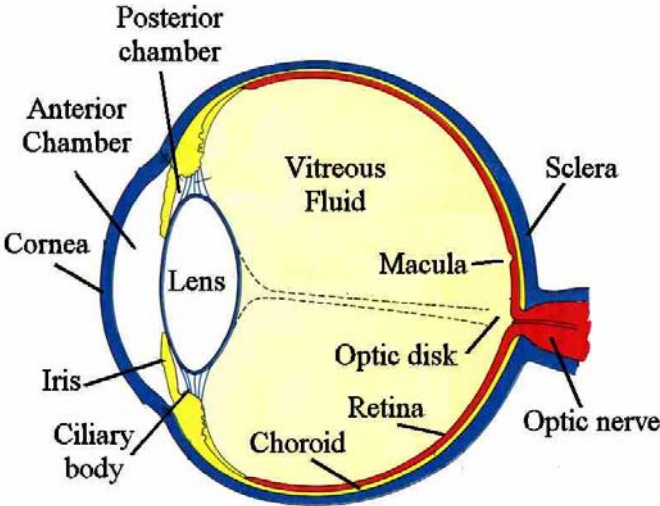
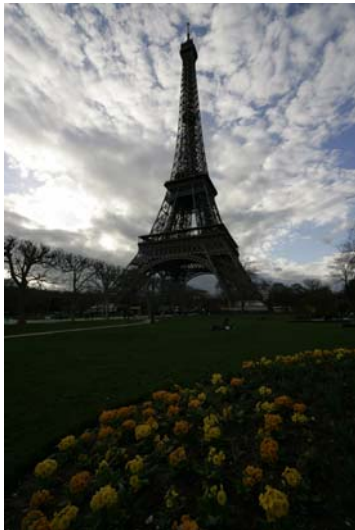


Last Lecture



Kimber, D.C.; C.E. Gray, and C.E. Stackpole. (1966). *Anatomy and Physiology*. MacMillan Co., NY. pg.335.



HDR Video

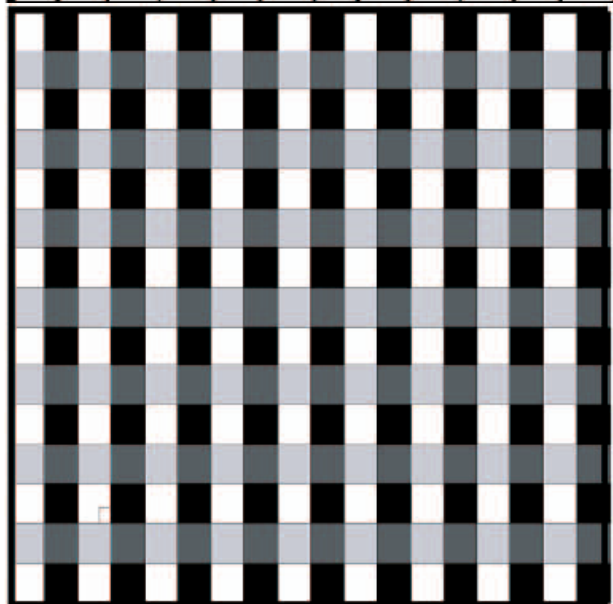
High Dynamic Range Video

Submitted to SIGGRAPH 2003

Paper #125

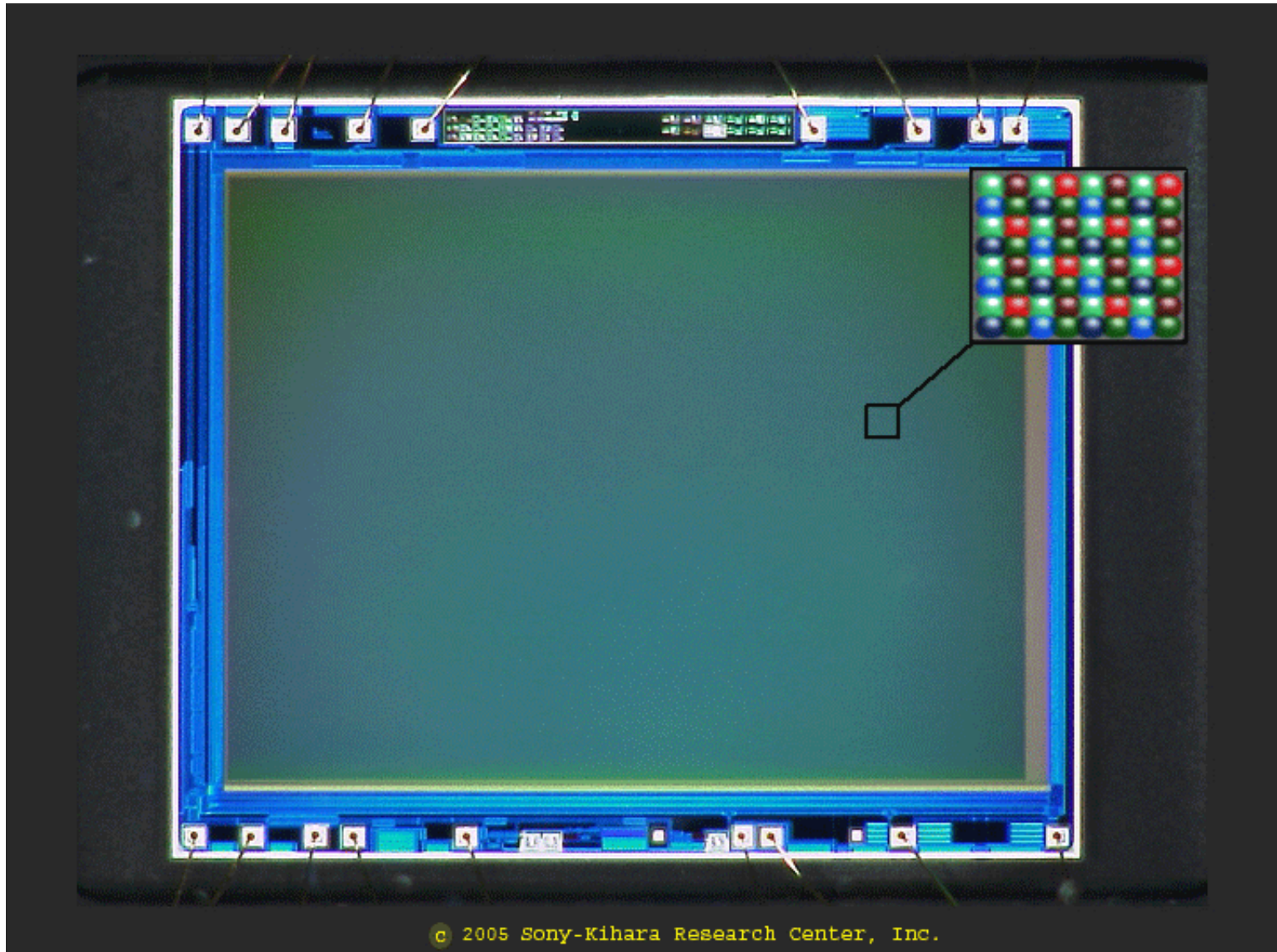
Assorted pixel (Single Exposure HDR)

R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B



R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B	G	B	G	B

Assorted pixel



Assorted pixel

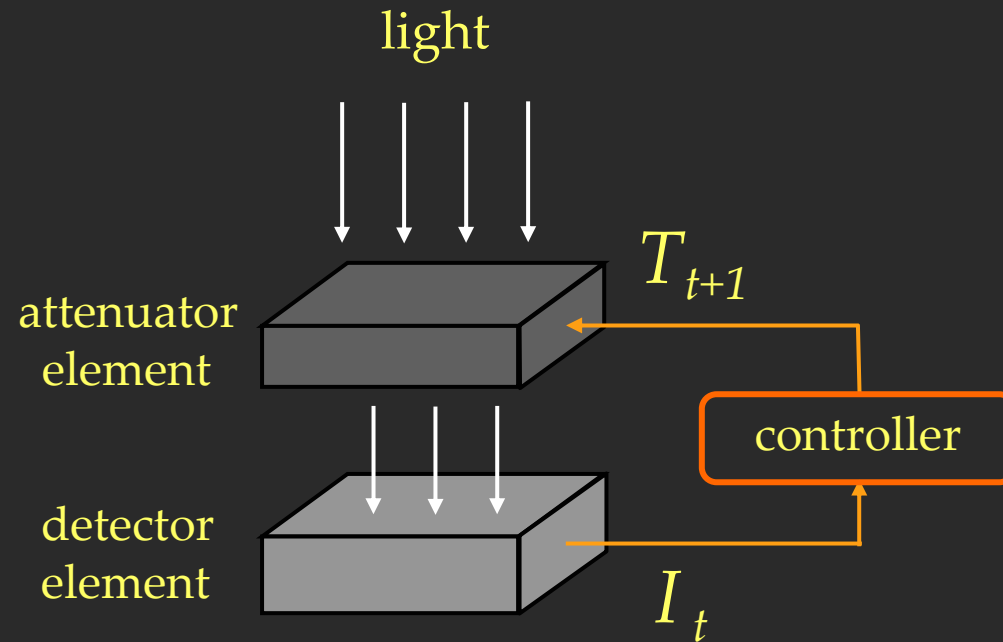
Normal Camera



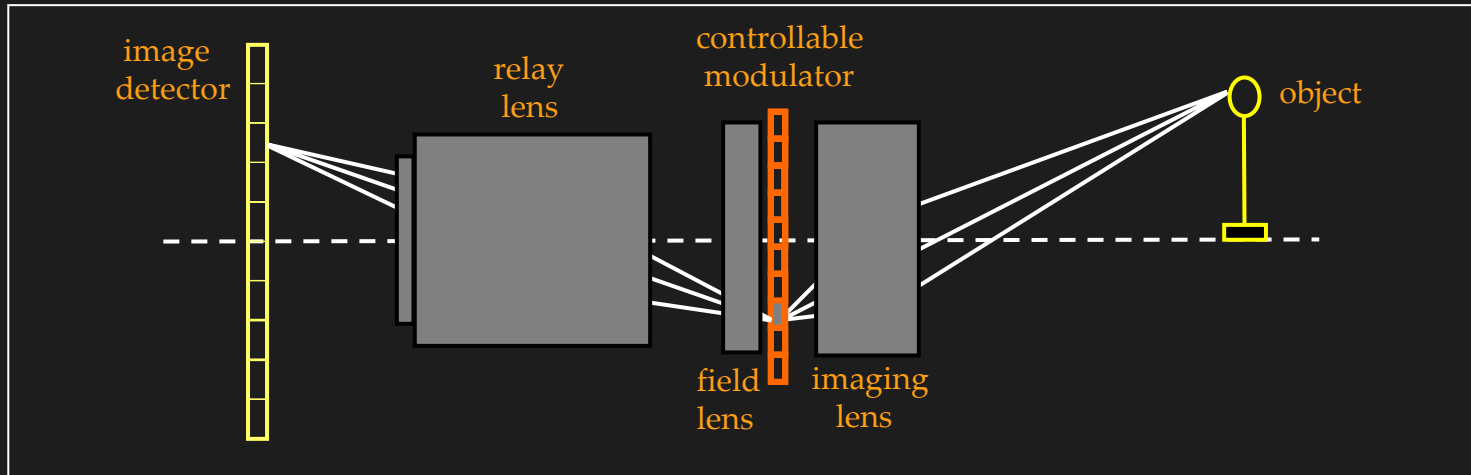
Assorted Pixel Camera



Pixel with Adaptive Exposure Control



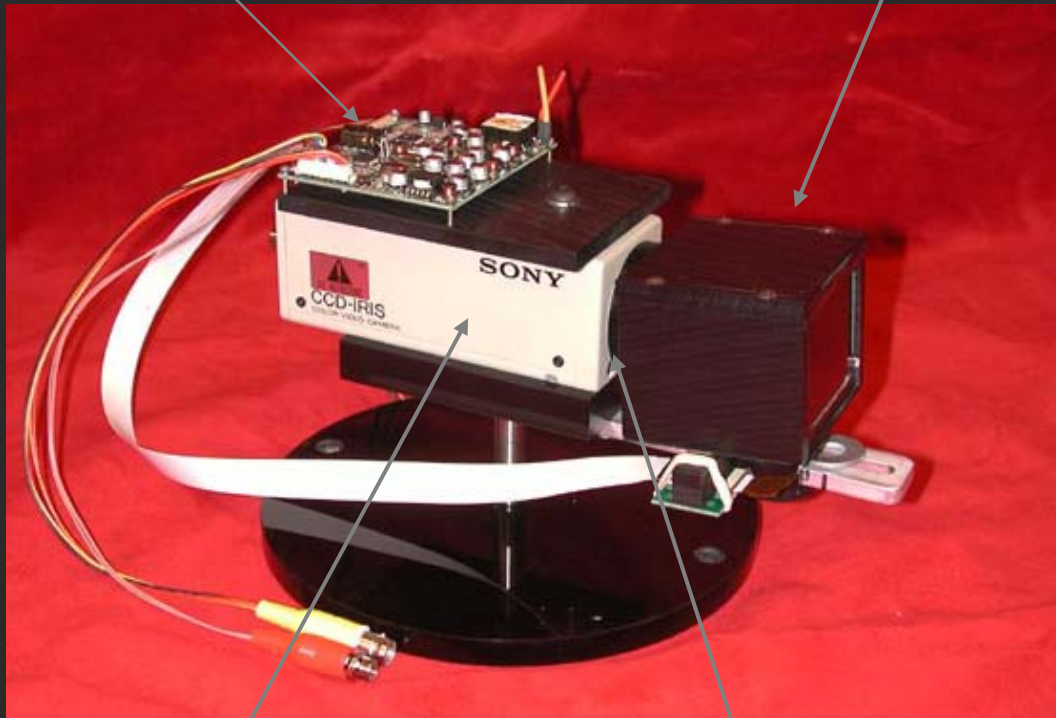
ADR Imaging with Spatial Light Modulator



