Signals and Sampling

Chapter 7 of “Physically Based Rendering” by Pharr&Humphreys

Chapter 7

Additional Reading

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

- Real World - continuous
- Digital (Computer) world - discrete
- Typically we have to either:
  - create discrete data from continuous or (e.g. rendering/ray-tracing, illumination models, morphing)
  - manipulate discrete data (textures, surface description, image processing,tone mapping)

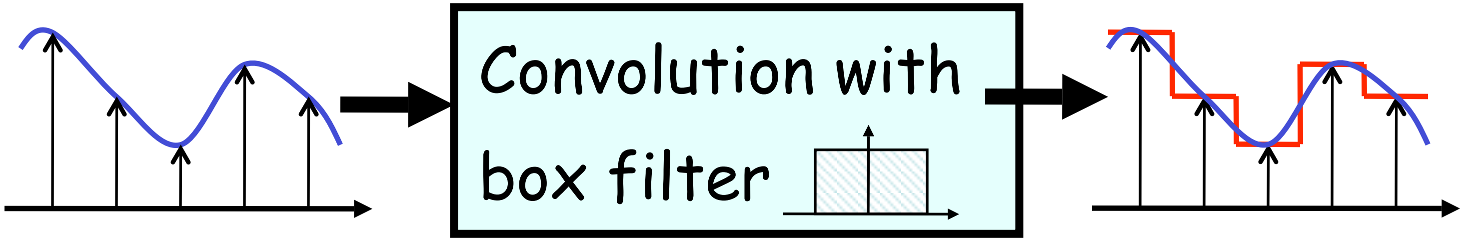
Motivation

- Artifacts occurring in sampling - aliasing:
  - Jaggies
  - Moire
  - Flickering small objects
  - Sparkling highlights
  - Temporal strobing
- Preventing these artifacts - Antialiasing

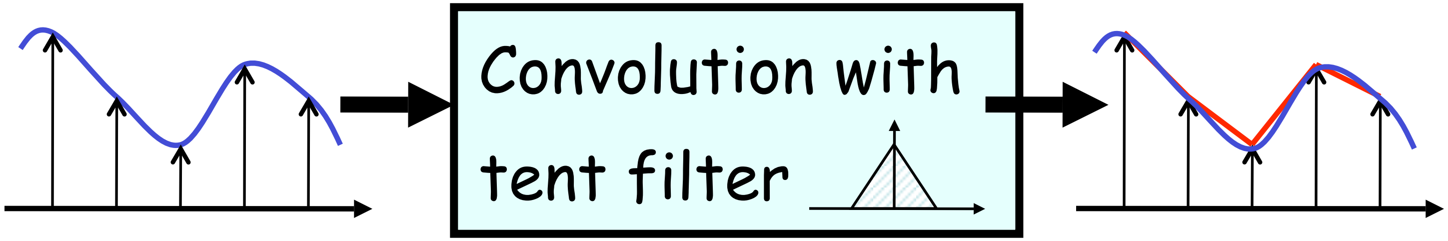
Motivation

Engineering approach:

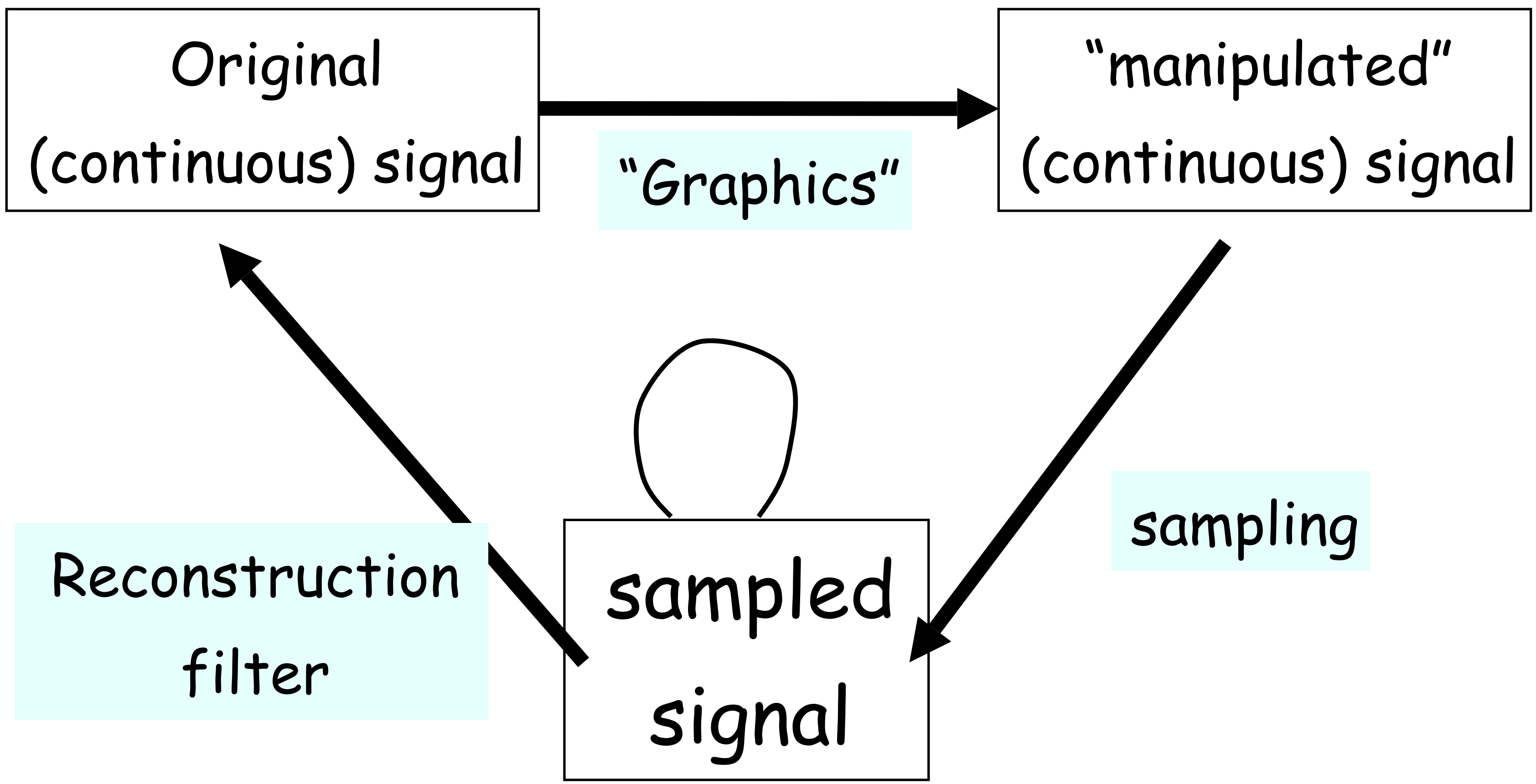
nearest neighbor:



linear filter:



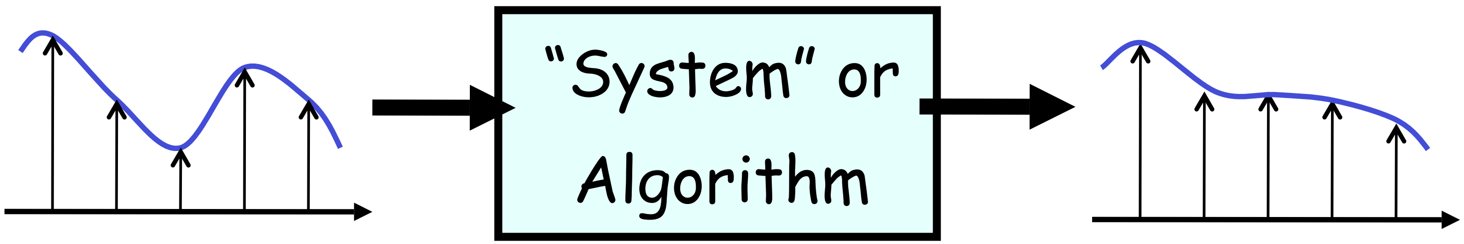
Motivation- Graphics



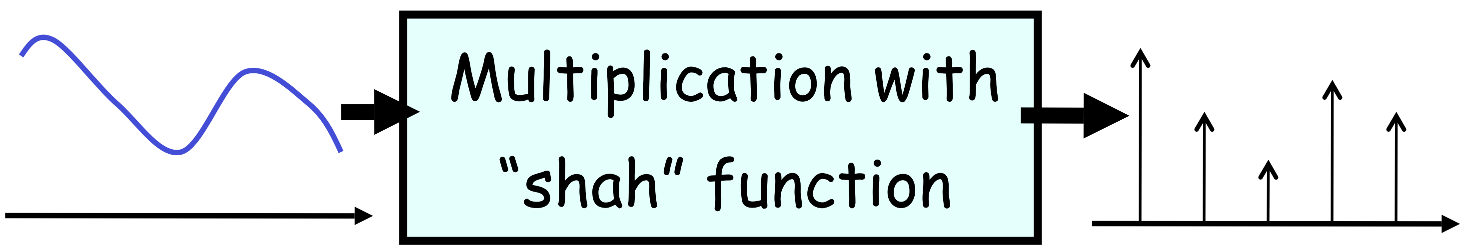
Motivation

Engineering approach:

- black-box

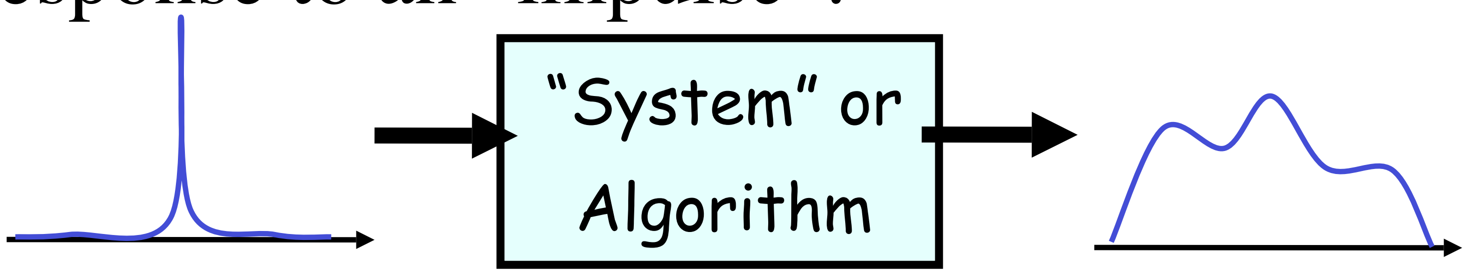


- discretization:



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

  $$\delta[0] = 1$$

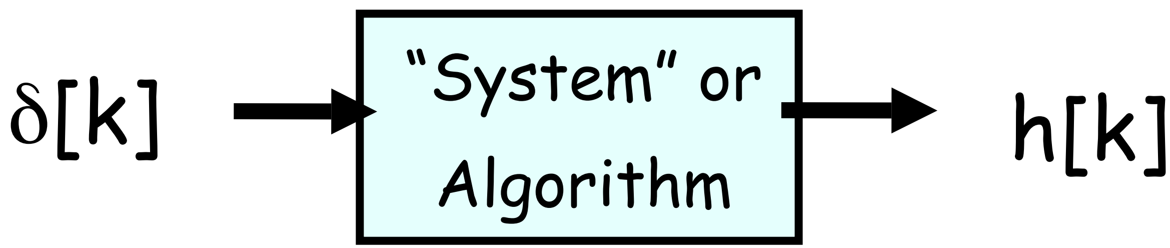
- Finite Impulse Response (FIR) vs.
- Infinite Impulse Response (IIR)

Convolution (3)

- An arbitrary 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]:

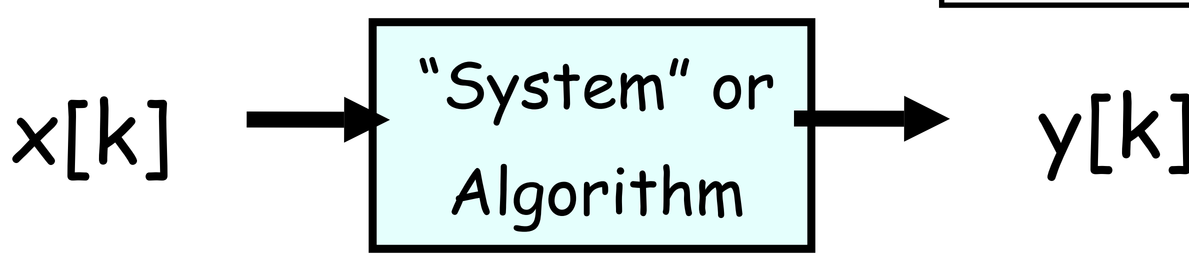


Convolution (4)

- for a time-invariant system h[k-n] would be the impulse response to a delayed impulse d[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.



Fourier Transforms

- Let's look at a special input sequence:

$$x[k] = e^{i\omega k}$$

- then:

$$y[k] = \sum_{n=-N}^{N} e^{i\omega(k-n)} h[n]$$

$$= e^{i\omega k} \sum_{n=-N}^{N} e^{-i\omega n} h[n]$$

$$= H(\omega) e^{i\omega k}$$

Fourier Transforms (2)

- Hence $e^{i\omega k}$ is an eigen-function 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 $< H(\omega)$
  - magnitude response $|H(\omega)|$

Properties

- Linear $$af(x) + bg(x) \Leftrightarrow aF(\omega) + bG(\omega)$$
- scaling $$f(ax) \Leftrightarrow 1/a\, F(\omega/a)$$
- convolution $$f(x) \otimes g(x) \Leftrightarrow F(\omega) \times G(\omega)$$
- 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 $$f(x - \tau) \Leftrightarrow e^{-i\tau} F(\omega)$$

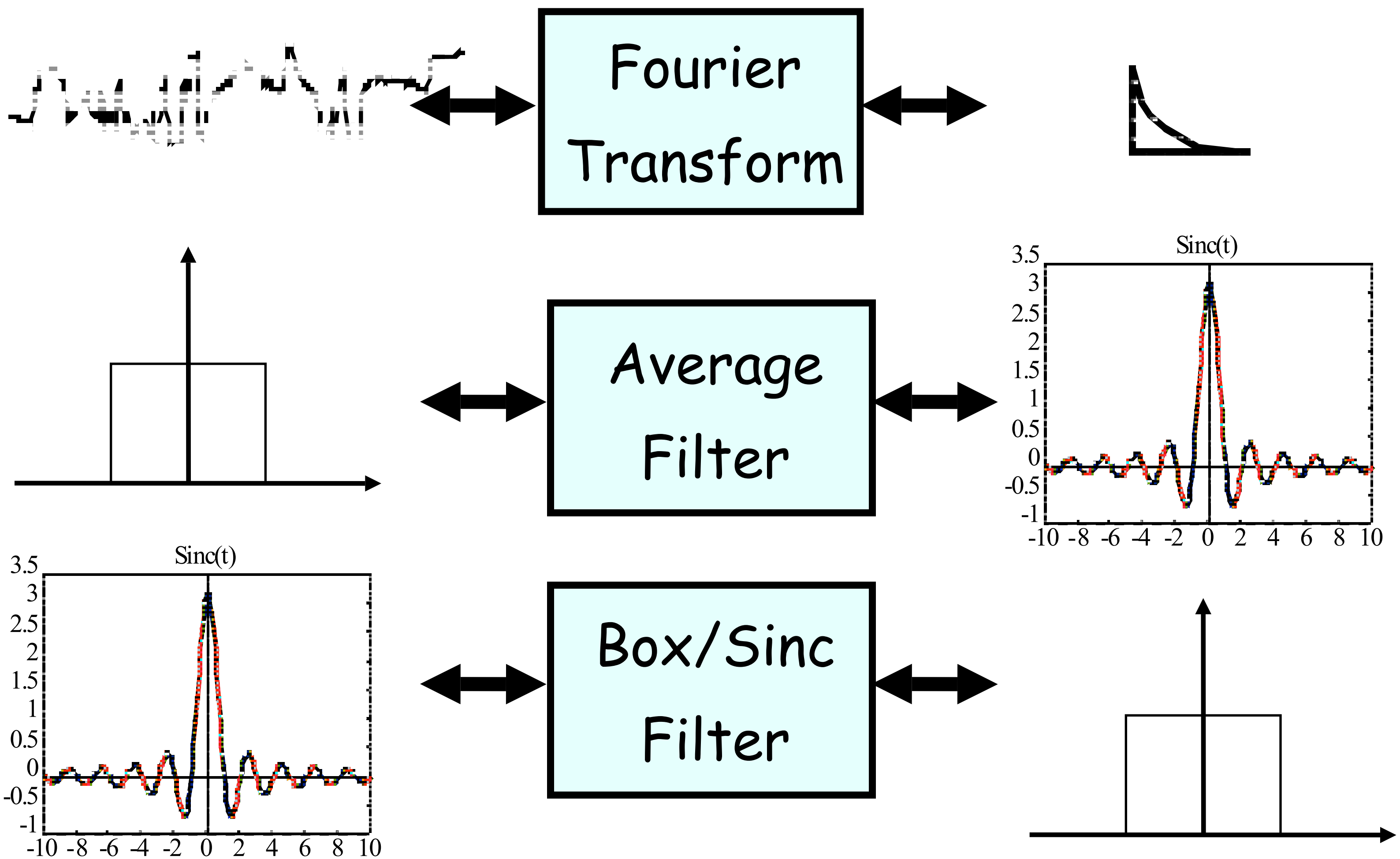
Properties (2)

- Parseval's Theorem

$$\int_{-\infty}^{\infty} f^2(x)dx \Leftrightarrow \int_{-\infty}^{\infty} F^2(\omega)d\omega$$

