Thinking in Frequency

Computer Vision

A U G Sankararao, IIIT Sri City

Previous class: Image Filtering

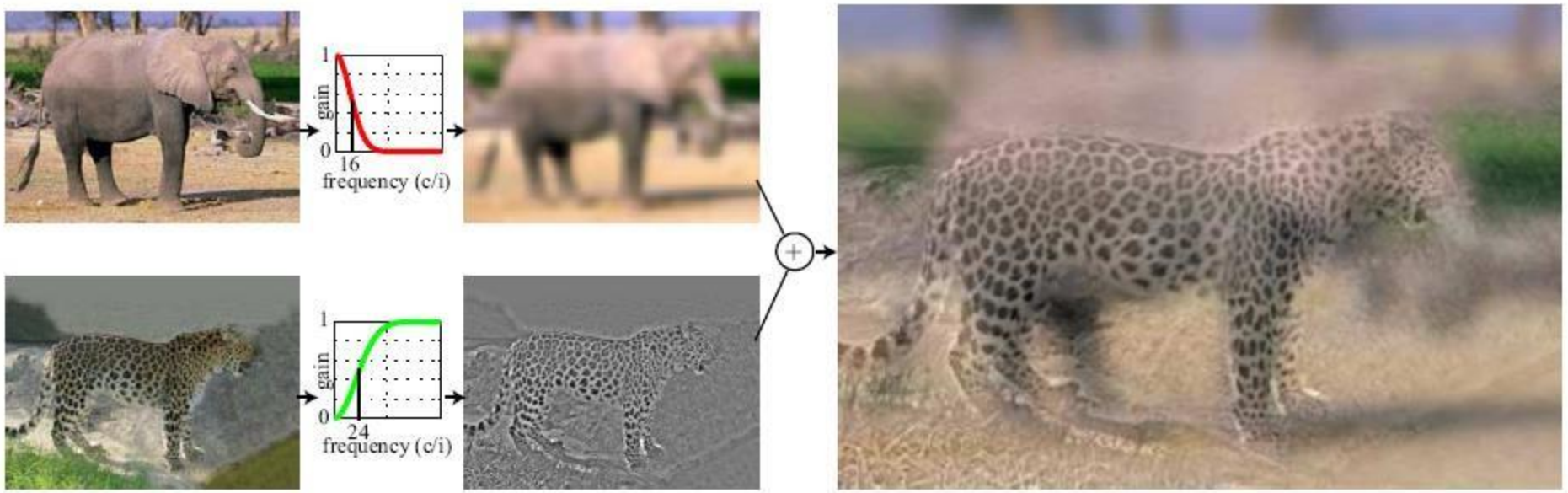
- Linear filtering is sum of dot product at each position
  - Can smooth, sharpen, translate (among many other uses)
- Gaussian filters
  - Low pass filters, separability, variance
- Attend to details:
  - filter size
- Noise models and nonlinear image filters

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Today's class

- Fourier transform and frequency domain
- Frequency view of filtering
- Image downsizing and interpolation

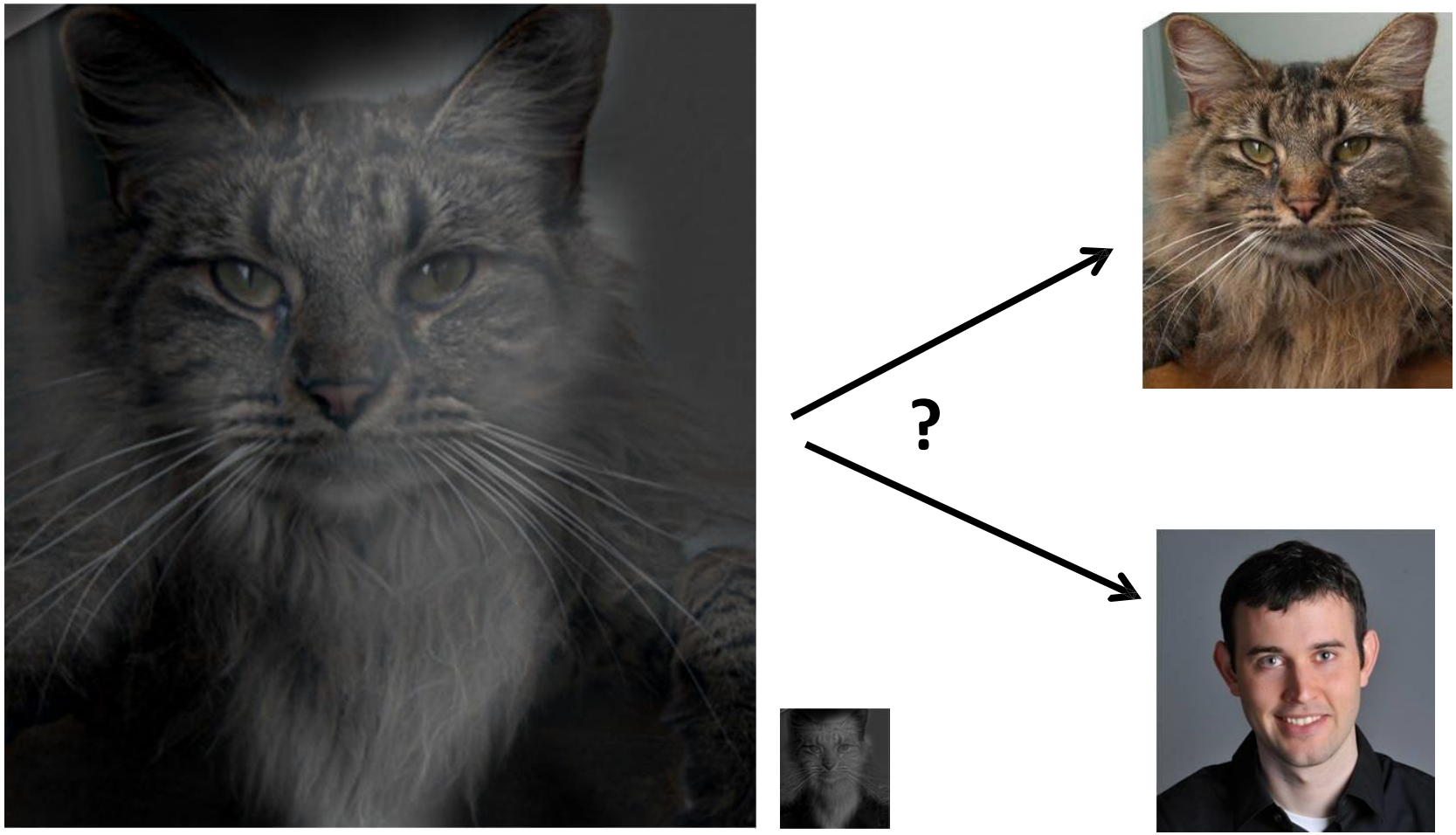
Hybrid Images



- A. Oliva, A. Torralba, P.G. Schyns, "Hybrid Images," SIGGRAPH 2006

Slide credit: Derek Hoiem

Why do we get different, distance-dependent interpretations of hybrid images?



Slide credit: Derek Hoiem

Why does a lower resolution image still make sense to us?

What do we lose?

Slide credit: Derek Hoiem

Image: http://www.flickr.com/photos/igorms/136916757/

Thinking in terms of frequency

Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

Any *periodic function can be rewritten as a weighted sum of sines and cosines of different frequencies.*

Slides: Efros

Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

Any *periodic function can be rewritten as a weighted sum of sines and cosines of different frequencies.*

- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!

Slides: Efros

Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

Any *periodic function can be rewritten as a weighted sum of sines and cosines of different frequencies.*

But it's (mostly) true!

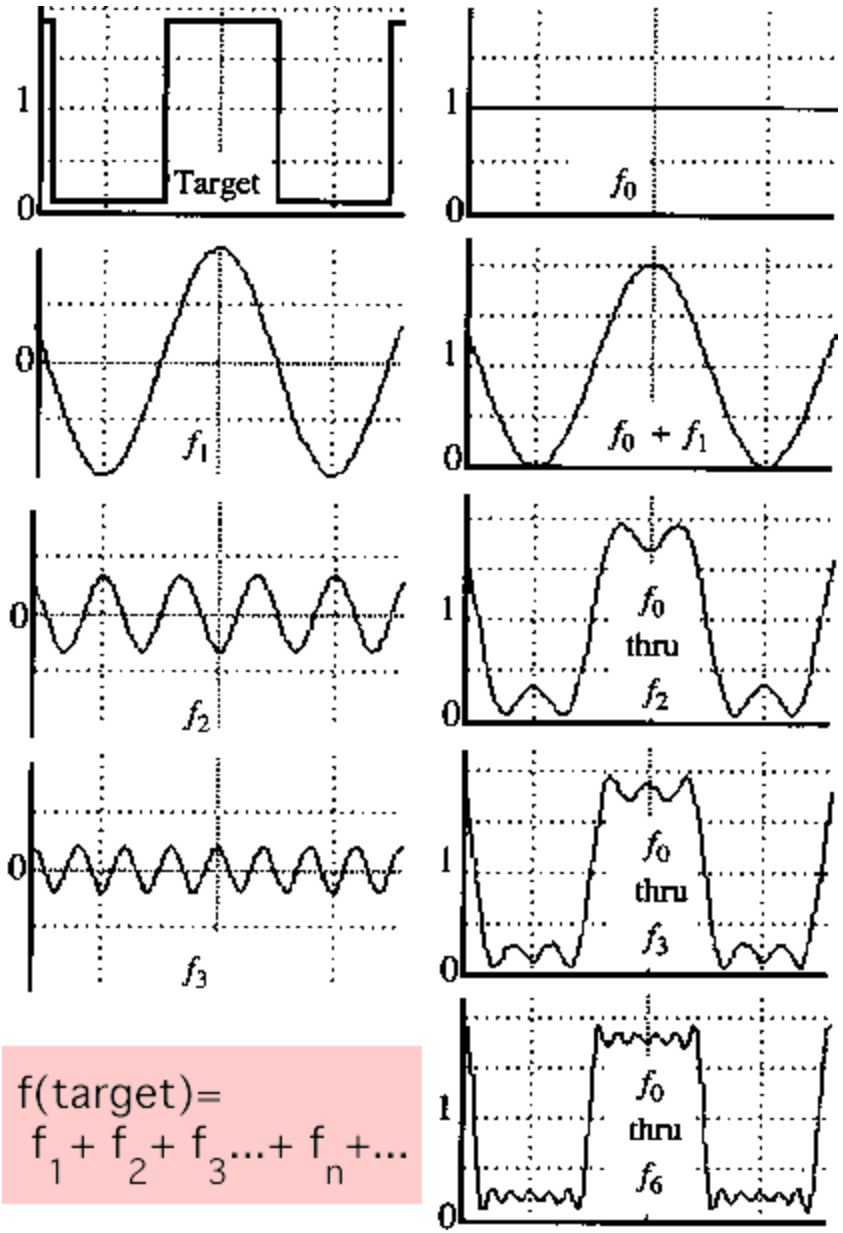
- called Fourier Series

A sum of sines

Our building block:

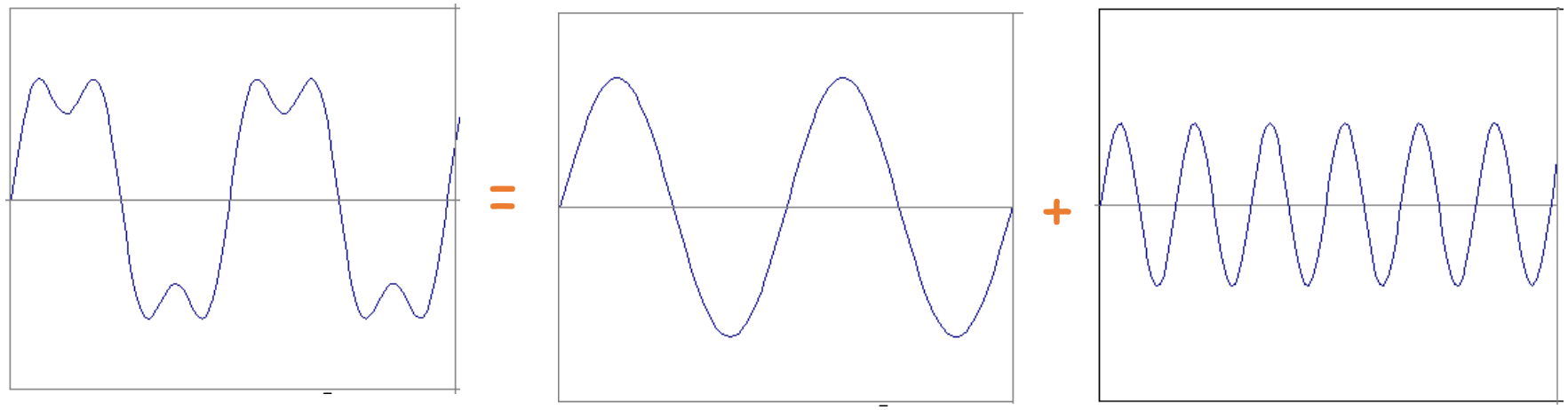
$$A\sin(\omega x + \phi)$$

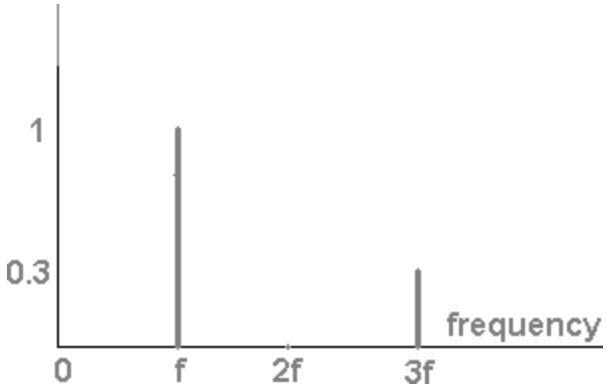
Add enough of them to get any signal *f(x)* you want!



Frequency Spectra

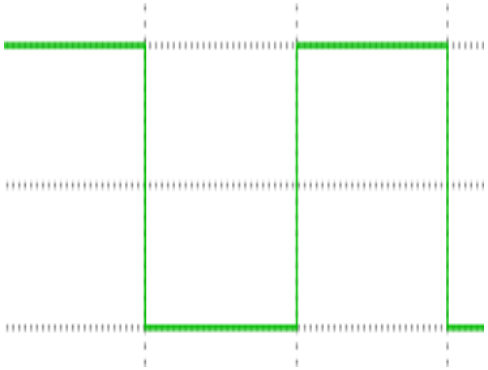
- example : $g(t) = \sin(2\pi f\, t) + (1/3)\sin(2\pi(3f)\, t)$



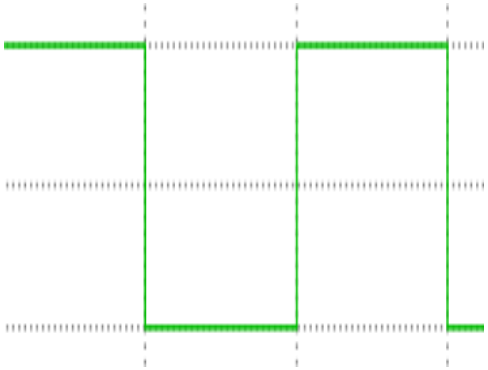


Slides: Efros

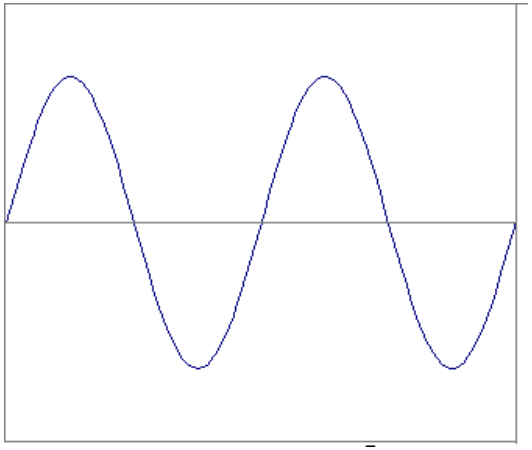
Frequency Spectra



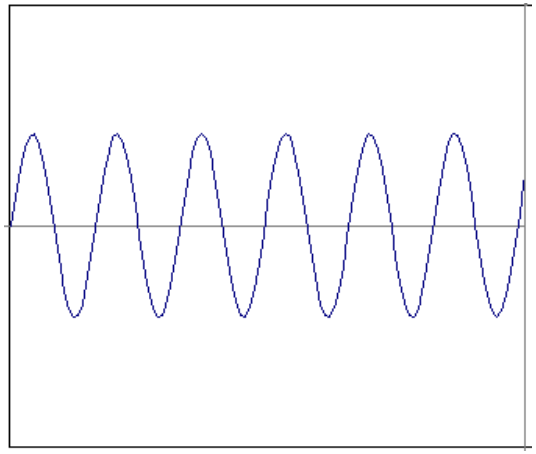
Frequency Spectra



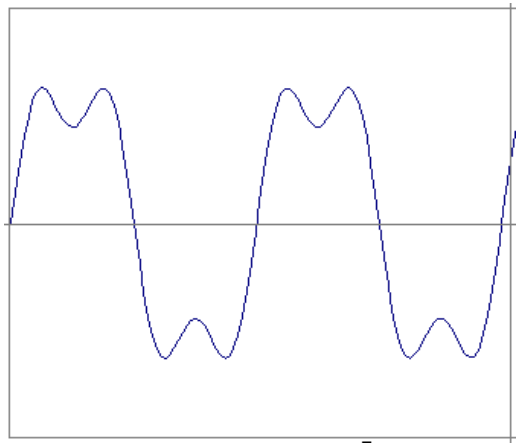
=



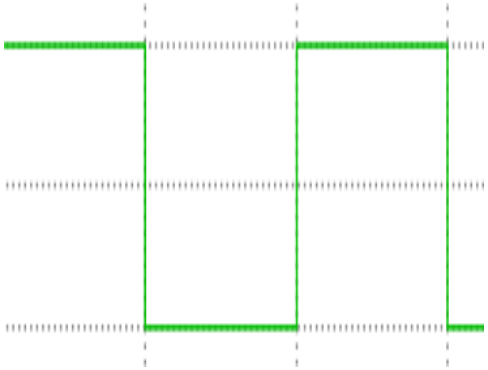
+



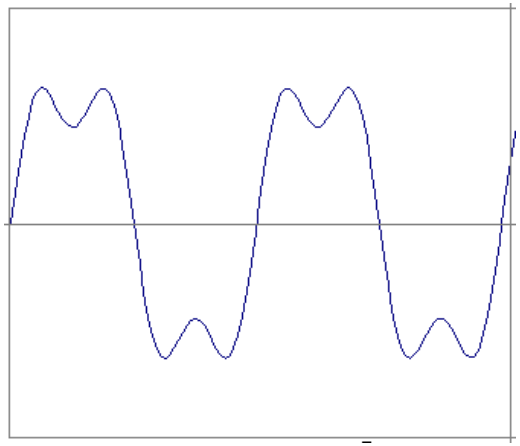
=



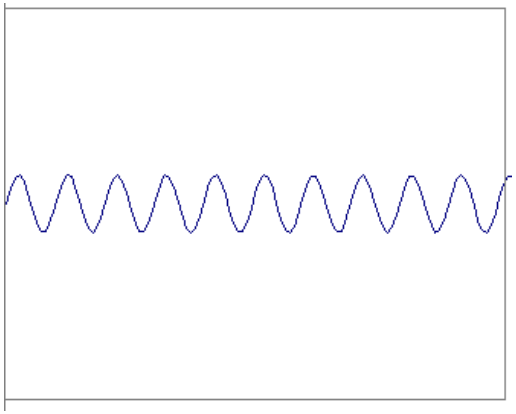
Frequency Spectra



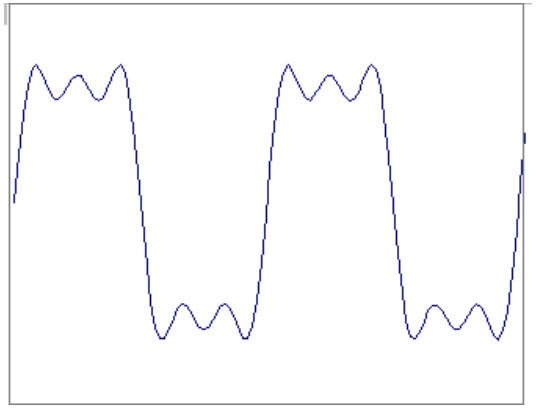
=



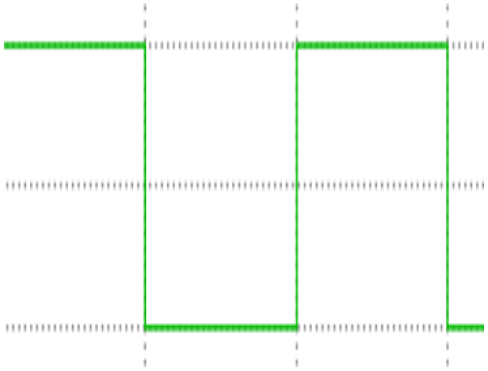
+



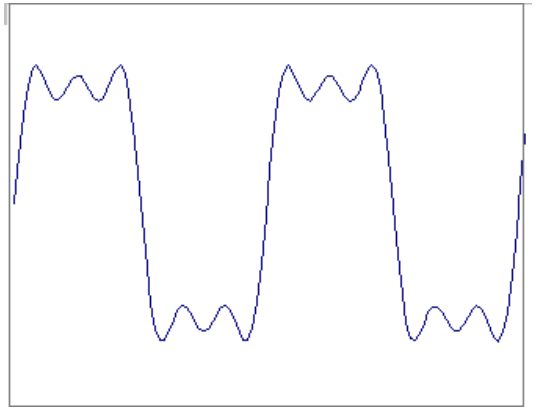
=



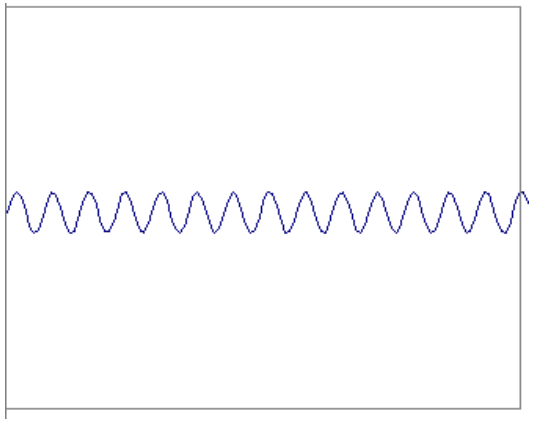
Frequency Spectra



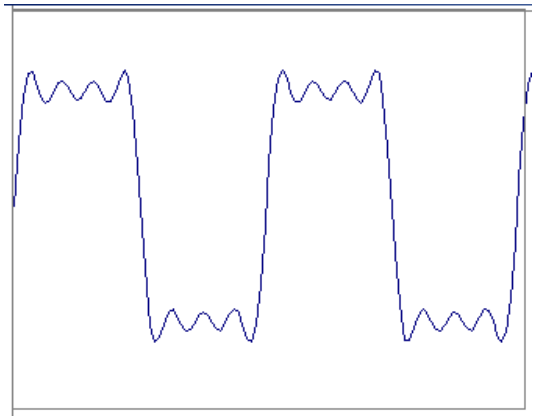
=



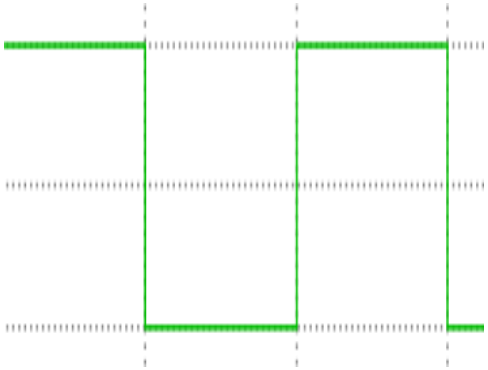
+



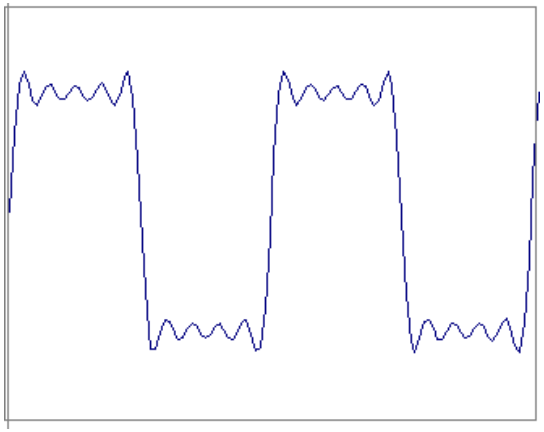
=



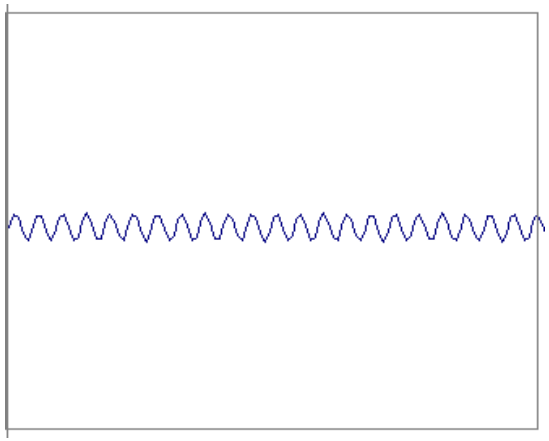
Frequency Spectra



=

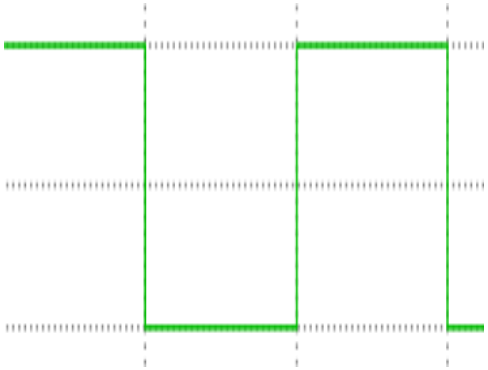


+



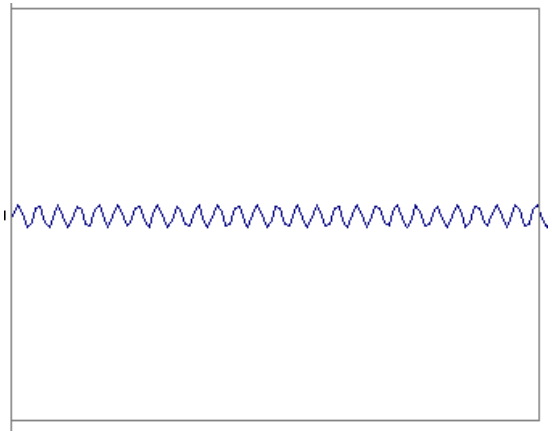
=

Frequency Spectra

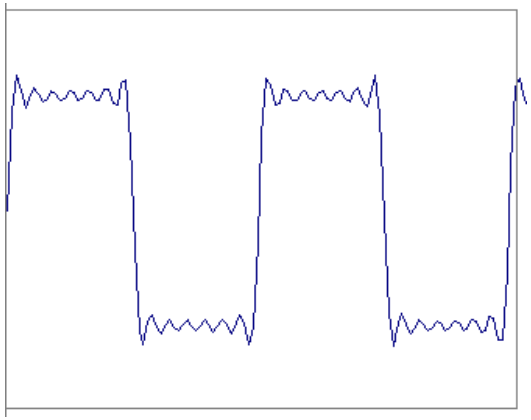


=

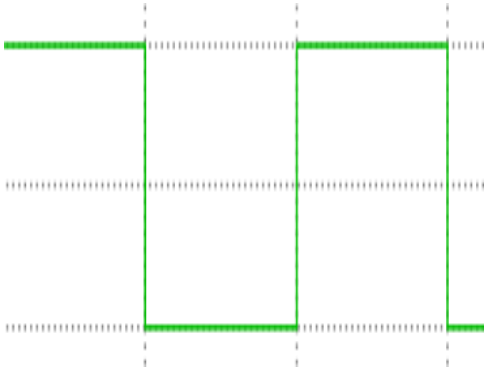
+



=

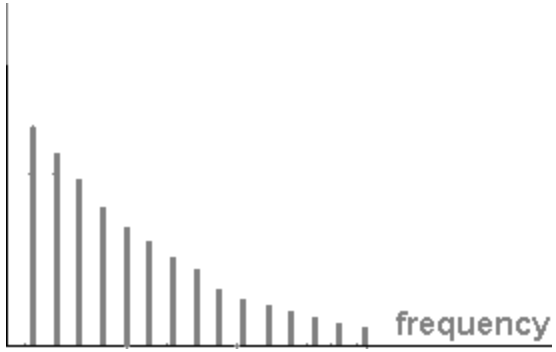


Frequency Spectra



=

$$A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi\, kt)$$



Fourier Transform

We want to understand the frequency ω of our signal. So, let's reparametrize the signal by ω instead of x:

For every ω from 0 to inf, (actually –inf to inf), $F(\omega)$ holds the amplitude A and phase ϕ of the corresponding sine

- How can F hold both? Complex number trick!

