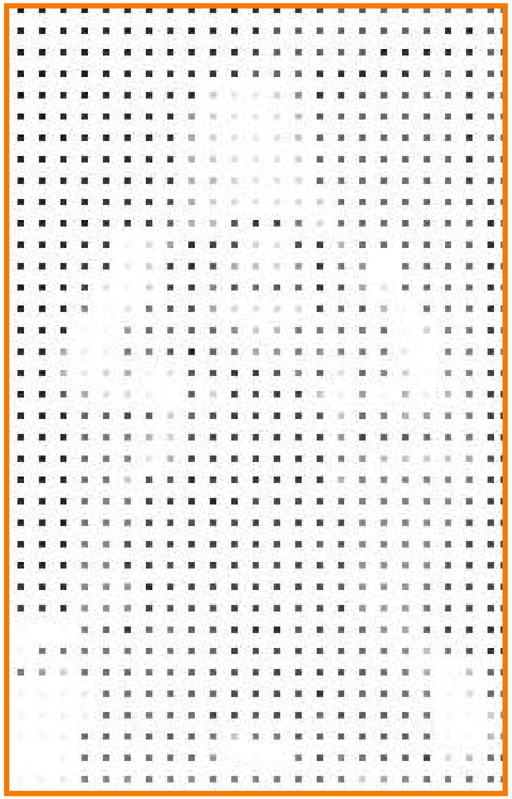
Image Processing

COS 426

What is a Digital Image?

A digital image is a discrete array of samples representing a continuous 2D function

Continuous function



Discrete samples

Limitations on Digital Images

- Spatial discretization
- Quantized intensity
- Approximate color (RGB)
- (Temporally discretized frames for digital video)

Image Processing

- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph

Digital Image Processing: Very Similar to Analog

- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph

Digital Image Processing: Account for Limitations

- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph

Digital Image Processing: Inherently new Operations

- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph
- Quantization
- Spatial / intensity tradeoff
  - Dithering

Digital Image Processing

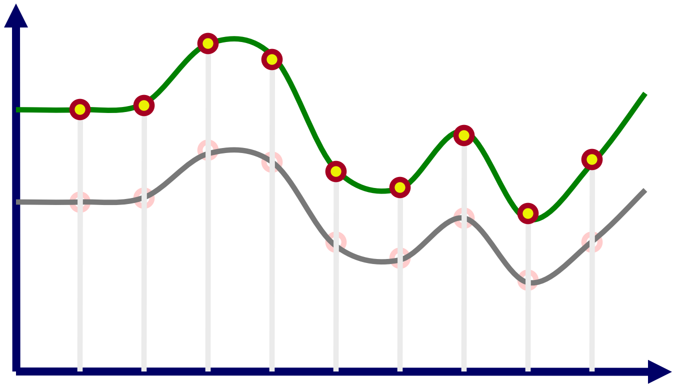
- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph
- Quantization
- Spatial / intensity tradeoff
  - Dithering

Adjusting Brightness

- Simply scale pixel components
  - o Must clamp to range (e.g., 0 to 1)

Original

Brighter



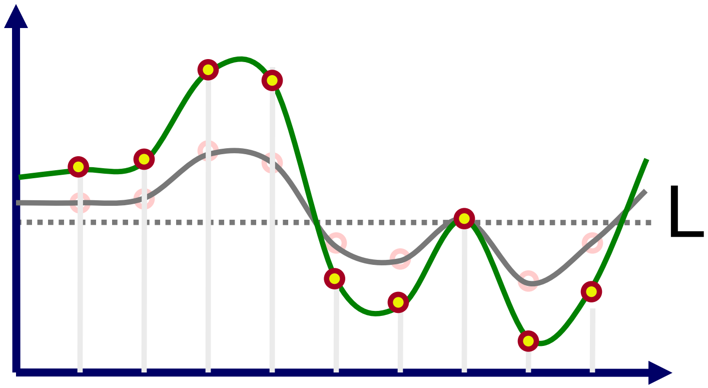
Note: this is “contrast” on your monitor!
“Brightness” adjusts black level (offset)

Adjusting Contrast

- Compute mean luminance L for all pixels
  - o luminance = 0.30*r + 0.59*g + 0.11*b
- Scale deviation from L for each pixel component
  - o Must clamp to range (e.g., 0 to 1)

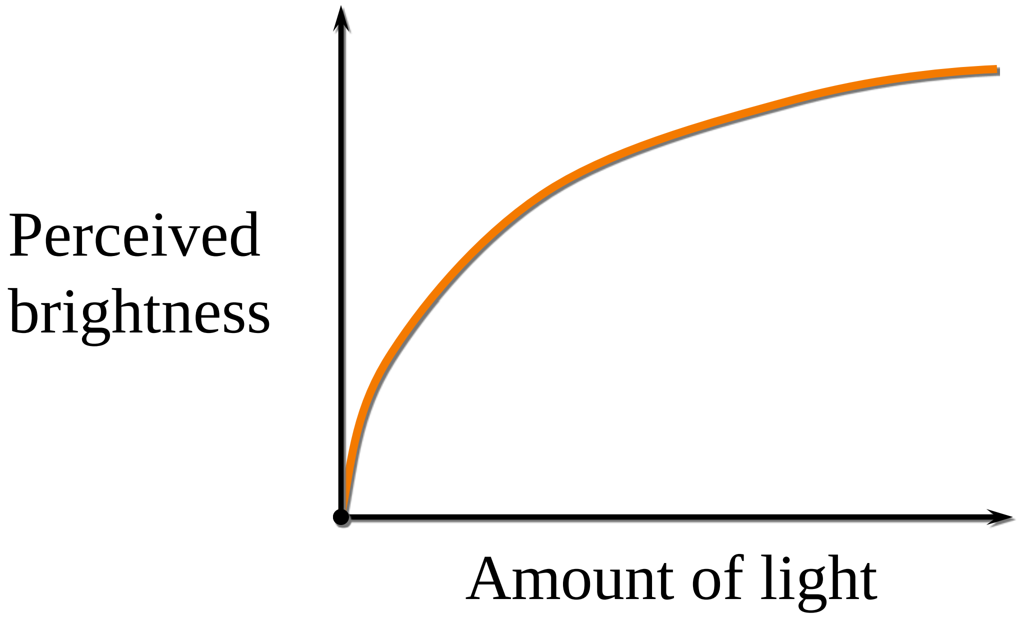
Original

More Contrast



Digression: Perception of Intensity

- Perception of intensity is nonlinear

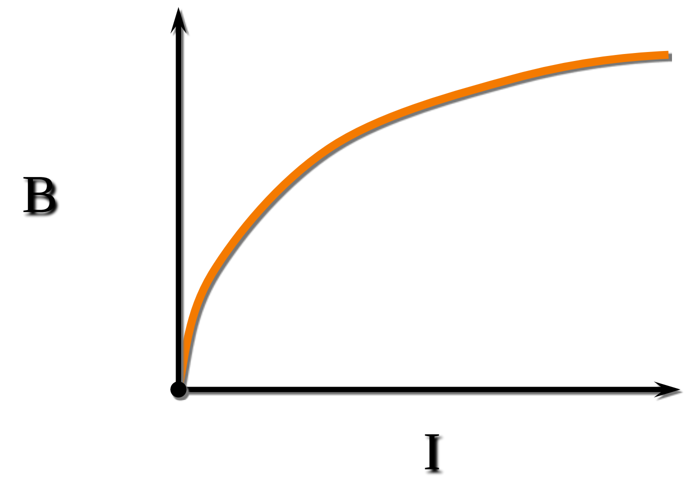


Modeling Nonlinear Intensity Response

- Brightness (*B*) usually modeled as a logarithm or power law of intensity (*I*)

$$B = k \log I$$

$$B = I^{1/3}$$



- Exact curve varies with ambient light, adaptation of eye

Cameras

- Original cameras based on Vidicon obey power law for Voltage (V) vs. Intensity (I):

$$V = I^{\gamma}$$

$$\gamma \approx 0.45$$

CRT Response

- Power law for Intensity (*I*) vs. applied voltage (*V*)

$$I = V^{\gamma}$$

$$\gamma \approx 2.5$$

- Vidicon + CRT = almost linear!
- Other displays (e.g. LCDs) contain electronics to emulate this law

CCD Cameras

- Camera gamma codified in NTSC standard
- CCDs have linear response to incident light
- Electronics to apply required power law
- So, pictures from most cameras (including digital still cameras) will have $\gamma = 0.45$
  - sRGB standard: partly-linear, partly power-law curve well approximated by $\gamma = 1 / 2.2$

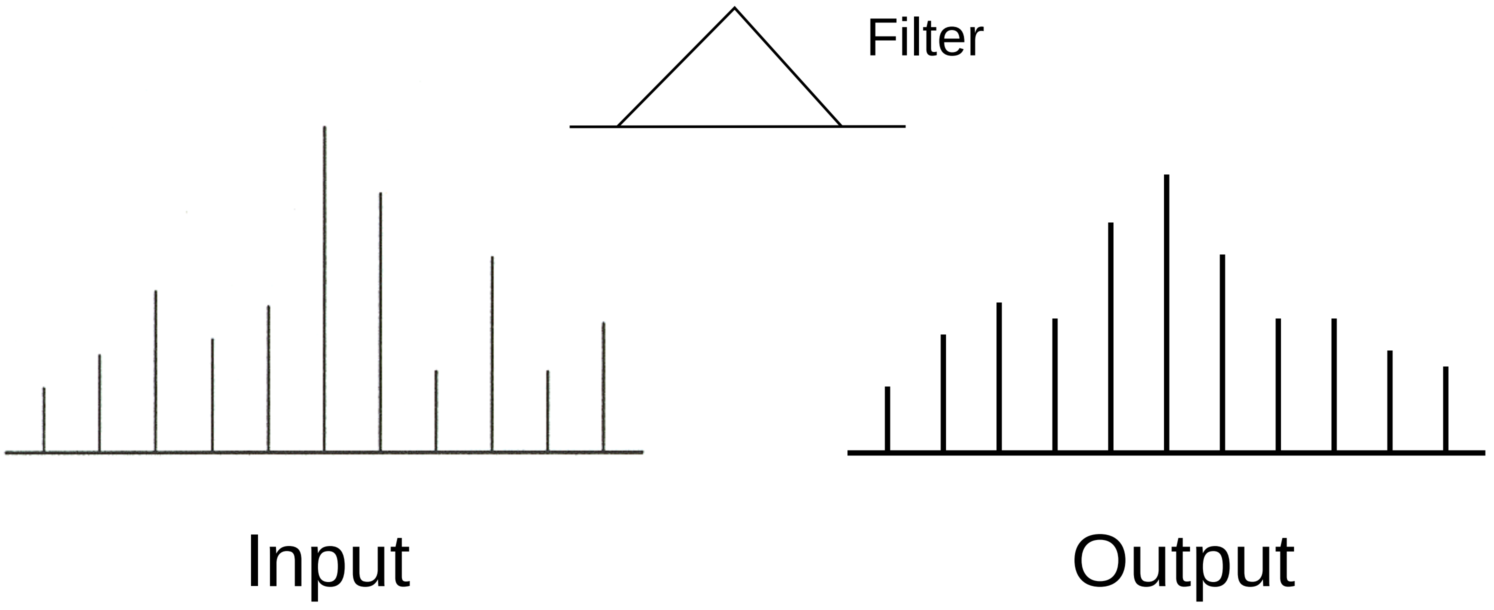
Digital Image Processing

- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph
- Quantization
- Spatial / intensity tradeoff
  - Dithering

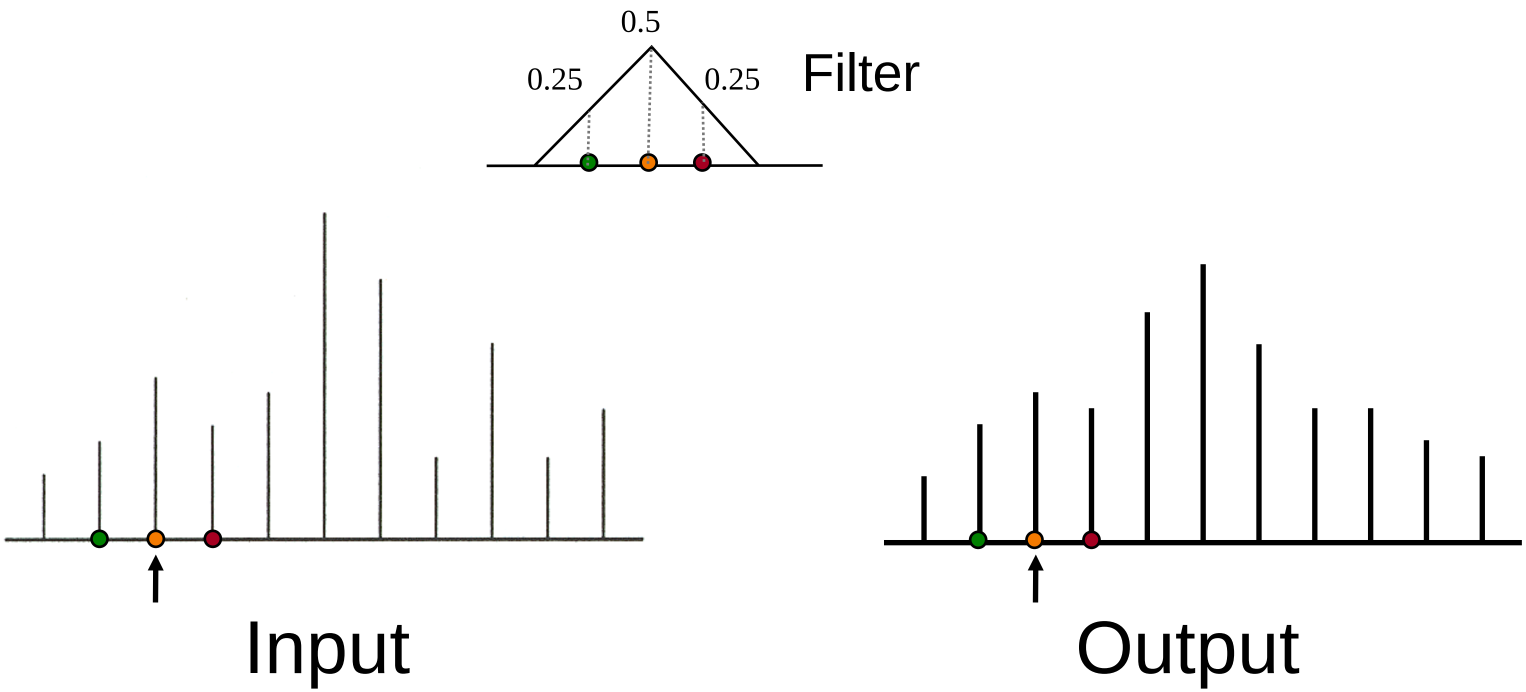
Basic Operation: Convolution

Output value is weighted sum of values in neighborhood of input image

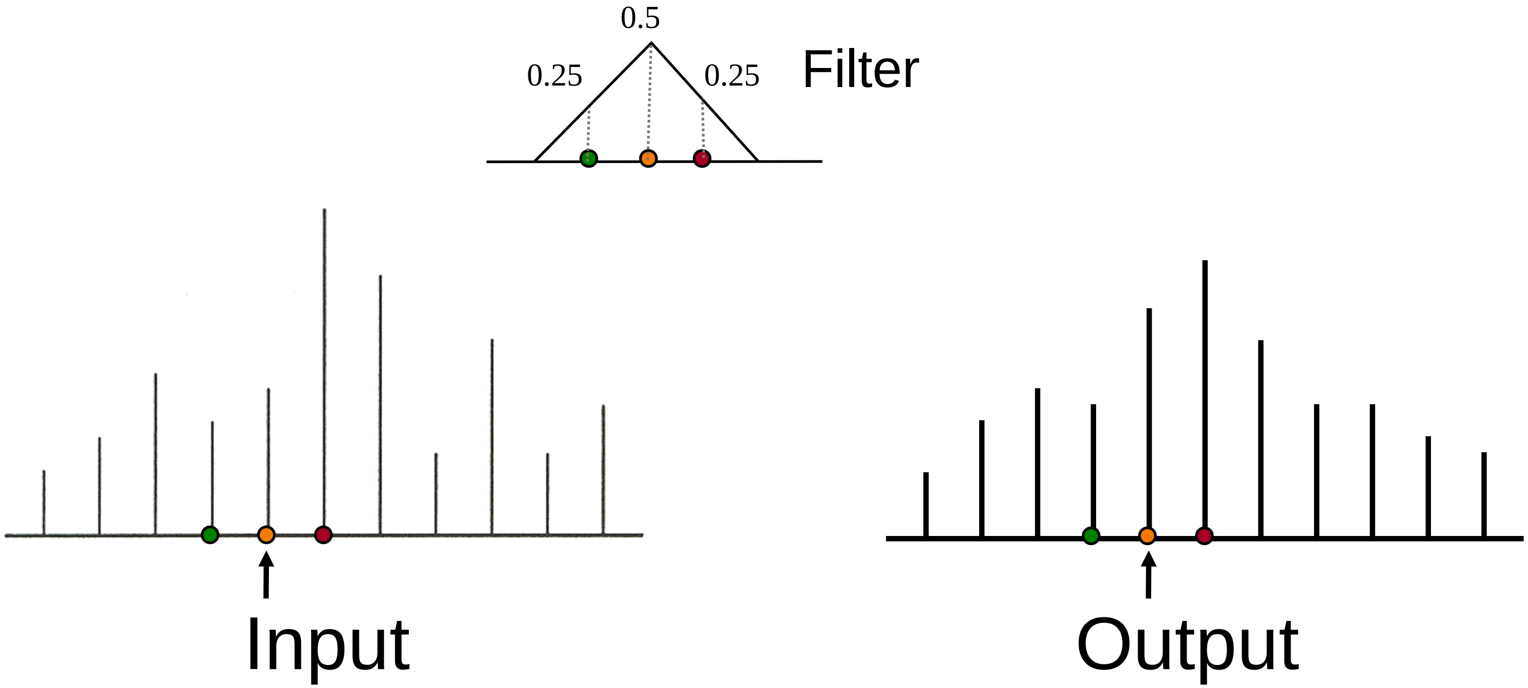
- Pattern of weights is the “filter” or “kernel”



Convolution with a Triangle Filter

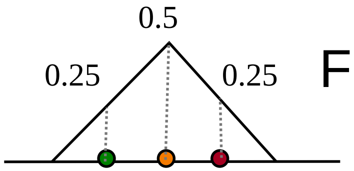


Convolution with a Triangle Filter

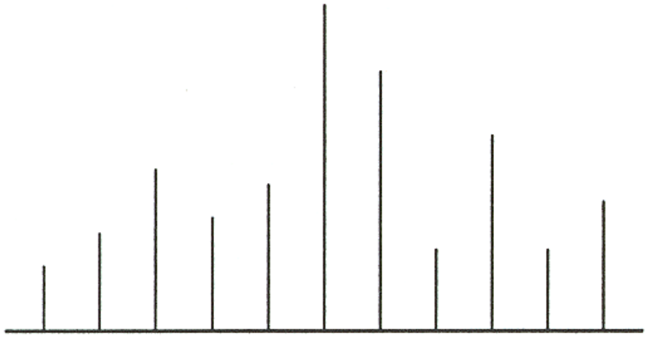


Convolution with a Triangle Filter

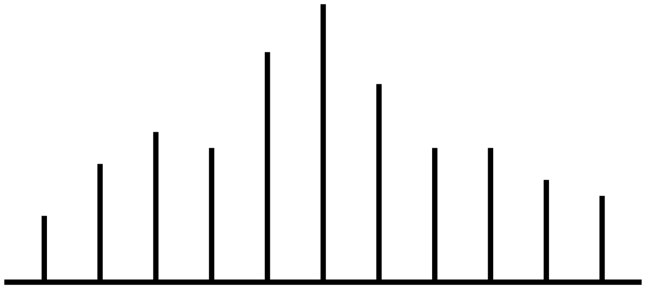
What if the filter runs off the end?



