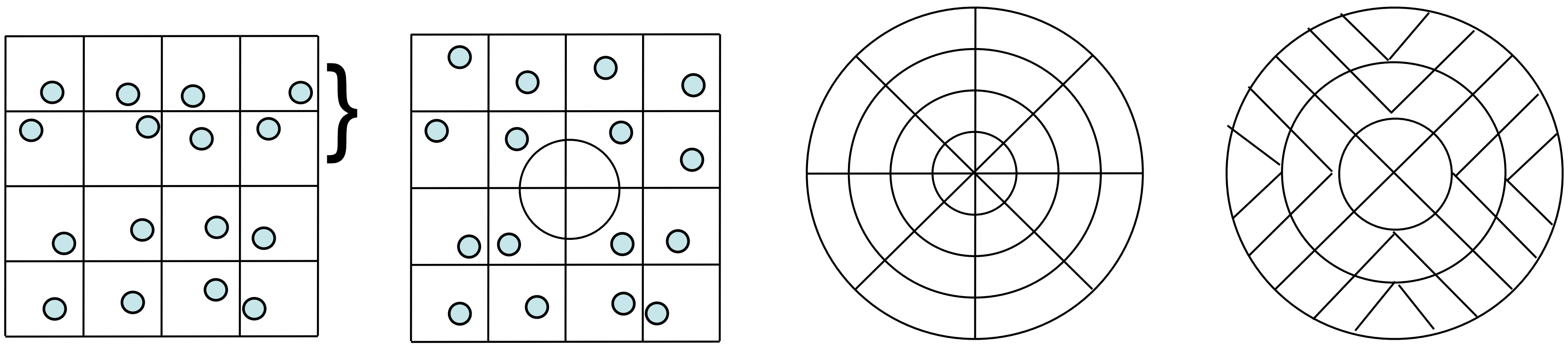
7	5	8	1	4	10	3	9	2	6
3	5	1	6	9	4	8	2	7	10
7	10	3	9	1	8	2	5	6	4

$s^3{}_1$ is from the 8-th cell from dimension 1

$n = 10$

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 ?

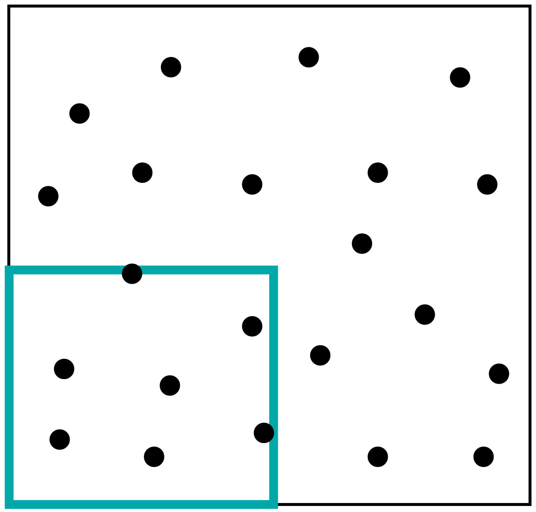
- How can we evaluate how well our samples are distributed in a more global manner?
  - No “holes”
  - No clumping
- Well distributed patterns are *low-discrepancy* – more evenly distributed
- Want to construct low-discrepancy sequence
- Most of these are deterministic!

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 $|7/22 - ¼| \leq D_{22}(B, P)$



Discrepancy

- Examples of sub-volume sets *B* of $[0,1]^d$:
  - All axis-aligned
  - All those sharing a corner at the origin (called *star discrepancy* $D_N^*(P)$)
- Asymptotically lowest discrepancy that has been obtained in *d* dimensions:

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

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:

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

What happens if B = all intervals?

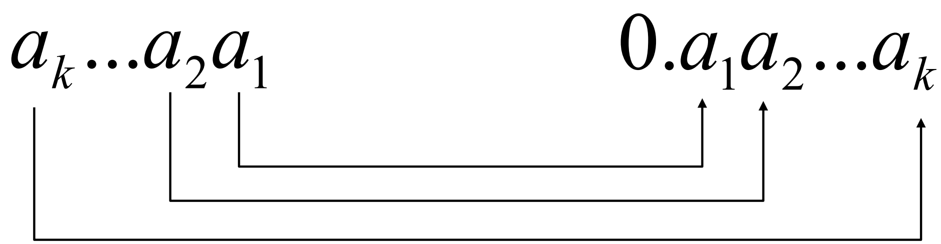
In general, $$D_N^*(x_1, \ldots, x_N) = \frac{1}{2N} + \max_{1 \le i \le N} \left| x_i - \frac{2i - 1}{2N} \right|$$

Pseudo-Random Sequences

- Radical inverse
  - Building block for high dimensional sequences
  - "inverts" an integer given in base b

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

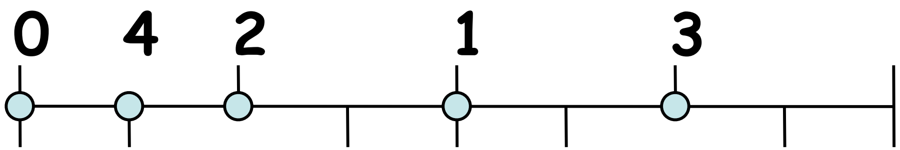
$$\Phi_b(n) = 0.a_1 a_2 \ldots a_k = a_1 b^{-1} + a_2 b^{-2} + a_3 b^{-3} + \ldots$$



Van Der Corput Sequence

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

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



i	binary form of i	radical inverse	x_i
0	0	0.0	0
1	1	0.1	0.5
2	10	0.01	0.25
3	11	0.11	0.75
4	100	0.001	0.125
5	101	0.101	0.625
6	110	0.011	0.375

Halton

- Use a *prime number basis* for each dimension
- Achieves best possible discrepancy asymptotically

$$x_i = (\Phi_2(i), \Phi_3(i), \Phi_5(i), \ldots, \Phi_{p_d}(i))$$

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

- Can be used if N, the number of samples, is not known in advance — all prefixes of a Halton sequence are well distributed