ADR Camera with LCD Attenuator

LCD Electronics

LCD Attenuator



Video Camera

Imaging Lens

ADAPTIVE DYNAMIC RANGE IMAGING

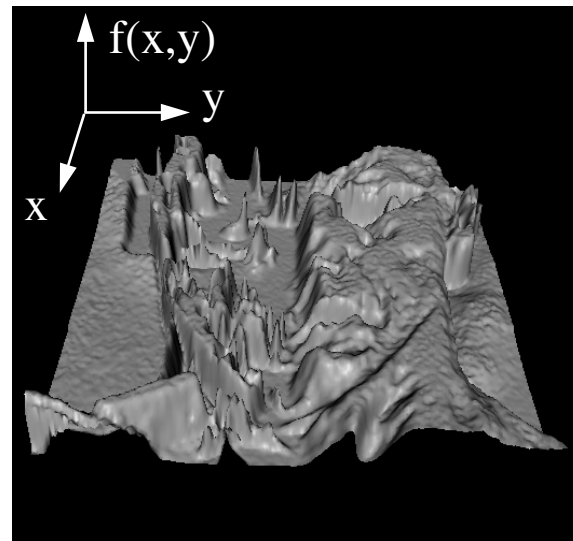
OPTICAL DYNAMIC ATTENUATION

Today

Image Processing: from basic concepts to latest techniques

- Filtering
- Edge detection
- Re-sampling and aliasing
- Image Pyramids (Gaussian and Laplacian)
- Removing handshake blur from a single image

Image as a discrete function



Represented by a matrix

i ↓

→ j

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

What is image filtering?

- Modify the pixels in an image based on some function of a local neighborhood of the pixels.

10	5	3
4	5	1
1	1	7

Local image data

Some function



	7	

Modified image data

Linear functions

- Simplest: linear filtering.
 - Replace each pixel by a linear combination of its neighbors.
- The prescription for the linear combination is called the “convolution kernel”.

10	5	3
4	5	1
1	1	7

Local image data

0	0	0
0	0.5	0
0	1	0.5

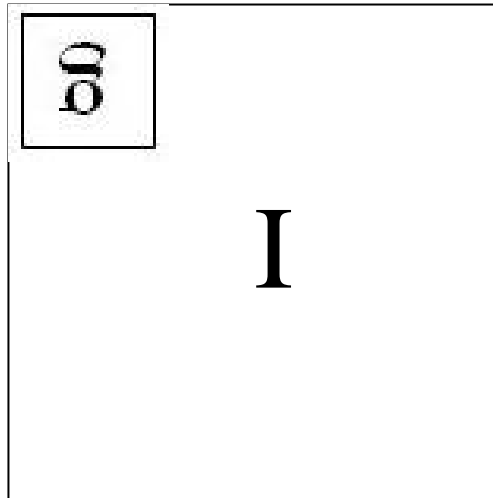
kernel

	7	

Modified image data

Convolution

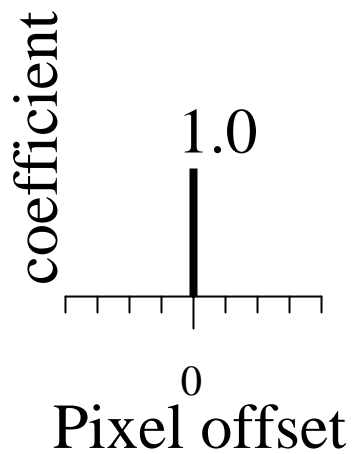
$$f[m, n] = I \otimes g = \sum_{k, l} I[m - k, n - l] g[k, l]$$



Linear filtering (warm-up slide)



original

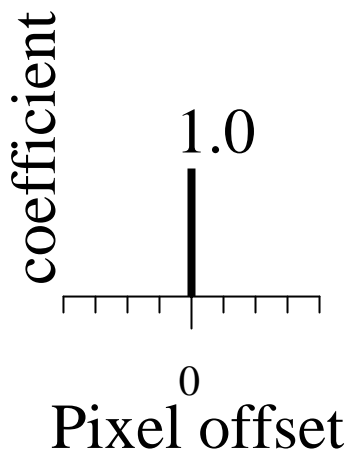


?

Linear filtering (warm-up slide)



original

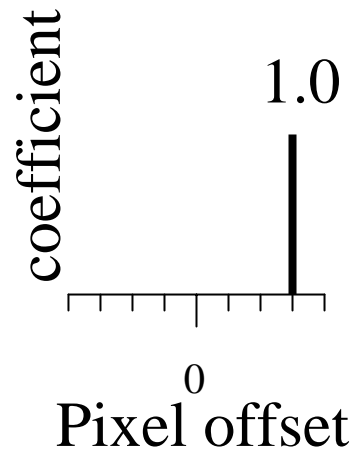


Filtered
(no change)

Linear filtering

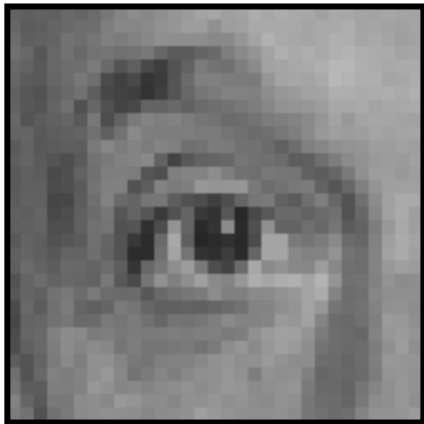


original

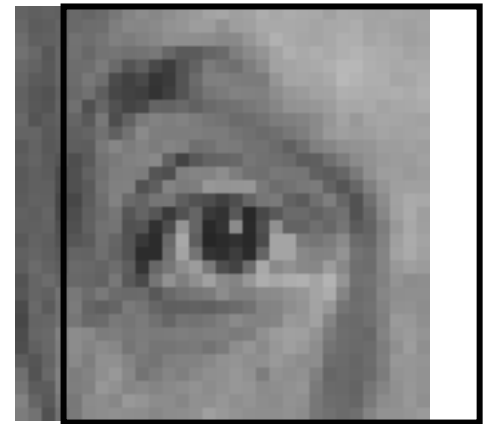
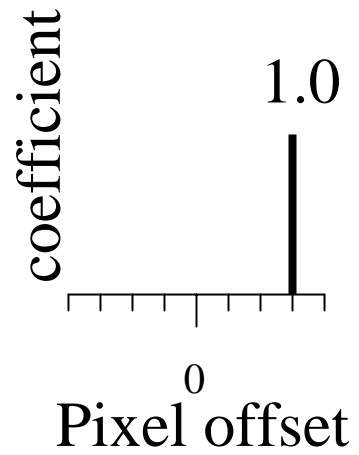


?

shift



original

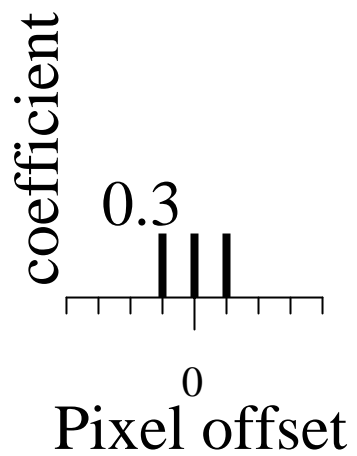


shifted

Linear filtering



original

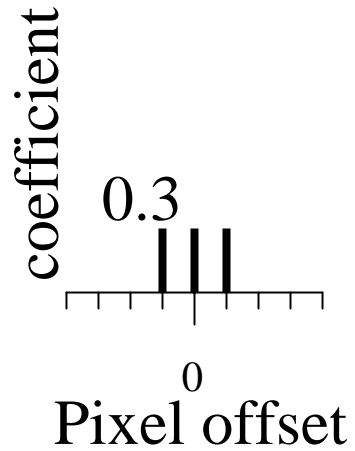


?

Blurring

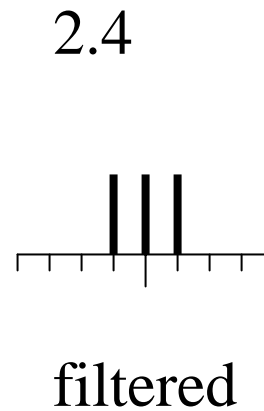
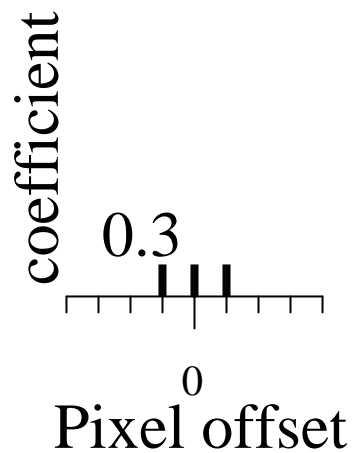
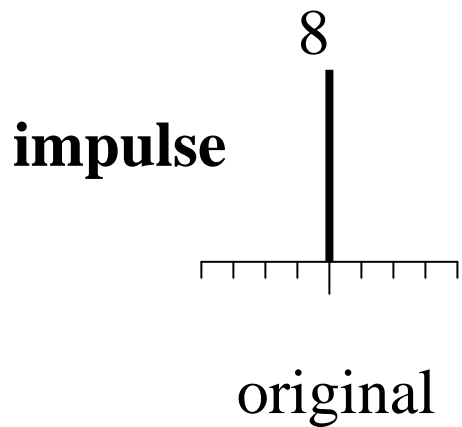


original



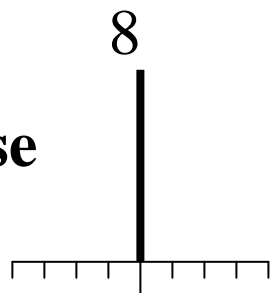
Blurred (filter applied in both dimensions).

Blur Examples



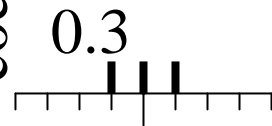
Blur Examples

impulse



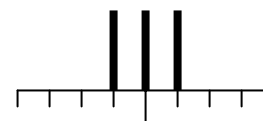
original

coefficient



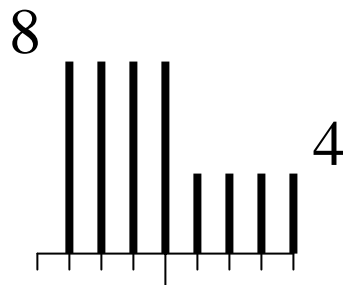
Pixel⁰ offset

2.4



filtered

edge

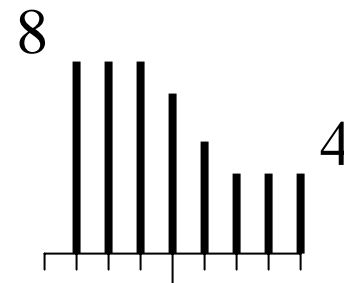


original

coefficient

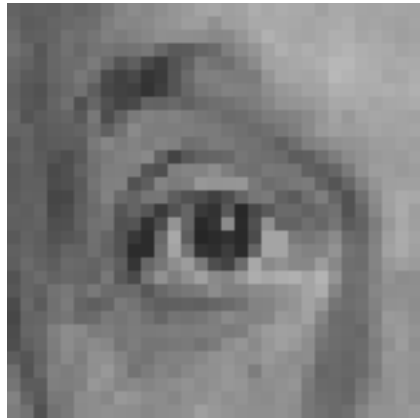


Pixel⁰ offset

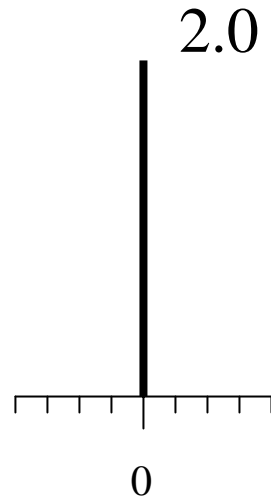


filtered

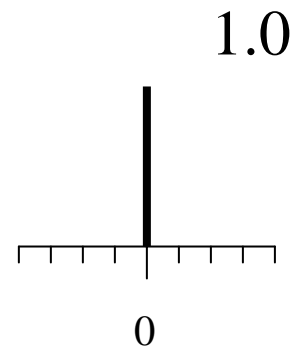
Linear filtering (warm-up slide)



original



—

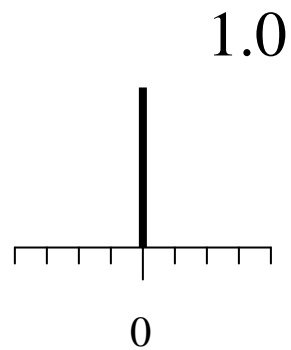
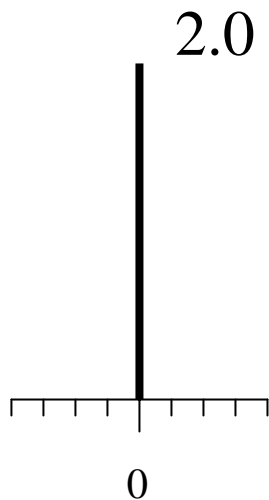


?