Filter



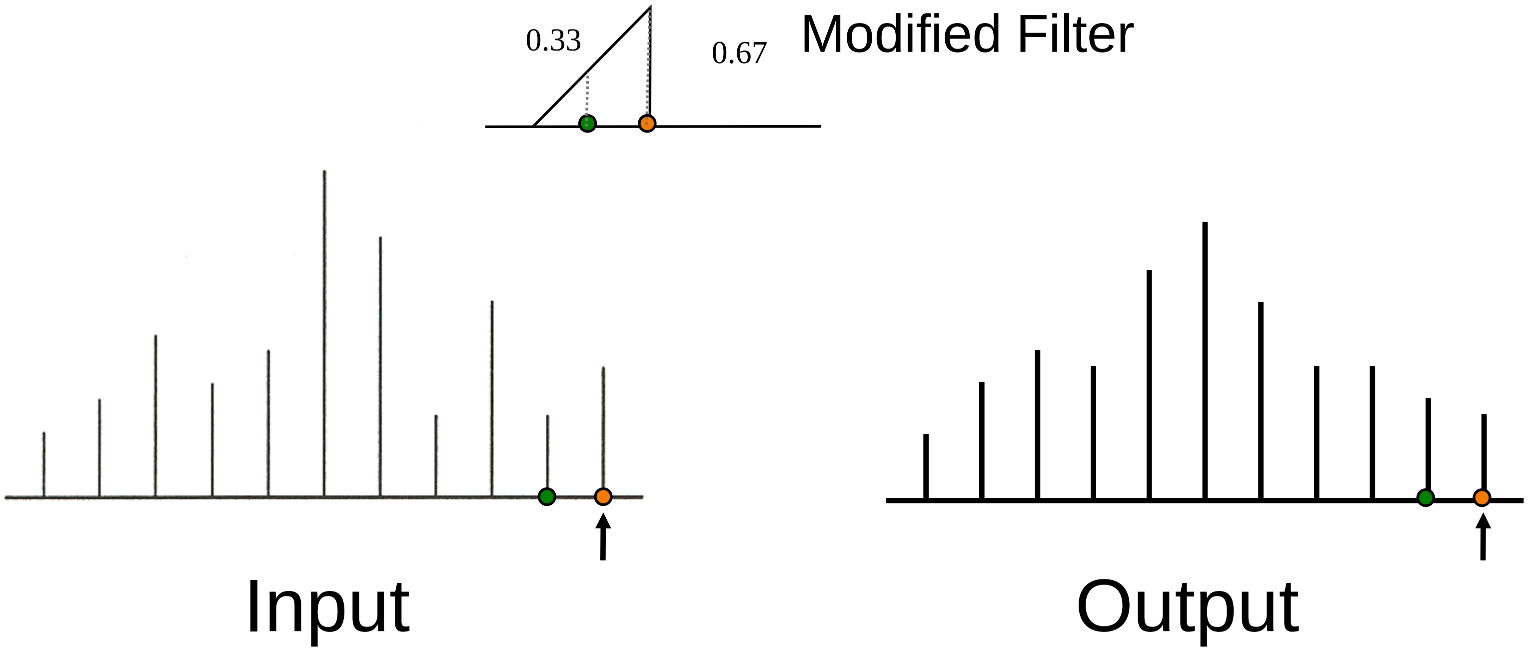
Input



Output

Convolution with a Triangle Filter

Common option: normalize the filter



Convolution with a Gaussian Filter

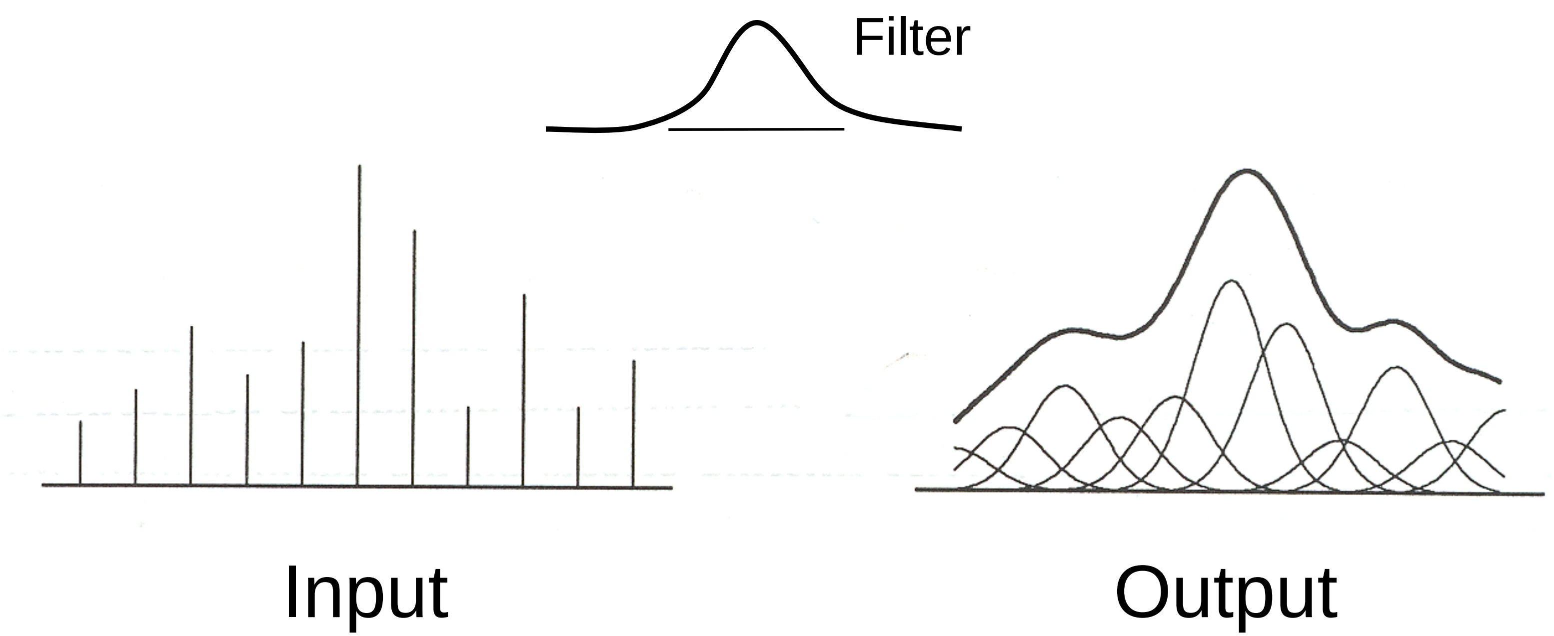
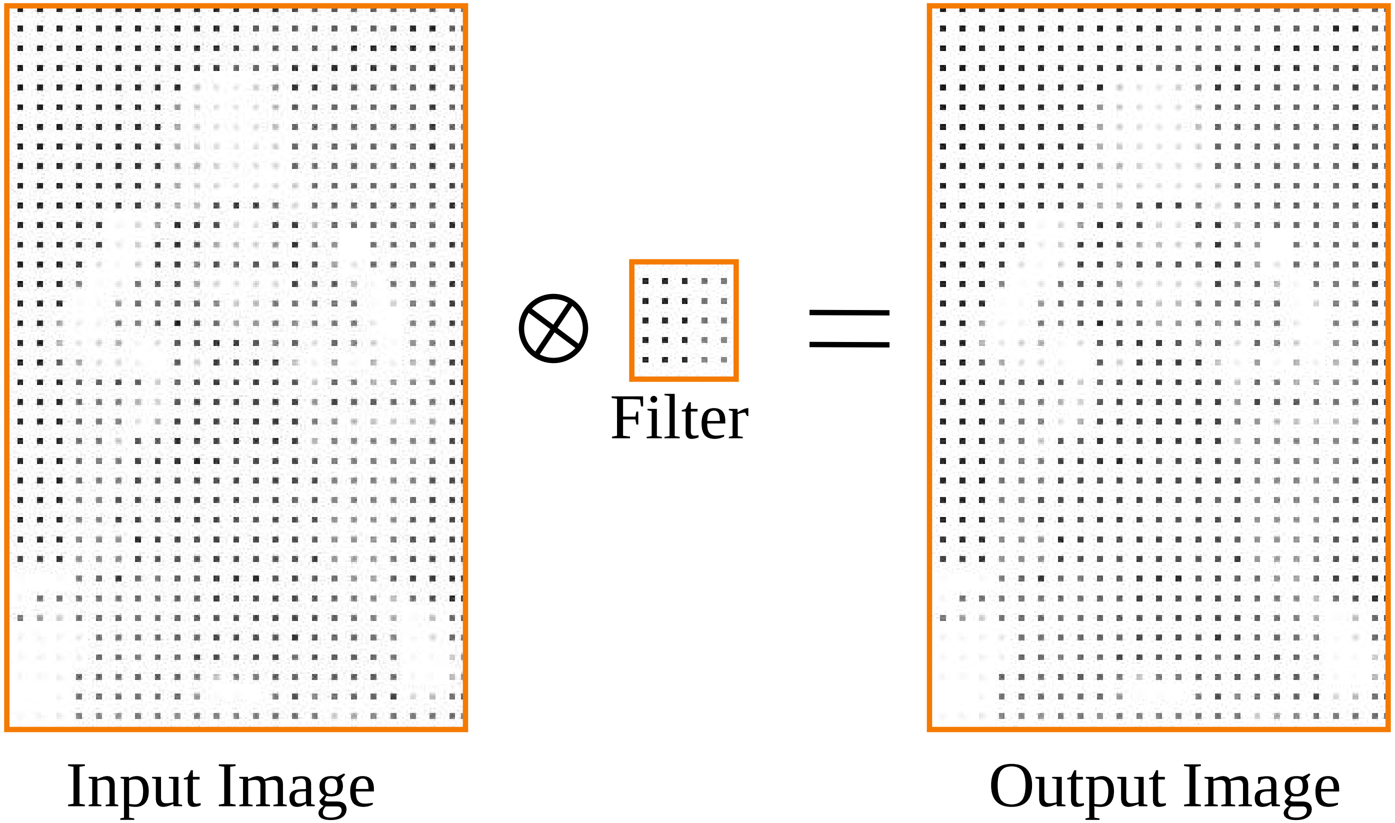


Figure 2.4 Wolberg

Linear Filtering

2D Convolution

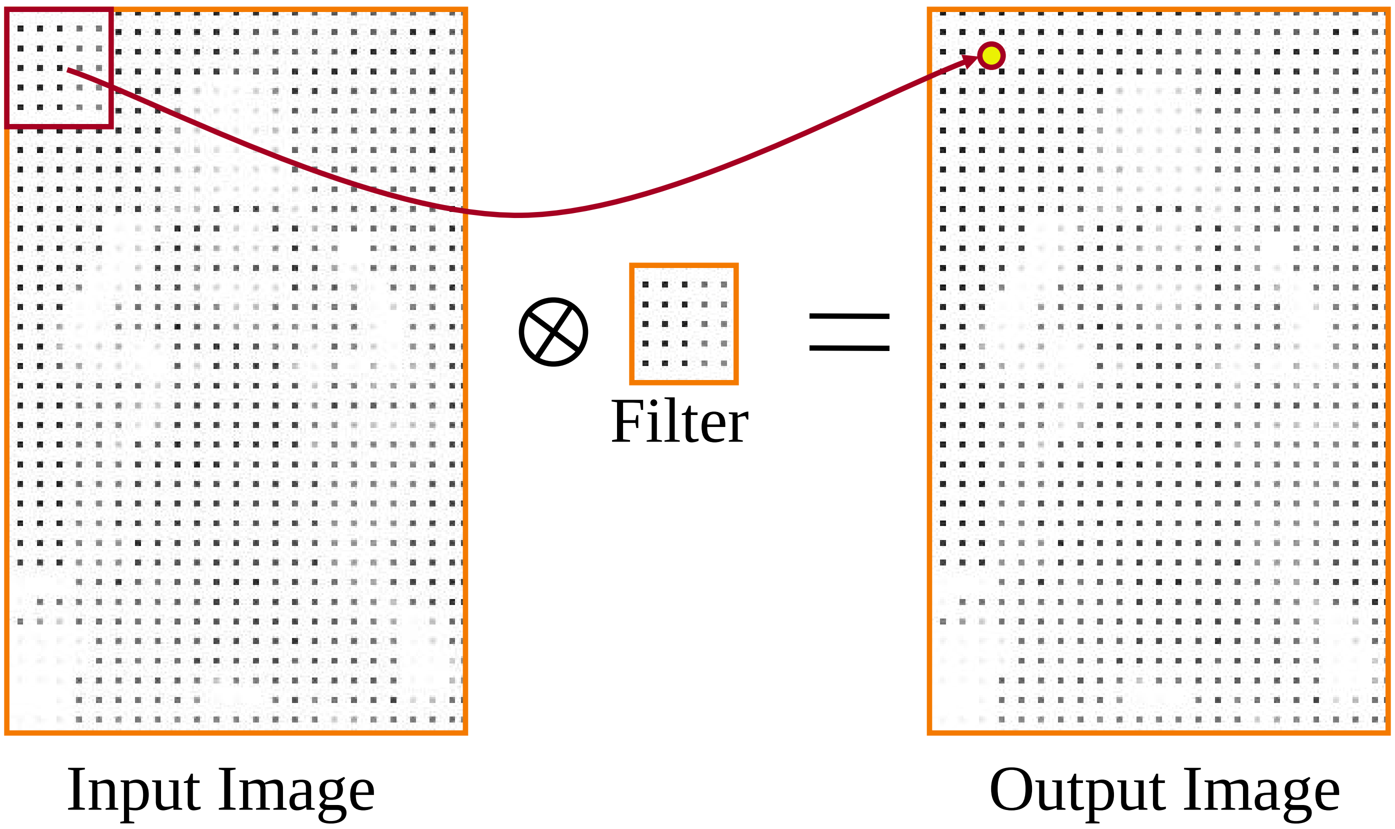
- Each output pixel is a linear combination of input pixels in neighborhood with weights prescribed by a filter



Linear Filtering

2D Convolution

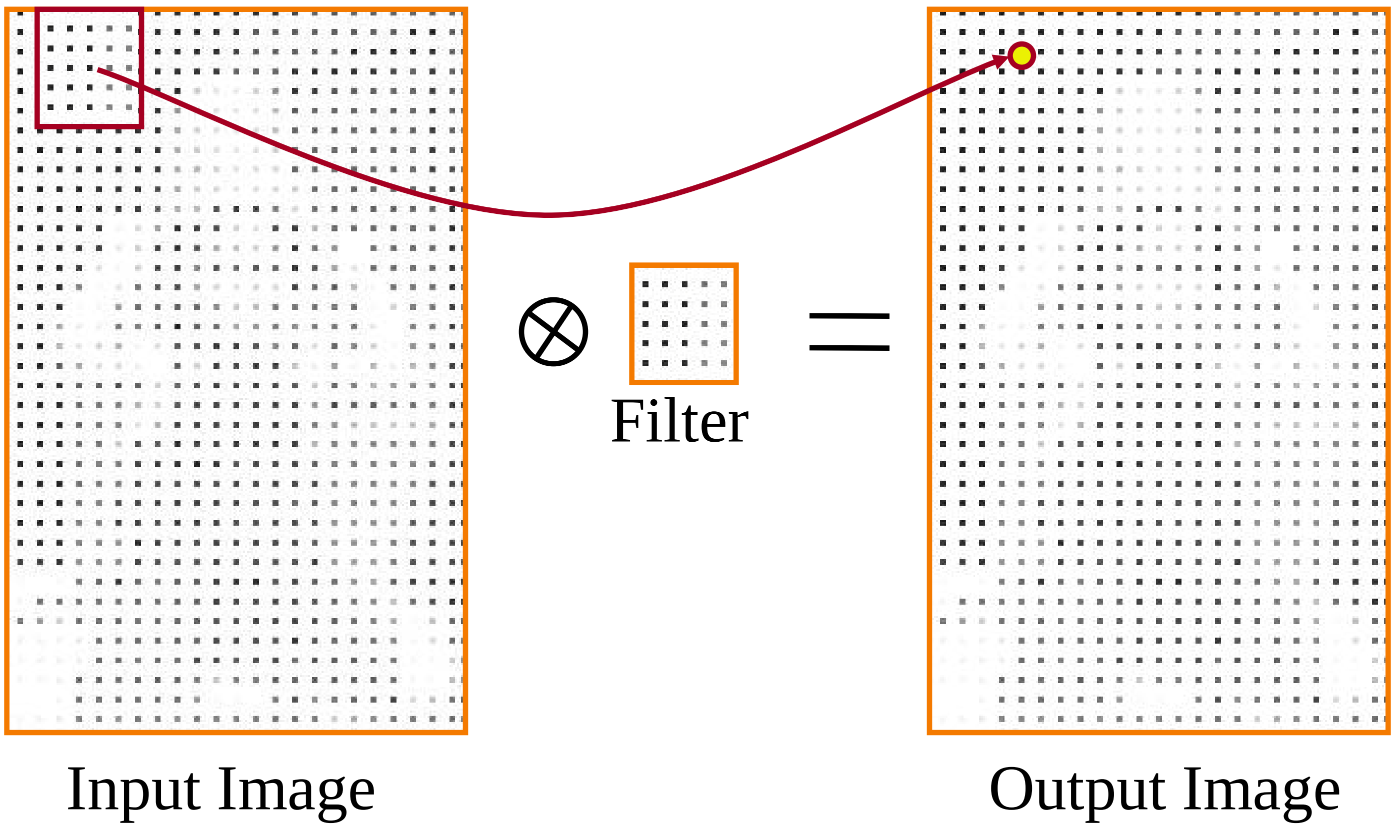
- Each output pixel is a linear combination of input pixels in neighborhood with weights prescribed by a filter