Slide: A. Bobick

Fourier Transform

We want to understand the frequency ω of our signal. So, let's reparametrize the signal by ω instead of x:

For every ω from 0 to inf, (actually –inf to inf), ***F(ω)*** holds the amplitude A and phase ϕ of the corresponding sine

- How can F hold both? Complex number trick!

$$A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$$

$$F(\omega) = R(\omega) + iI(\omega)$$

Even *Odd*

$$\phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$

Slide: A. Bobick

Fourier Transform

We want to understand the frequency ω of our signal. So, let's reparametrize the signal by ω instead of x:

For every ω from 0 to inf, (actually –inf to inf), $F(\omega)$ holds the amplitude A and phase ϕ of the corresponding sine

- How can F hold both? Complex number trick!

$$A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$$

$$F(\omega) = R(\omega) + iI(\omega)$$

Even *Odd*

$$\phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$

And we can go back:

Slide: A. Bobick

Fourier Transform

- FT stores the magnitude and phase at each frequency
  - Magnitude encodes how much signal is there at a particular frequency
  - Phase encodes spatial information (indirectly)

Amplitude: $A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$

Phase: $\phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$

Slide: Huang

Discrete Fourier Transform

$$F(k) = \sum_{x=0}^{x=N-1} f(x) e^{-i\frac{2\pi kx}{N}}$$

k = -N/2..N/2

$$e^{ik} = \cos k + i \sin k \qquad i = \sqrt{-1}$$

Credit: A. Bobick

Continuous vs Discrete FT

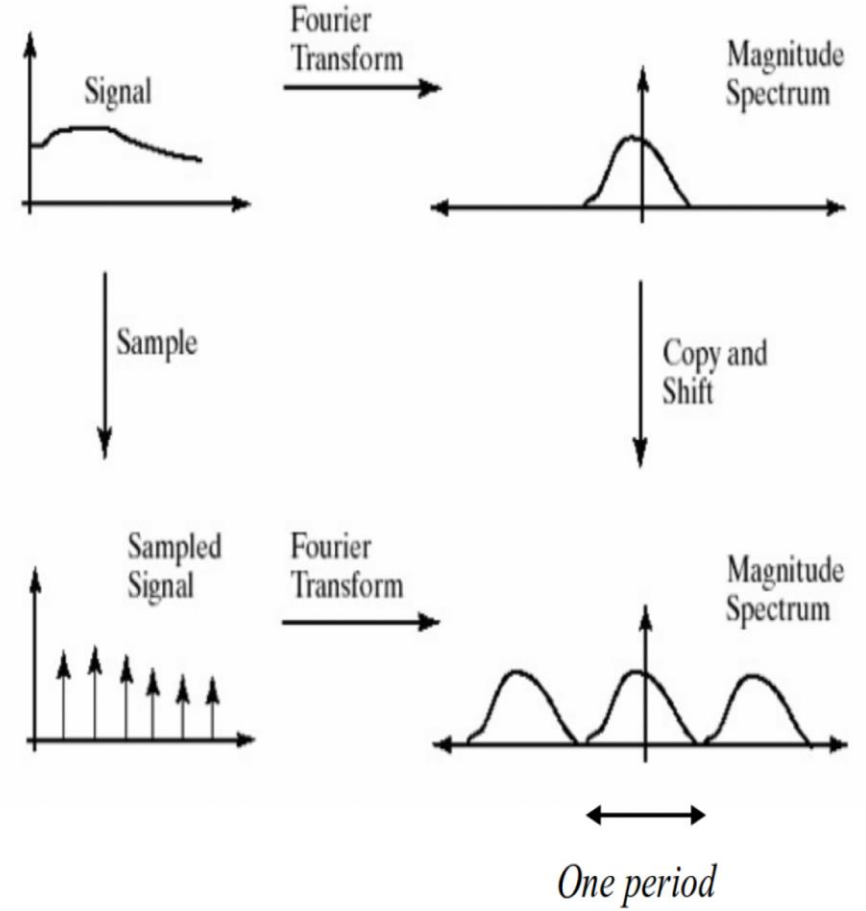
- **Continuous Fourier transform (FT):**

$$H(\omega) = \int_{-\infty}^{\infty} h(x) e^{-j\omega x} dx$$

- **Discrete Fourier Transform (DFT):**

$$H(\omega) = \sum_{0}^{N-1} h(x) e^{-j\frac{2\pi k}{N}} \quad ;$$

where N is the length of the sampled signal.



Credit: Elgammal, Rutgers University

Discrete Fourier Transform

$$F(k) = \sum_{x=0}^{x=N-1} f(x) e^{-i\frac{2\pi kx}{N}}$$

k = -N/2..N/2

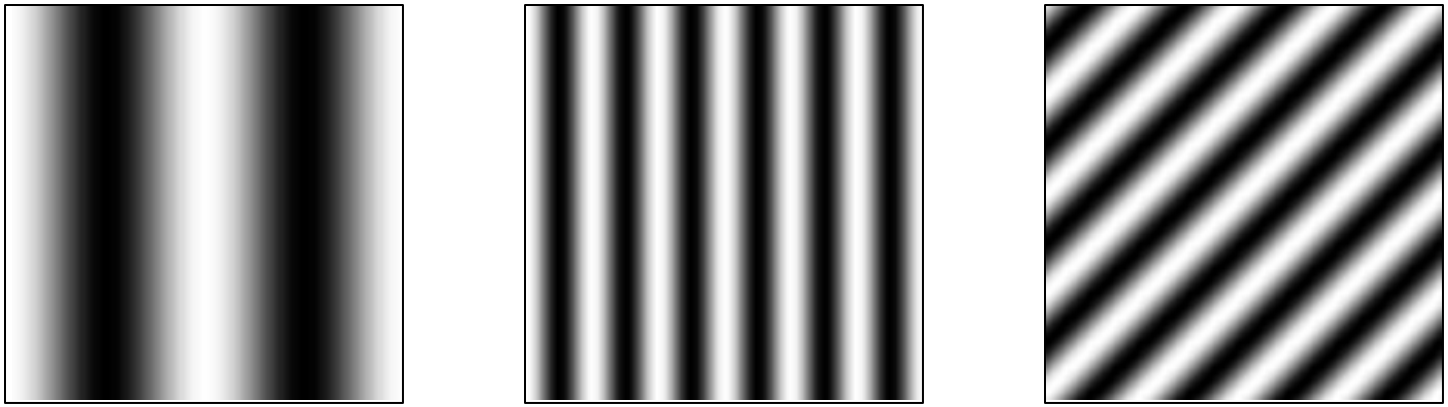
$$e^{ik} = \cos k + i \sin k \qquad i = \sqrt{-1}$$

$$F(k_x, k_y) = \sum_{x=0}^{x=N-1} \sum_{y=0}^{y=N-1} f(x,y) e^{-i\frac{2\pi(k_x x + k_y y)}{N}}$$

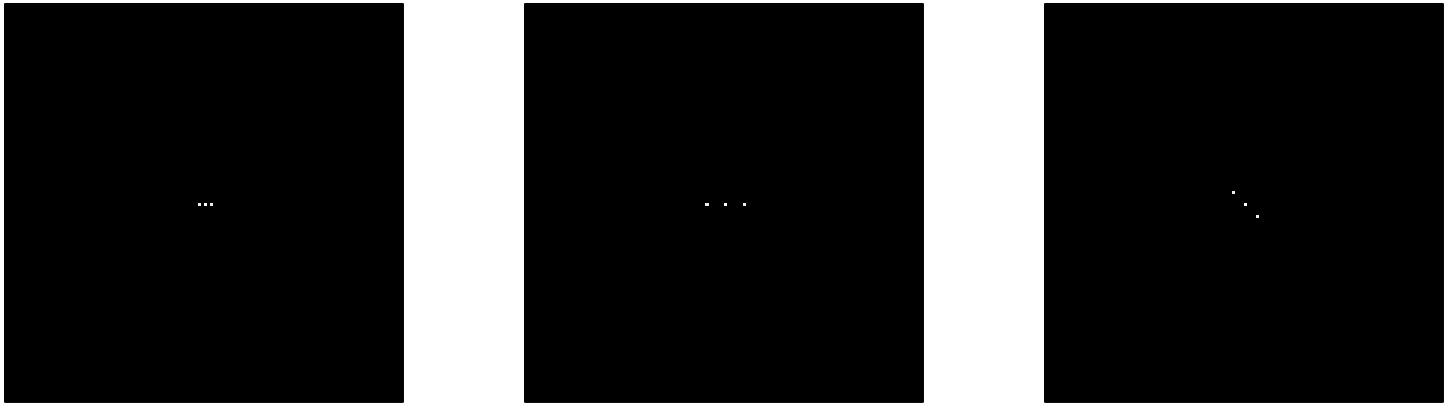
Credit: A. Bobick

Fourier analysis in images

Intensity Image



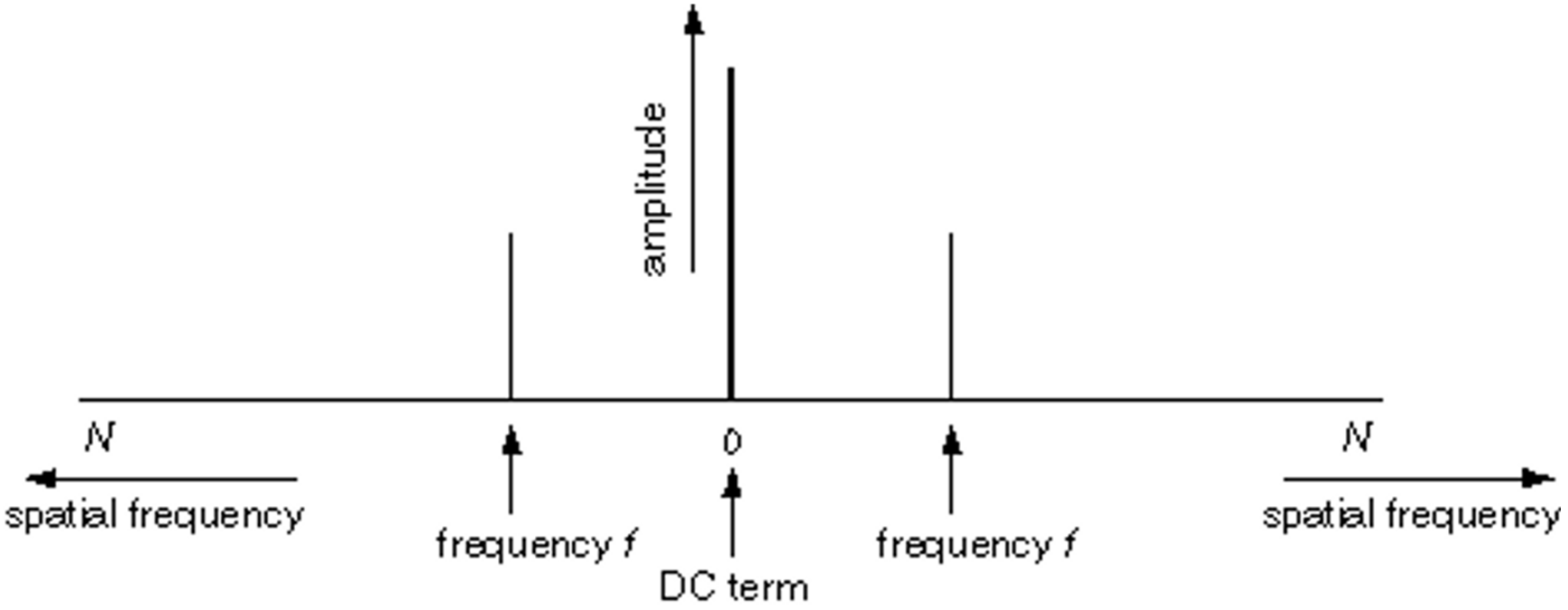
Fourier Image



http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

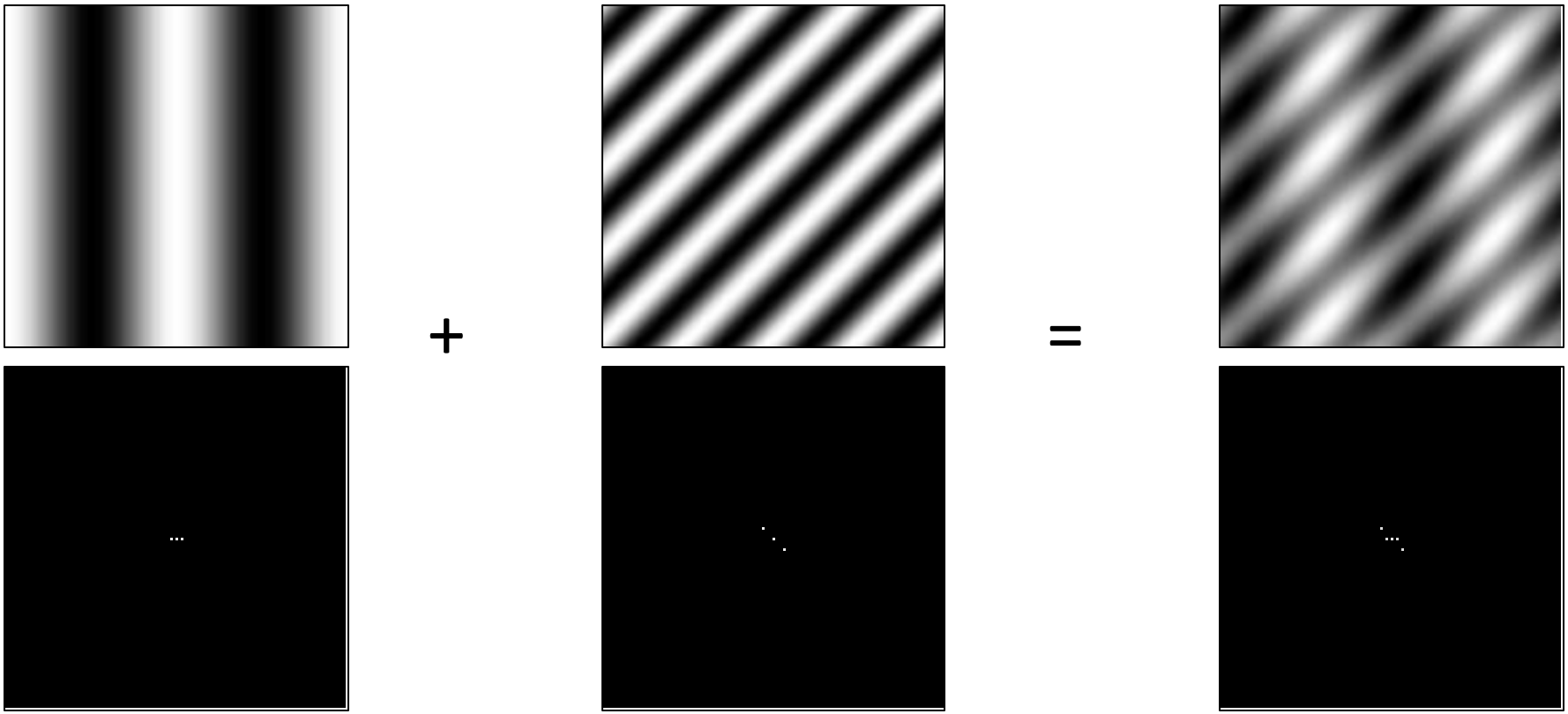
Fourier analysis in images

Mirror-image reflections along the axes



http://dsp.stackexchange.com/questions/4825/why-is-the-fft-mirrored

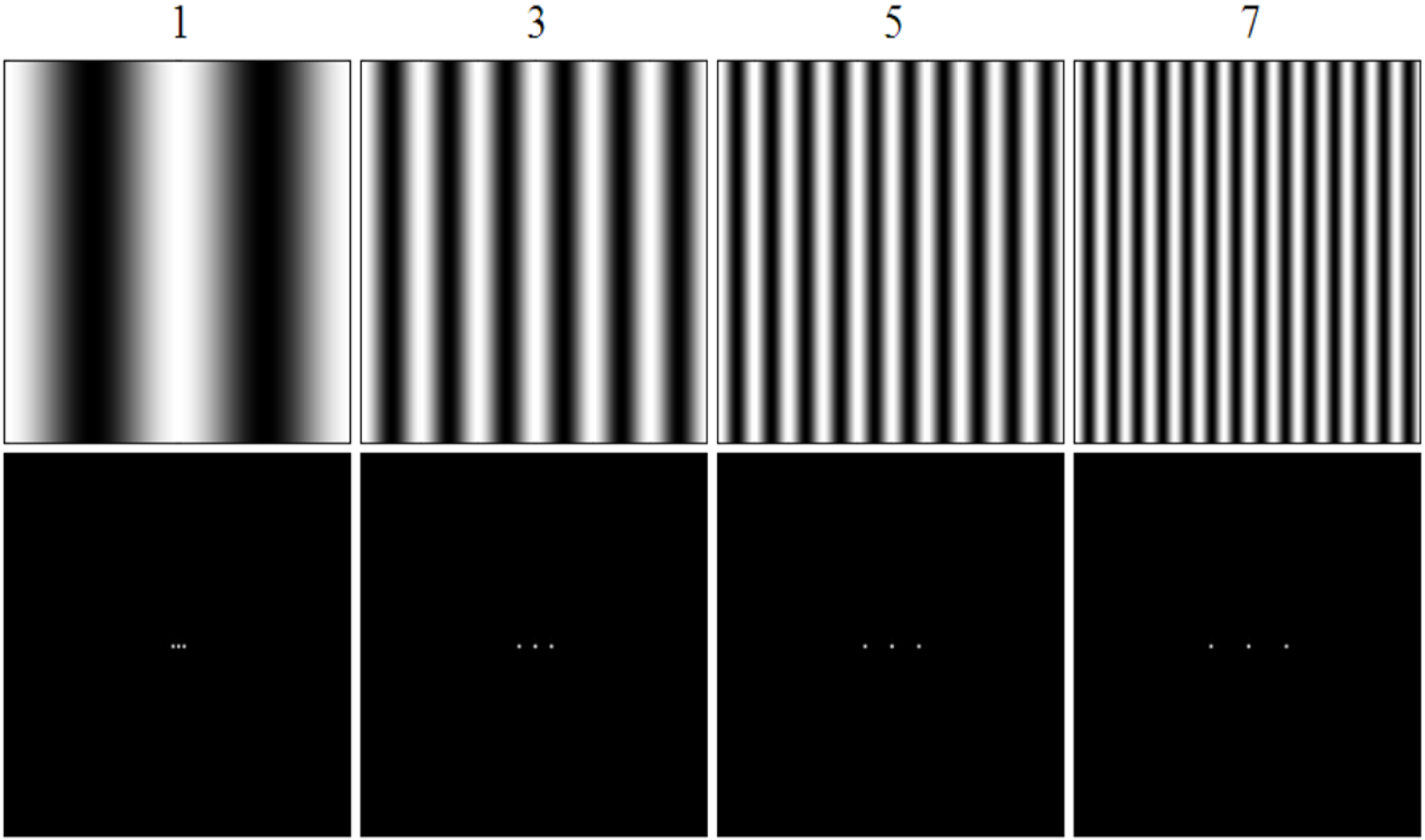
Signals can be composed



http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

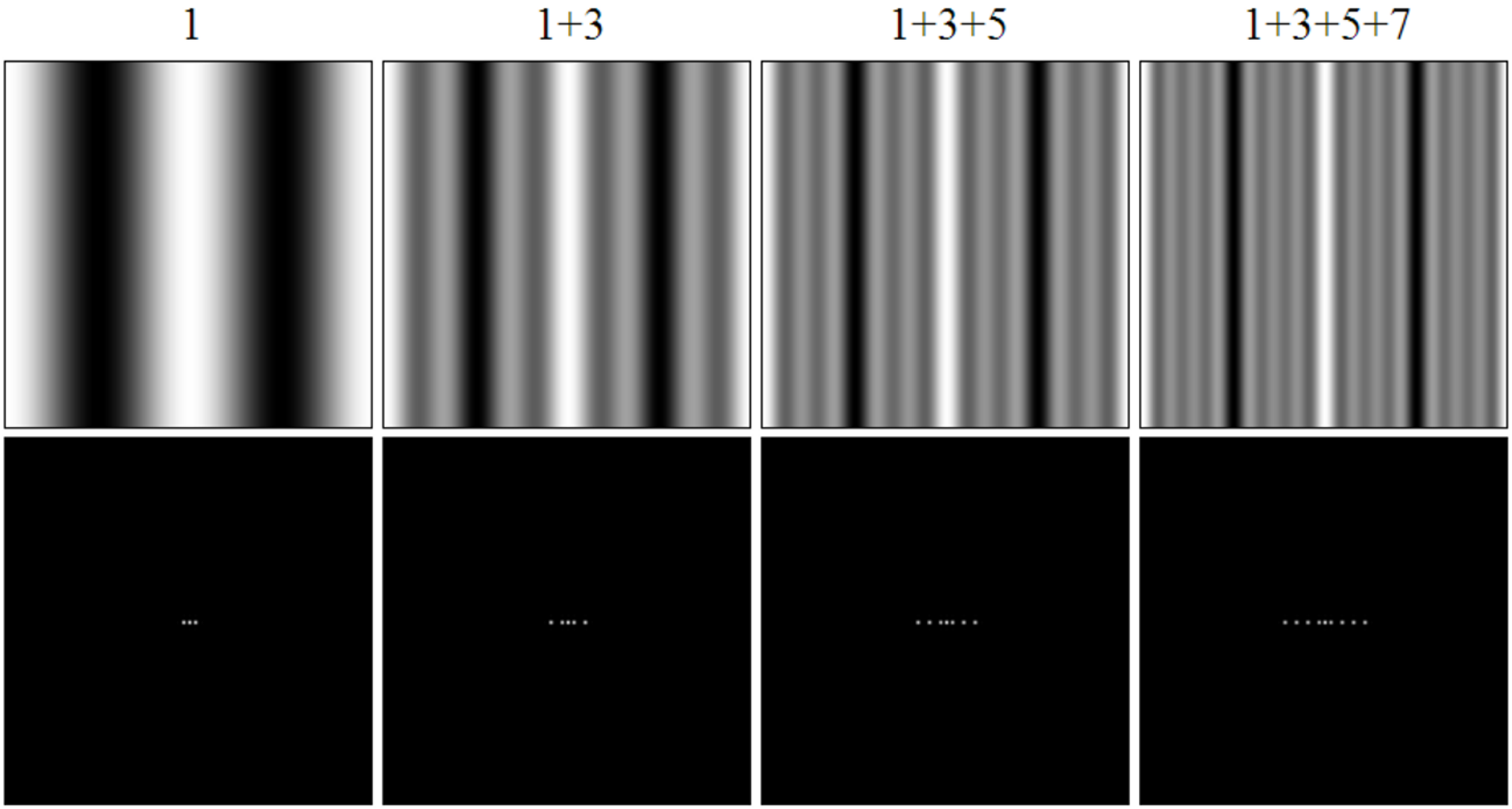
More: http://www.cs.unm.edu/~brayer/vision/fourier.html

Fourier analysis in images

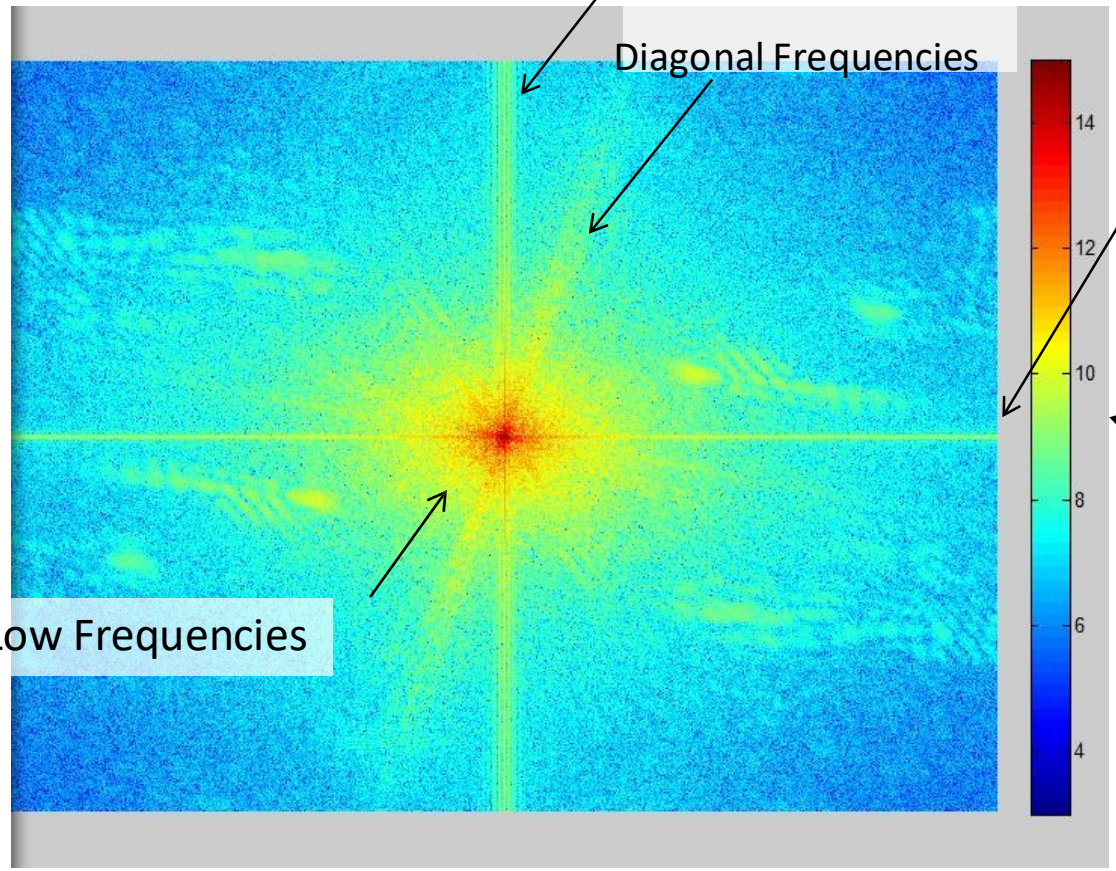


http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

Signals can be composed

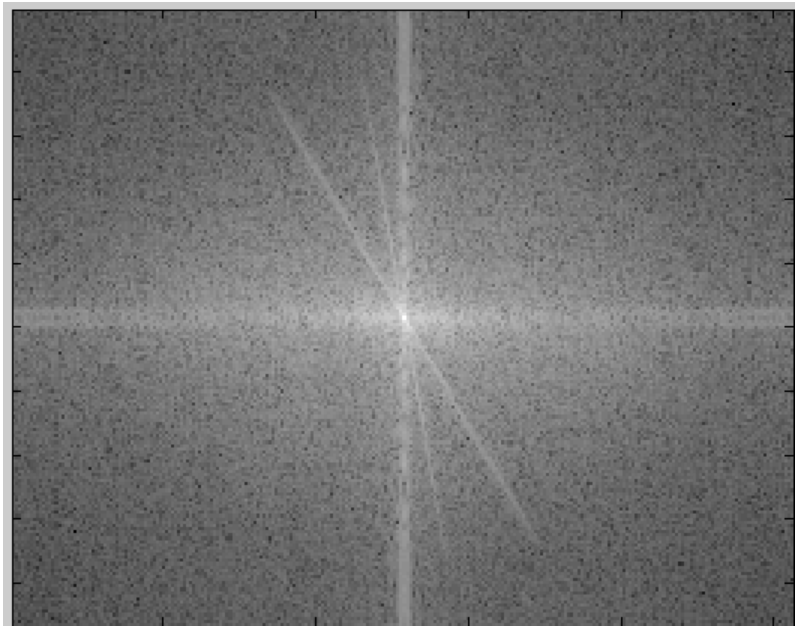


http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering
More: http://www.cs.unm.edu/~brayer/vision/fourier.html


Strong Vertical Frequency
(Sharp Horizontal Edge)
Diagonal Frequencies
Strong Horz. Frequency
(Sharp Vert. Edge)
Log Magnitude
Low Frequencies
14
12
10
8
6
4