- preserves "Energy" - overall signal content

Transforms Pairs

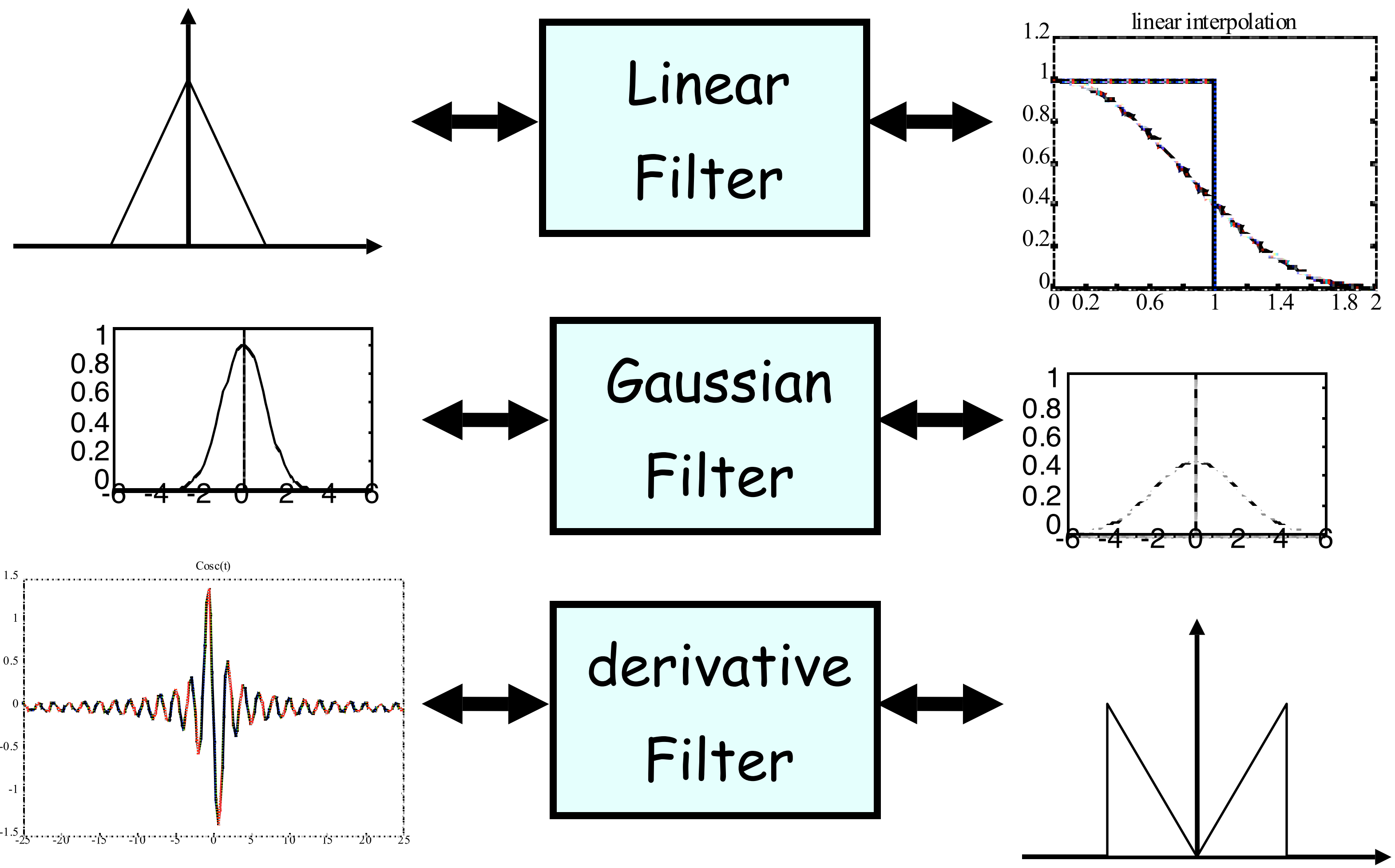


Transform Pairs - Shah

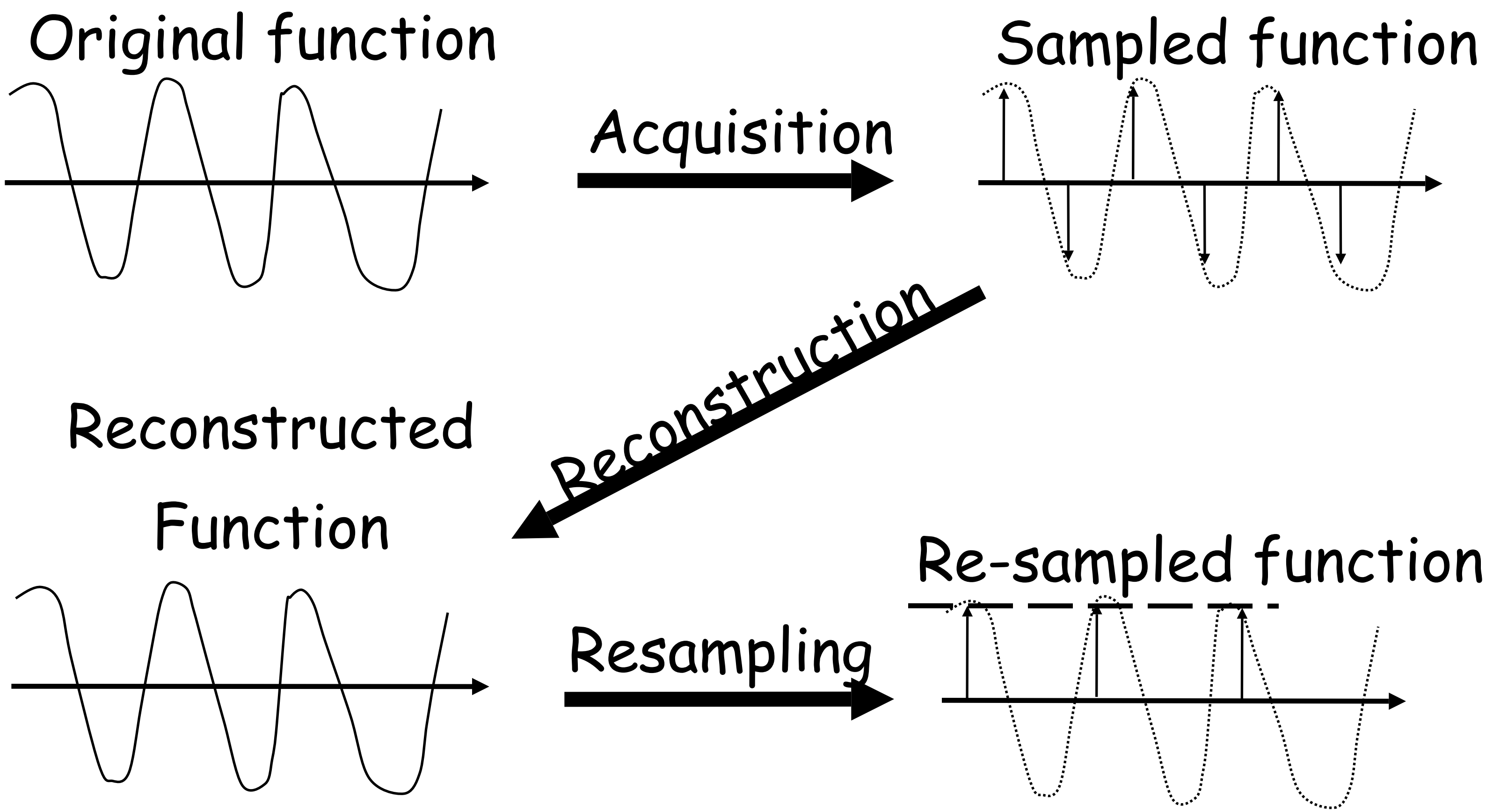
- Sampling = Multiplication with a Shah function:

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

Transforms Pairs (2)

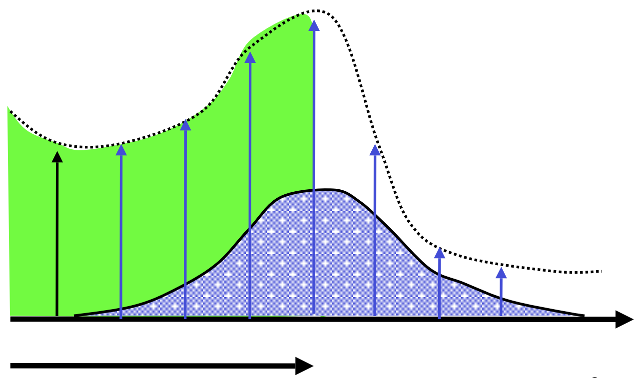


General Process

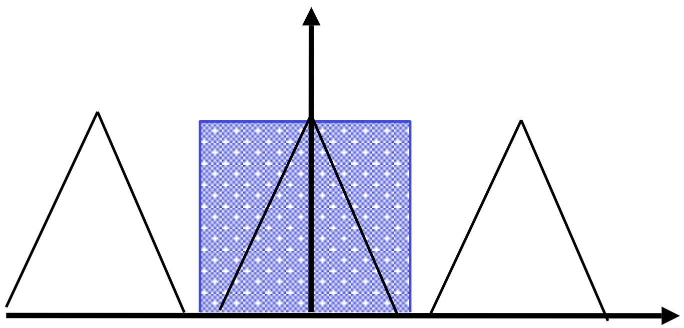


How? - Reconstruction

Spatial Domain:



Frequency Domain:



Mathematically:

- Convolution: f(x)*h(x)

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

Evaluated at discrete points (sum)

- Multiplication:

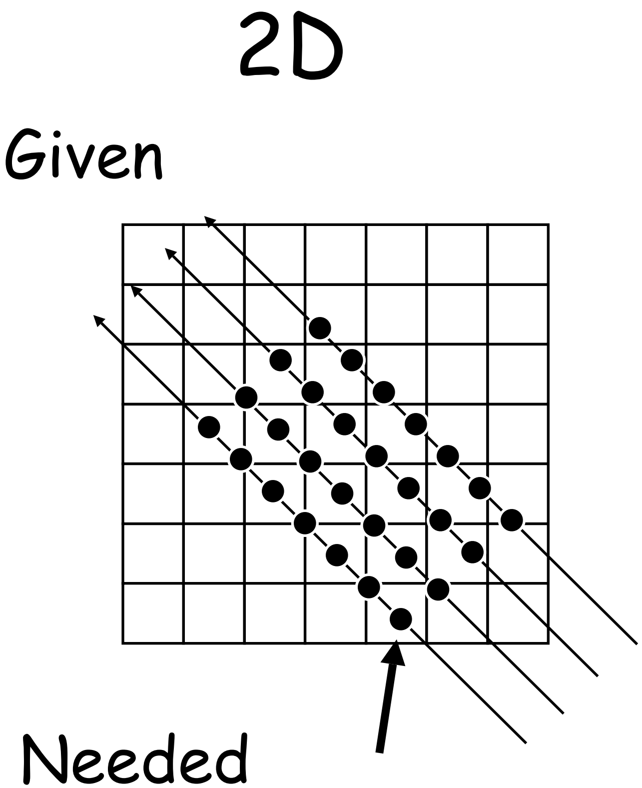
$$F(\omega) \times H(\omega)$$

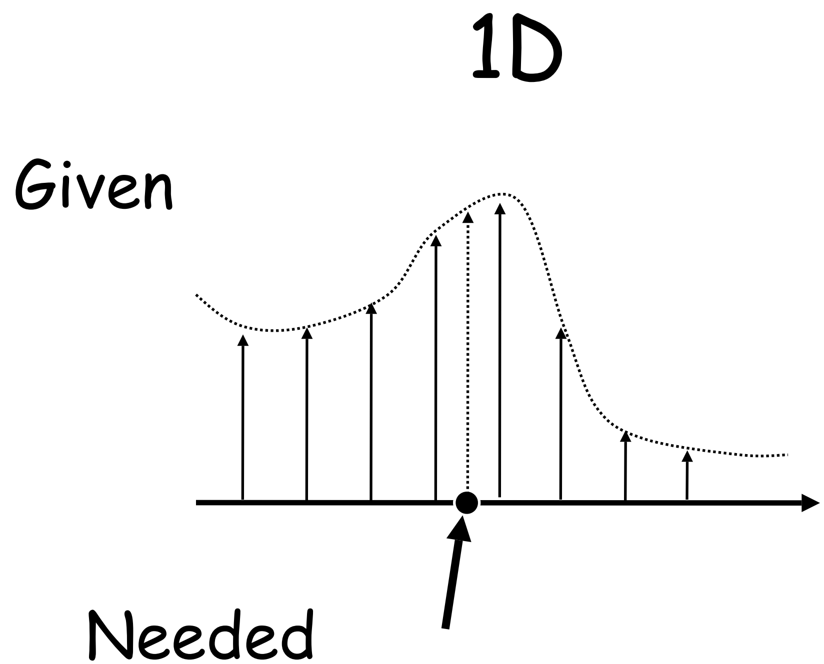
online demo

Sampling Theorem

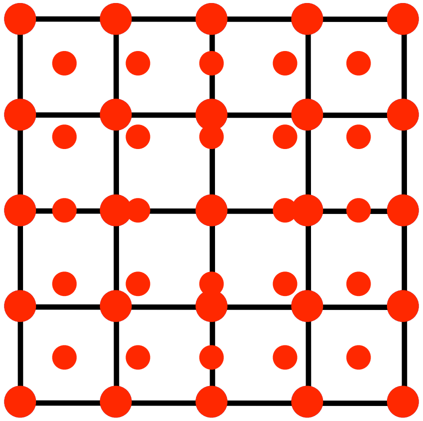
- A signal can be reconstructed from its samples without loss of information if the original signal has no frequencies above 1/2 of the sampling frequency
- For a given <u>bandlimited</u> function, the rate at which it must be sampled is called the **Nyquist frequency**