Linear Filtering

2D Convolution

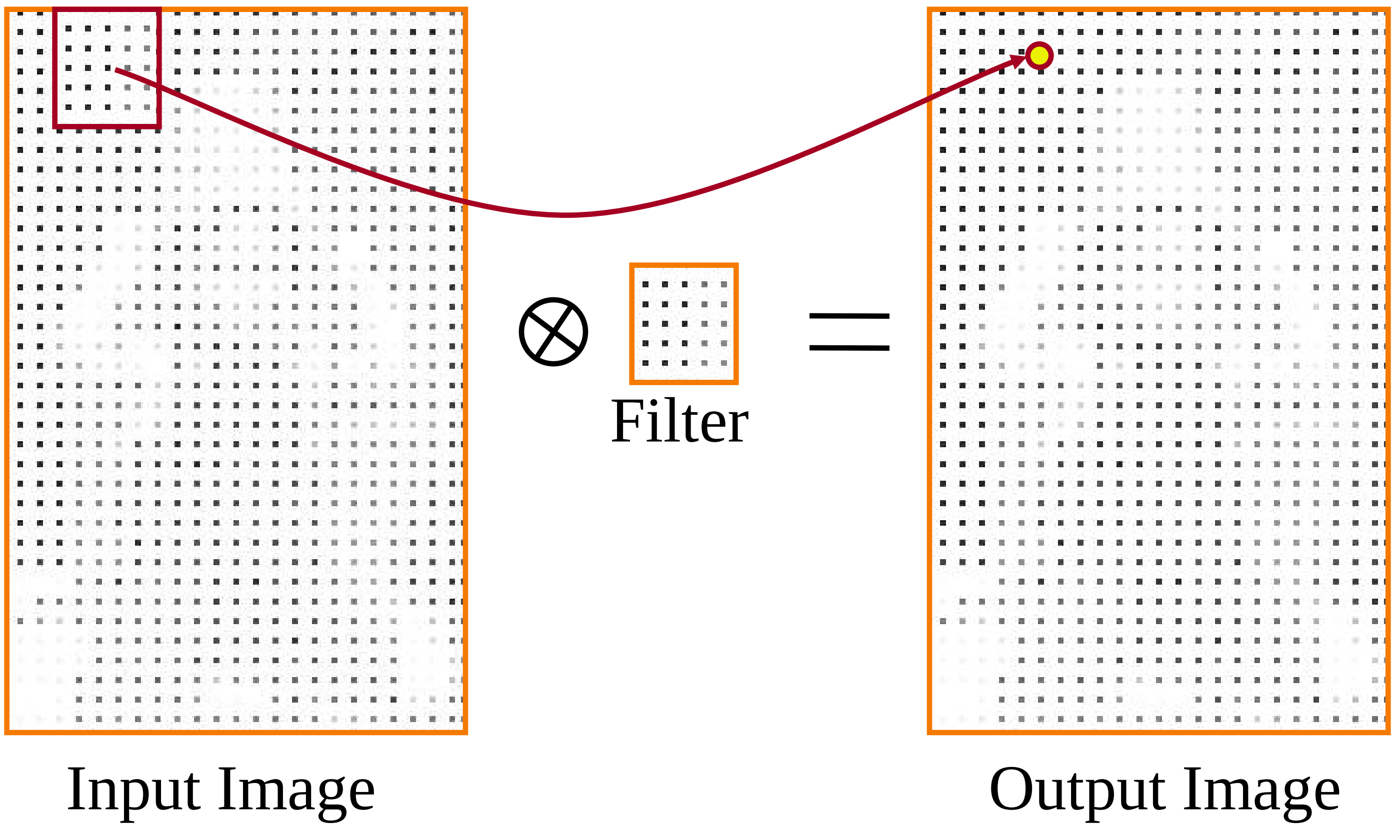
- Each output pixel is a linear combination of input pixels in neighborhood with weights prescribed by a filter



Linear Filtering

2D Convolution

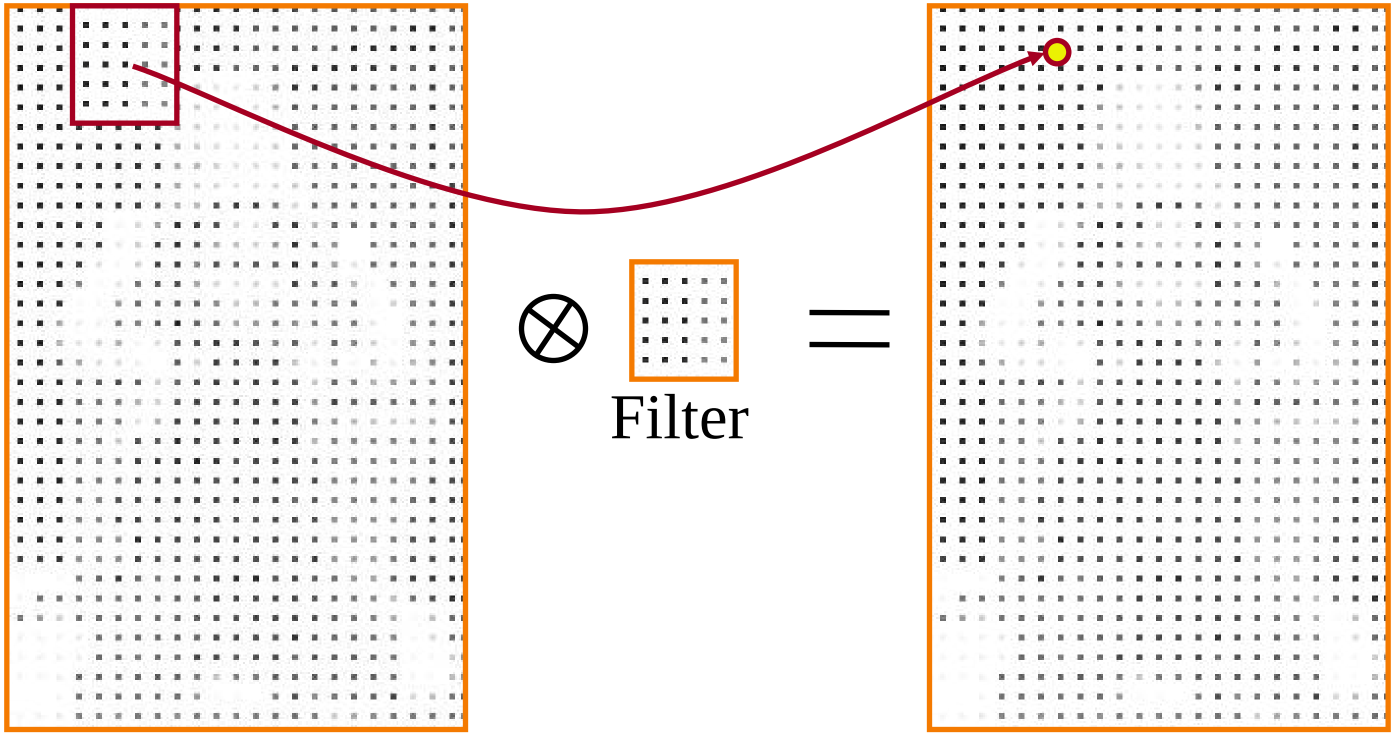
- Each output pixel is a linear combination of input pixels in neighborhood with weights prescribed by a filter



Linear Filtering

2D Convolution

- Each output pixel is a linear combination of input pixels in neighborhood with weights prescribed by a filter



Input Image

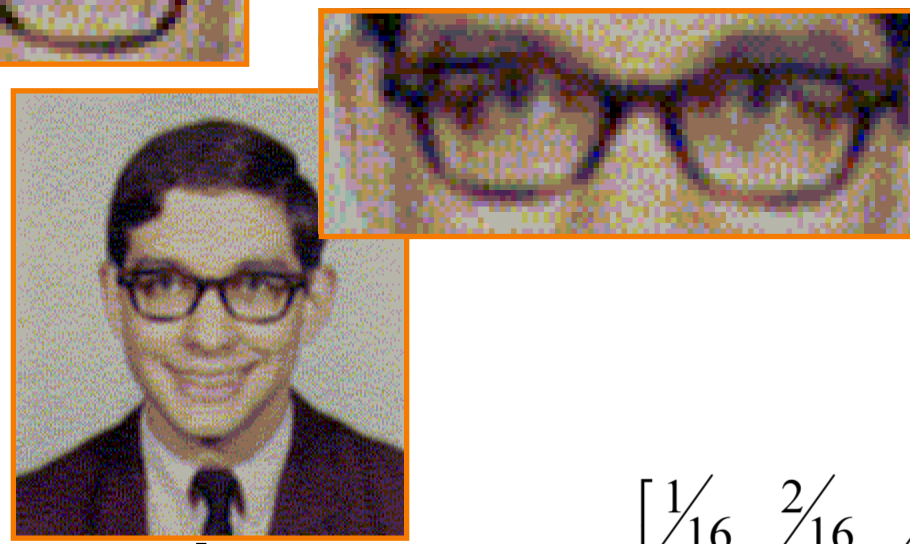
Output Image

Blur

Convolve with a filter whose entries sum to one

- Each pixel becomes a weighted average of its neighbors

Original



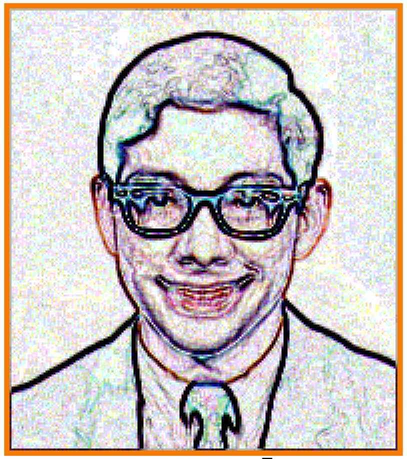
Blur

$$\text{Filter} = \begin{bmatrix} 1/16 & 2/16 & 1/16 \\ 2/16 & 4/16 & 2/16 \\ 1/16 & 2/16 & 1/16 \end{bmatrix}$$

Edge Detection

Convolve with a filter that finds differences between neighbor pixels

Original



Detect edges

$$\text{Filter} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & +8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Sharpen

Sum detected edges with original image

Original

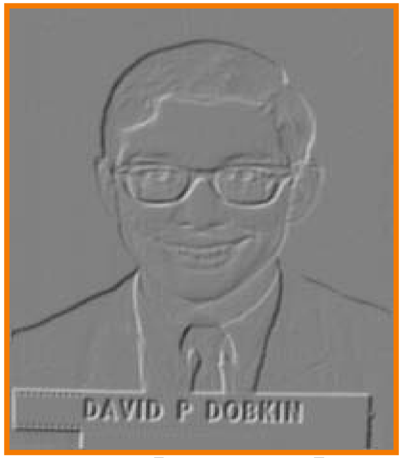
Sharpened

$$\text{Filter} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & +9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Emboss

Convolve with a filter that highlights gradients in particular directions

Original



Embossed

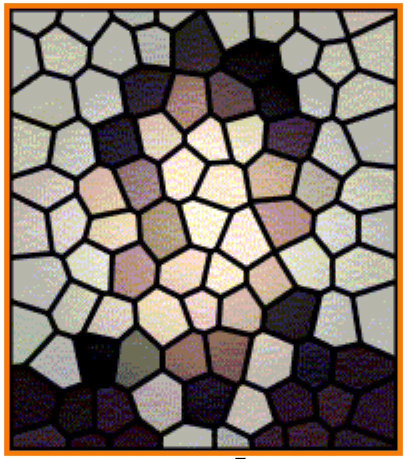
$$\text{Filter} = \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Non-Linear Filtering

Each output pixel is a non-linear function of input pixels in neighborhood (filter depends on input)


Original


Oil


Stain Glass

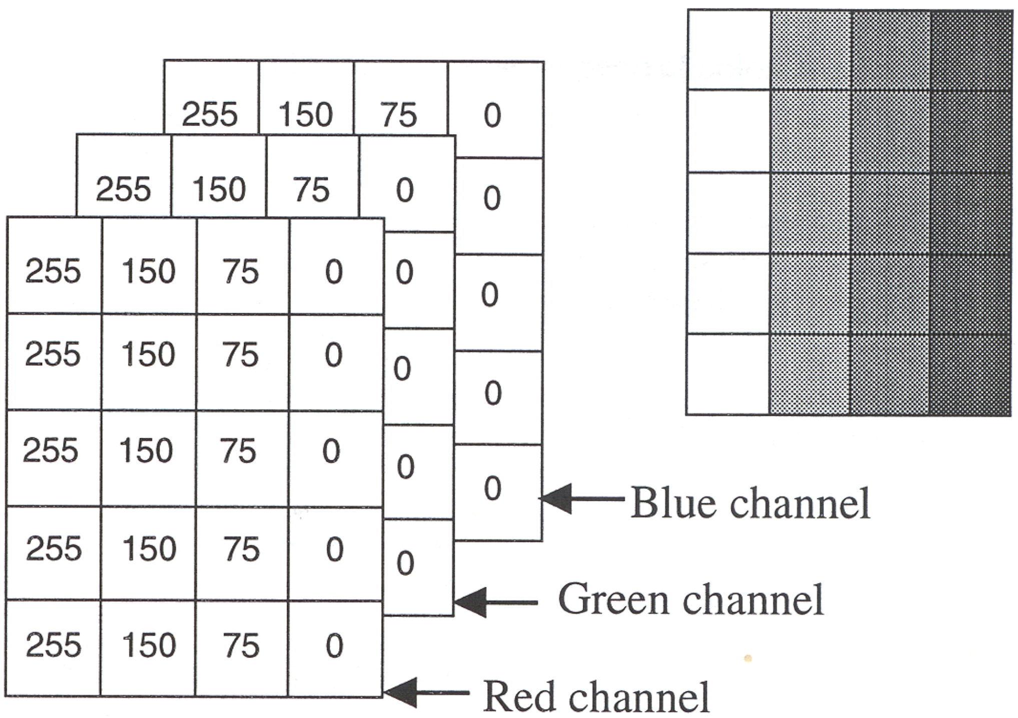
Digital Image Processing

- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph
- Quantization
- Spatial / intensity tradeoff
  - Dithering

Quantization

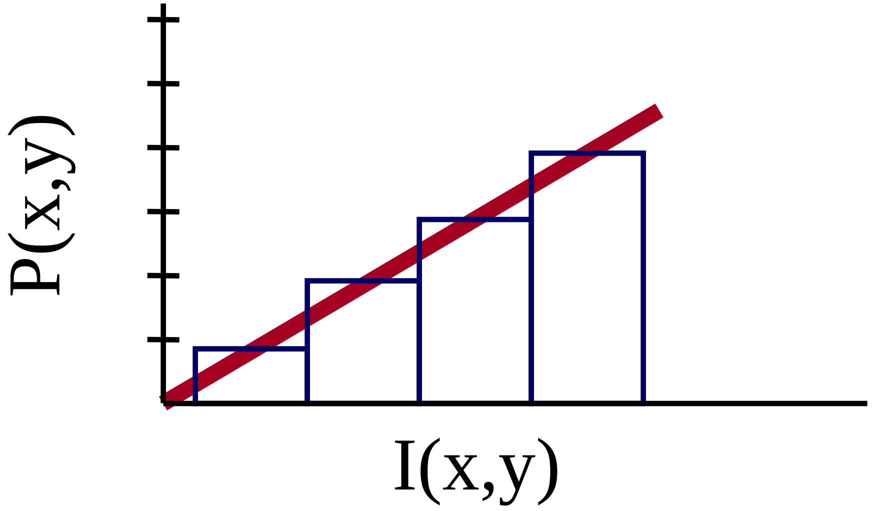
Reduce intensity resolution

- Frame buffers have limited number of bits per pixel
- Physical devices have limited dynamic range



Uniform Quantization

P(x, y) = round(I(x, y))
where round() chooses nearest value that can be represented.

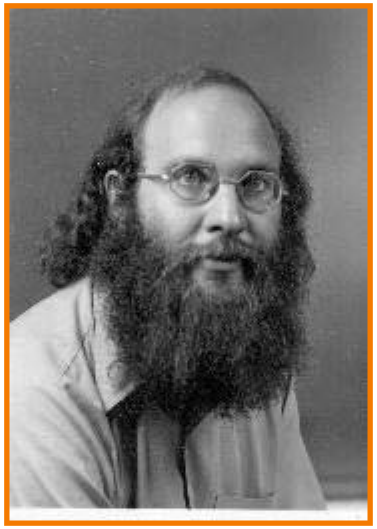


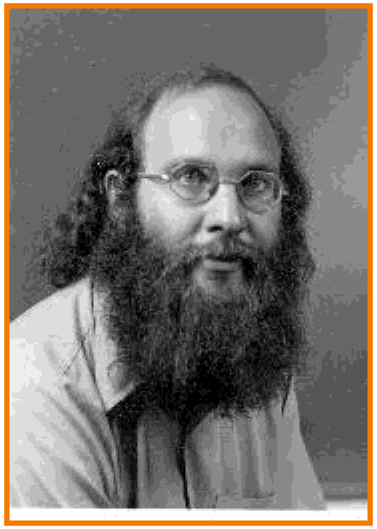
I(x,y)

P(x,y)
(2 bits per pixel)

Uniform Quantization

Images with decreasing bits per pixel:


8 bits


4 bits


2 bits


1 bit

Notice contouring.