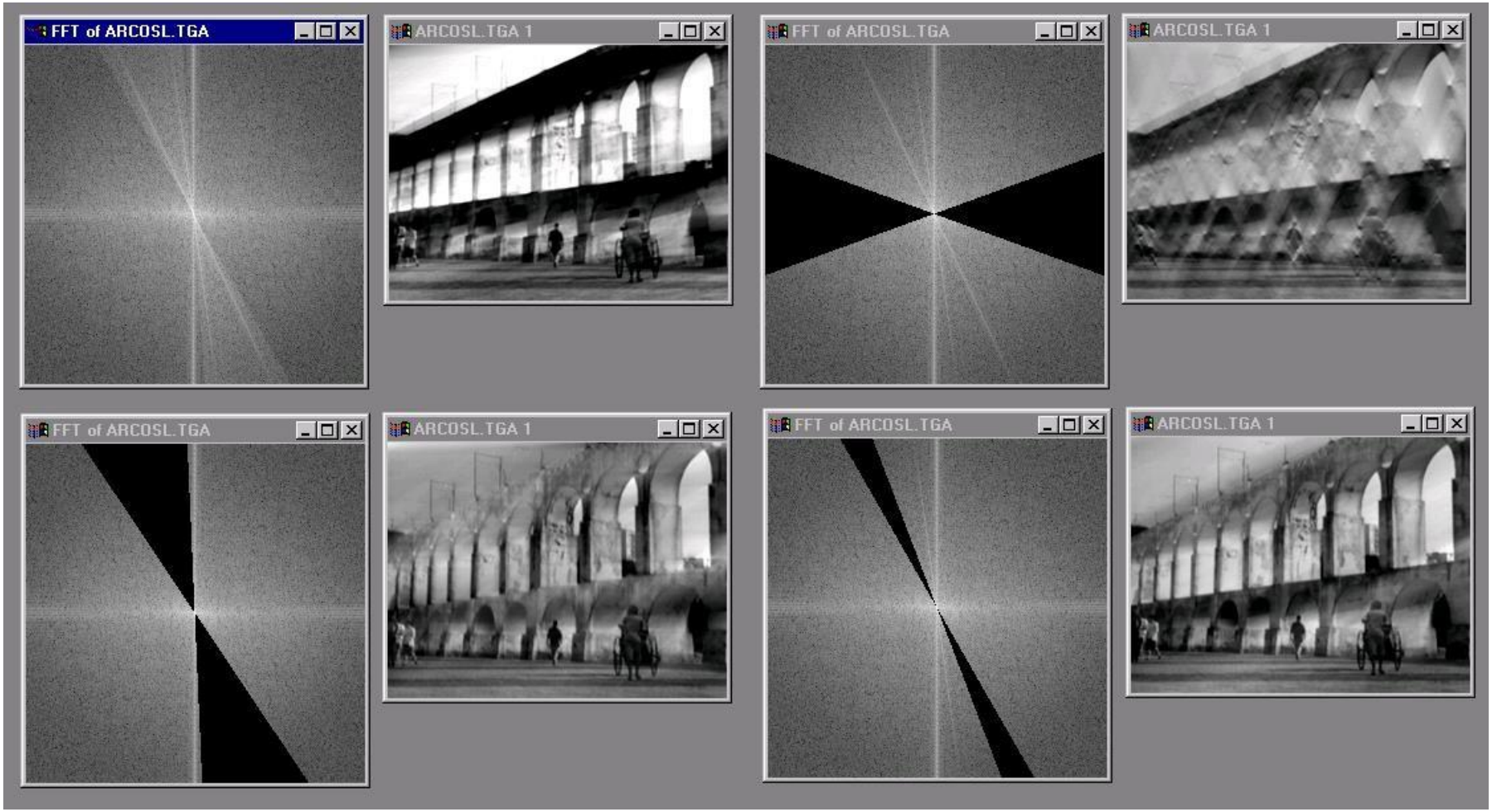
Slide: Huang

Example



Slide: Huang

Can change spectrum, then reconstruct

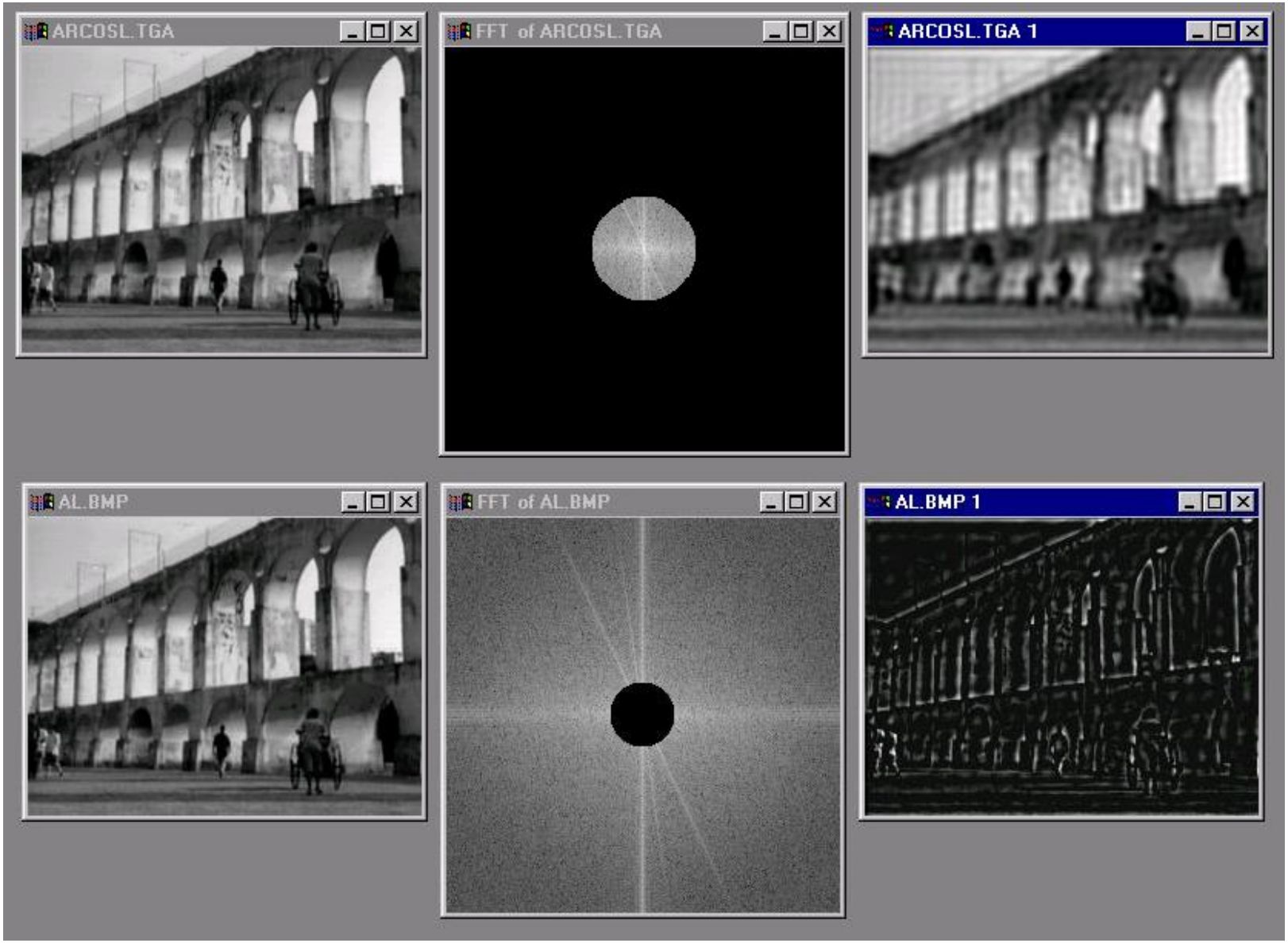


Slide: Huang

Low and High Pass filters

- Low-Pass Filters: Filters that allow low frequencies to pass through (block high frequencies).
  - Example: Gaussian filter
- High-Pass Filters: Filters that allow high frequencies to pass through (block low frequencies).
  - Example? Edge filter

Low and High Pass filtering



Slide: Huang

Fast Fourier Transform (FFT)

- Number of arithmetic operations to compute FT of N numbers (i.e., function defined at N points) is proportional to N^2
- Possible to reduce this to $N \log(N)$ using Fast Fourier Transform (FFT)
- FFT is a recursive divide-and-conquer algorithm for computing DFT
- Applications of FFT? Examples: Convolution, correlation

For more, see https://www.karlsims.com/fft.html

The Convolution Theorem

- The Fourier transform of the convolution of two functions is the product of their Fourier transforms

$$\mathrm{F}[g * h] = \mathrm{F}[g]\mathrm{F}[h]$$

Slide: Huang

The Convolution Theorem

- The Fourier transform of the convolution of two functions is the product of their Fourier transforms

$$\mathrm{F}[g * h] = \mathrm{F}[g]\mathrm{F}[h]$$

- The inverse Fourier transform of the product of two Fourier transforms is the convolution of the two inverse Fourier transforms

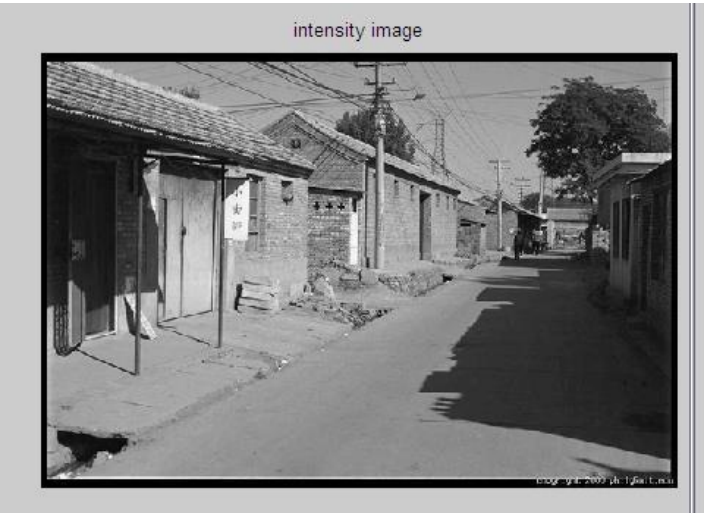
$$\mathrm{F}^{-1}[\mathrm{G}H] = \mathrm{F}^{-1}[G] * \mathrm{F}^{-1}[H]$$

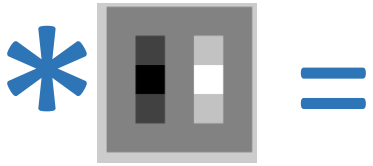
- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!

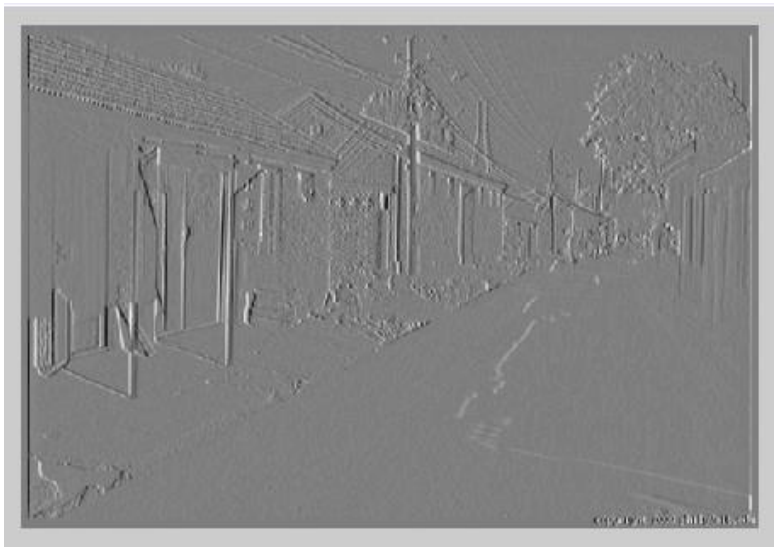
Slide: Huang

Filtering in spatial domain

-1	0	1
-2	0	2
-1	0	1

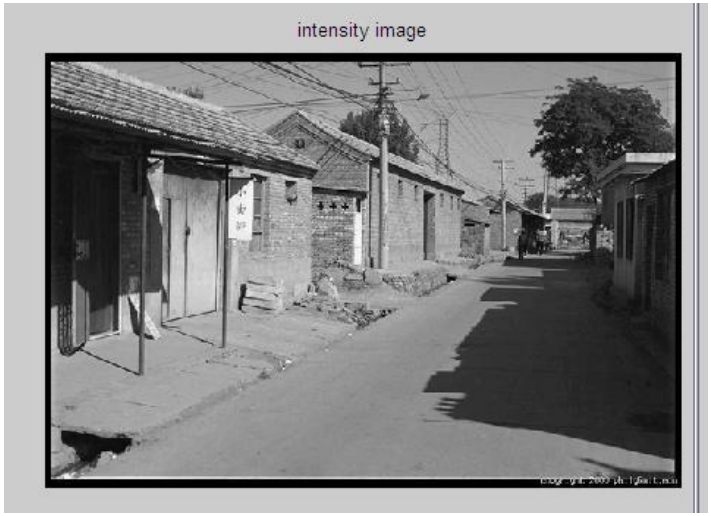


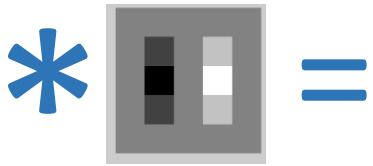


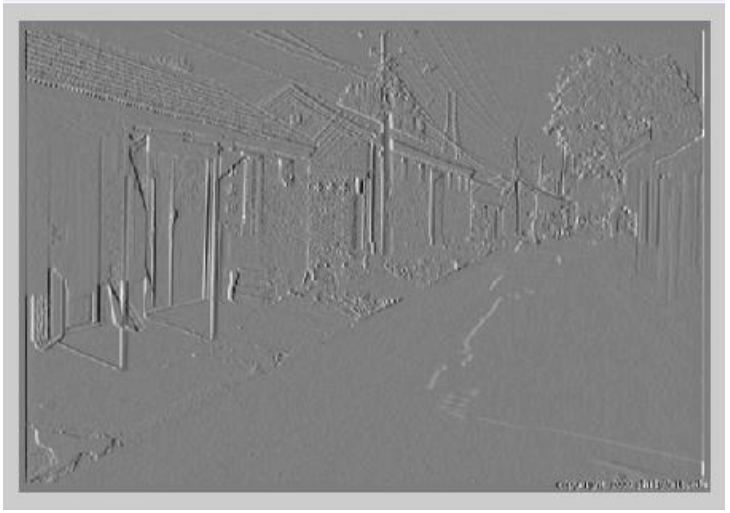


Slide: Huang

Spatial domain

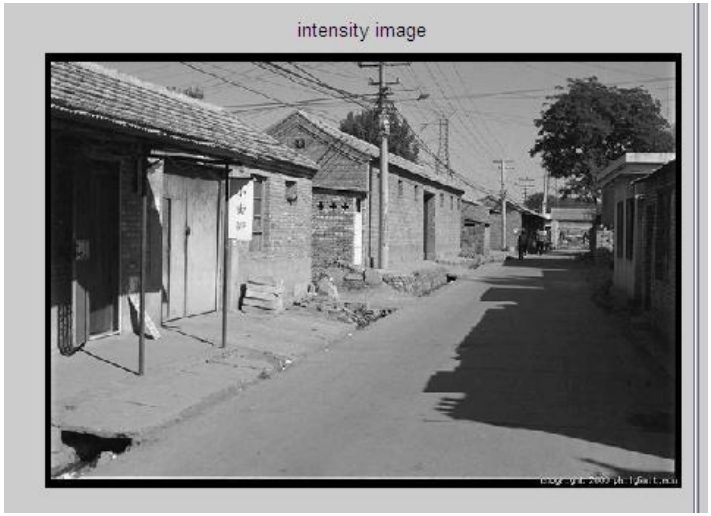


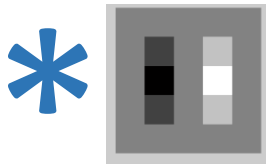


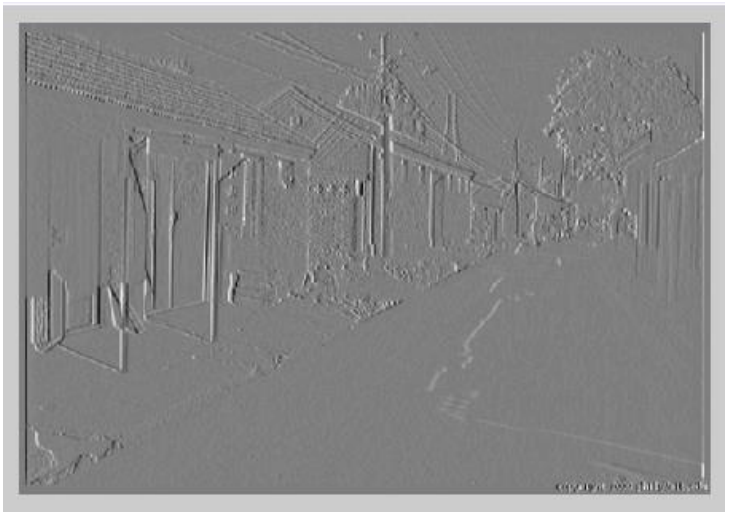


Slide: Huang

Spatial domain





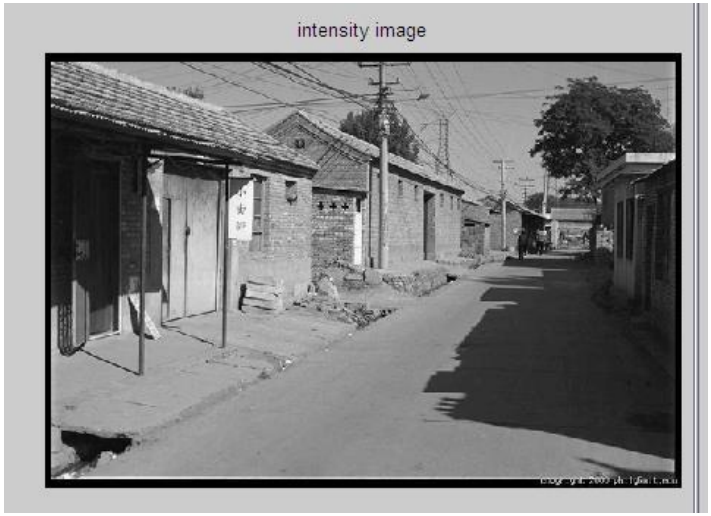


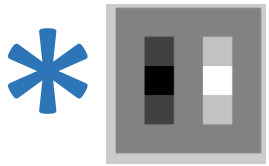
FFT

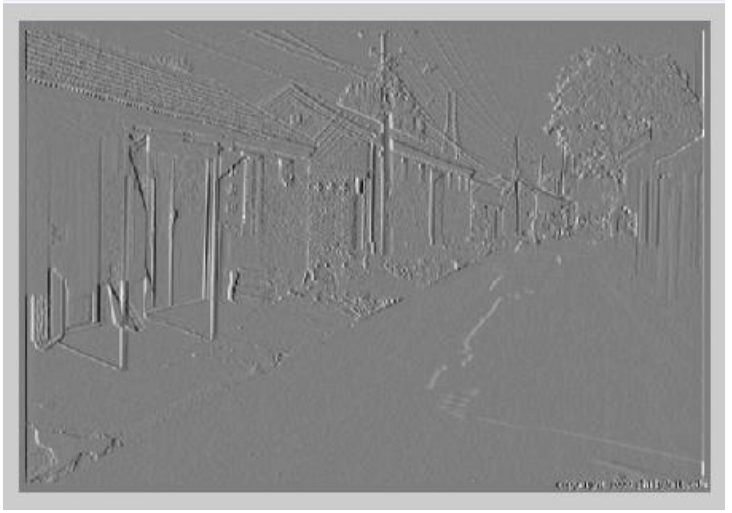
FFT

Slide: Huang

Spatial domain

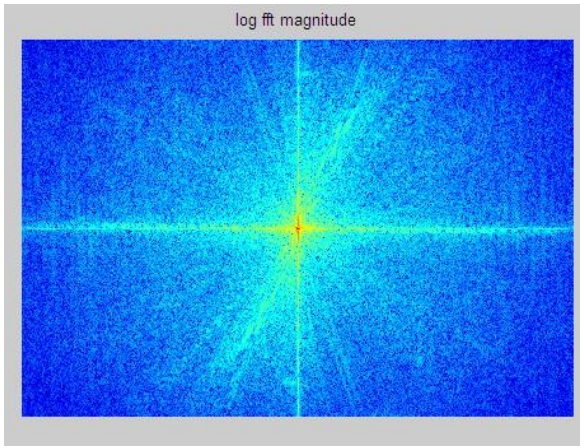


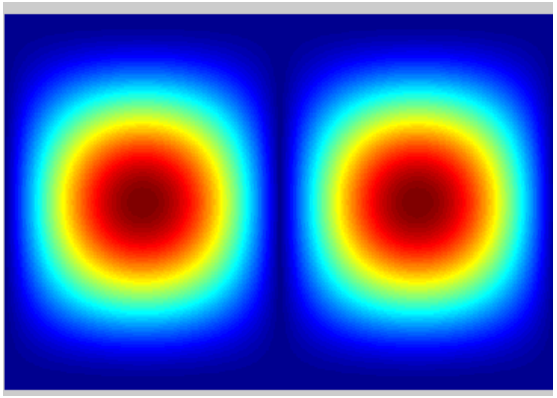




FFT

FFT

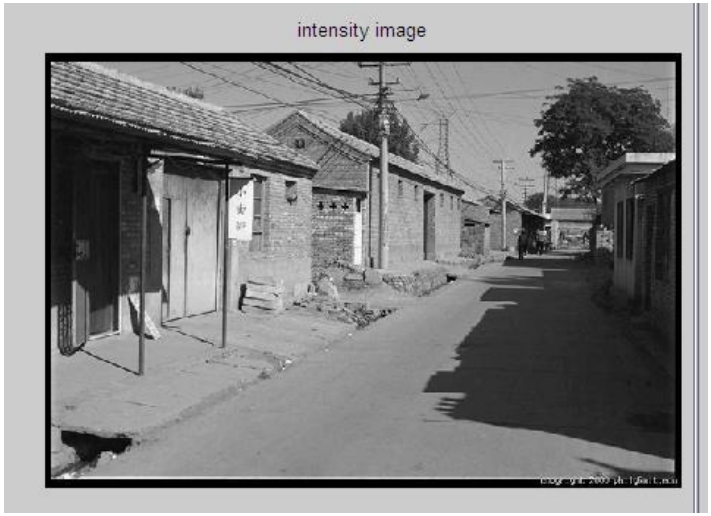


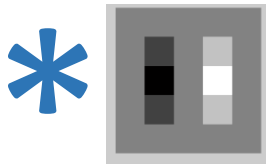


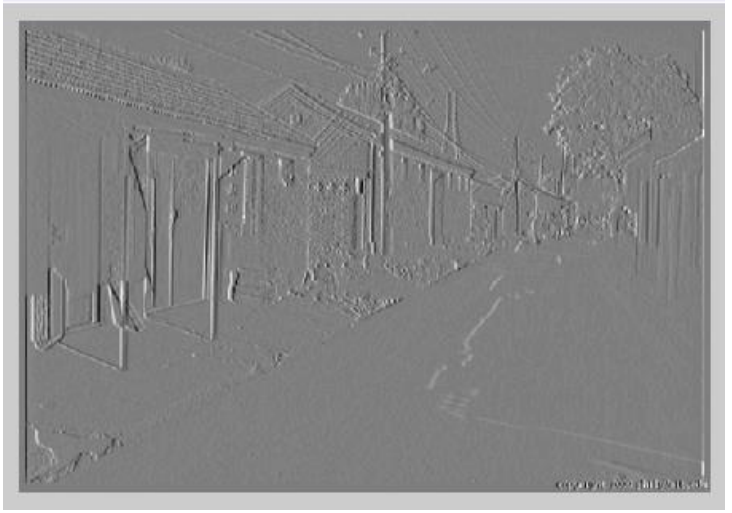
Frequency domain

Slide: Huang

Spatial domain



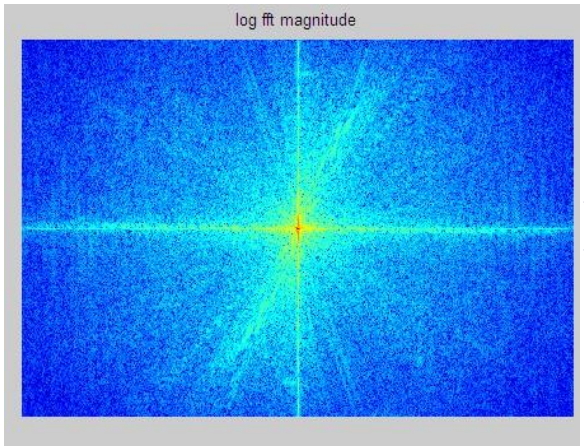


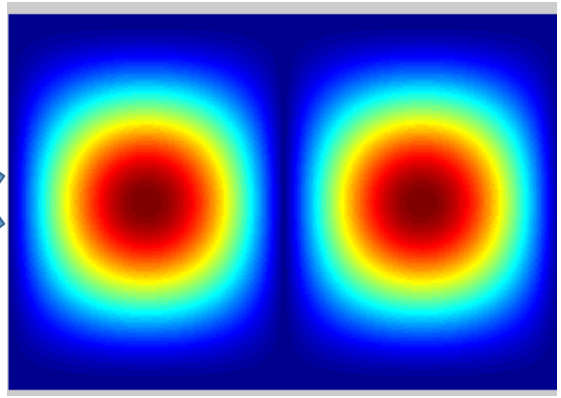


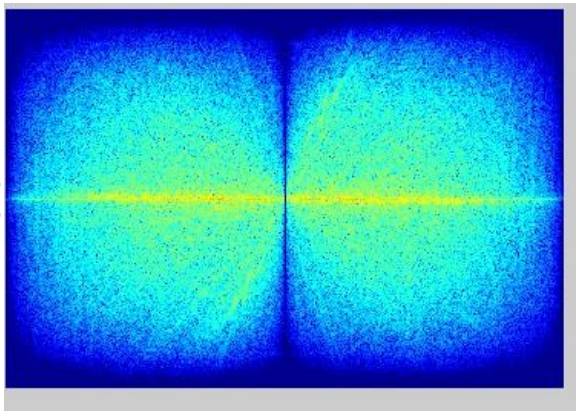
FFT

FFT

Inverse FFT







Frequency domain

Slide: Huang

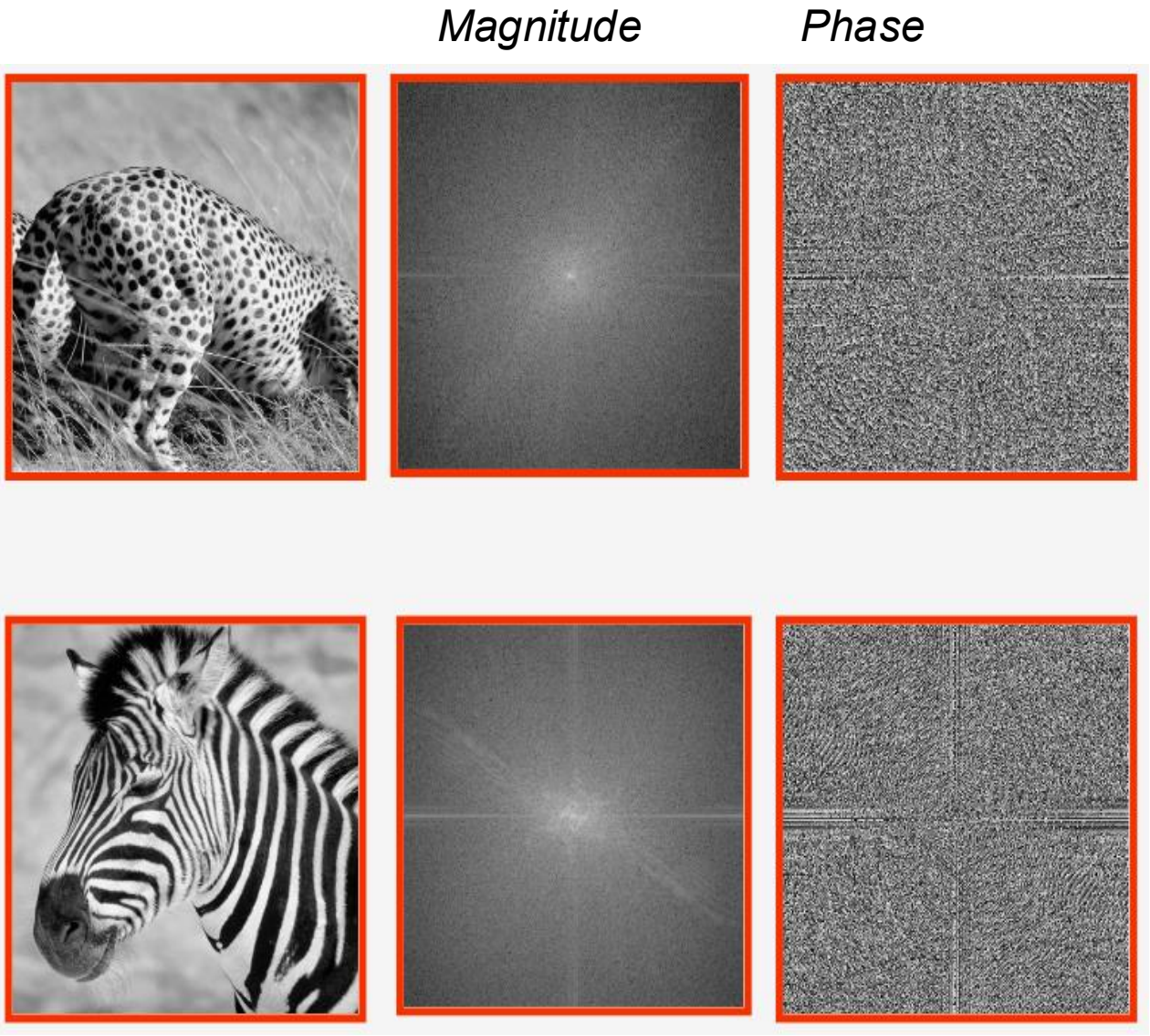
Questions

Which has more information, the phase or the magnitude?