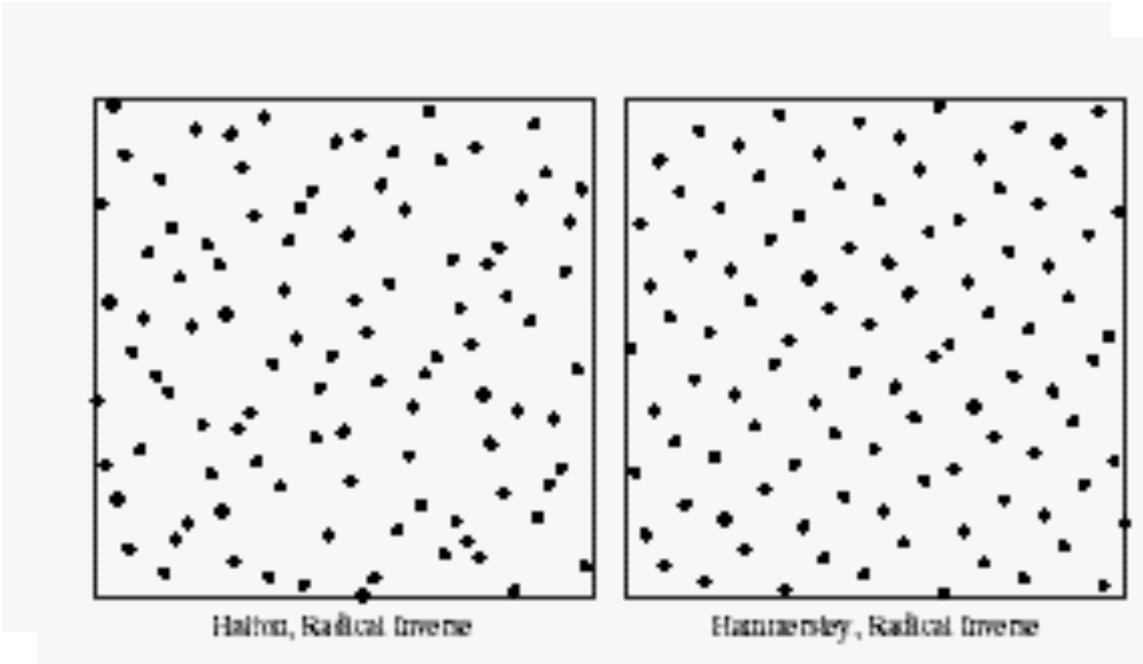
Hammersley Sequences

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

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

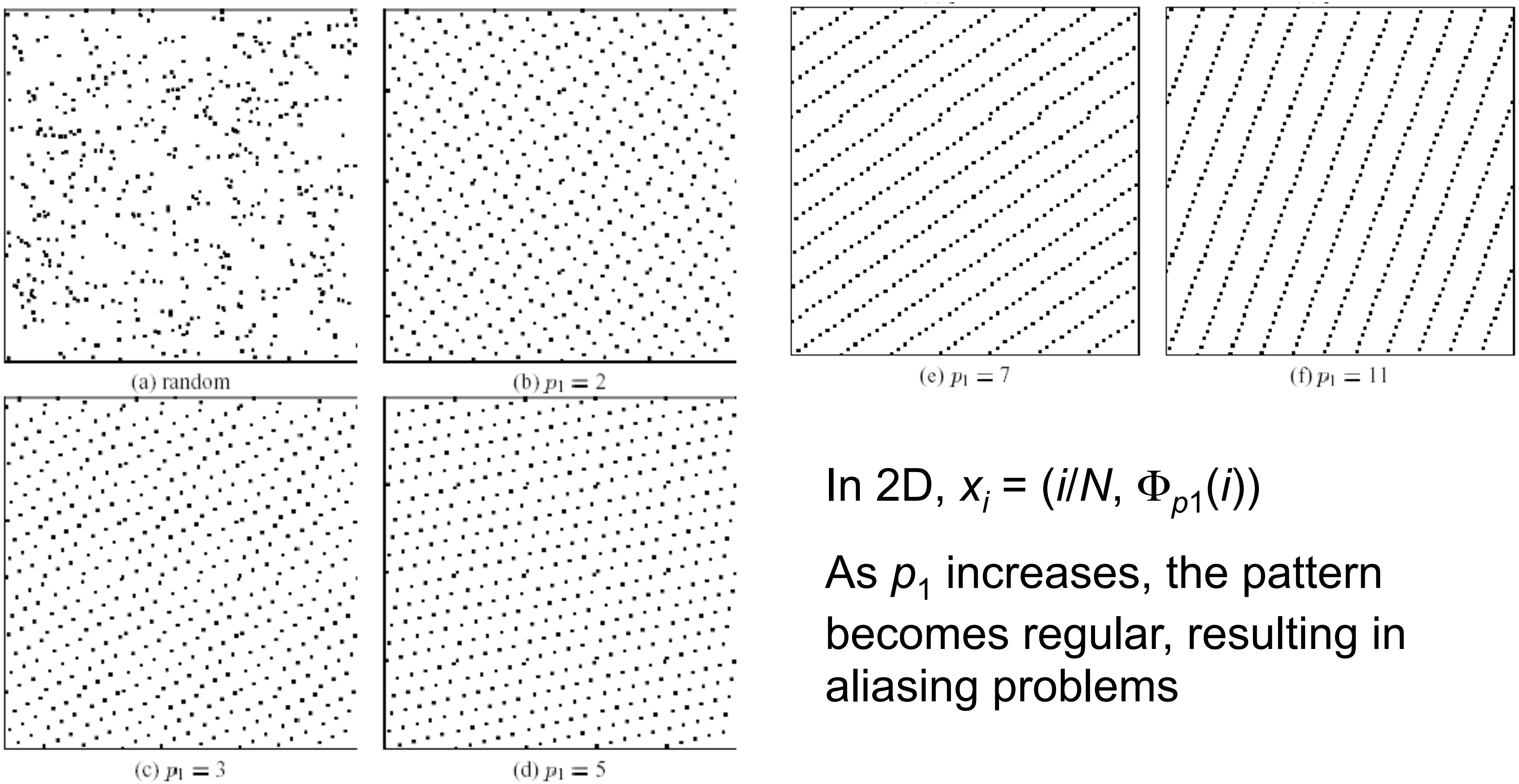
Prime numbers

Halton vs. Hammersley



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

Hammersley Sequences