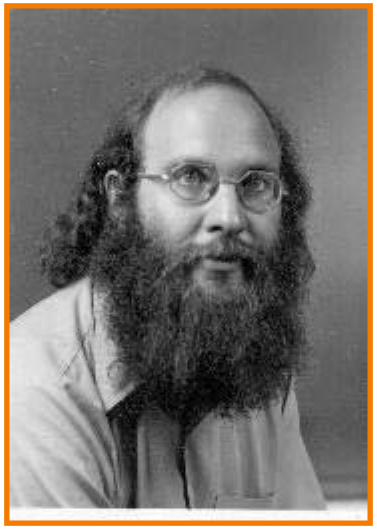
Reducing Effects of Quantization

- Intensity resolution / spatial resolution tradeoff
- Dithering
  - Random dither
  - Ordered dither
  - Error diffusion dither
- Halftoning
  - Classical halftoning

Dithering

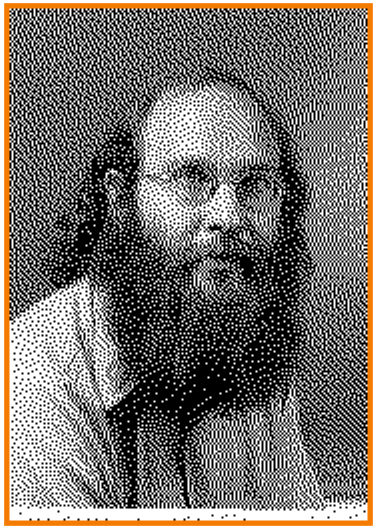
Distribute errors among pixels

- o Exploit spatial integration in our eye
- o Display greater range of perceptible intensities



Original
(8 bits)

Uniform
Quantization
(1 bit)

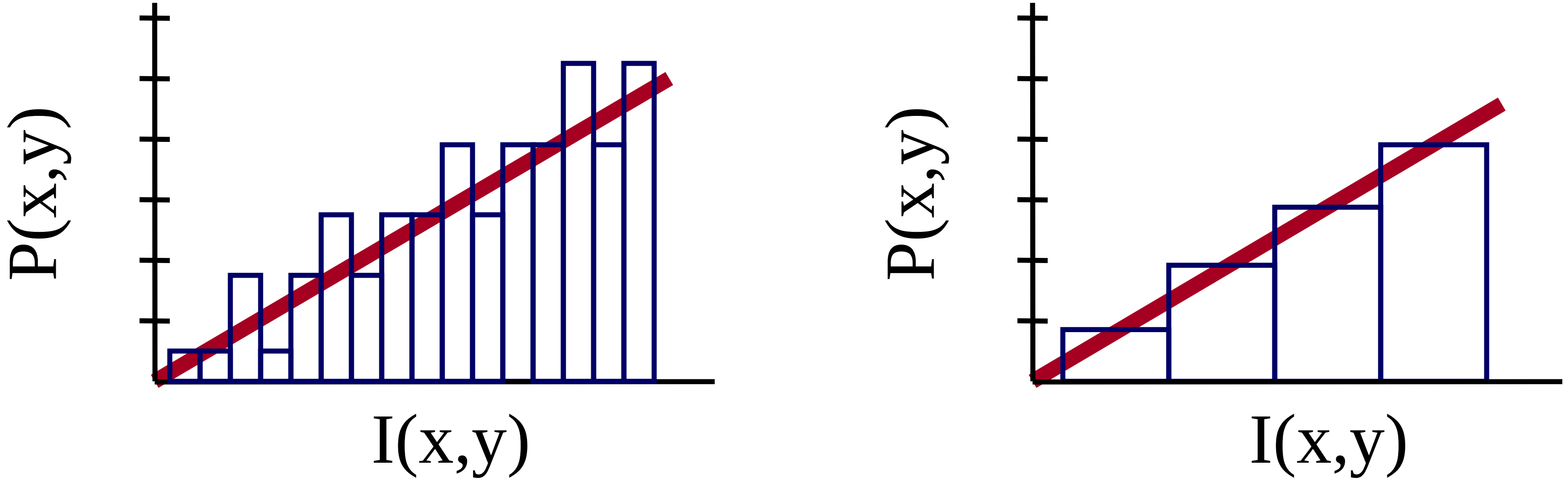


Floyd-Steinberg
Dither
(1 bit)

Random Dither

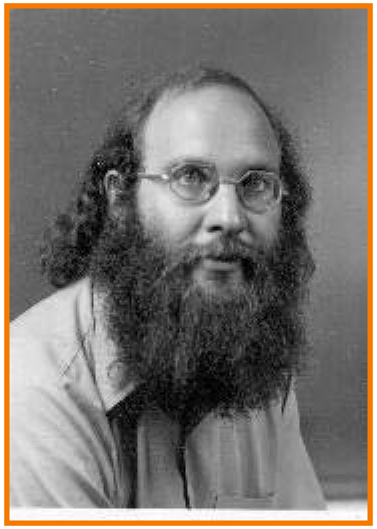
Randomize quantization errors

- Errors appear as noise



$$P(x, y) = \text{round}(\ I(x, y) + \text{noise}(x,y)\ )$$

Random Dither


Original
(8 bits)


Uniform
Quantization
(1 bit)


Random
Dither
(1 bit)

Ordered Dither

Pseudo-random quantization errors

- Matrix stores pattern of threshholds

i = x mod n

j = y mod n

e = I(x,y) - trunc(I(x,y))

threshold = $(D(i,j)+1)/(n^2+1)$

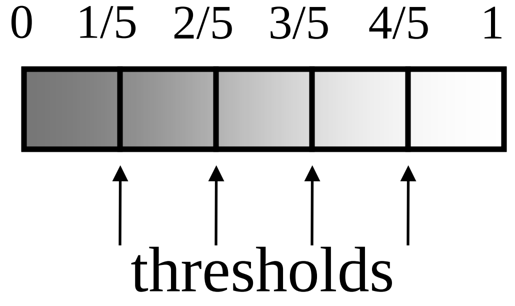
if (e > threshold)

    P(x,y) = ceil(I(x, y))

else

    P(x,y) = floor(I(x,y))

$$D_2 = \begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}$$



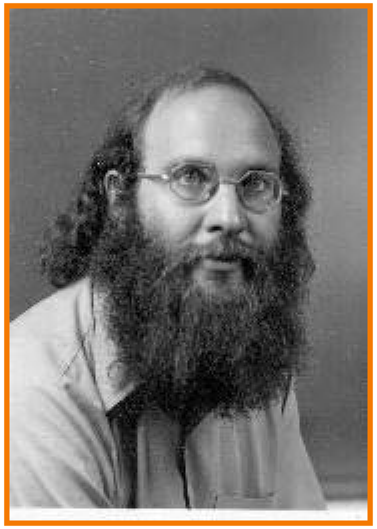
Ordered Dither

Bayer's ordered dither matrices

$$D_n = \begin{bmatrix} 4D_{n/2} + D_2(1,1)U_{n/2} & 4D_{n/2} + D_2(1,2)U_{n/2} \\ 4D_{n/2} + D_2(2,1)U_{n/2} & 4D_{n/2} + D_2(2,2)U_{n/2} \end{bmatrix}$$

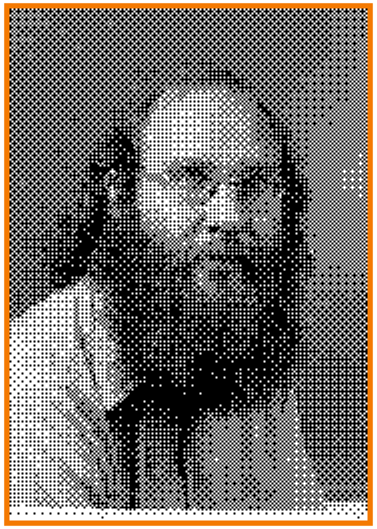
$$D_2 = \begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \qquad D_4 = \left[\begin{array}{cc|cc} 15 & 7 & 13 & 5 \\ 3 & 11 & 1 & 9 \\ \hline 12 & 4 & 14 & 6 \\ 0 & 8 & 2 & 10 \end{array}\right]$$

Ordered Dither



Original
(8 bits)

Random
Dither
(1 bit)

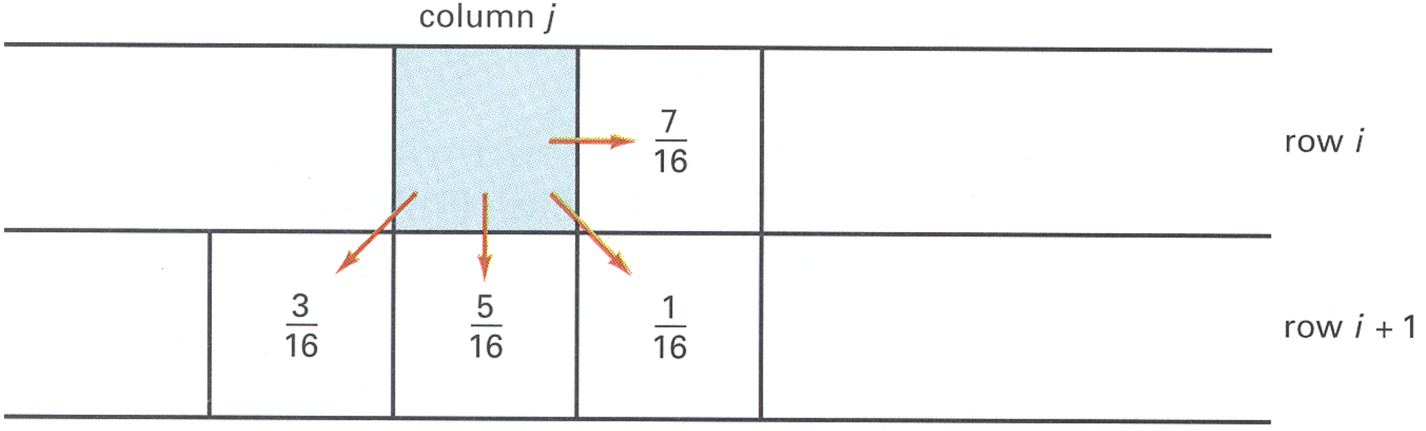


Ordered
Dither
(1 bit)

Error Diffusion Dither

Spread quantization error over neighbor pixels

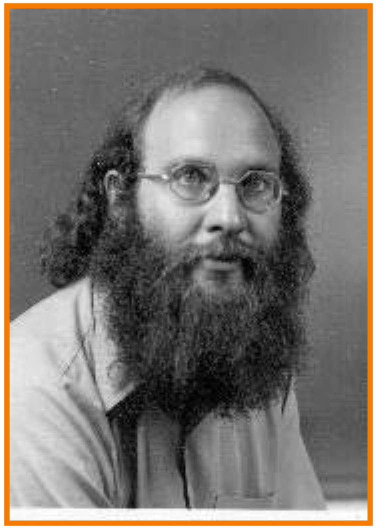
- Error dispersed to pixels right and below
- Floyd-Steinberg weights:



$$3/16 + 5/16 + 1/16 + 7/16 = 1.0$$

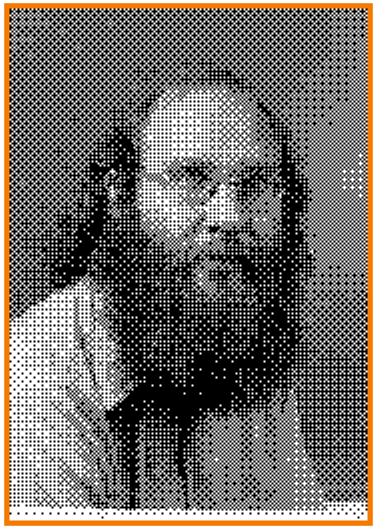
Figure 14.42 from H&B

Error Diffusion Dither

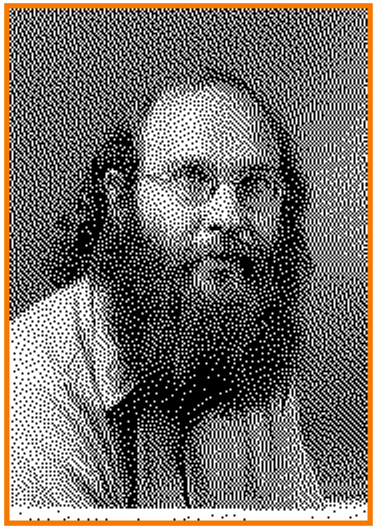


Original
(8 bits)

Random
Dither
(1 bit)



Ordered
Dither
(1 bit)



Floyd-Steinberg
Dither
(1 bit)

Reducing Effects of Quantization

- Dithering
  - Random dither
  - Ordered dither
  - Error diffusion dither

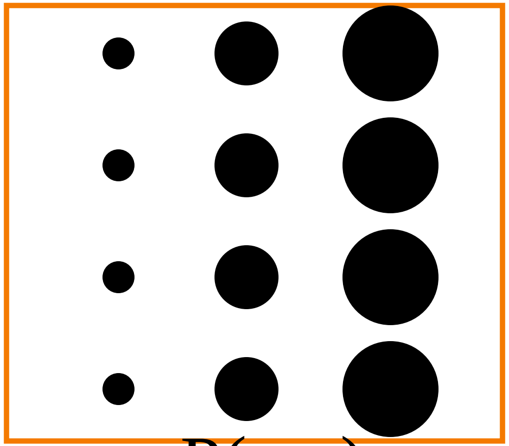
- Halftoning
  - Classical halftoning

Classical Halftoning

Use dots of varying size to represent intensities

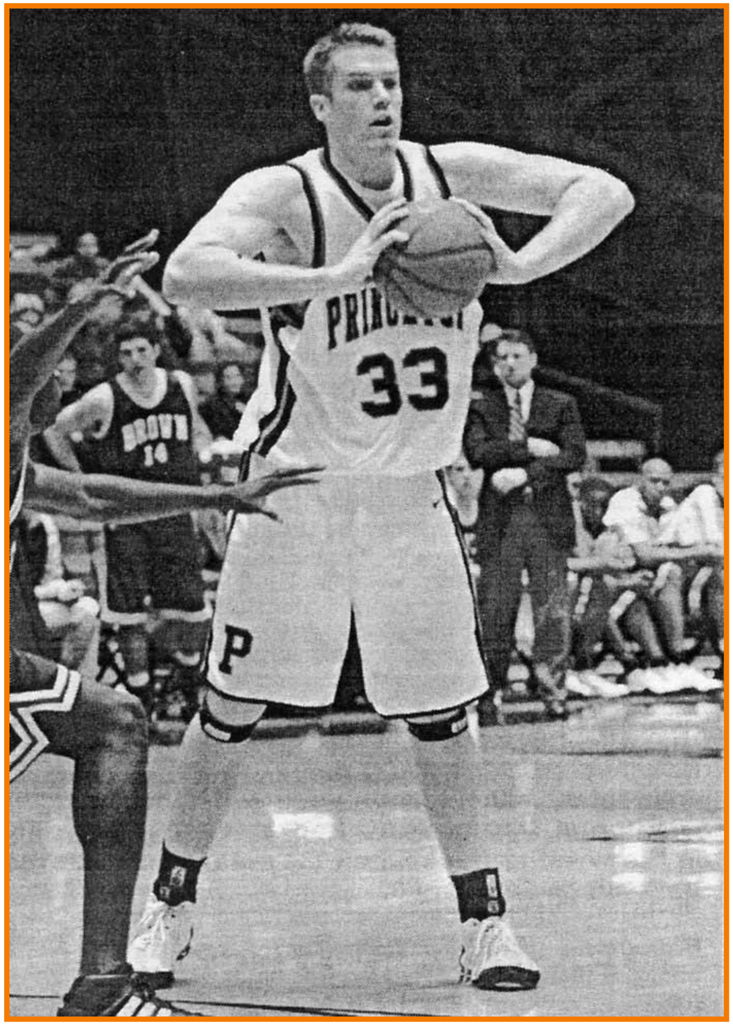
- Area of dots proportional to intensity in image

I(x,y)

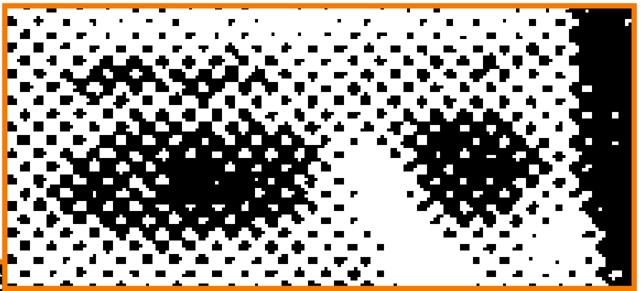


P(x,y)

Classical Halftoning

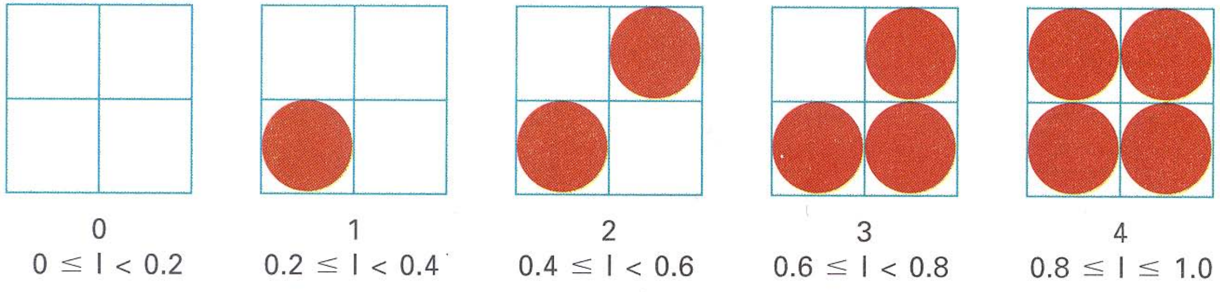


From *Town Topics*, Princeton



Digital Halftone Patterns

Use cluster of pixels to represent intensity



Q: In this case, would we use four "halftoned" pixels in place of one original pixel?

