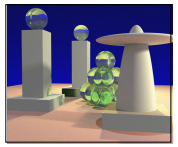
Computer Graphics (Fall 2011)

CS 184 Guest Lecture: Sampling and Reconstruction

Ravi Ramamoorthi


Some slides courtesy Thomas Funkhouser and Pat Hanrahan
Adapted version of CS 283 lecture http://inst.eecs.berkeley.edu/~cs283/fa10

Outline

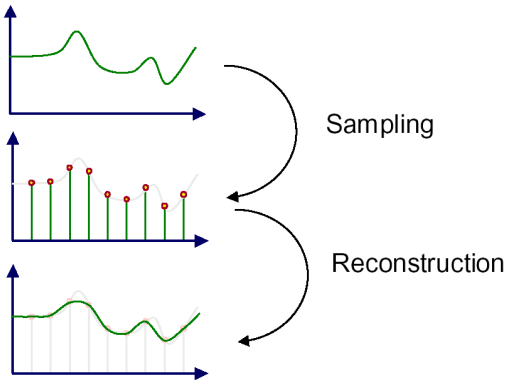
- *Basic ideas of sampling, reconstruction, aliasing*
- Signal processing and Fourier analysis
- Implementation of digital filters

- Section 14.10 of FvDFH (you really should read)

Some slides courtesy Tom Funkhouser

Sampling and Reconstruction

- An image is a 2D array of samples
- Discrete samples from real-world continuous signal



3

Sampling and Reconstruction

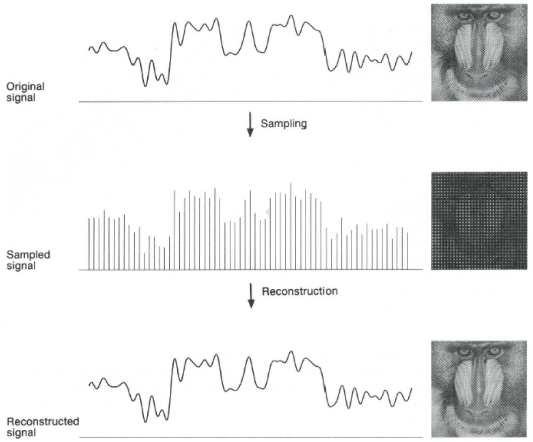
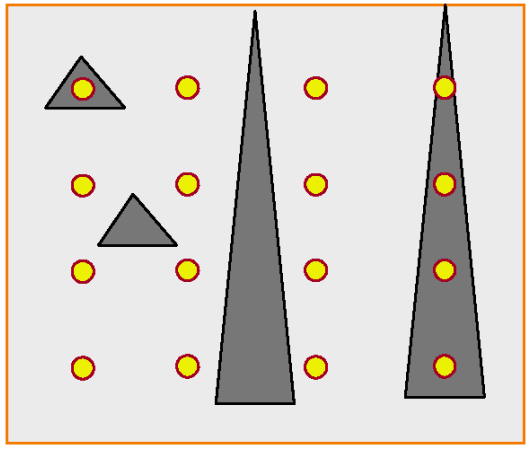


Figure 19.9 FvDFH

4

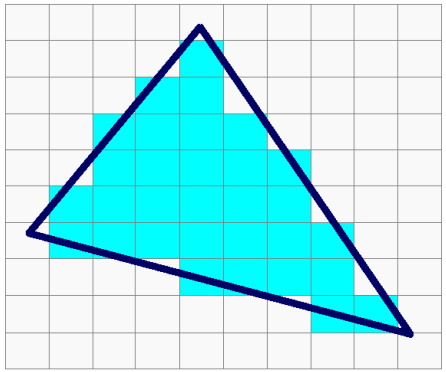
(Spatial) Aliasing



5

(Spatial) Aliasing

- Jaggies probably biggest aliasing problem

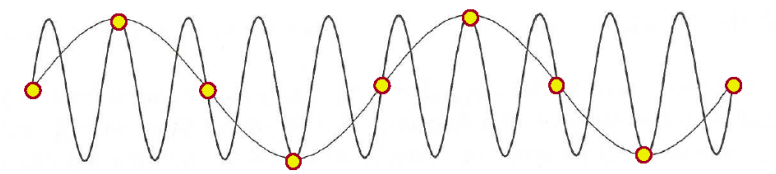


"Jaggies"