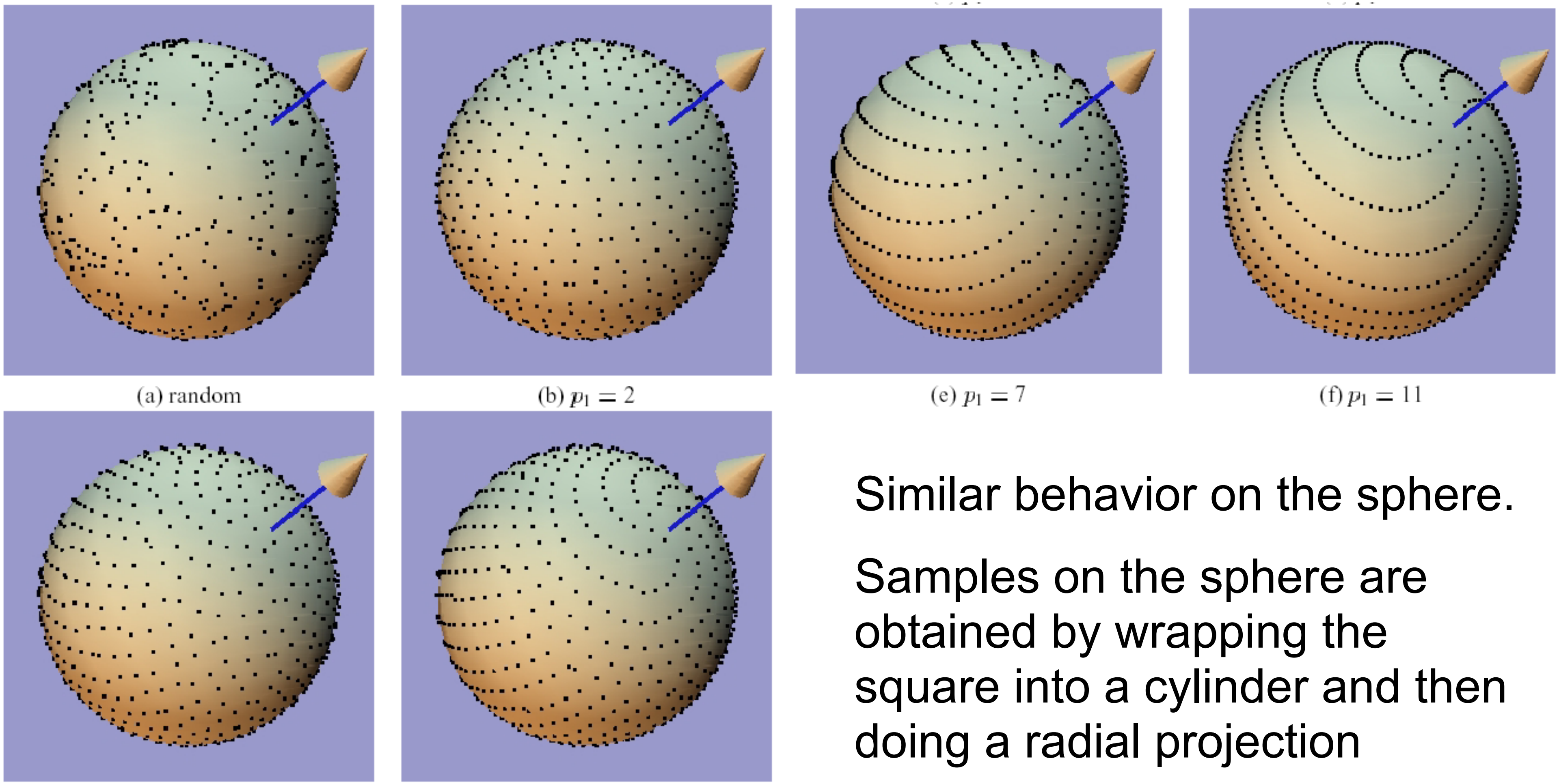


(a) random (b) $p_1 = 2$ (c) $p_1 = 3$ (d) $p_1 = 5$ (e) $p_1 = 7$ (f) $p_1 = 11$

In 2D, $x_i = (i/N, \Phi_{p1}(i))$

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

Hammersley Sequences



(a) random (b) $p_1 = 2$ (e) $p_1 = 7$ (f) $p_1 = 11$

(c) $p_1 = 3$ (d) $p_1 = 5$

Similar behavior on the sphere.