Linear Filtering (no change)

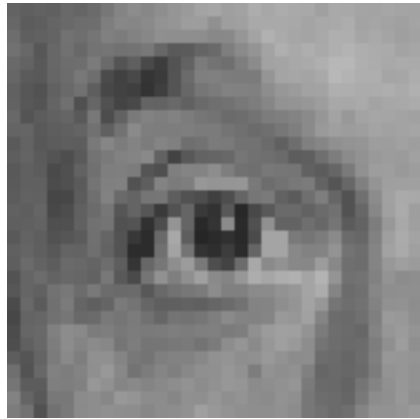


original

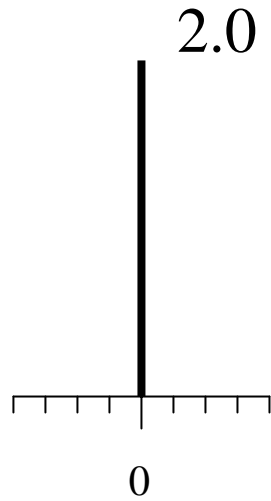


Filtered
(no change)

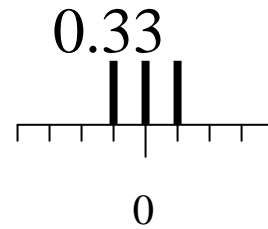
Linear Filtering



original



—

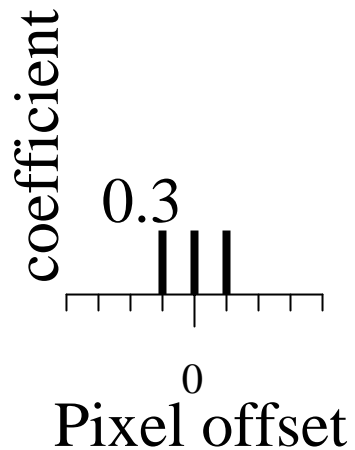


?

(remember blurring)



original

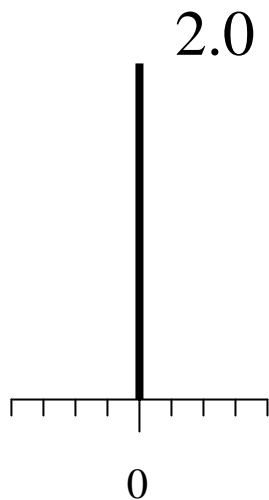


Blurred (filter applied in both dimensions).

Sharpening

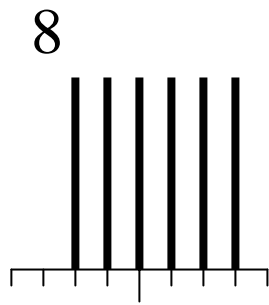


original

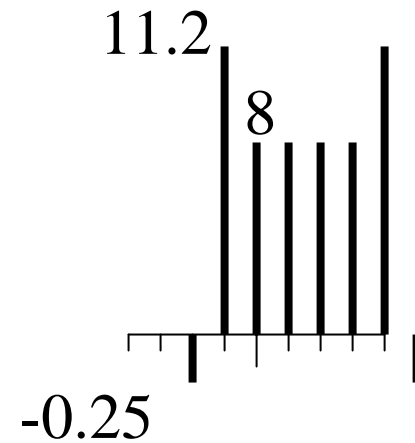
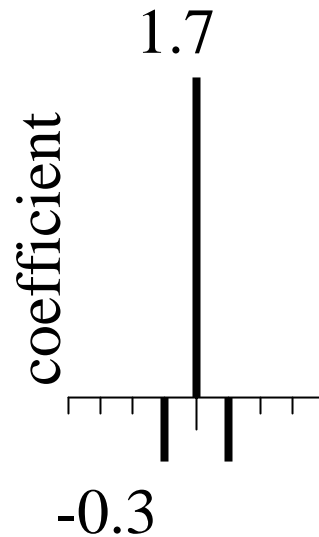


Sharpened
original

Sharpening example

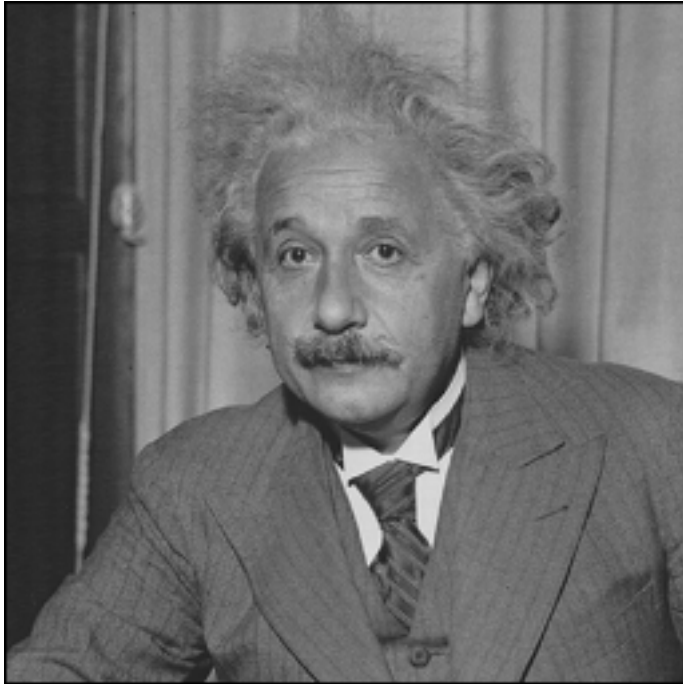


original

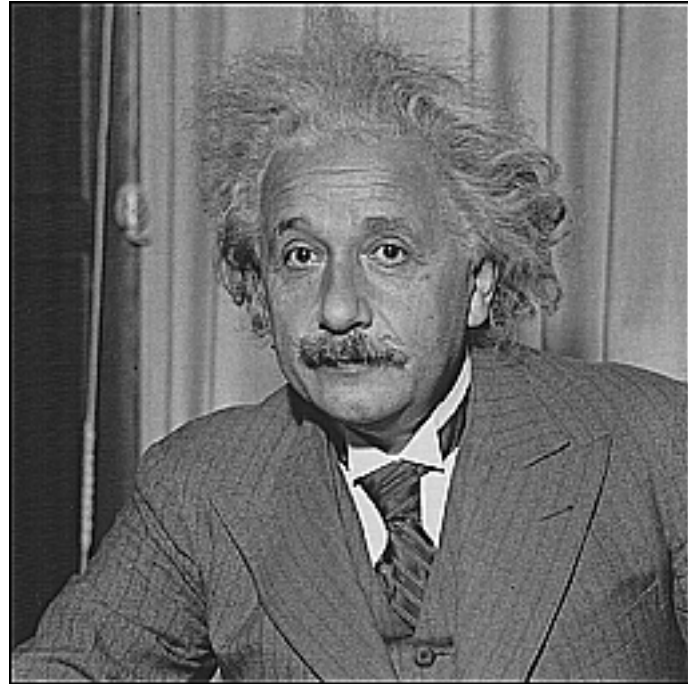


Sharpened
(differences are
accentuated; constant
areas are left untouched).

Sharpening



before



after

Spatial resolution and color



original



R



G



B

Blurring the G component



original



processed



R



G



B

Blurring the R component



original



processed



R



G



B

Blurring the B component



original



processed



R



G



B

Lab Color Component



L A rotation of the
color
coordinates into
directions that
are more
perceptually
meaningful:
L: luminance,
a: red-green,
b: blue-yellow

Blurring L



original



processed



L



a



b

Blurring a



original



processed



L



a



b

Blurring b



original



processed



L



a

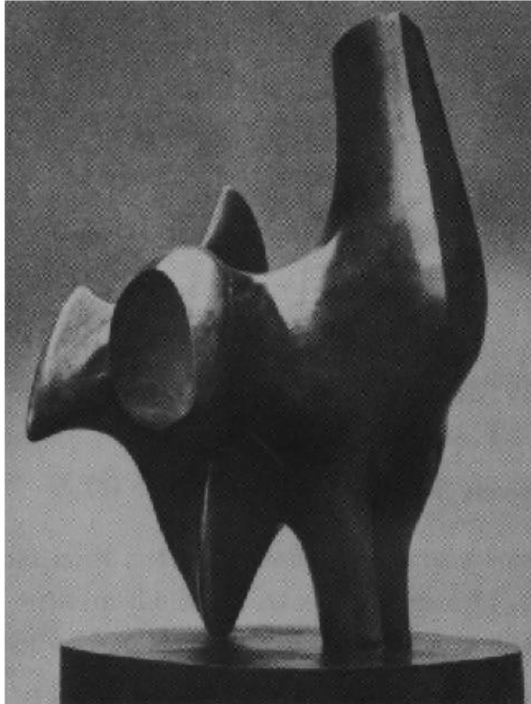


b

Application to image compression

- (compression is about hiding differences from the true image where you can't see them).

Edge Detection



- Convert a 2D image into a set of curves
 - Extracts salient features of the scene
 - More compact than pixels

How can you tell that a pixel is on an edge?

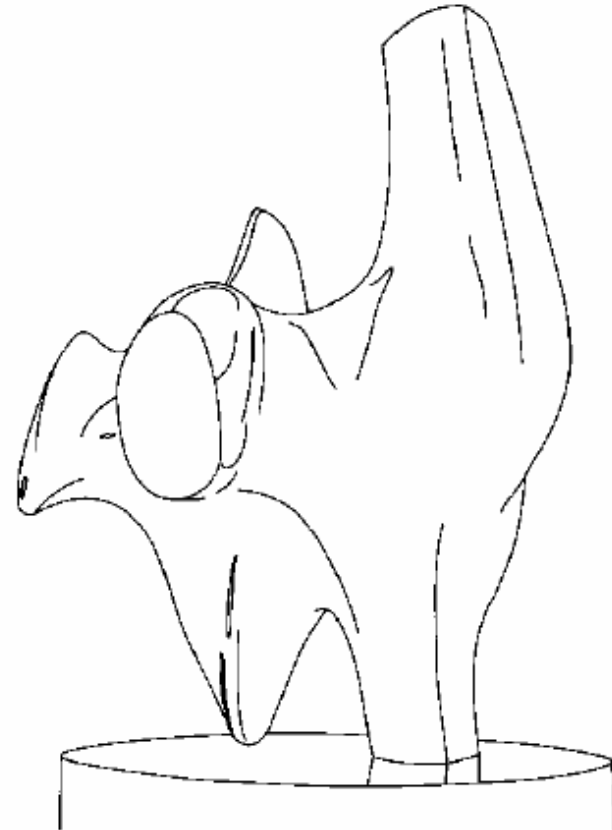
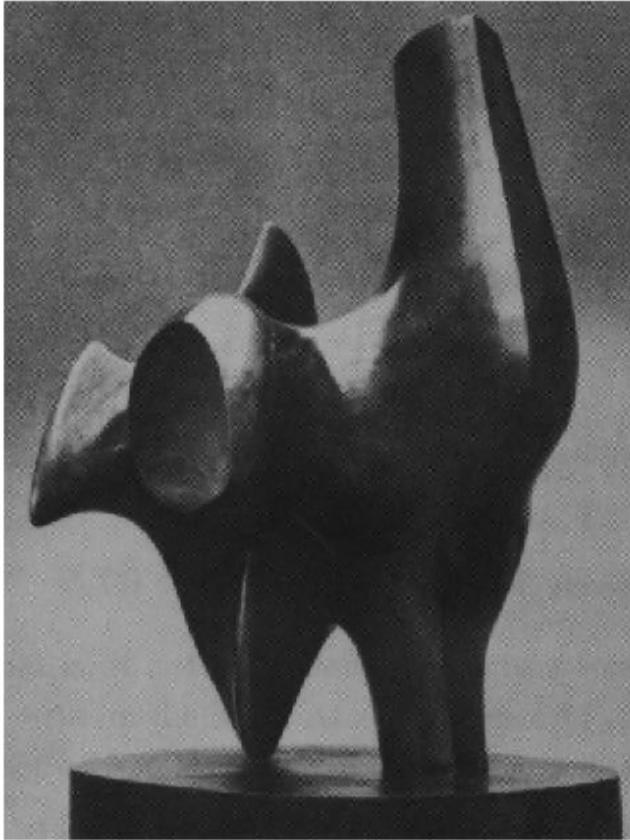
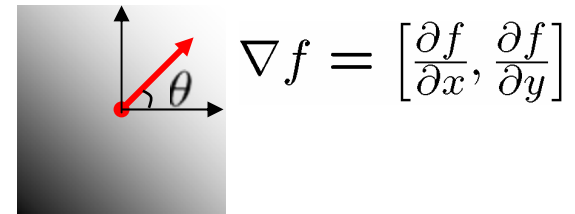
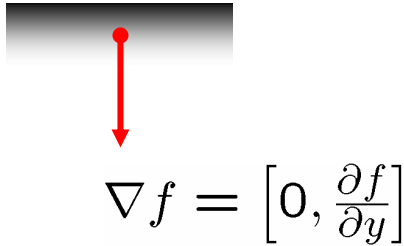
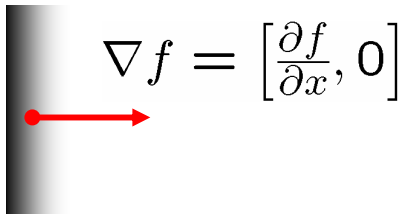