Example





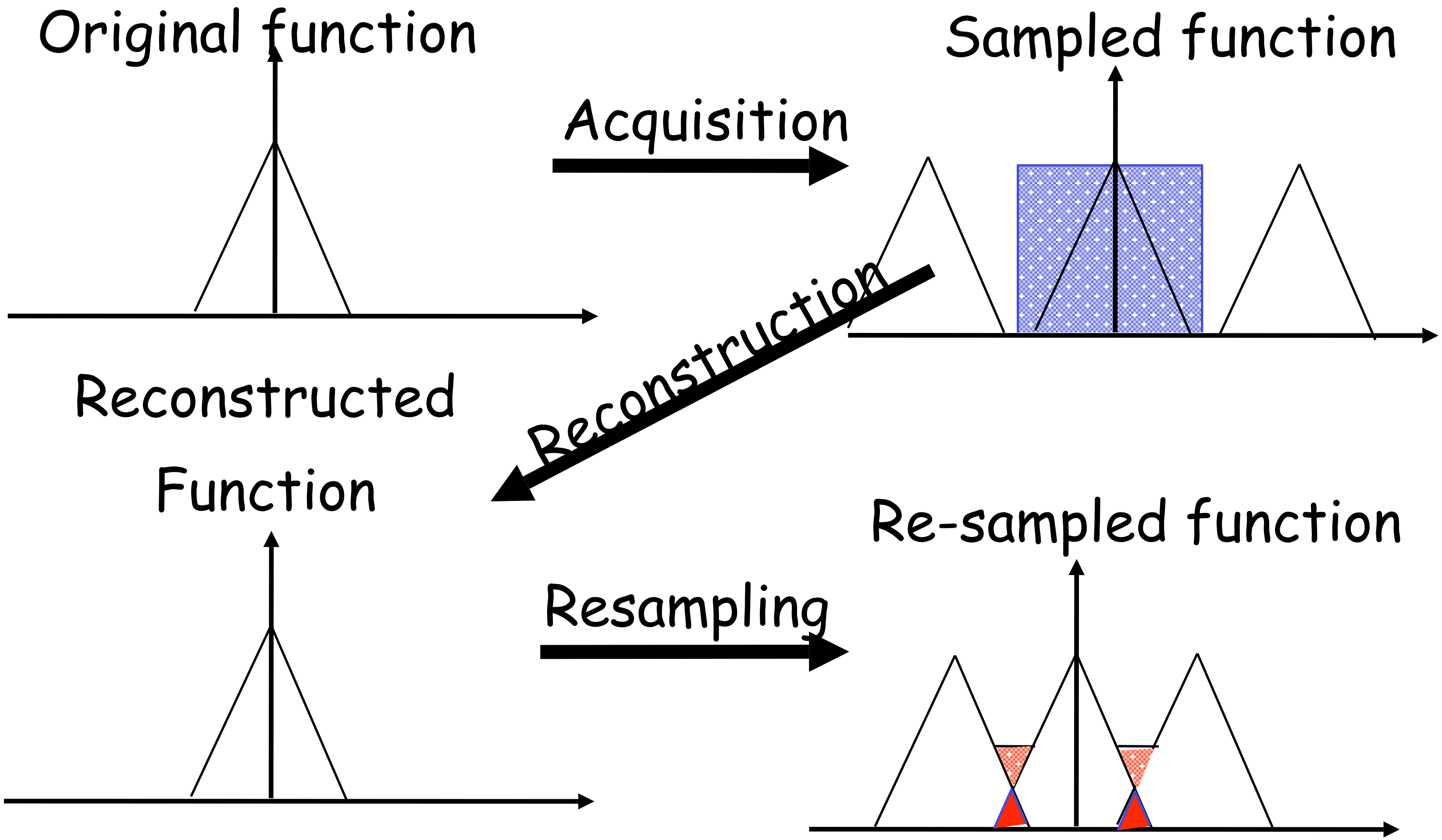
Example



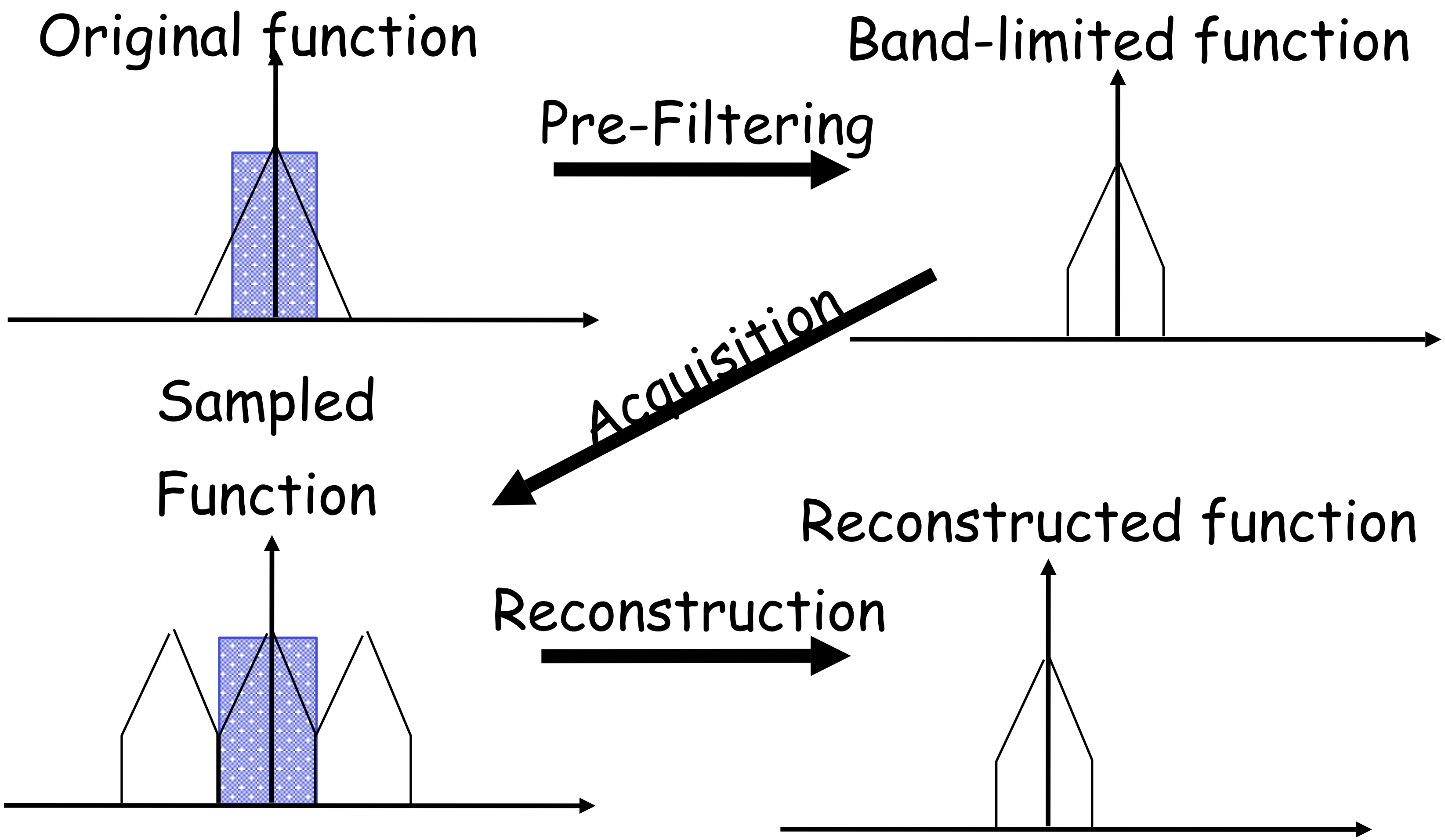
Nearest neighbor

Linear Interpolation

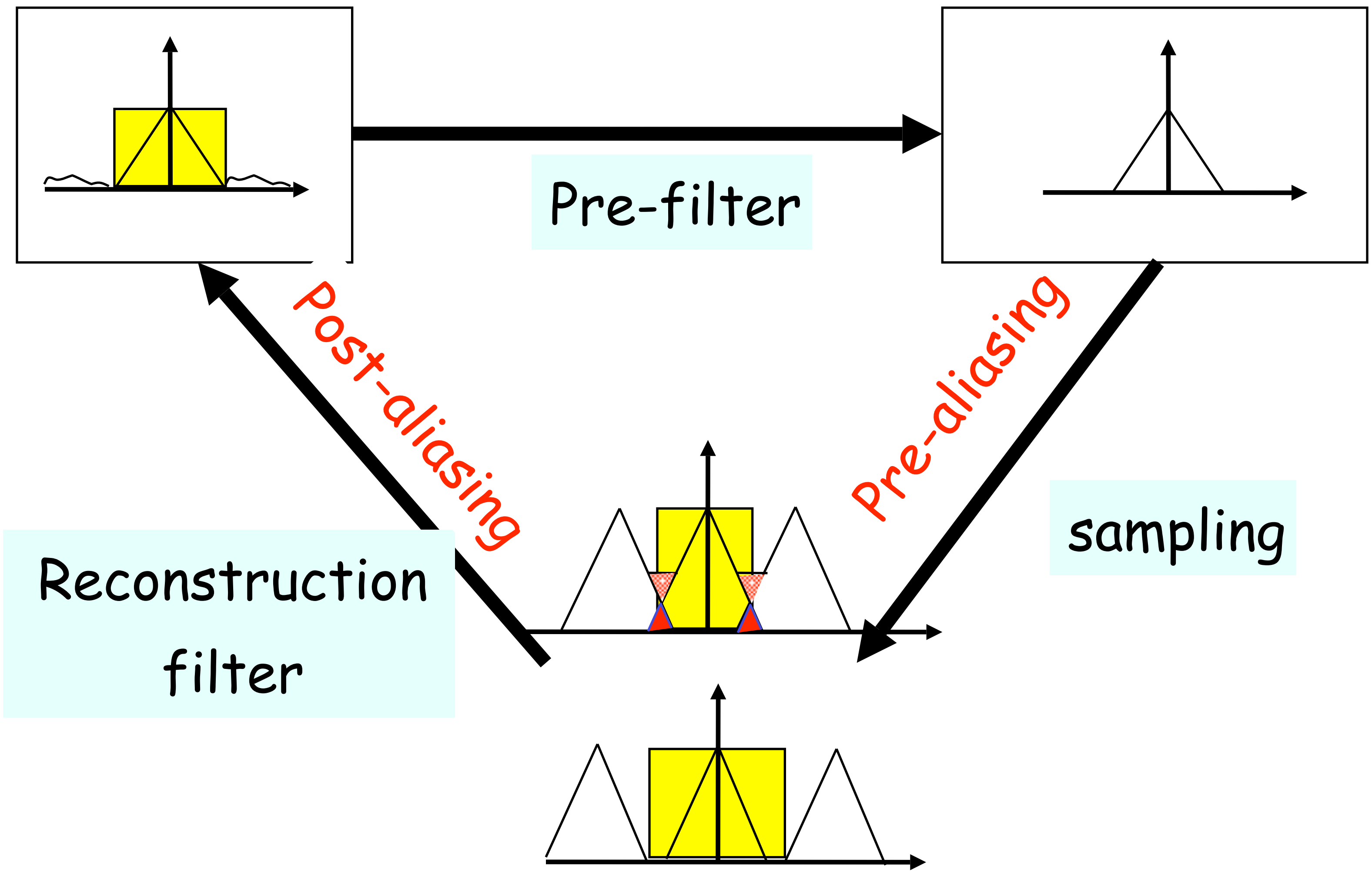
General Process - Frequency Domain



Pre-Filtering



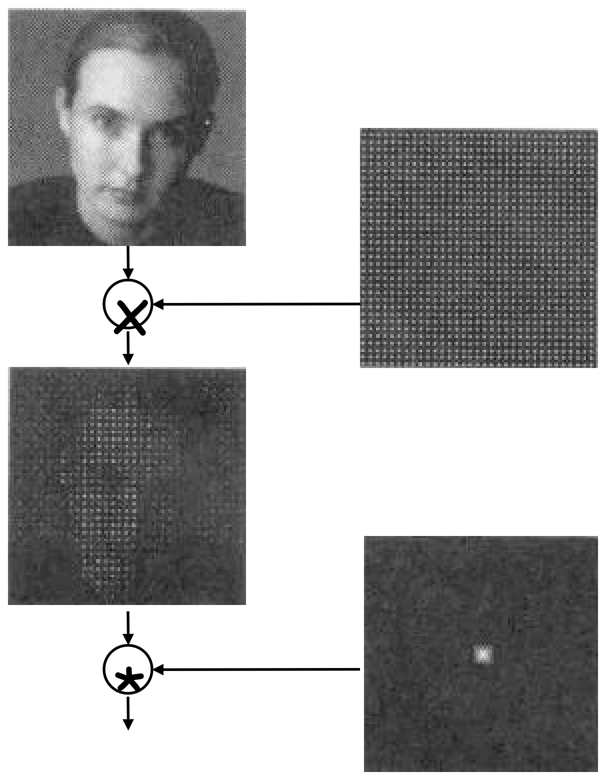
Once Again ...



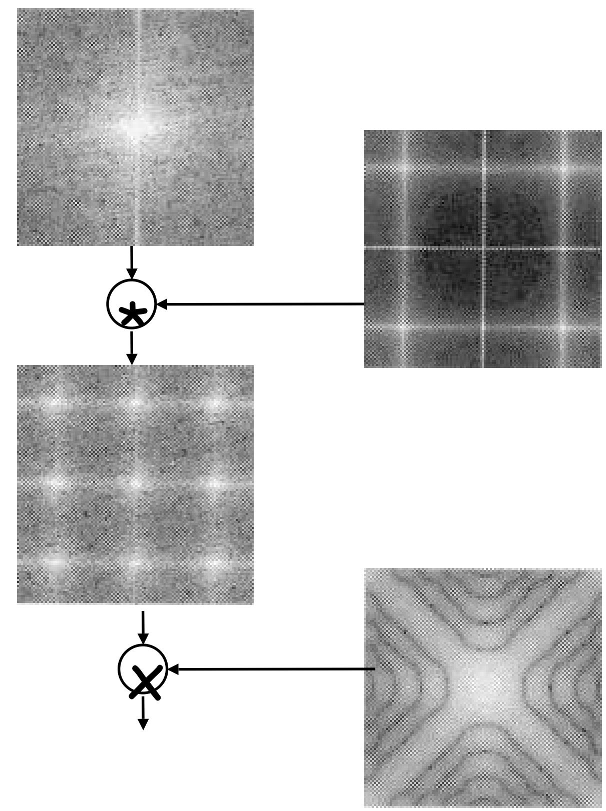
Pre-filter
Pre-aliasing
sampling
Post-aliasing
Reconstruction filter

Pipeline - Example

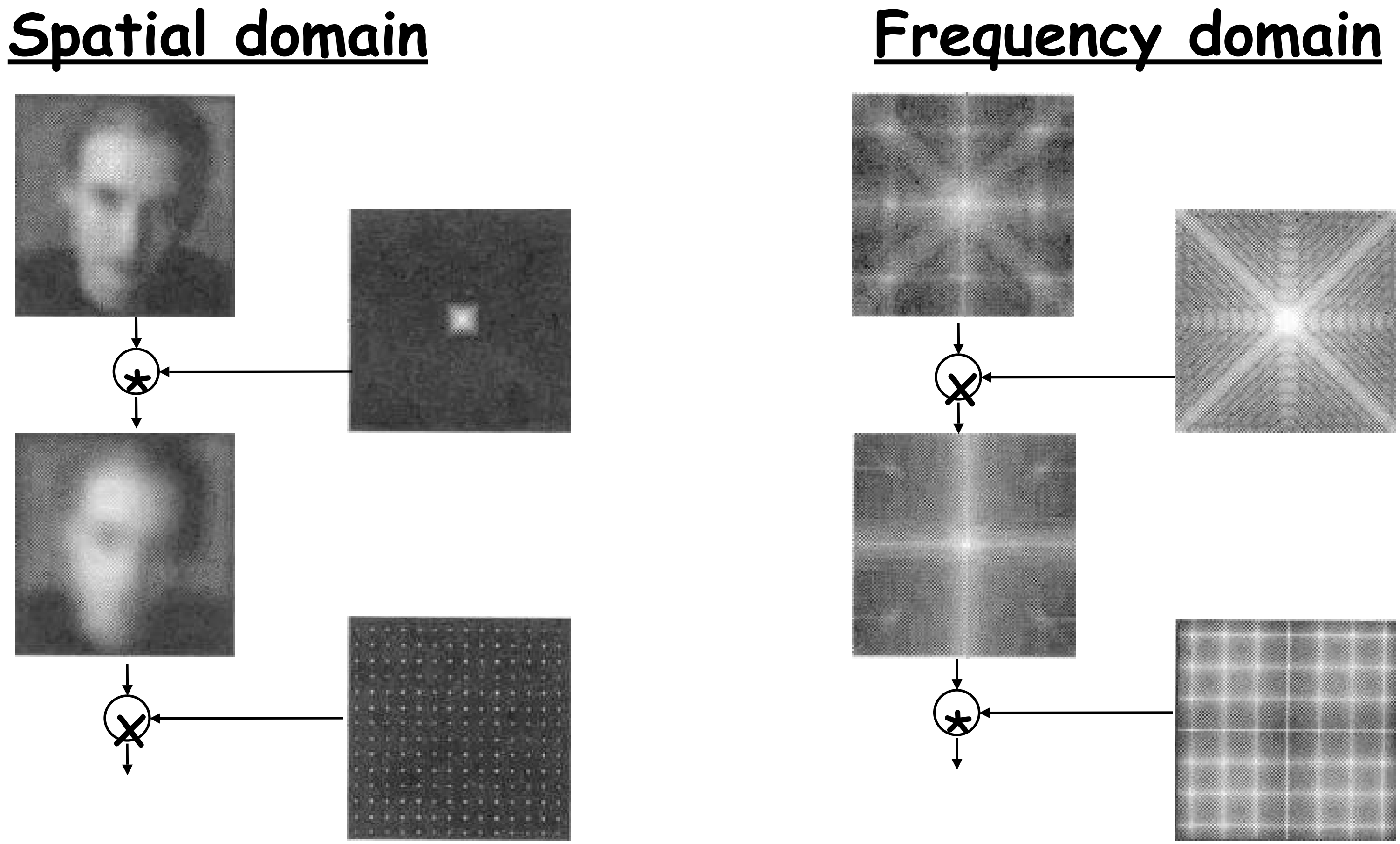
Spatial domain



Frequency domain

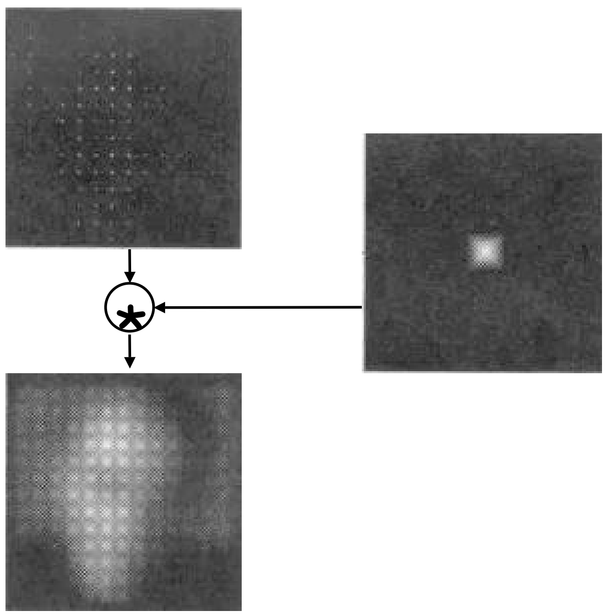


Pipeline - Example (2)

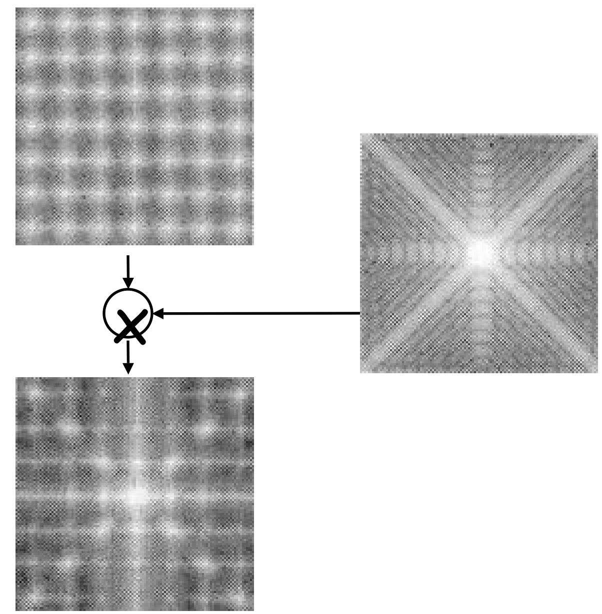


Pipeline - Example (3)

Spatial domain



Frequency domain



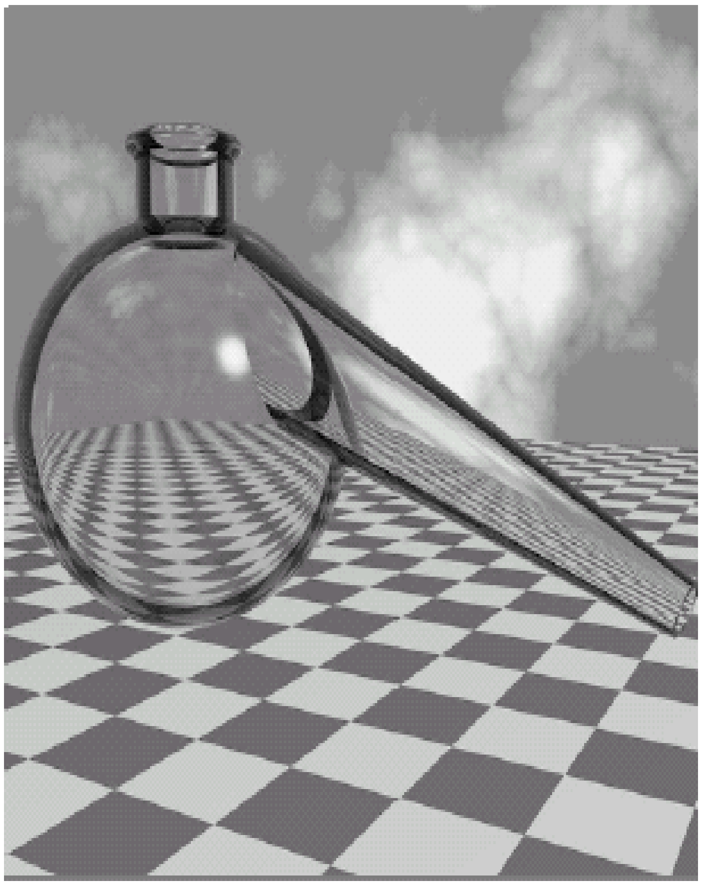
Sources of Aliasing

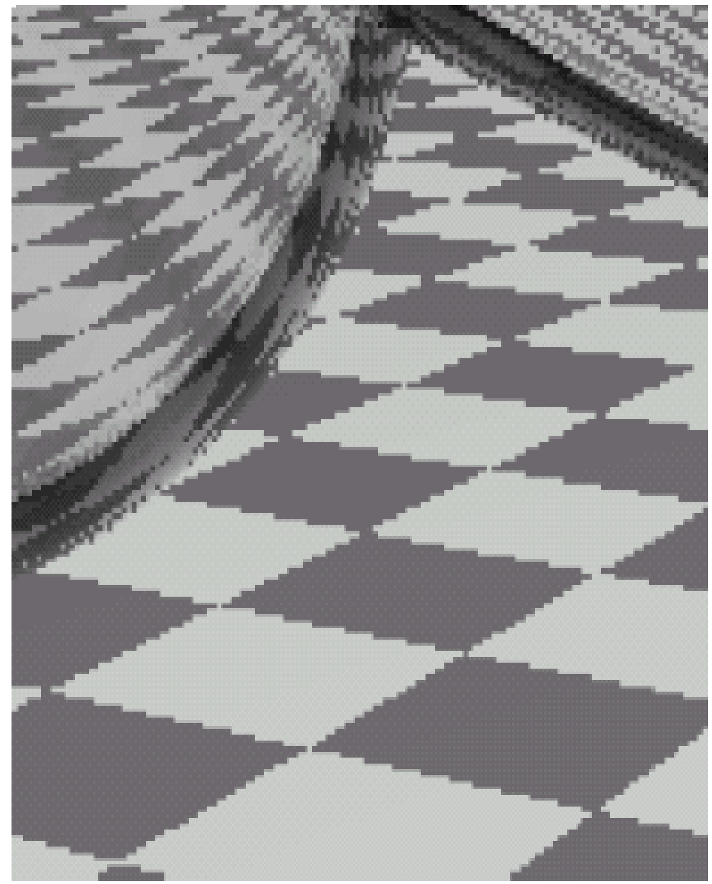
- Non-bandlimited signal

- Low sampling rate (below Nyquist)

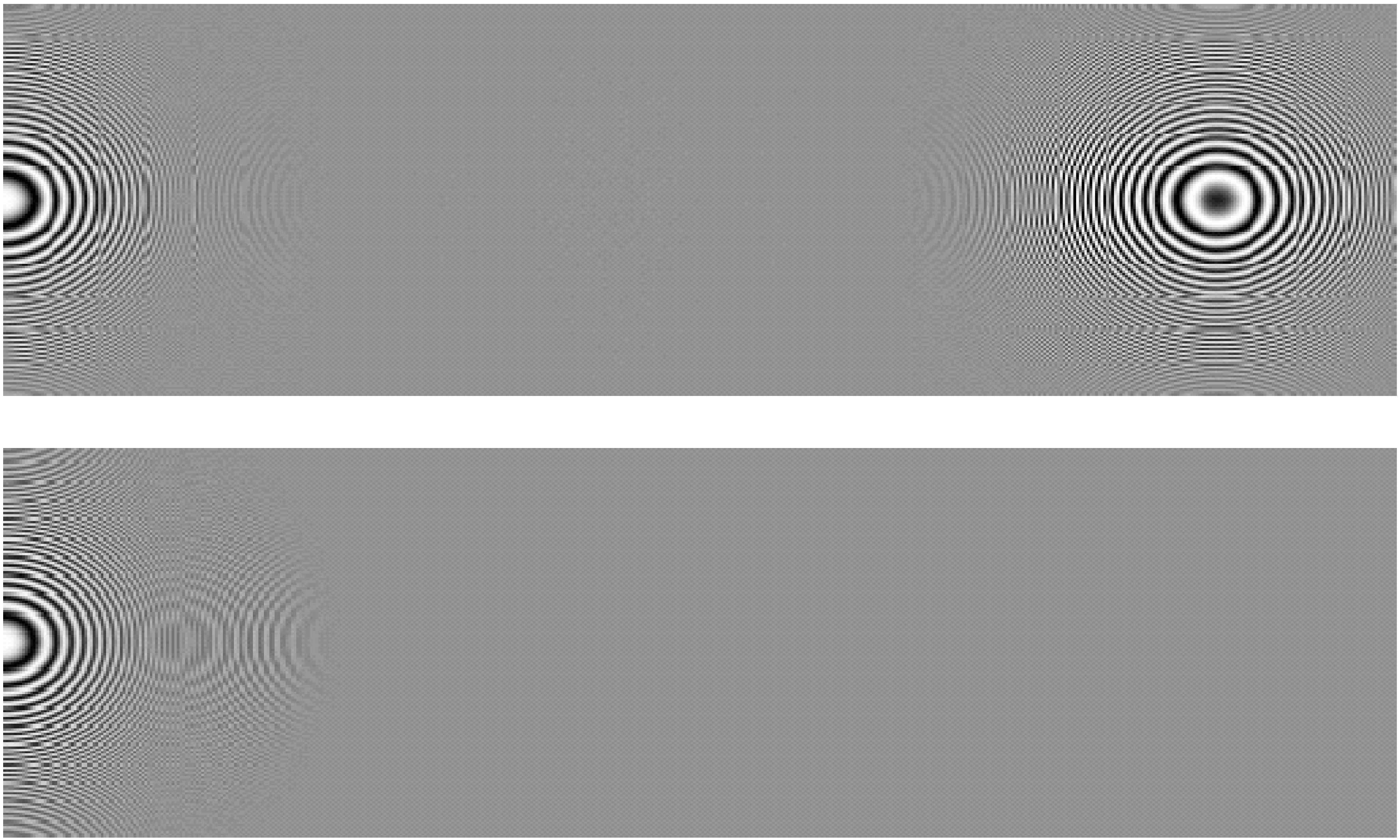
- Non perfect reconstruction

Aliasing





Bandlimited

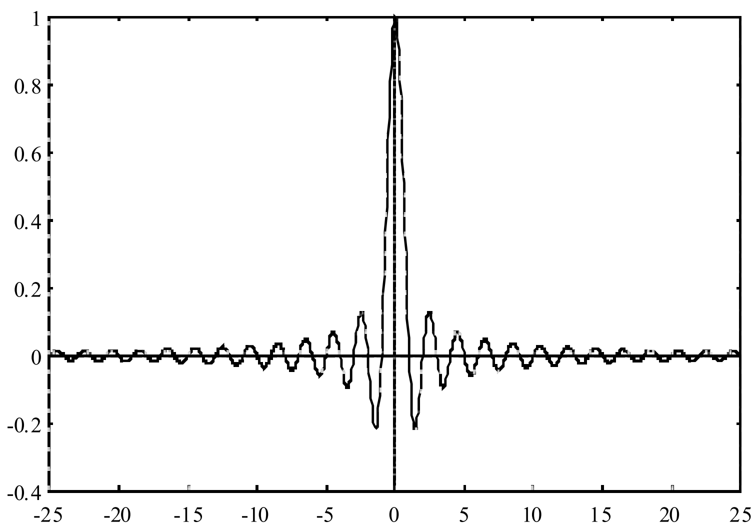


Interpolation

Spatial Domain:

