Thinking in Frequency

Computer Vision

Jia-Bin Huang, Virginia Tech

Dali: “Gala Contemplating the Mediterranean Sea” (1976)

Administrative stuffs

- Course website: http://bit.ly/vt-computer-vision-fall-2016

- Office hours - Jia-Bin (440 Whittemore Hall)
  - Monday at 1:00 PM – 2:00 PM (final project) – sign up here
  - Friday at 3:00 PM – 4:00 PM (lectures, HW discussions)

- MATLAB tutorial session by Akrit
  - Friday 3-4 PM, Whittemore Hall 340A
  - Bring your laptop with MATLAB installed

- HW 1 will be posted tomorrow (Sept 2). Due date: Sept 19.

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, extrapolation, cropping
- Noise models and nonlinear image filters

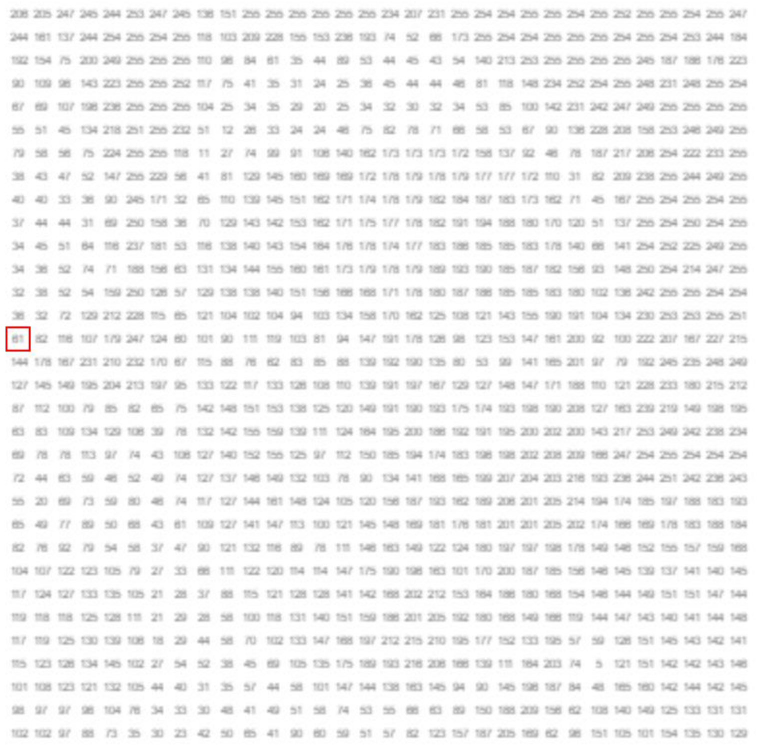
$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Today’s class

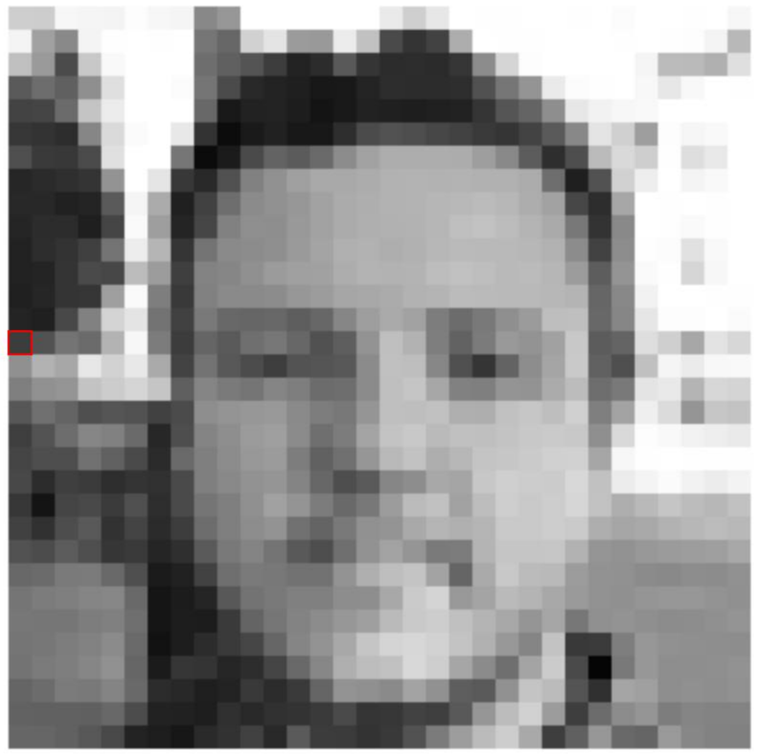
- Review of image filtering in spatial domain
  - Application: representing textures
  - Noise models and nonlinear image filters
- Fourier transform and frequency domain
- Frequency view of filtering
- Image downsizing and interpolation