6

Sampling and Aliasing

- Artifacts due to undersampling or poor reconstruction
- Formally, high frequencies masquerading as low
- E.g. high frequency line as low freq jaggies

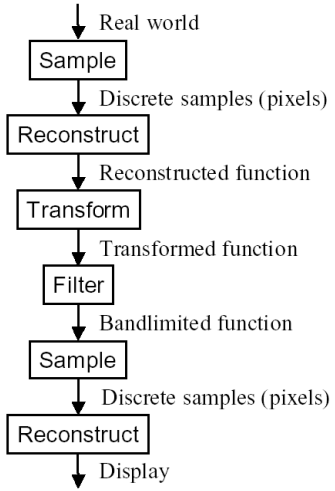


Under-sampling

Figure 14.17 FvDFH

7

Image Processing pipeline



8

Outline

- Basic ideas of sampling, reconstruction, aliasing
- *Signal processing and Fourier analysis*
- Implementation of digital filters

- Section 14.10 of textbook

Motivation

- Formal analysis of sampling and reconstruction
- Important theory (signal-processing) for graphics
- Also relevant in rendering, modeling, animation

Ideas

- Signal (function of time generally, here of space)
- Continuous: defined at all points; discrete: on a grid
- High frequency: rapid variation; Low Freq: slow variation
- Images are converting continuous to discrete. Do this sampling as best as possible.
- Signal processing theory tells us how best to do this
- Based on concept of frequency domain Fourier analysis

Sampling Theory

Analysis in the frequency (not spatial) domain

- Sum of sine waves, with possibly different offsets (phase)
- Each wave different frequency, amplitude

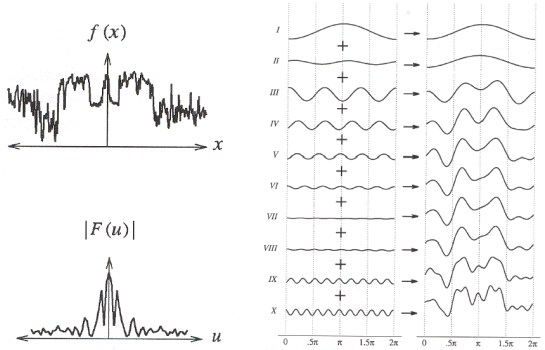


Figure 2.6 Wolberg

Fourier Transform

- Tool for converting from spatial to frequency domain
- Or vice versa
- One of most important mathematical ideas
- Computational algorithm: Fast Fourier Transform
  - One of 10 great algorithms scientific computing
  - Makes Fourier processing possible (images etc.)
  - Not discussed here, but look up if interested

Fourier Transform

- Simple case, function sum of sines, cosines

- Continuous infinite case

Fourier Transform

- Simple case, function sum of sines, cosines
- Continuous infinite case

14

Fourier Transform

- Simple case, function sum of sines, cosines
- Discrete case

15

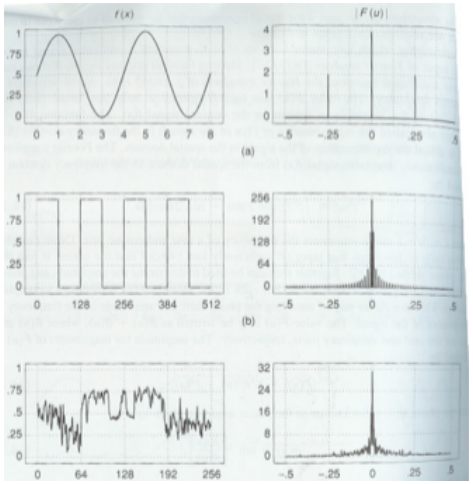
Fourier Transform

- Simple case, function sum of sines, cosines

- Discrete case

Fourier Transform: Examples 1

Single sine curve
(+constant DC term)