Figure 14.37 from H&B

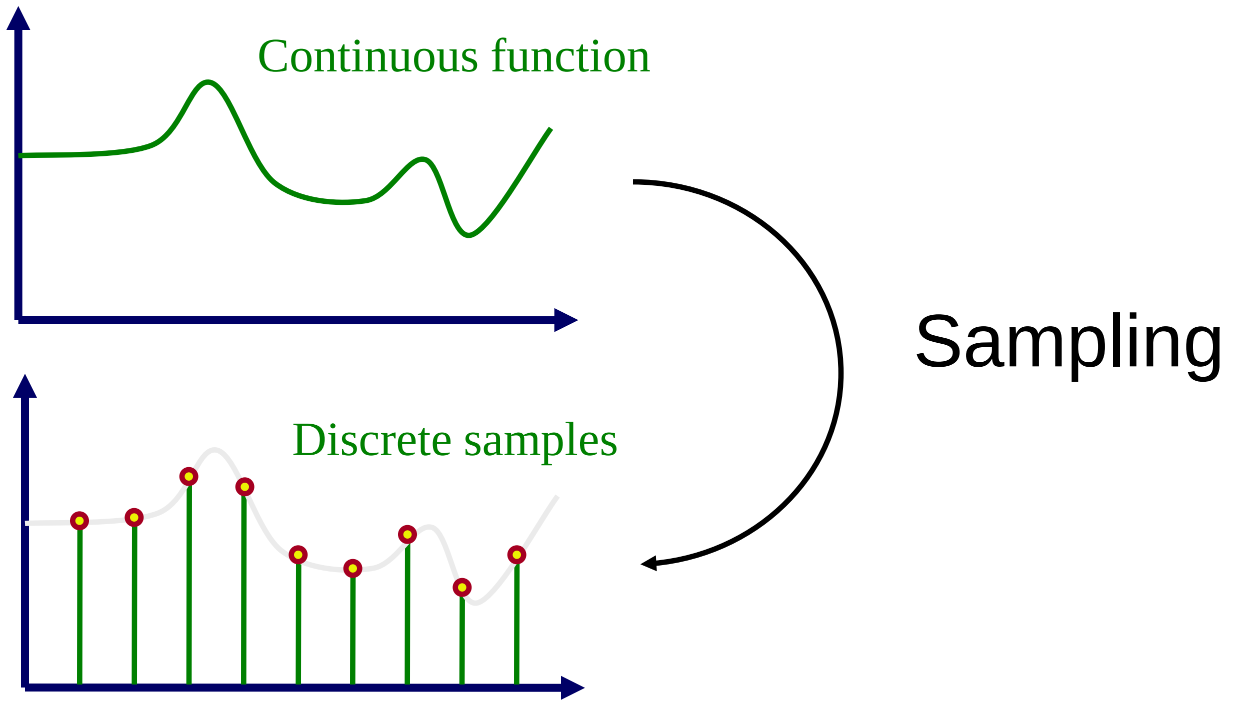
Digital Image Processing

- Changing intensity/color
  - Linear: scale, offset, etc.
  - Nonlinear: gamma, saturation, etc.
  - Add random noise
- Filtering over neighborhoods
  - Blur
  - Detect edges
  - Sharpen
  - Emboss
  - Median
- Moving image locations
  - Scale
  - Rotate
  - Warp
- Combining images
  - Composite
  - Morph
- Quantization
- Spatial / intensity tradeoff
  - Dithering

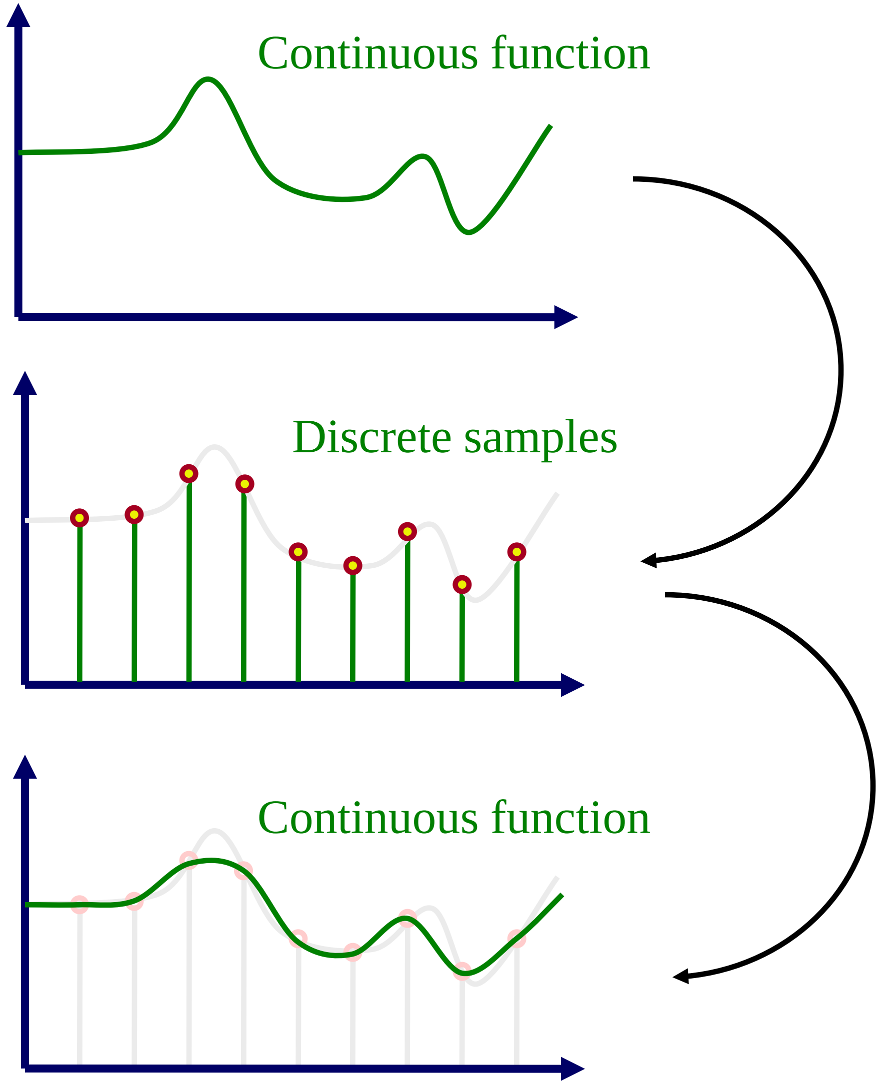
Digital Image Processing

When implementing operations that move pixels, must account for the fact that digital images are sampled versions of continuous ones

Sampling and Reconstruction



Sampling and Reconstruction



Sampling

Reconstruction

Sampling and Reconstruction

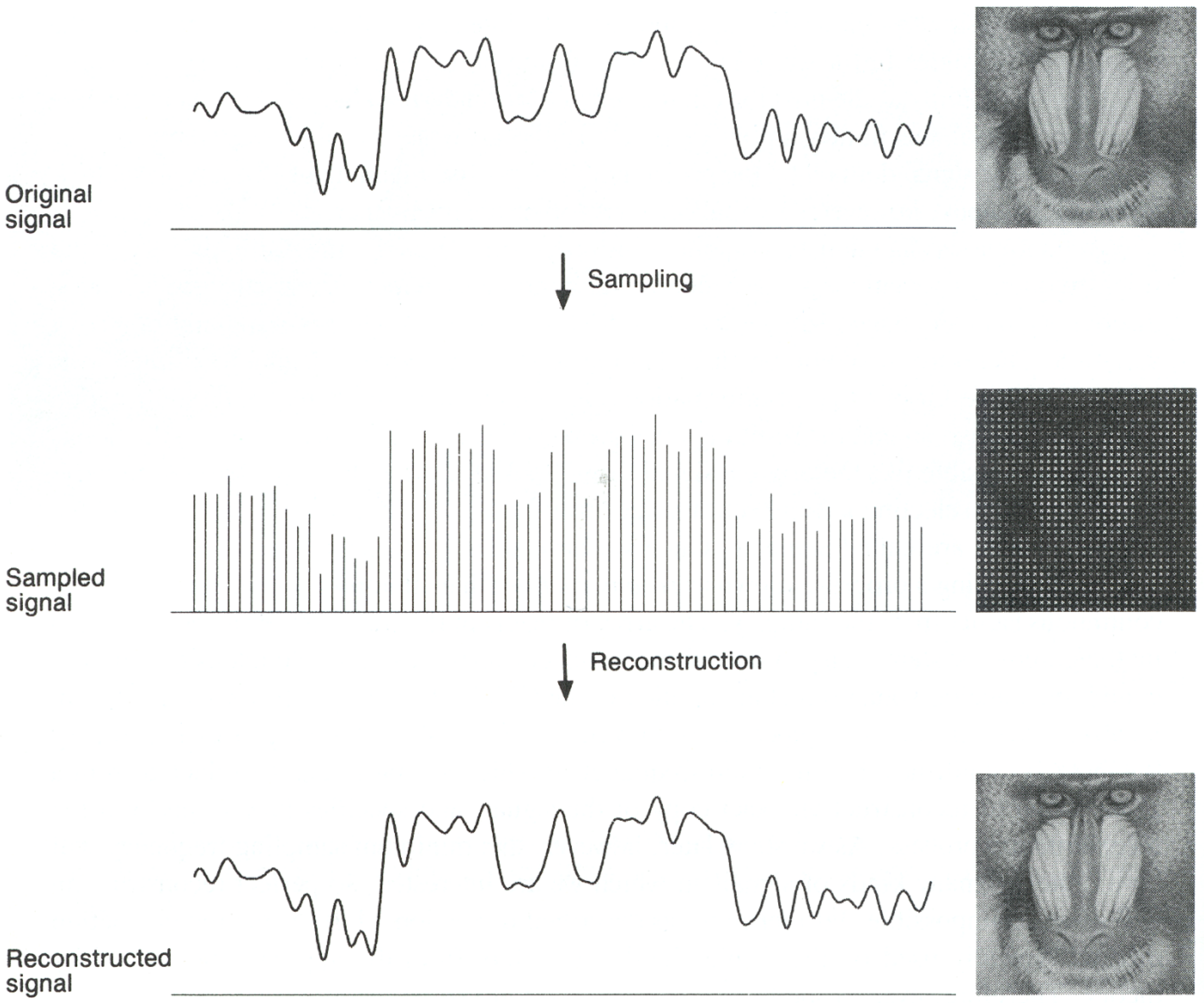
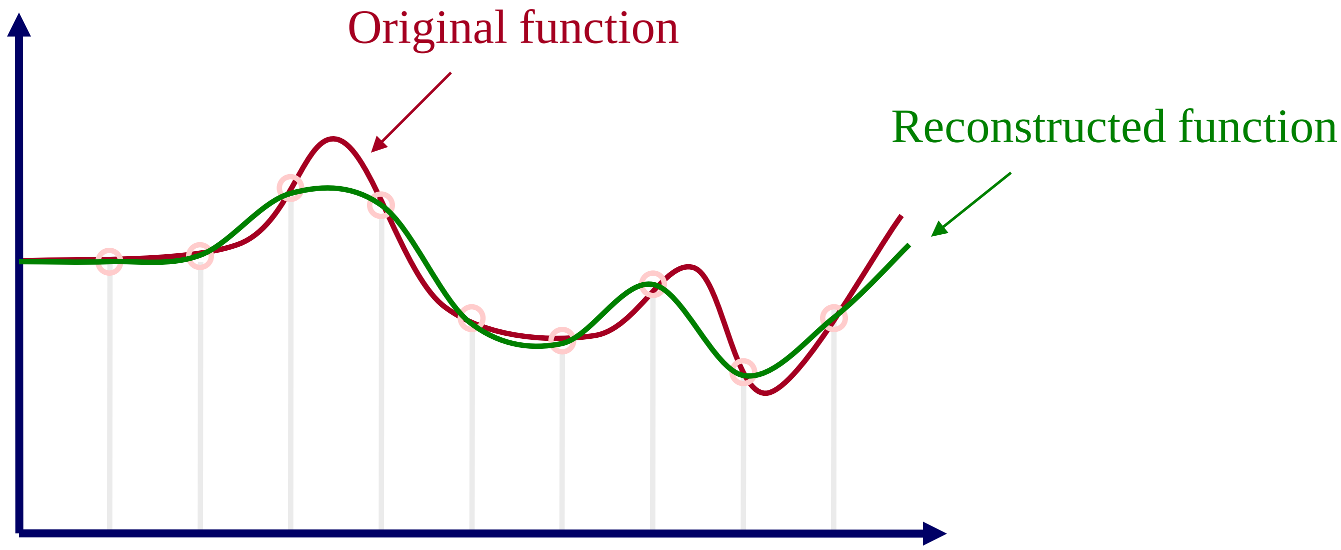


Figure 19.9 FvDFH

Sampling Theory

How many samples are enough?

- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Sampling Theory

What happens when use too few samples?

- Aliasing

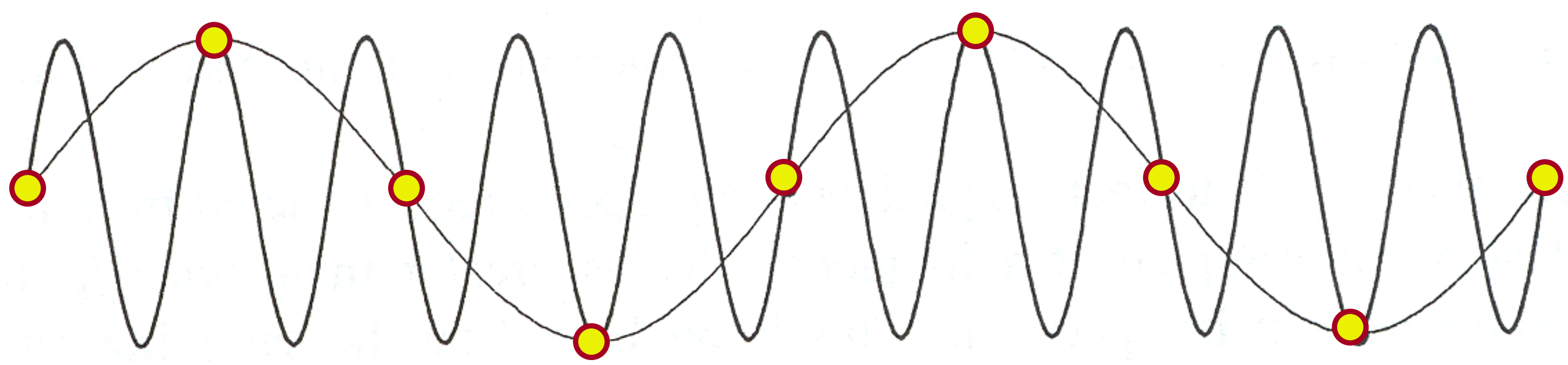
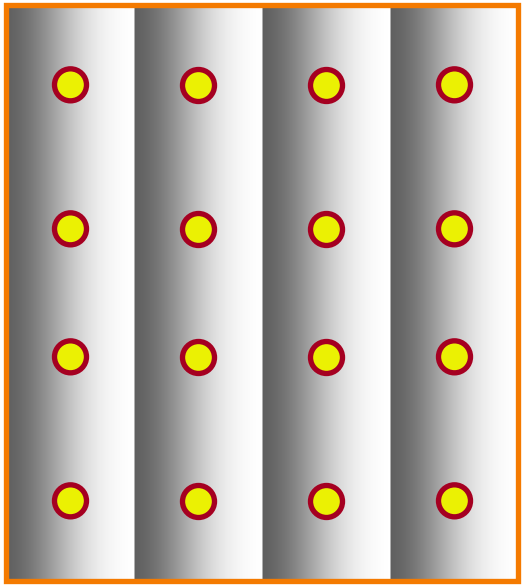


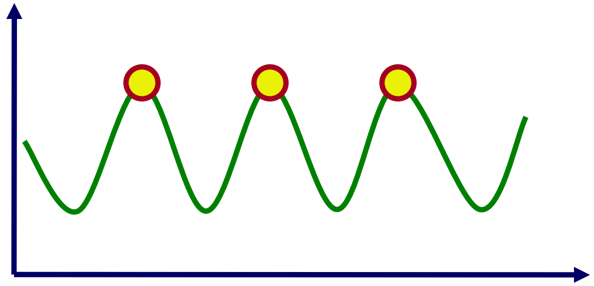
Figure 14.17 FvDFH

Sampling Theory

What happens when use too few samples?

- Aliasing

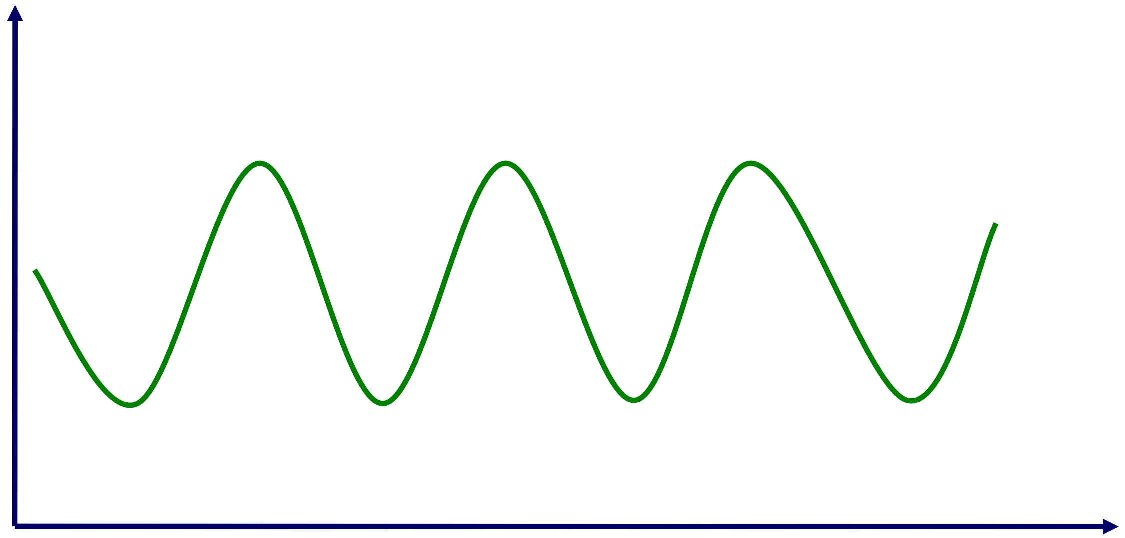




Sampling Theory

How many samples are enough to avoid aliasing?

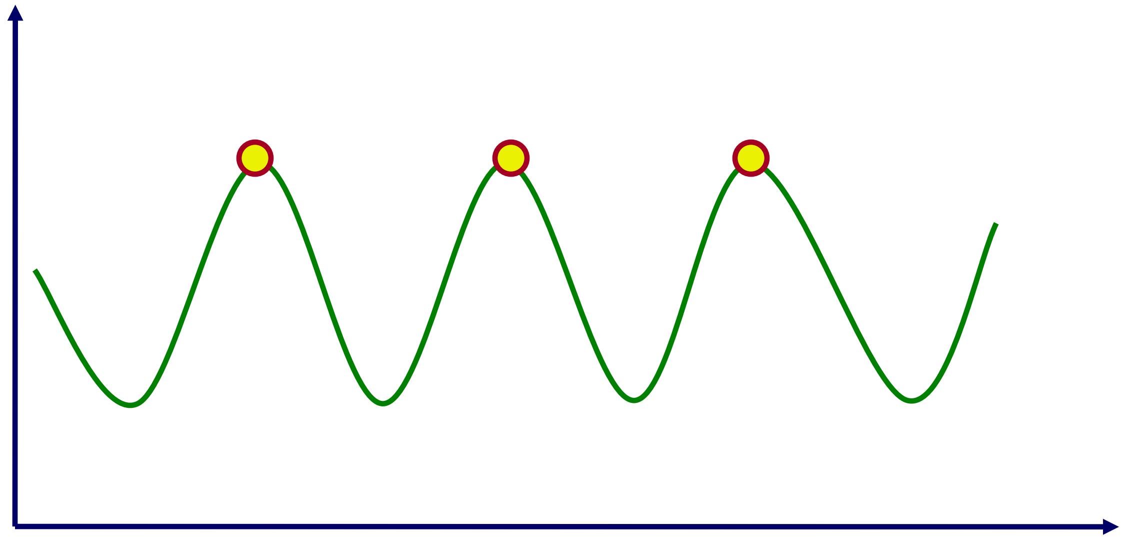
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Sampling Theory

How many samples are enough to avoid aliasing?