- convolution is exact

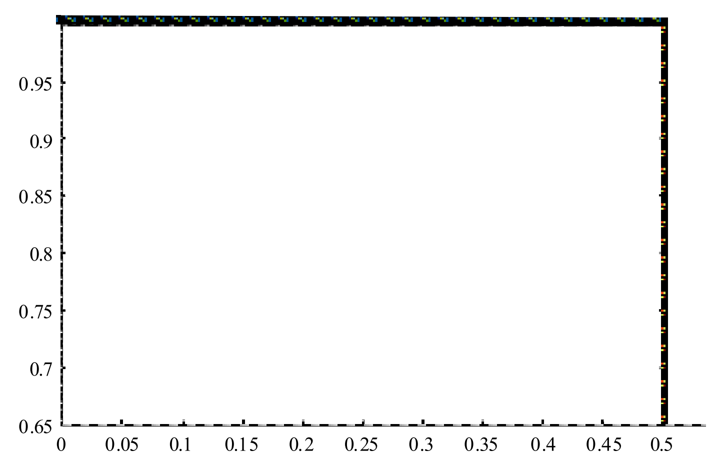
$$f_r(x) - f(x) = 0$$



Frequency Domain:

- cut off freq. replica

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

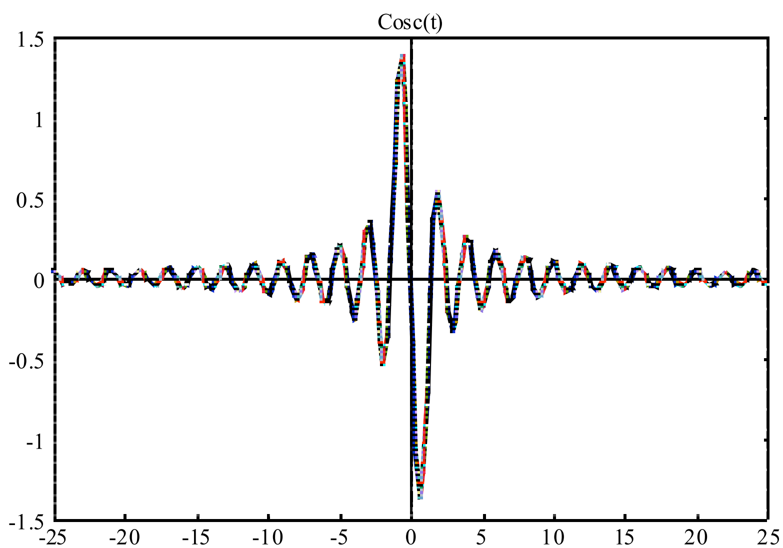


Derivatives

Spatial Domain:

- convolution is exact

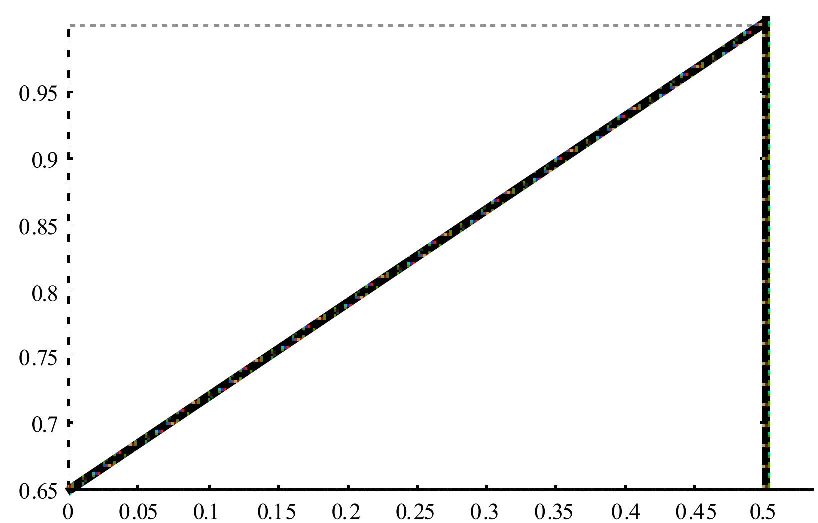
$$f_r^d(x) - f'(x) = 0$$



Frequency Domain:

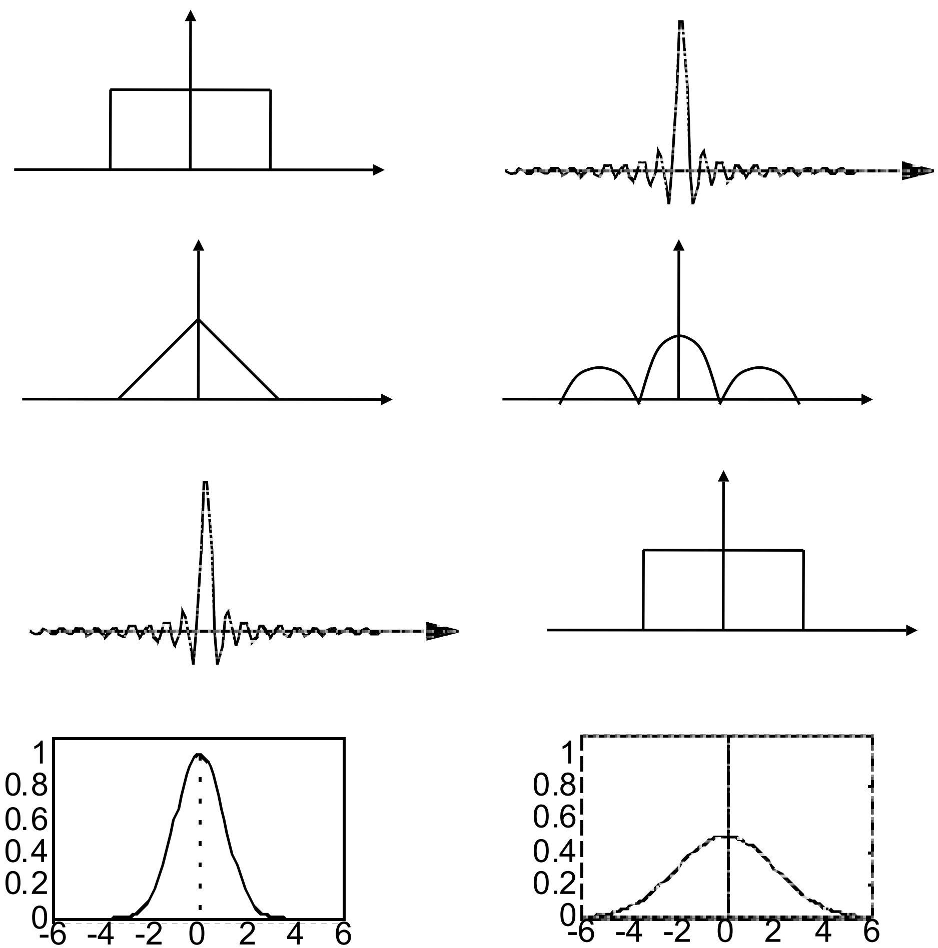
- cut off freq. replica

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



Reconstruction Kernels

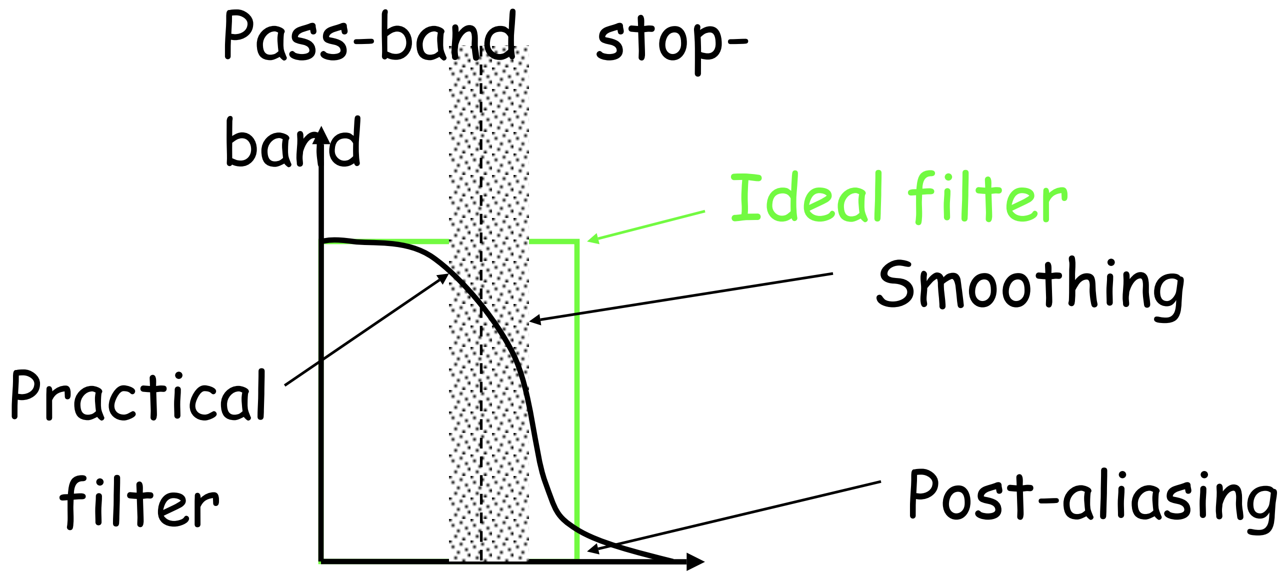
- Nearest Neighbor (Box)
- Linear
- Sinc
- Gaussian
- Many others



Spatial d. Frequency d.

Ideal Reconstruction

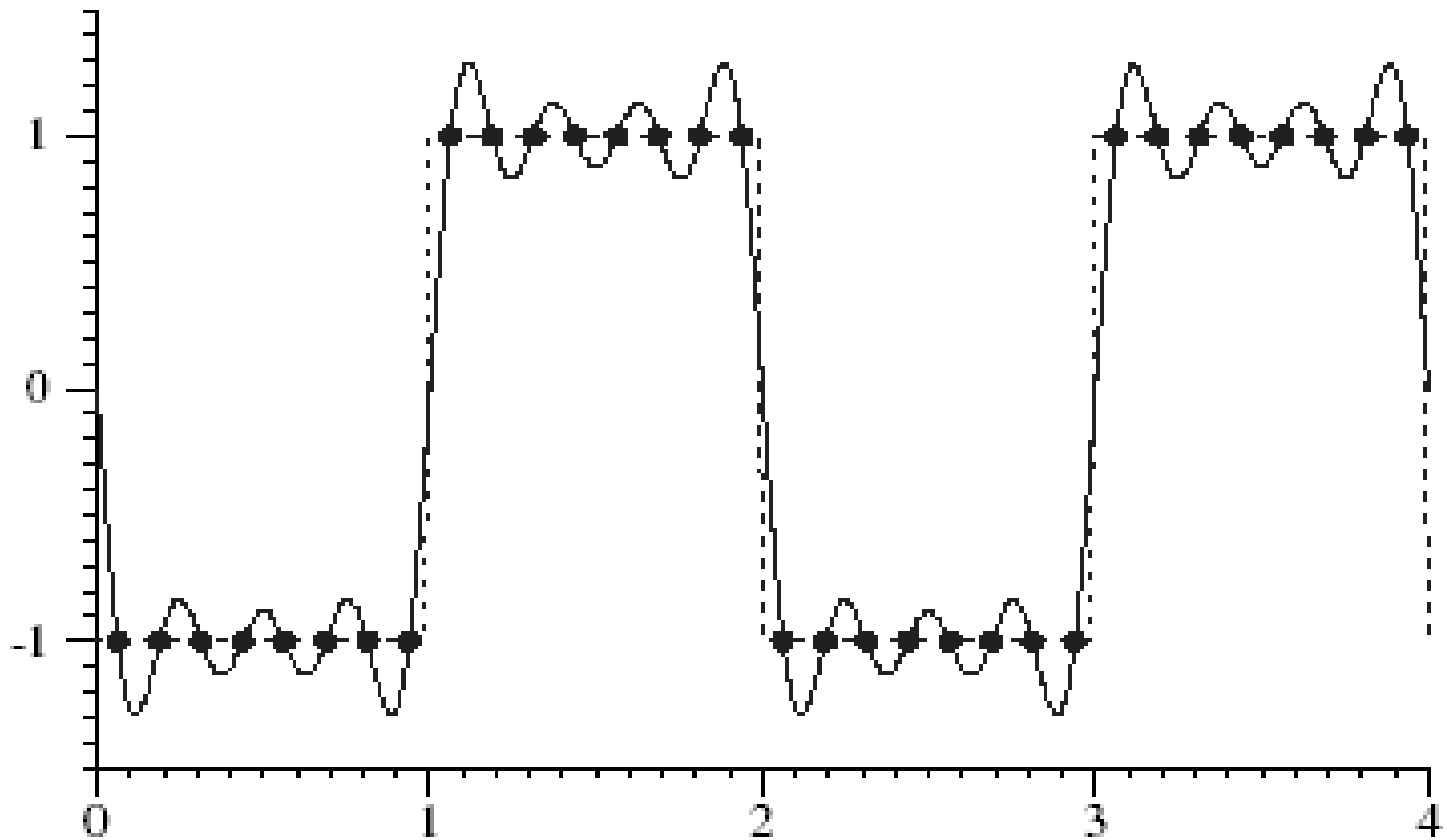
- Box filter in frequency domain =
- Sinc Filter in spatial domain
- impossible to realize (really?)



Ideal Reconstruction

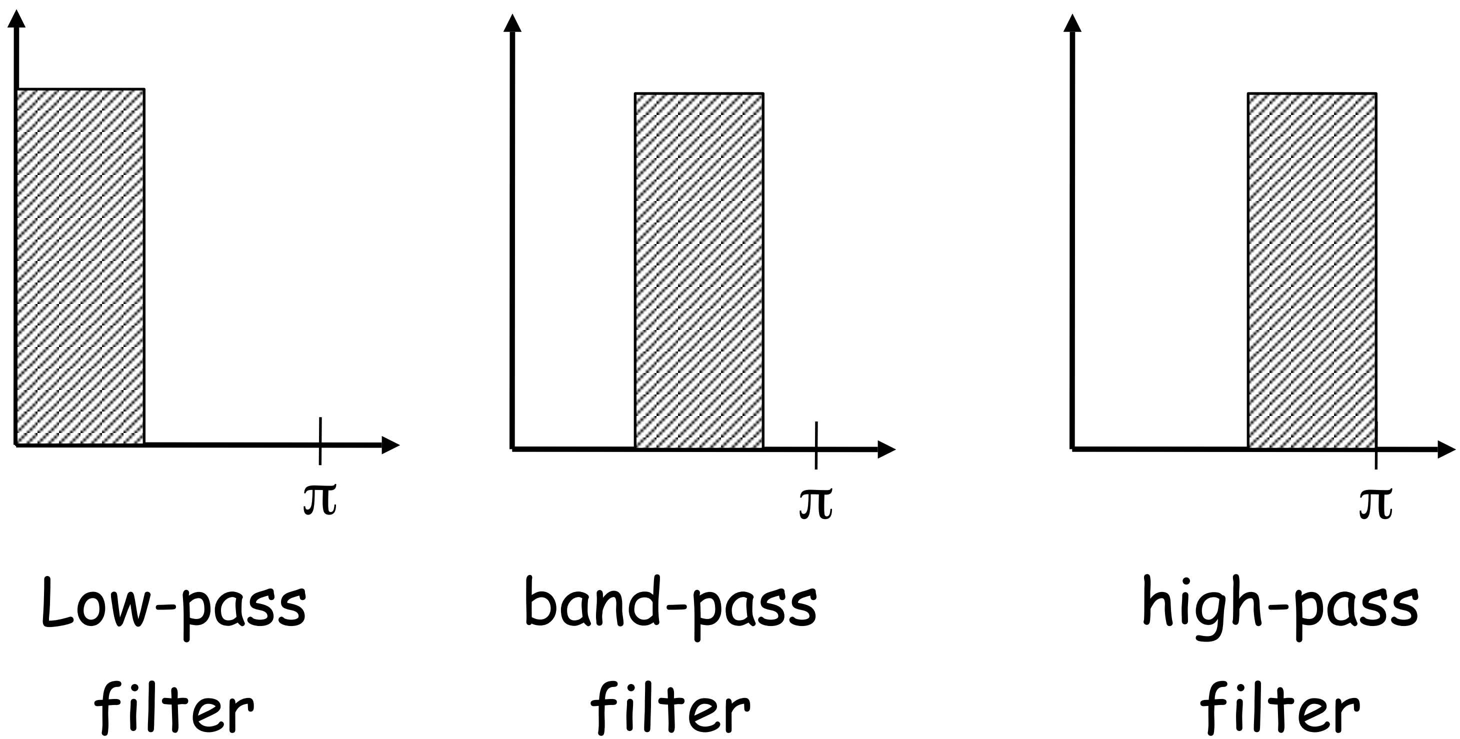
- Use the sinc function – to bandlimit the sampled signal and remove all copies of the spectra introduced by sampling
- But:
  - The sinc has infinite extent and we must use simpler filters with finite extents.
  - The windowed versions of sinc may introduce ringing artifacts which are perceptually objectionable.