What happens if you take the phase from one image and combine it with the magnitude from another image?

Slide: Huang

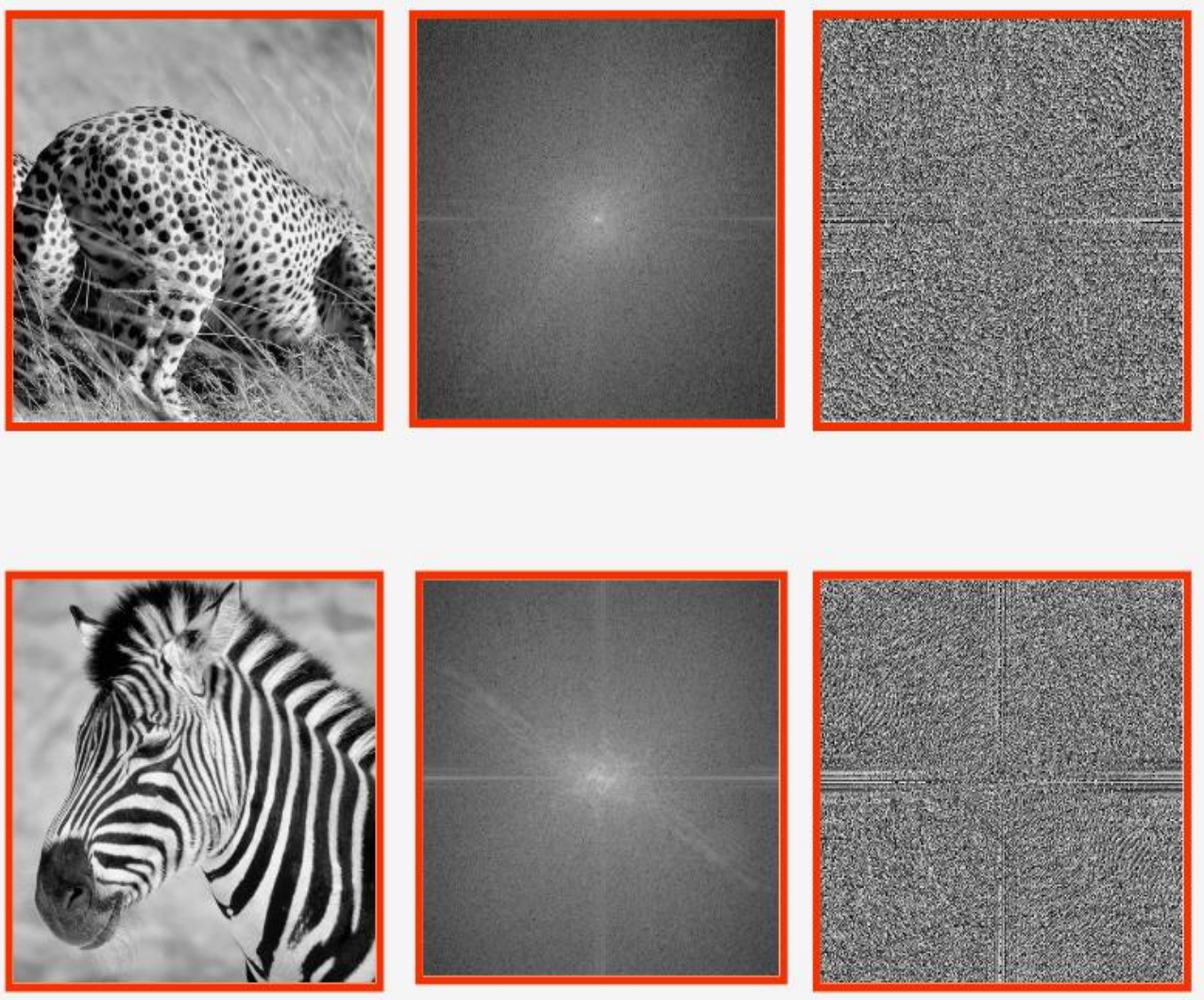
Phase vs. Magnitude



Slide: Huang

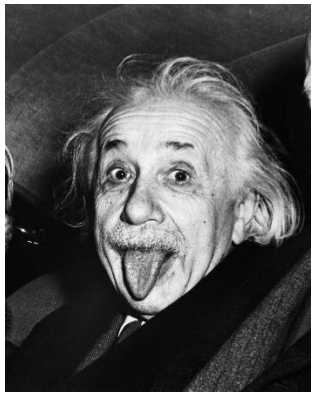
Phase vs. Magnitude

Magnitude *Phase* *Swap phase and reconstruct*



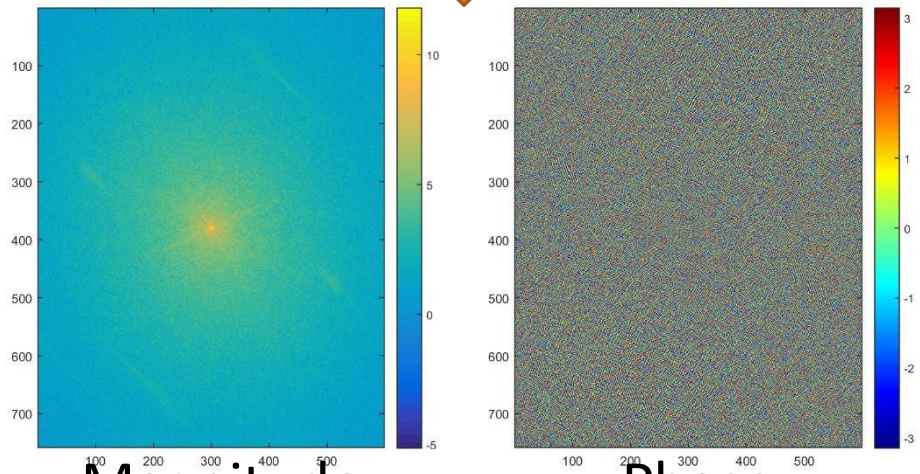
Credit: Forsyth and Ponce, Computer Vision: A Modern Approach, 2003

Phase vs. Magnitude



Intensity image

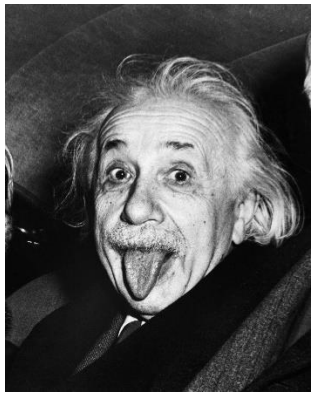
FFT



Magnitude Phase

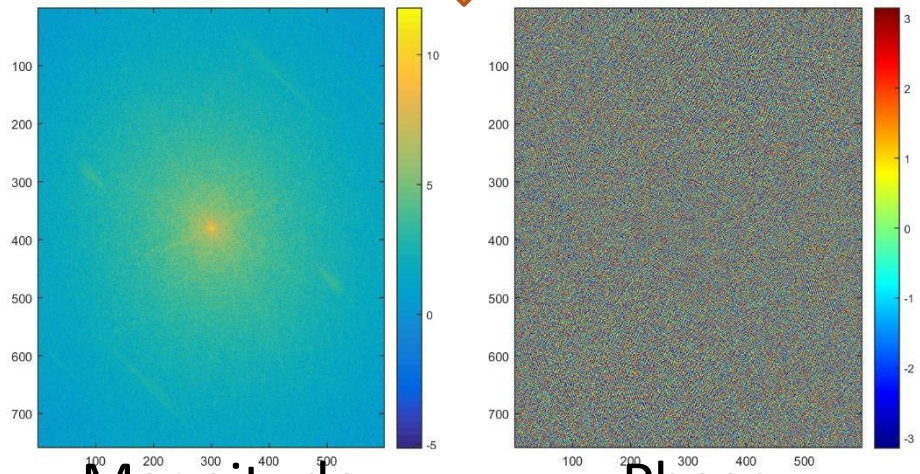
Slide: Huang

Phase vs. Magnitude



Intensity image

FFT

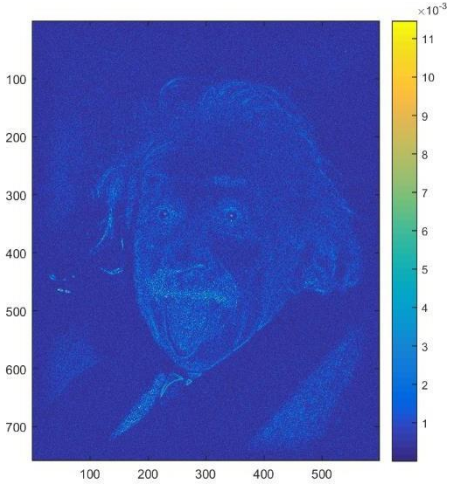


Magnitude

Phase

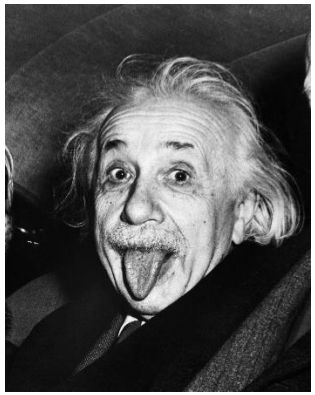
Use random magnitude

Inverse FFT



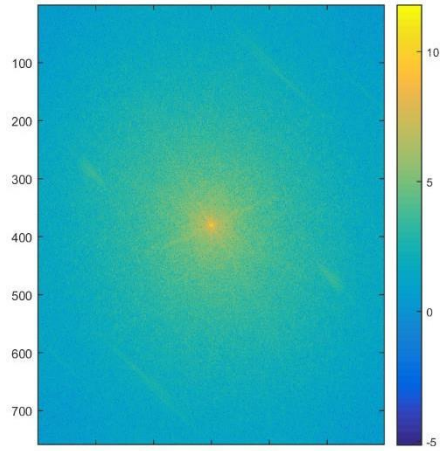
Slide: Huang

Phase vs. Magnitude

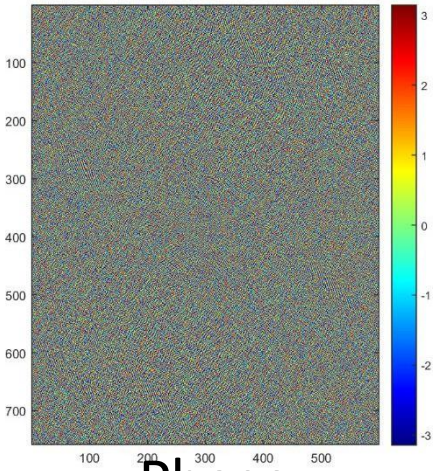


Intensity image

FFT



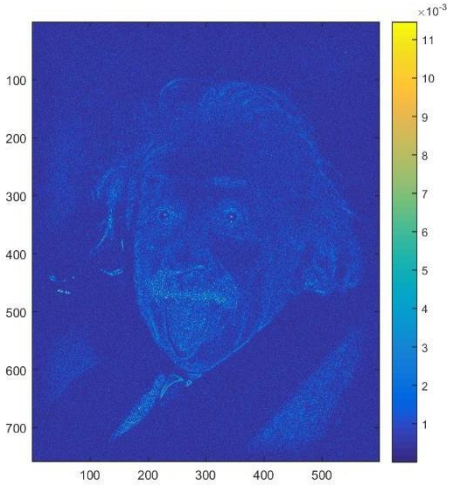
Magnitude



Phase

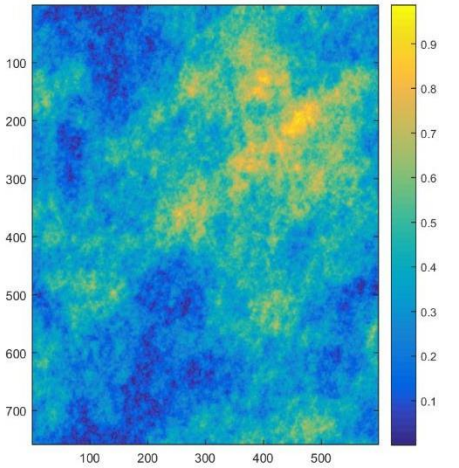
Use random magnitude

Inverse FFT



Use random phase

Inverse FFT

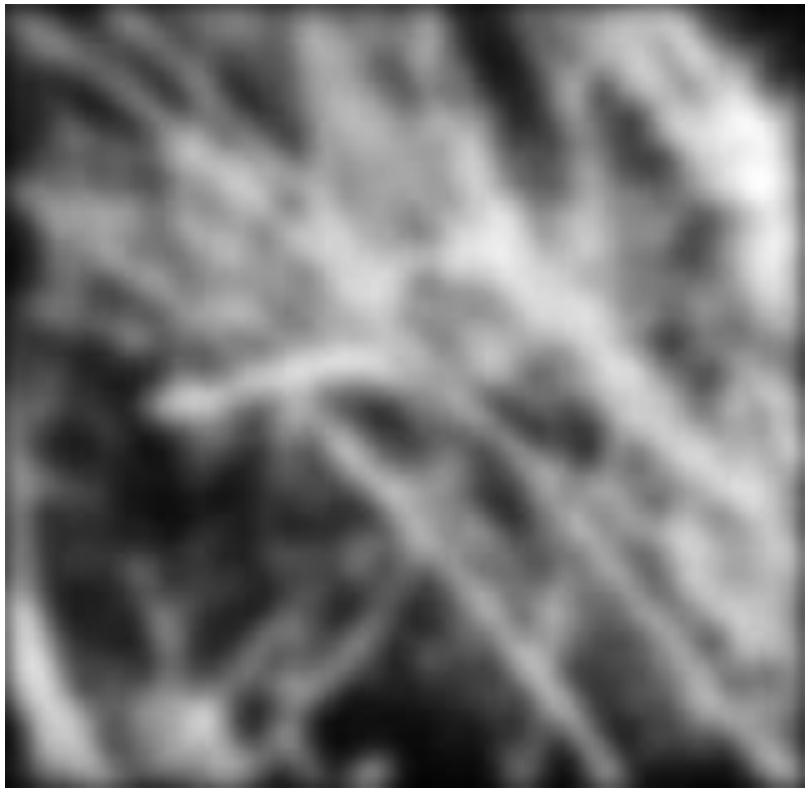


Slide: Huang

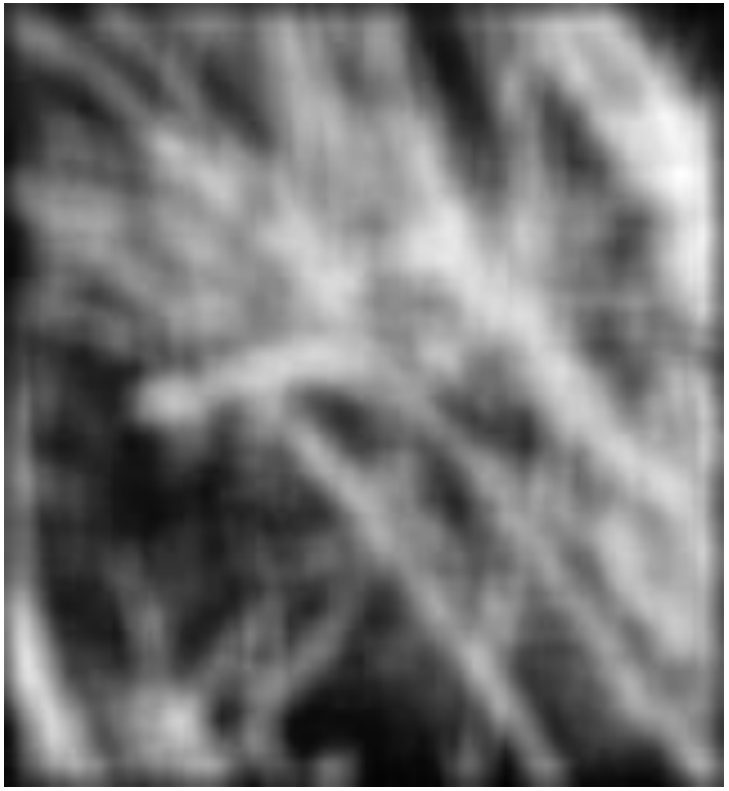
Filtering

Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

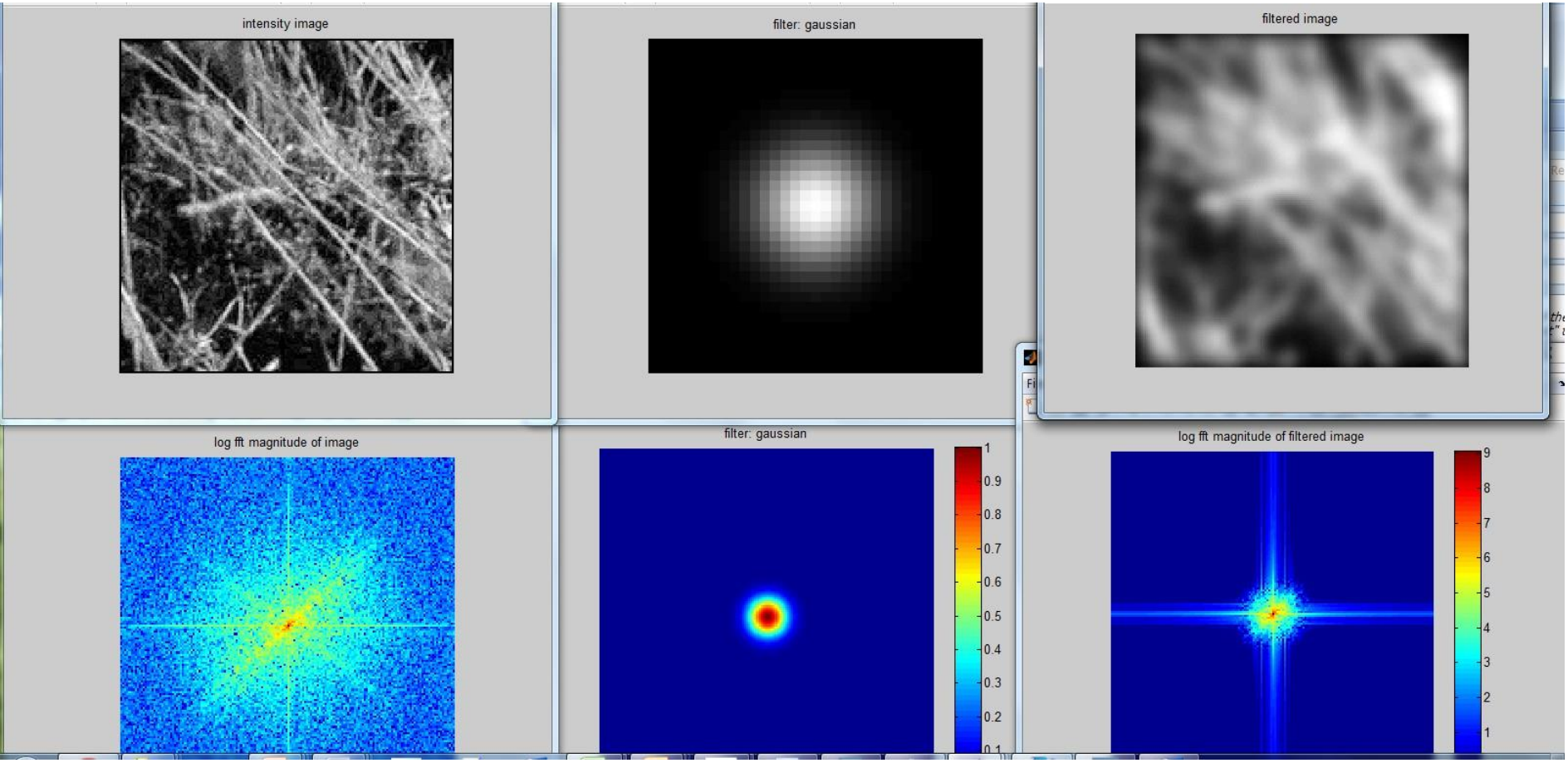
Gaussian 



Box filter 

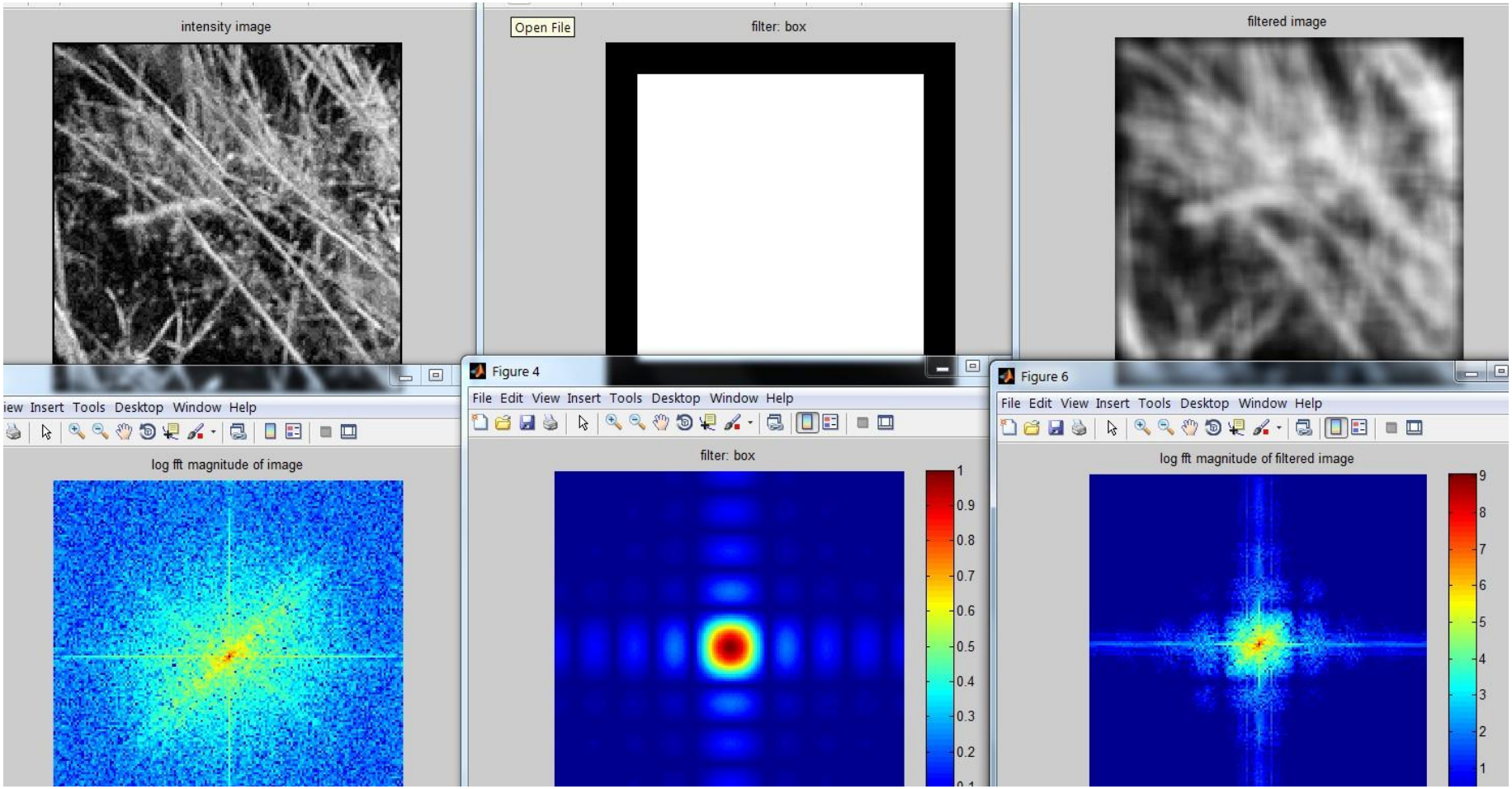


Gaussian



Slide: Huang

Box Filter



Slide: Huang

Image half-sizing

This image is too big to fit on the screen. How can we reduce it?

How to generate a half-sized version?