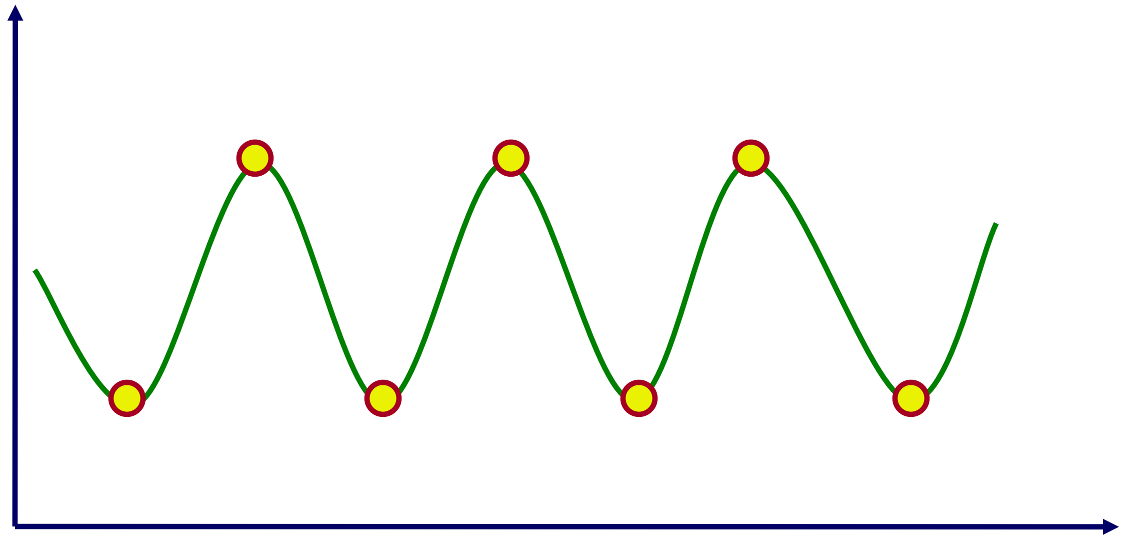
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Sampling Theory

How many samples are enough to avoid aliasing?

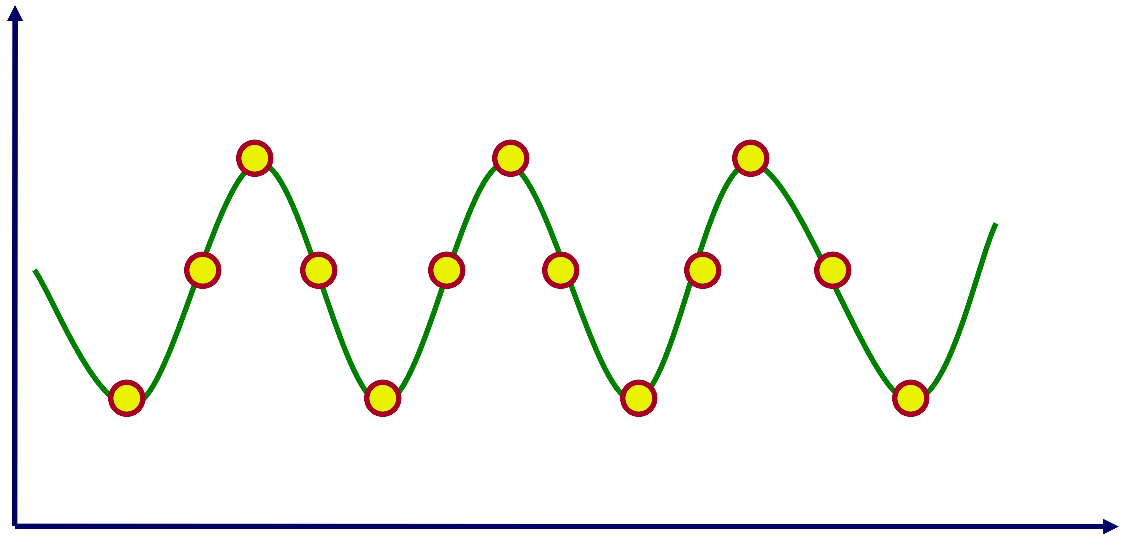
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Sampling Theory

How many samples are enough to avoid aliasing?

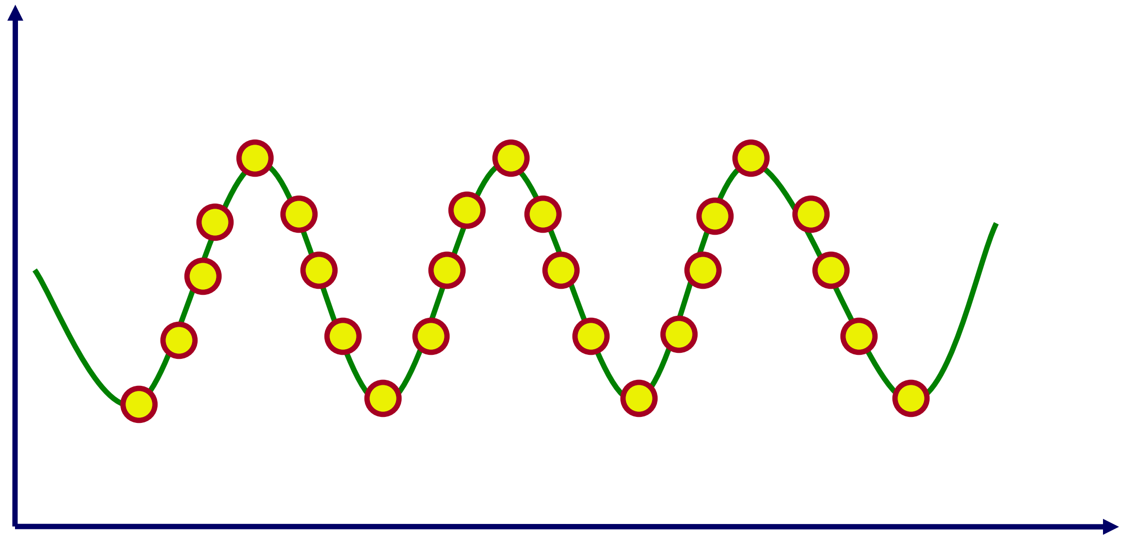
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Sampling Theory

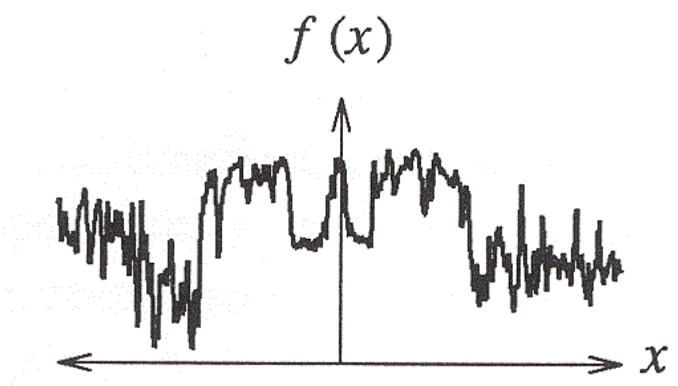
How many samples are enough to avoid aliasing?

- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?

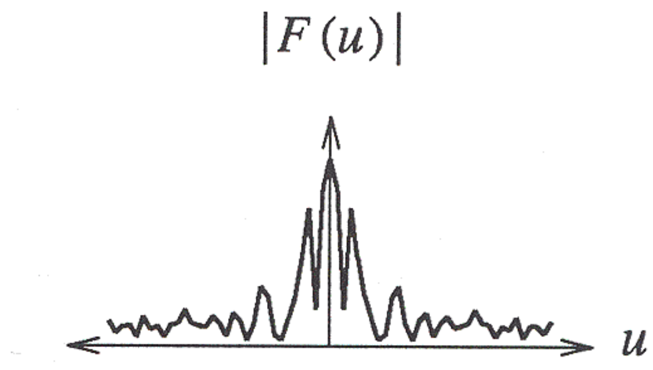


Spectral Analysis

- Spatial domain:
  - o Function: f(x)
  - o Filtering: convolution

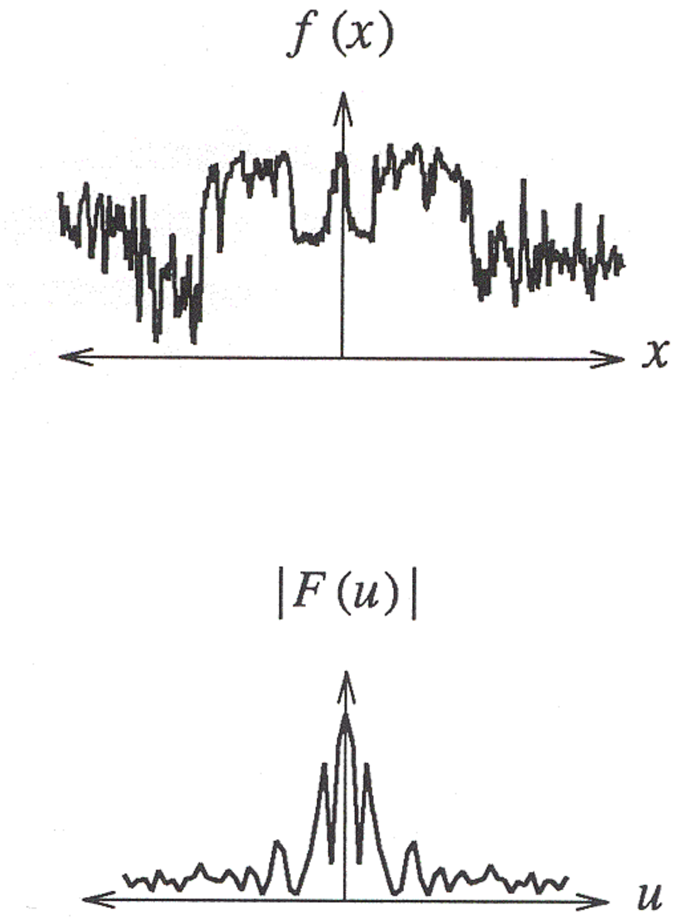


- Frequency domain:
  - o Function: F(u)
  - o Filtering: multiplication



Any signal can be written as a sum of periodic functions.

Fourier Transform



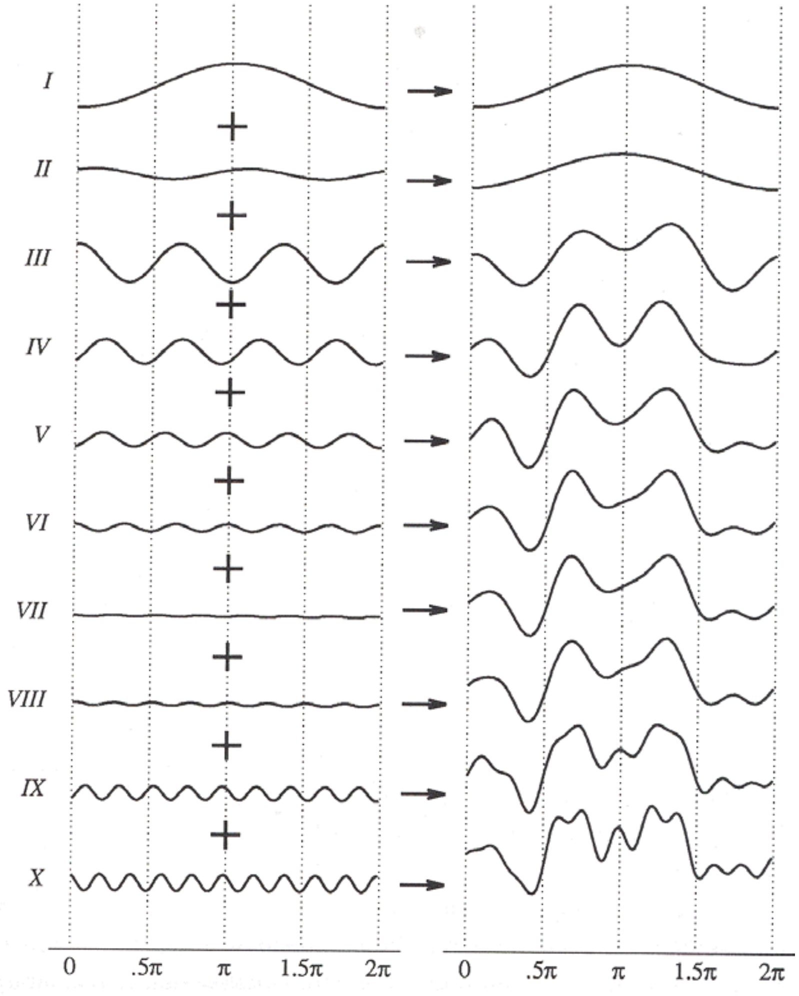


Figure 2.6 Wolberg

Fourier Transform

- Fourier transform:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi xu} dx$$

- Inverse Fourier transform:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{+i2\pi ux} du$$

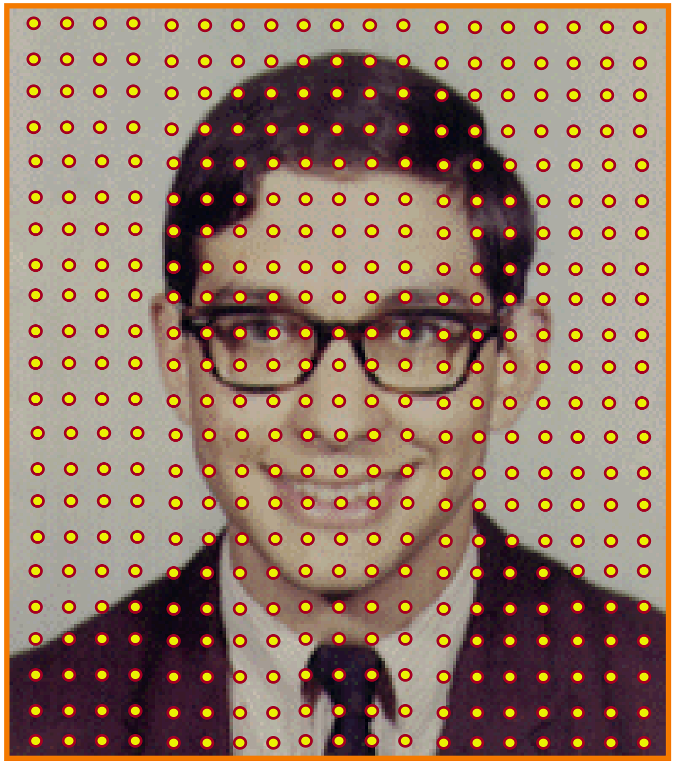
Sampling Theorem

- A signal can be reconstructed from its samples, if the original signal has no frequencies above 1/2 the sampling frequency - Shannon
- The minimum sampling rate for bandlimited function is called "Nyquist rate"

A signal is bandlimited if its highest frequency is bounded. The frequency is called the bandwidth.

Image Processing

- Consider reducing the image resolution


Original image

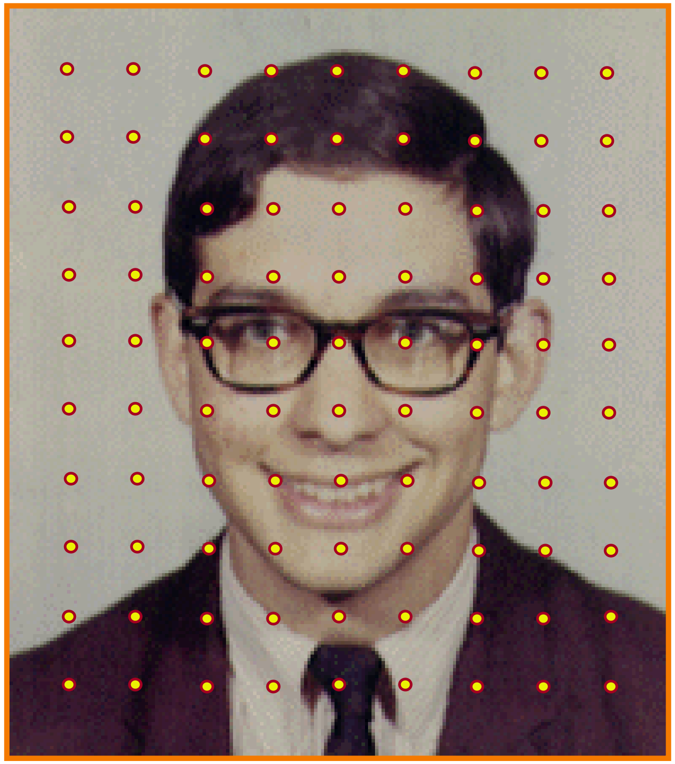
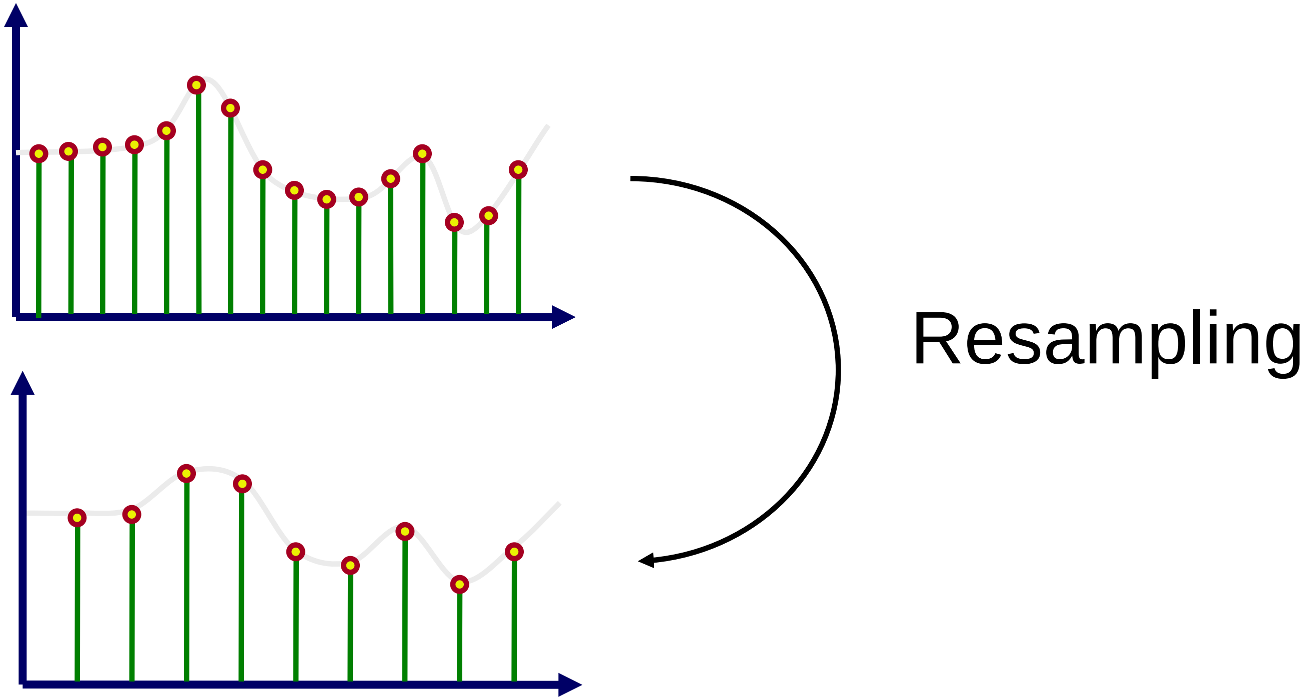