Reconstructing with Sinc



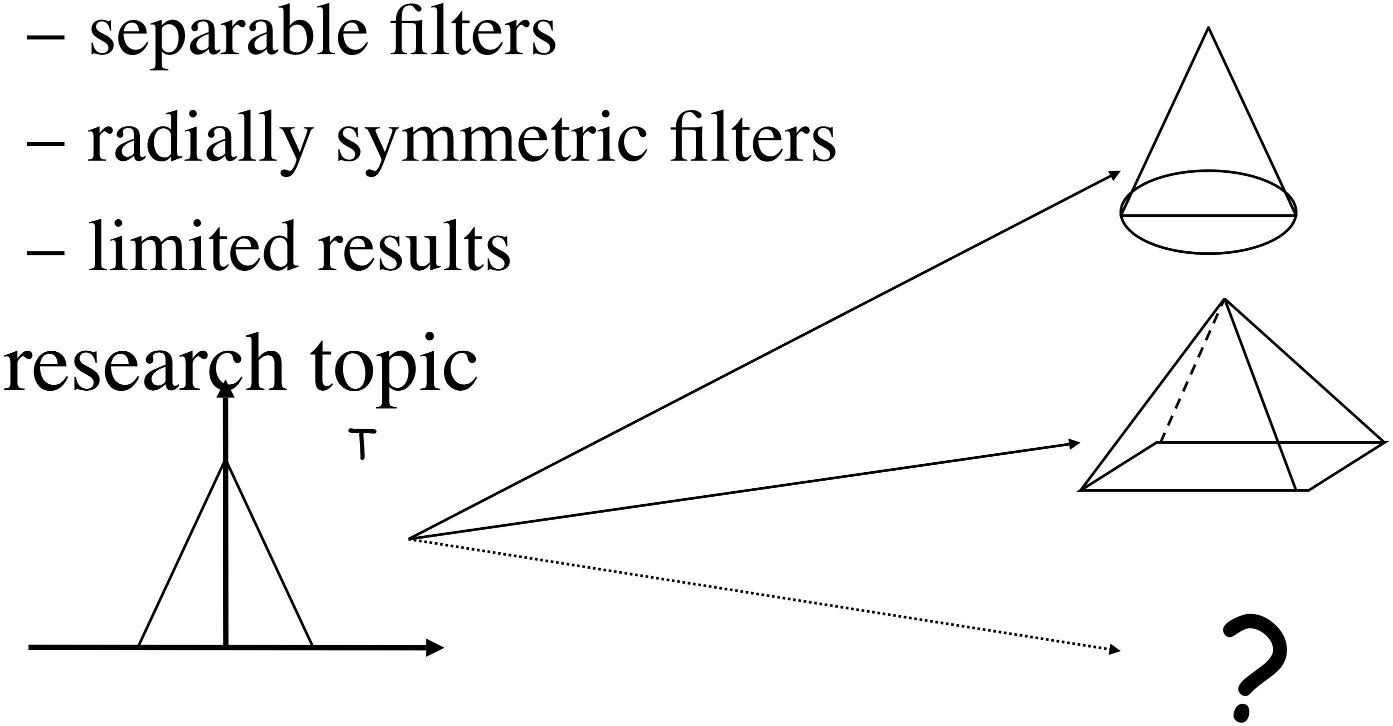
Ideal Reconstruction

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



Higher Dimensions?

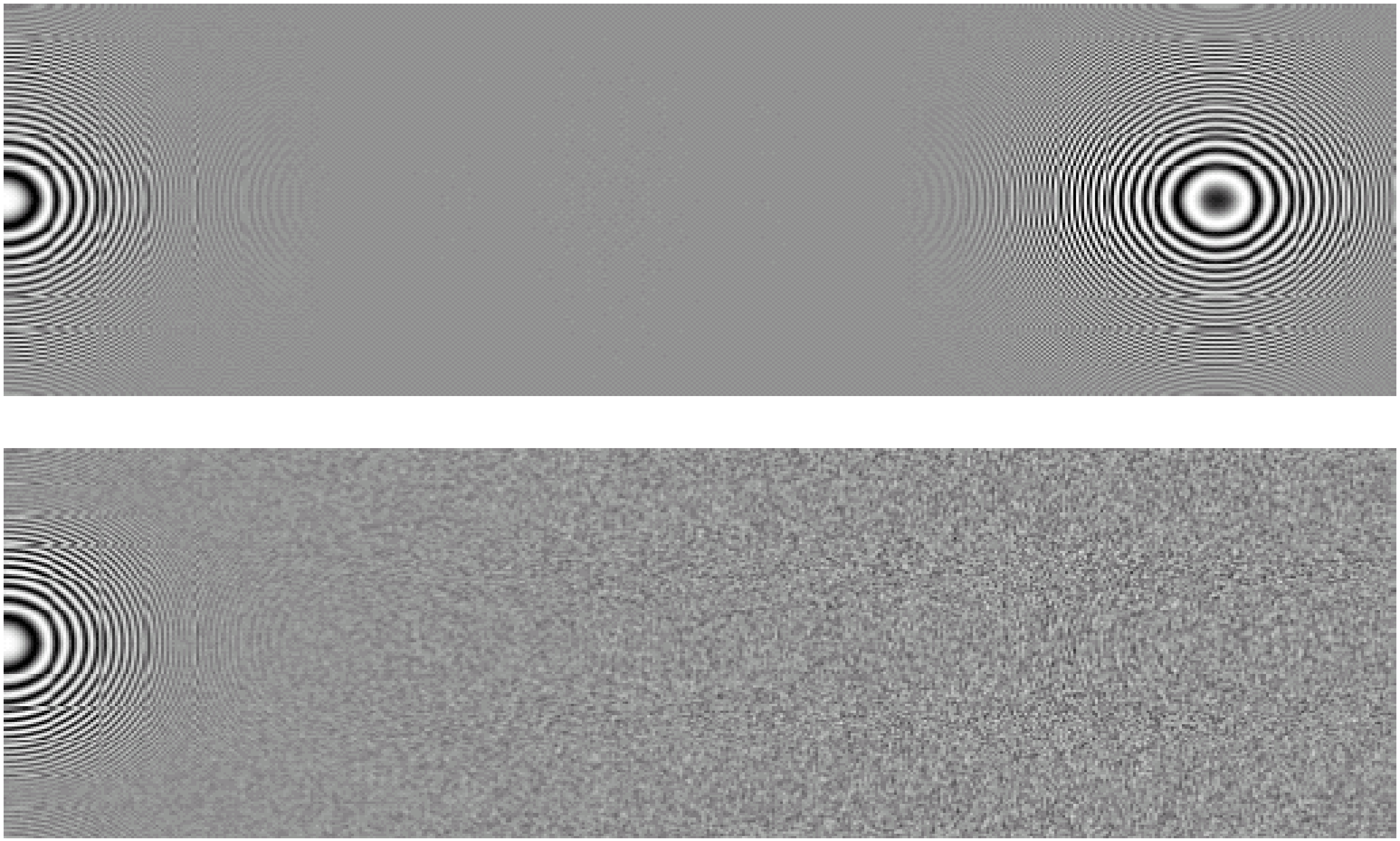
- Design typically in 1D
- extensions to higher dimensions (typically):
  - separable filters
  - radially symmetric filters
  - limited results
- research topic



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
  - frequencies below the Nyquist frequency are attenuated
- Ringing (overshoot)
  - occurs when trying to sample/reconstruct discontinuity
- Anisotropy
  - caused by not spherically symmetric filters

Aliasing vs. Noise

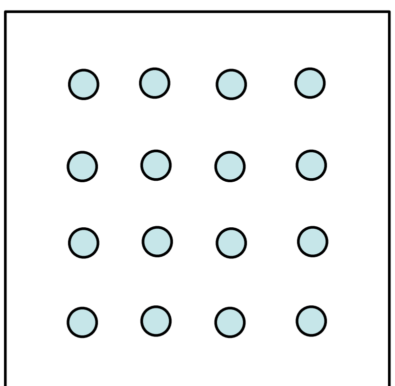


Antialiasing

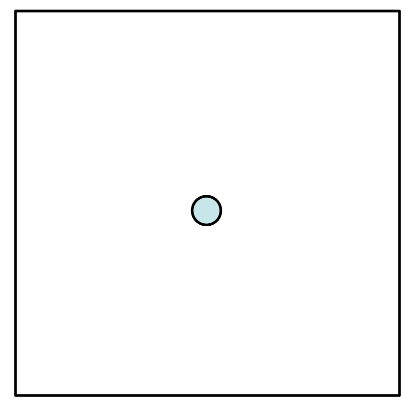
- Antialiasing = Preventing aliasing
- 1. Analytically pre-filter the signal
  - Solvable for points, lines and polygons
  - Not solvable in general (e.g. procedurally defined images)
- 2. Uniform supersampling and resample
- 3. Nonuniform or stochastic sampling

Uniform Supersampling

- Increasing the sampling rate moves each copy of the spectra further apart, potentially reducing the overlap and thus aliasing
- Resulting samples must be resampled (filtered) to image sampling rate



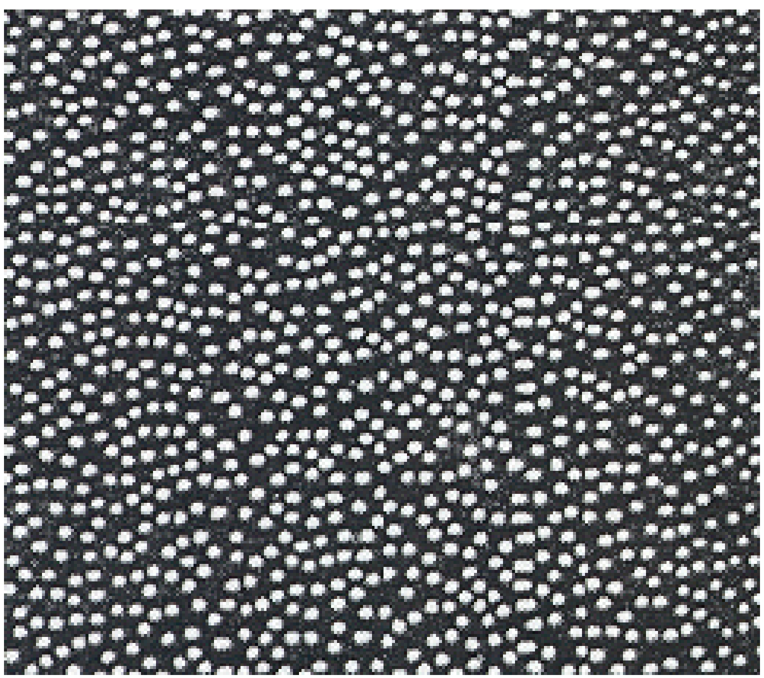
$$Pixel = \sum_k w_k \times Sample_k$$



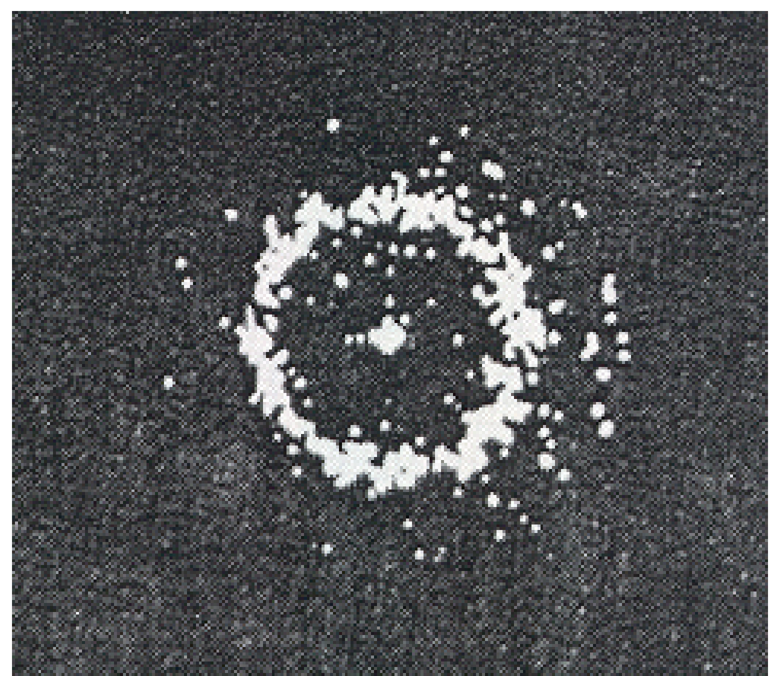
Distribution of Extrafoveal Cones

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

Monkey eye cone distribution



Fourier Transform

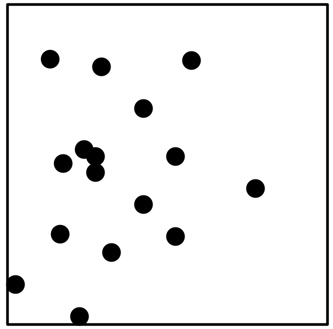


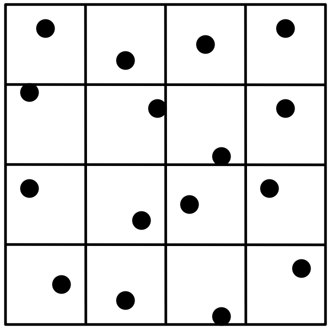
Non-Uniform Sampling - Intuition

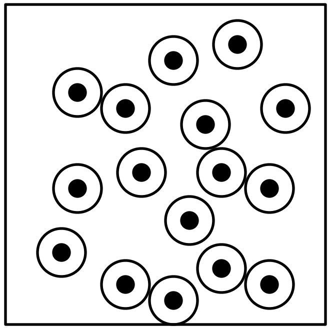
- Uniform sampling
  - The spectrum of uniformly spaced samples is also a set of uniformly spaced spikes
  - Multiplying the signal by the sampling pattern corresponds to placing a copy of the spectrum at each spike (in freq. space)
  - Aliases are coherent, and very noticeable
- Non-uniform sampling
  - Samples at non-uniform locations have a different spectrum; a single spike plus noise
  - Sampling a signal in this way converts aliases into broadband noise
  - Noise is incoherent, and much less objectionable

Non-Uniform Sampling - Patterns

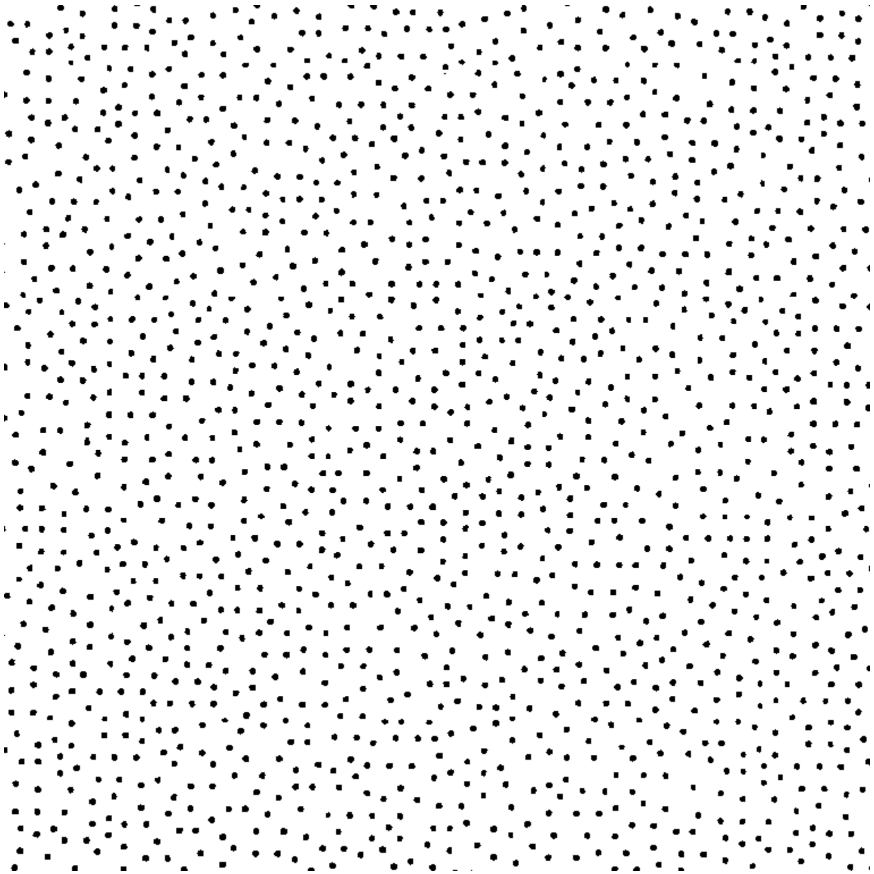
- Poisson
  - Pick n random points in sample space
- Uniform Jitter
  - Subdivide sample space into n regions
- Poisson Disk
  - Pick n random points, but not too close