Image gradient

- The gradient of an image:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

- The gradient points in the direction of most rapid change in intensity



- The gradient direction is given by:

$$\theta = \tan^{-1} \left(\frac{\partial f / \partial y}{\partial f / \partial x} \right)$$

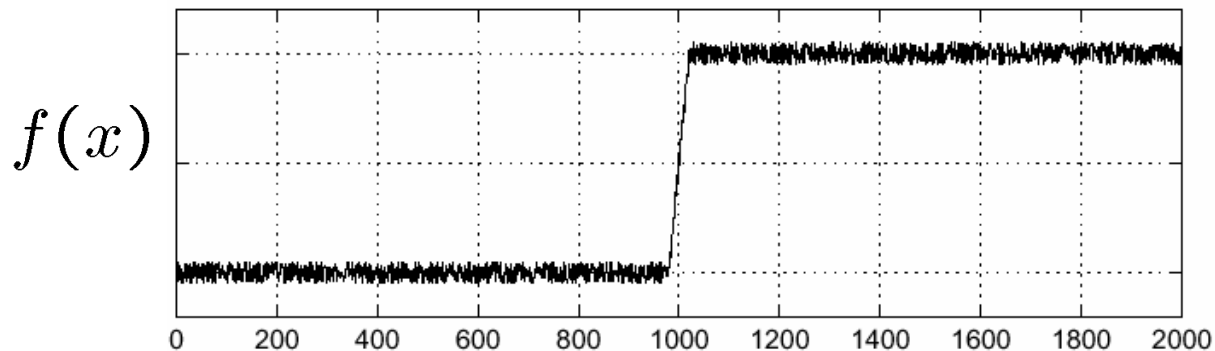
– how does the gradient relate to the direction of the edge?

- The *edge strength* is given by the gradient magnitude

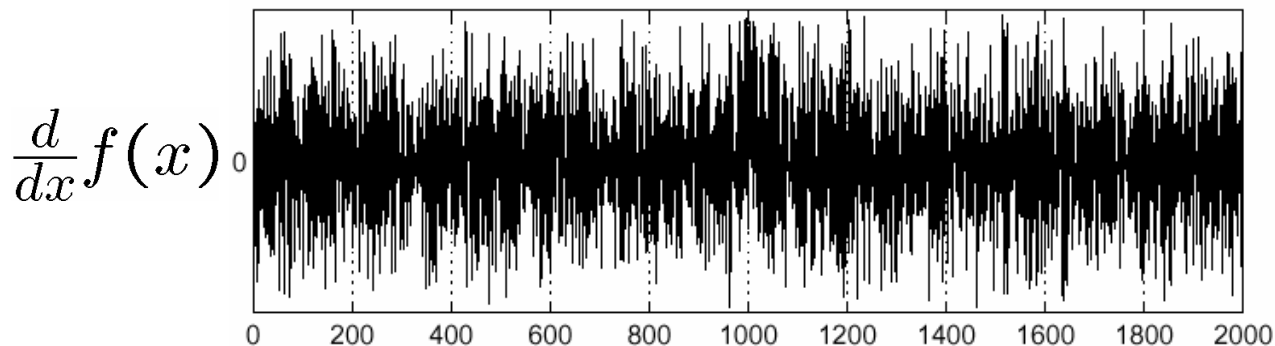
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Effects of noise

- Consider a single row or column of the image
 - Plotting intensity as a function of position gives a signal

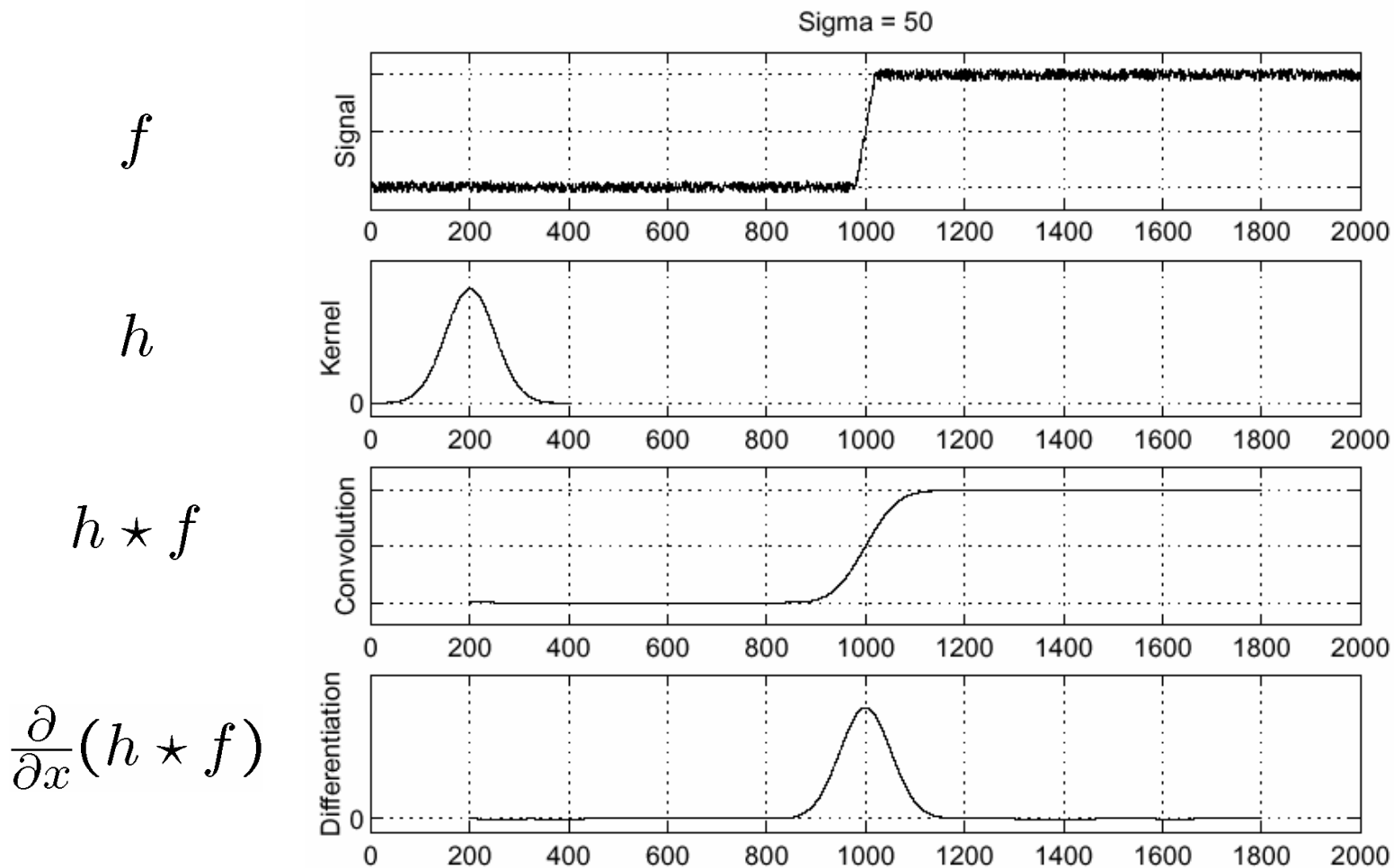


How to compute a derivative?



- Where is the edge?

Solution: smooth first

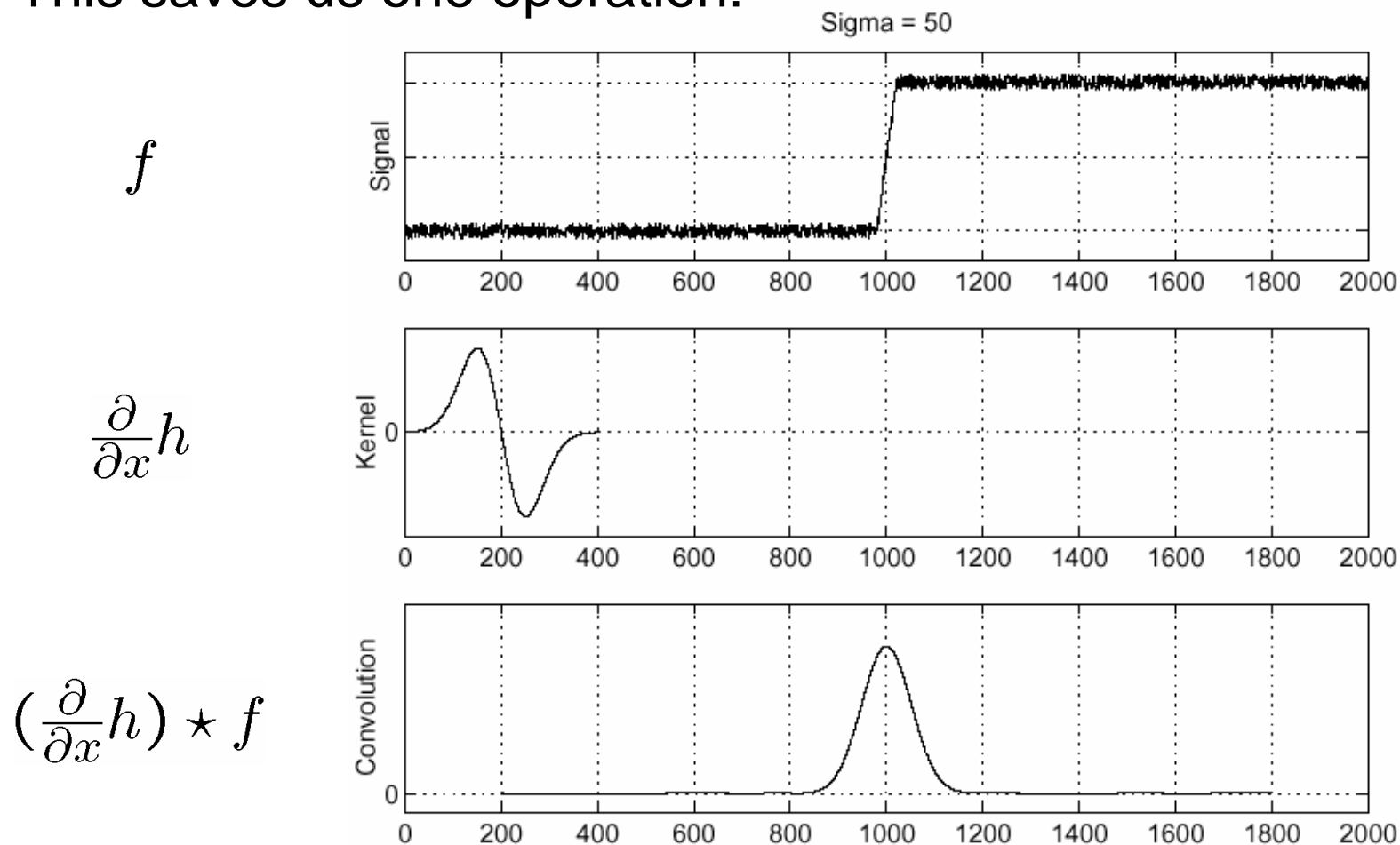


- Where is the edge?
- Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Derivative theorem of convolution

$$\frac{\partial}{\partial x}(h \star f) = \left(\frac{\partial}{\partial x}h\right) \star f$$