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

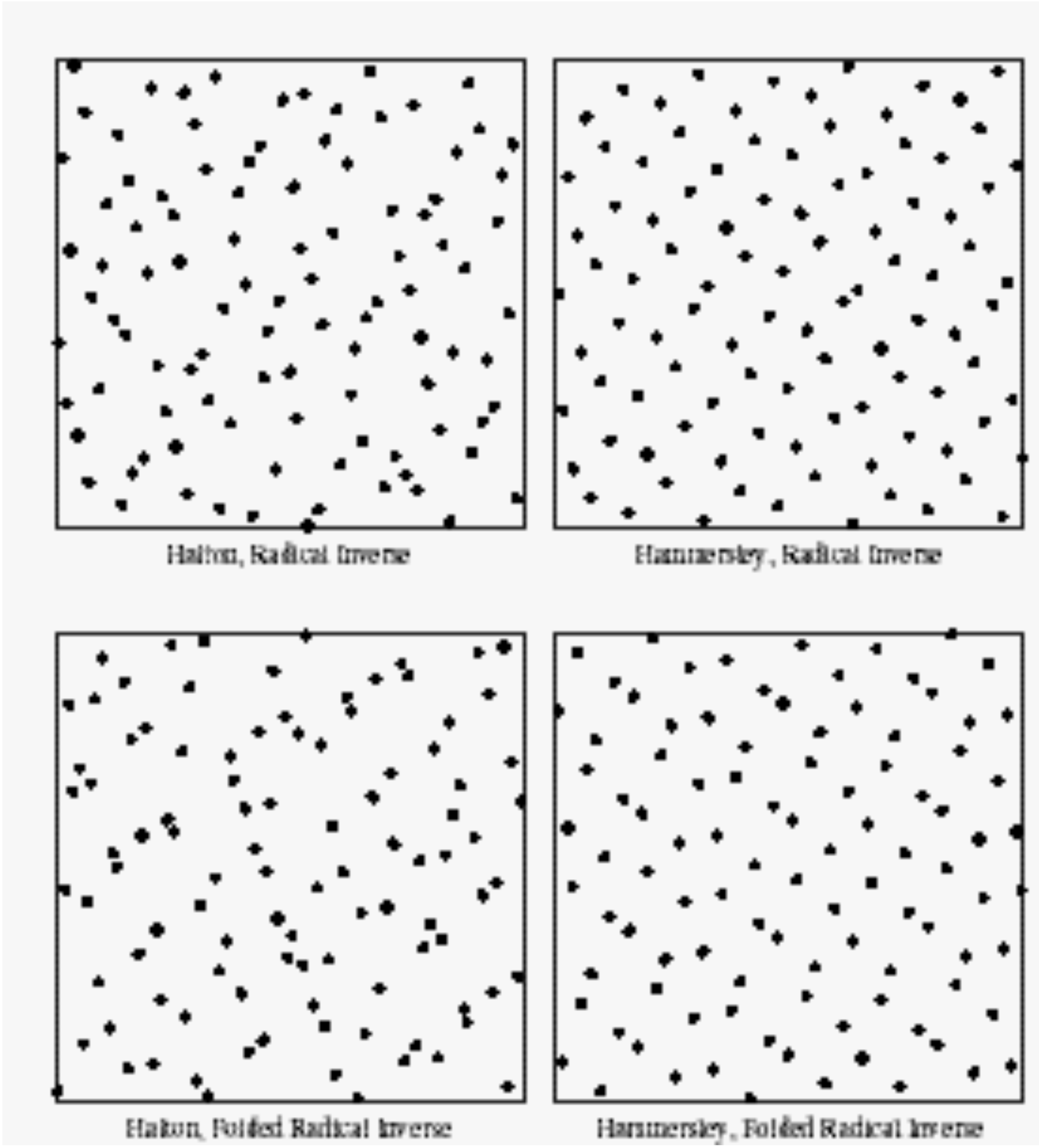
Folded Radical Inverse

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

$$\Phi_b(n) = \sum_{i=1}^{\infty} a_i \frac{1}{b^i}$$

$$\Phi_b(n) = \sum_{i=1}^{\infty} ((a_i + i - 1) \bmod b) \frac{1}{b^i}$$

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

$$E = \prod_{i=1}^{d} \left[a_i b^{-c_i}, (a_i + 1) b^{-c_i} \right) \text{ where } \sum_{i=1}^{d} c_i = m - t$$

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

Optimal in absolute terms

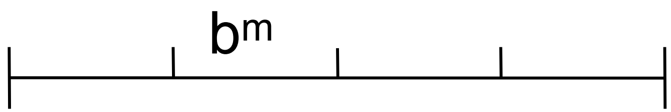
Reference: www.mathdirect.com/products/qrn/resources/Links/QRDemonstration_lnk_4.html

(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

$$\{x_i \mid kb^m \leq i < (k+1)b^m\}$$

  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.

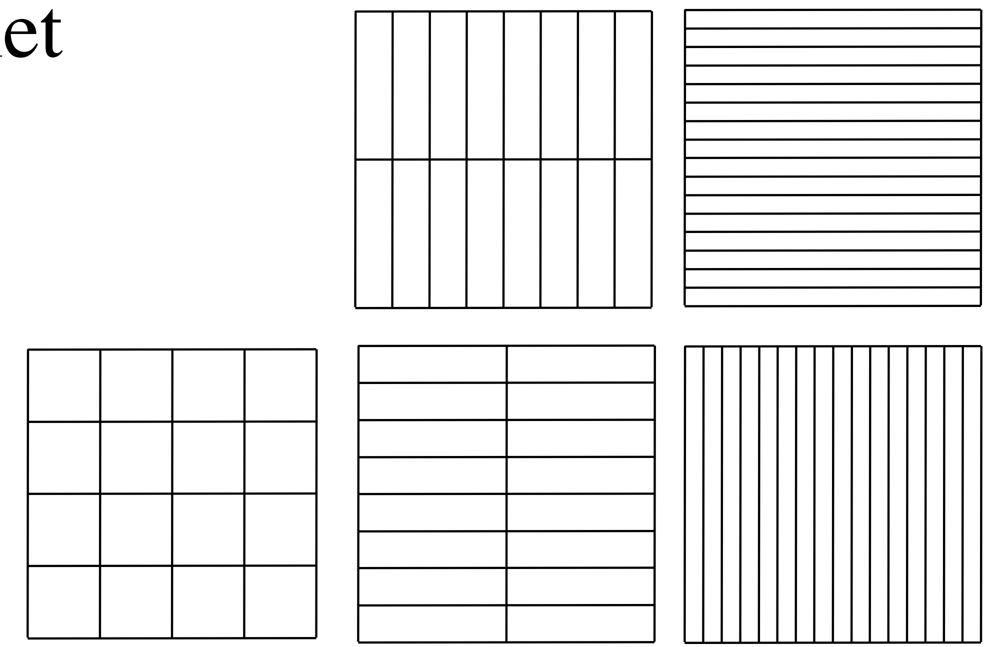
Reference: www.mathdirect.com/products/qrn/resources/Links/QRDemonstration_lnk_4.html

