X39RSO/A4M39RSO

Sampling & Anti-aliasing

Vlastimil Havran
ČVUT v Praze – CTU Prague
Verze 2011

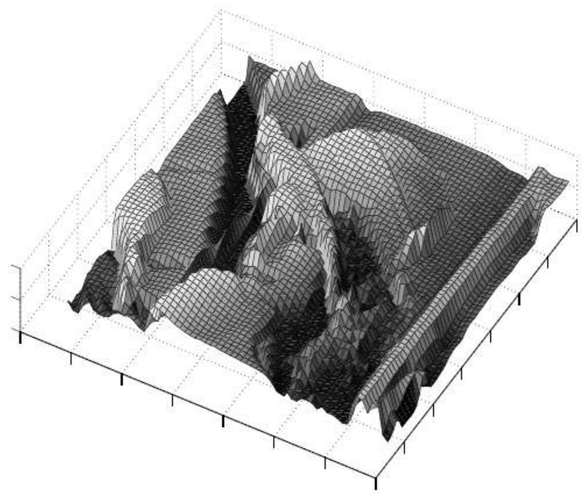
Prepared orginally by Daniel Sýkora 2006

Introduction

What Is an (Animated) Image?

- **Theory** - Continuous function of two/three variables:

$$I(x,y,t): \mathbf{R}^3 \text{ -> } \mathbf{R}$$



- **Practice** – Time varying matrix of pixels
- One animation frame = equidistant <u>samples</u> of $I(x,y,t_i)$:

$$I[x,y,t]: \mathbf{N}^2 \text{ -> } \mathbf{R}$$

$I[x,y,t]$ **=>** luminance, color RGB, etc.

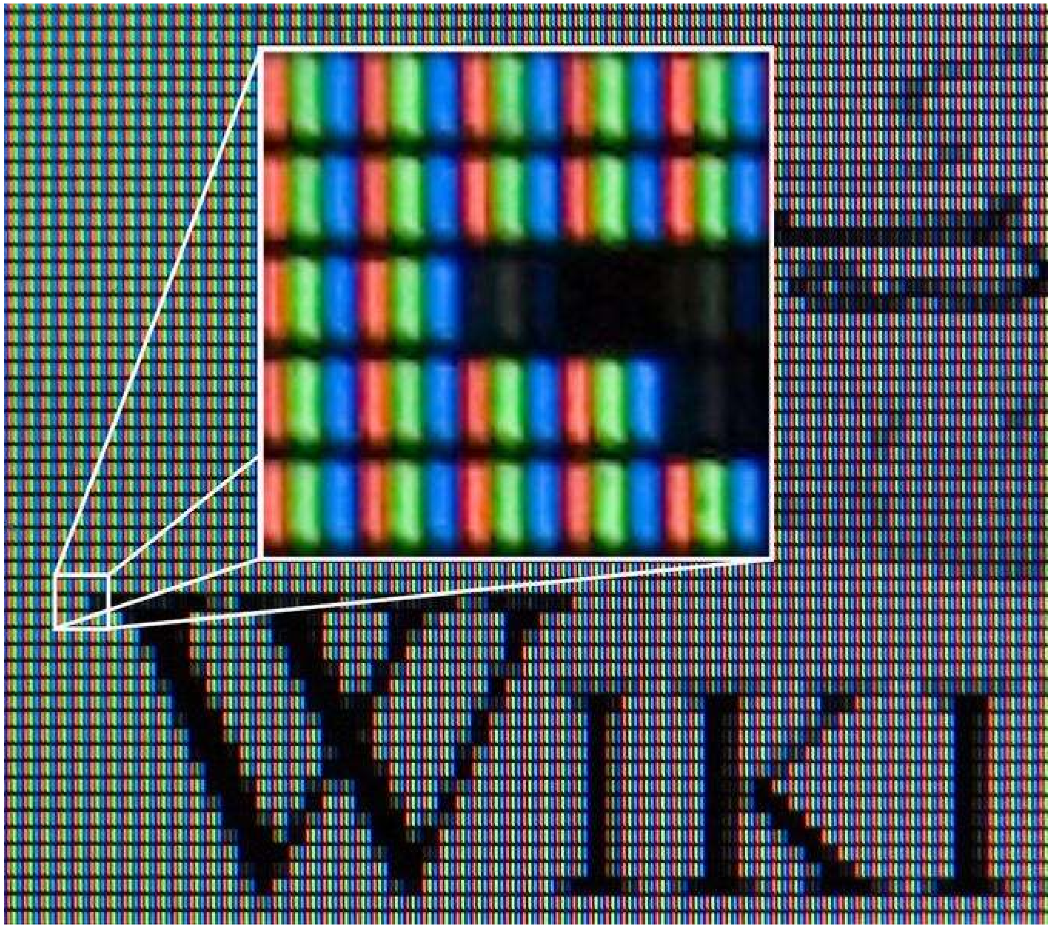
Imagers = Signal Sampling

- **Physical imagers** (imager = imaging device)
  - Integrate over sensor area and time
  - Integration: each photon hit => increase pixel value
  - Examples
    - Retina – rods / cones
    - CCD array
    - Film
- **Virtual imagers** – computer graphics cameras
  - Sample continuous image function at specific location and time instant
  - No integration takes place – has to be simulated

Displays = Signal Reconstruction

- Take image samples
- Reconstruct continuous image (one pixel = small light)

- Examples
  - CRT
  - LCD



from Wikipedia

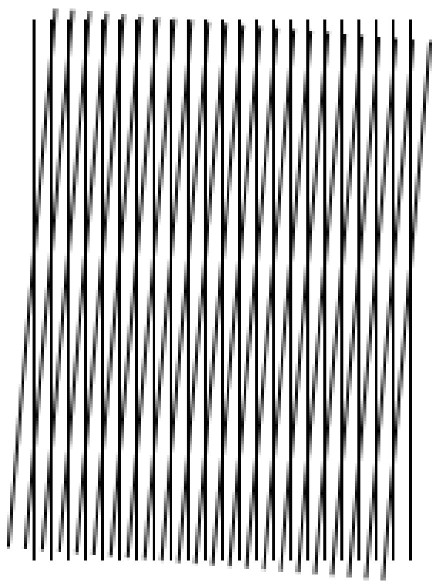
Image Sampling

- **Real devices**
  - Sample = Integral over (small) area and (short) time
- **Rendering**
  - Sample = Point sample
  - Consequences – Aliasing
    - Jagged edges
    - Moire patterns
    - Flickering of small objects
    - Sparkling highlights
    - Temporal flickering
  - Preventing aliasing – anti-aliasing

Moire Patterns

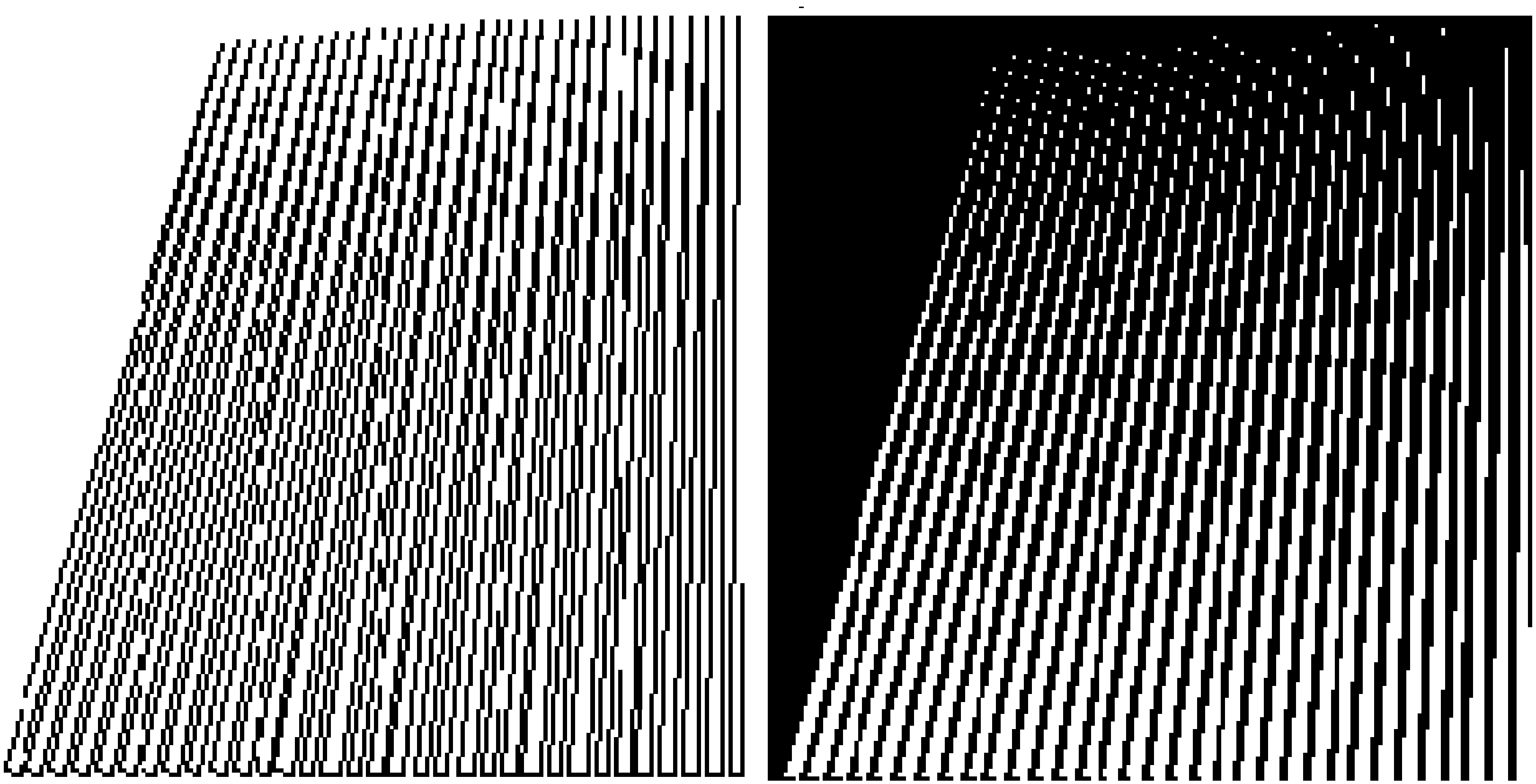
- **Definition from Wikipedia**

  A moiré pattern is an interference pattern created, for example, when two grids are overlaid at an angle, or when they have slightly different mesh sizes.



A moiré pattern, formed by two sets of parallel lines, one set inclined at an angle of 5° to the other.

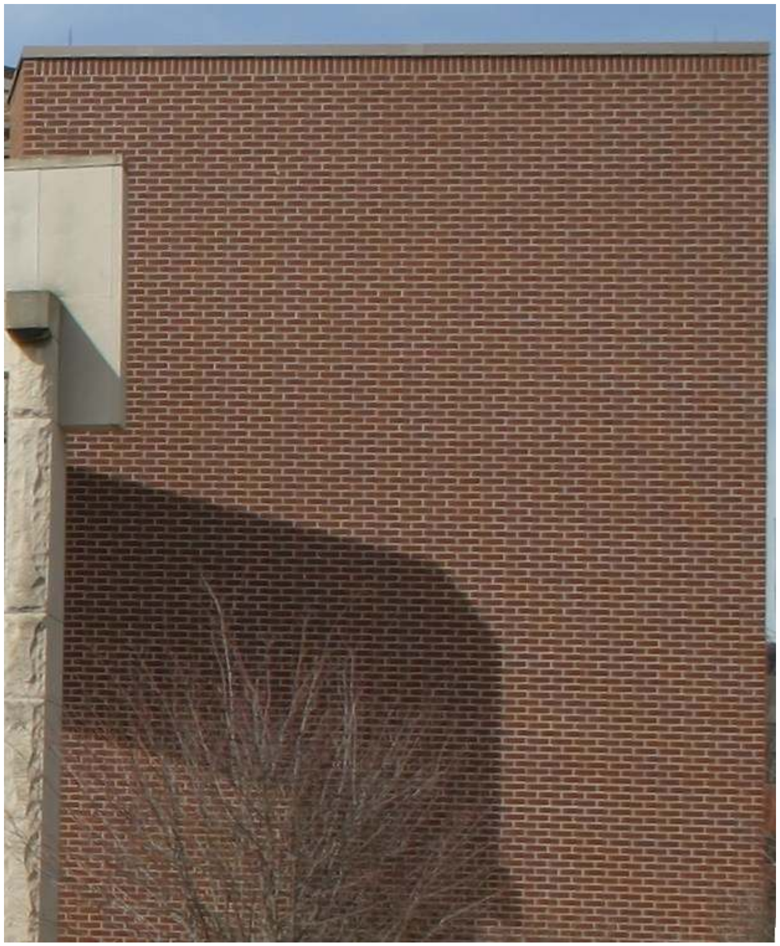
Another Moire Pattern (by Chris Cooksey)



From Wikipedia

Moire Patterns

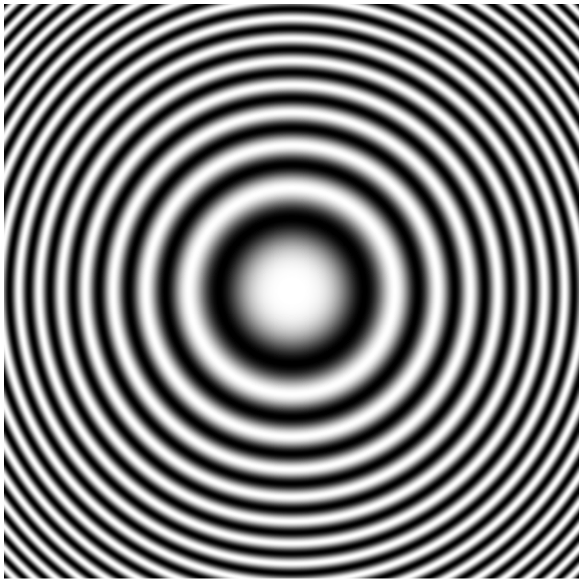
Moire Patterns

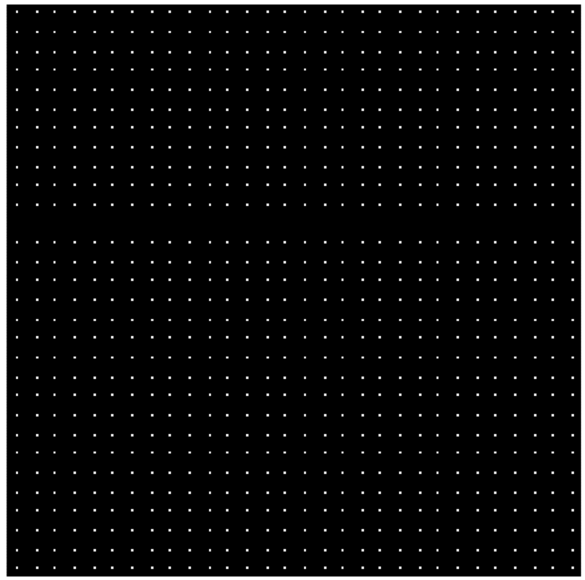


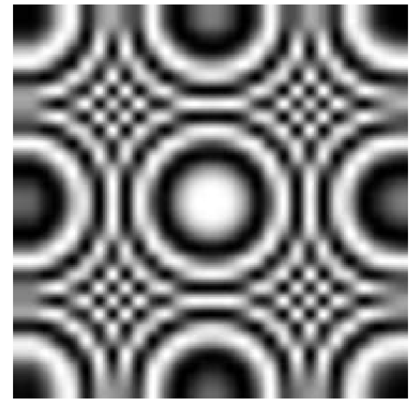
Original Image

Improperly subsampled image

Moire Patterns







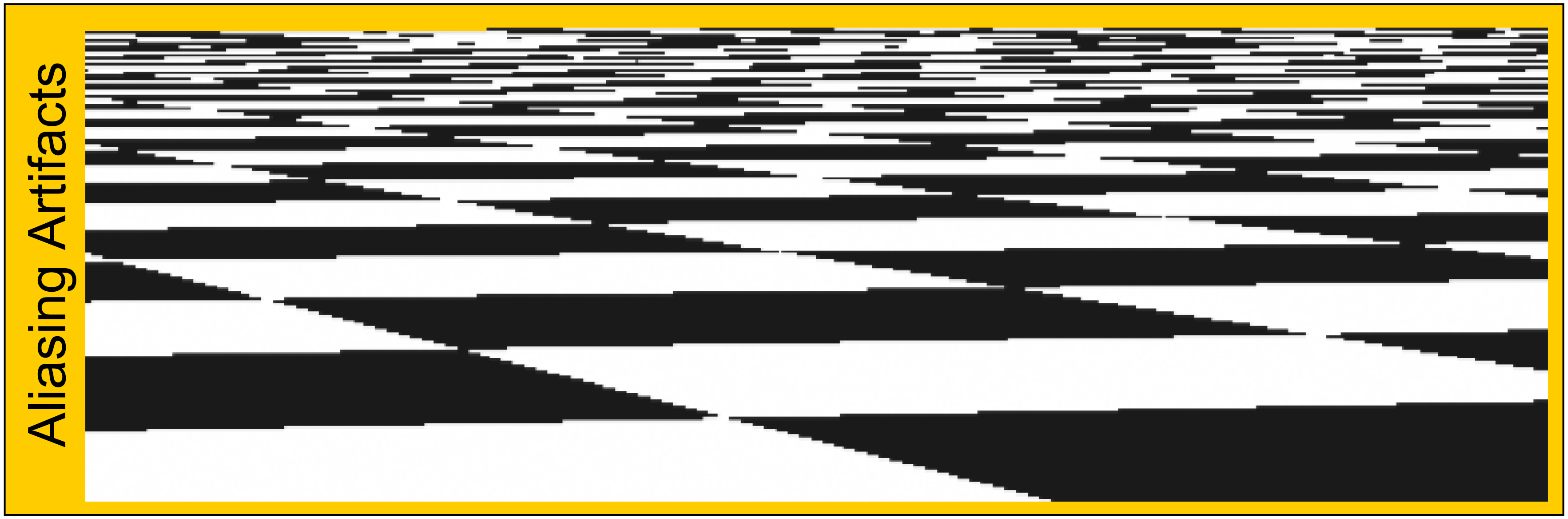
Continuous image
("Zone plate")