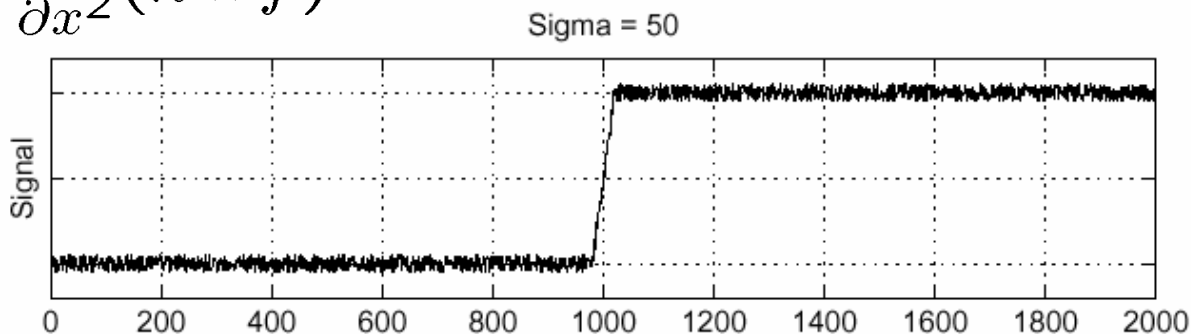
- This saves us one operation:



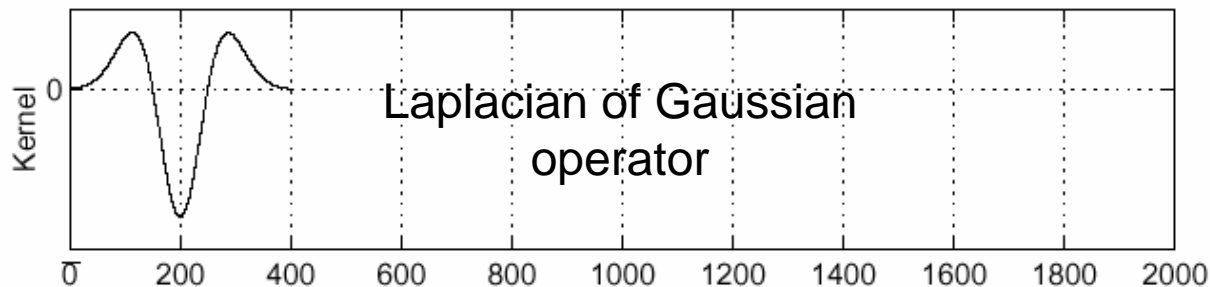
Laplacian of Gaussian

- Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

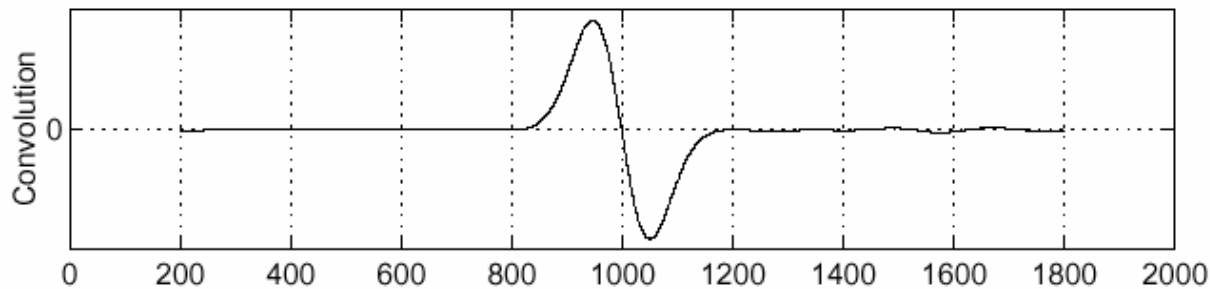
f



$\frac{\partial^2}{\partial x^2}h$



$(\frac{\partial^2}{\partial x^2}h) \star f$

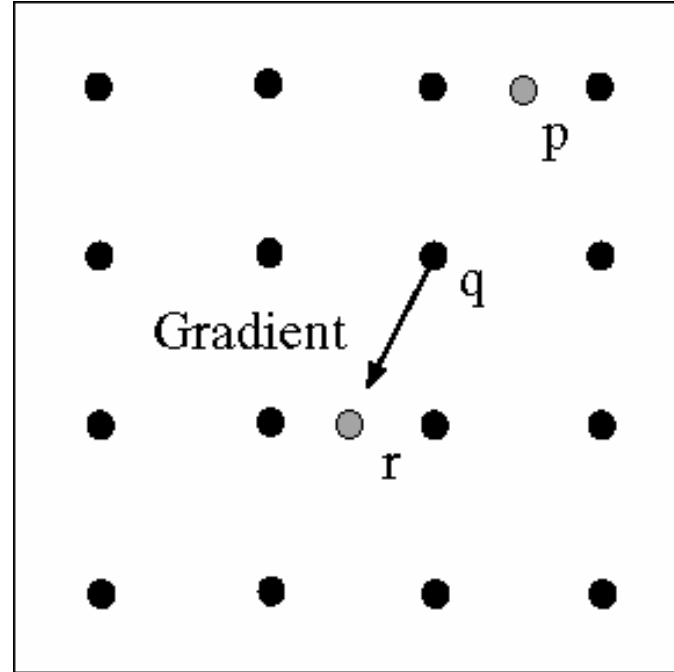
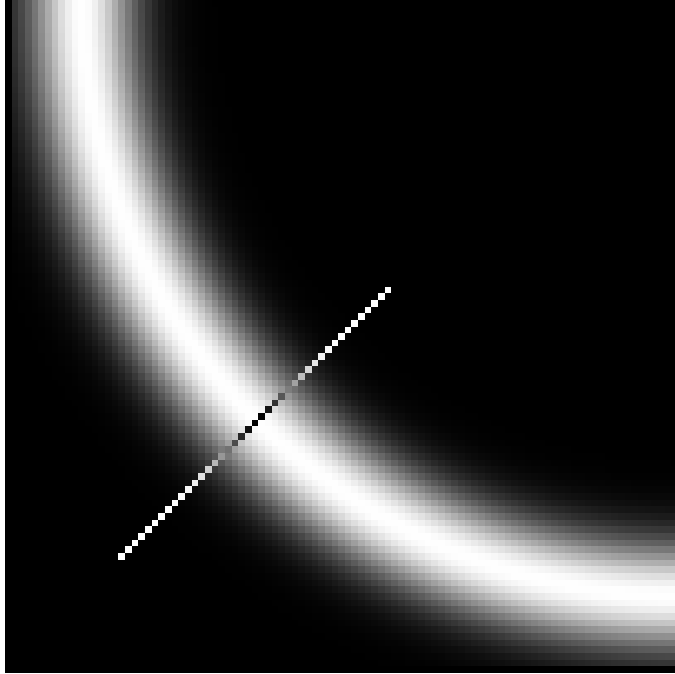


- Where is the edge? • Zero-crossings of bottom graph

Canny Edge Detector

- Smooth image I with 2D Gaussian: $G * I$
- Find local edge normal directions for each pixel $\theta = \arctan \frac{I_y}{I_x}$
- Along this direction, compute image gradient $|\nabla(G * I)|$
- Locate edges by finding max gradient magnitude (**Non-maximum suppression**)

Non-maximum Suppression



- Check if pixel is local maximum along gradient direction
 - requires checking interpolated pixels p and r

The Canny Edge Detector



original image (Lena)

The Canny Edge Detector



magnitude of the gradient

The Canny Edge Detector



After non-maximum suppression

Canny Edge Detector



original



Canny with $\sigma = 1$



Canny with $\sigma = 2$

- The choice of σ depends on desired behavior
 - large σ detects large scale edges
 - small σ detects fine features

Image Scaling

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

How to generate a half-sized version?

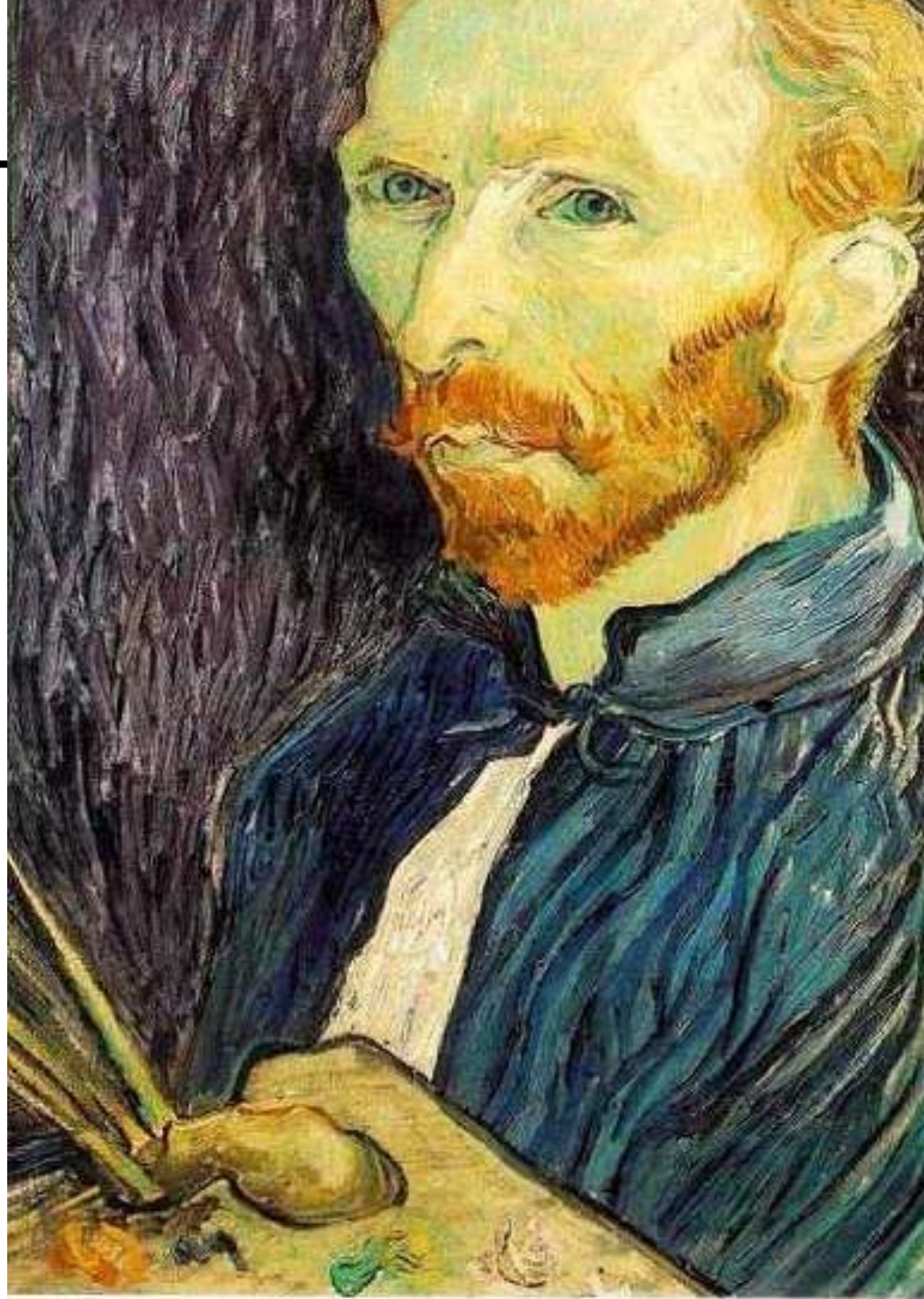
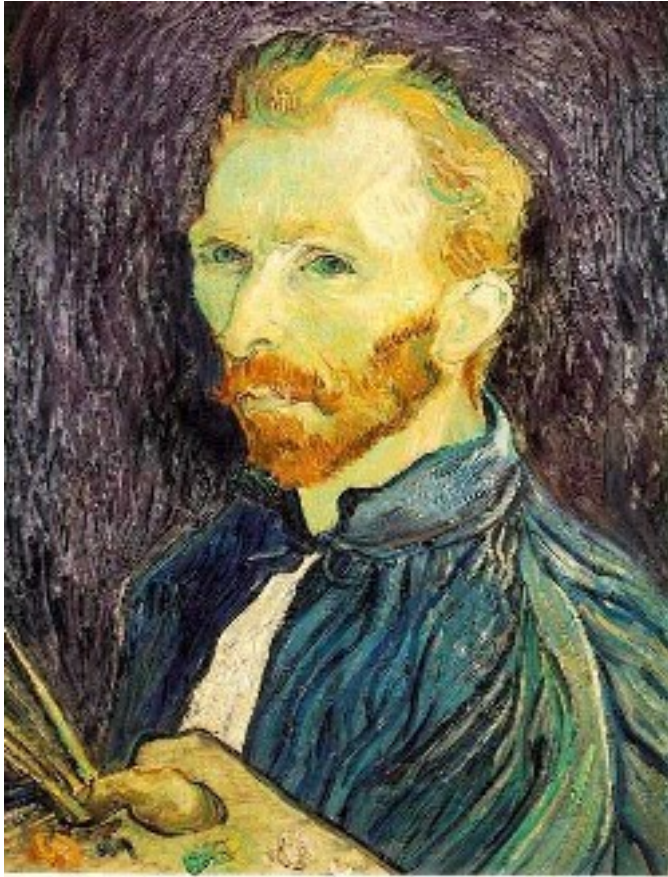