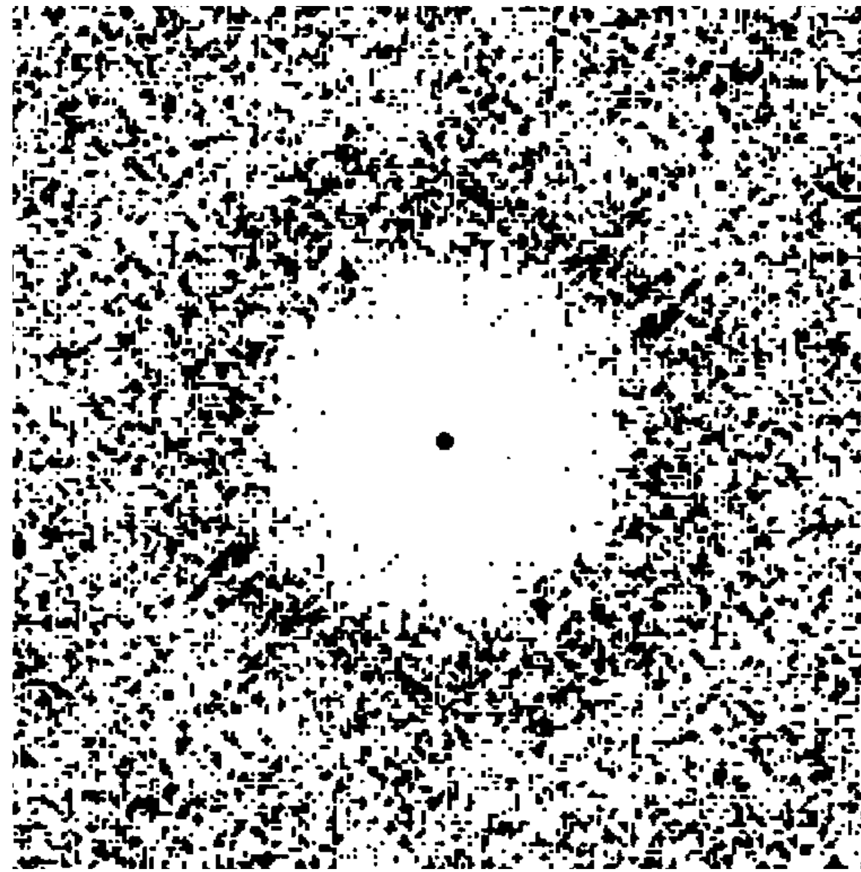




Poisson Disk Sampling

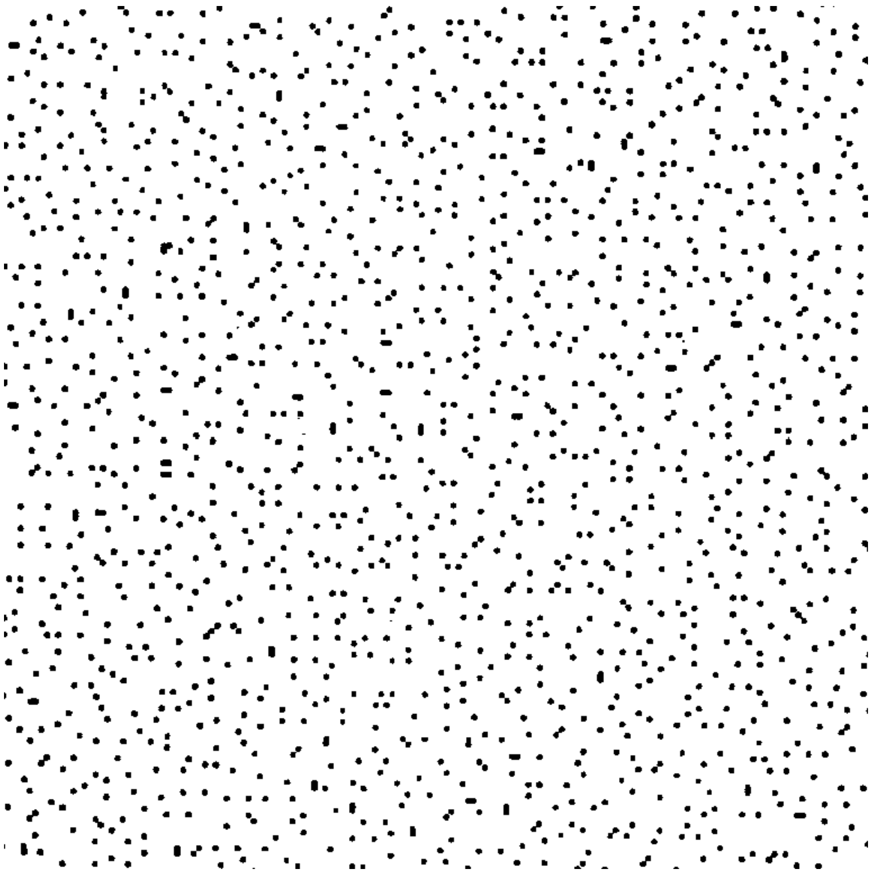


Spatial Domain

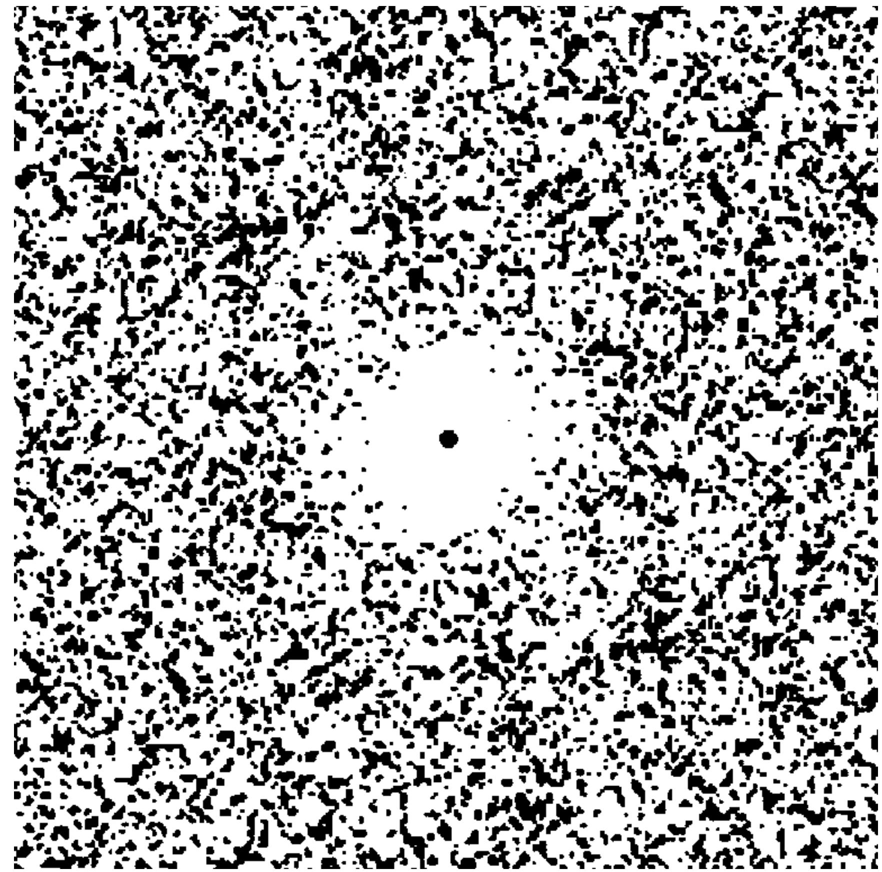


Fourier Domain

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

- Put at least one sample in each strata
- Multiple samples in strata do no good
- Also have samples far away from each other

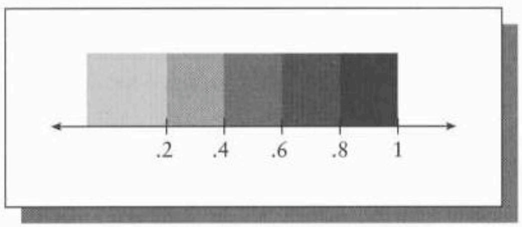
- Graphics: jittering

Stratification

- OR
  - Split up the integration domain in N disjoint sub-domains or strata
  - Evaluate the integral in each of the sub-domains separately with one or more samples.
- More precisely:

$$\int_0^1 f(x)dx = \int_0^{\alpha_1} f(x)dx + \int_{\alpha_1}^{\alpha_2} f(x)dx + \ldots + \int_{\alpha_{m-2}}^{\alpha_{m-1}} f(x)dx + \int_{\alpha_{m-1}}^{1} f(x)dx$$

Stratification



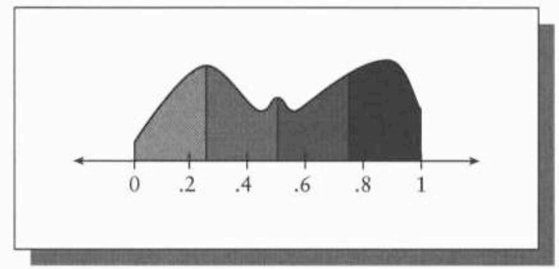
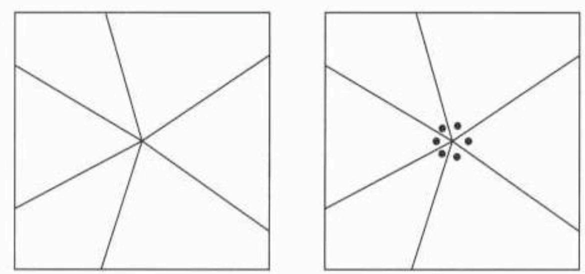


FIGURE 9.11

A signal in the interval [0,1] broken into four equal strata.



(a) (b)

More Jittered Sequences

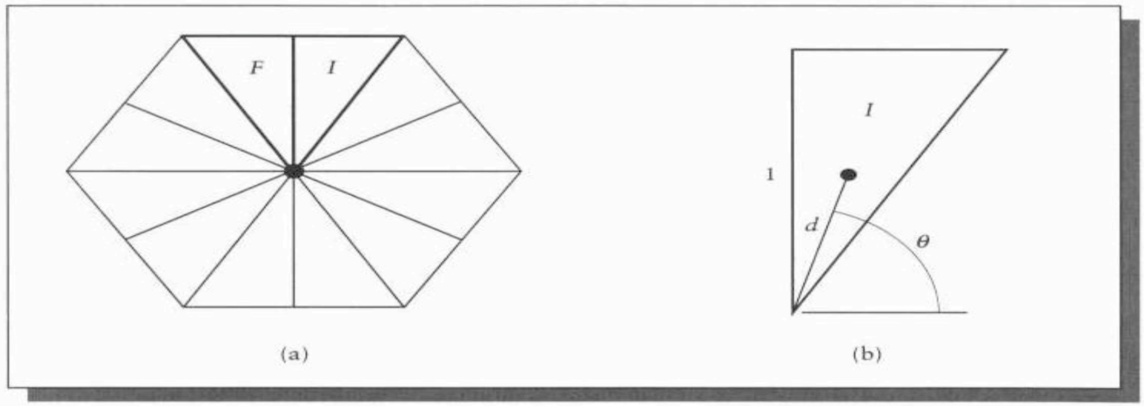


FIGURE 10.20
A hexagon broken up into twelve equivalent regions. (a) The initial (I) and flipped (F) regions. (b) Finding a point within I.

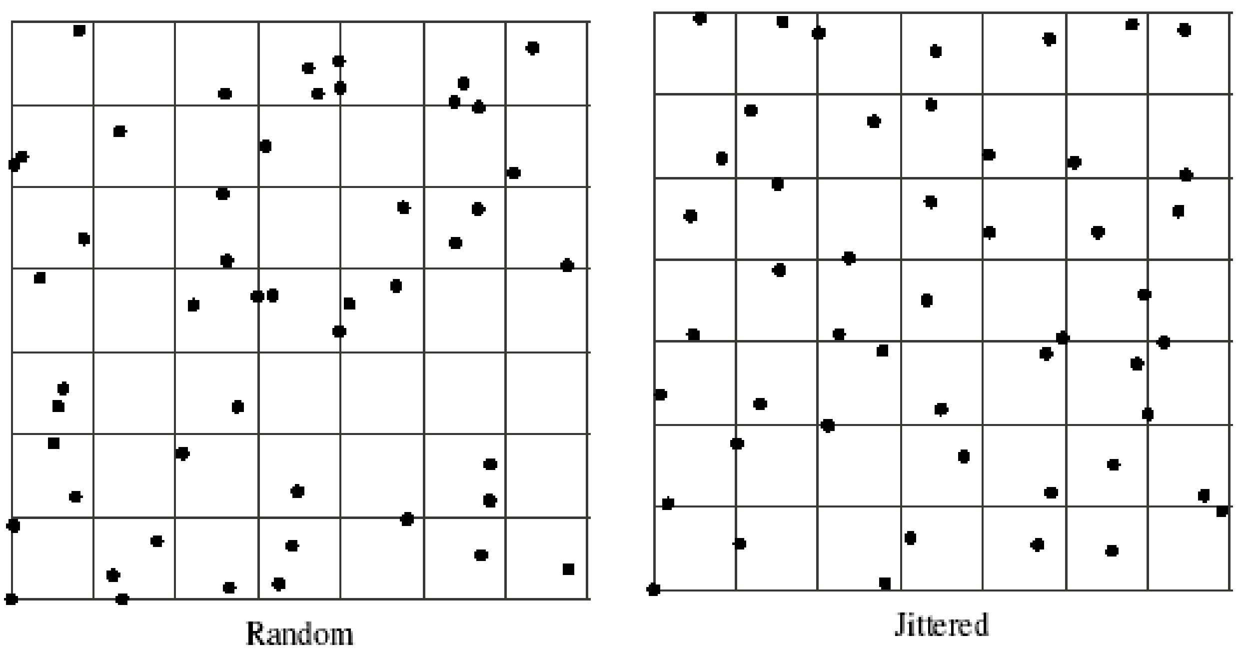
for $r \leftarrow 0$ to $h-1$ for $c \leftarrow 0$ to $w-1$	*Scan all rows and columns.*
$\theta \leftarrow$ randomInterval$(\pi/3, \pi/2)$ $d \leftarrow$ range$(1/(2\cos\theta))$ $\Delta x \leftarrow d\cos\theta$ $\Delta y \leftarrow d\sin\theta$	*Pick a random point in the primary region.*

if flip() then
  $\Delta x \leftarrow -\Delta x$
  endif
	Perhaps flip it into region F.

$\phi \leftarrow (\pi/3)*$ randomInteger(0, 5)	*Pick one of six sides to rotate into.*
$\Delta x' \leftarrow \Delta x \cos\phi + \Delta y \sin\phi$ $\Delta y' \leftarrow -\Delta x \sin\phi + \Delta y \cos\phi$	*Rotate the jitter vector.*
$k \leftarrow (rw) + c$ $x_k \leftarrow (3c)/\sqrt{3} + \Delta x'$ $y_k \leftarrow 2(r + (c \bmod 2)) + \Delta y'$	*Add the jitter into the hexagon center.*
endfor endfor	

Jitter

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



Texture Example

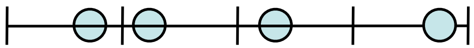
Exact – 256 samples/pixel

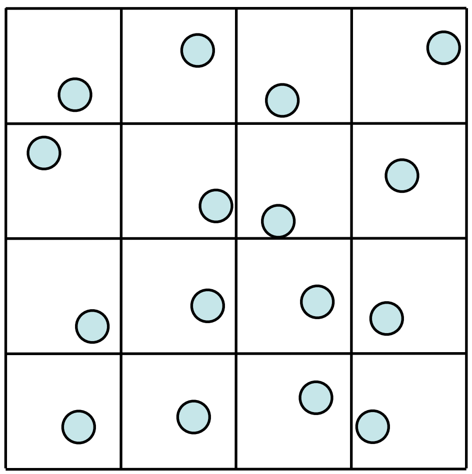
Jitter with 1 sample/pixel

1 sample/pixel

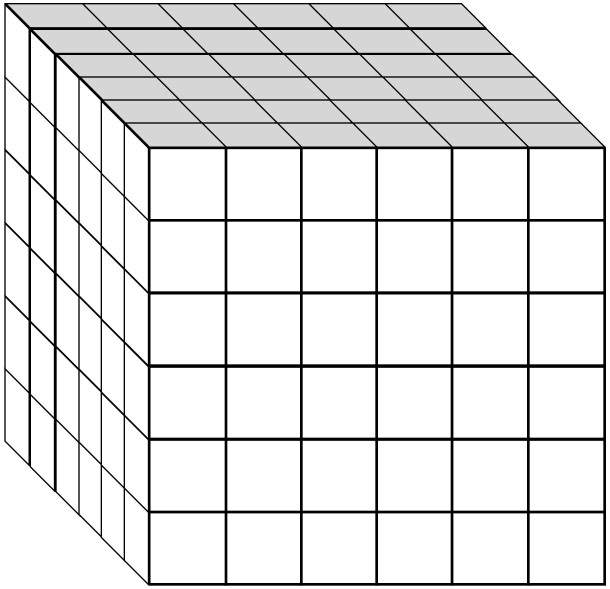
Jitter with 4 samples/pixel

Multiple Dimensions

- Too many samples
- 1D 
- 2D



3D

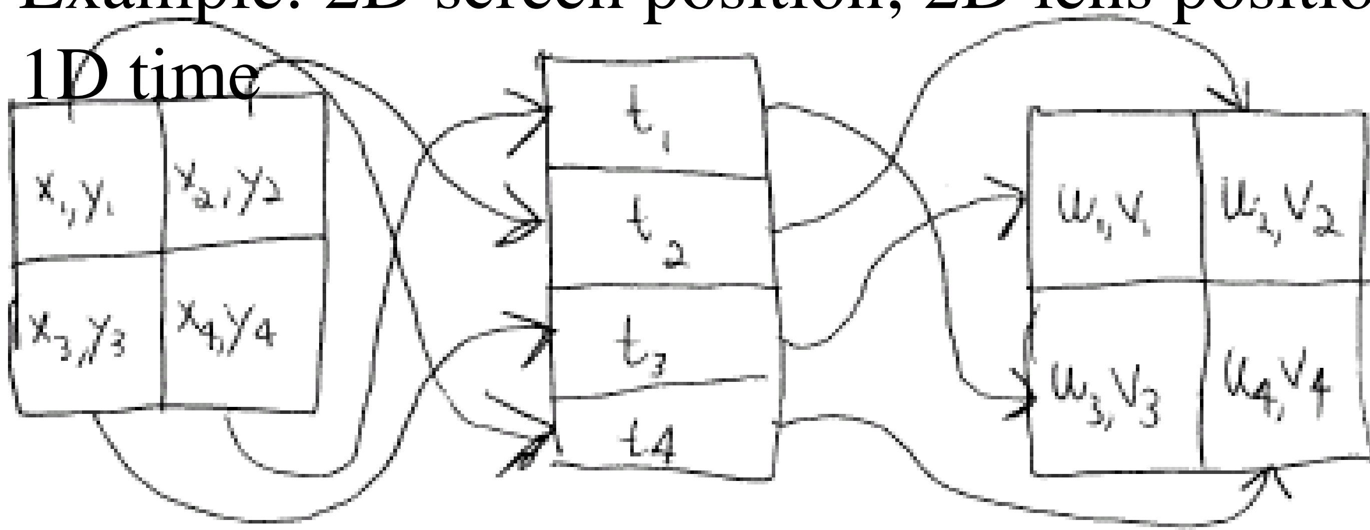


Jitter Problems

- 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
  - We can either look in non-separable geometries
  - Latin Hypercube (or N-Rook) sampling

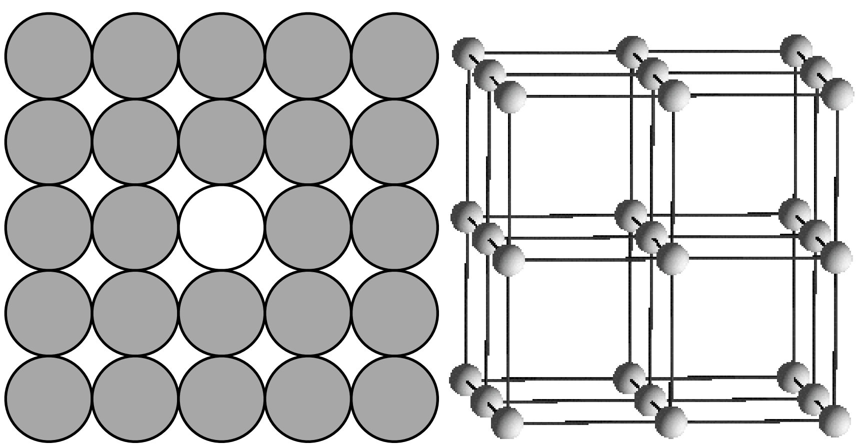
Multiple Dimensions

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

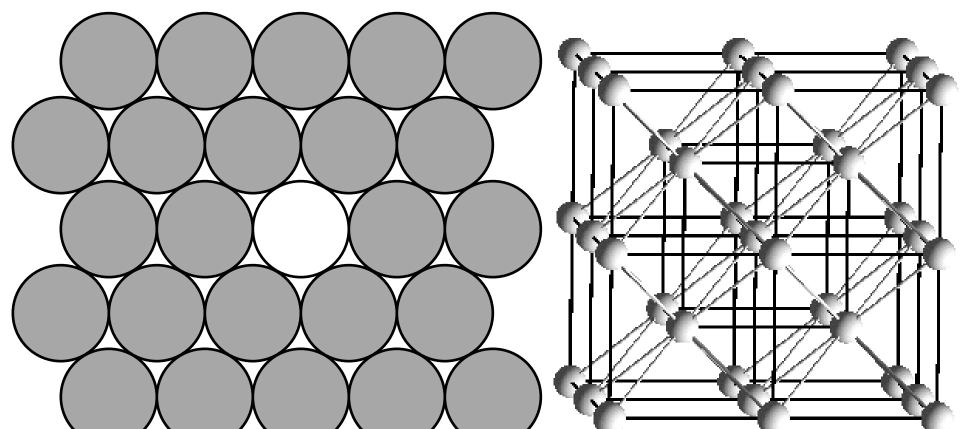


Optimal sampling lattices

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



Cartesian

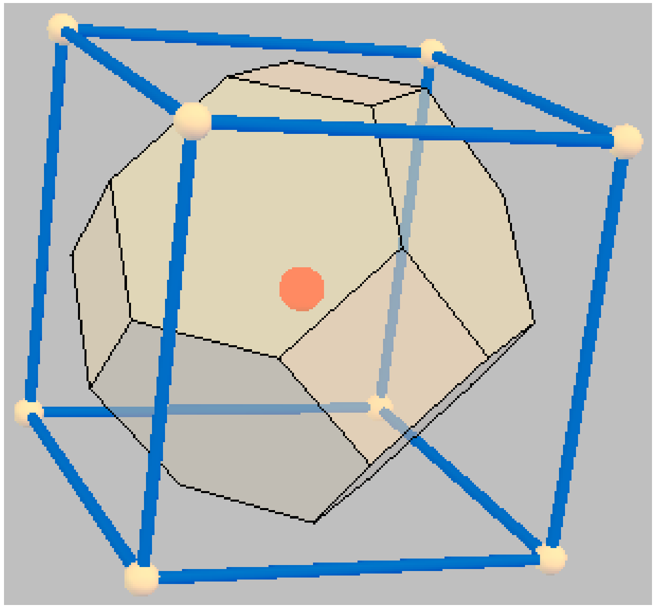


Hexagonal

Optimal sampling lattices

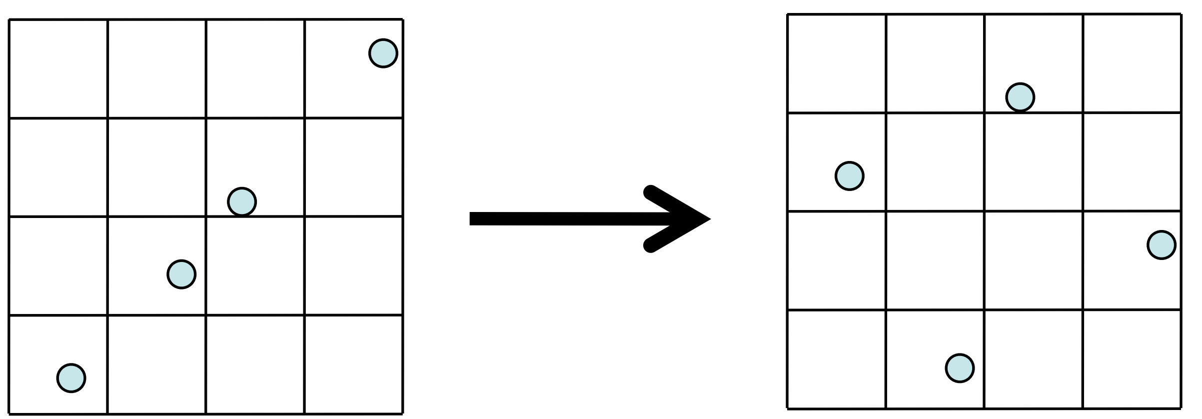
- We have to deal with different geometry
- 2D - hexagon
- 3D - truncated octahedron

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.