$\sin x^2 + y^2$

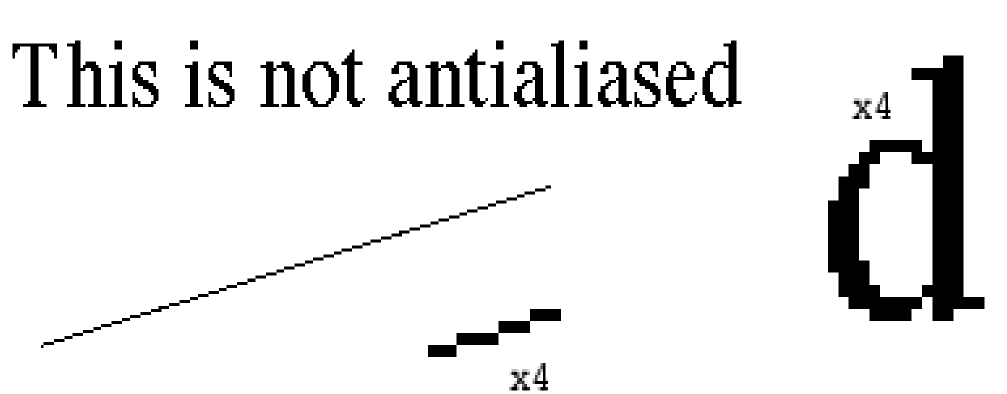
Image samples

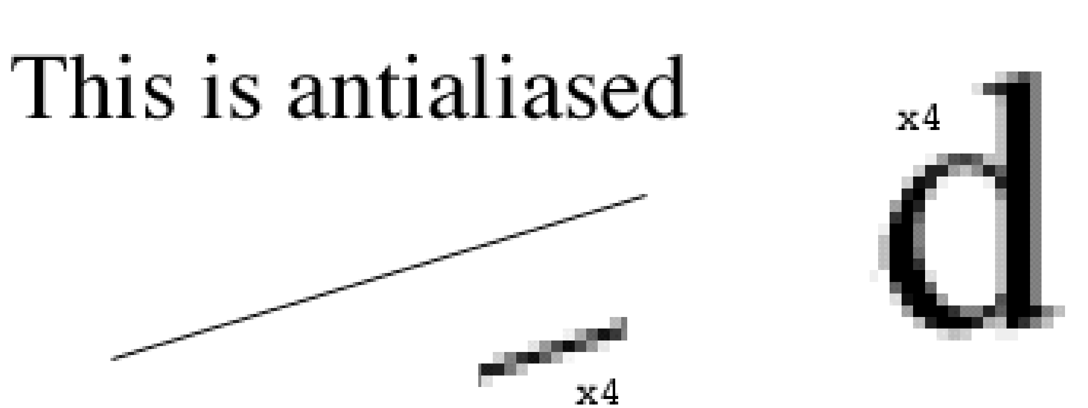
Reconstructed image

Texture Aliasing



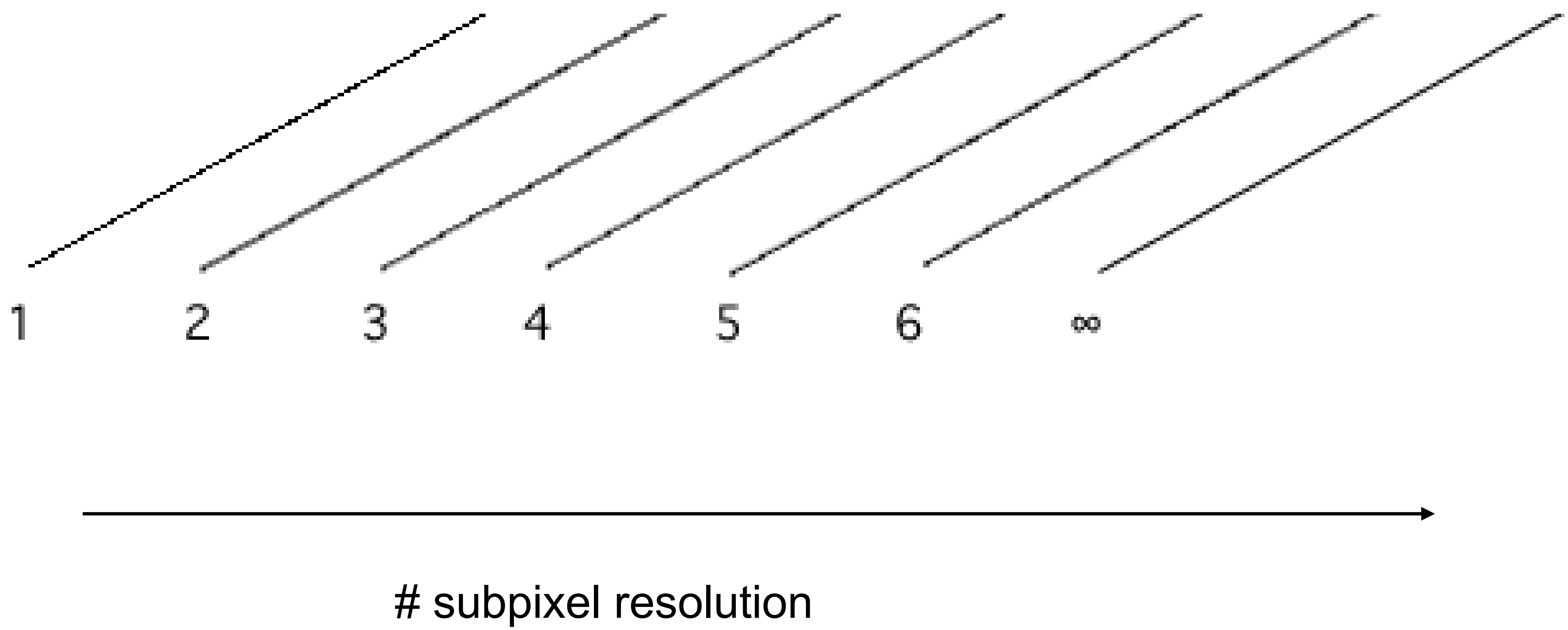
Character antialiasing





Line antialiasing

- Subpixel sampling is not sufficient !



Fourier Transform & Convolution

Fourier Transform

- Any function = weighted sum of sines & cosines
- Fourier transform computes weights for sines / cosines of different frequencies (or sine + phase shift)
- Spectrum of *f* = Image of *f* after Fourier transform

Fourier Transform

$$\textbf{time}\{x\} \quad \Longleftrightarrow \quad \textbf{frequency}\{u\}$$

$$\text{forward:} \quad F(u) = \int_{-\infty}^{\infty} f(x) \cdot e^{-2\pi jux} \mathrm{d}x$$

$$\text{inverse:} \quad f(x) = \int_{-\infty}^{\infty} F(u) \cdot e^{+2\pi jux} \mathrm{d}u$$

$$\text{basis functions:} \quad e^{j\alpha} = \cos(\alpha) + j\sin(\alpha)$$

$$\text{cosine transform:} \quad \textbf{Re}(F(u)) = \int_{-\infty}^{\infty} F(x) \cdot \cos(2\pi ux) \mathrm{d}x$$

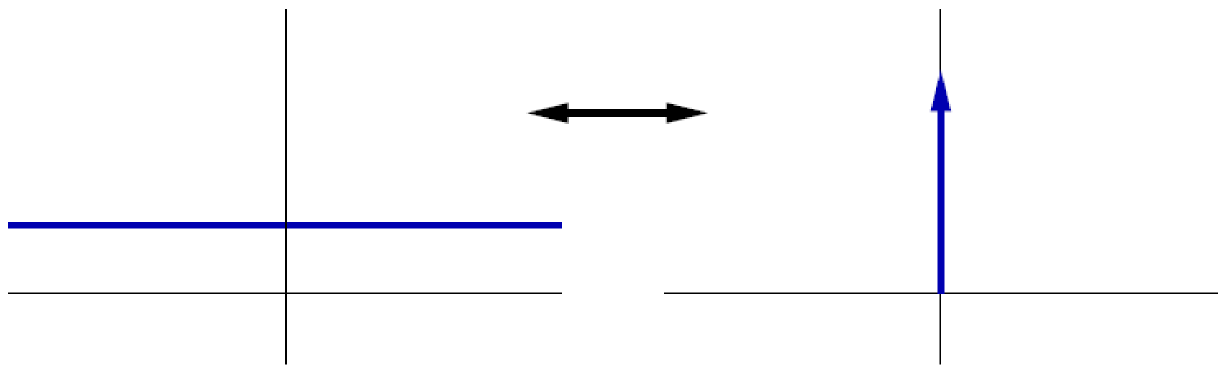
$$\text{sine transform:} \quad \textbf{Im}(F(u)) = -\int_{-\infty}^{\infty} F(x) \cdot \sin(2\pi ux) \mathrm{d}x$$

$$\text{amplitude:} \quad |F(u)| = \sqrt{\textbf{Re}(F(u))^2 + \textbf{Im}(F(u))^2}$$

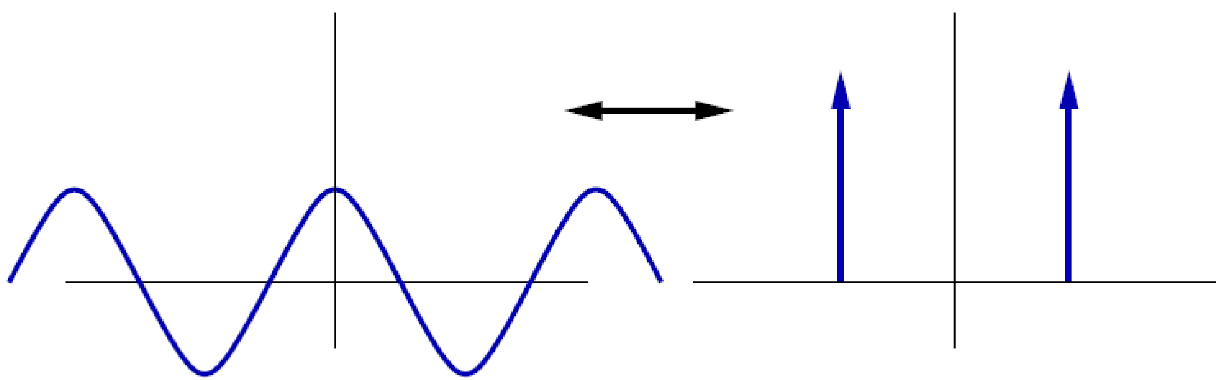
$$\text{phase:} \quad \Phi(F(u)) = \tan^{-1}\left(\frac{\textbf{Im}(F(u))}{\textbf{Re}(F(u))}\right)$$

Fourier Transform Pairs (Examples)

$$1 \iff \delta(u)$$

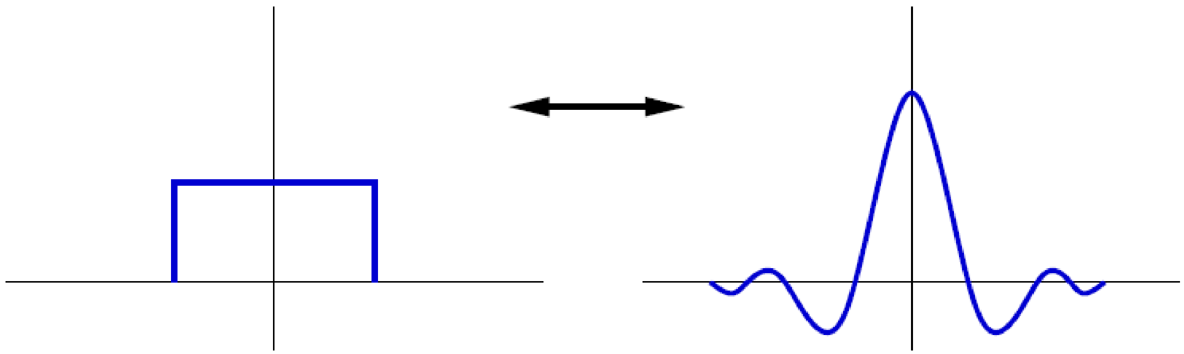


$$\cos(2\pi kx) \iff \frac{1}{2}(\delta(u+k)+\delta(u-k))$$

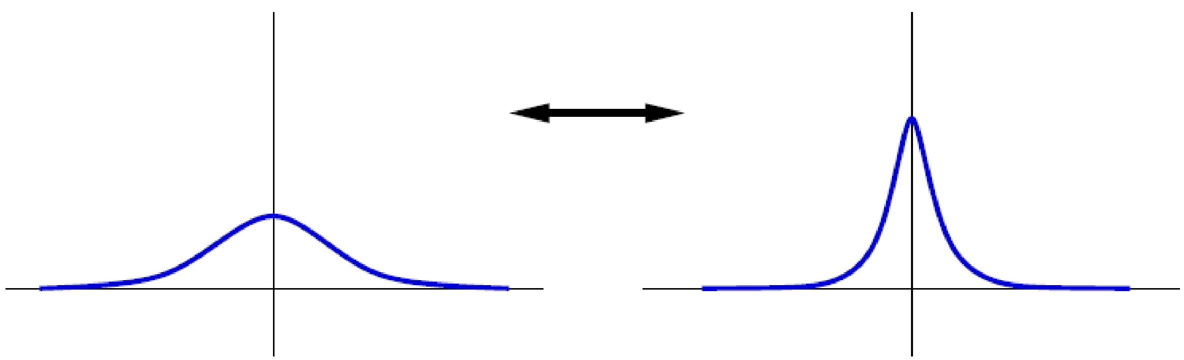


Fourier Transform Pairs (Examples)

$$\mathbf{1}(x+k) - \mathbf{1}(x-k) \iff \frac{1}{\pi u}\sin(2\pi k u)$$



$$\exp\left(-kx^2\right) \iff \sqrt{\frac{\pi}{k}}\exp\left(-\frac{\pi^2}{k}u^2\right)$$