Image sub-sampling



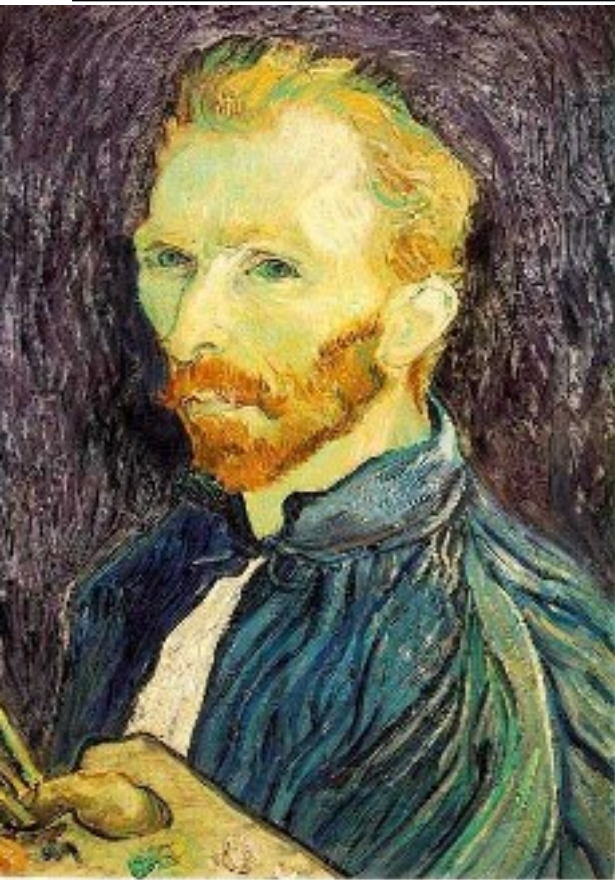
1/4



1/8

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

Image sub-sampling



1/2



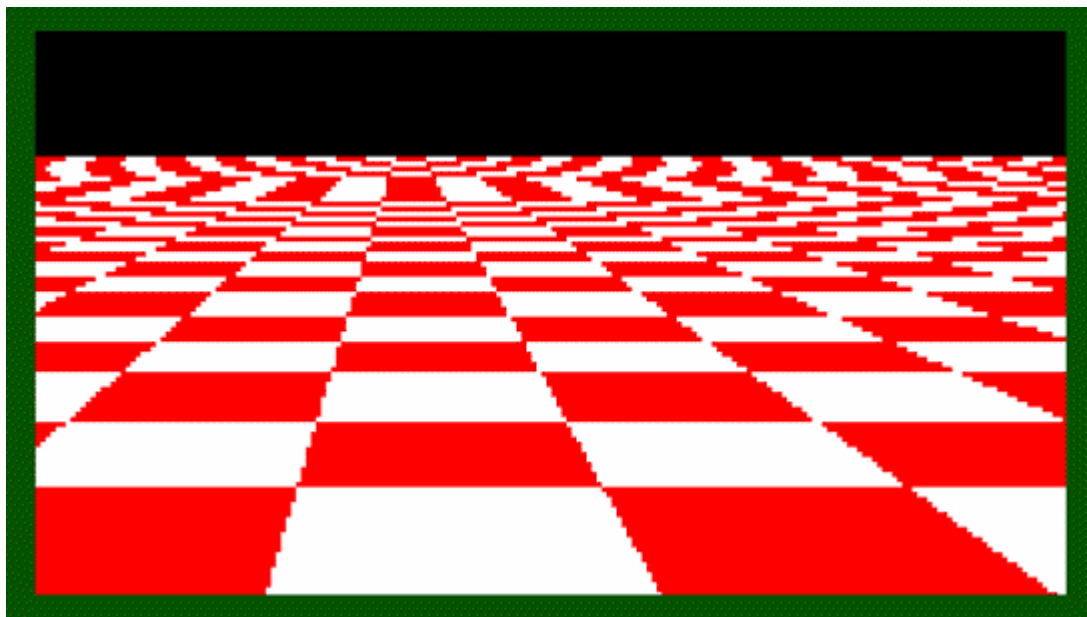
1/4 (2x zoom)



1/8 (4x zoom)

Why does this look so cruffy?

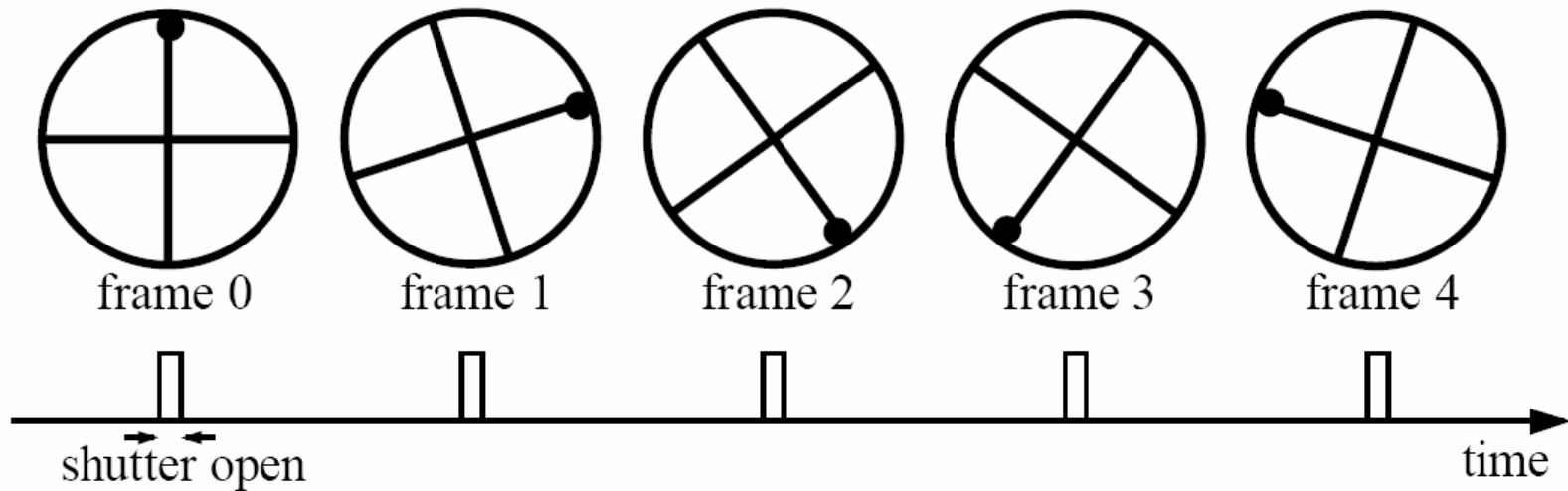
Even worse for synthetic images



Really bad in video

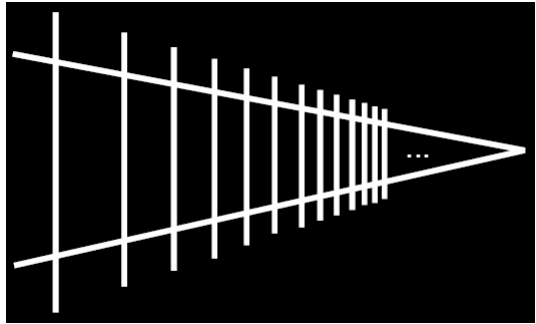
Imagine a spoked wheel moving to the right (rotating clockwise).
Mark wheel with dot so we can see what's happening.

If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):



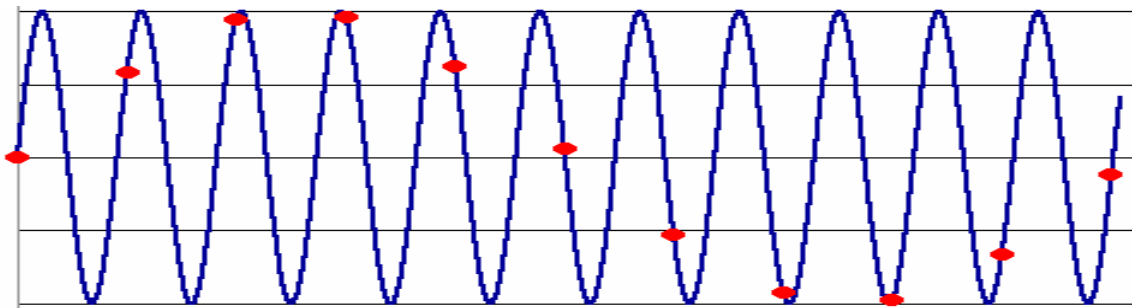
Without dot, wheel appears to be rotating slowly backwards!
(counterclockwise)

Alias: n., an assumed name



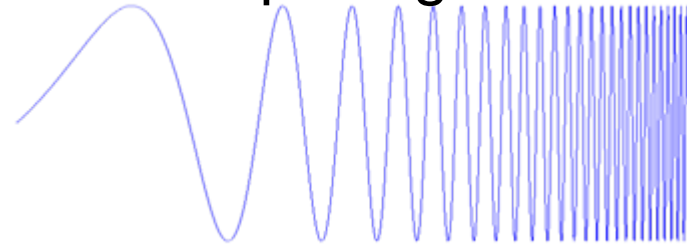
Picket fence receding
Into the distance will
produce aliasing...

WHY?

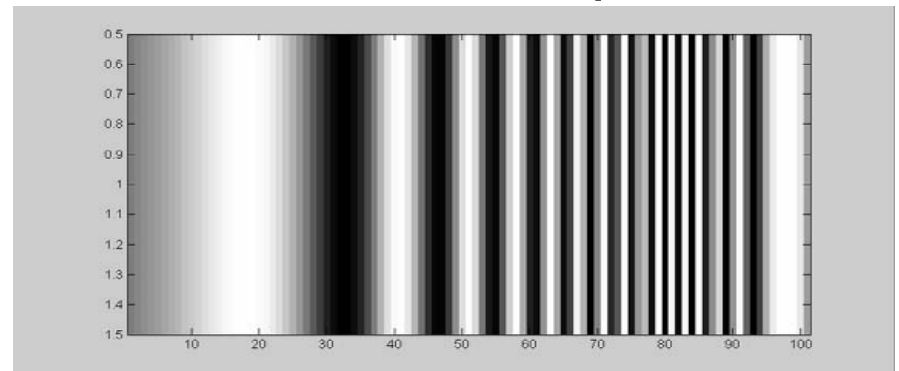


Not enough samples

Input signal:



Matlab output:



`x = 0:.05:5; imagesc(sin((2.^x).*x))`

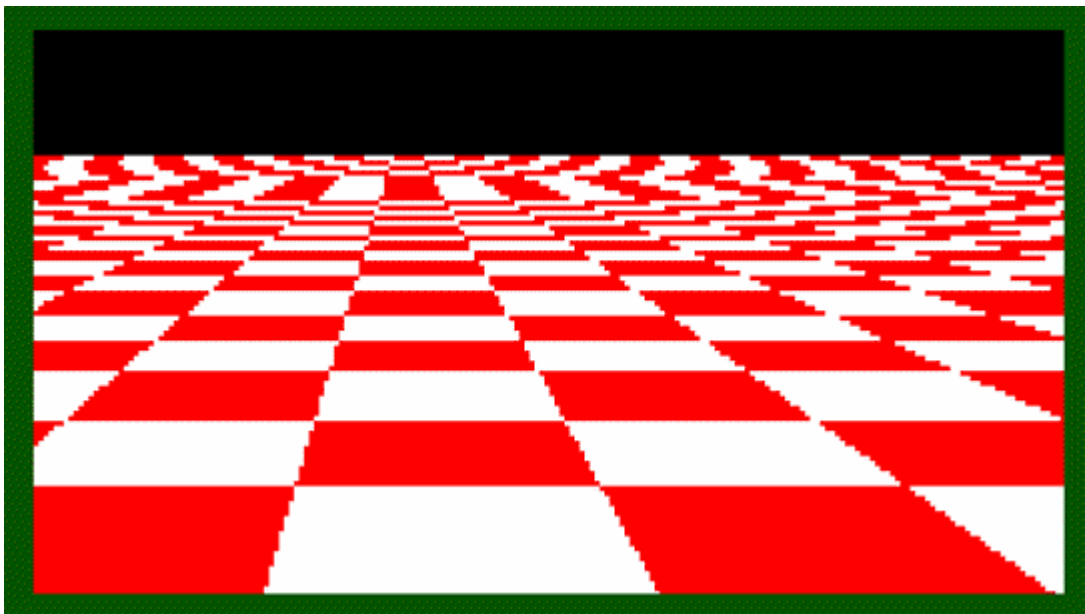
Aj-aj-aj:
Alias!