1/4 resolution

Image Processing

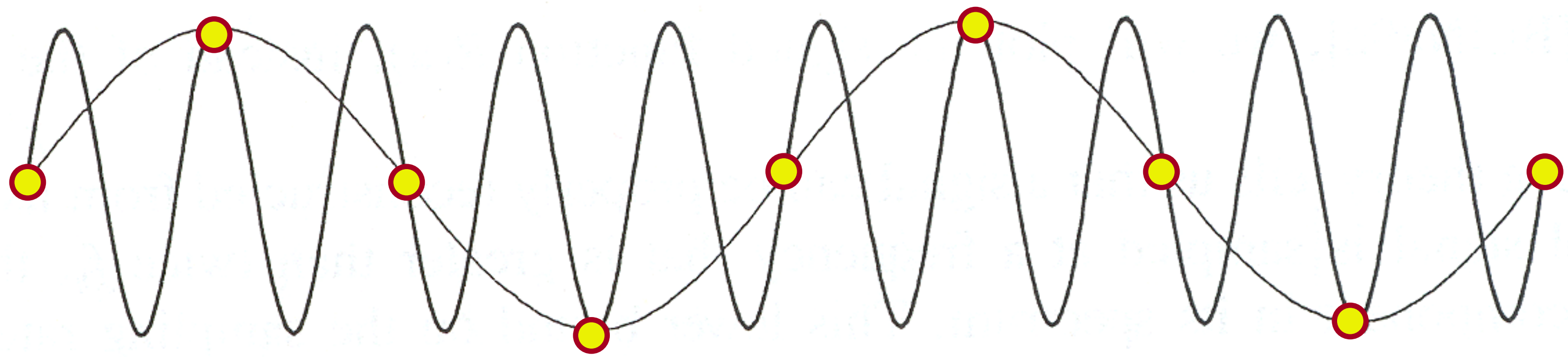
- Image processing is a resampling problem



Sampling Theorem

- A signal can be reconstructed from its samples, if the original signal has no frequencies above 1/2 the sampling frequency - Shannon

Aliasing will occur if the signal is under-sampled

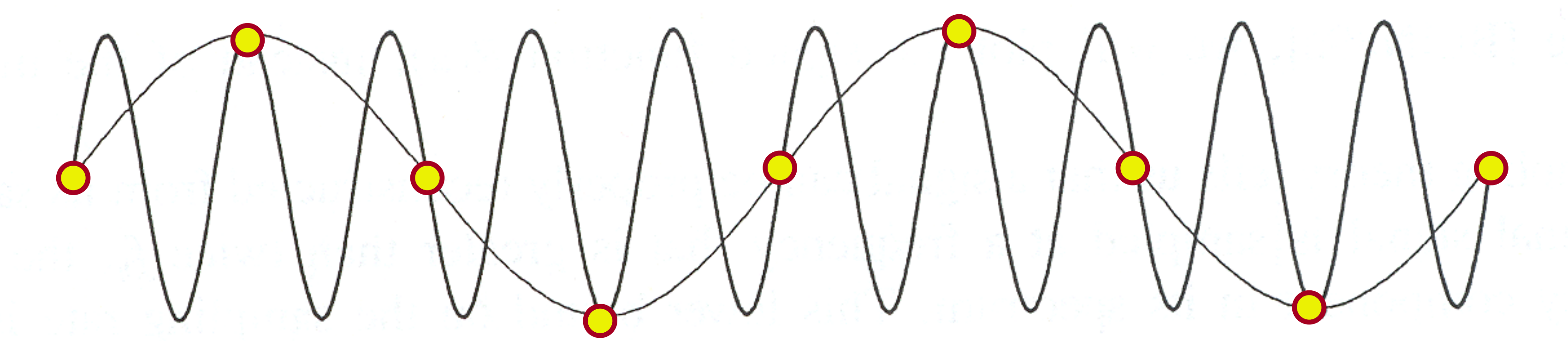


Under-sampling

Figure 14.17 FvDFH

Aliasing

- In general:
  - o Artifacts due to under-sampling or poor reconstruction
- Specifically, in graphics:
  - o Spatial aliasing
  - o Temporal aliasing

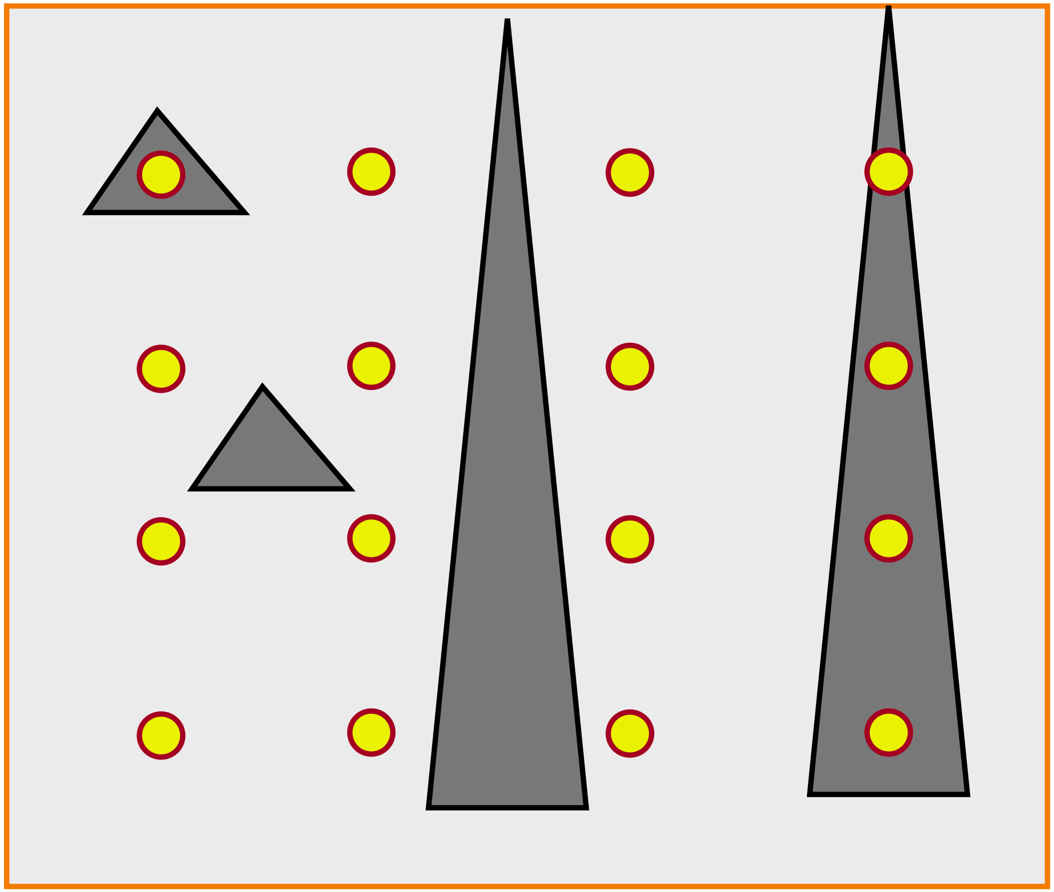


Under-sampling

Figure 14.17 FvDFH

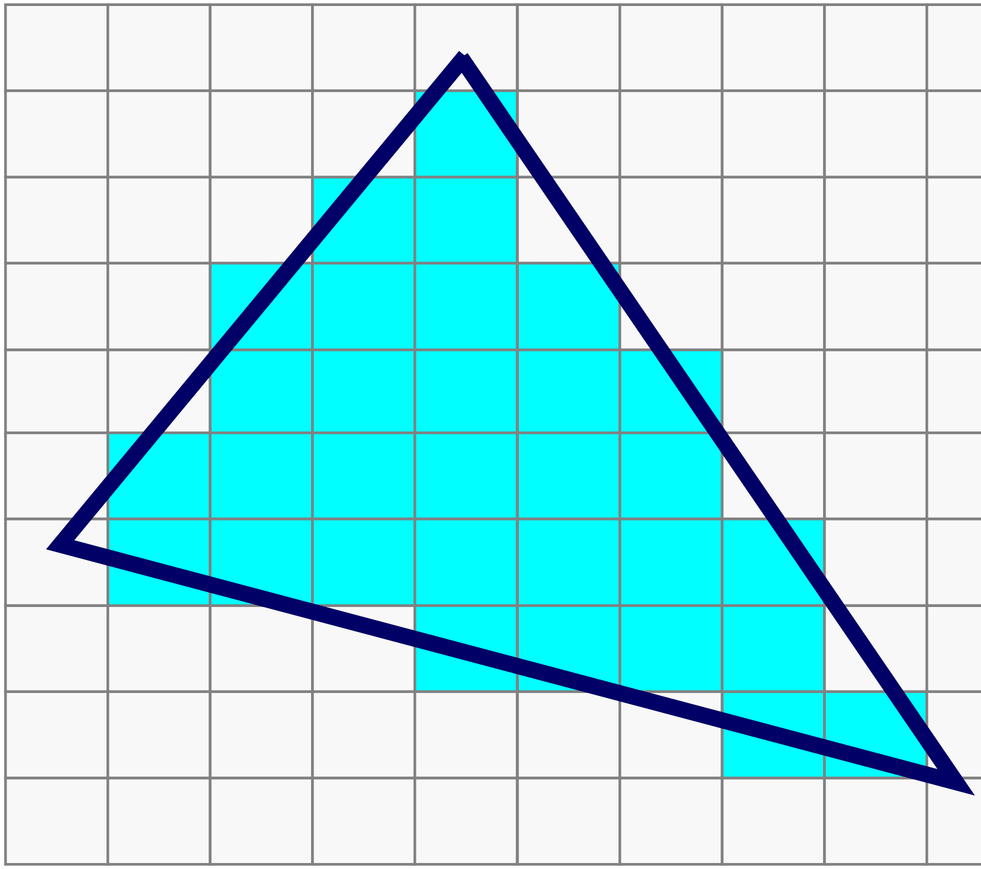
Spatial Aliasing

Artifacts due to limited spatial resolution



Spatial Aliasing

Artifacts due to limited spatial resolution

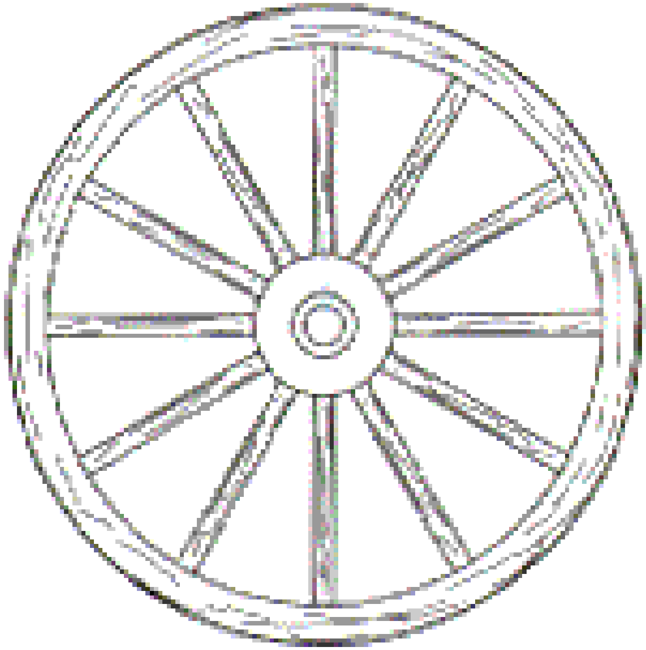


"Jaggies"

Temporal Aliasing

Artifacts due to limited temporal resolution

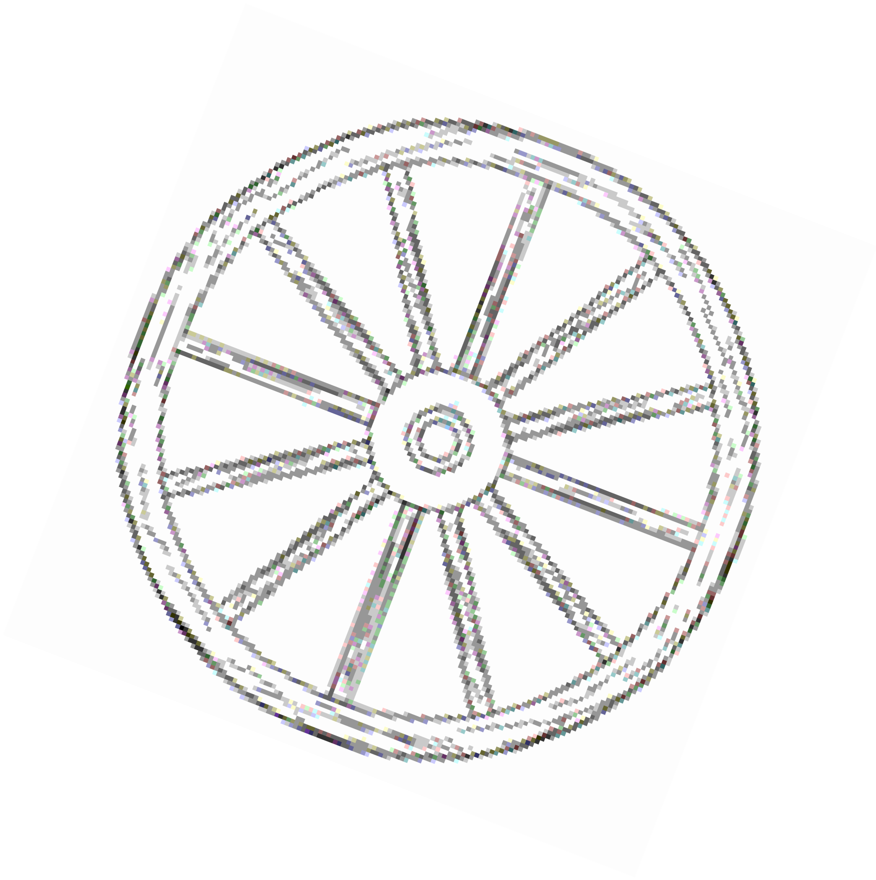
- Strobing
- Flickering



Temporal Aliasing

Artifacts due to limited temporal resolution

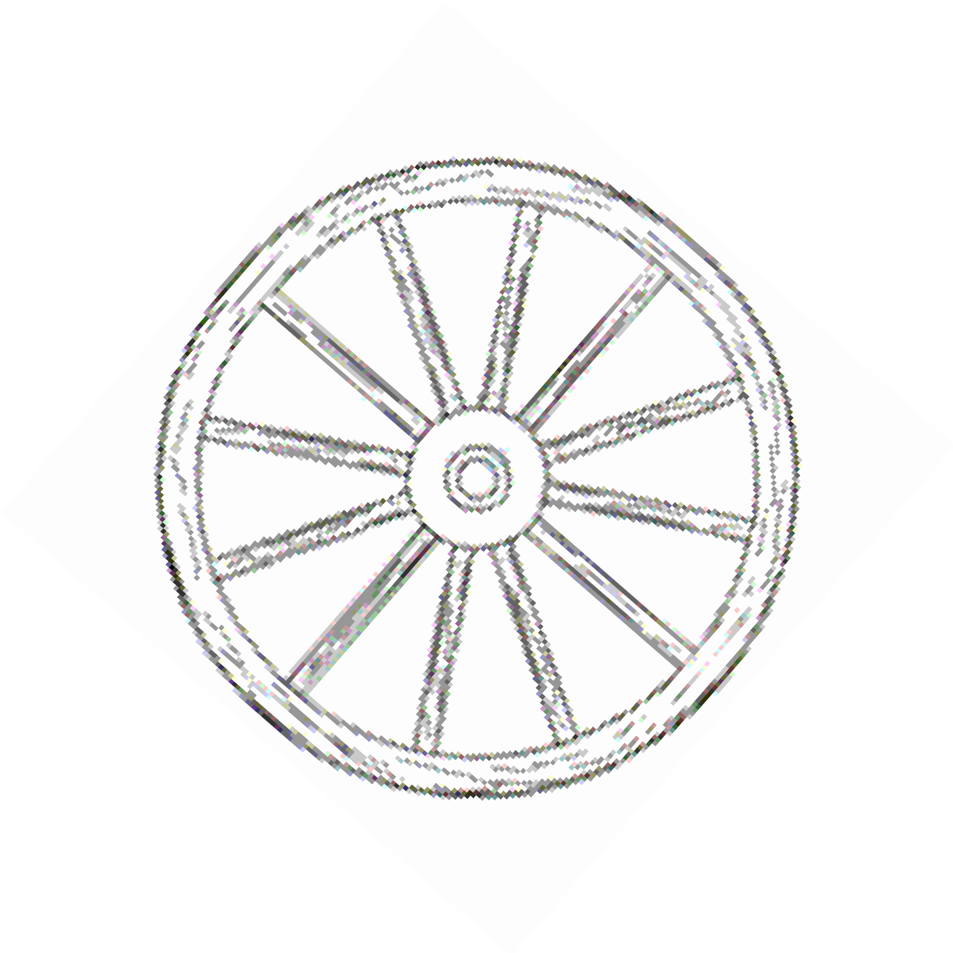
- Strobing
- Flickering



Temporal Aliasing

Artifacts due to limited temporal resolution

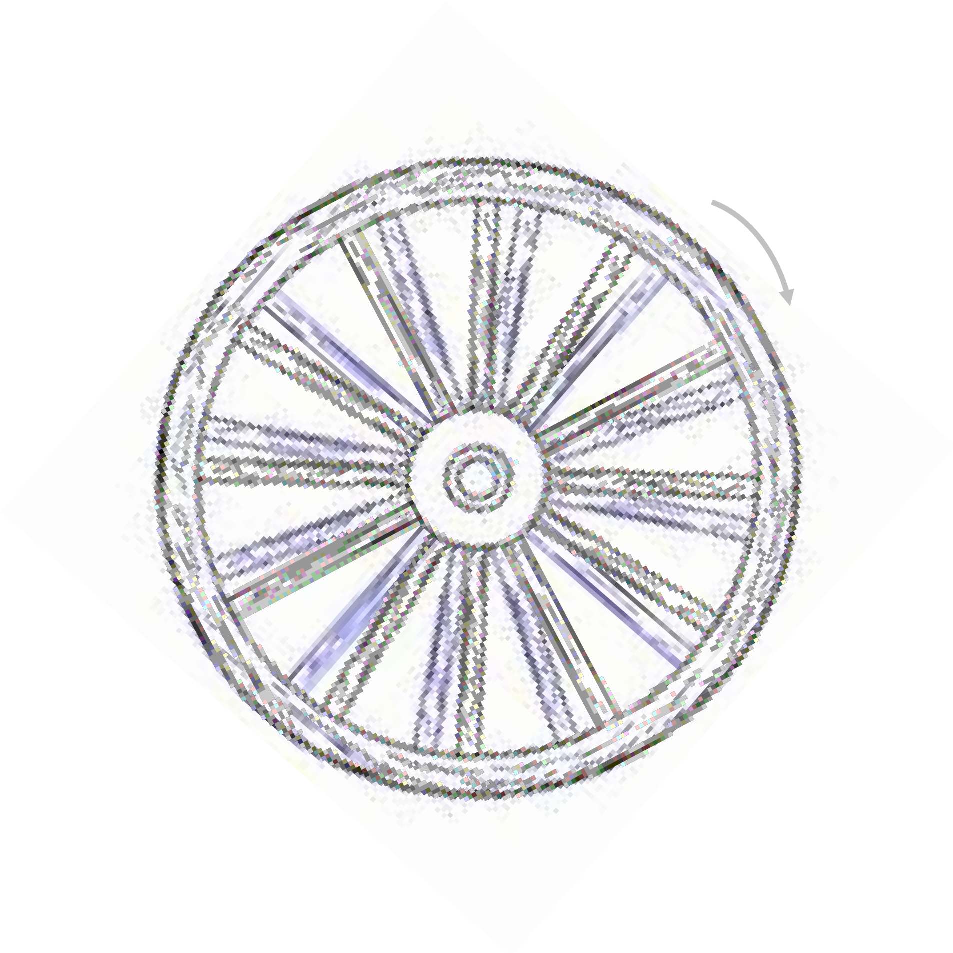
- Strobing
- Flickering



Temporal Aliasing

Artifacts due to limited temporal resolution

- Strobing
- Flickering



Antialiasing

- Sample at higher rate
  - Not always possible
  - Doesn't always solve problem
- Pre-filter to form bandlimited signal
  - Form bandlimited function using low-pass filter
  - Trades aliasing for blurring