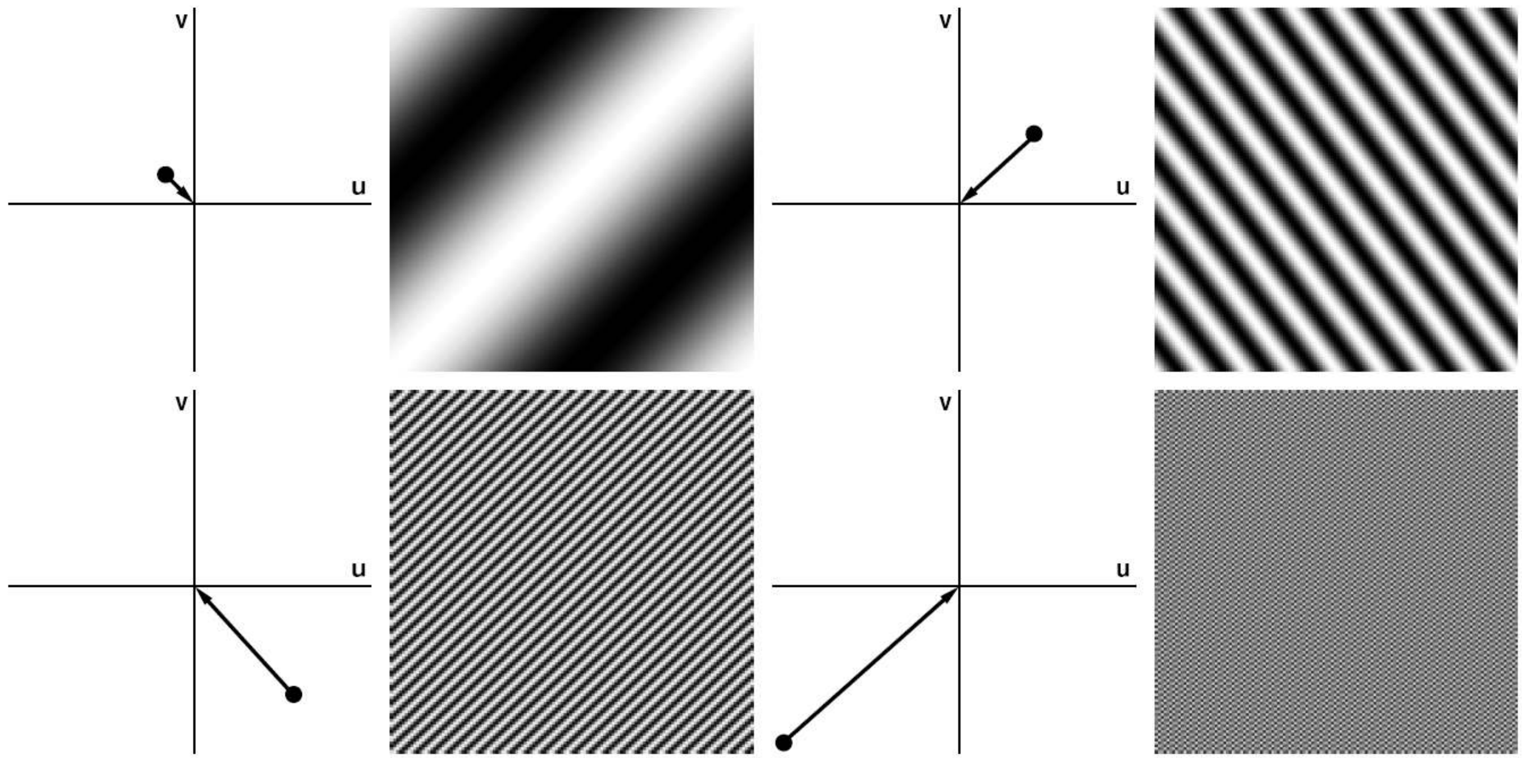


Fourier Transform in 2D

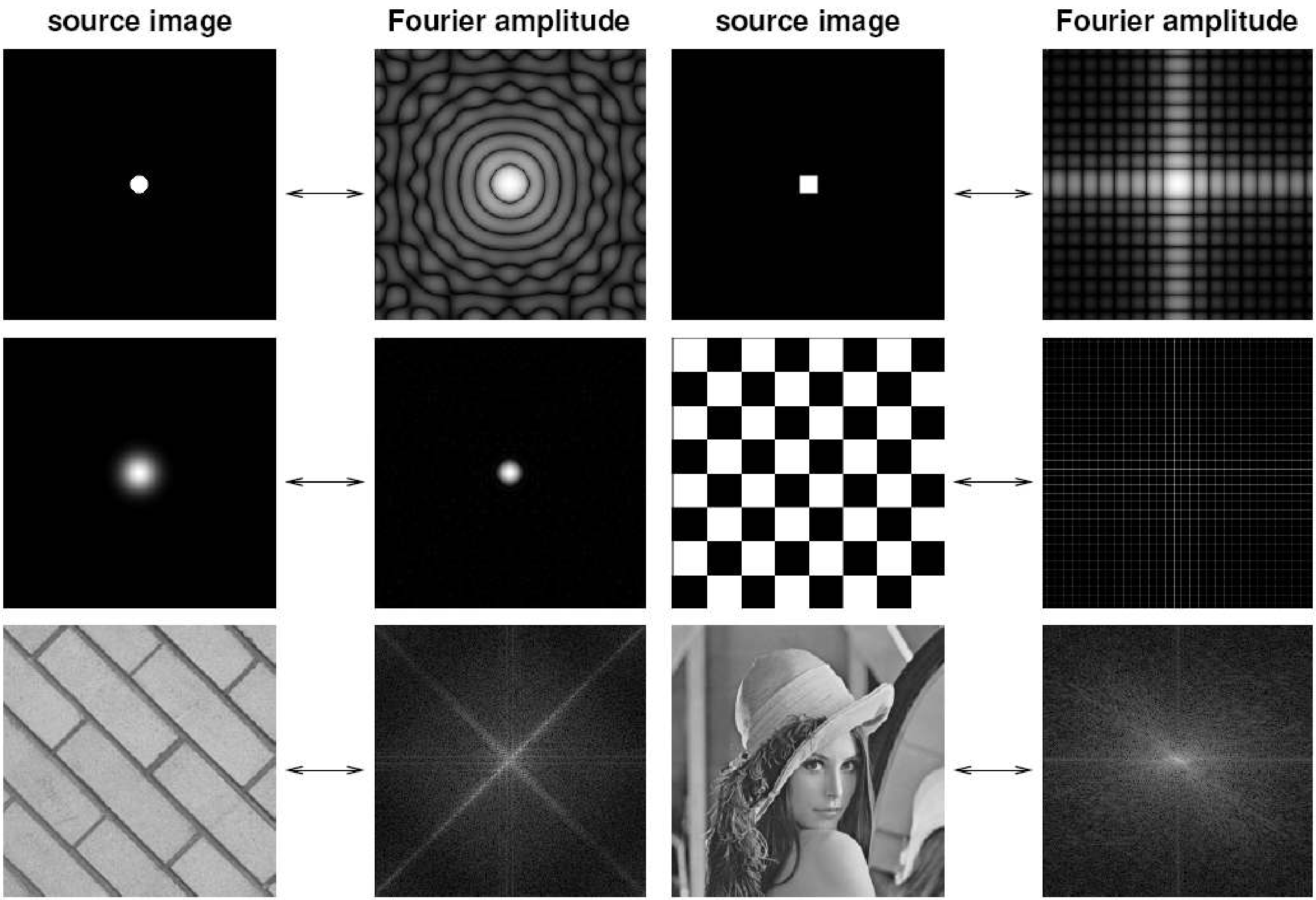
position$\{x, y\}$ $\iff$ **frequency & orientation**$\{u, v\}$

forward: $$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\cdot e^{-2\pi j(ux+vy)}\mathrm{d}x\mathrm{d}y$$

inverse: $$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)\cdot e^{+2\pi j(ux+vy)}\mathrm{d}u\mathrm{d}v$$

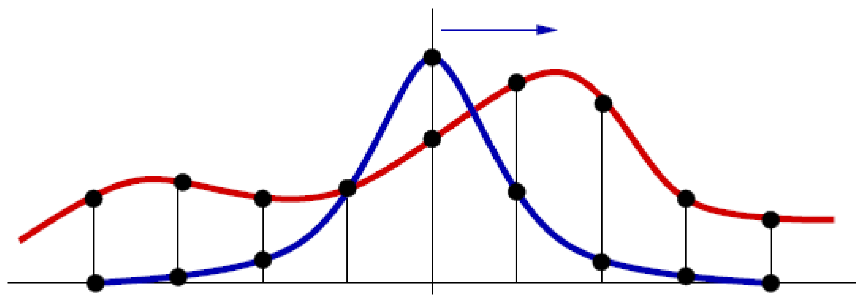


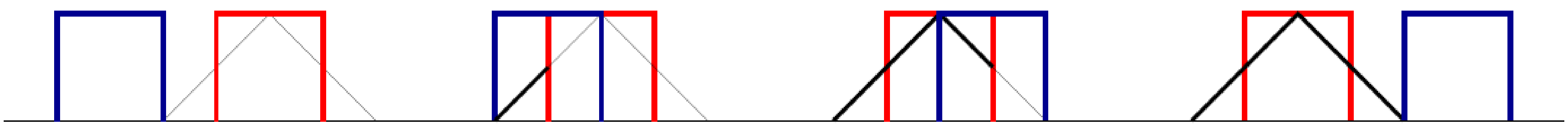
Fourier Transform in 2D – Examples



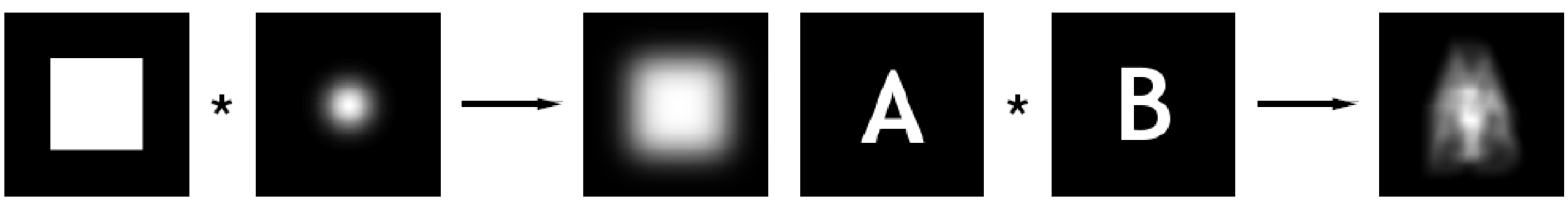
Convolution

$$h(x) = f \otimes g = \int f(x')g(x-x')\,dx'$$





In 2D



Convolution Theorem

- Multiplication in the frequency domain is equivalent to convolution in the space domain and vice versa.

$$f \otimes g \leftrightarrow F \times G$$

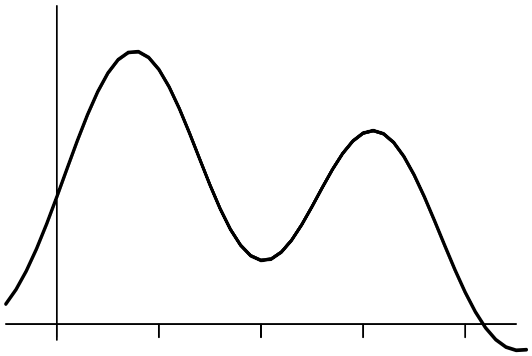
$$f \times g \leftrightarrow F \otimes G$$

Sampling & Reconstruction
The Sampling Theorem

Sampling

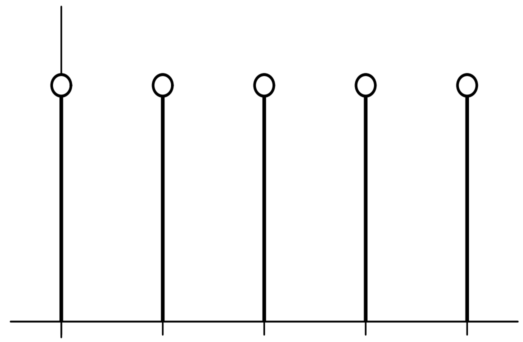
Continuous signal | Sampling (Shah) function | Sampled signal

Spatial domain



$f(x)$

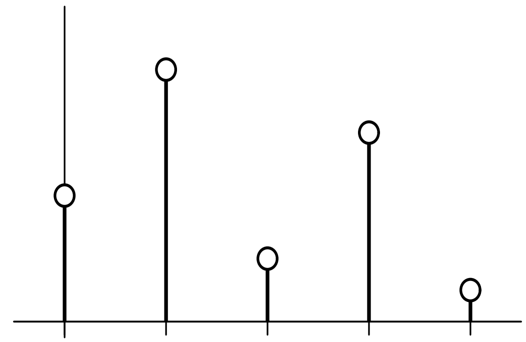
multiplication $\times$



$\mathrm{III}(x) = \sum_{-\infty}^{\infty} \delta(x - nT)$

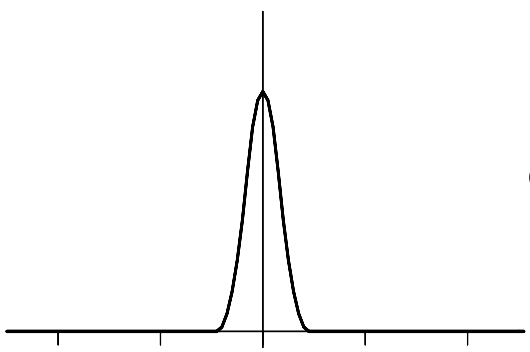
$T \ldots$ sampling period

$=$



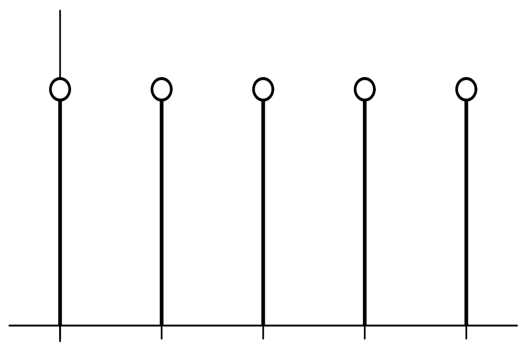
$f_s(x) = f(x) \cdot \mathrm{III}(x)$

Frequency domain



$F(u)$

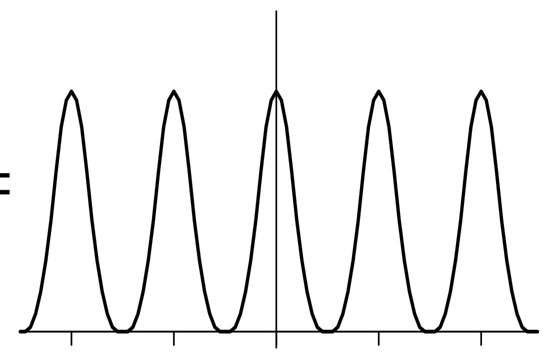
convolution $\otimes$



$\mathrm{III}(u) = \sum_{-\infty}^{\infty} \delta(u - n/T)$

$1/T \ldots$ sampling freq.

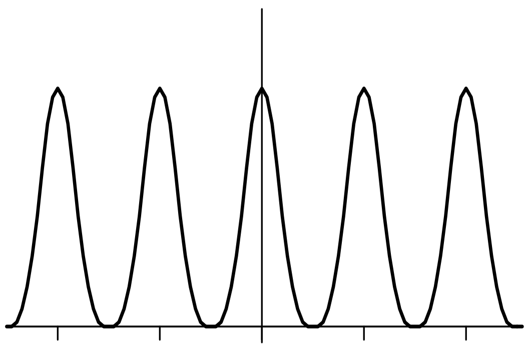
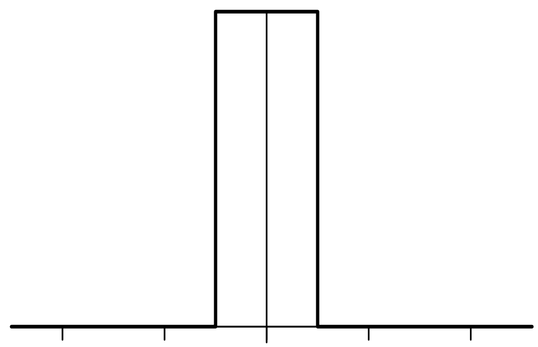
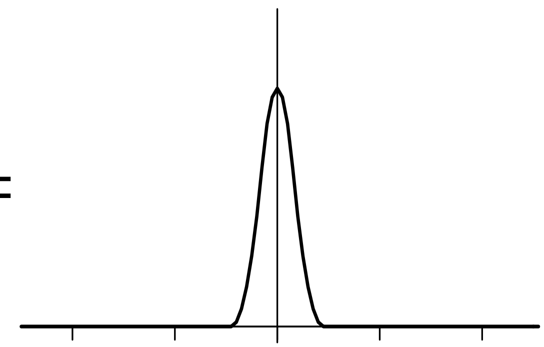
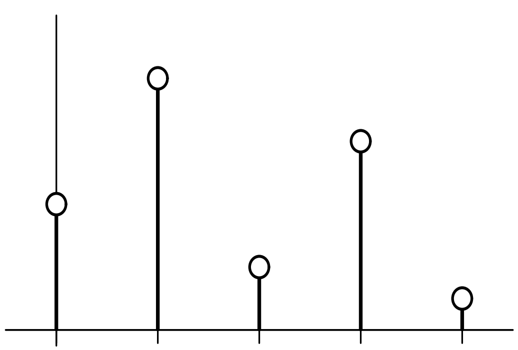
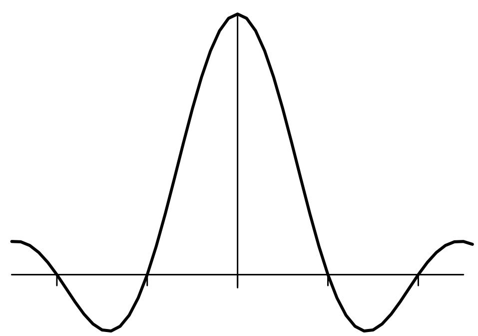
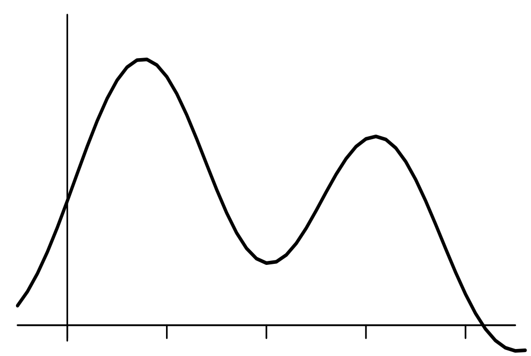
$=$



$F_s(u) = F(u) \otimes \mathrm{III}(u)$

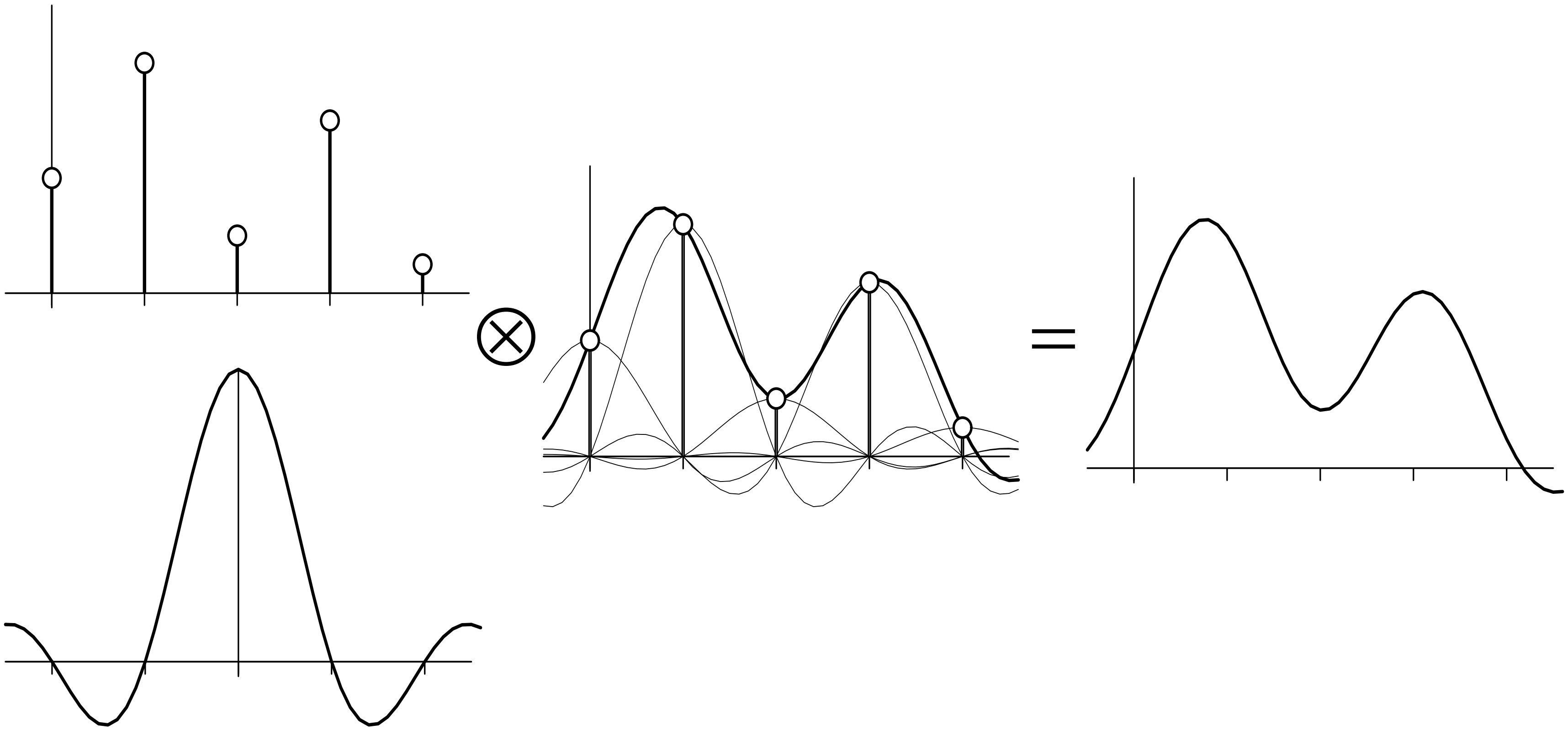
replicated & shifted spectra!