(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 rectangle

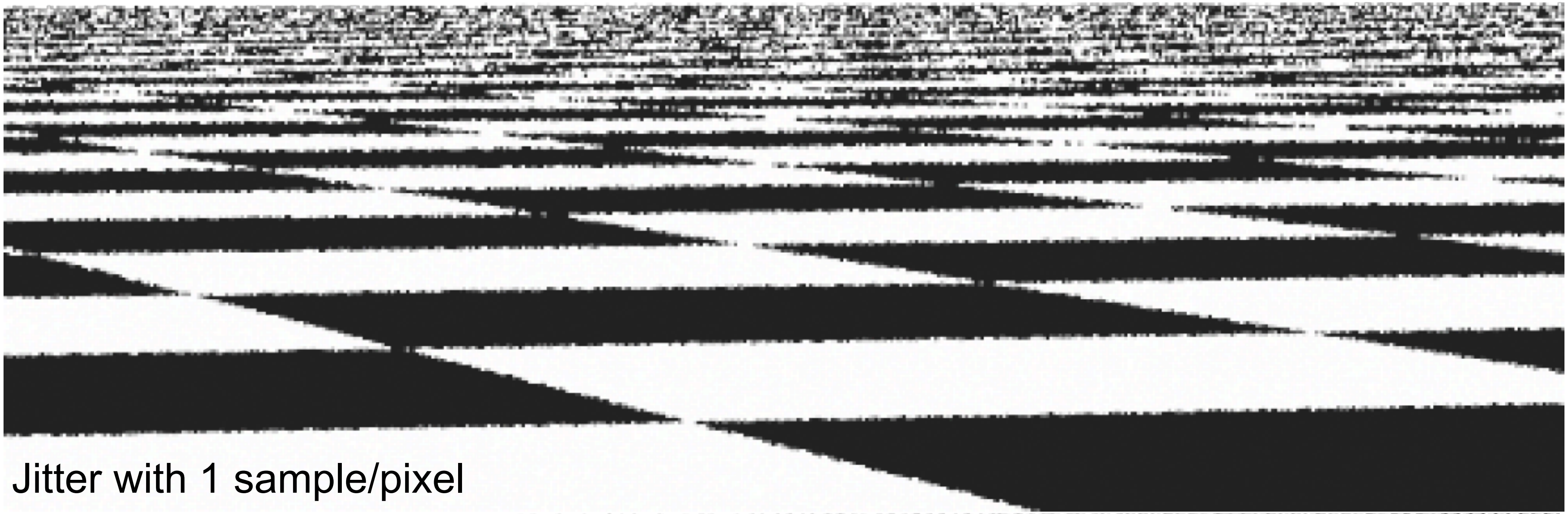


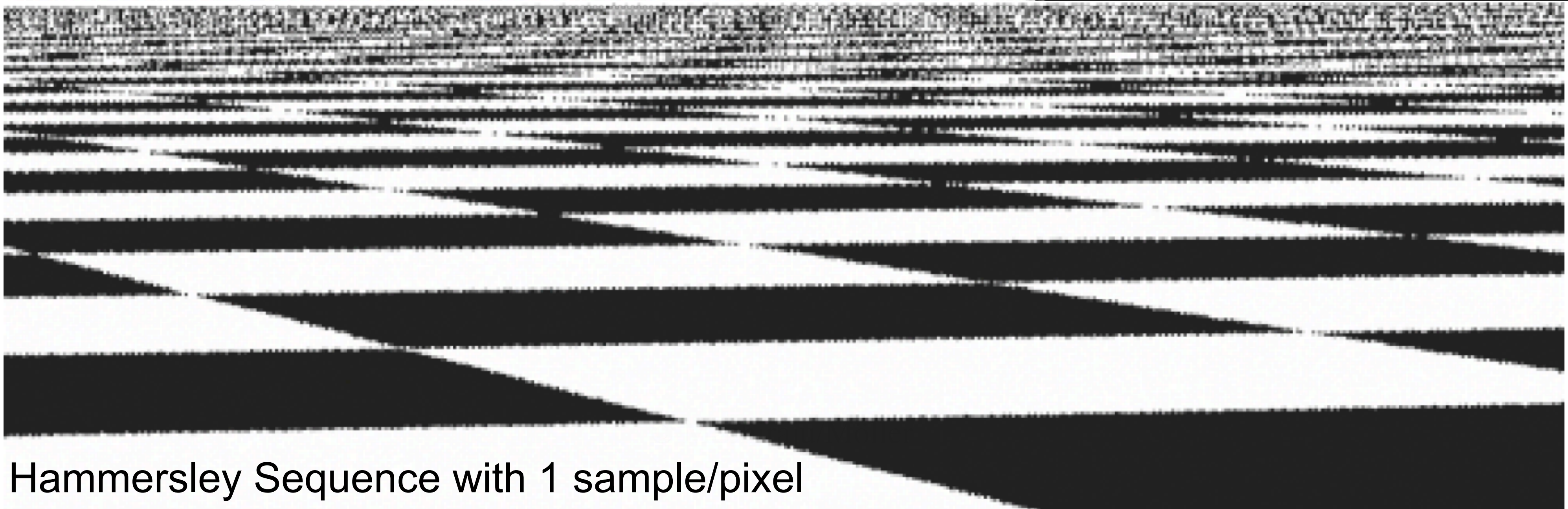
N-rook property

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

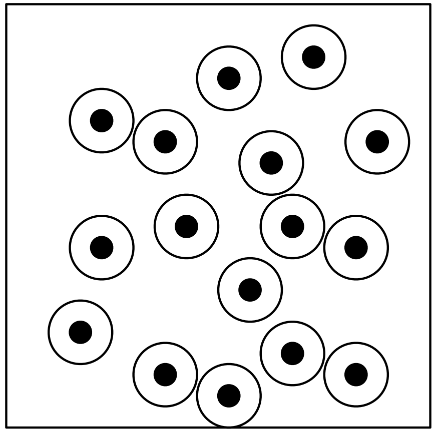

Jitter with 1 sample/pixel


Hammersley Sequence with 1 sample/pixel

Best-Candidate Sampling

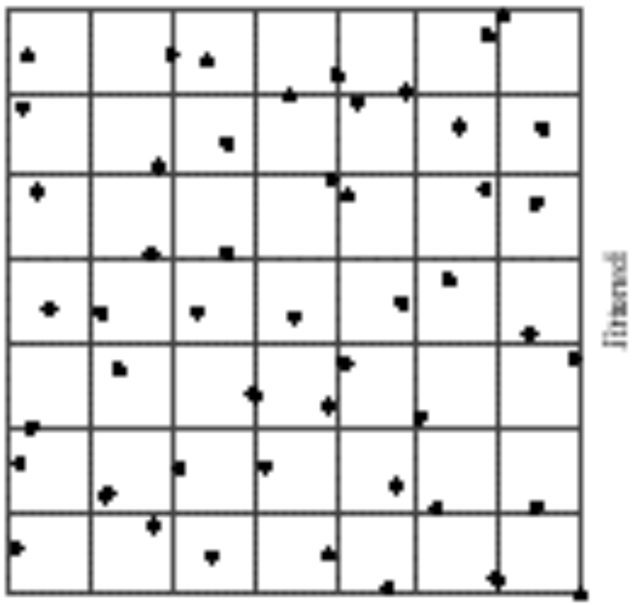
- Jittered stratification
  - Randomness (inefficient)
  - Clustering problems between adjacent strata
  - Undersampling (“holes”)
- Low Discrepancy Sequences
  - No explicit preventing two samples from coming to close
- “Ideal”: Poisson disk distribution
  - too computationally expensive
- Best Sampling - approximation to Poisson disk –a form of *farthest point sampling*