Fourier Transform Examples 2

- Common examples

17

Fourier Transform Properties

- Common properties
  - Linearity:
  - Derivatives: [integrate by parts]
  - 2D Fourier Transform
- Convolution (next)

18

Fourier Transform Properties

- Common properties
  - Linearity:
  - Derivatives: [integrate by parts]
  - 2D Fourier Transform
- Convolution (next)

Fourier Transform Properties

- Common properties
  - Linearity:
  - Derivatives: [integrate by parts]
  - 2D Fourier Transform
- Convolution (next)

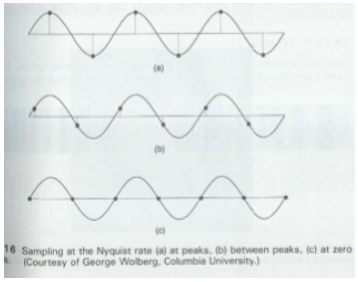
18

Fourier Transform Properties

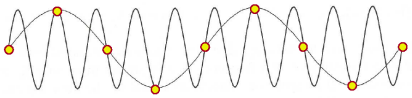
- Common properties
  - Linearity:
  - Derivatives: [integrate by parts]
  - 2D Fourier Transform
- Convolution (next)

Sampling Theorem, Bandlimiting

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



16 Sampling at the Nyquist rate (a) at peaks, (b) between peaks, (c) at zero (Courtesy of George Wolberg, Columbia University.)



Under-sampling

Figure 14.17 FvDFH

Sampling Theorem, Bandlimiting

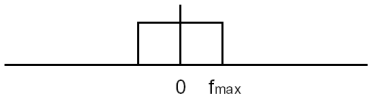
- A signal can be reconstructed from its samples, if the original signal has no frequencies above half the sampling frequency – Shannon
- The minimum sampling rate for a bandlimited function is called the Nyquist rate
- A signal is bandlimited if the highest frequency is bounded. This frequency is called the bandwidth
- In general, when we transform, we want to filter to bandlimit before sampling, to avoid aliasing

Antialiasing

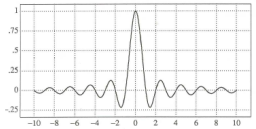
- Sample at higher rate
  - Not always possible
  - Real world: lines have infinitely high frequencies, can't sample at high enough resolution
- Prefilter to bandlimit signal
  - Low-pass filtering (blurring)
  - Trade blurriness for aliasing

Ideal bandlimiting filter

- Formal derivation is homework exercise
  - Frequency domain



  - Spatial domain



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

Figure 4.5 Wolberg

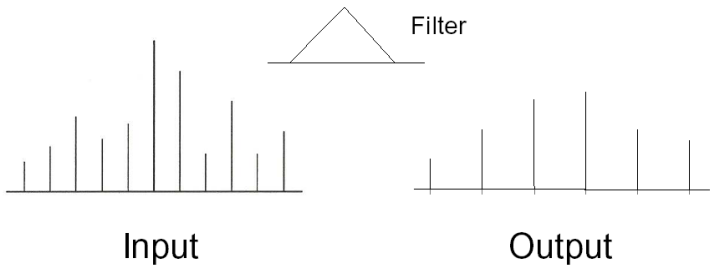
Outline

- Basic ideas of sampling, reconstruction, aliasing
- *Signal processing and Fourier analysis*
  - *Convolution*
- Implementation of digital filters

- Section 14.10 of FvDFH

Convolution 1

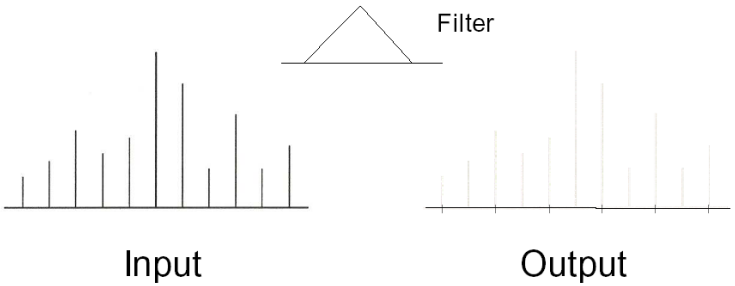
- Spatial domain: output pixel is weighted sum of pixels in neighborhood of input image
  - Pattern of weights is the “filter”