Reconstruction

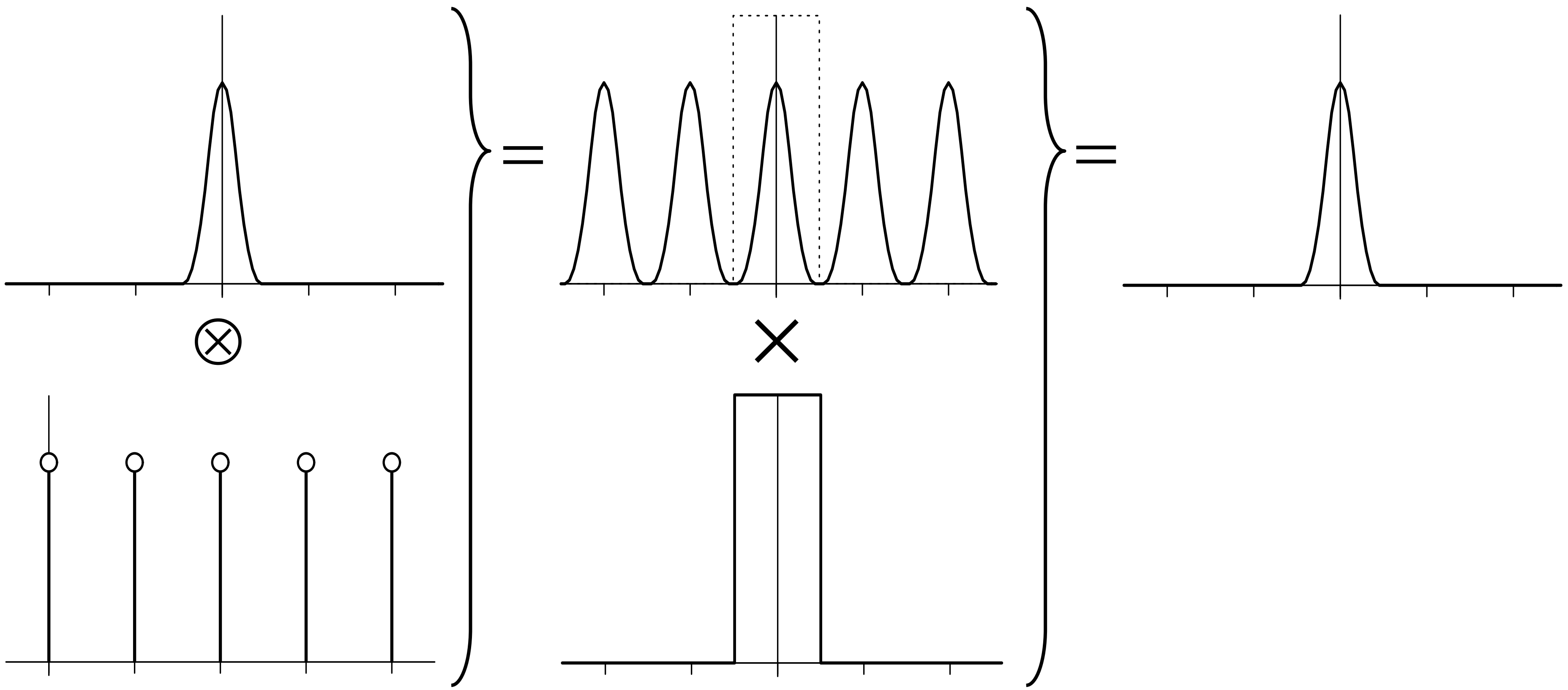
	Sampled signal		(Ideal) Reconstruction filter		Reconstructed Continuous signal
Frequency domain	 $F_s(u)$	$\times$ multiplication	 Box : $\text{II}(u) = \begin{cases} 1 & \lvert x \rvert < 1/2\,T^{-1} \\ 0 & \text{otherwise} \end{cases}$	$=$	 $F'(u) = F_s(u) \cdot \text{II}(u)$ replicas cut off
Spatial domain	 $f_s(x)$	$\otimes$ convolution	 $\text{sinc}_T(x) = \sin(xT)/xT$ $T \ldots$ sampling period	$=$	 $f'(x) = f_s(x) \otimes \text{III}(x)$

Reconstruction in Spatial Domain – Details

- **Convolution** with the sinc function
  - weighted sum of shifted sinc kernels



Sampling and Reconstruction Summary – Frequency Domain

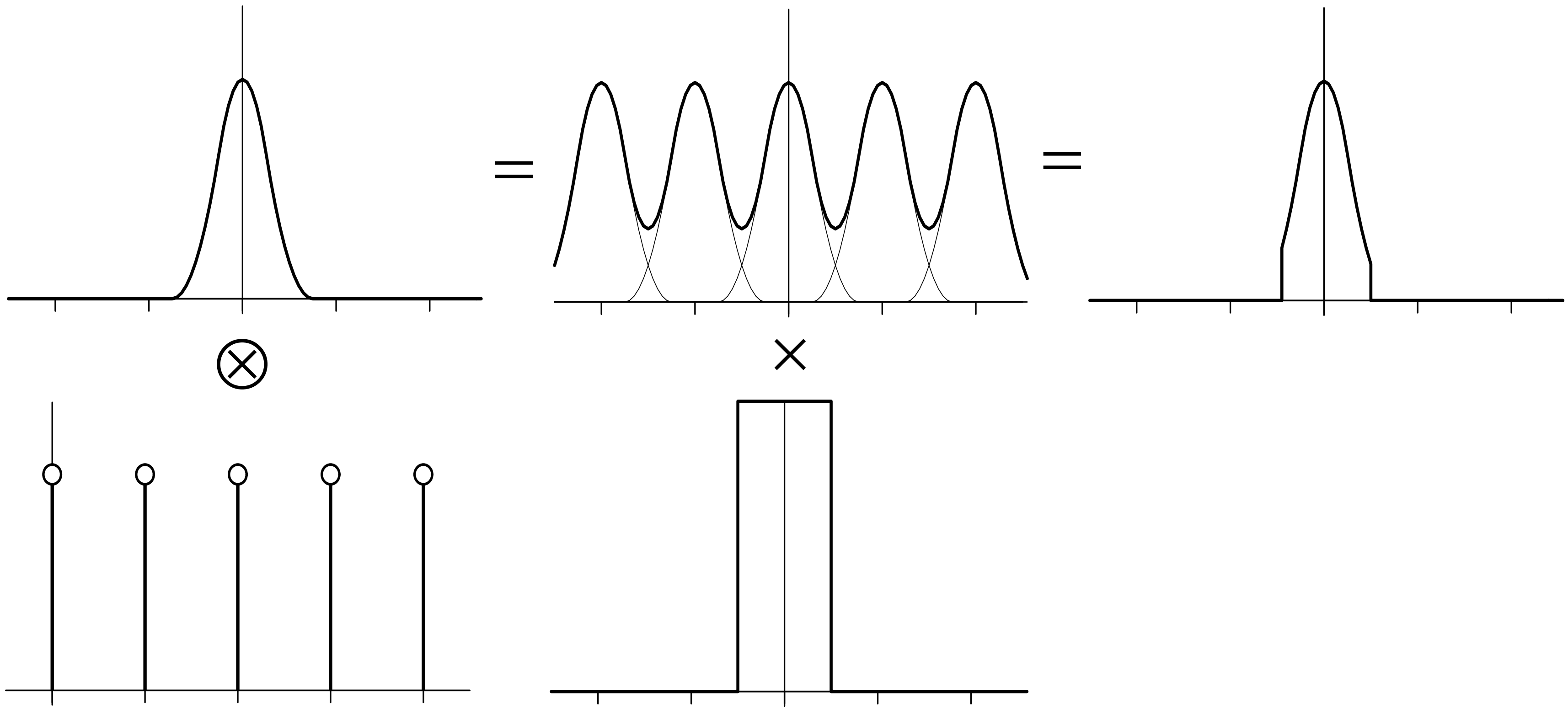


Sampling

Reconstruction

Reconstructed spectrum

Aliasing due to Undersampling



- If spectrum replicas overlap, impossible to reconstruct original signal

Sampling Theorem

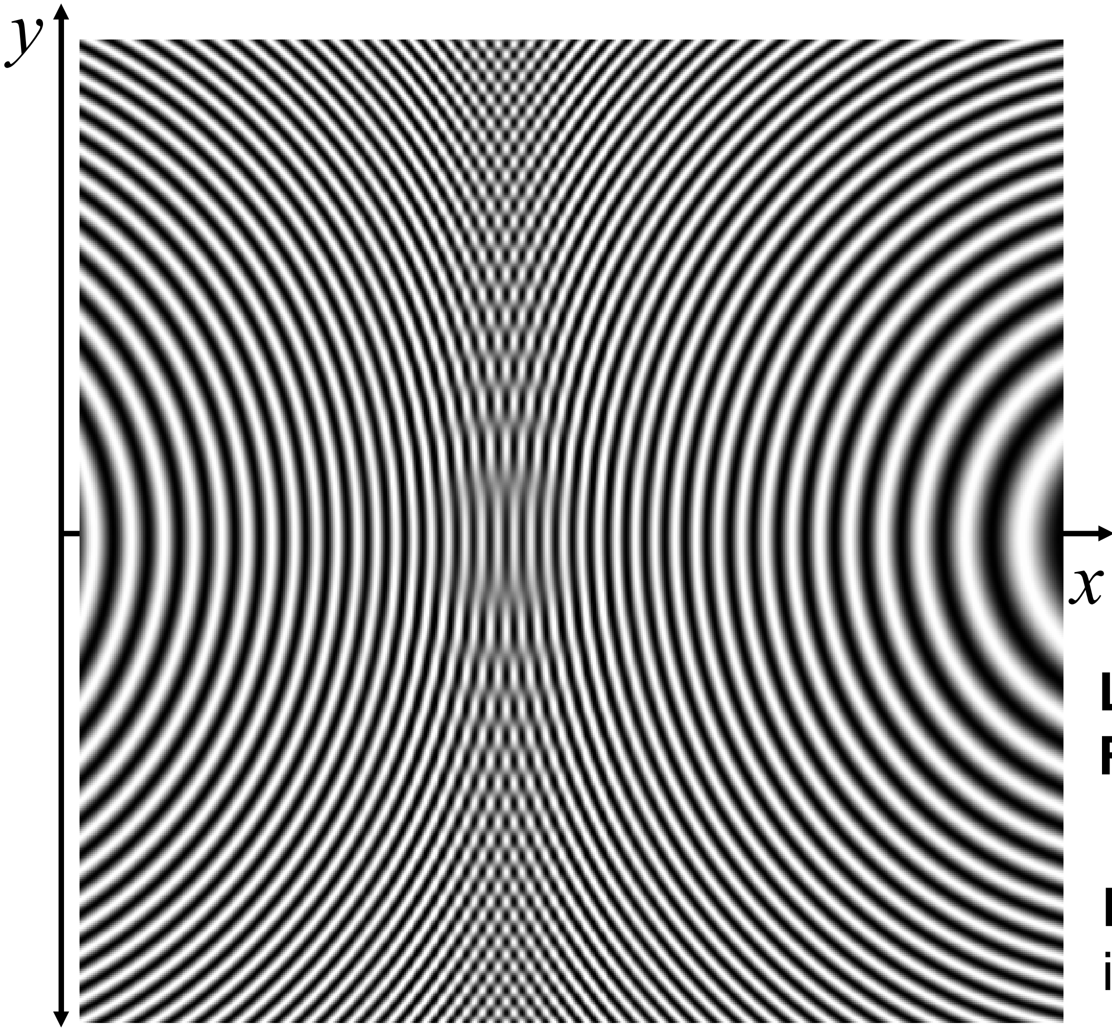
- Claude Shannon, 1949

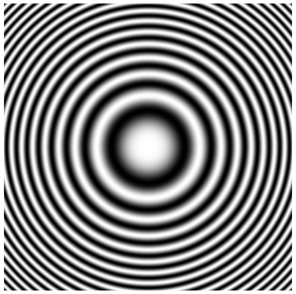
    A signal can be reconstructed from its samples without loss of information, if the original signal has no frequencies above 1/2 the Sampling frequency.

- *DEF: Bandlimited function.* There is some frequency u_{max}, above which the spectrum is identically zero.
- For a given bandlimited function, the rate at which it must be sampled ($2u_{max}$) is called the Nyquist Frequency.

Pat Hanrahan

Sampling a "Zone Plate"





Zone plate: $\sin x^2 + y^2$

Sampled at 128x128
Reconstructed to 512x512 using a 30-wide windowed sinc

Left rings: part of signal
Right rings: prealiasing

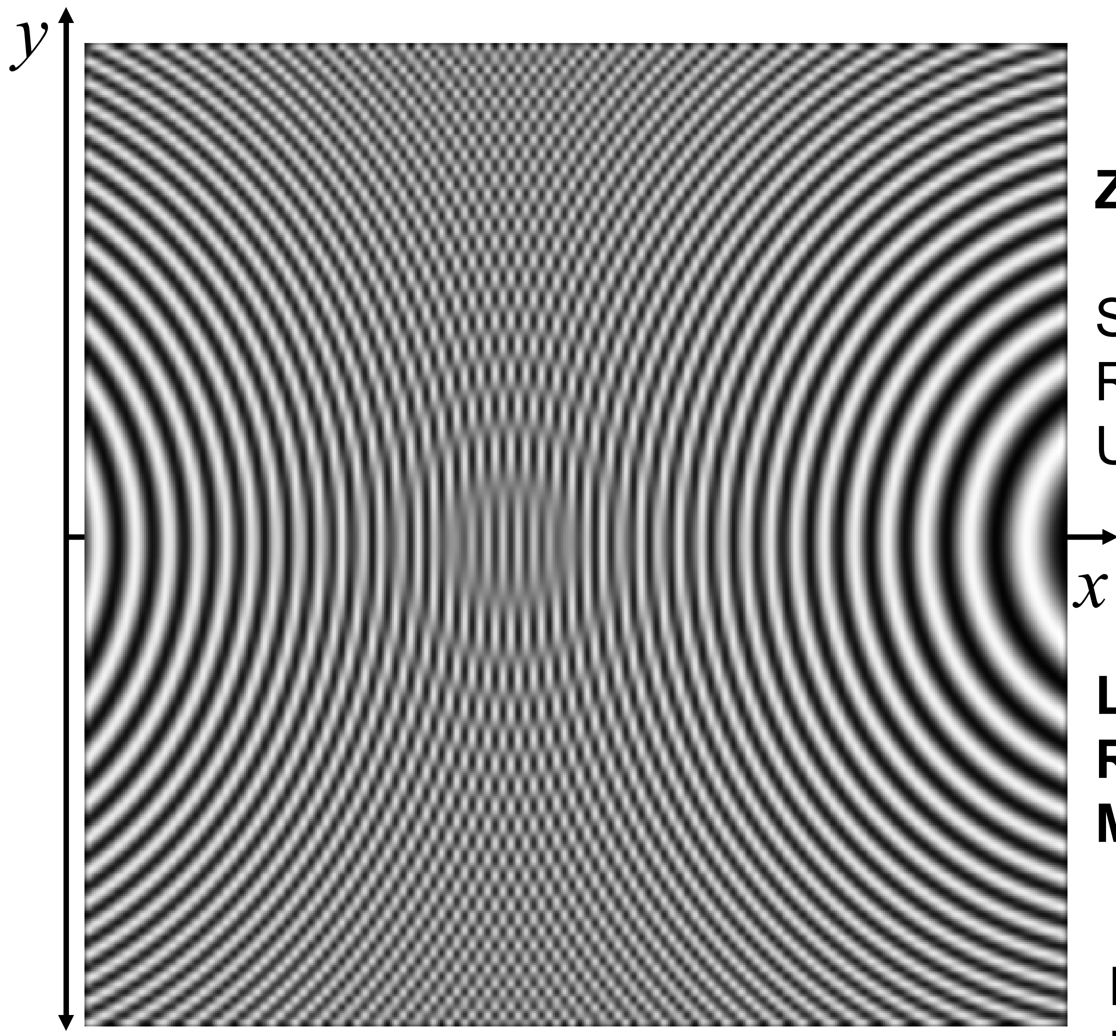
Prealiasing: due to inadequate sampling

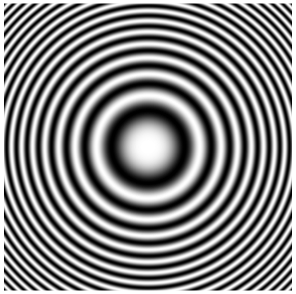
Ideal Reconstruction

- Ideally, use a perfect low-pass filter - the sinc function - to bandlimit the sampled signal and thus remove all copies of the spectra introduced by sampling
- Unfortunately,
  - The sinc has infinite extent and we must use simpler filters with finite extents. Physical processes in particular do not reconstruct with sincs
  - The sinc may introduce ringing which are perceptually objectionable