Poisson Disk

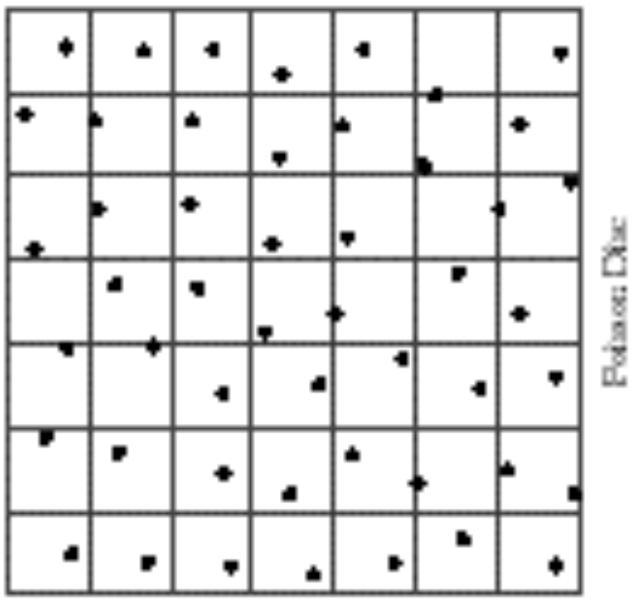


- Comes from structure of eye – rods and cones
- Dart Throwing
- No two points are closer than a threshold
- Very expensive
- Compromise – Best Candidate Sampling
  - Every new sample is to be farthest from previous samples amongst a set of randomly chosen candidates
  - Compute pattern which is reused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