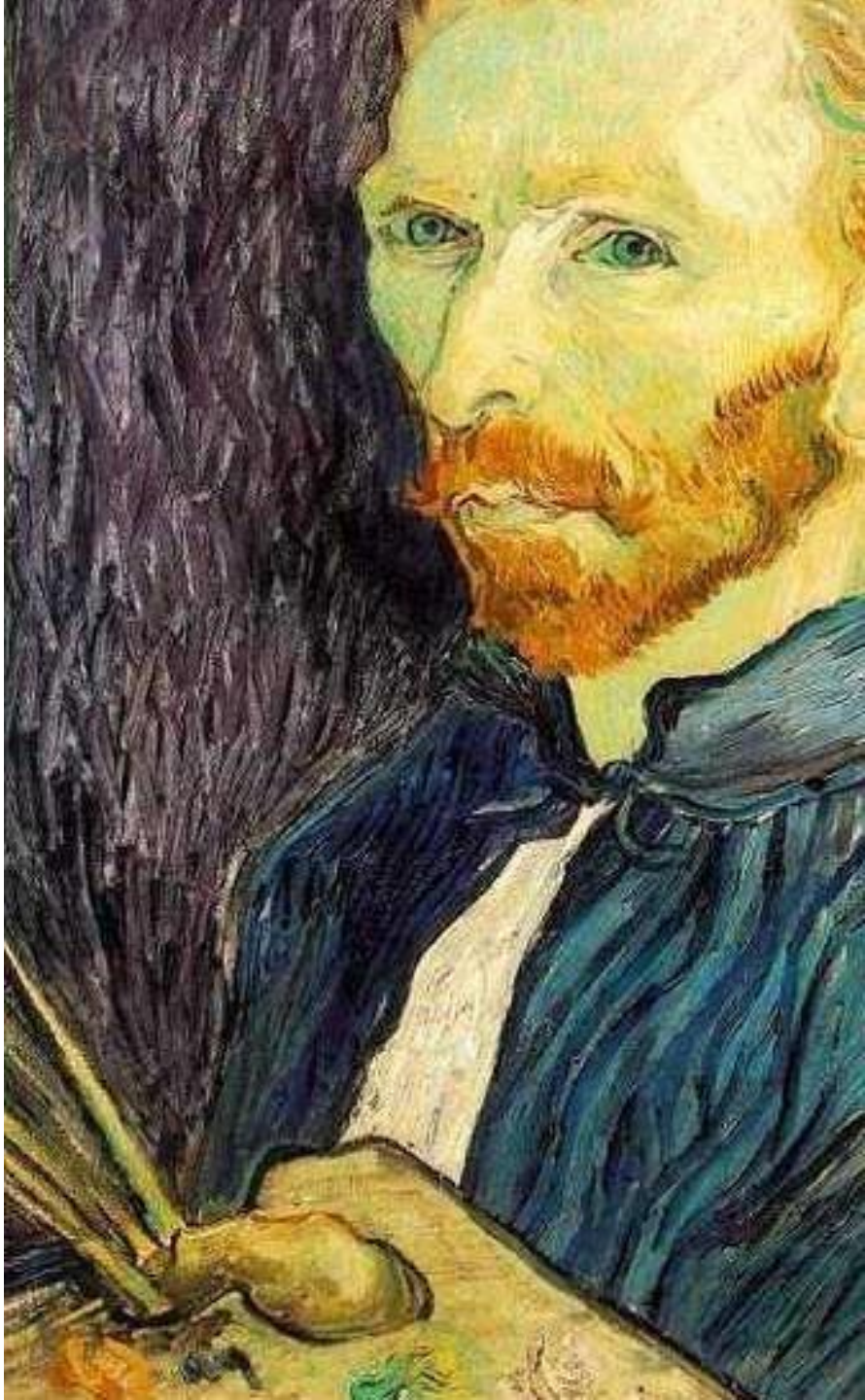
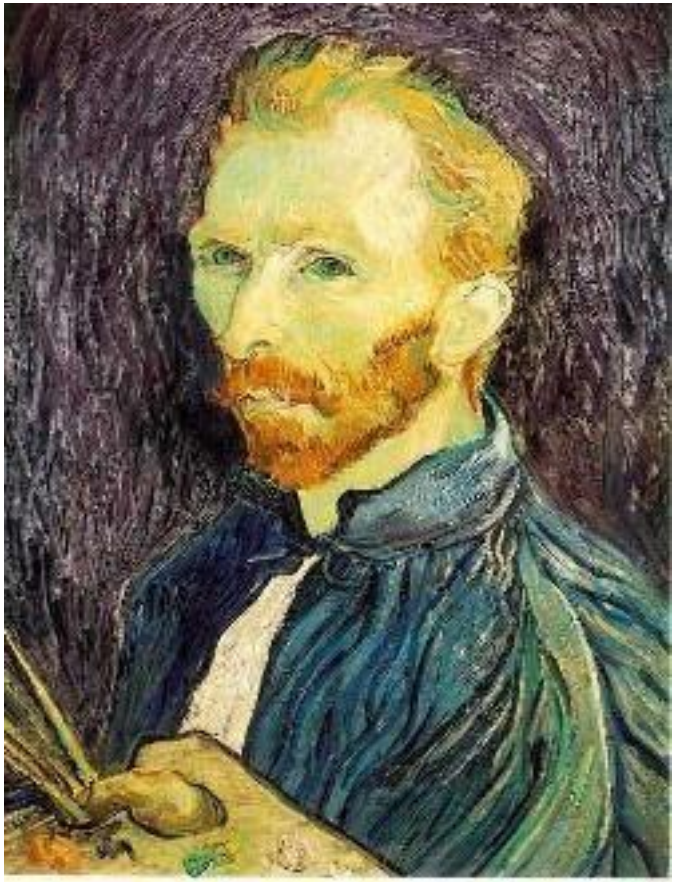


Image sub-sampling



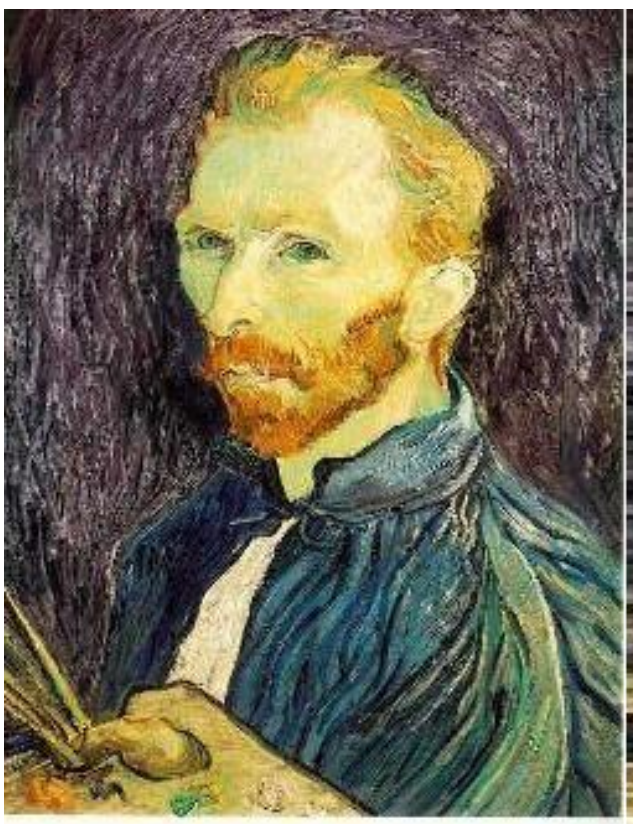
1/4

1/8

Throw away every other row and column to create a 1/2 size image
- called *image sub-sampling*

Slide by Steve Seitz

Image sub-sampling



1/2

1/4 (2x zoom)

1/8 (4x zoom)

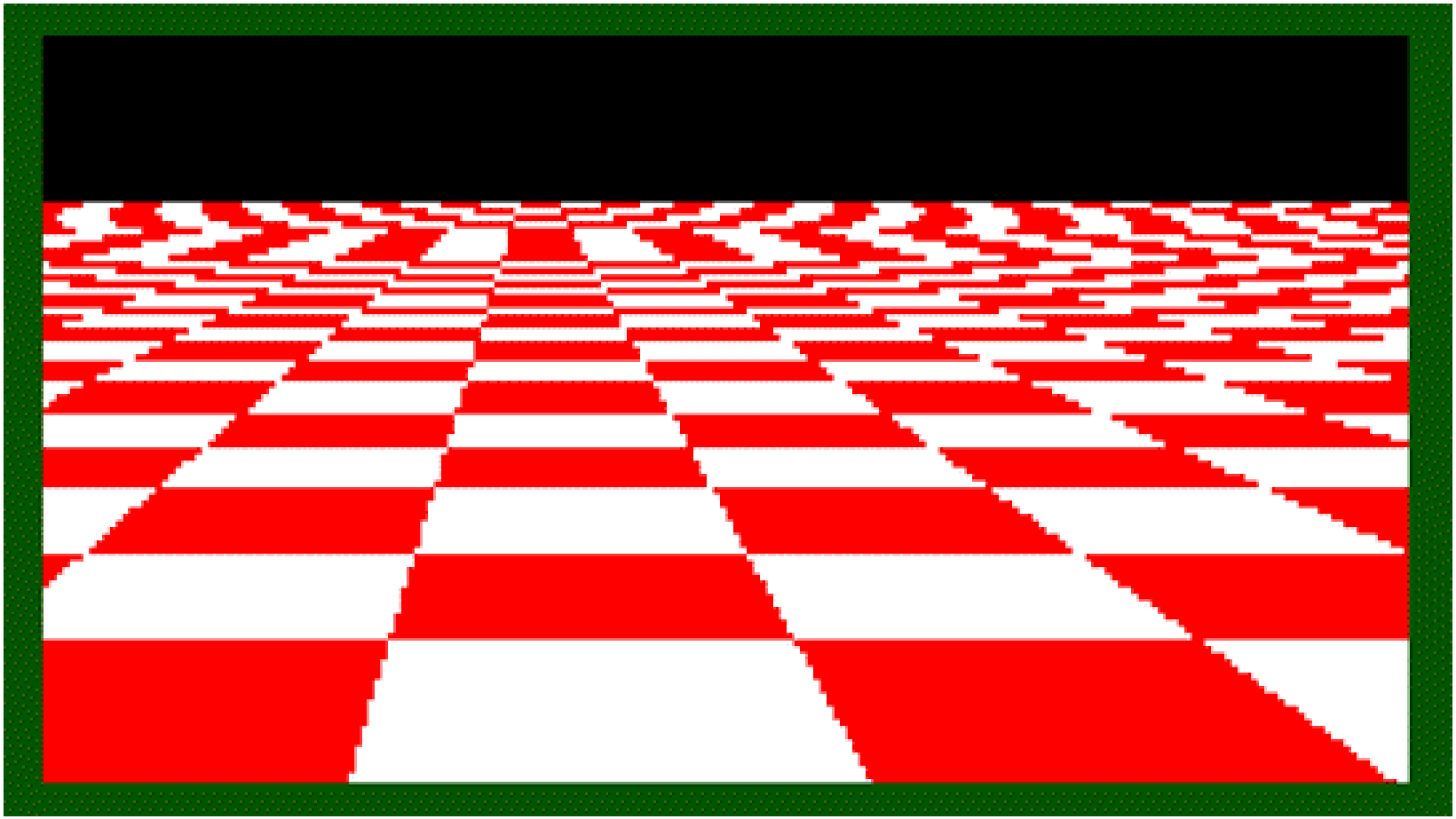
Why does this look so crufty?
Aliasing! What do we do?

Slide by Steve Seitz

Image sub-sampling

Source: F. Durand

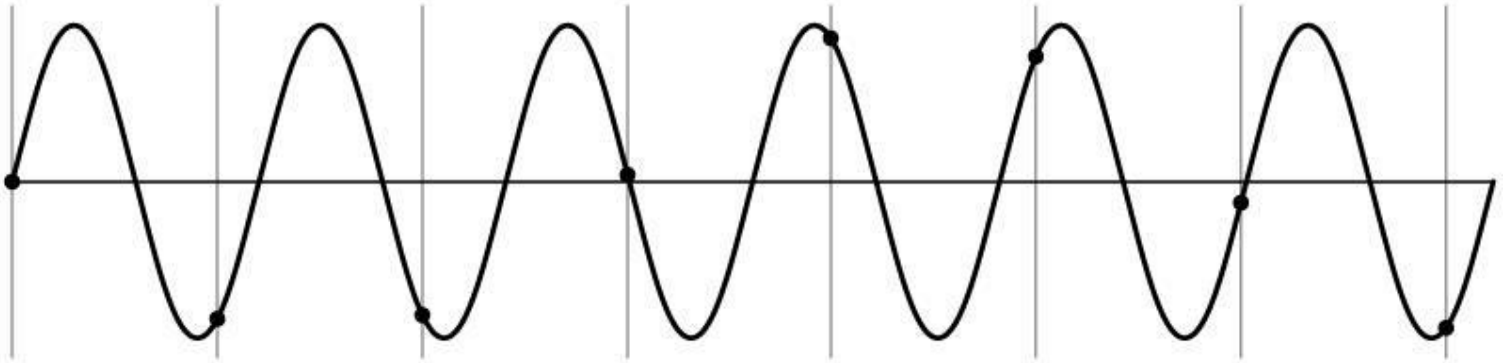
Even worse for synthetic images



Source: L. Zhang

Aliasing problem

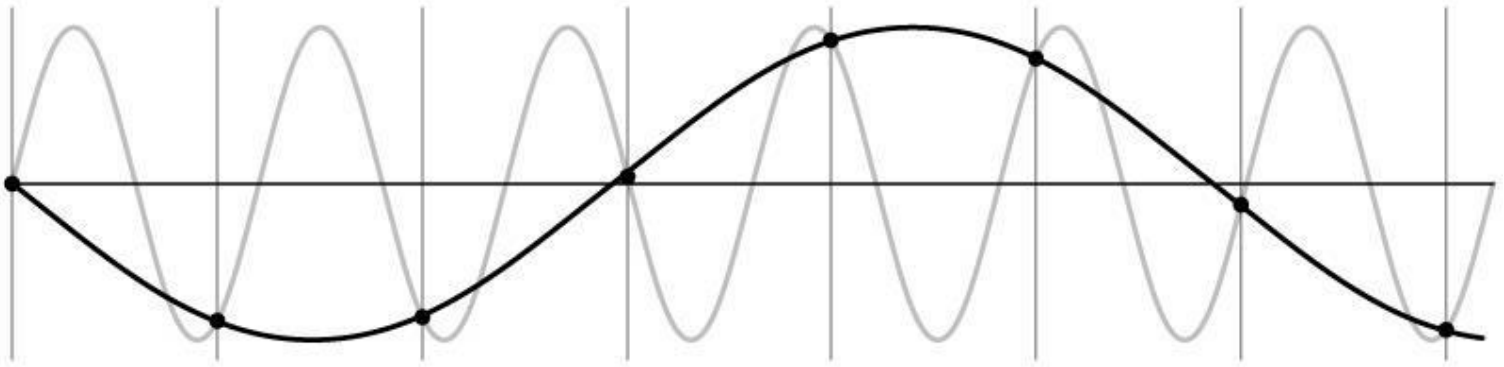
- 1D example (sinewave):



Source: S. Marschner

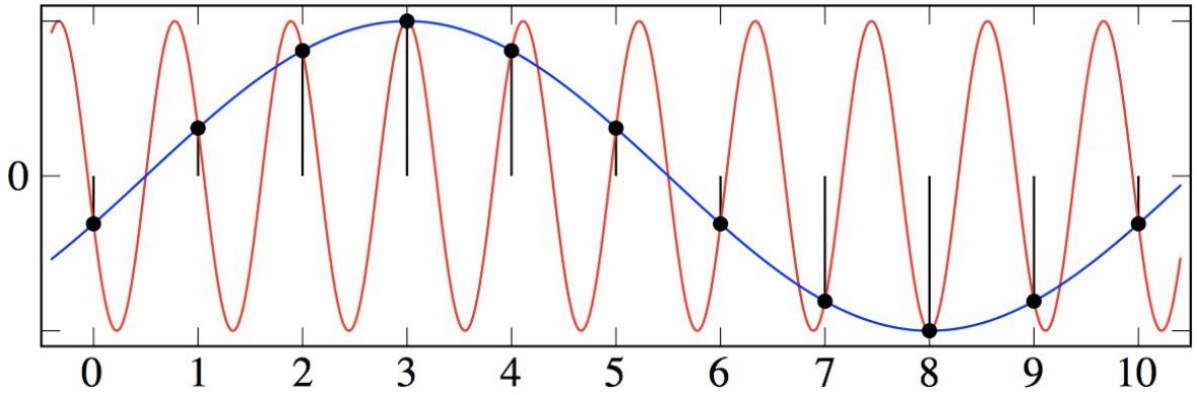
Aliasing problem

- 1D example (sinewave):



Source: S. Marschner

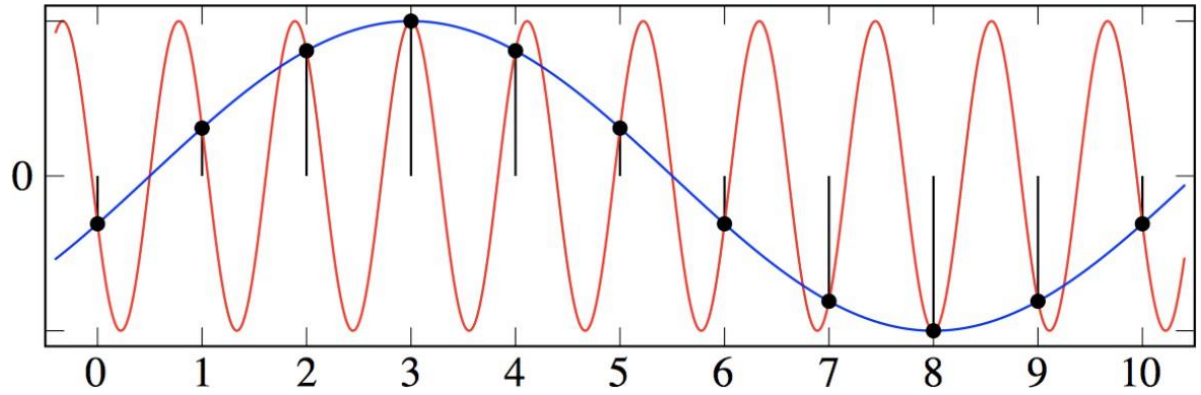
Aliasing



- Occurs when your sampling rate is not high enough to capture the amount of detail in your image
- High frequencies look like low frequencies
- Can give you the wrong signal/image—an *alias*

Source: L. Zhang

Aliasing

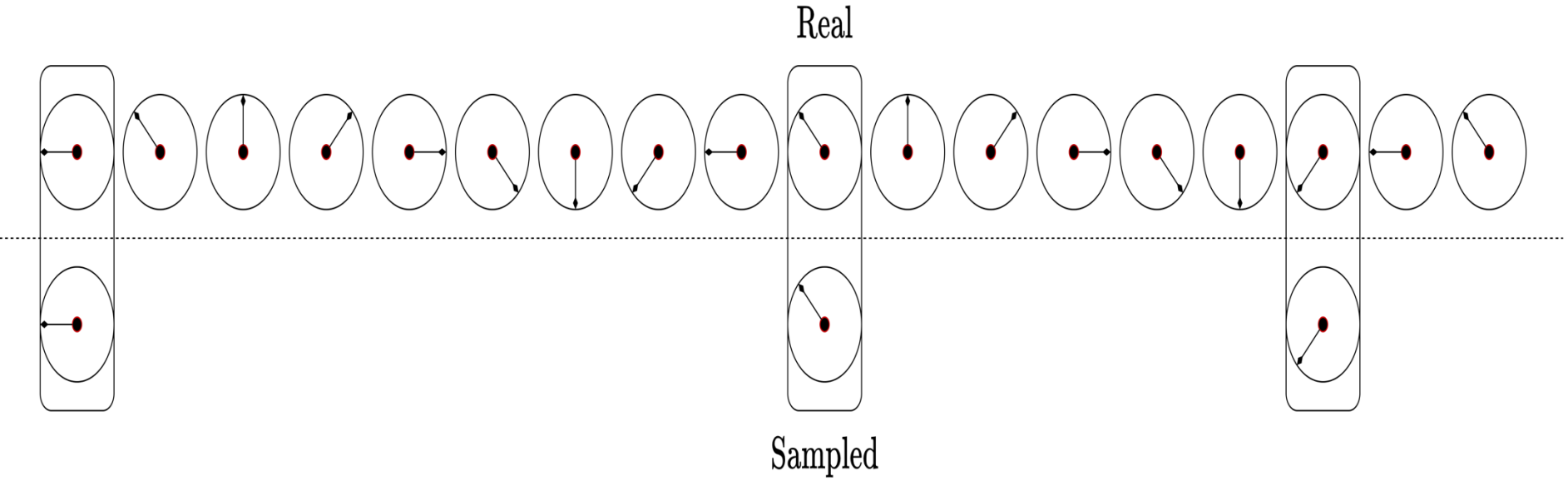


- Occurs when your sampling rate is not high enough to capture the amount of detail in your image
- Unresolved, high frequencies look like resolved low frequencies
- Can give you the wrong signal/image—an *alias*

- To avoid aliasing:
  - sampling rate ≥ 2 * max frequency in the image
    - said another way: ≥ two samples per cycle
  - this minimum sampling rate is called the **Nyquist rate**

Source: L. Zhang

Aliasing: Video

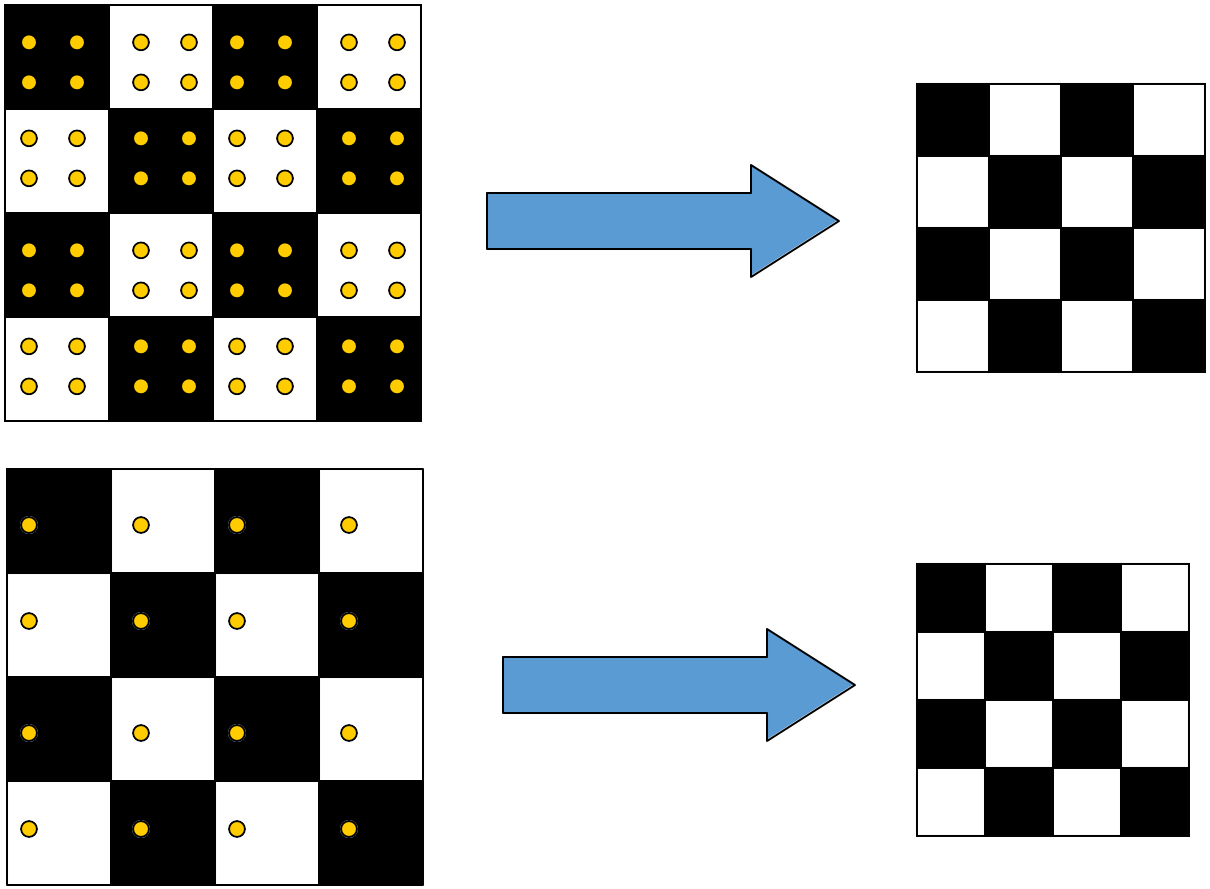


Wagon Wheel effect: Wheels spins in the opposite direction at high speed.

https://www.youtube.com/watch?v=jHS9JGkEOmA

Source: L. Zhang

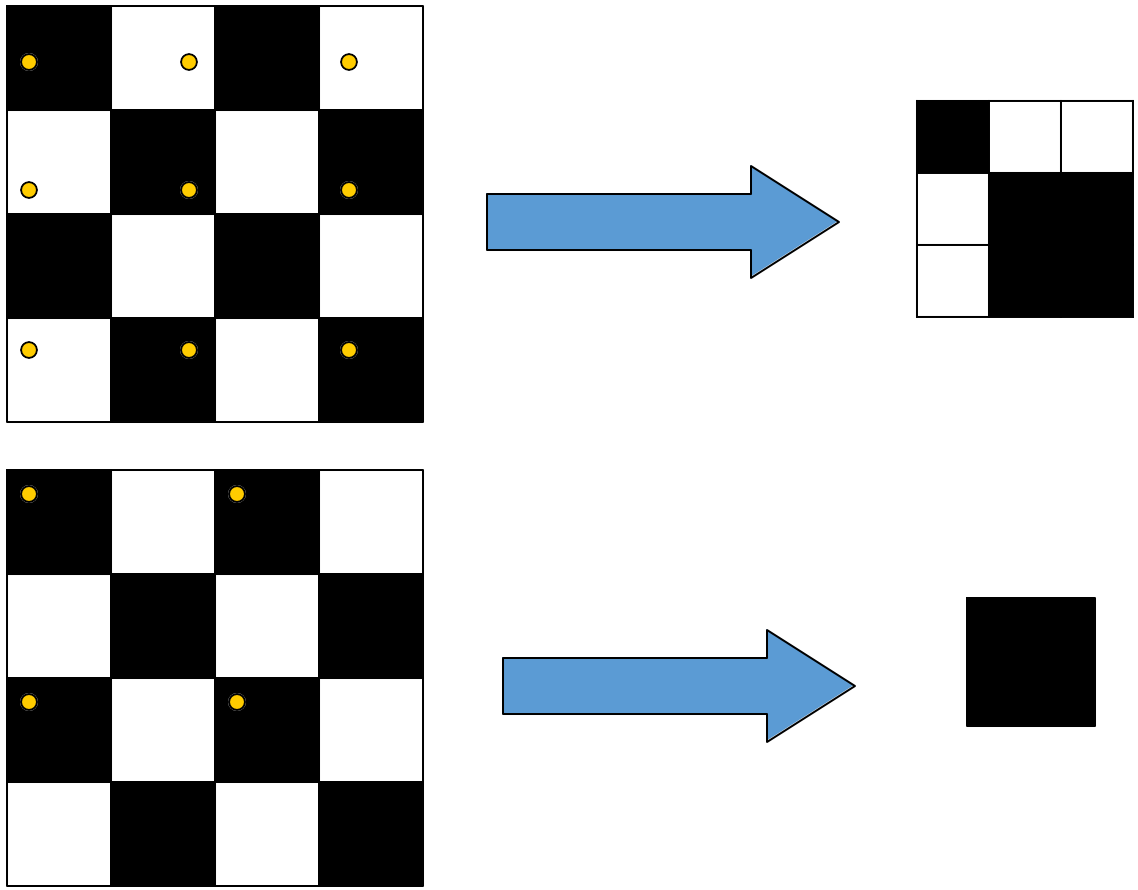
Sampling an image



Examples of GOOD sampling

Slide: Huang

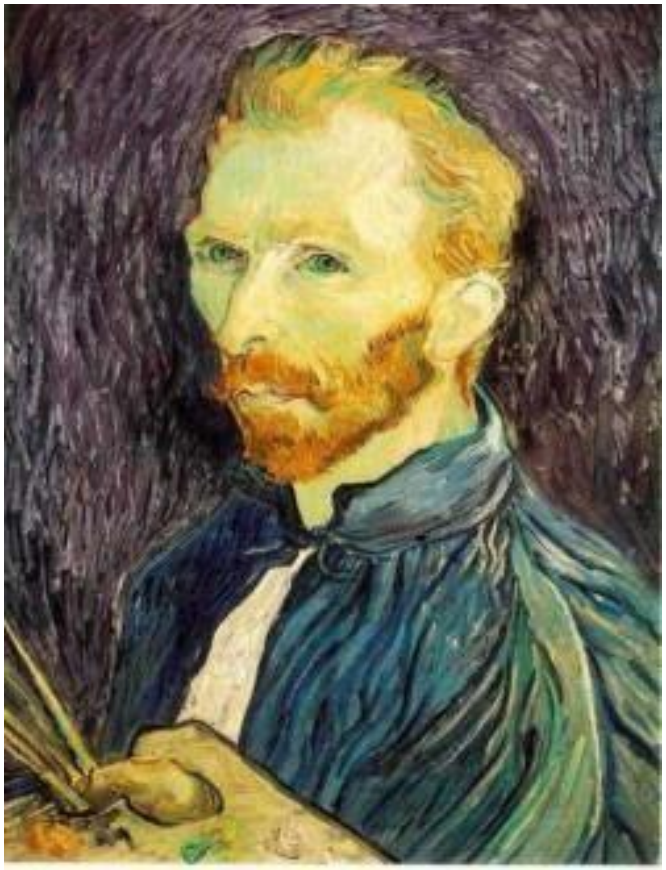
Undersampling



Examples of BAD sampling -> Aliasing

Slide: Huang

Gaussian (low-pass) pre-filtering



Gaussian 1/2

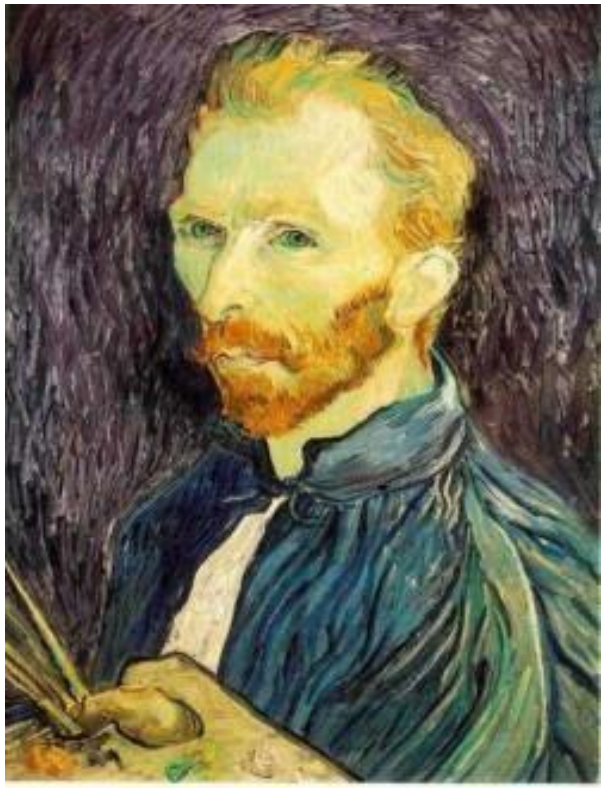
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Subsampling with Gaussian pre-filtering



Gaussian 1/2

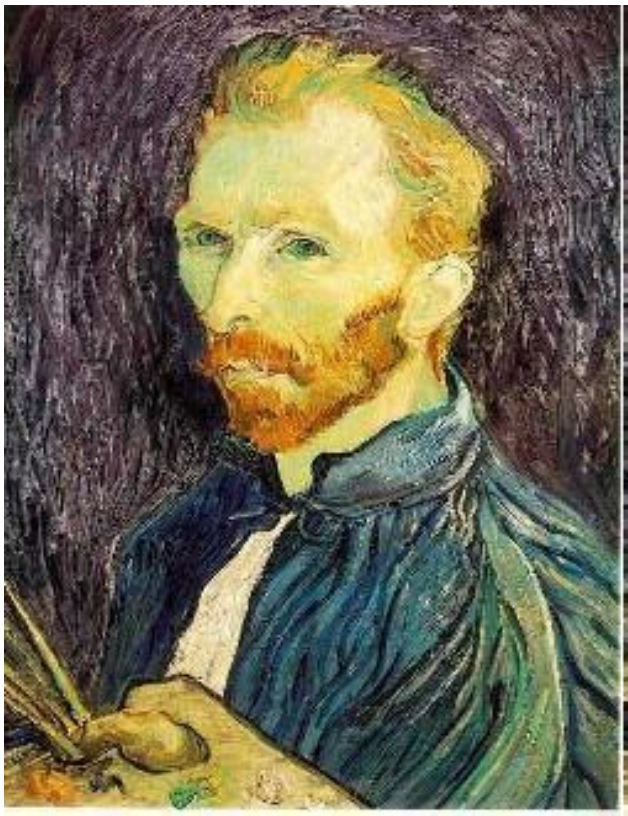
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Compare with...