Pat Hanrahan

Sampling a "Zone Plate"





Zone plate: $\sin x^2 + y^2$

Sampled at 128x128
Reconstructed to 512x512
Using optimal cubic filter

Left rings: part of signal
Right rings: prealiasing
Middle rings: postaliasing

Postaliasing: due to inappropriate reconstruction

Antialiasing

Antialiasing Techniques

- Antialiasing = Preventing aliasing

1. Analytically prefilter the signal
   - Solvable for points, lines, polygons and **image textures**
   - Not solvable in general
     e.g. procedurally defined geometry or textures

2. Uniform supersampling and resampling

3. Non-uniform or stochastic sampling

Antialiasing Techniques

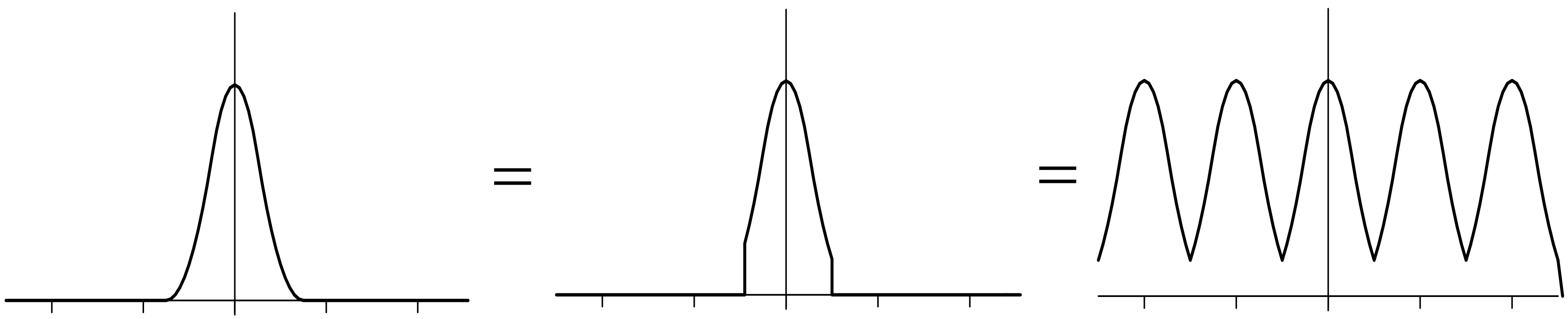
- Antialiasing = Preventing aliasing

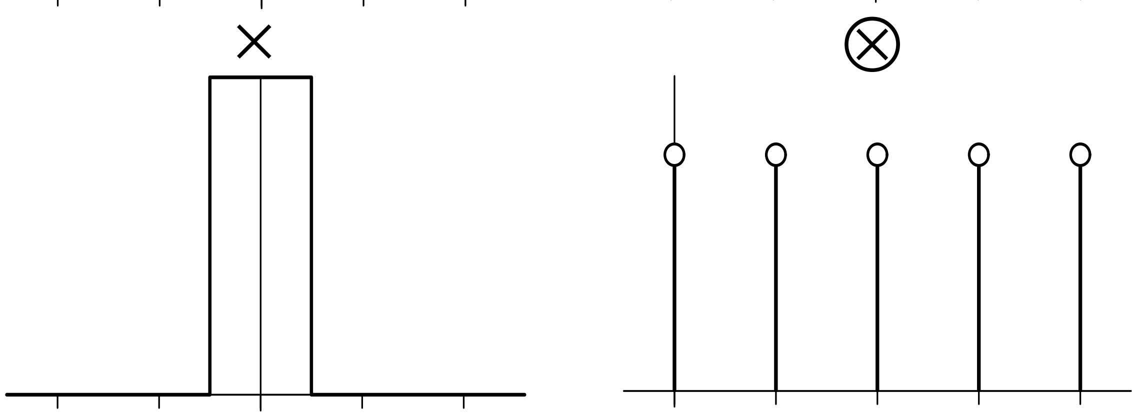
1. Analytically prefilter the signal
   - Solvable for points, lines, polygons and **image textures**
   - Not solvable in general
     e.g. procedurally defined geometry or textures
2. Uniform supersampling and resampling
3. Nonuniform or stochastic sampling

Antialiasing by Prefiltering

1. First bandlimit the signal (cut off high frequencies = “blur”)
2. Then sample

- Sampling process in **frequency domain**:





Band-limit

Sample

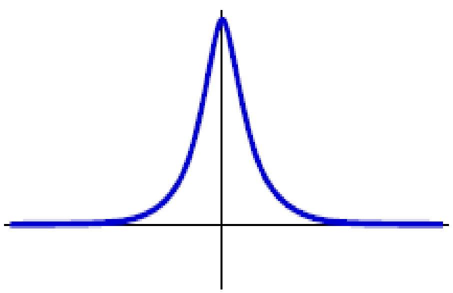
- Replicas do not overlap
  - They would without pre-filtering
- Bandlimited signal can be reconstructed
  - Not the very original

Antialiasing by Prefiltering

- Sampling process in **spatial domain:**
  1. Convolve with ideal prefilter, *h* (ideally h = sinc)
  2. Sample: multiply by III(x)

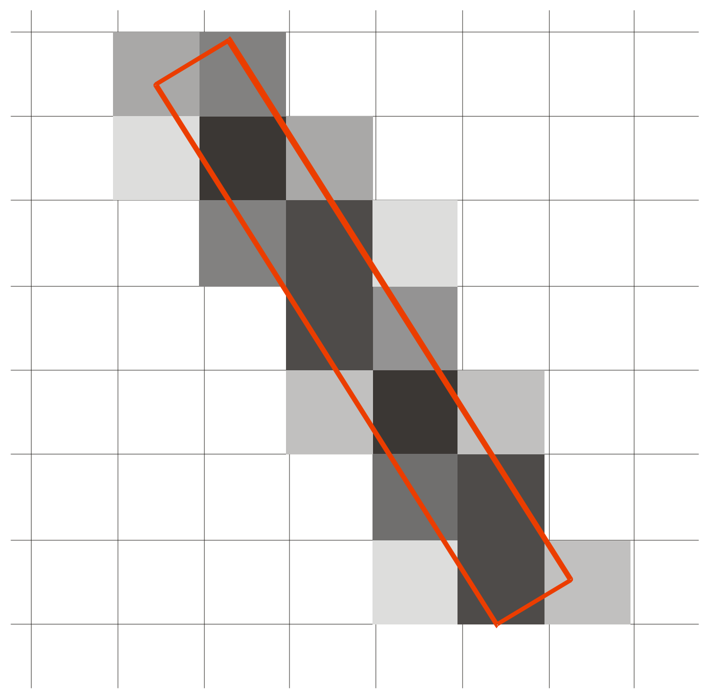
$$f_s(x) = \left[f(x) \otimes h(x) \right] \times \mathrm{III}(x)$$

- In practice:
  - sinc replaced by a locally supported filter, e.g. truncated Gaussian
  - taking filtered samples
    - filter centered at the sample location



Anti-aliased Lines With Pre-filtering

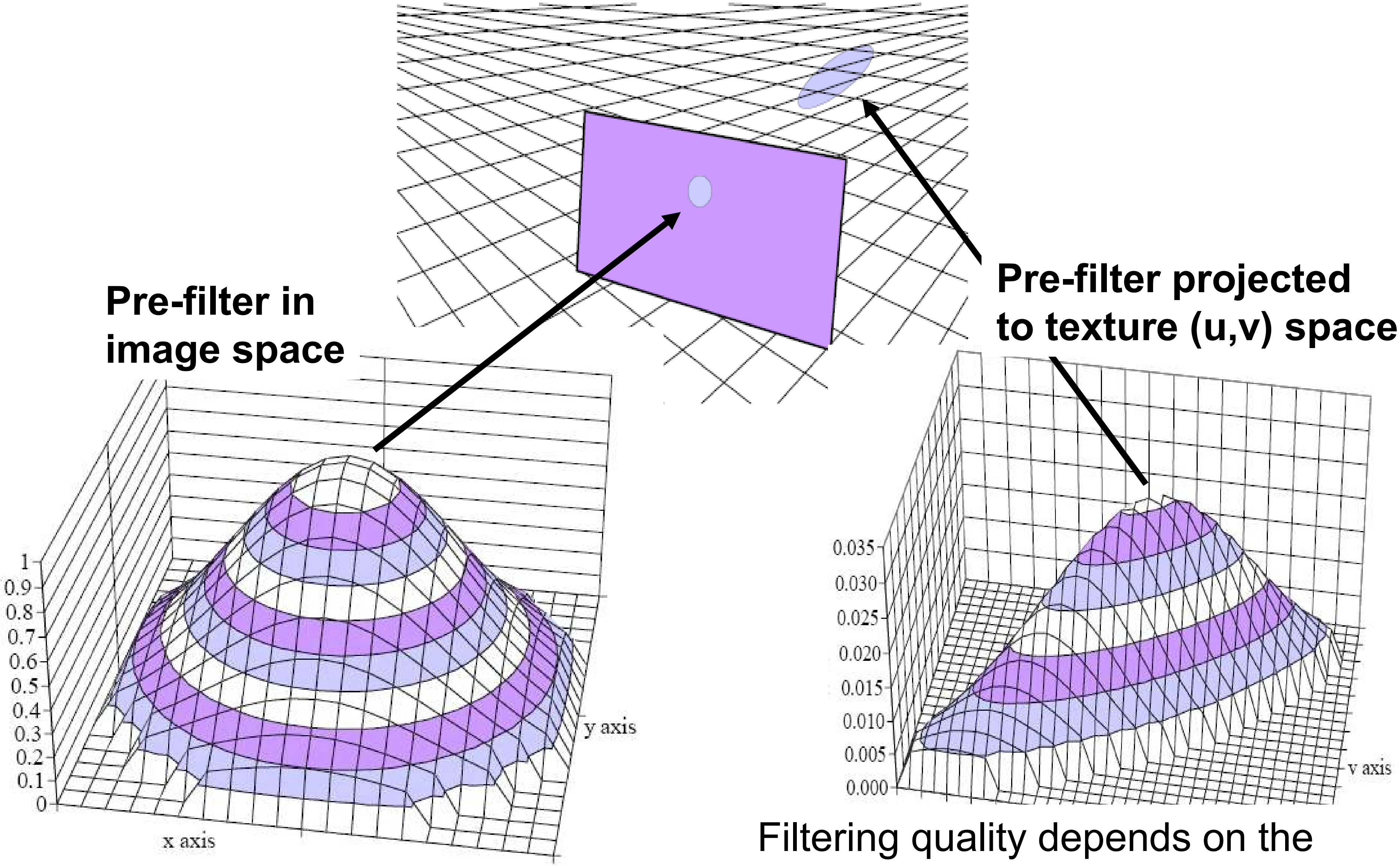
- Practice
  - Compute analytically the pixel area covered by the line
  - Assign a color based on this analytically computed coverage
- The same thing in terms of the signal theory
  1. Convolve with a box-shaped pre-filter (box=pixel)
  2. Sample at pixel centers
- Beware - box pre-filter is bad!
  - Spectrum is sinc – leaves a lot of high frequencies
  - So this method of line antialiasing is not optimal (but often sufficient)



Texture Antialiasing by Prefiltering

- Pre-filter placed at the pixel
- Projected to texture space
- Convolution computed as a weighted sum of texels

Texture Antialiasing by Prefiltering



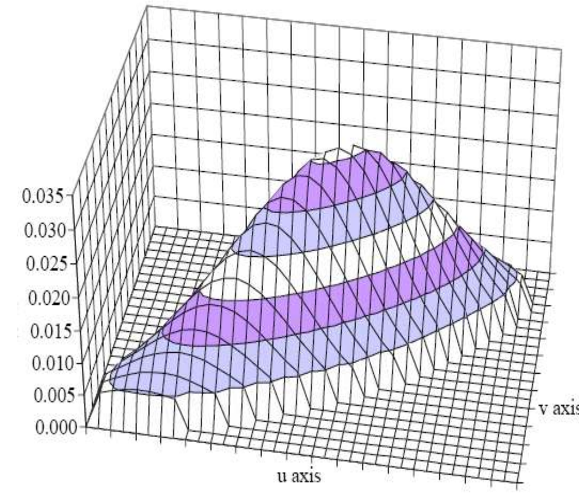
Filtering quality depends on the approximation of the projected filter.

Trilinear Filtering

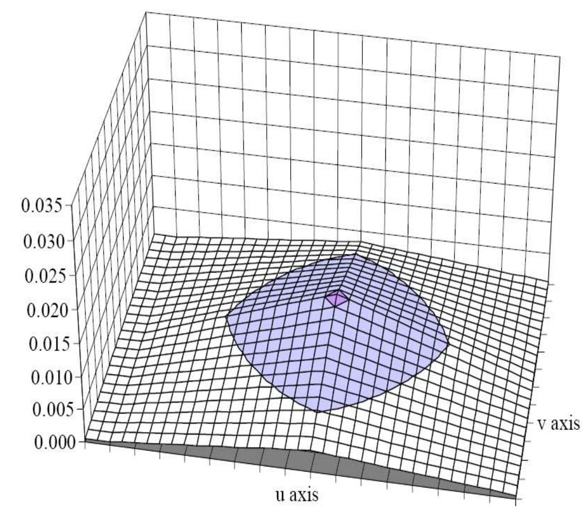
- Most commonly used in GPUs

1. Choose a MIP-map level, so that the projected filter covers approximately 4 texels
2. Bilinear texture interpolation in two adjacent MIP-map levels
3. Linear interpolation between the two levels

- Only isotropic filtering
- Poor approximation of the projected prefilter

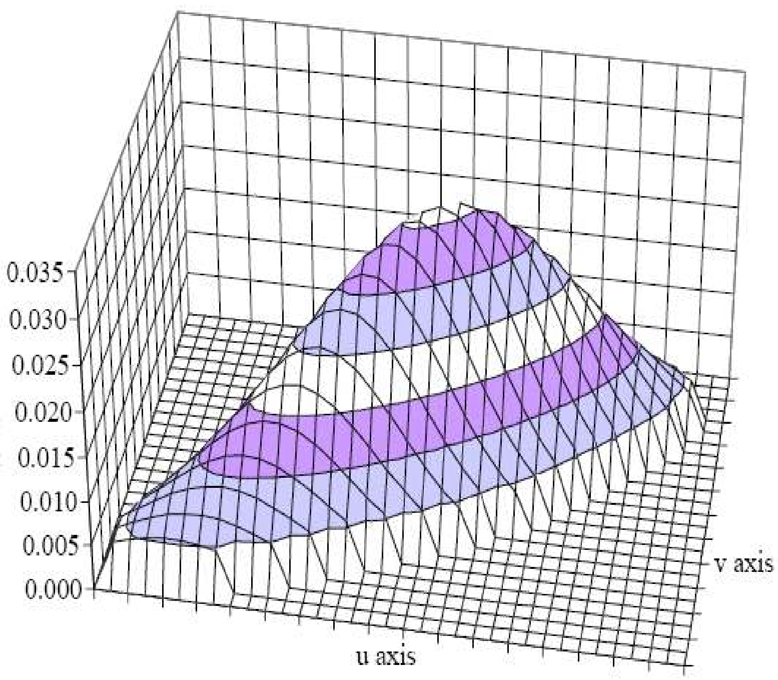


$\approx$

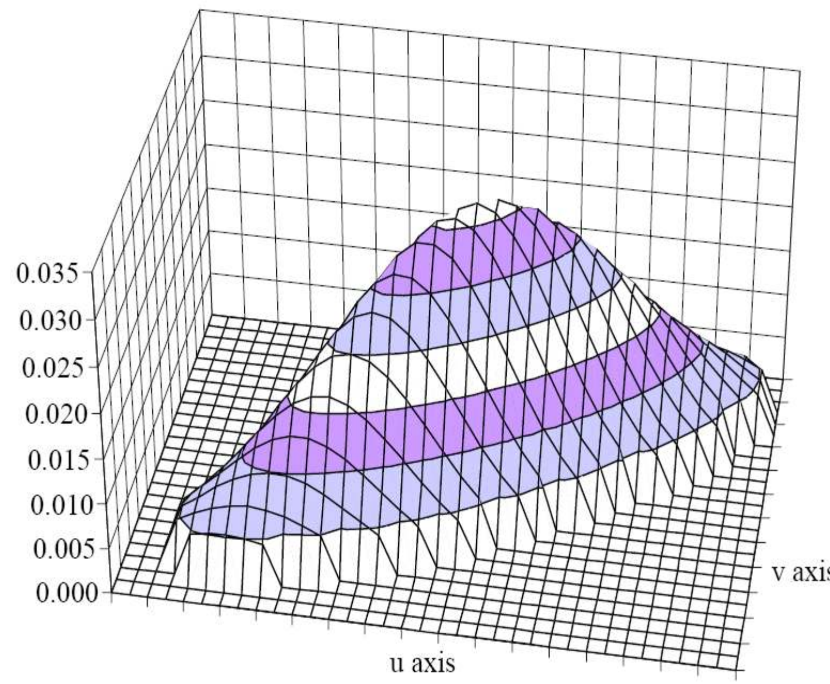


EWA Texture Filtering

- EWA = Elliptical Weighted Average
- Approximated by an elliptical gaussian – close match
- Allows anisotropic filtering