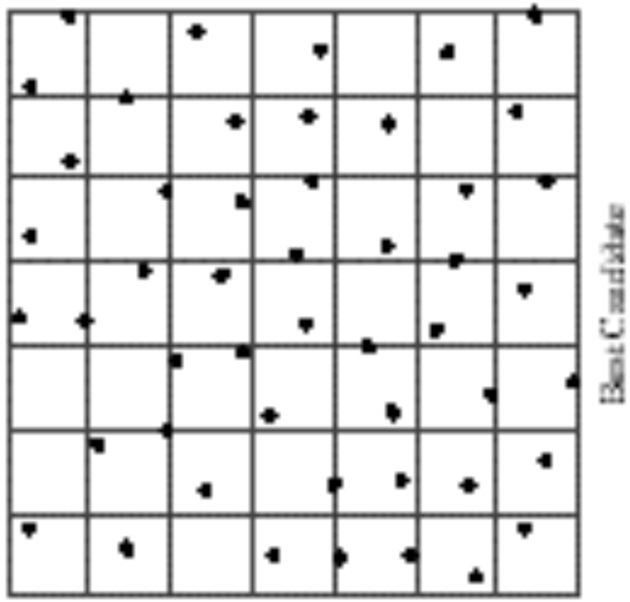
Best-Candidate Sampling



Jittered



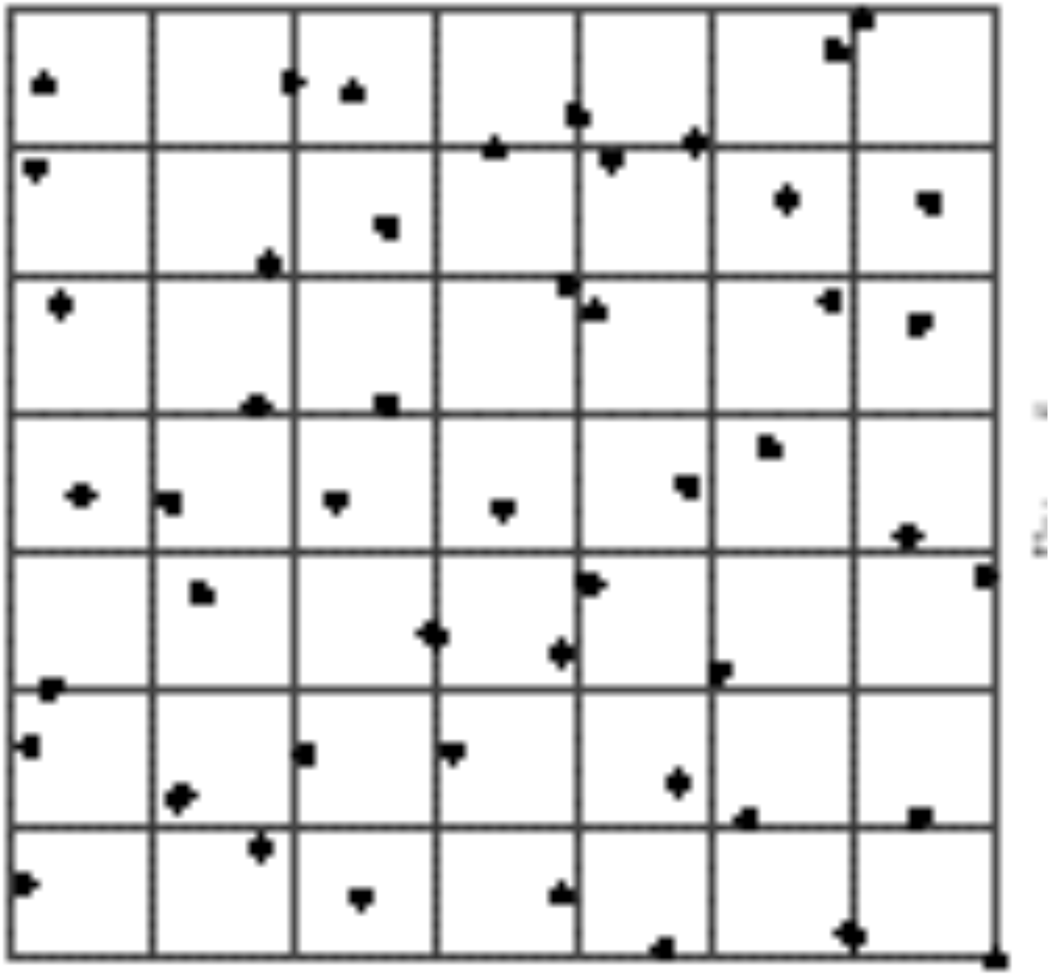
Poisson Disk



Best Candidate

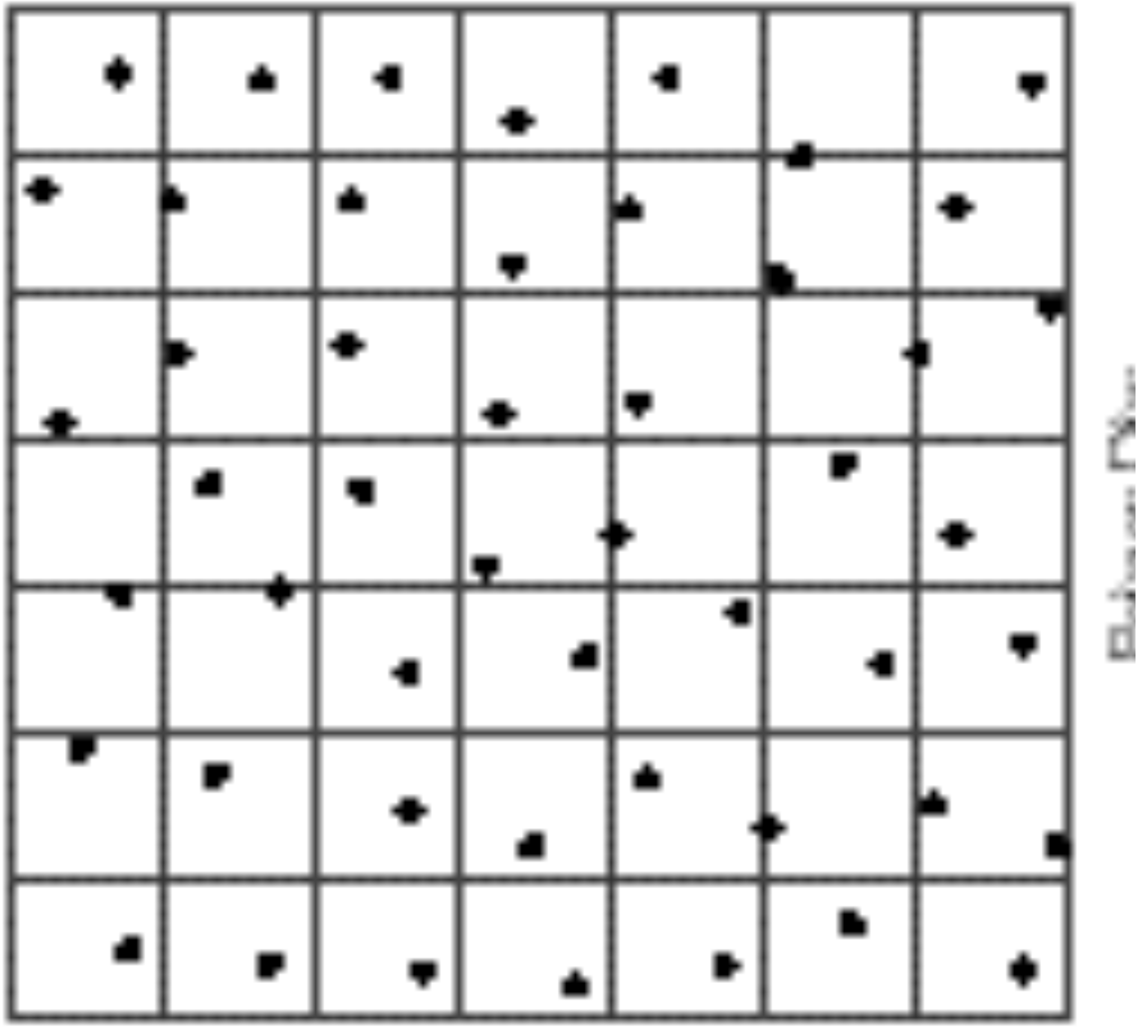
Best-Candidate Sampling

Jittered



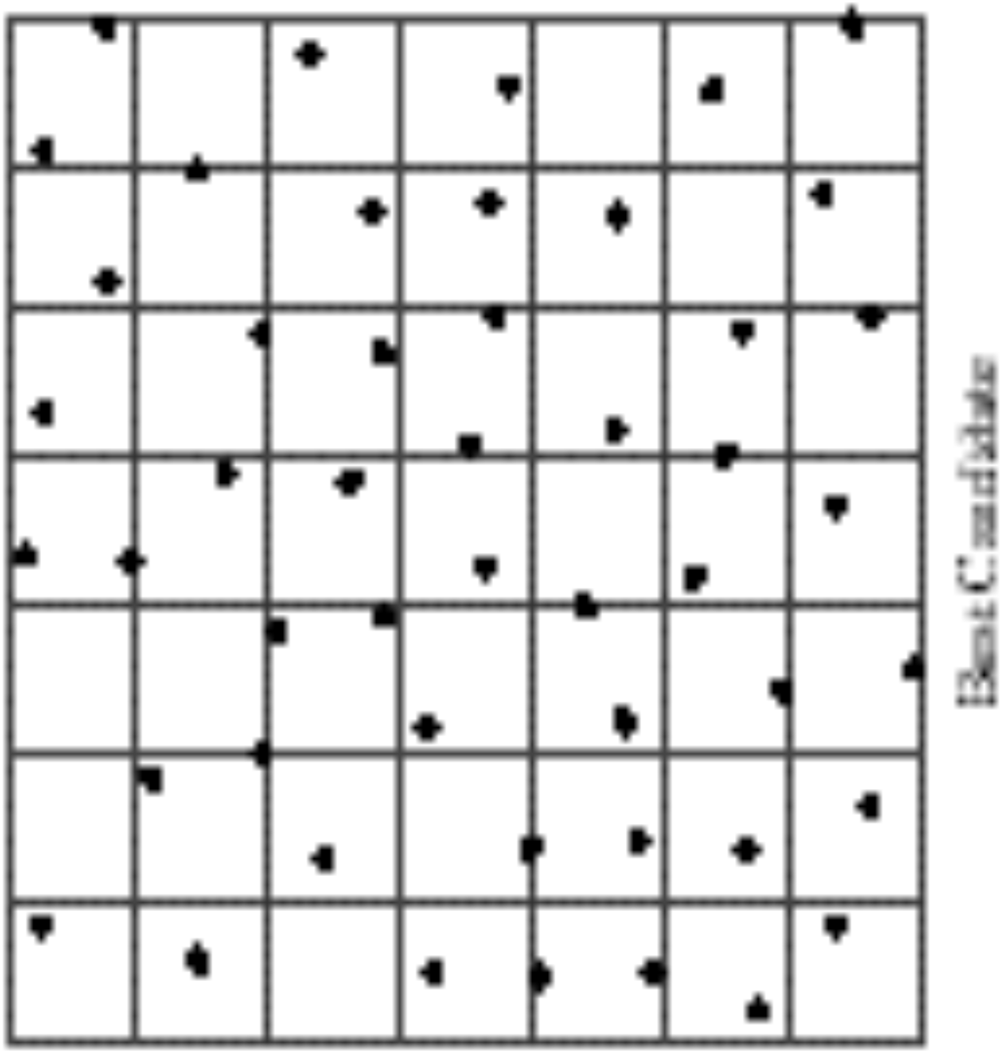
Best-Candidate Sampling

Poisson Disk



Best-Candidate Sampling

Best Candidate



Dart throwing

$i \leftarrow 0$
`while` $i < N$
  $x_i \leftarrow$ `unit()`
  $y_i \leftarrow$ `unit()`
  reject $\leftarrow$ `false`
	Throw a dart.

`for` $k \leftarrow 0$ `to` $i-1$ $d \leftarrow (x_i - x_k)^2 + (y_i - y_k)^2$	*Check the distance to all other samples.*