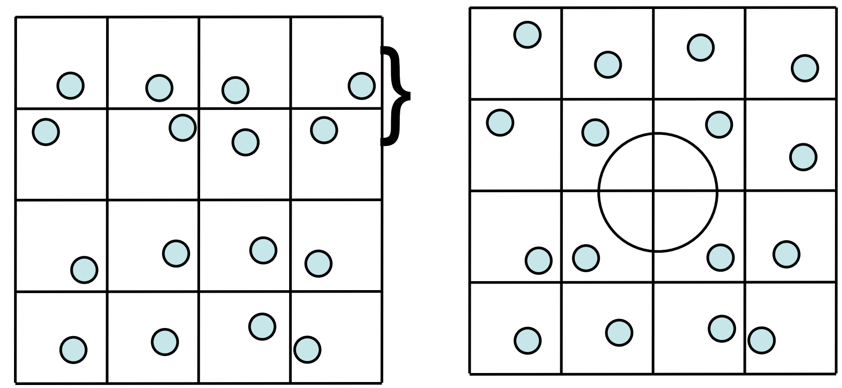
Latin Hypercubes - N-Rooks

- Distributing n samples in D dimensions, even if n is not a power of D
- Divide each dimension in n strata
- Generate a jittered sample in each of the n diagonal entries
- Random shuffle in each dimension

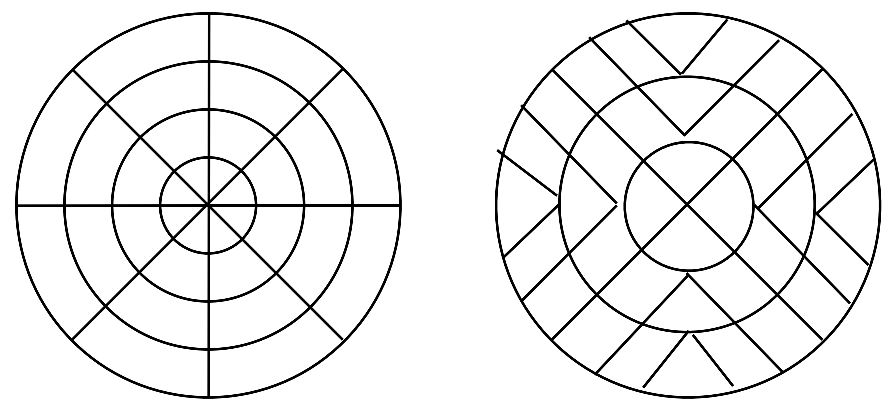


Stratification - problems

- Clamping (LHS helps)
- Could still have large empty regions



- Other geometries, e.g. stratify circles or spheres?



How good are the samples ?

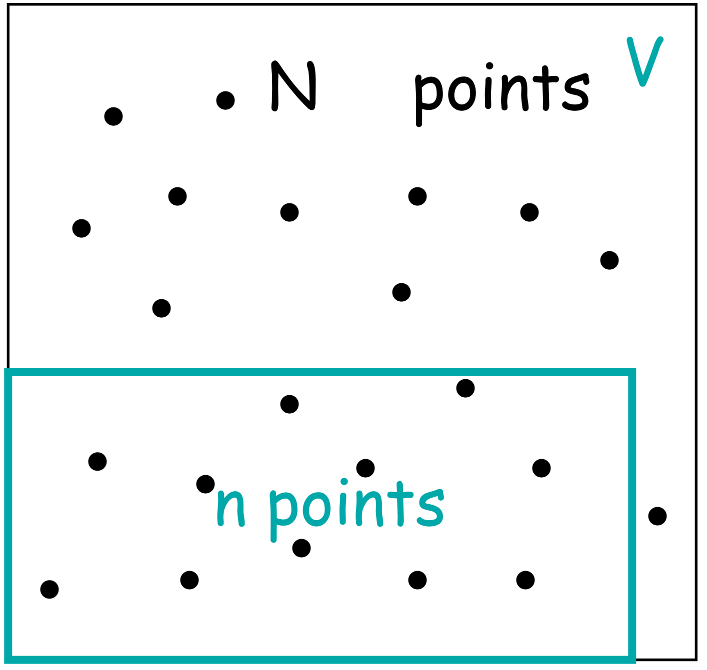
- How can we evaluate how well our samples are distributed?
  - No “holes”
  - No clamping
- Well distributed patterns have low *discrepancy*
  - Small = evenly distributed
  - Large = clustering
- Construct low discrepancy sequence

Discrepancy

- D_N - Maximum difference between the fraction of N points x_i and relative size of volume $[0,1]^n$
- Pick a set of sub-volumes B of $[0,1]^n$

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

- D_N ->0 when N is very large

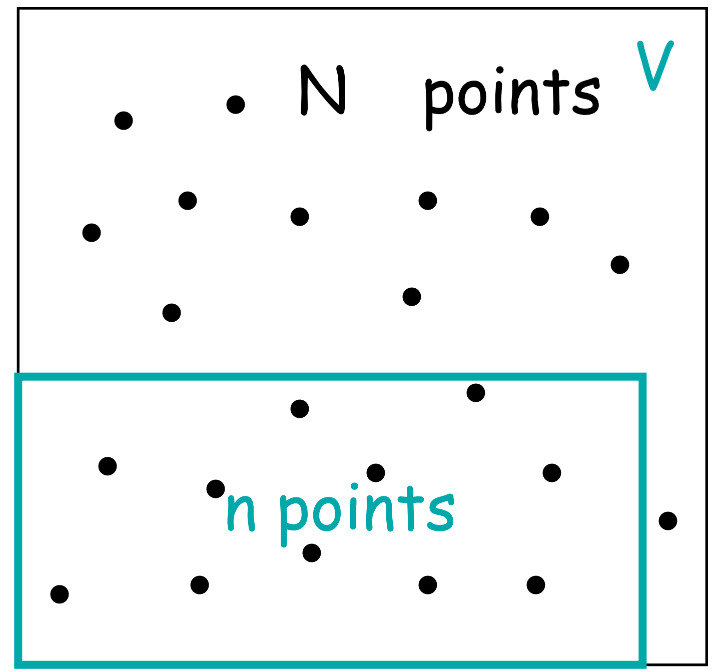


Discrepancy

- Examples of sub-volumes B of $[0,1]^d$:
  - Axis-aligned
  - Share a corner at the origin (star discrepancy)
- Best discrepancy that has been obtained in d dimensions:

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

$D_N^*(P)$



Discrepancy

- How to create low-discrepancy sequences?
  - Deterministic sequences!! Not random anymore
  - Also called pseudo-random
  - Advantage - easy to compute
- 1D:

$$x_i = \frac{i}{N} \quad \Rightarrow \quad D_N^*\left(x_1,\ldots,x_n\right) = \frac{1}{N}$$

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

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

Pseudo-Random Sequences

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

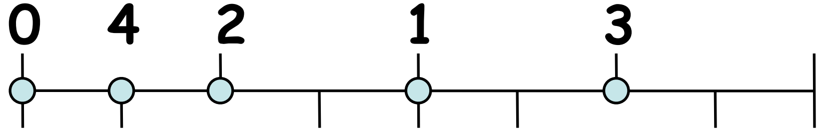
$$n = a_k \ldots a_2 a_1 = a_1 b^0 + a_2 b^1 + a_3 b^2 + \ldots$$

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

Van Der Corput Sequence

- Most simple sequence $x_i = \Phi_2(i)$
- Uses radical inverse of base 2
- Achieves minimal possible 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

- Can be used if N is not known in advance
- All prefixes of a sequence are well distributed
- Use prime number bases for each dimension
- Achieves best possible discrepancy

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

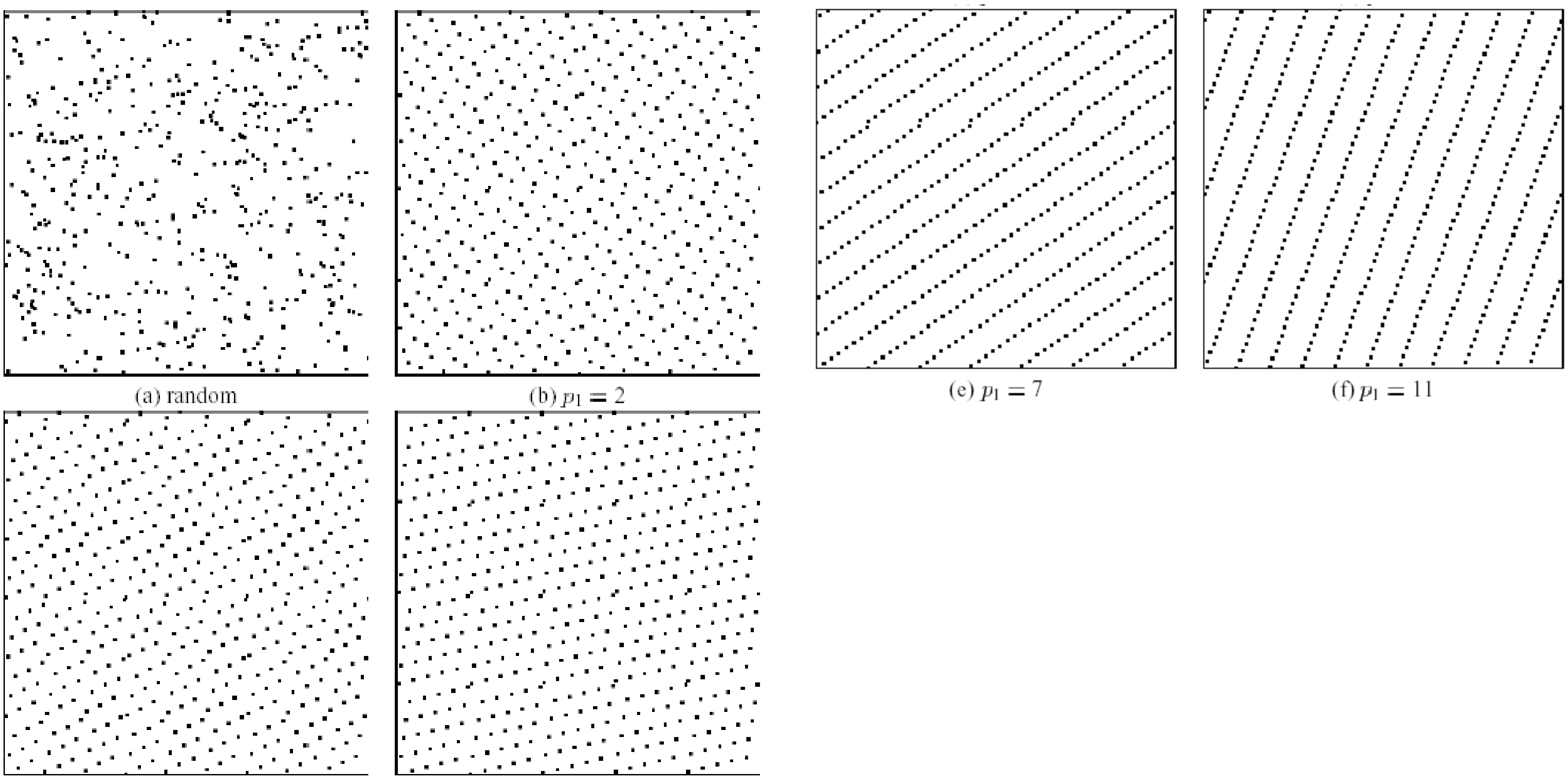
$$D_N^*(P) = O\left(\frac{(\log N)^d}{N}\right)$$

Hammersley Sequences

- Similar to Halton
- Need to know total number of samples in advance
- Better discrepancy than Halton

$$x_i = (\frac{i - 1/2}{N}, \Phi_{b_1}(i), \Phi_{b_2}(i), \ldots, \Phi_{b_{d-1}}(i))$$

Hammersley Sequences



(a) random

(b) $p_1 = 2$

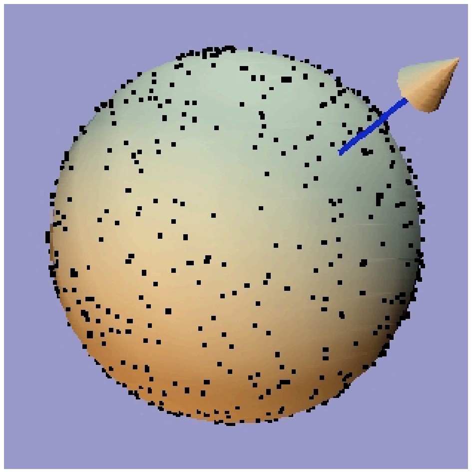
(c) $p_1 = 3$

(d) $p_1 = 5$

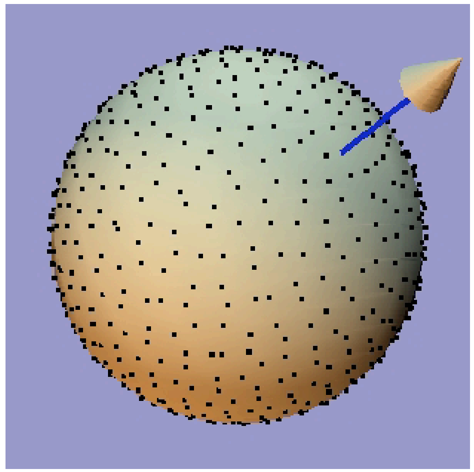
(e) $p_1 = 7$

(f) $p_1 = 11$

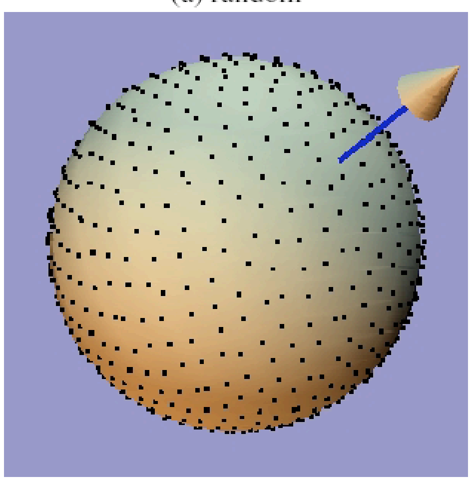
Hammersley Sequences



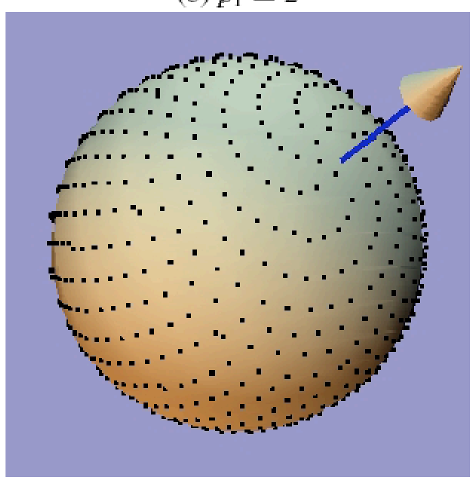
(a) random



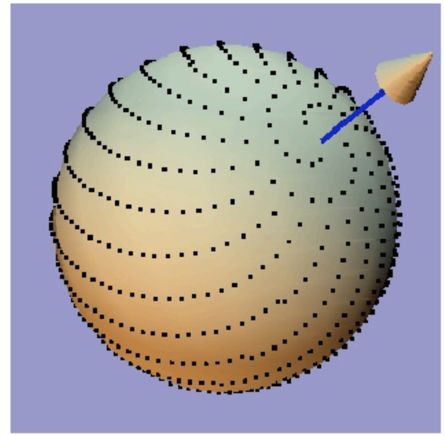
(b) $p_1 = 2$



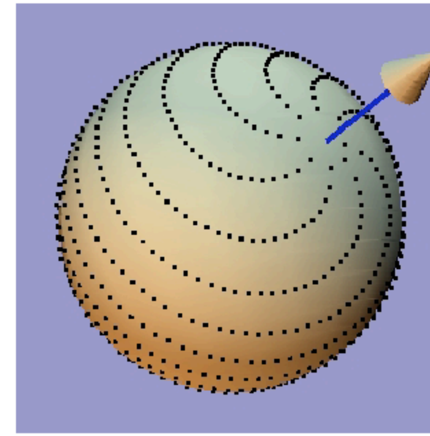
(c) $p_1 = 3$



(d) $p_1 = 5$



(e) $p_1 = 7$



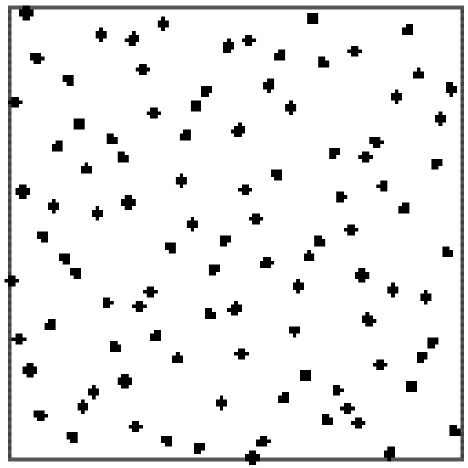
(f) $p_1 = 11$

Folded Radical Inverse

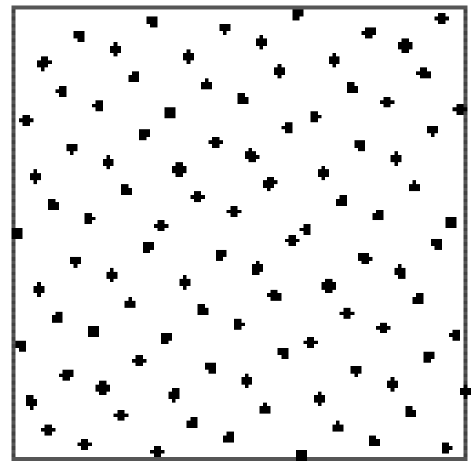
- Hammersley-Zaremba
- Halton-Zaremba
- Improves discrepancy

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

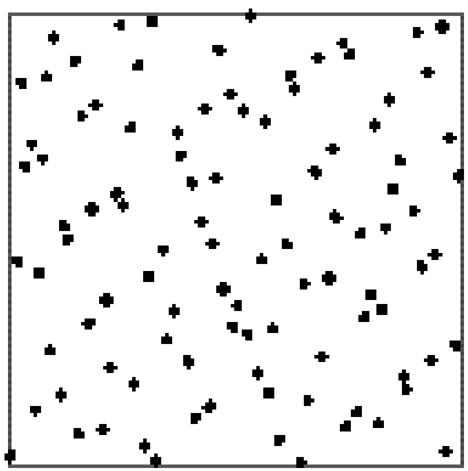
Examples



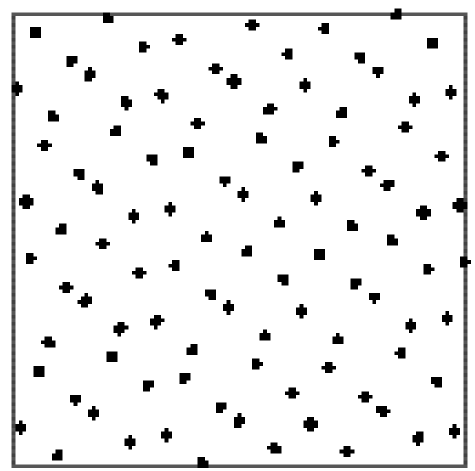
Halton, Radical Inverse



Hammersley, Radical Inverse



Halton, Folded Radical Inverse



Hammersley, Folded Radical Inverse

(t,m,d) nets

- The most successful constructions of low-discrepancy sequences are based on (t,m,d)-nets and (t,d)-sequences.
- Basis b; $0 \le t \le m$
- Is a point set in $[0,1]^d$ consisting of b^m points, such that every box