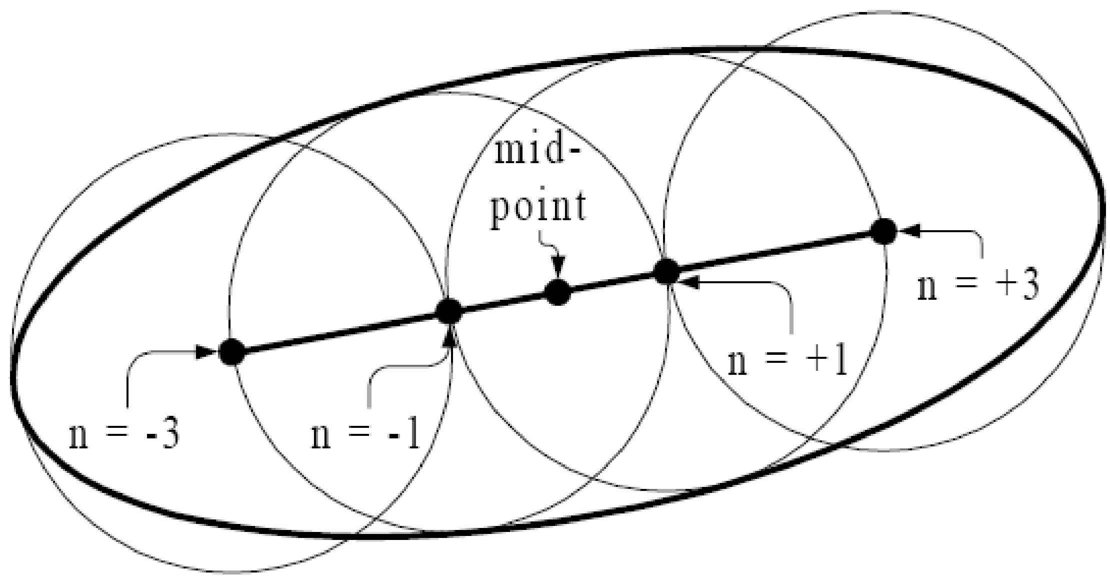


≈



Anisotropic Filtering on the GPU

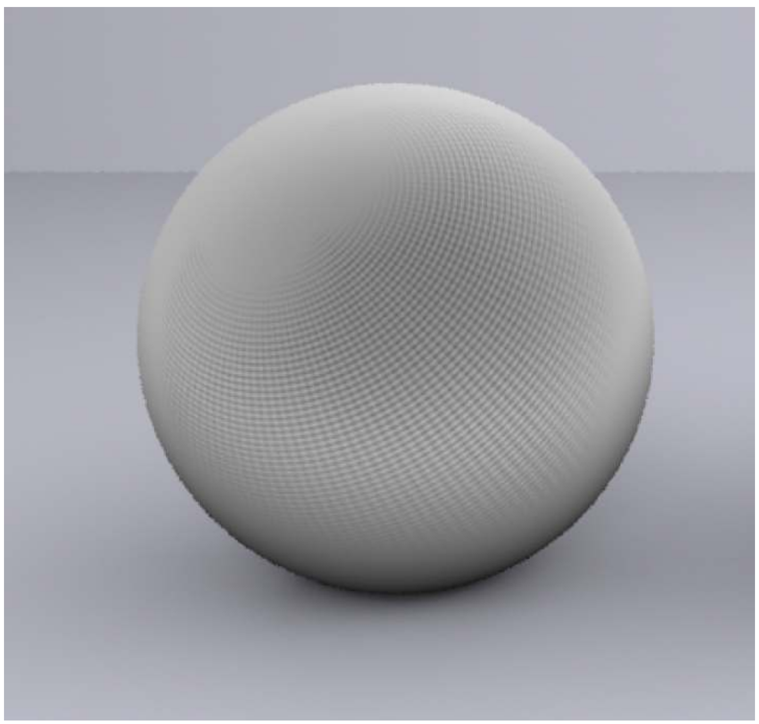
- Approximate projected pre-filter by a number of tri-linear look-ups
- E.g. 4 x aniso



Texture Filtering Quality

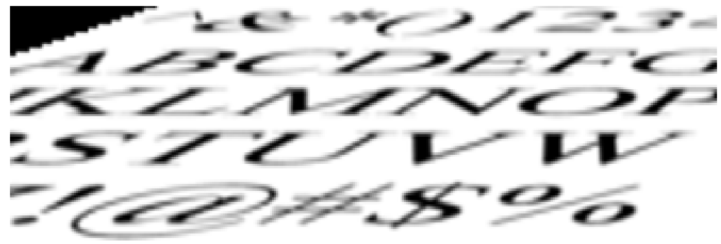
Trilinear

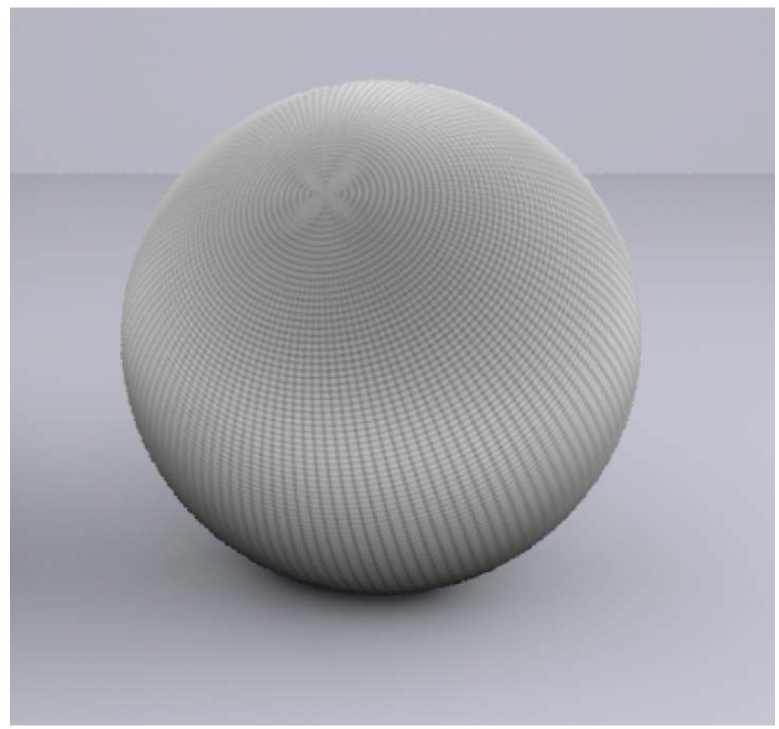
- Bad overall quality - tent filter
- Blurs near silhouettes - Isotropic



EWA

- Better overall quality - Gauss filter
- Silhouettes preserved - Anisotropic





Ray Differentials

- Texture filtering for reflected and refracted rays

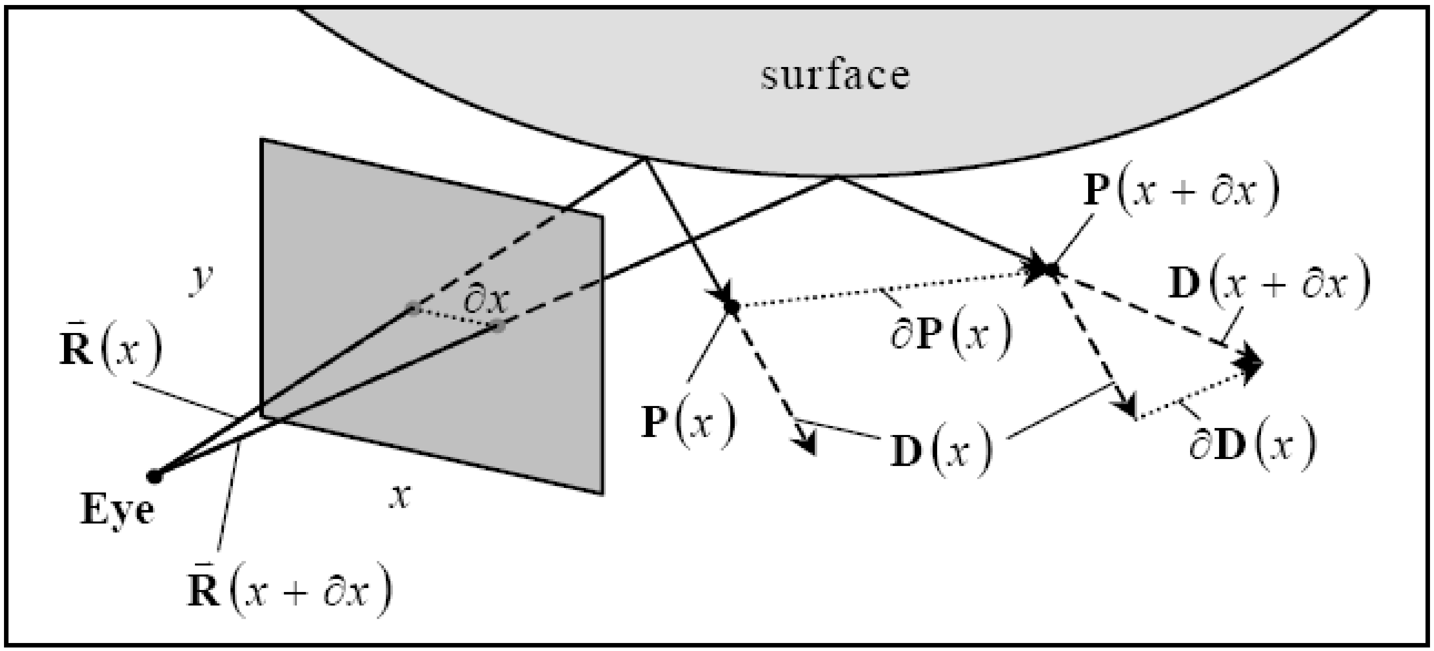


Figure 1: A Ray Differential. The diagram above illustrates the positions and directions of a ray and a differentially offset ray after a reflection. The difference between these positions and directions represents a ray differential.

Ray Differentials

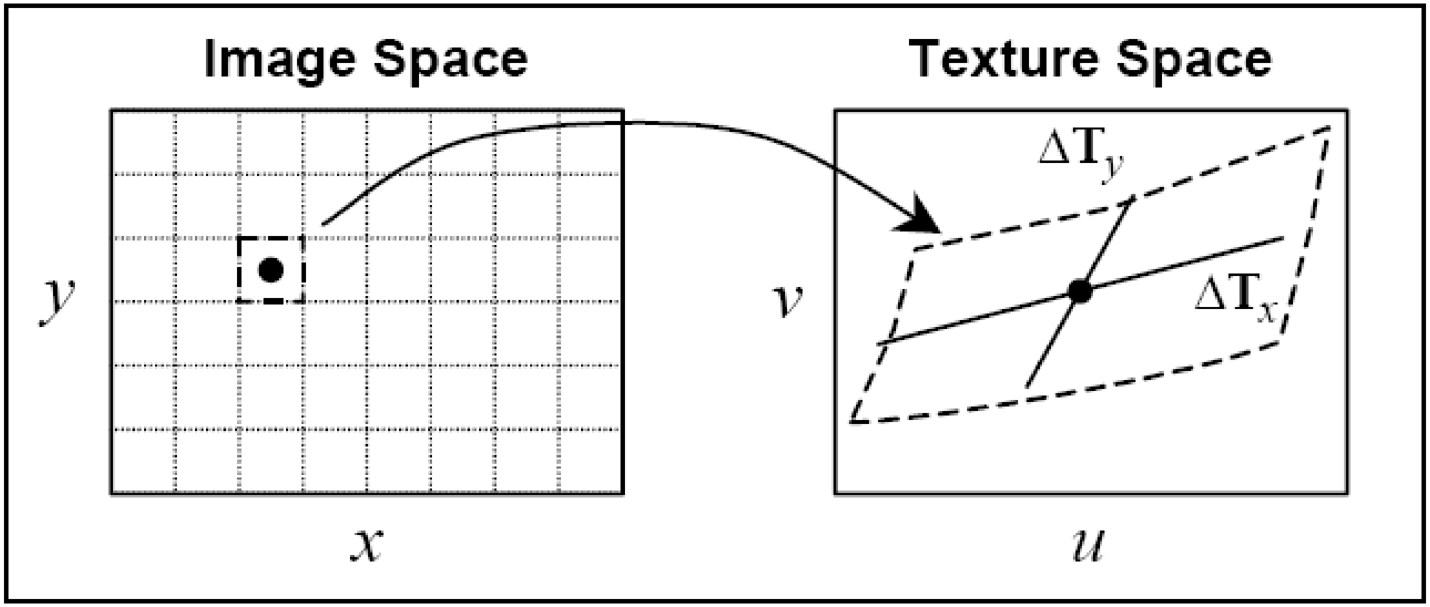


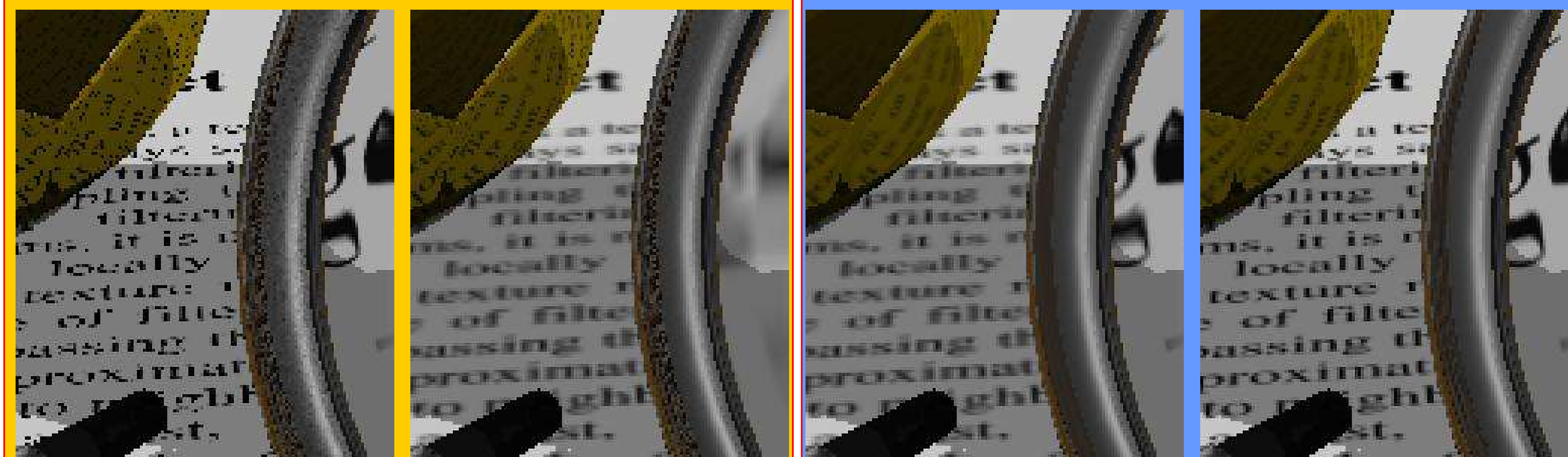
Figure 2: Texture Filtering Kernel. A pixel's footprint in image space can map to an arbitrary region in texture space. This region can be estimated by a parallelogram formed by a first-order differential approximation of the ratios between rate of change in texture space and image space coordinates.