24

Convolution 2

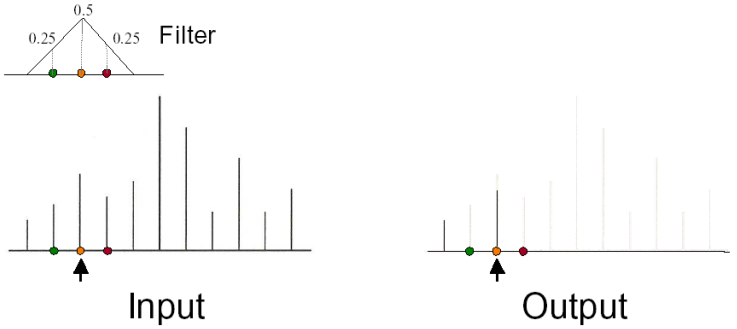
- Example 1:



25

Convolution 3

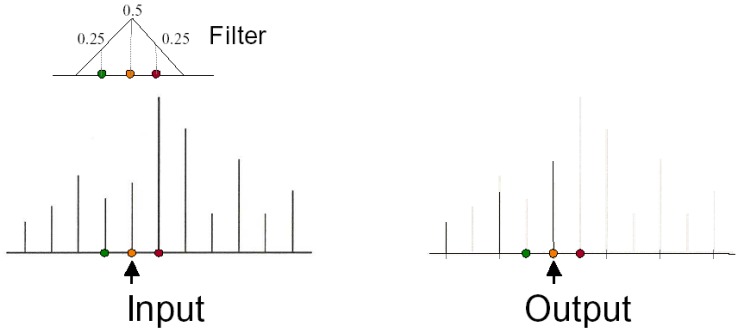
- Example 1:



26

Convolution 4

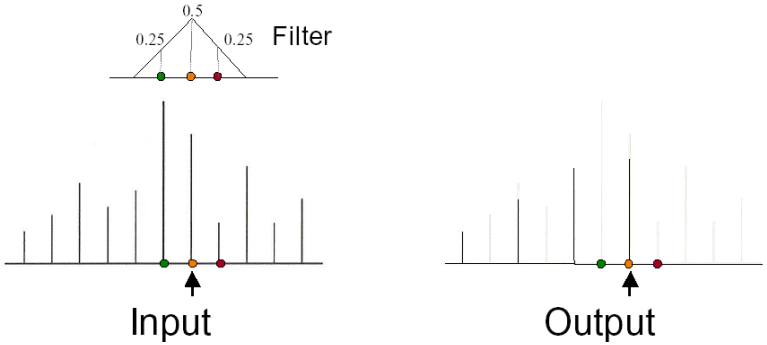
- Example 1:



27

Convolution 5

- Example 1:



28

Convolution in Frequency Domain

- Convolution (f is signal ; g is filter [or vice versa])
- Fourier analysis (frequency domain multiplication)

29

Convolution in Frequency Domain

- Convolution (f is signal ; g is filter [or vice versa])
- Fourier analysis (frequency domain multiplication)

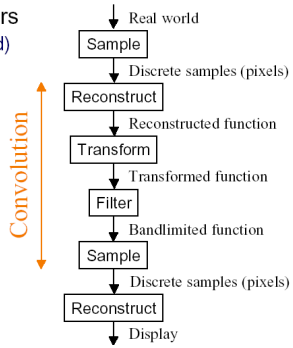
Convolution in Frequency Domain

- Convolution (f is signal ; g is filter [or vice versa])
- Fourier analysis (frequency domain multiplication)

29

Practical Image Processing

- Discrete convolution (in spatial domain) with filters for various digital signal processing operations
- Easy to analyze, understand effects in frequency domain
  - E.g. blurring or bandlimiting by convolving with low pass filter
    - Finite low-pass filters
      - Point sampling (bad)
      - Triangle filter
      - Gaussian filter