`if` $d < (2r_p)^2$ `then`
      reject $\leftarrow$ `true`
      `break`
      `endif`
	This one is too close—forget it.

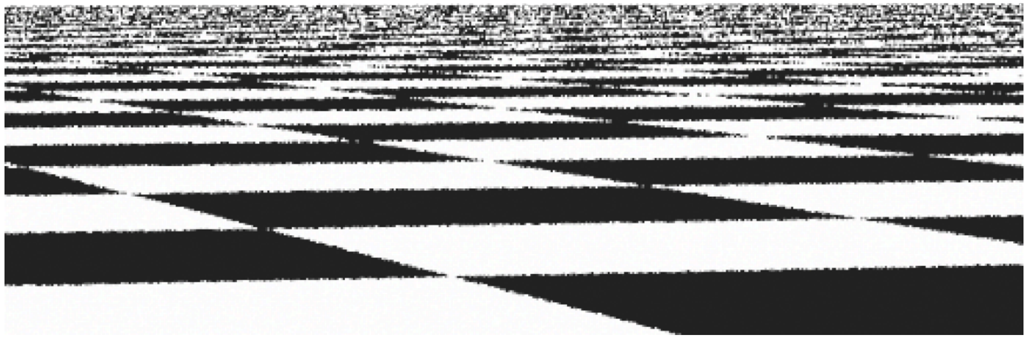
`endfor`	

`if not` *reject* `then`
    $i \leftarrow i + 1$
    `endif`
	Append this one to the pattern.

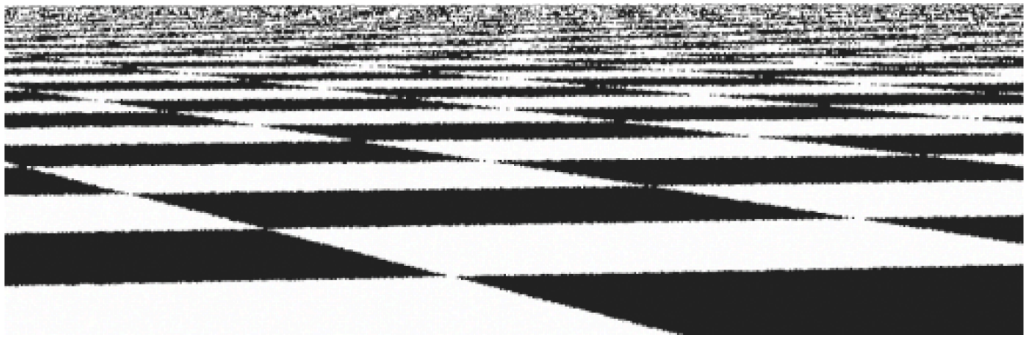
`endwhile`	

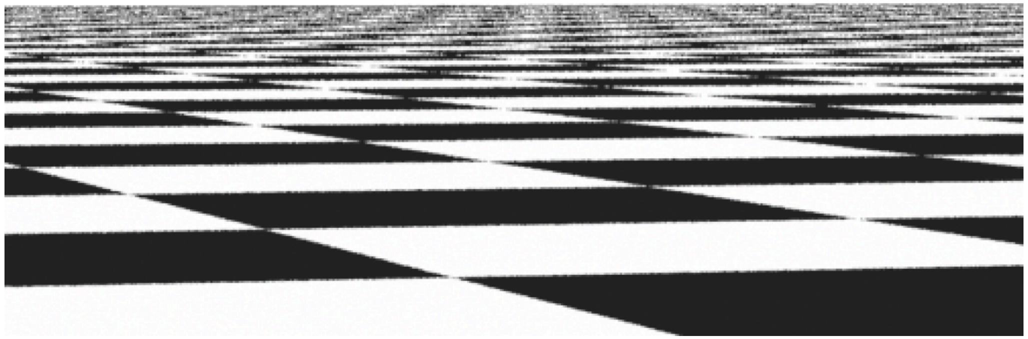
Texture

Jitter with 1 sample/pixel



Best Candidate with 1 sample/pixel





Jitter with 4 sample/pixel

Best Candidate with 4 sample/pixel

Next

- Rendering Equation
- Probability Theory
- Monte Carlo Techniques