Ray Differentials

Homan Igehy, **Tracing Ray Differentials**
In *Proc. of SIGGRAPH '99*. 1999
http://graphics.stanford.edu/papers/trd/

Sample Scene

Footprint based on distance | Footprint based ray differential



Bilinear base texture | Trilinear mip map | Trilinear mip map | Anisotropic mip map

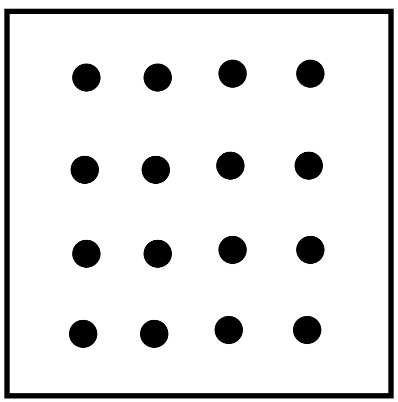
Antialiasing Techniques

- Antialiasing = Preventing aliasing

1. Analytically prefilter the signal
   - Solvable for points, lines, polygons and **image textures**
   - Not solvable in general
     e.g. procedurally defined geometry or textures
2. Uniform supersampling and resampling
3. Nonuniform or stochastic sampling

Uniform Supersampling

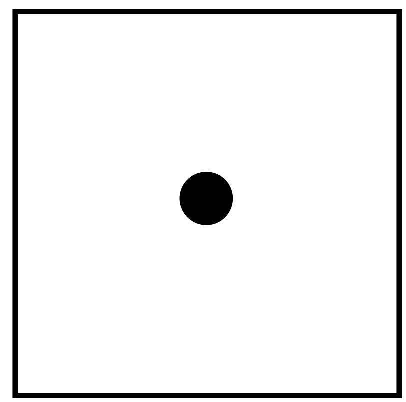
- Increasing the sampling rate moves each copy of the spectra further apart, potentially reducing the overlap and thus aliasing
- Resulting samples must be resampled (filtered) to image sampling rate



Samples

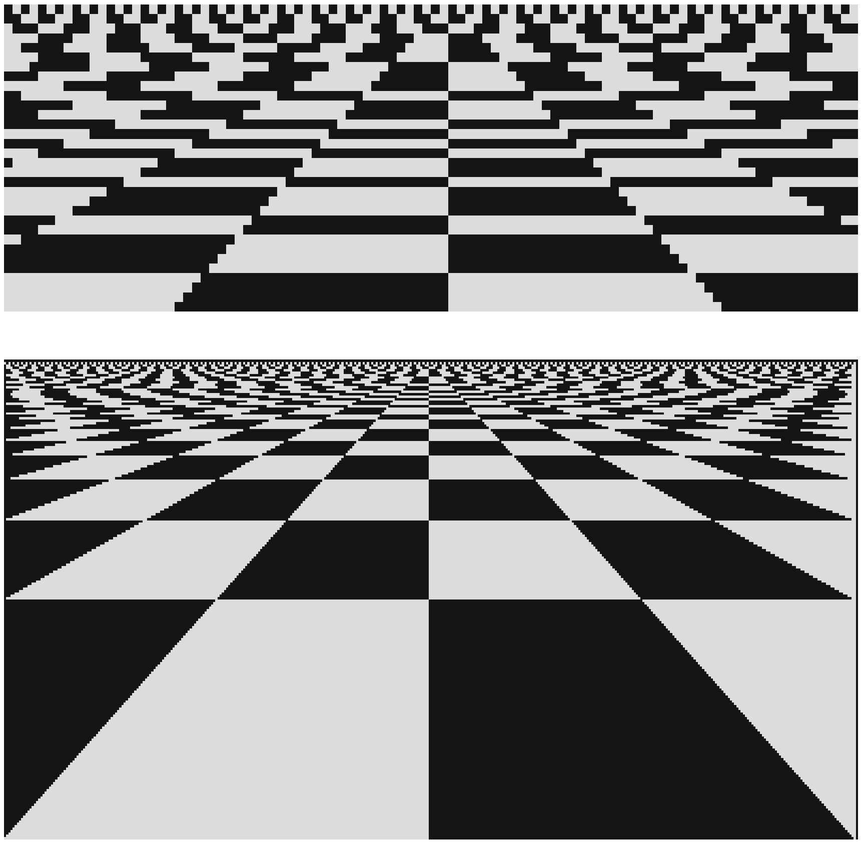
$$Pixel = \sum_{s} w_s \cdot Sample_s$$

resampling filter evaluated at the sample location

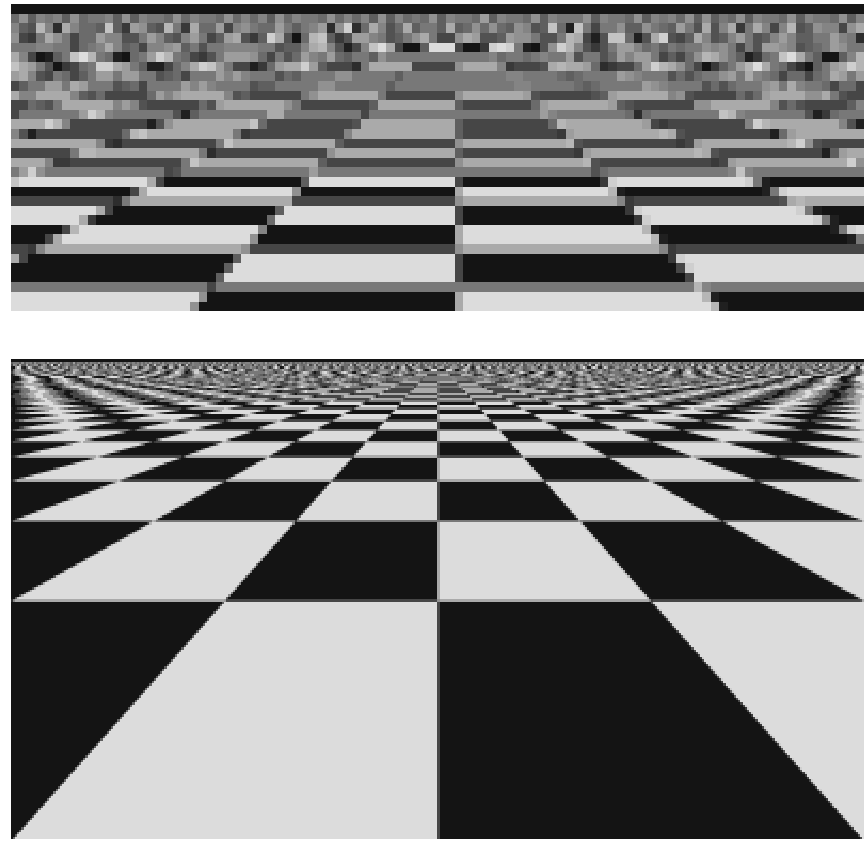


Pixel

Point vs. Supersampled



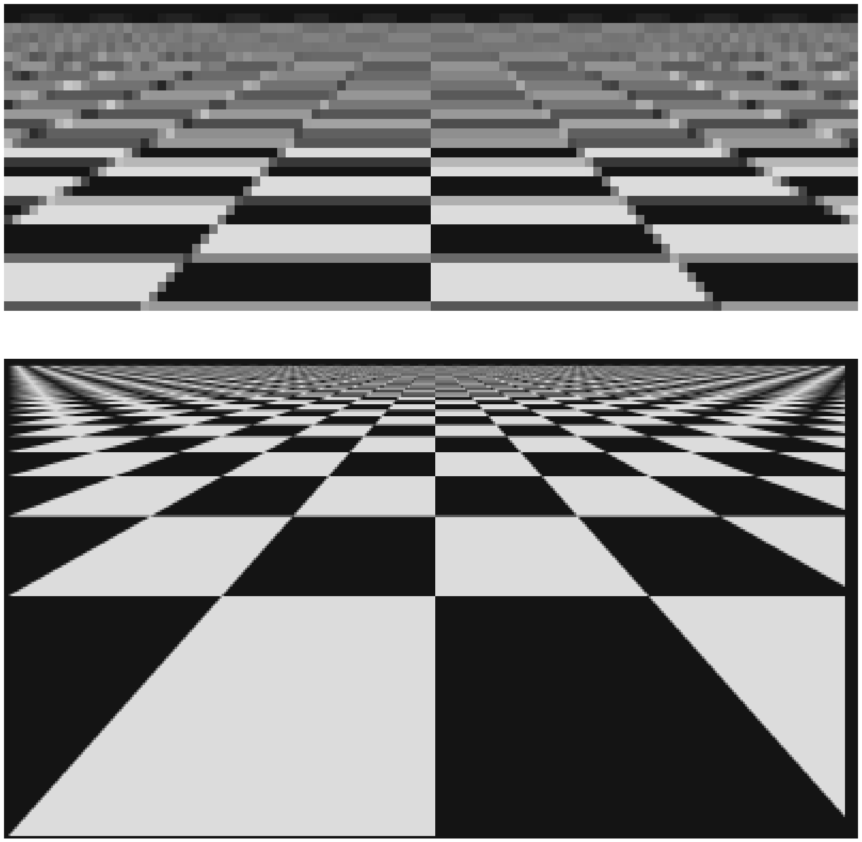
Point



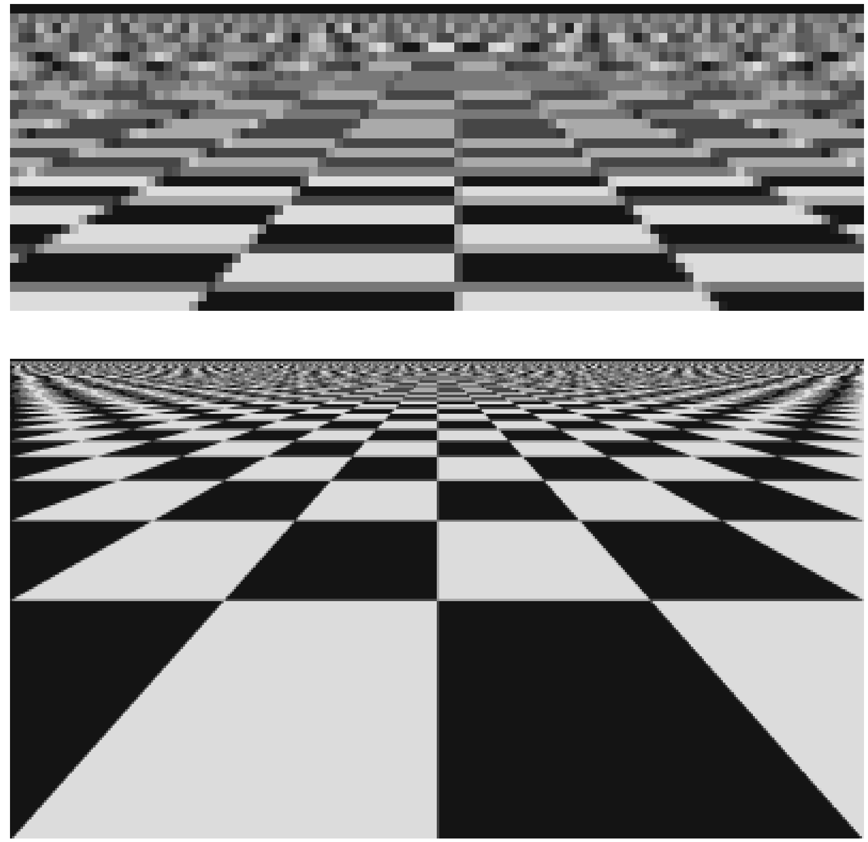
4x4 Supersampled

Checkerboard sequence by Tom Duff

Analytic vs. Supersampled



Exact Area



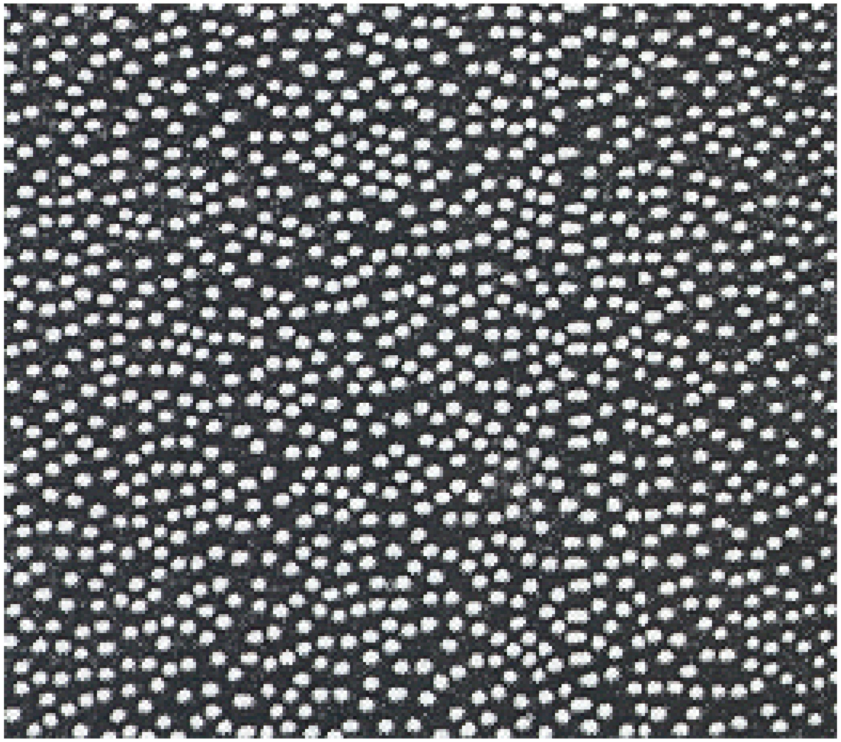
4x4 Supersampled

Antialiasing Techniques

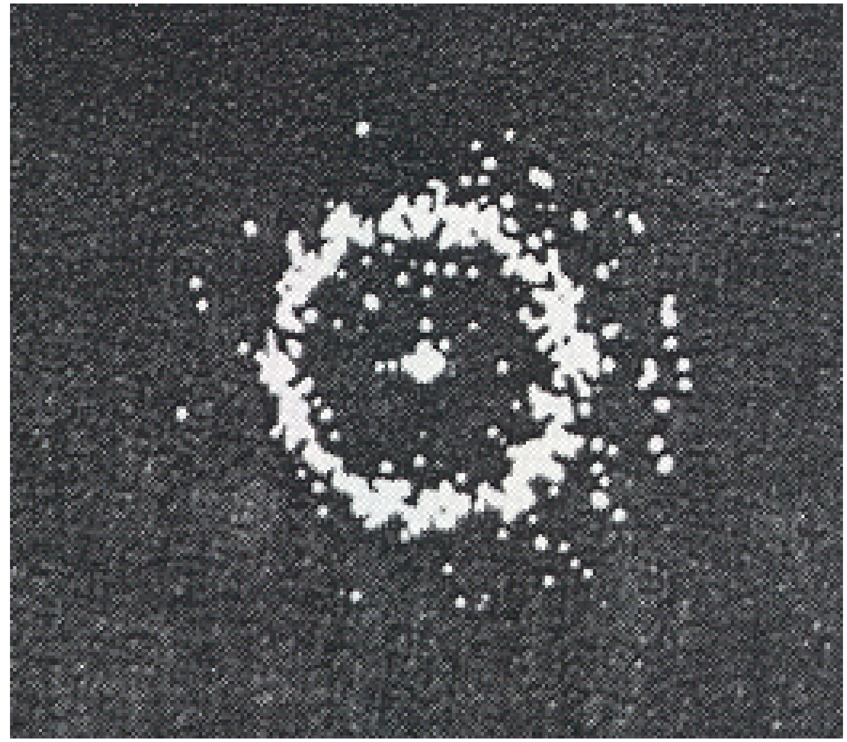
- Antialiasing = Preventing aliasing

1. Analytically prefilter the signal
   - Solvable for points, lines, polygons and **image textures**
   - Not solvable in general
     e.g. procedurally defined geometry or textures
2. Uniform supersampling and resample
3. Nonuniform or stochastic sampling

Distribution of Extrafoveal Cones



Monkey eye
cone distribution

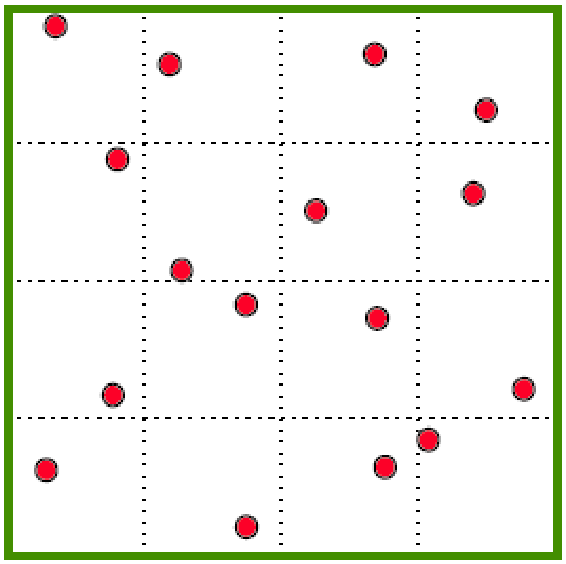


Fourier transform

Yellot theory

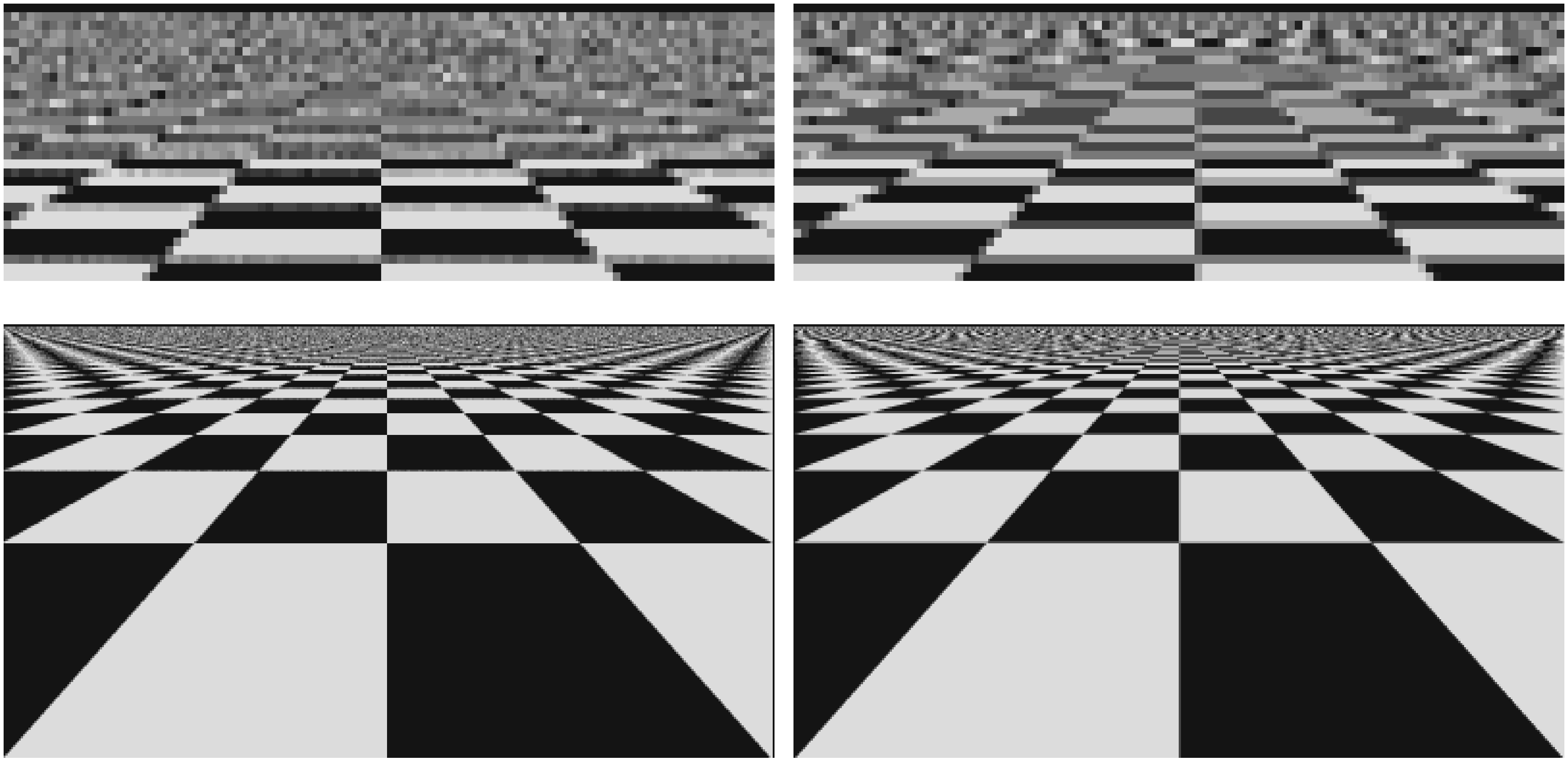
- Aliases replaced by noise
- Visual system less sensitive to high freq noise

Jittering



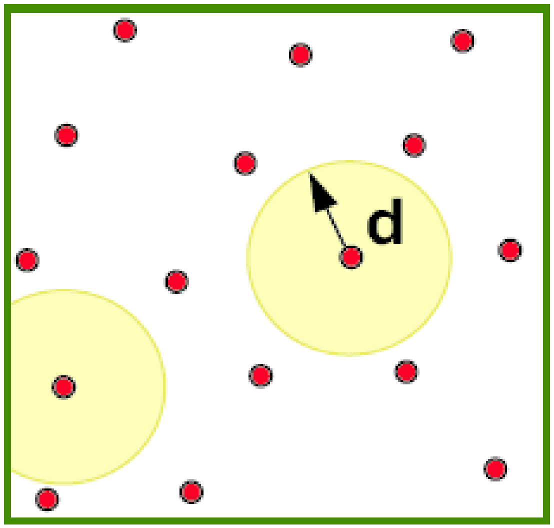
- Jittering = stratified sampling on a grid
- Prevents clustering of random points
- Better sample distribution than pure random sampling
- However, clusters of up to four points can appear in 2D!