Outline

- Basic ideas of sampling, reconstruction, aliasing
- Signal processing and Fourier analysis
- *Implementation of digital filters*

- Section 14.10 of FvDFH

Discrete Convolution

- Previously: Convolution as mult in freq domain
  - But need to convert digital image to and from to use that
  - Useful in some cases, but not for small filters
- Previously seen: Sinc as ideal low-pass filter
  - But has infinite spatial extent, exhibits spatial ringing
  - In general, use frequency ideas, but consider implementation issues as well
- Instead, use simple discrete convolution filters e.g.
  - Pixel gets sum of nearby pixels weighted by filter/mask

2	0	-7
5	4	9
1	-6	-2

Implementing Discrete Convolution

- Fill in each pixel new image convolving with old
  - Not really possible to implement it in place
  - More efficient for smaller kernels/filters f
- Normalization
  - If you don't want overall brightness change, entries of filter must sum to 1. You may need to normalize by dividing
- Integer arithmetic
  - Simpler and more efficient
  - In general, normalization outside, round to nearest int

Outline

- Implementation of digital filters
  - Discrete convolution in spatial domain
  - *Basic image-processing operations*
  - Antialiased shift and resize

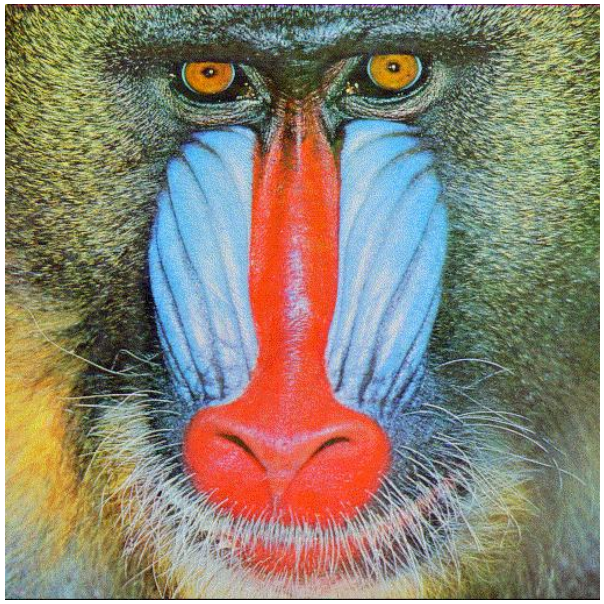
Basic Image Processing

- *Blur*
- Sharpen
- Edge Detection

All implemented using convolution with different filters

Blurring

- Used for softening appearance
- Convolve with gaussian filter
  - Same as mult. by gaussian in freq. domain, so reduces high-frequency content
  - Greater the spatial width, smaller the Fourier width, more blurring occurs and vice versa
- How to find blurring filter?



38

39

40

41

Blurring Filter

- In general, for symmetry f(u,v) = f(u) f(v)
  - You might want to have some fun with asymmetric filters
- We will use a Gaussian blur
  - Blur width sigma depends on kernel size n (3,5,7,11,13,19)

42

Discrete Filtering, Normalization

- Gaussian is infinite
  - In practice, finite filter of size n (much less energy beyond 2 sigma or 3 sigma).
  - Must renormalize so entries add up to 1
- Simple practical approach
  - Take smallest values as 1 to scale others, round to integers
  - Normalize. E.g. for n = 3, sigma = ½

43

Discrete Filtering, Normalization

- Gaussian is infinite
  - In practice, finite filter of size n (much less energy beyond 2 sigma or 3 sigma).
  - Must renormalize so entries add up to 1
- Simple practical approach
  - Take smallest values as 1 to scale others, round to integers
  - Normalize. E.g. for n = 3, sigma = ½

Discrete Filtering, Normalization

- Gaussian is infinite
  - In practice, finite filter of size n (much less energy beyond 2 sigma or 3 sigma).
  - Must renormalize so entries add up to 1
- Simple practical approach
  - Take smallest values as 1 to scale others, round to integers
  - Normalize. E.g. for n = 3, sigma = ½