1/2

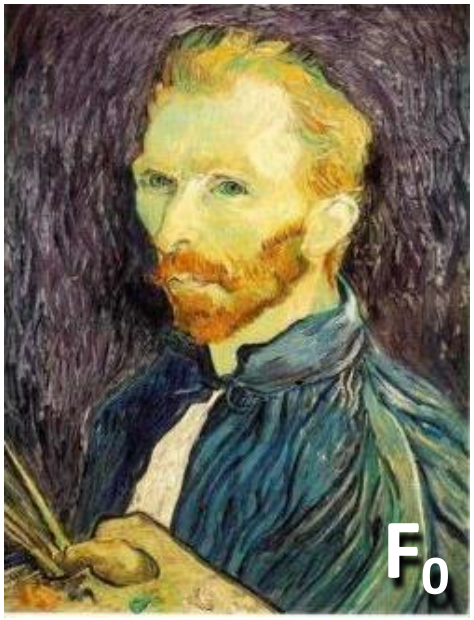
1/4 (2x zoom)

1/8 (4x zoom)

Source: S. Seitz

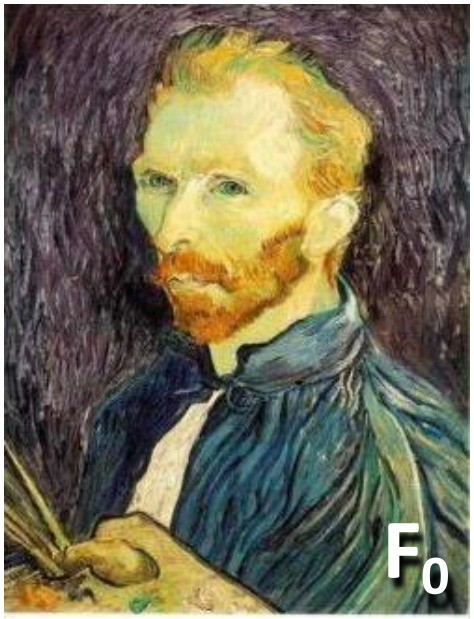
Gaussian pre-filtering

- Solution: filter the image, *then* subsample

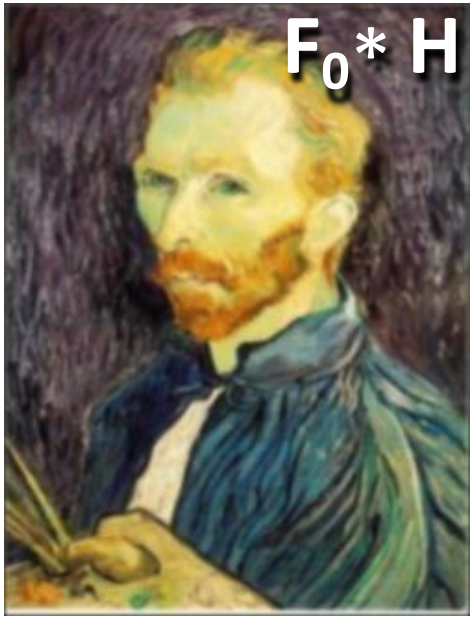


Gaussian pre-filtering

- Solution: filter the image, *then* subsample

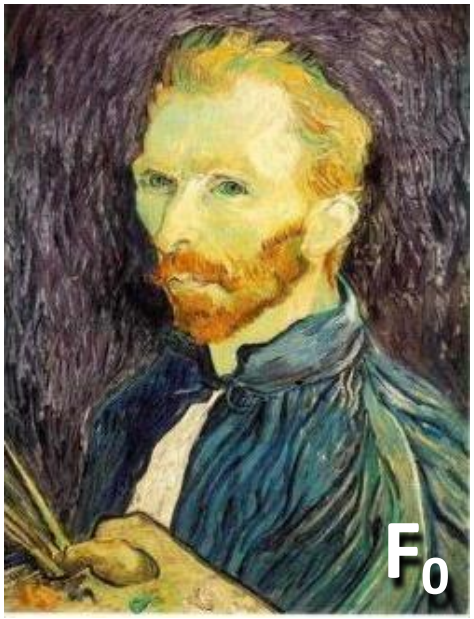


blur



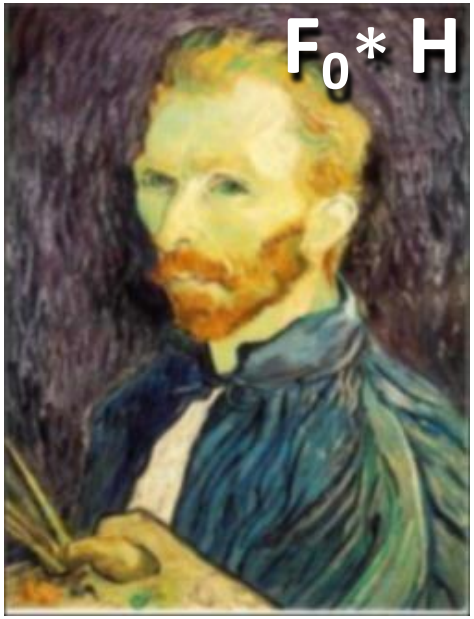
Gaussian pre-filtering

- Solution: filter the image, *then* subsample



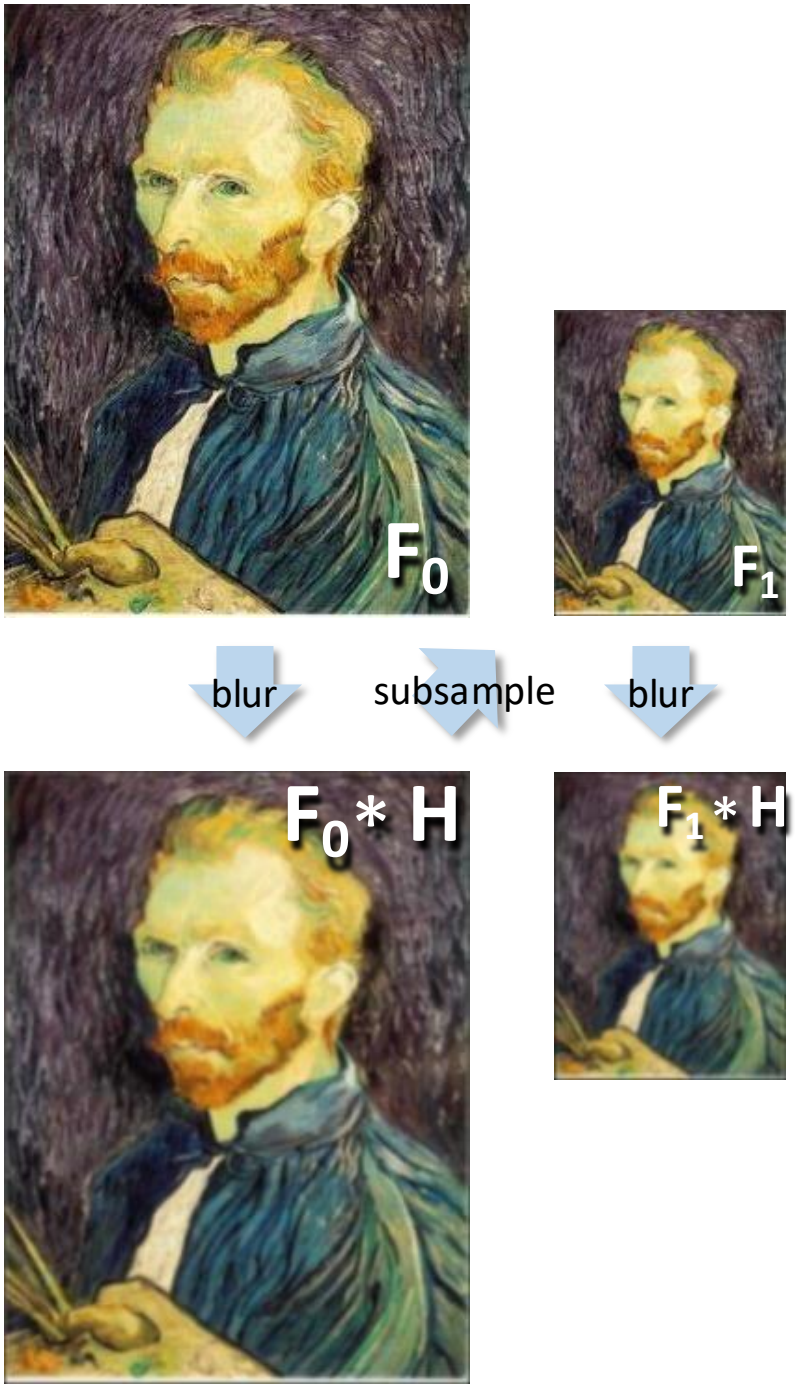
blur

subsample



Gaussian pre-filtering

- Solution: filter the image, *then* subsample

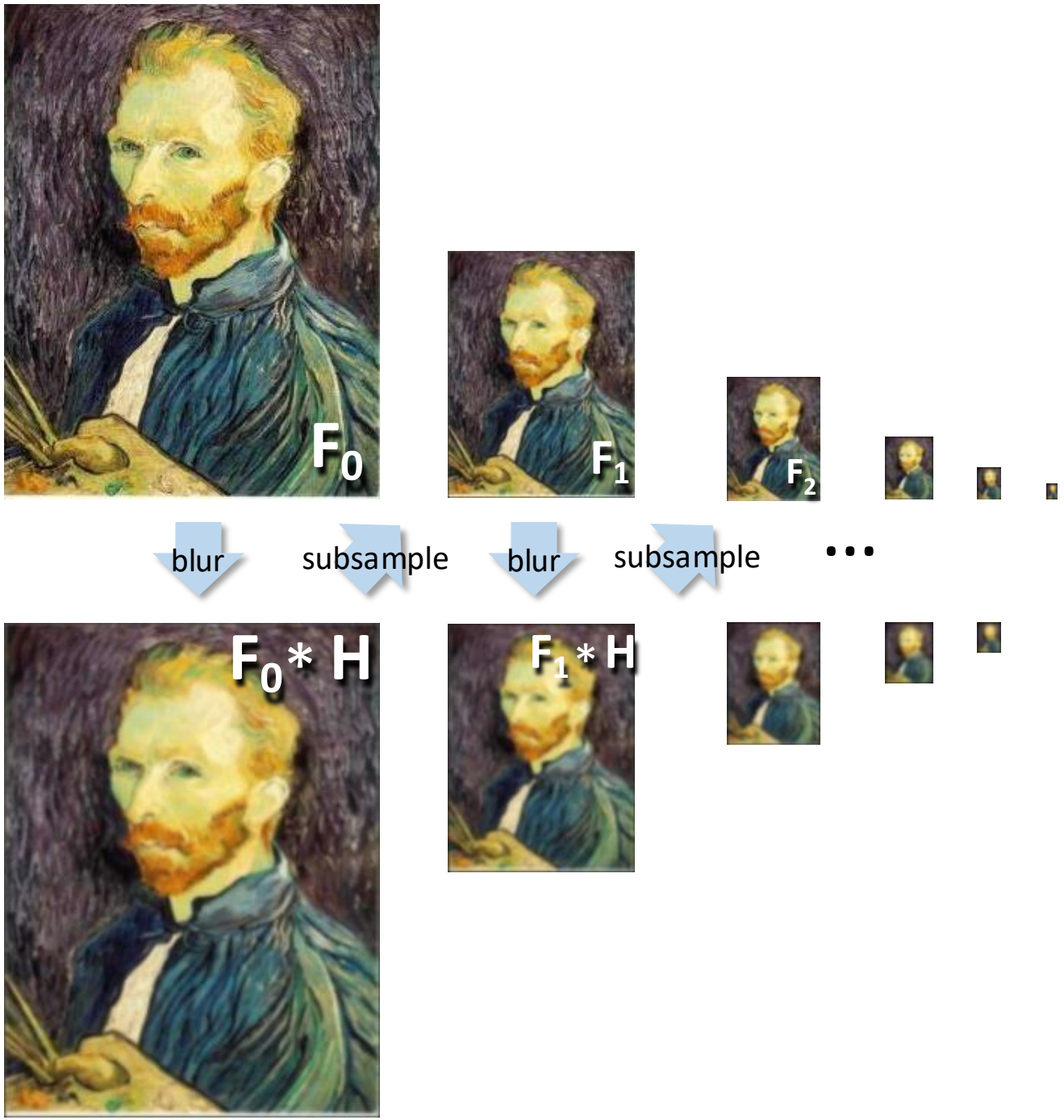


Gaussian pre-filtering

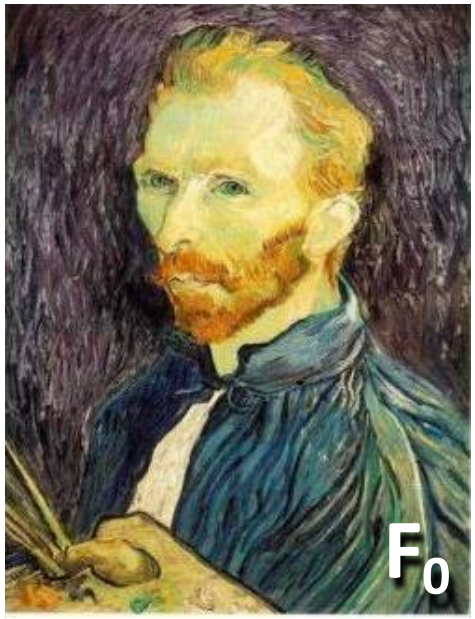
- Solution: filter the image, *then* subsample

Gaussian pre-filtering

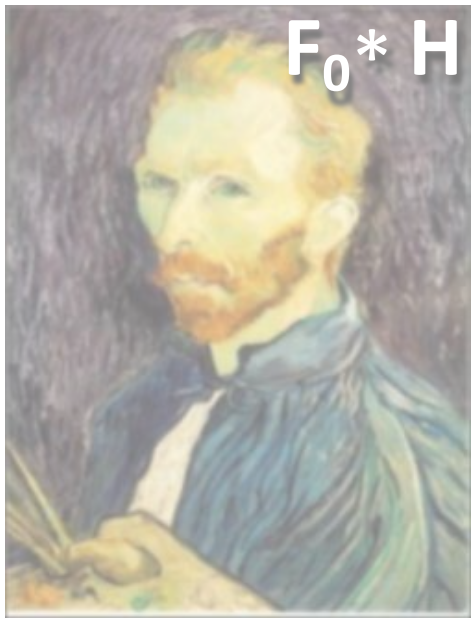
- Solution: filter the image, *then* subsample



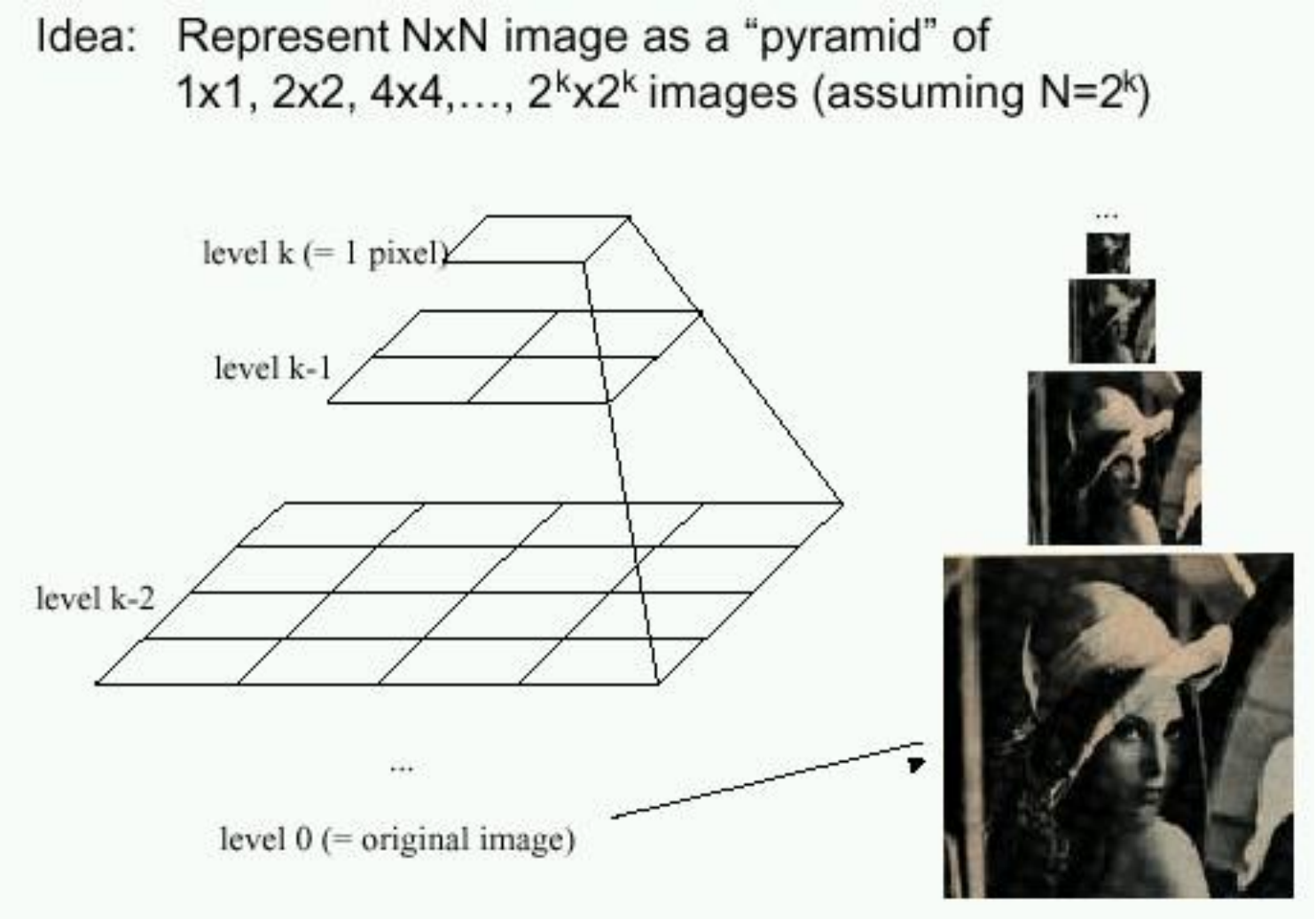
Gaussian pyramid



blur subsample blur subsample • • •



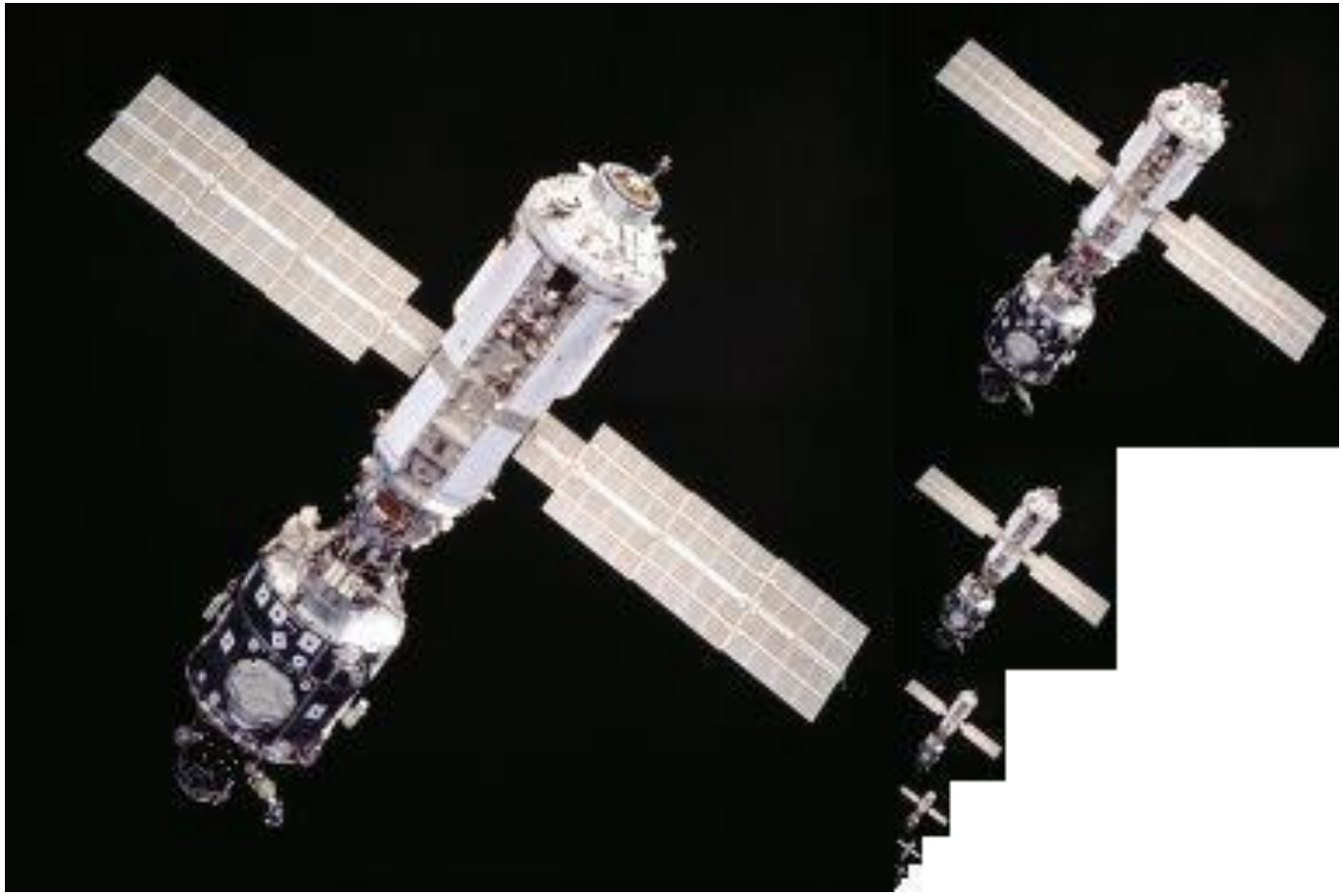
Gaussian pyramids [Burt and Adelson, 1983]



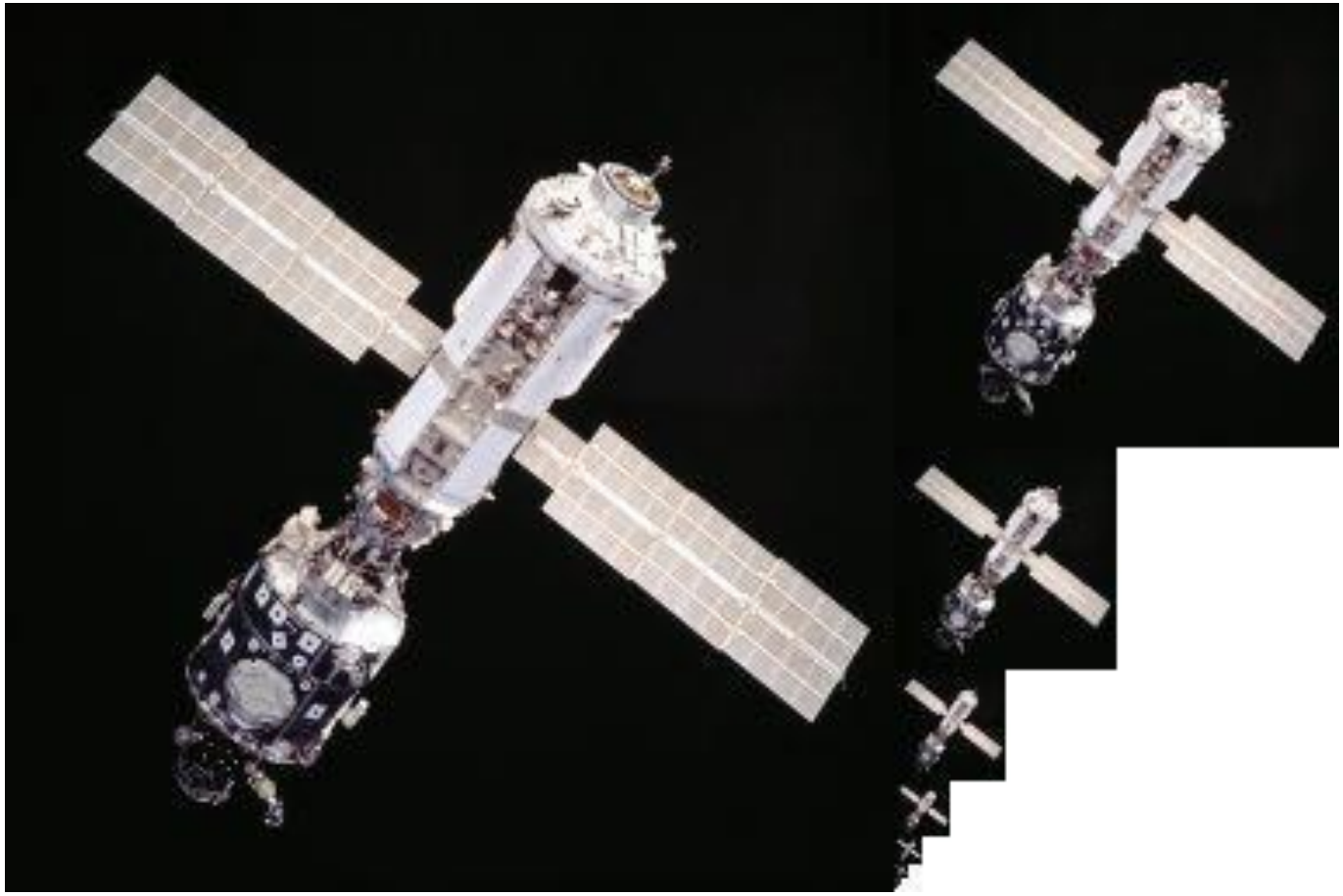
Gaussian Pyramids have all sorts of applications in computer vision

Source: S. Seitz

Gaussian Pyramid



Gaussian Pyramid



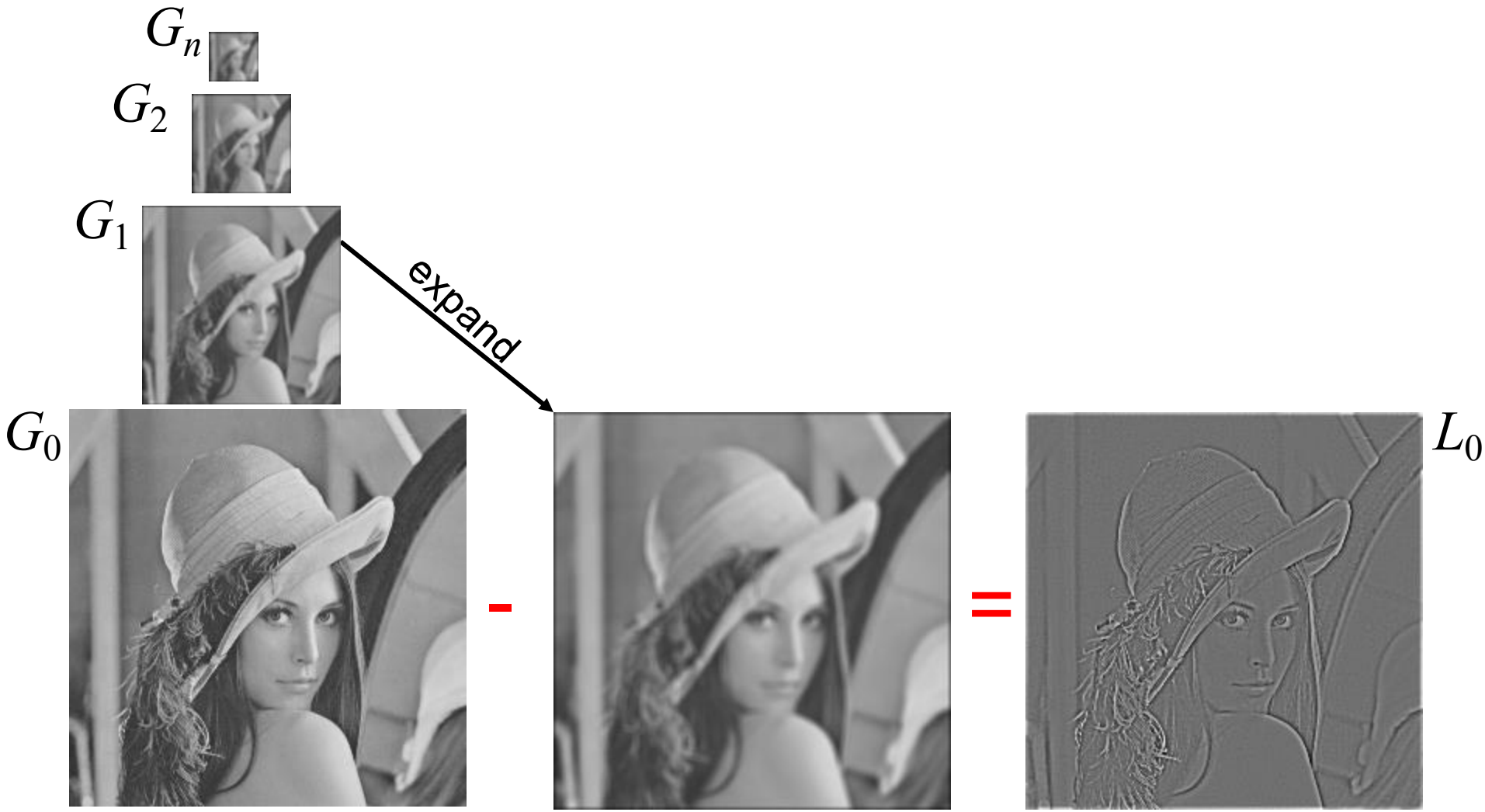
- How much space does a Gaussian pyramid take compared to the original image?