Image Processing

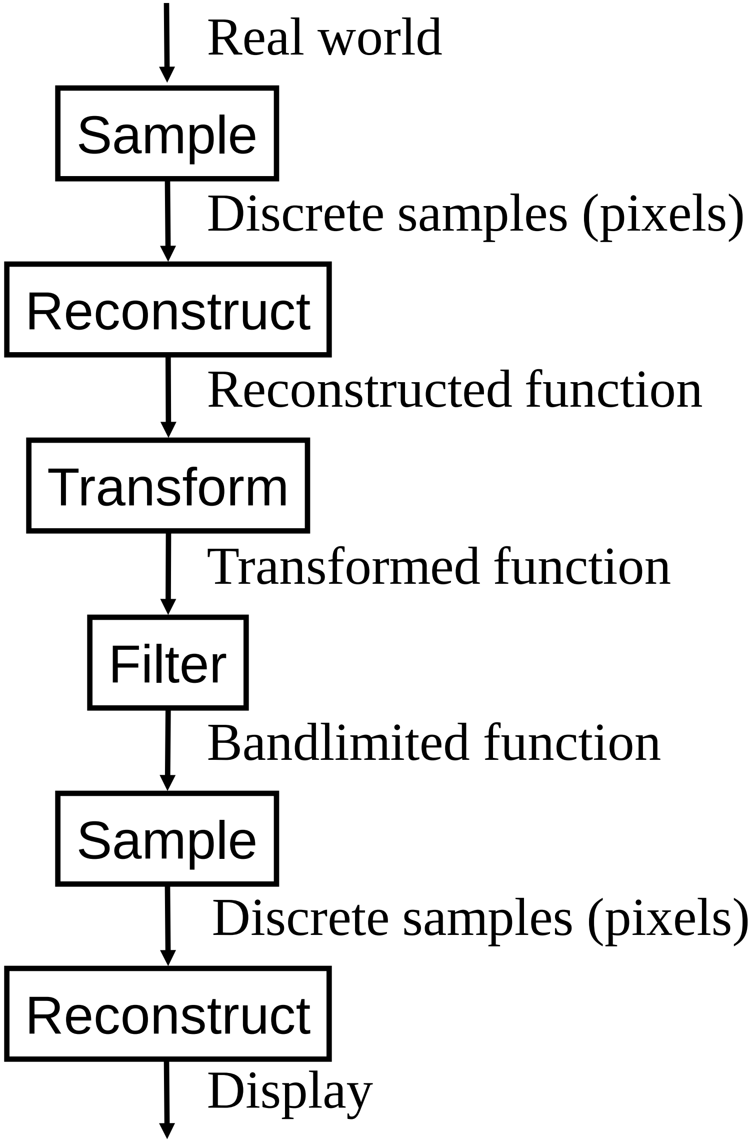


Image Processing

Real world

Sample

Discrete samples (pixels)

Reconstruct

Reconstructed function

Transform

Transformed function

Filter

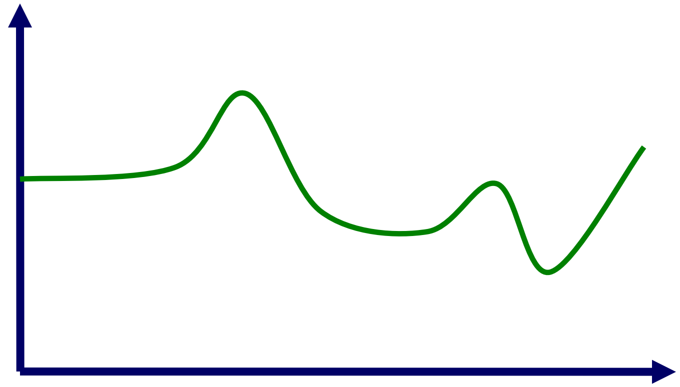
Bandlimited function

Sample

Discrete samples (pixels)

Reconstruct

Display



Continuous Function

Image Processing

Real world

Sample

Discrete samples (pixels)

Reconstruct

Reconstructed function

Transform

Transformed function

Filter

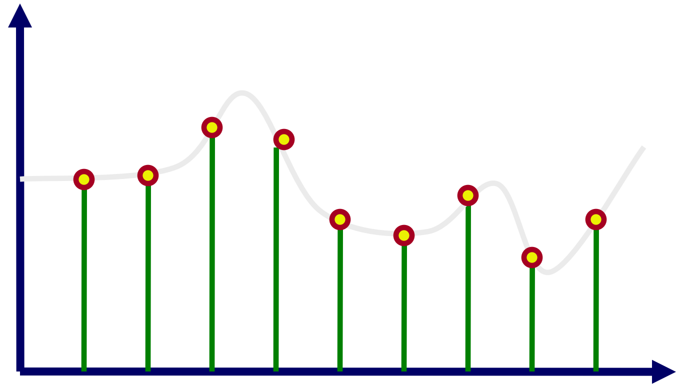
Bandlimited function

Sample

Discrete samples (pixels)

Reconstruct

Display



Discrete Samples

Image Processing

Real world

Sample

Discrete samples (pixels)

Reconstruct

Reconstructed function

Transform

Transformed function

Filter

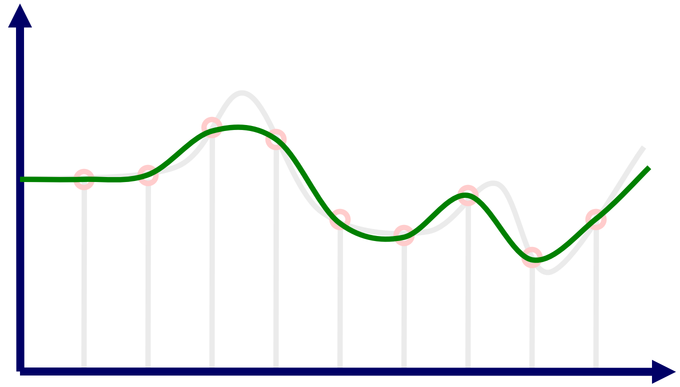
Bandlimited function

Sample

Discrete samples (pixels)

Reconstruct

Display



Reconstructed Function

Image Processing

Real world

Sample

Discrete samples (pixels)

Reconstruct

Reconstructed function

Transform

Transformed function

Filter

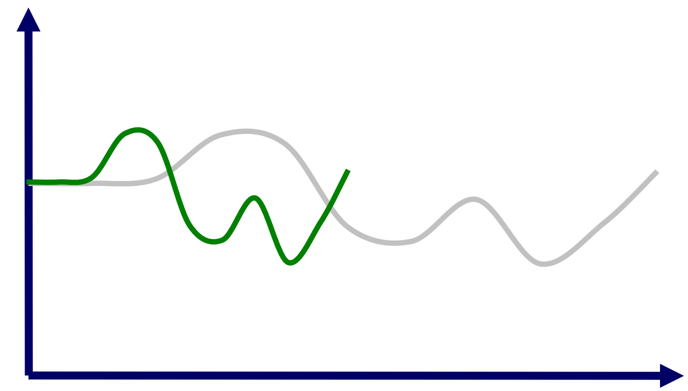
Bandlimited function

Sample

Discrete samples (pixels)

Reconstruct

Display



Transformed Function

Image Processing

Real world

Sample

Discrete samples (pixels)

Reconstruct

Reconstructed function

Transform

Transformed function

Filter

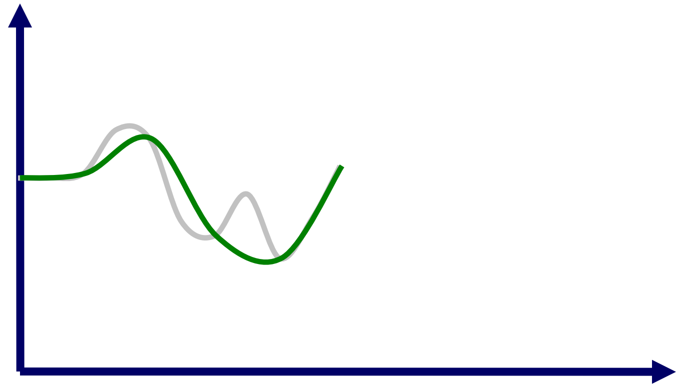
Bandlimited function

Sample

Discrete samples (pixels)

Reconstruct

Display



Bandlimited Function

Image Processing

Real world

Sample

Discrete samples (pixels)

Reconstruct

Reconstructed function

Transform

Transformed function

Filter

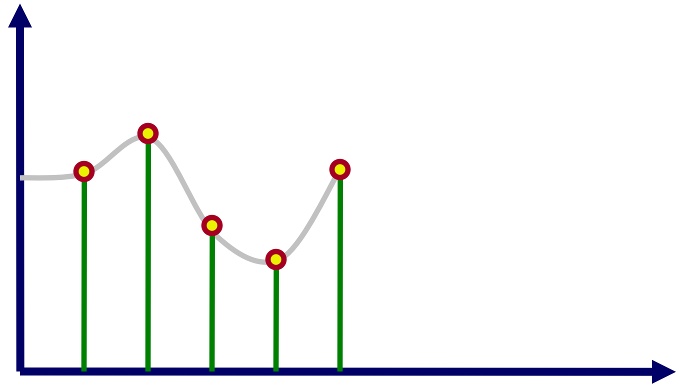
Bandlimited function

Sample

Discrete samples (pixels)

Reconstruct

Display



Discrete samples

Image Processing

Real world

Sample

Discrete samples (pixels)

Reconstruct

Reconstructed function

Transform

Transformed function

Filter

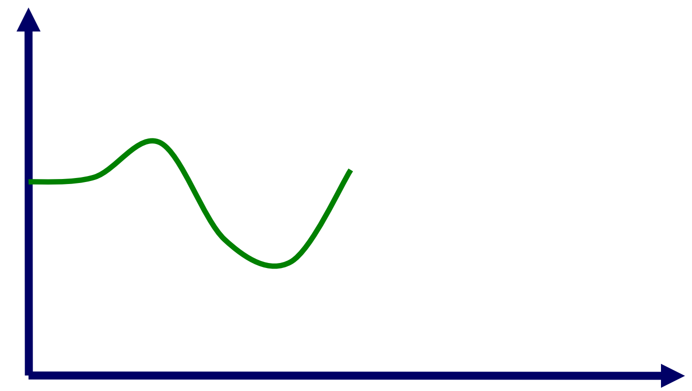
Bandlimited function

Sample

Discrete samples (pixels)

Reconstruct

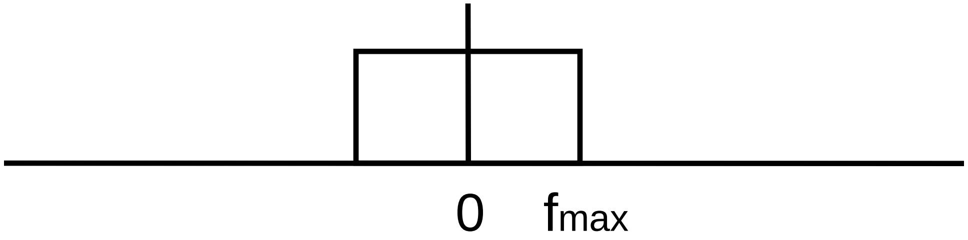
Display



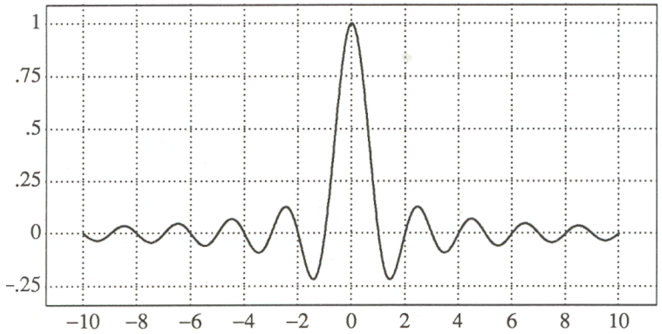
Display

Ideal Bandlimiting Filter

- Frequency domain



- Spatial domain

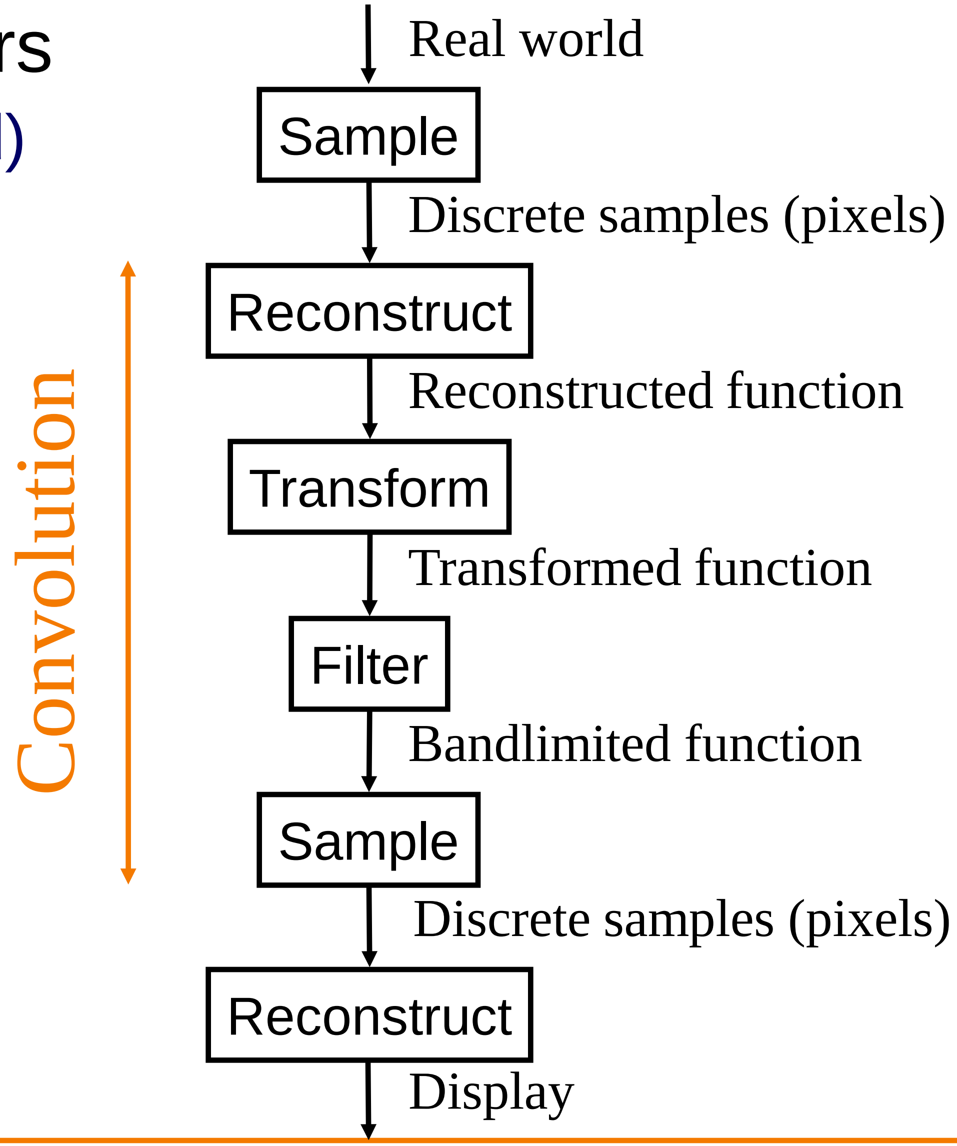


$$Sinc(x) = \frac{\sin \pi x}{\pi x}$$

Figure 4.5 Wolberg

Practical Image Processing

- Finite low-pass filters
  - o Point sampling (bad)
  - o Box filter
  - o Triangle filter
  - o Gaussian filter



Scaling

- Resample with triangle or Gaussian filter

Original

1/4X resolution

4X resolution

Summary

- Image filtering
  - o Compute new values for image pixels based on function of old values
- Halftoning and dithering
  - o Reduce visual artifacts due to quantization
  - o Distribute errors among pixels
    - » Exploit spatial integration in our eye
- Sampling and reconstruction
  - o Reduce visual artifacts due to aliasing
  - o Filter to avoid undersampling
    - » Blurring is better than aliasing