43

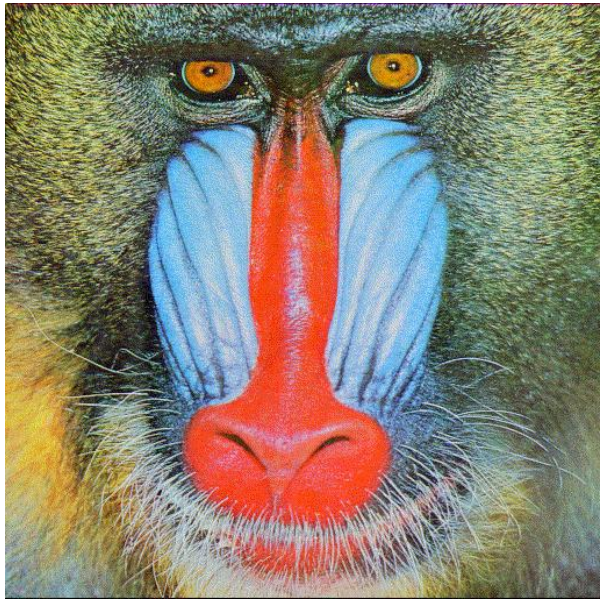
Basic Image Processing

- Blur
- *Sharpen*
- Edge Detection

All implemented using convolution with different filters

Sharpening Filter

- Unlike blur, want to accentuate high frequencies
- Take differences with nearby pixels (rather than avg)



46

47

48

Basic Image Processing

- Blur
- Sharpen
- *Edge Detection*

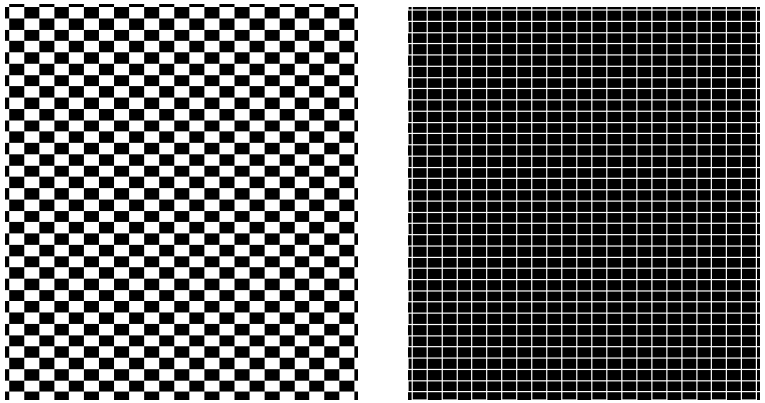
All implemented using convolution with different filters

49

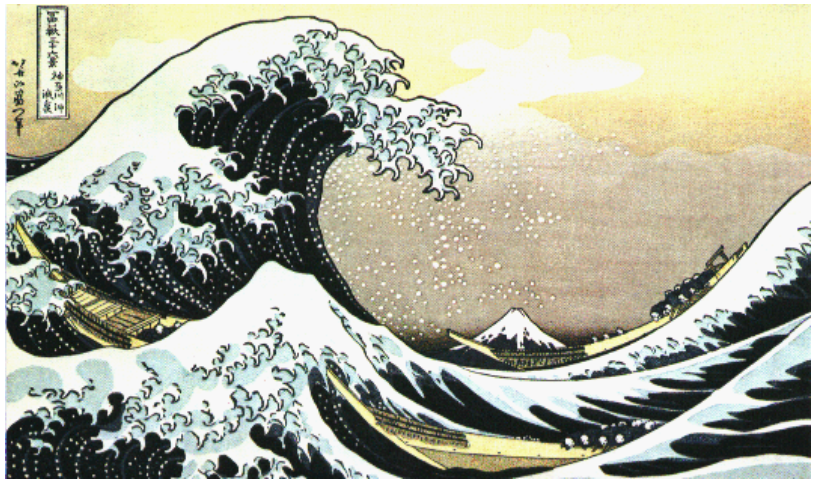
Edge Detection

- Complicated topic: subject of many PhD theses
- Here, we present one approach (Sobel edge detector)
- Step 1: Convolution with gradient (Sobel) filter
  - Edges occur where image gradients are large
  - Separately for horizontal and vertical directions
- Step 2: Magnitude of gradient
  - Norm of horizontal and vertical gradients
- Step 3: Thresholding
  - Threshold to detect edges