The Laplacian Pyramid

Gaussian Pyramid

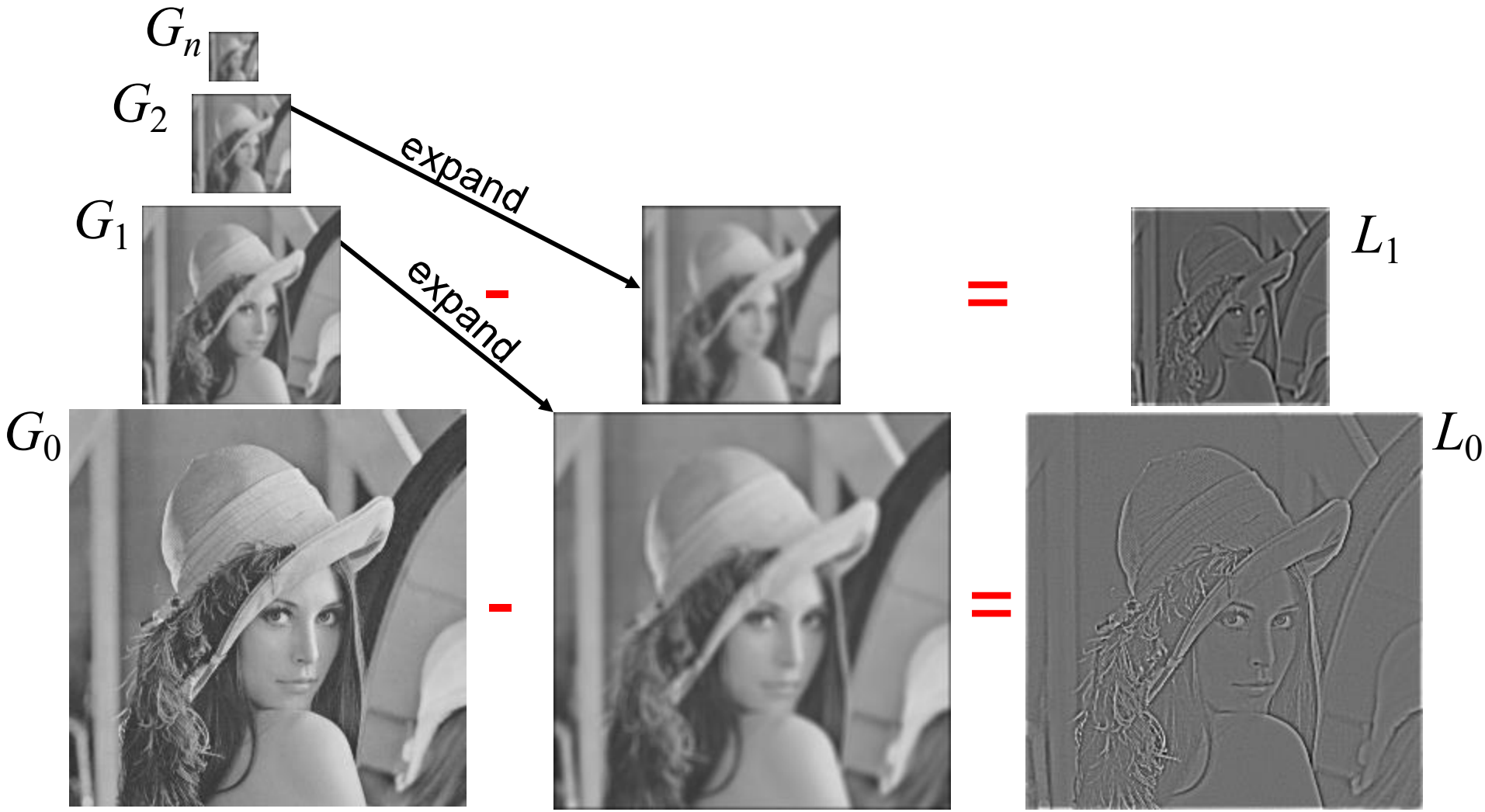
The Laplacian Pyramid

Gaussian Pyramid



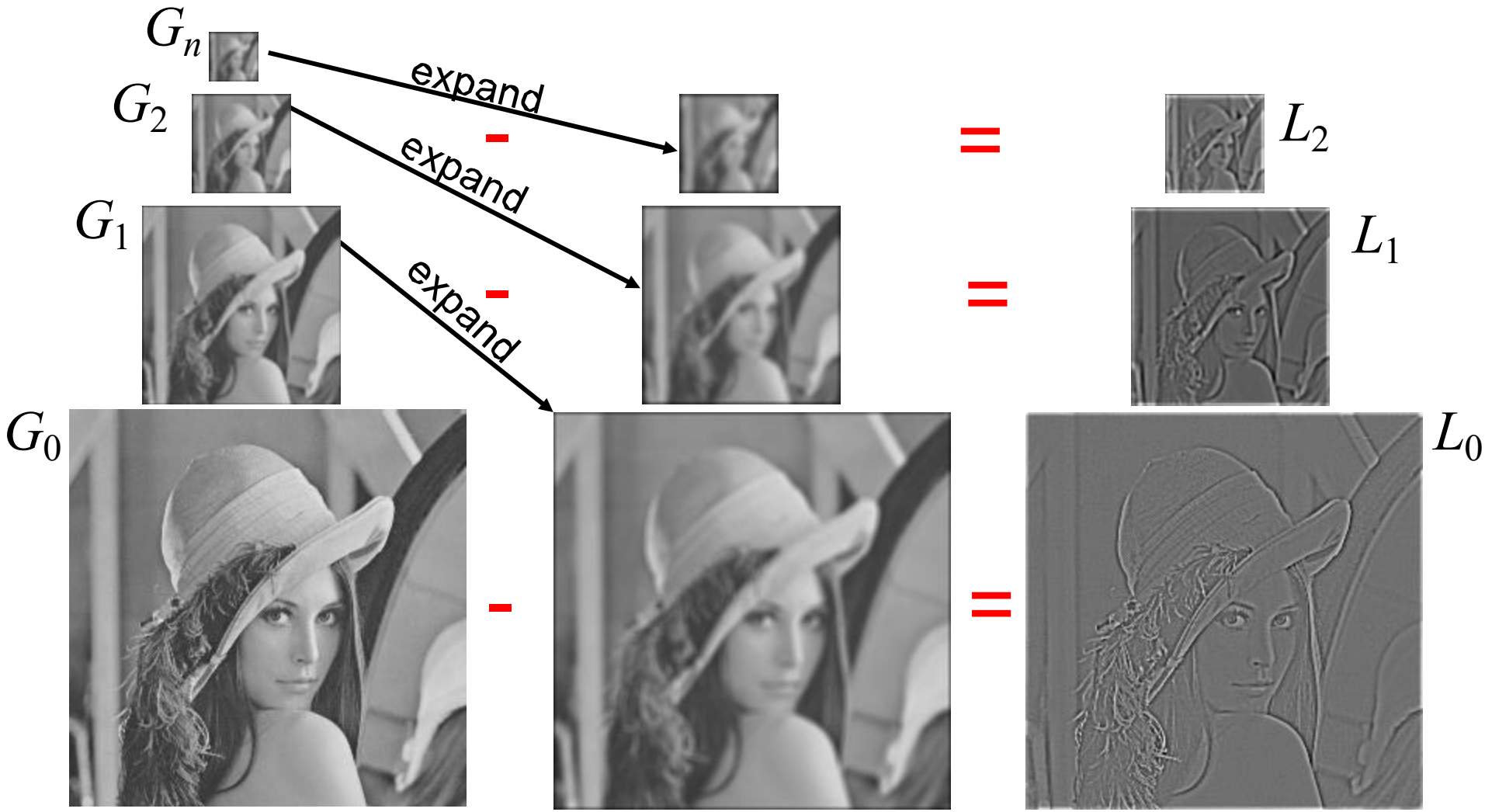
The Laplacian Pyramid

Gaussian Pyramid

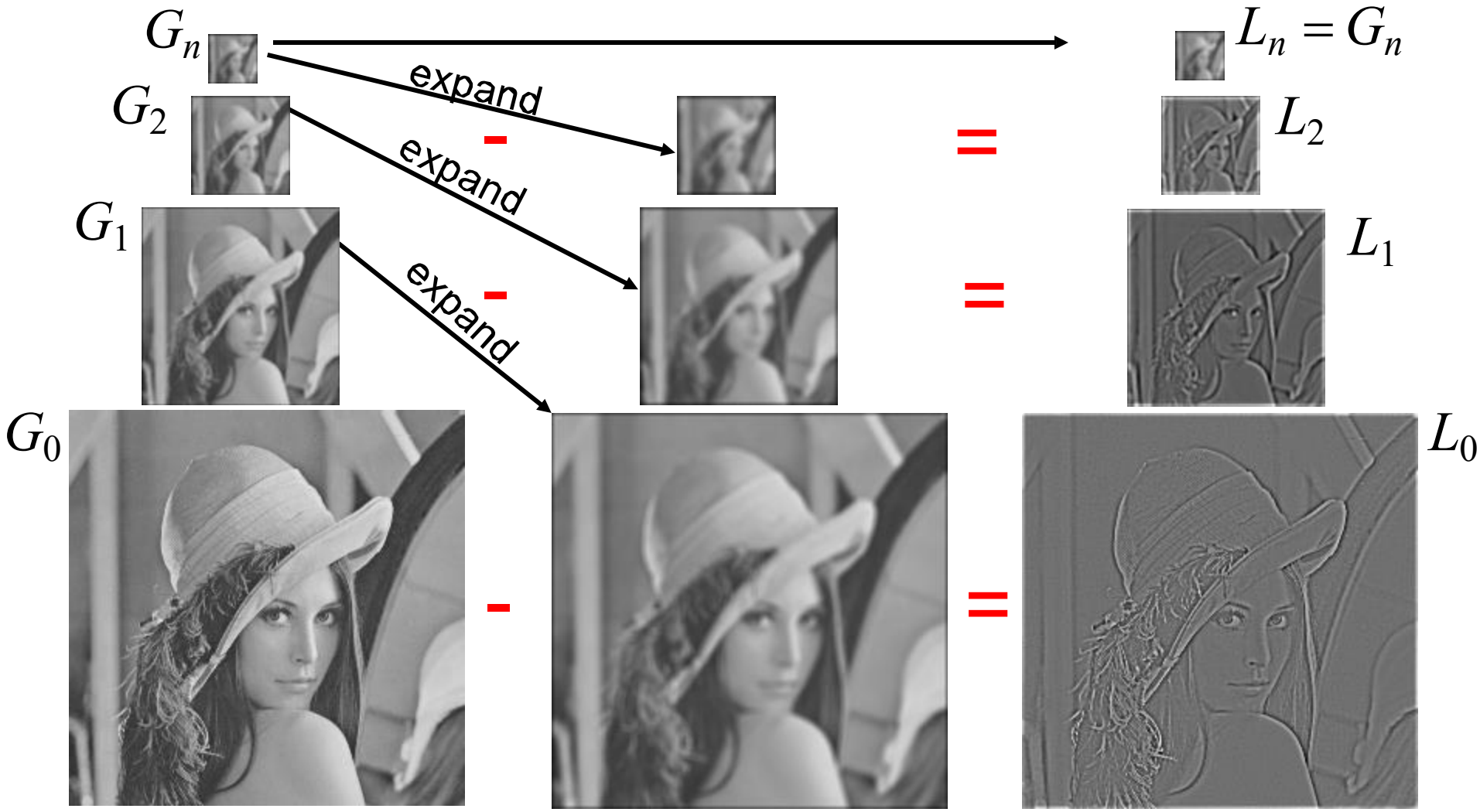


The Laplacian Pyramid

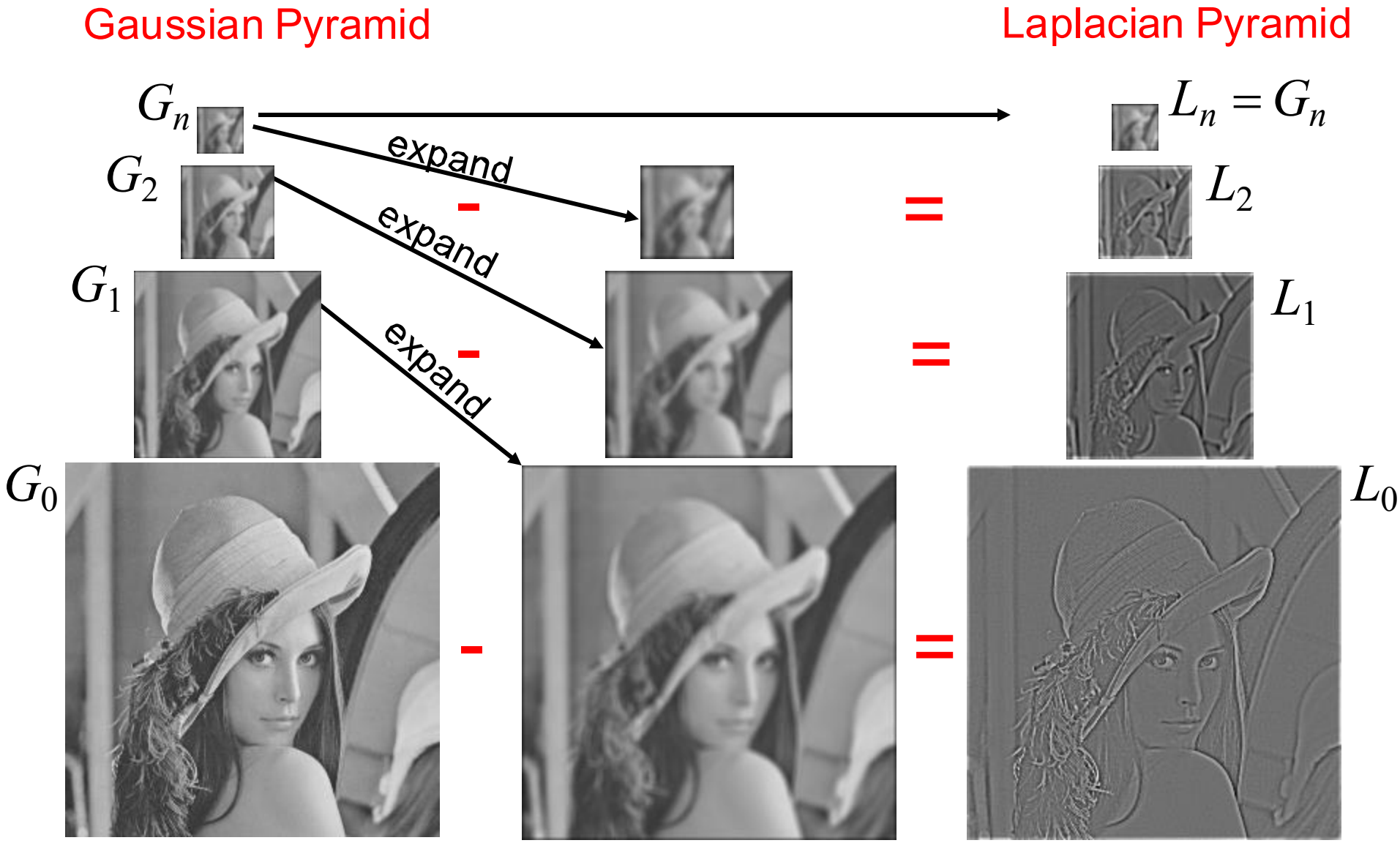
Gaussian Pyramid



The Laplacian Pyramid



The Laplacian Pyramid



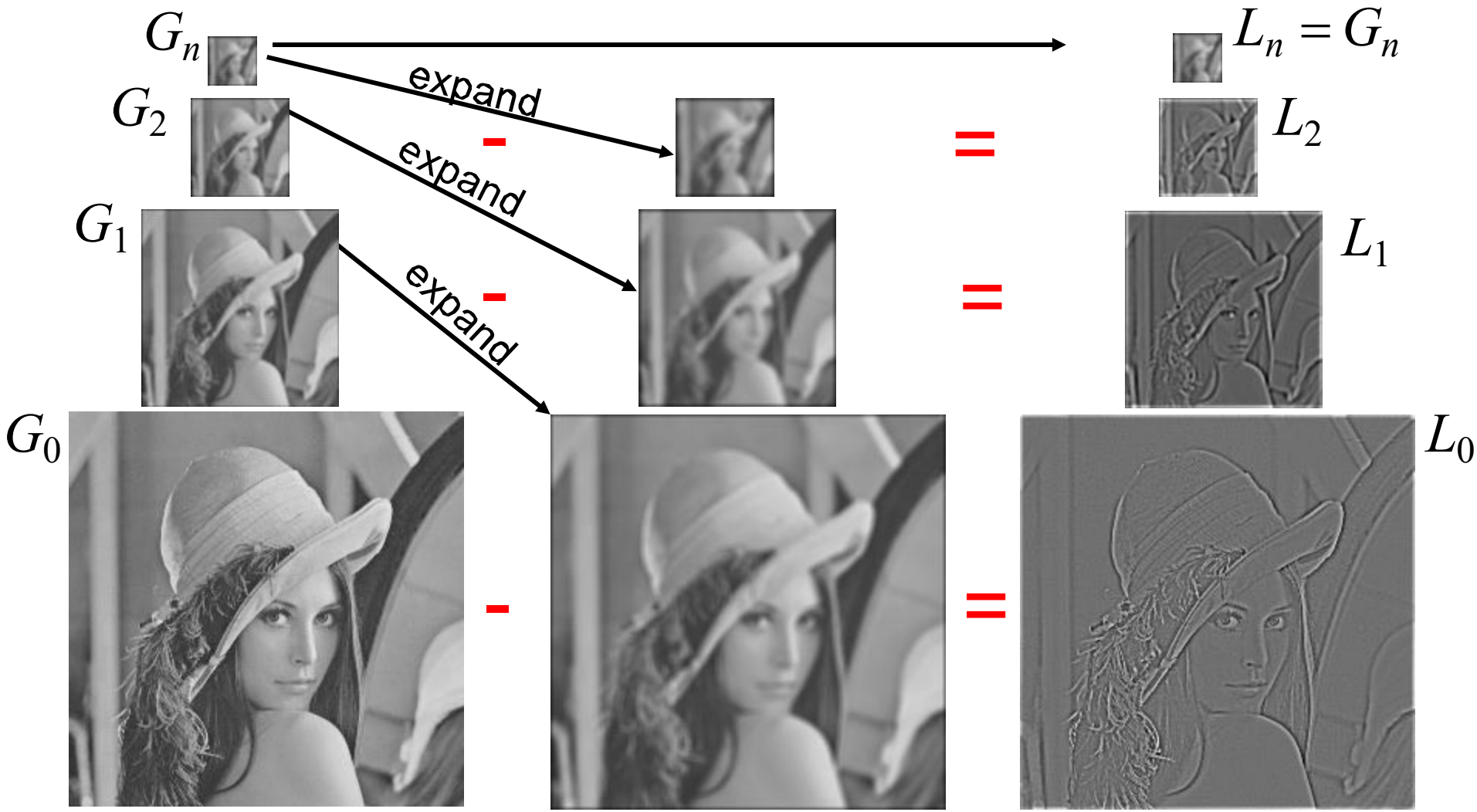
The Laplacian Pyramid

$$L_i = G_i - \text{expand}(G_{i+1})$$

$$G_i = L_i + \text{expand}(G_{i+1})$$

Gaussian Pyramid

Laplacian Pyramid



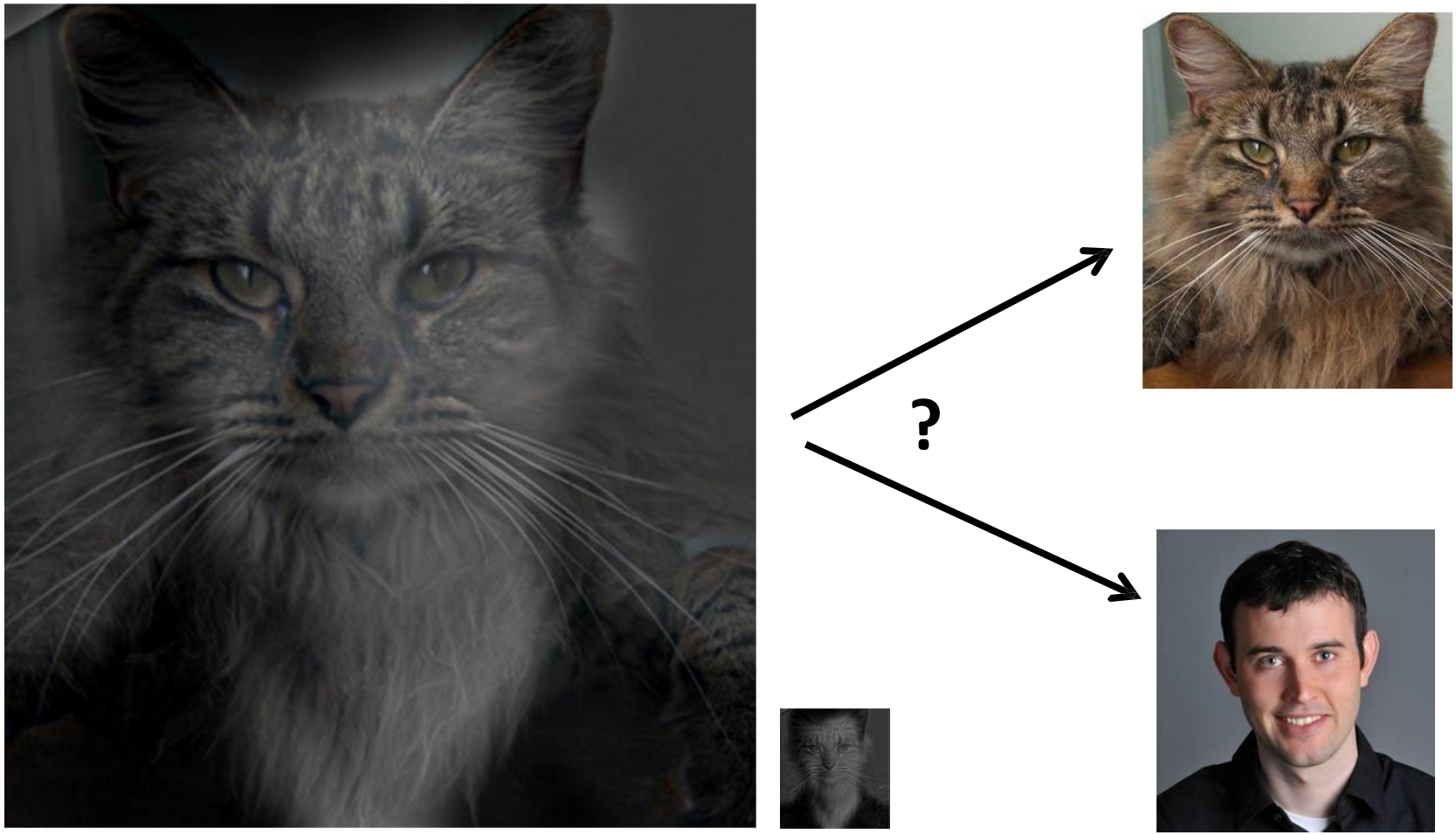
Why does a lower resolution image still make sense to us?

What do we lose?

Slide credit: Derek Hoiem

Image: http://www.flickr.com/photos/igorms/136916757/

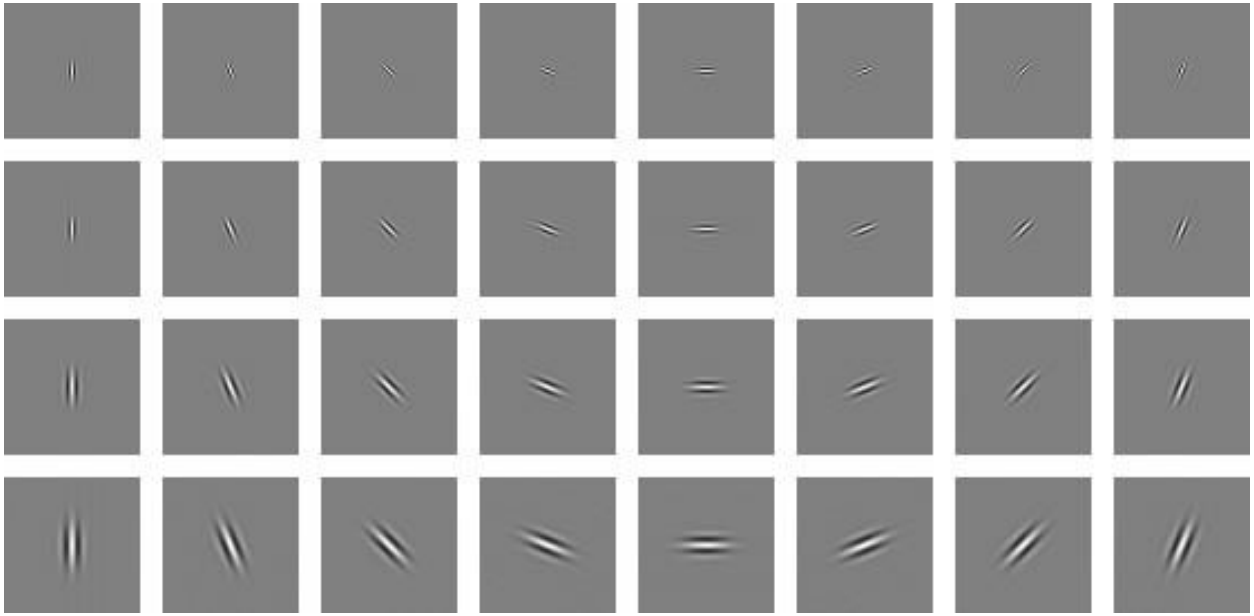
Why do we get different, distance-dependent interpretations of hybrid images?