Jittered vs. Uniform Supersampling



4x4 Jittered Sampling **4x4 Uniform**

Poisson Disk Sampling



- Gives by far the best quality for image sampling
- No sample closer to any other than a specified threshold *d*
- Prevents sample clumping – better than jittering
- Efficient implementation did not exist for long time
- Common approach
  - Precompute pattern for a block of *N* x *N* pixels
  - Reuse a randomly rotated version of the pattern

Implementation of Poisson Disk Sampling

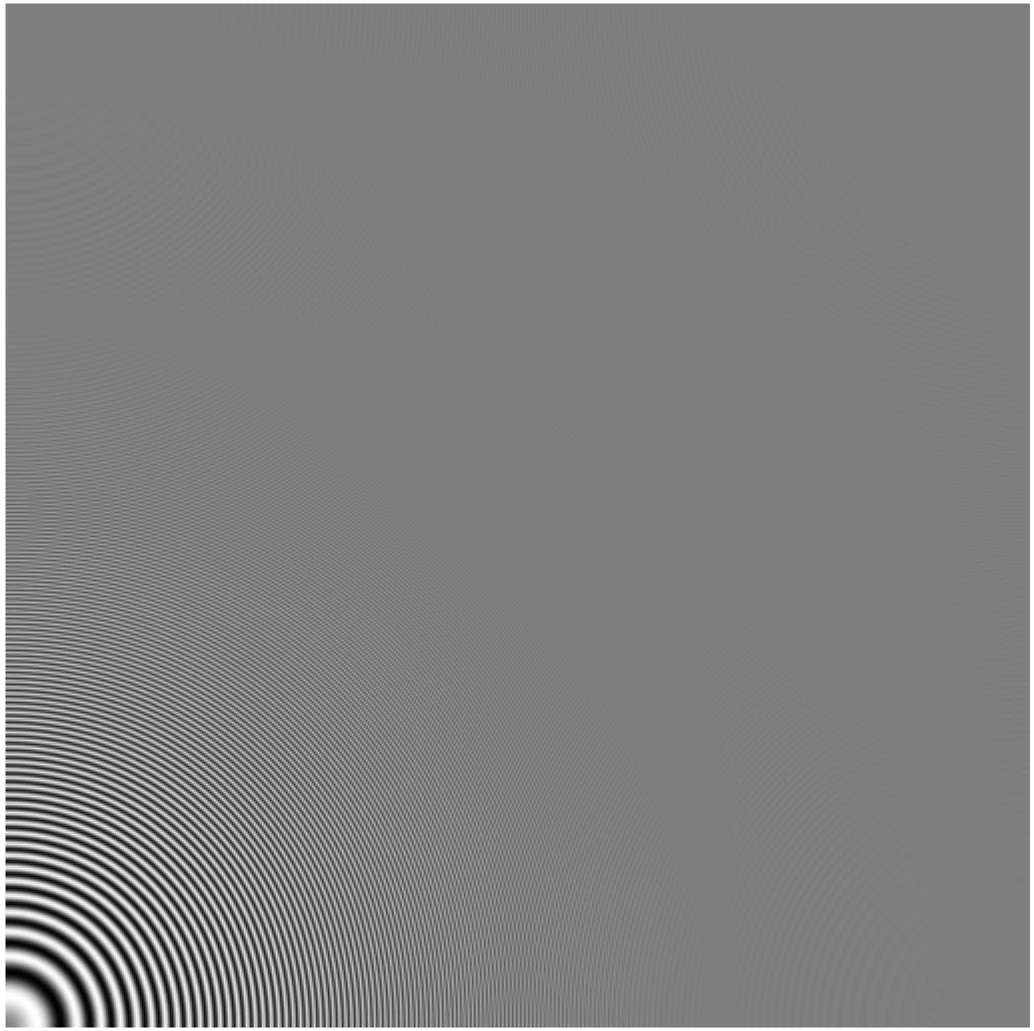
- Dart Throwing
  1. Create Candidates Randomly
  2. Discard if too close to an existing point
  - Extremely slow
  - Problem. How to set *d* for a desired number of points?
- Best Candidate Sampling (Mitchell)
  - Generates the pattern progressively
  1. Choose first sample randomly
  2. To generate (k+1)-th sample
     - generate *k.q* independent candidates
     - choose one farthest from the *k* existing samples
  - Bigger *q* -> better pattern quality

Implementation of Poisson Disk Sampling – Recent Advances

- Kopf et al. Recursive Wang Tiles for Real-Time Blue Noise, SIGGRAPH 2006.
- Dunbar and Humphreys. A spatial Data Structure for Fast Poisson-Disk Generation. SIGGRAPH 2006

- see videos at
  - http://johanneskopf.de/publications/blue_noise/
  - http://www.cs.virginia.edu/~gfx/pubs/antimony/

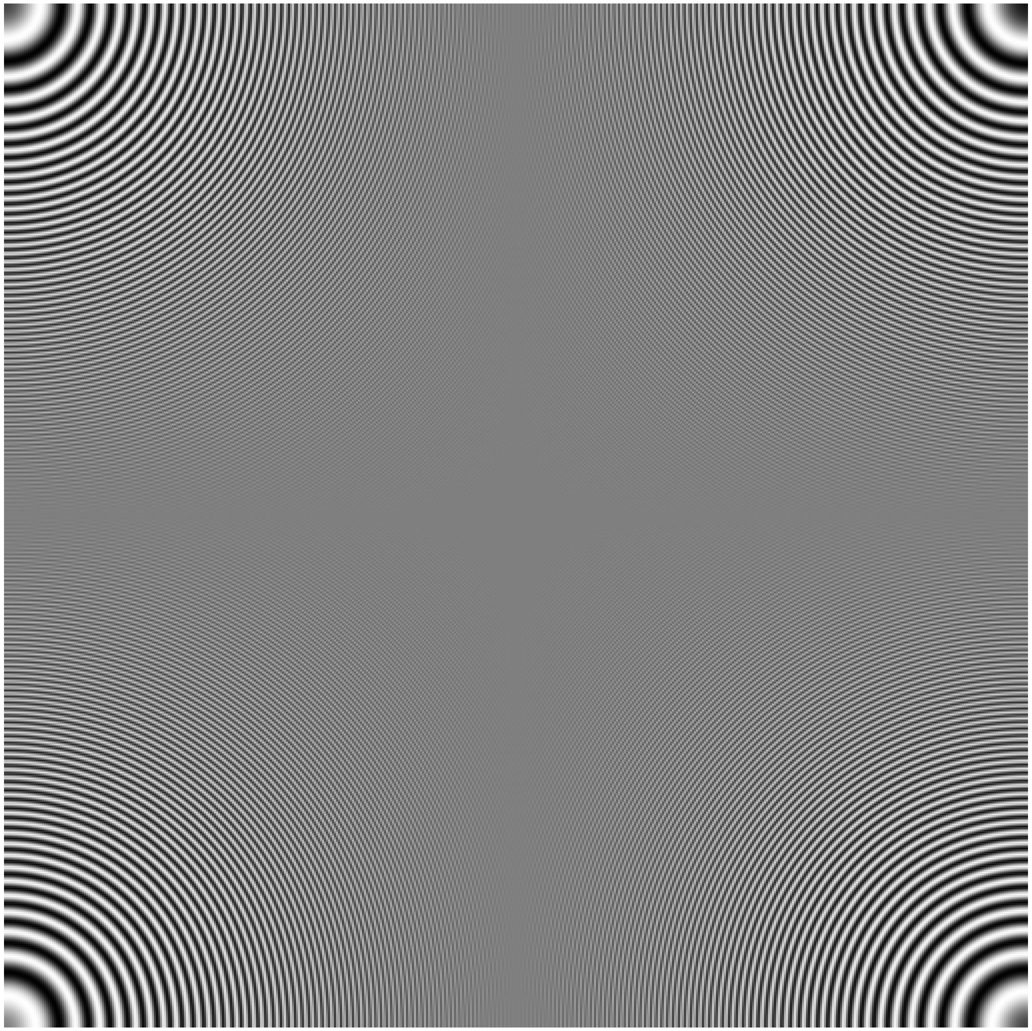
Various Sampling Patterns



Reference Image
"Zone Plate"
1,048,576 random samples/pixel

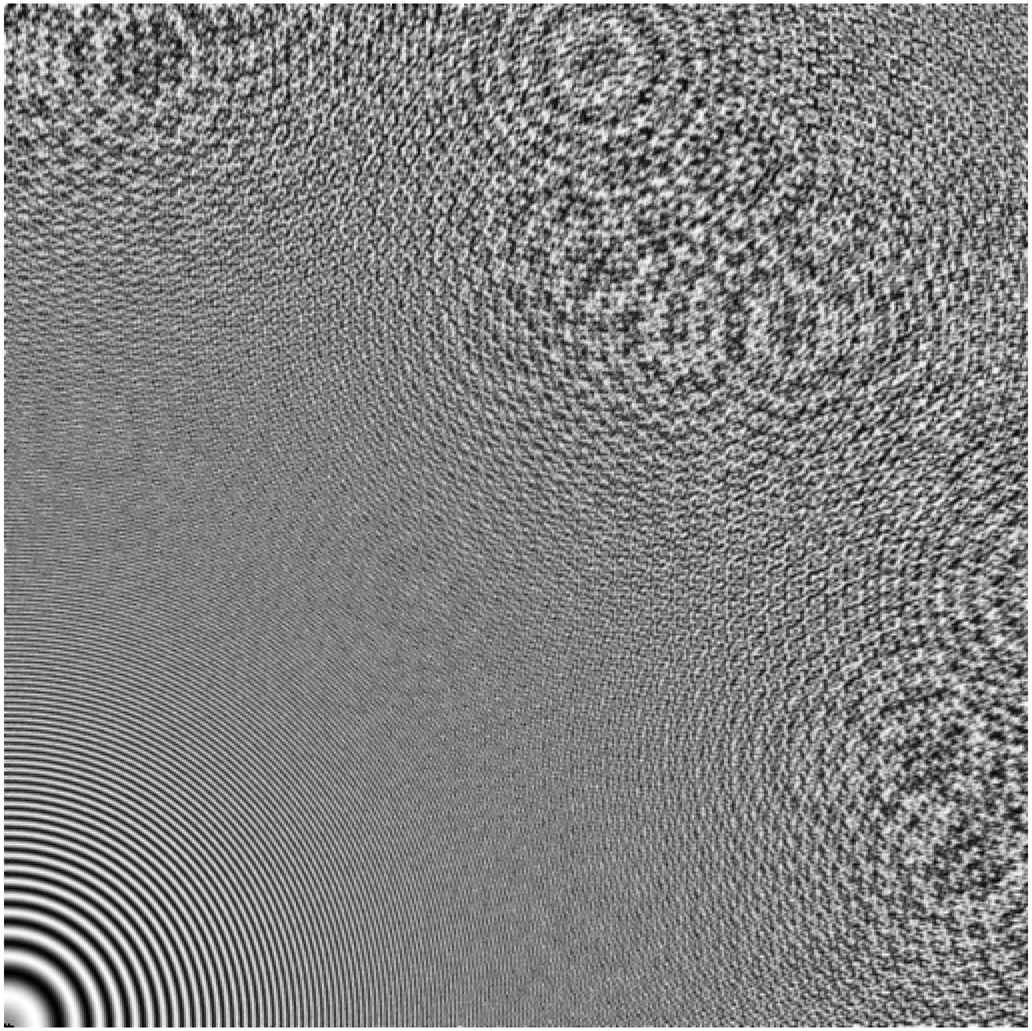
Johannes Kopf

Various Sampling Patterns



Rectilinear, 1 sample/pixel
RMS: -8.154799 dB
Pattern Generation: 17ms

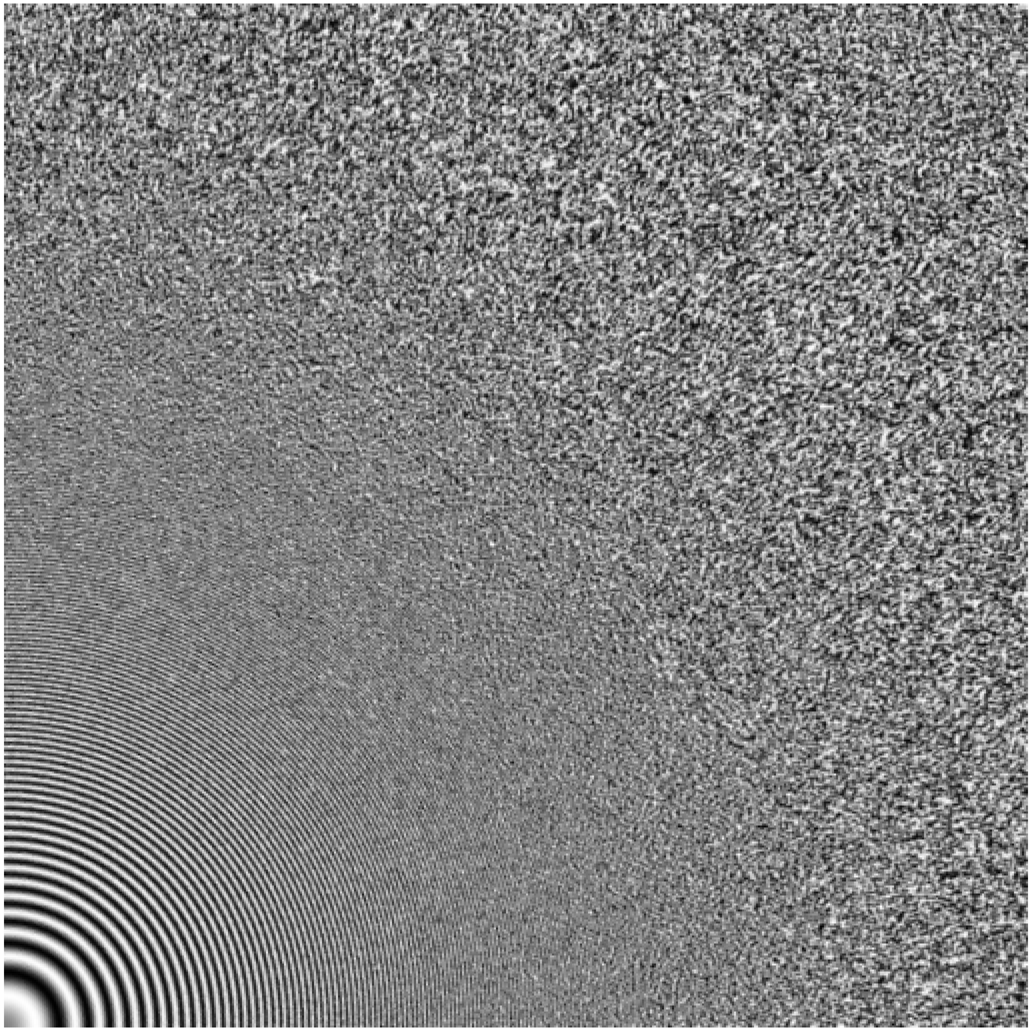
Various Sampling Patterns



Jittered Grid, 1 sample/pixel
RMS: -8.121792 dB
Pattern Generation: 25ms

Johannes Kopf

Various Sampling Patterns



Kopf – Poisson disk, 1 sample/pixel
RMS: -8.246348 dB
Pattern Generation: 17ms

Johannes Kopf

Conclusion

- Alias makes images ugly
- Rendering software **must** take care of antialiasing in order to produce compelling images without visible artifacts
- Signal analysis in Frequency domain explains aliasing and suggests antialiasing solutions
- Most common antialising techniques in graphics are
  - Pre-filtering
  - Supersampling (regular / stochastic)