Edge Detection

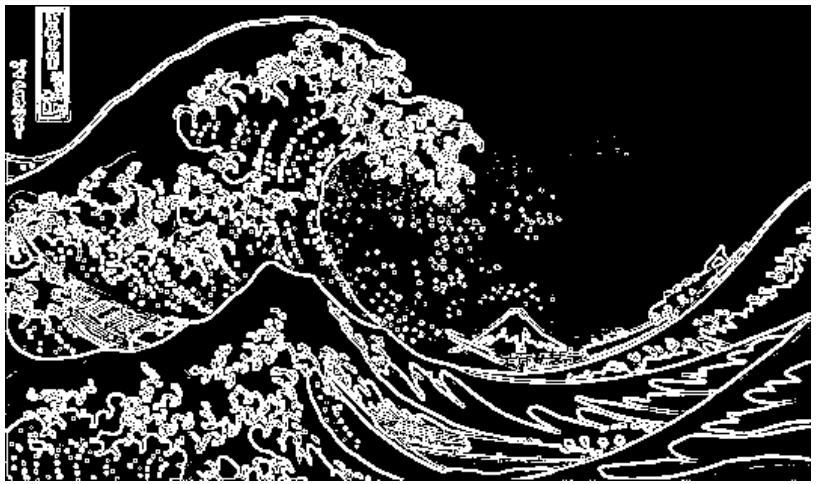


Edge Detection



52

Edge Detection



53

Details

- Step 1: Convolution with gradient (Sobel) filter
  - Edges occur where image gradients are large
  - Separately for horizontal and vertical directions
- Step 2: Magnitude of gradient
  - Norm of horizontal and vertical gradients
- Step 3: Thresholding

Outline

- Implementation of digital filters
  - Discrete convolution in spatial domain
  - Basic image-processing operations
  - *Antialiased shift and resize*

Antialiased Shift

Shift image based on (fractional) s_x and s_y

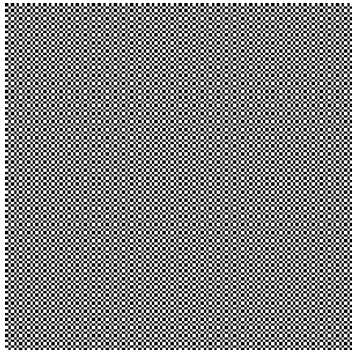
- Check for integers, treat separately
- Otherwise convolve/resample with kernel/filter h:

Antialiased Scale Magnification

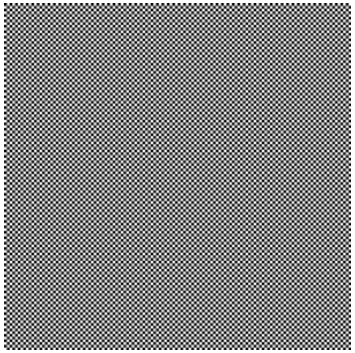
Magnify image (scale s or $\gamma > 1$)

- Interpolate between orig. samples to evaluate frac vals
- Do so by convolving/resampling with kernel/filter:
- *Treat the two image dimensions independently (diff scales)*

Antialiased Scale Minification



checkerboard.bmp 300x300: point sample



checkerboard.bmp 300x300: Mitchell

Antialiased Scale Minification

Minify (reduce size of) image

- Similar in some ways to mipmapping for texture maps
- We use *fat* pixels of size 1/γ, with new size γ*orig size (γ is scale factor < 1).
- Each fat pixel must integrate over corresponding region in original image using the filter kernel.

γ