Aliasing

- occurs when your sampling rate is not high enough to capture the amount of detail in your image
- Can give you the wrong signal/image—an *alias*
- Where can it happen in images?
- During image synthesis:
 - **sampling** continuous signal into discrete signal
 - e.g. ray tracing, line drawing, function plotting, etc.
- During image processing:
 - **resampling** discrete signal at a different rate
 - e.g. Image warping, zooming in, zooming out, etc.
- To do sampling right, need to understand the structure of your signal/image
- **Enter Monsieur Fourier...**

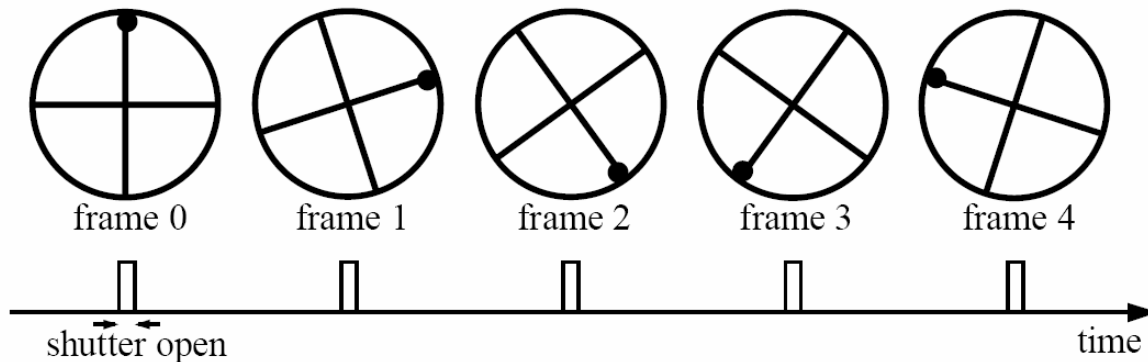
Antialiasing

- What can be done?
 1. Raise sampling rate by *oversampling*
 - *Sample at k times the resolution*
 - continuous signal: easy
 - discrete signal: need to interpolate



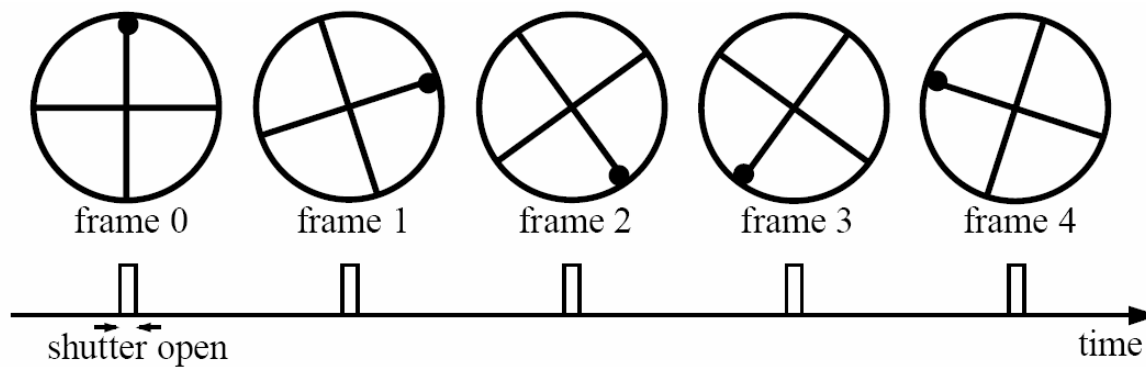
Antialiasing

- What can be done?
 1. Raise sampling rate by *oversampling*
 - *Sample at k times the resolution*
 - continuous signal: easy
 - discrete signal: need to interpolate



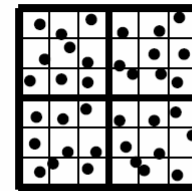
Antialiasing

- What can be done?
 1. Raise sampling rate by *oversampling*
 - Sample at k times the resolution
 - continuous signal: easy
 - discrete signal: need to interpolate
 2. Lower the max frequency by *prefiltering*
 - Smooth the signal enough
 - Works on discrete signals



Antialiasing

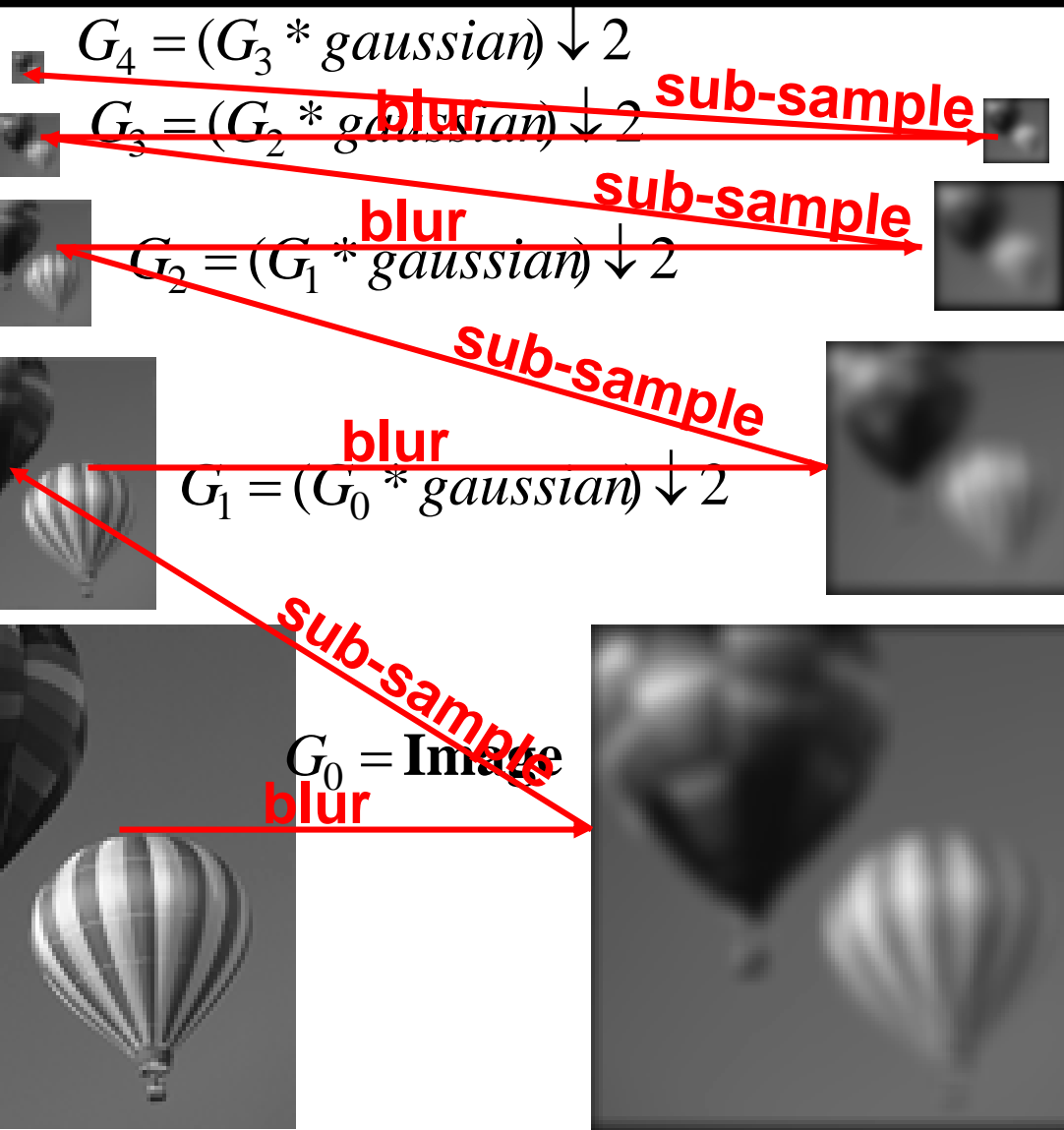
- What can be done?
 1. Raise sampling rate by *oversampling*
 - *Sample at k times the resolution*
 - continuous signal: easy
 - discrete signal: need to interpolate
 2. Lower the max frequency by *prefiltering*
 - Smooth the signal enough
 - Works on discrete signals
 3. Improve sampling quality with better sampling (CS559)