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

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

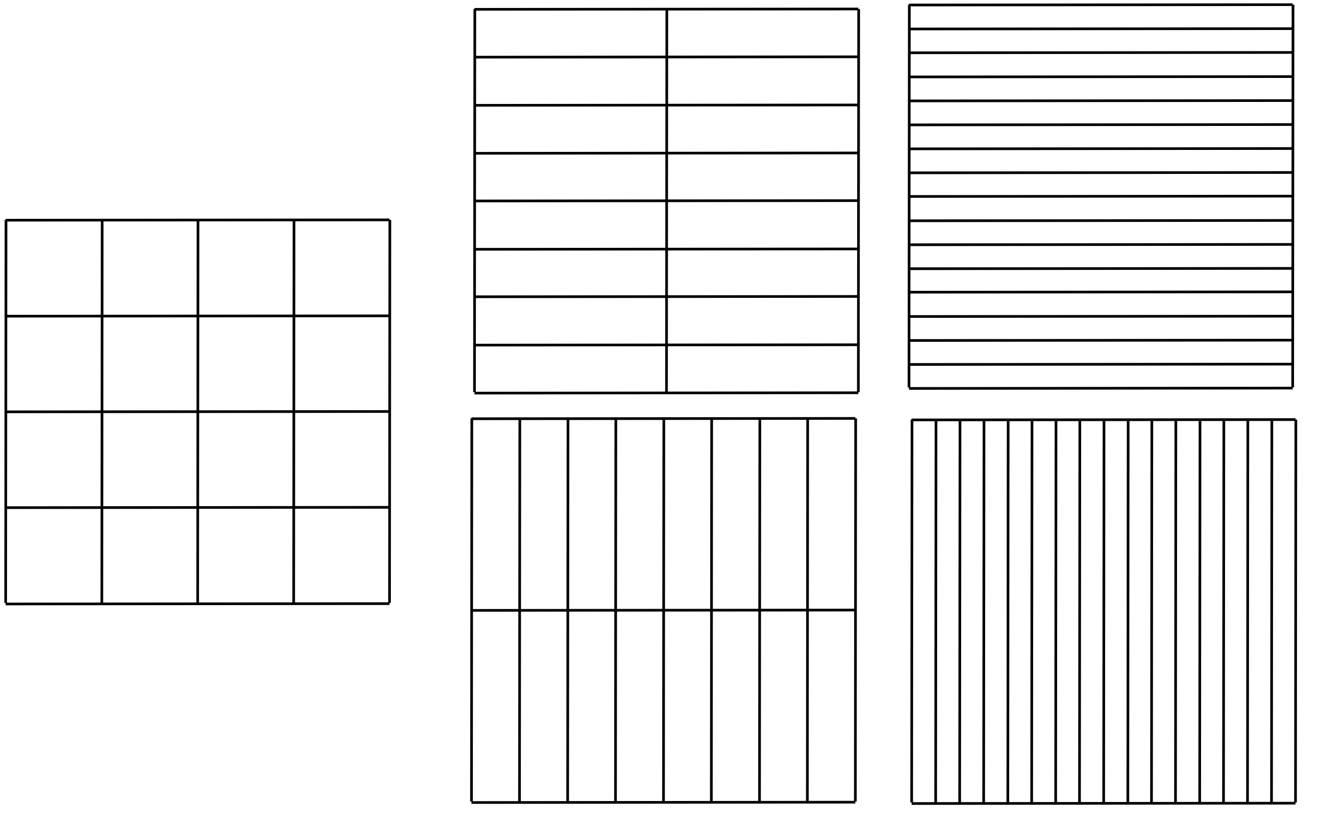
(t,d) Sequences

- (t,m,d)-Nets ensures, that all samples are uniformly distributed for any integer subdivision of our space.
- (t,d)-sequence is a sequence xi of points in $[0,1]^d$ such that for all integers $0 \leq k$ and m>t, the point set $\{x_n | kb^m \leq n < (k+1)b^m\}$

  is a (t,m,d)-net in base b.
- The number t is the quality parameter. Smaller values of t yield more uniform nets and sequences because b-ary boxes of smaller volume still contain points.

(0,2) Sequences

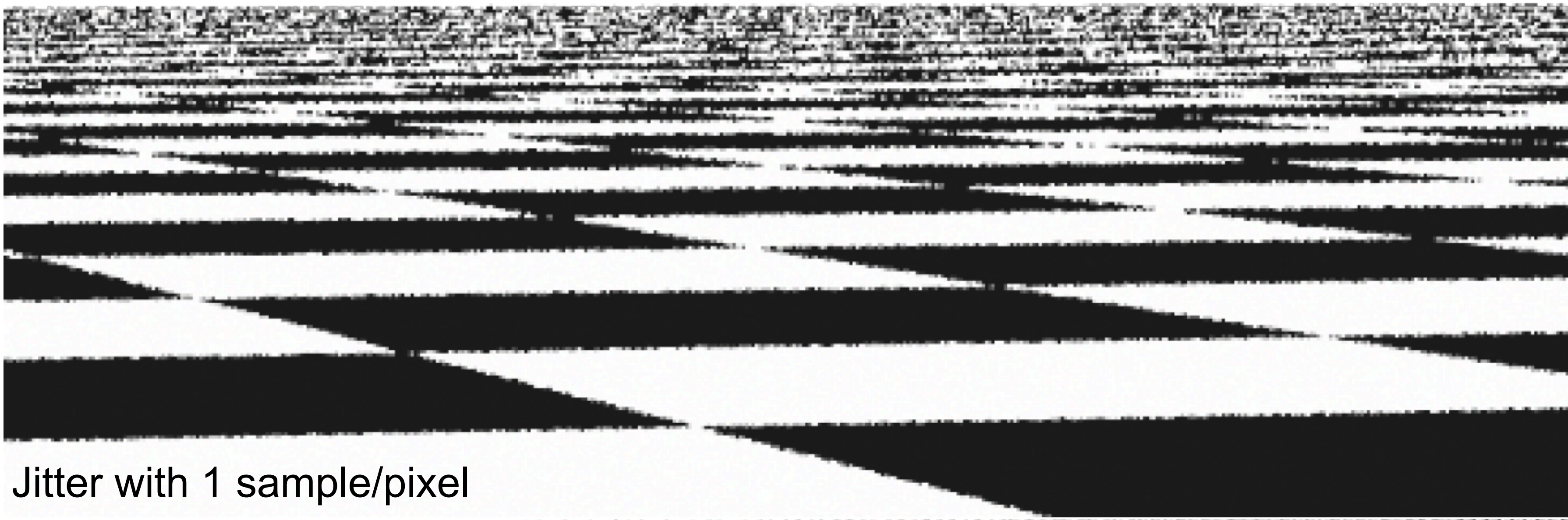
- Used in pbrt for the Low-discrepancy sampler
- Base 2



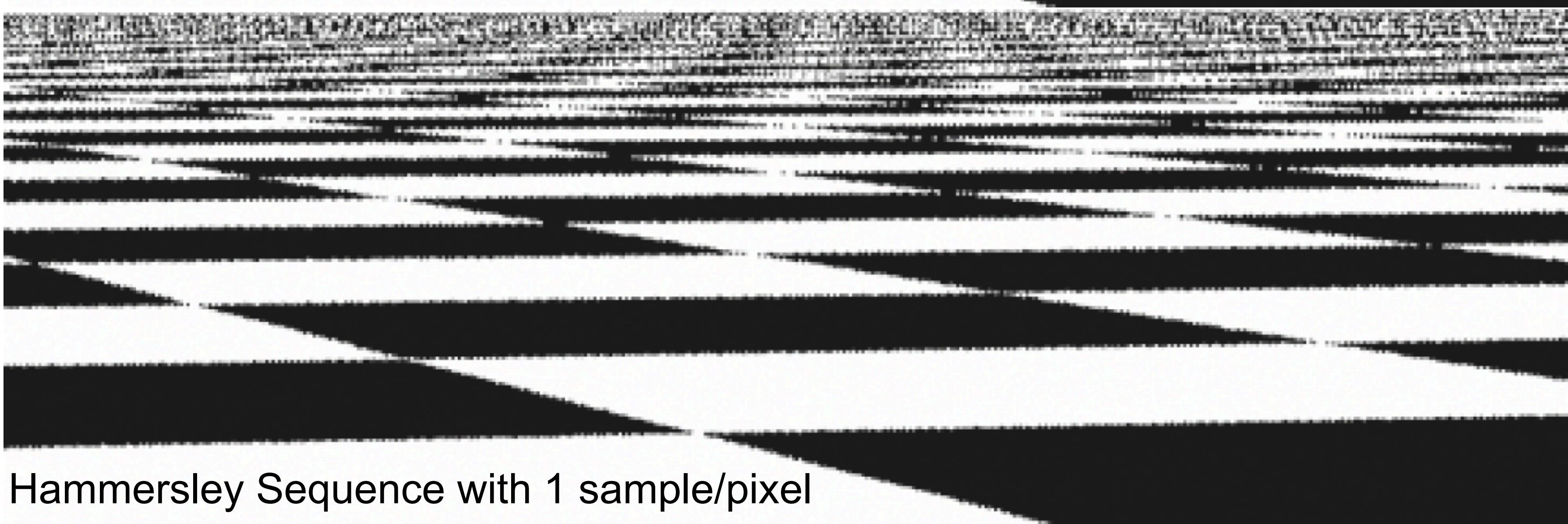
Practical Issues

- 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



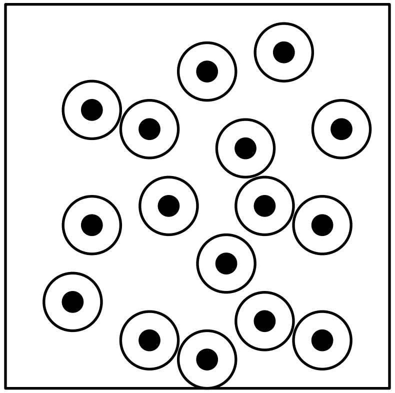
Hammersley Sequence with 1 sample/pixel

Best-Candidate Sampling

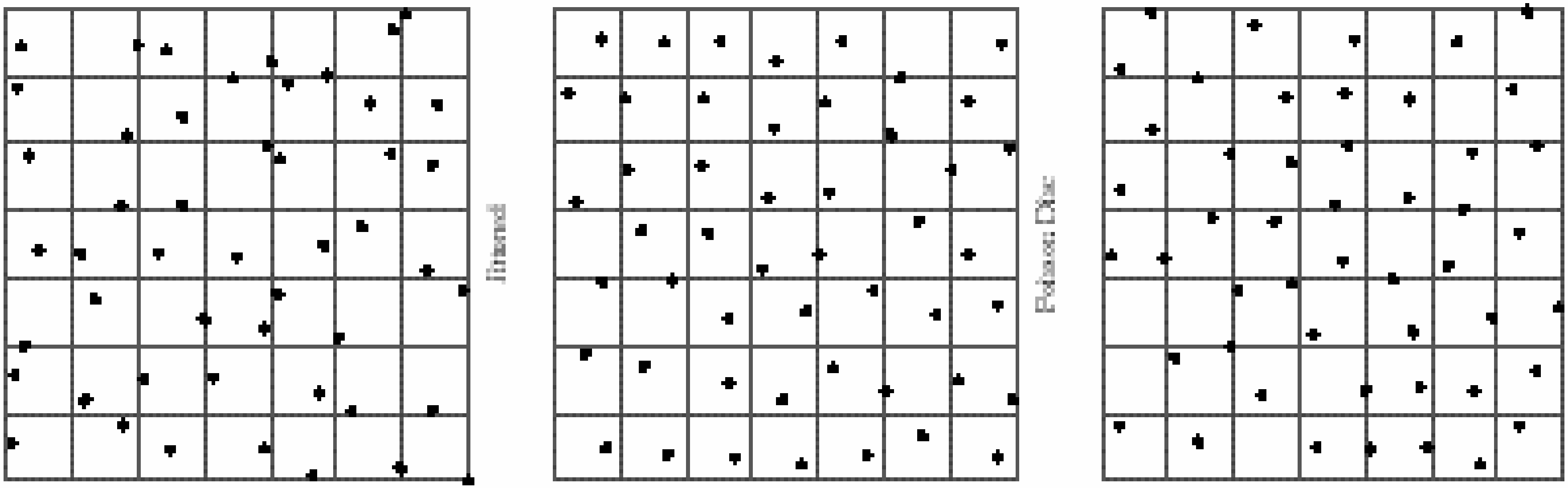
- Jittered stratification
  - Randomness (inefficient)
  - Clustering problems
  - Undersampling ("holes")
- Low Discrepancy Sequences
  - Still (visibly) aliased
- "Ideal": Poisson disk distribution
  - too computationally expensive
- Best Sampling - approximation to Poisson disk

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
  - Compute pattern which is reused by tiling the image plane (translating and scaling).
  - Toroidal topology
  - Effects the distance between points on top to bottom



Best-Candidate Sampling



Jittere
d

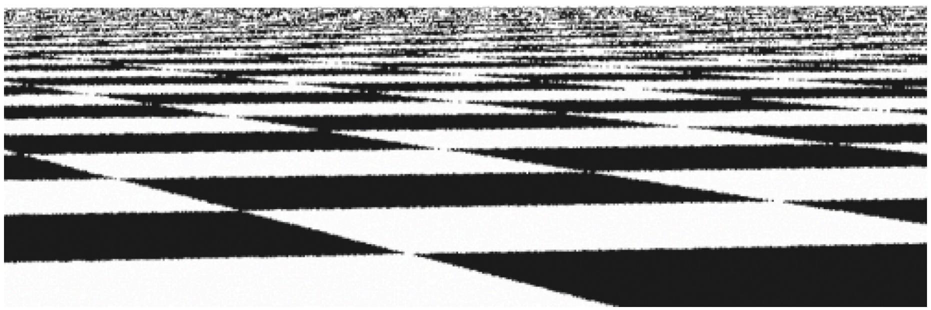
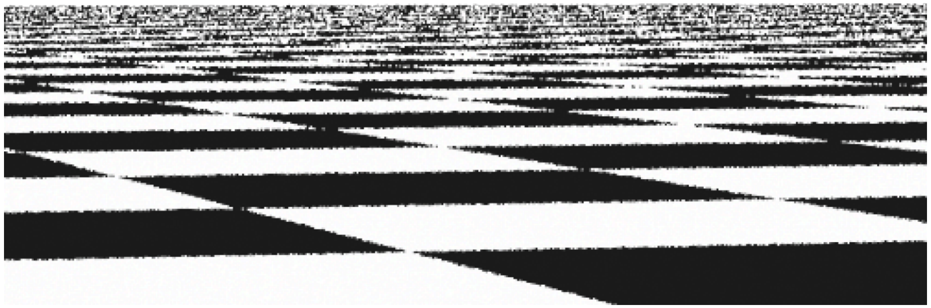
Poisson Disk

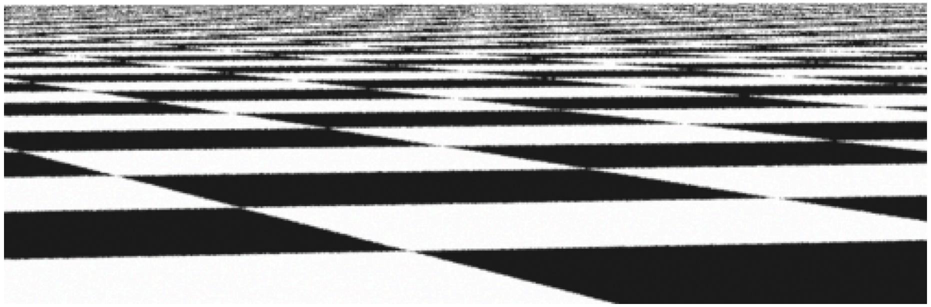
Best Candidate

Best-Candidate Sampling

```
i ← 0
while i < N
  x_i ← unit()                          Throw a dart.
  y_i ← unit()
  reject ← false
  for k ← 0 to i − 1                    Check the distance to all other samples.
    d ← (x_i − x_k)^2 + (y_i − y_k)^2
    if d < (2r_p)^2 then
      reject ← true                     This one is too close—forget it.
      break
      endif
    endfor
  if not reject then
    i ← i + 1                           Append this one to the pattern.
    endif
endwhile
```

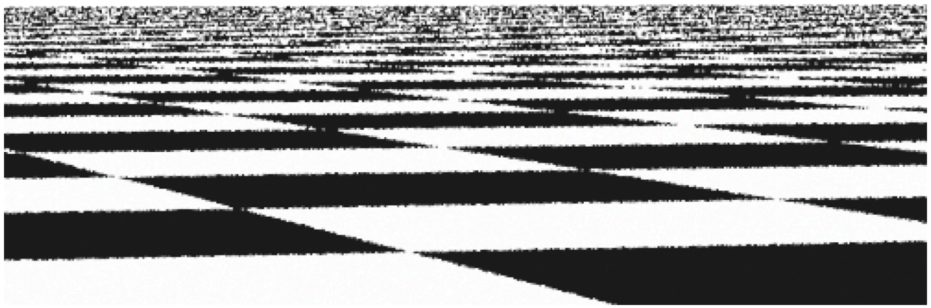
Texture

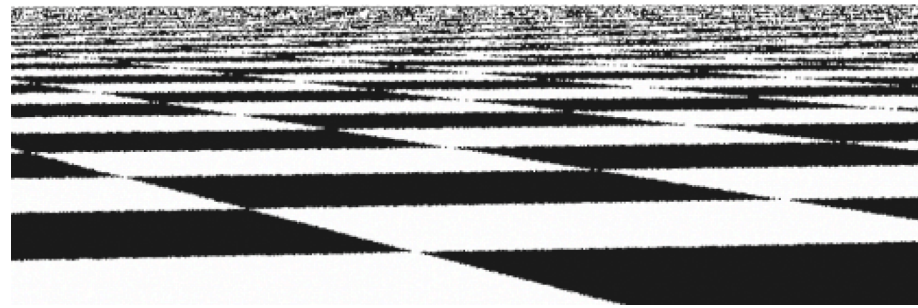


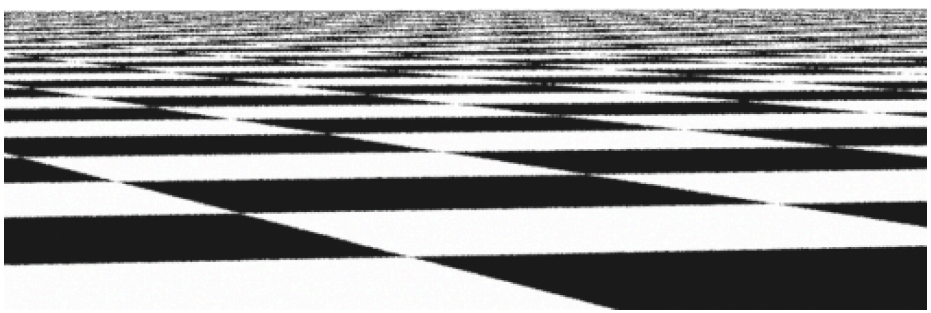
Texture

Jitter with 1 sample/pixel



Best Candidate with 1 sample/pixel





Jitter with 4 sample/pixel

Best Candidate with 4 sample/pixel

Next

- Probability Theory
- Monte Carlo Techniques
- Rendering Equation