Slide credit: Derek Hoiem

Clues from Human Perception

- Early processing in humans filters for various orientations and scales of frequency

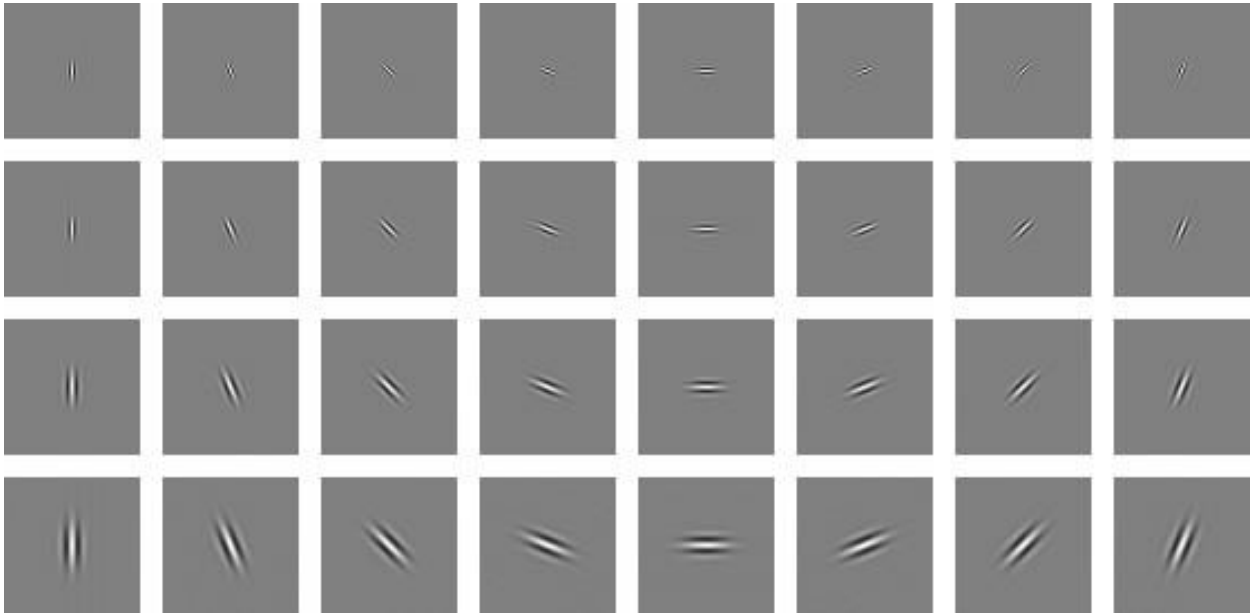


Early Visual Processing: Multi-scale edge and blob filters

Slide: Huang

Clues from Human Perception

- Early processing in humans filters for various orientations and scales of frequency
- Perceptual cues in the mid-high frequencies dominate perception

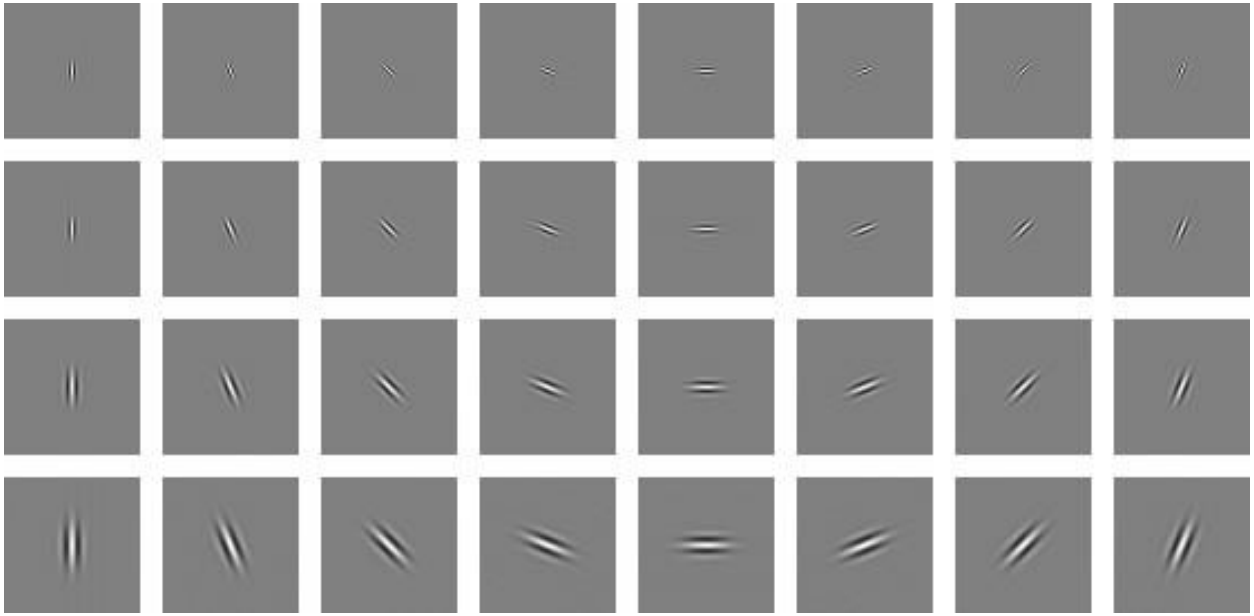


Early Visual Processing: Multi-scale edge and blob filters

Slide: Huang

Clues from Human Perception

- Early processing in humans filters for various orientations and scales of frequency
- Perceptual cues in the mid-high frequencies dominate perception
- When we see an image from far away, we are effectively subsampling it



Early Visual Processing: Multi-scale edge and blob filters

Slide: Huang

Upsampling

- This image is too small for this screen: 
- How can we make it 10 times as big?

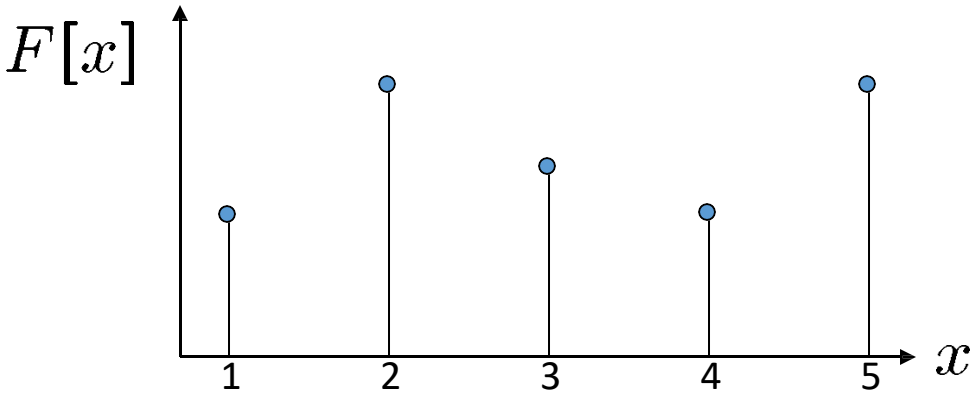
Slide: Huang

Upsampling

- This image is too small for this screen: 
- How can we make it 10 times as big?
- Simplest approach:
  repeat each row
  and column 10 times
- (“Nearest neighbor
  interpolation”)

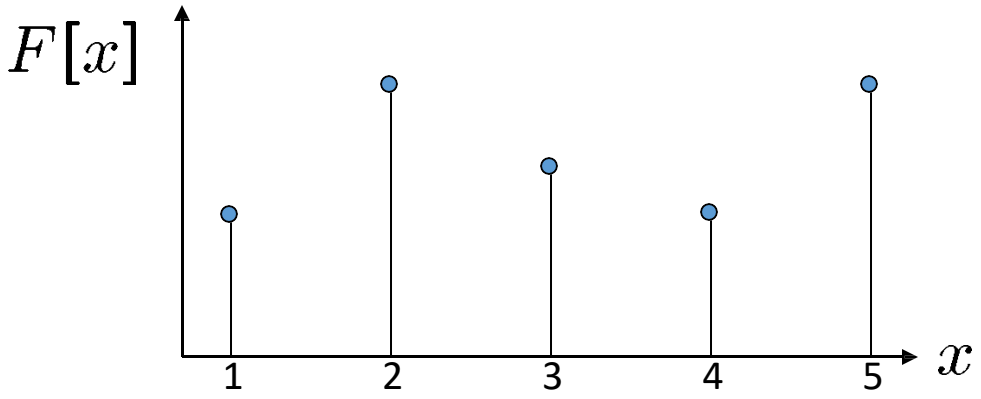
Slide: Huang

Image interpolation



Adapted from: S. Seitz

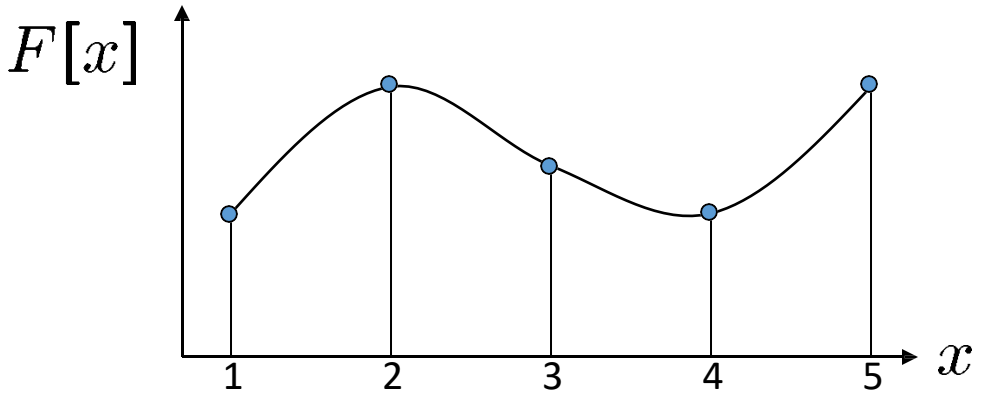
Image interpolation



- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

Adapted from: S. Seitz

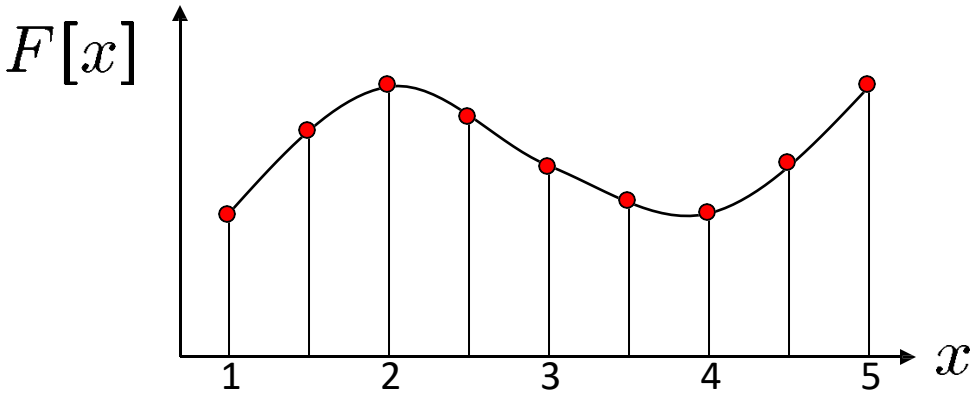
Image interpolation



- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

Adapted from: S. Seitz

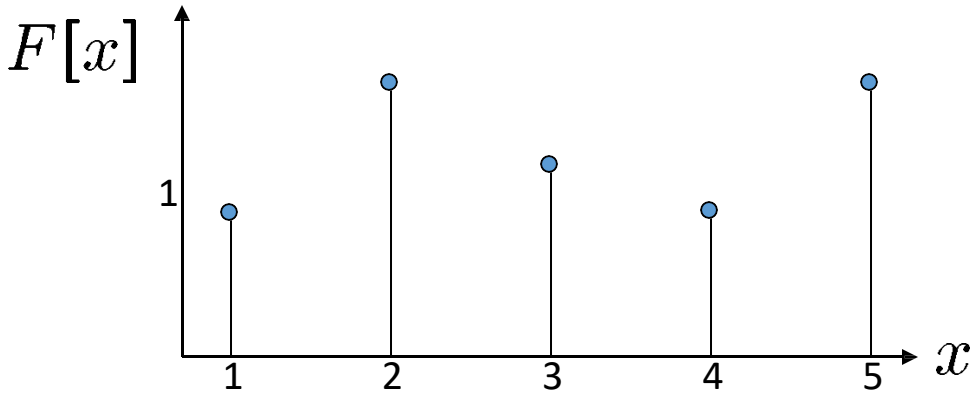
Image interpolation



- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

Adapted from: S. Seitz

Image interpolation

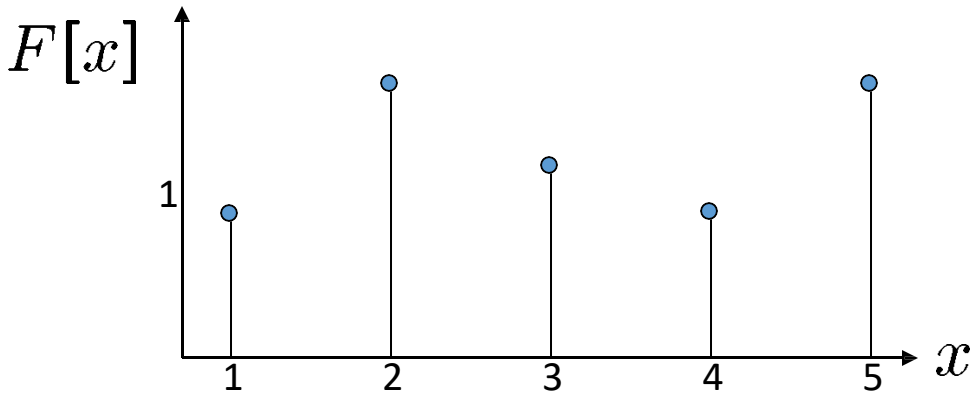


d = 1 in this example

- What if we don't know f ?

Adapted from: S. Seitz

Image interpolation

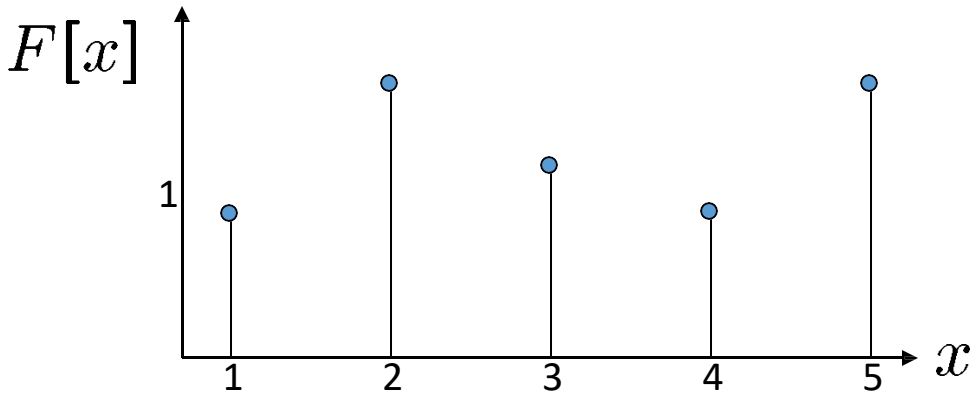


d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering

Adapted from: S. Seitz

Image interpolation



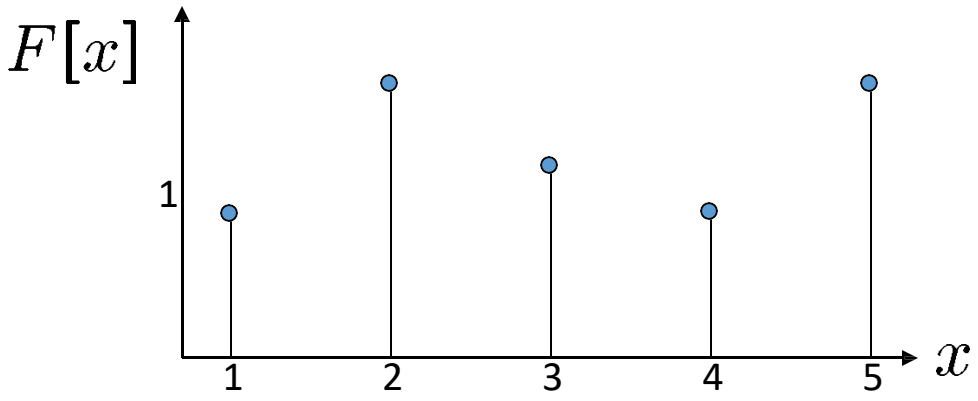
d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

Adapted from: S. Seitz

Image interpolation



d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

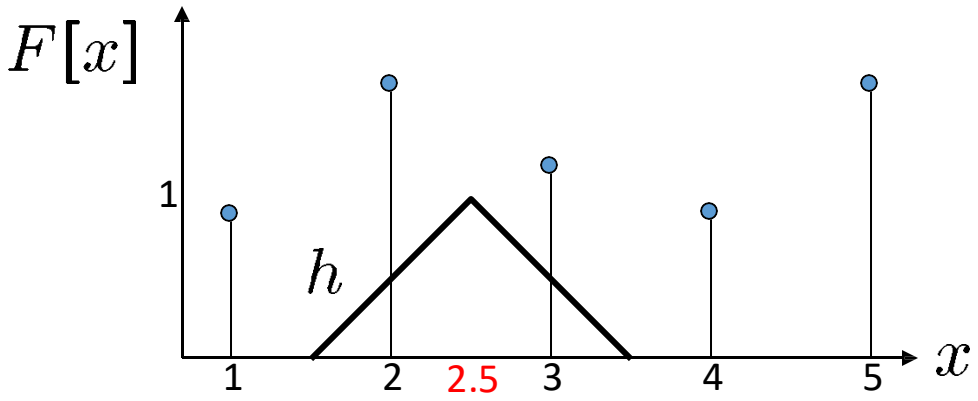
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

  - Reconstruct by convolution with a *reconstruction filter, h*

$$\tilde{f} = h * f_F$$

Adapted from: S. Seitz

Image interpolation



d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

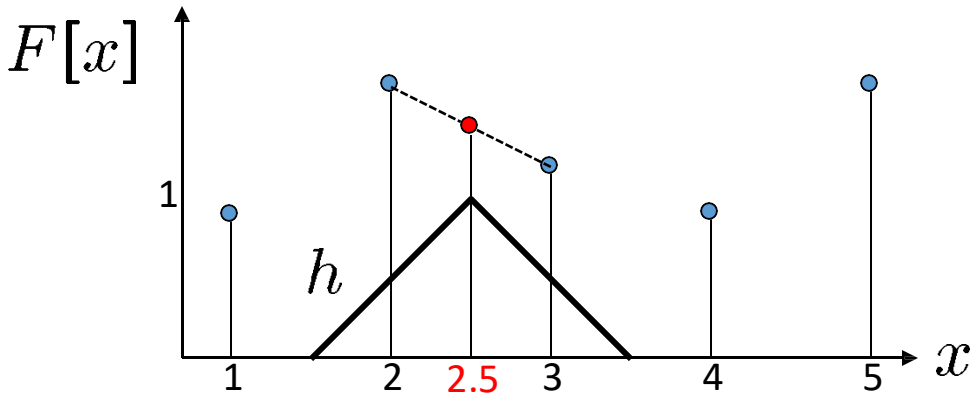
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

  - Reconstruct by convolution with a *reconstruction filter, h*

$$\tilde{f} = h * f_F$$

Adapted from: S. Seitz

Image interpolation



d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

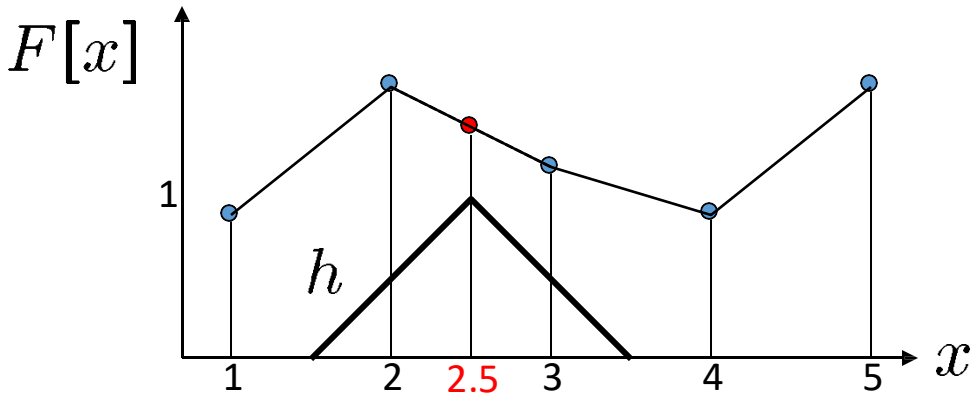
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

  - Reconstruct by convolution with a *reconstruction filter, h*

$$\tilde{f} = h * f_F$$

Adapted from: S. Seitz

Image interpolation



d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

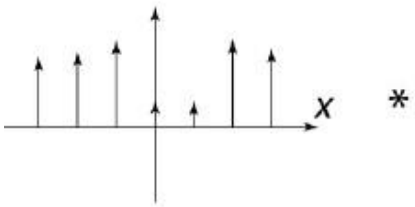
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

  - Reconstruct by convolution with a *reconstruction filter or interpolation kernel, h*

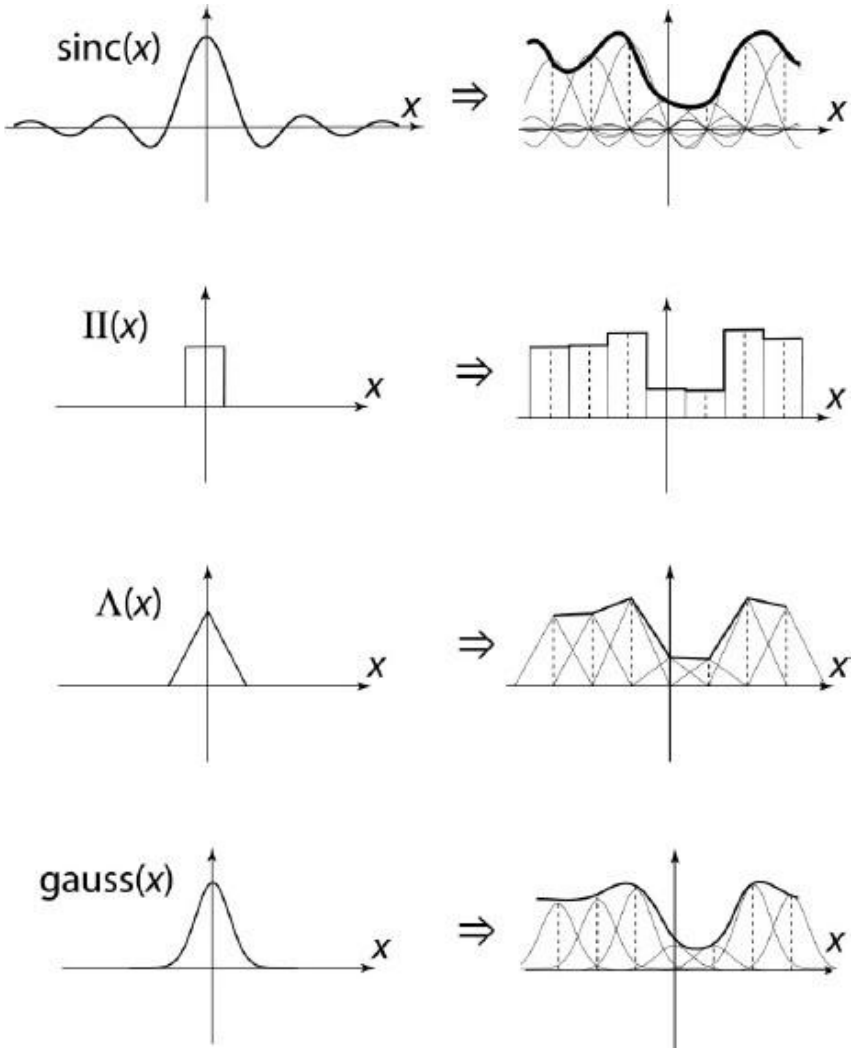
$$\tilde{f} = h * f_F$$

Adapted from: S. Seitz

Image interpolation



*



"Ideal" reconstruction

Nearest-neighbor interpolation

Linear interpolation

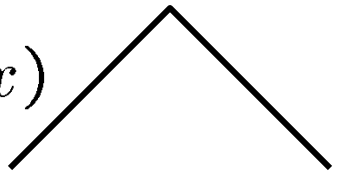
Gaussian reconstruction

Source: B. Curless

Reconstruction filters

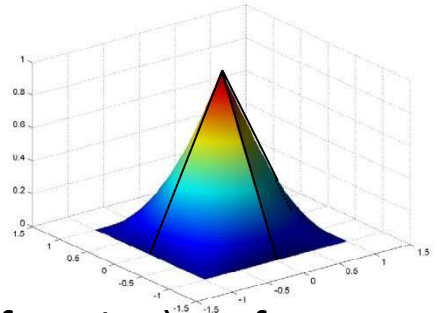
- What does the 2D version of this hat function look like?

$h(x)$



performs
linear interpolation

$h(x, y)$



(tent function) performs
bilinear interpolation

Slide: Huang

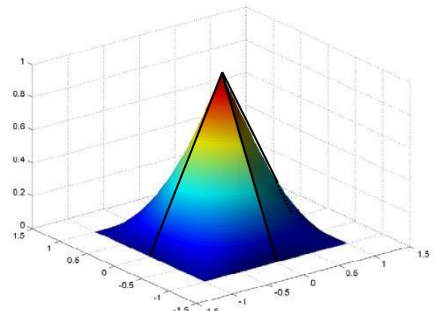
Reconstruction filters

- What does the 2D version of this hat function look like?

$h(x)$

performs
linear interpolation

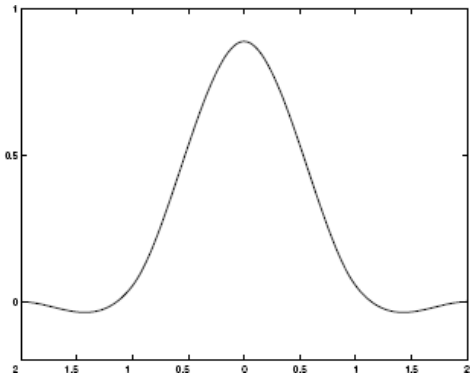
$h(x, y)$



(tent function) performs
bilinear interpolation

Better filters give better resampled images

- **Bicubic** is common choice



Cubic reconstruction filter

$$r(x) = \frac{1}{6}\begin{cases} (12-9B-6C)|x|^3 + (-18+12B+6C)|x|^2 + (6-2B) & |x|<1 \\ ((-B-6C)|x|^3 + (6B+30C)|x|^2 + (-12B-48C)|x| + (8B+24C) & 1\le|x|<2 \\ 0 & \textit{otherwise} \end{cases}$$

Slide: Huang

Image interpolation

Original image:  x 10

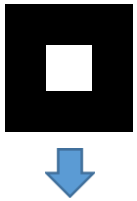
Nearest-neighbor interpolation

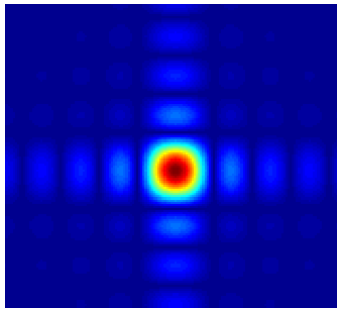
Bilinear interpolation

Bicubic interpolation

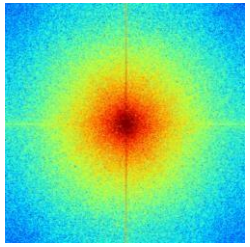
Slide: Huang

Things to Remember

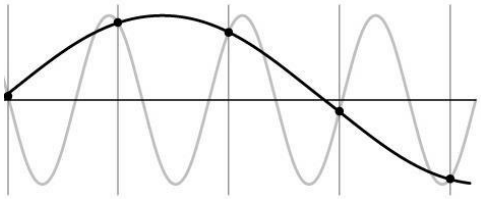




- Sometimes it makes sense to think of images and filtering in the frequency domain
  - Fourier analysis



- Can be faster to filter using FFT for large images (N logN vs. N^2 for auto-correlation)



- Remember to low-pass before sampling

Acknowledgements

- Thanks to the following researchers for making their teaching/research material online
  - Forsyth
  - Steve Seitz
  - Noah Snavely
  - J.B. Huang
  - Derek Hoiem
  - D. Lowe
  - A. Bobick
  - S. Lazebnik
  - K. Grauman
  - R. Zaleski

Thank you

- Next class:
  - Edge Detection