jittered,
9 samples per pixel

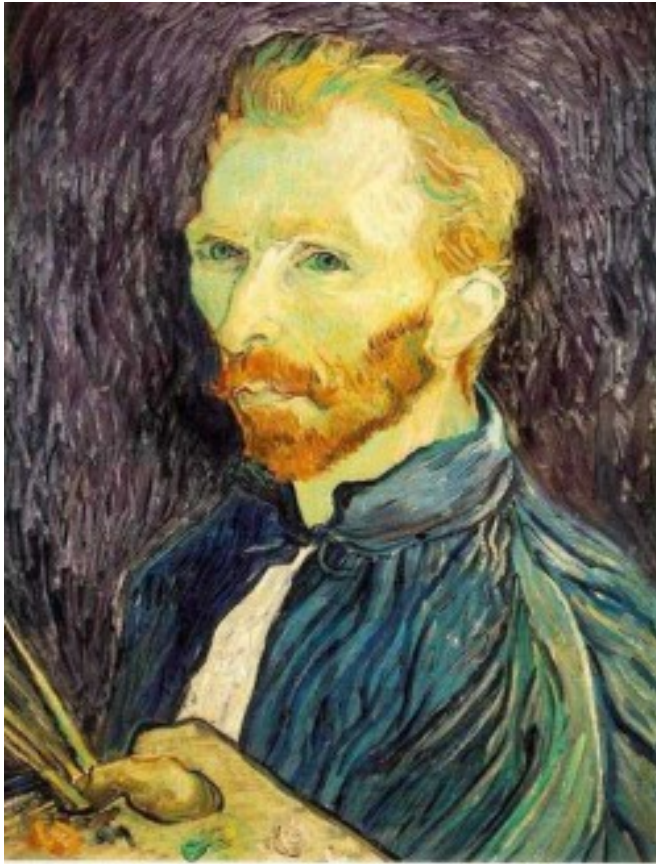
The Gaussian Pyramid

Low resolution



High resolution

Gaussian pre-filtering



Gaussian 1/2



G 1/4



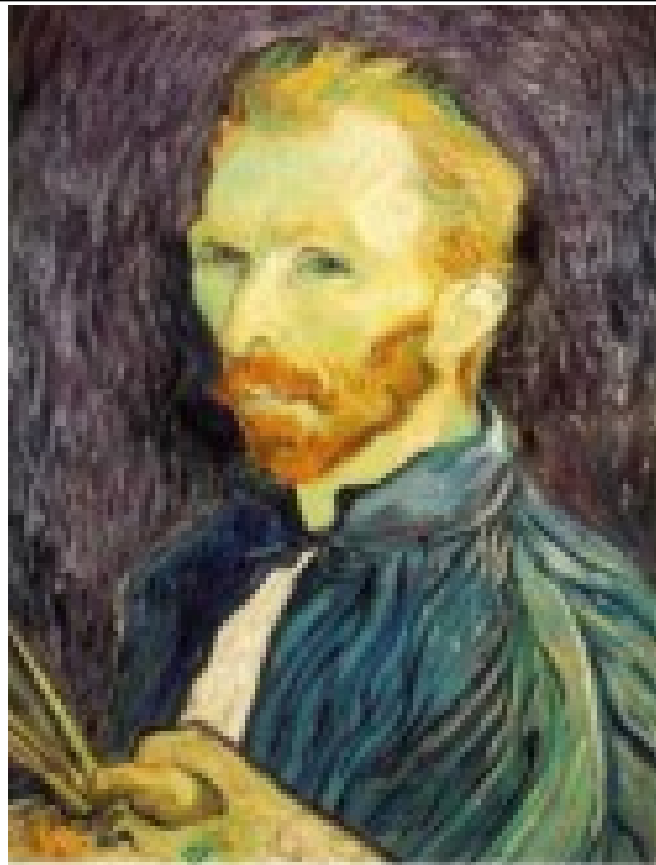
G 1/8

- Solution: filter the image, *then* subsample

Subsampling with Gaussian pre-filtering



Gaussian 1/2



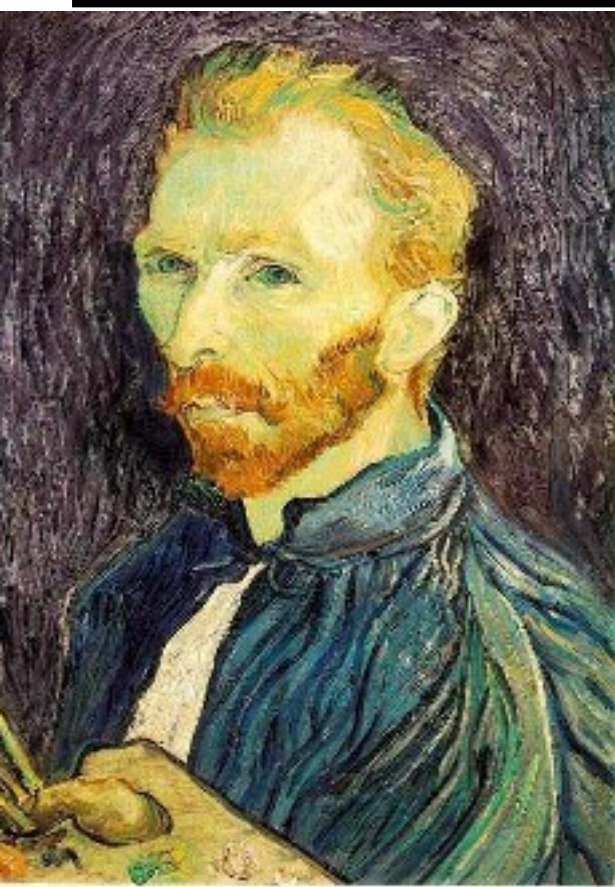
G 1/4



G 1/8

- Solution: filter the image, *then* subsample

Compare with...



1/2

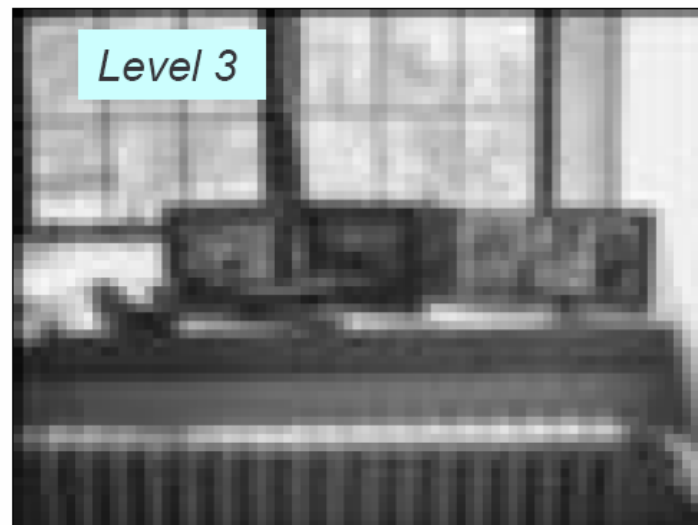
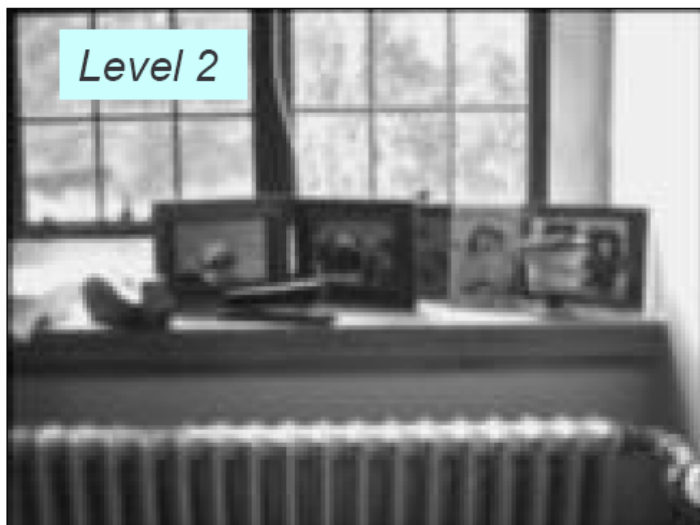


1/4 (2x zoom)



1/8 (4x zoom)

Pyramids at Same Resolution



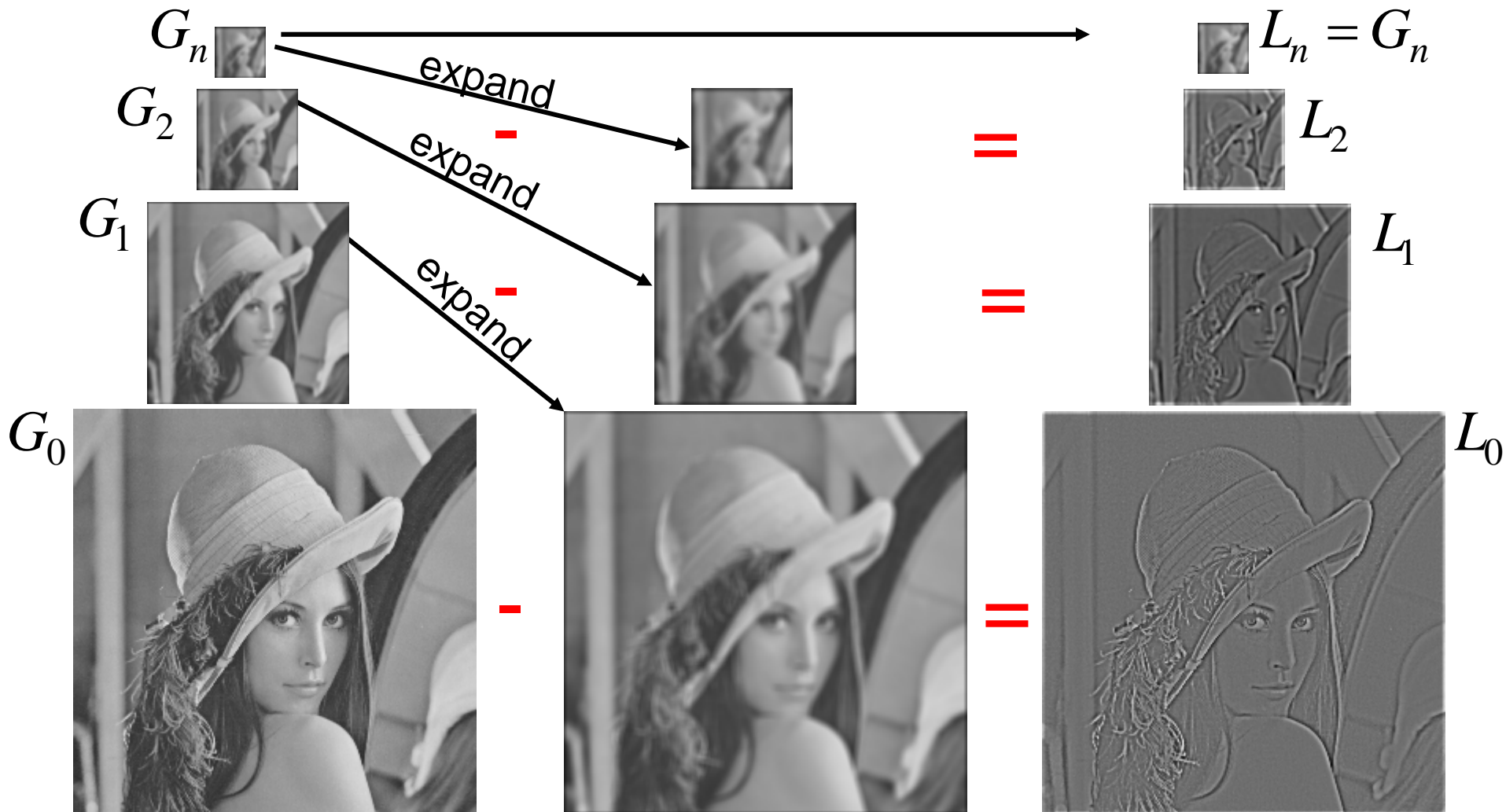
The Laplacian Pyramid

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

Gaussian Pyramid

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

Laplacian Pyramid



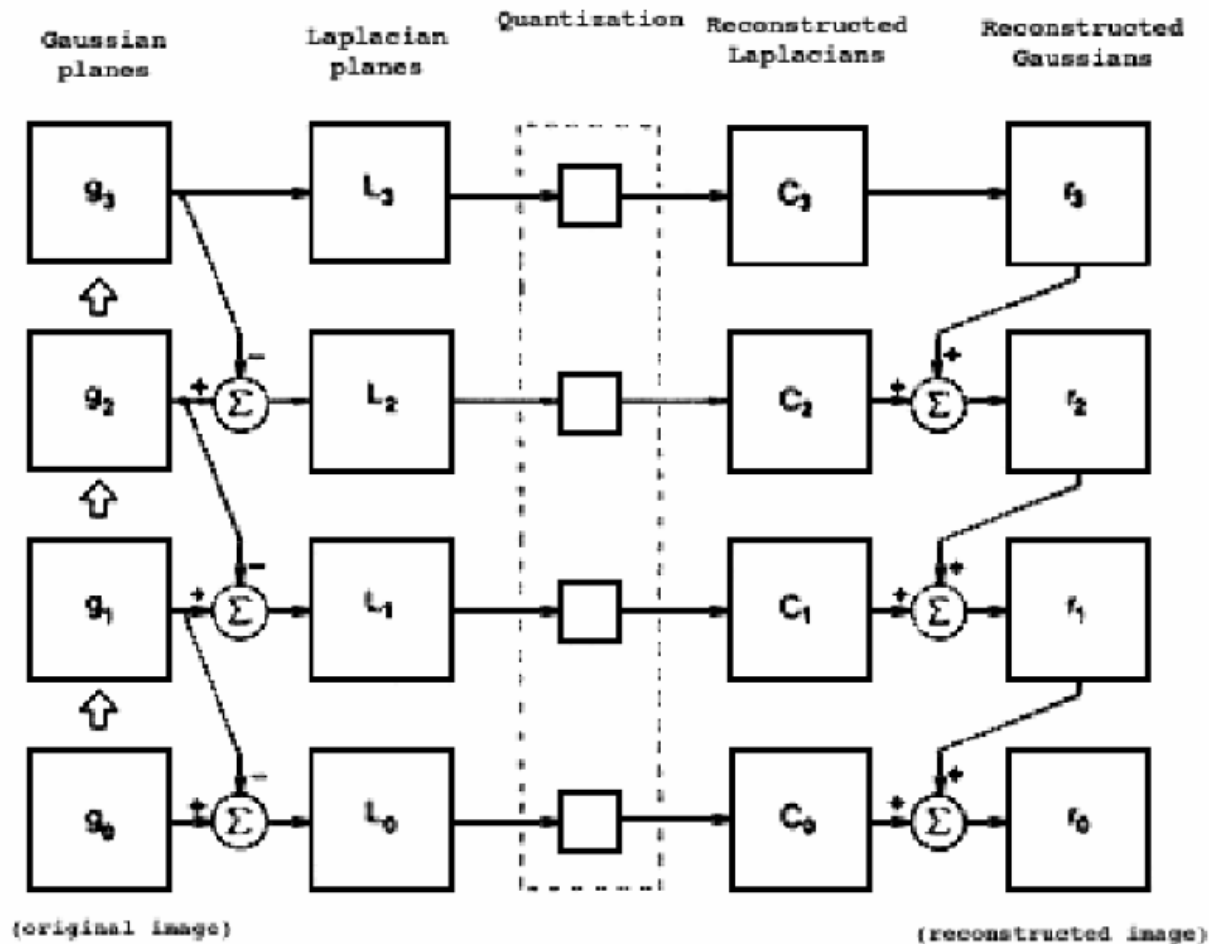


Fig. 10. A summary of the steps in Laplacian pyramid coding and decoding. First, the original image g_0 (lower left) is used to generate Gaussian pyramid levels g_1, g_2, \dots through repeated local averaging. Levels of the Laplacian pyramid L_0, L_1, \dots are then computed as the differences between adjacent Gaussian levels. Laplacian pyramid elements are quantized to yield the Laplacian pyramid code C_0, C_1, C_2, \dots . Finally, a reconstructed image r_0 is generated by summing levels of the code pyramid.

Recap

Image Processing: from basic concepts to latest techniques

- Filtering
- Edge detection
- Re-sampling and aliasing
- Image Pyramids (Gaussian and Laplacian)
- Next ...