Demo

- http://setosa.io/ev/image-kernels/



208	205	247	245	244	253	247	245	138	151	255	255	255	255	255	255	234	207	231	255	254	254	255	255	254	255	252	255	255	254	255	247
244	181	137	244	254	255	254	255	118	103	209	228	155	153	238	193	74	52	68	173	255	254	254	255	255	255	254	255	254	253	244	184
192	154	75	200	249	255	255	255	110	98	84	61	35	44	89	53	44	45	43	54	140	213	253	255	255	255	255	245	187	188	178	223
90	109	98	143	223	255	255	252	117	75	41	35	31	24	25	38	45	44	44	48	81	118	148	234	252	254	255	248	231	248	255	254
67	69	107	198	238	255	255	255	104	25	34	35	29	20	25	34	32	30	32	34	53	85	100	142	231	242	247	249	255	255	255	255
55	51	45	134	218	251	255	232	51	12	28	33	24	24	48	75	82	78	71	68	58	53	67	90	138	228	208	158	253	248	249	255
79	58	58	75	224	255	255	118	11	27	74	99	91	108	140	162	173	173	173	172	158	137	92	48	78	187	217	208	254	222	233	255
38	43	47	52	147	255	229	58	41	81	129	145	160	169	169	172	178	179	178	179	177	177	172	110	31	82	209	238	255	244	249	255
40	40	33	38	90	245	171	32	65	110	139	145	151	162	171	174	178	179	182	184	187	183	173	162	71	45	167	255	254	255	254	255
37	44	44	31	69	250	158	38	70	129	143	142	153	162	171	175	177	178	182	191	194	188	180	170	120	51	137	255	254	250	254	255
34	45	51	64	118	237	181	53	118	138	140	143	154	164	178	178	174	177	183	188	185	185	183	178	140	68	141	254	252	225	249	255
34	38	52	74	71	188	158	63	131	134	144	155	160	161	173	179	178	179	189	193	190	185	187	182	158	93	148	250	254	214	247	255
32	38	52	54	159	250	128	57	129	138	138	140	151	156	168	168	171	178	180	187	188	185	185	183	180	102	138	242	255	255	254	254
38	32	72	129	212	228	115	65	121	104	102	104	94	103	134	158	170	162	125	108	121	143	155	190	191	104	134	230	253	253	255	251
61	82	118	107	179	247	124	60	101	90	111	119	103	81	94	147	191	178	128	98	123	153	147	161	200	92	100	222	207	167	227	215
144	178	167	231	210	232	170	67	115	88	78	82	83	85	88	139	192	190	135	80	53	99	141	165	201	97	79	192	245	235	248	249
127	145	149	195	204	213	197	95	133	122	117	133	128	108	110	139	191	197	167	129	127	148	147	171	188	110	121	228	233	180	215	212
87	112	100	79	85	82	65	75	142	148	151	153	138	125	120	149	191	190	193	175	174	193	198	190	208	127	163	239	219	149	198	195
63	83	109	134	129	108	39	78	132	142	155	159	139	111	124	164	195	200	188	192	191	195	200	202	200	143	217	253	249	242	238	234
69	78	78	113	97	74	43	108	127	140	152	155	125	97	112	150	185	194	174	183	198	198	202	208	209	188	247	254	255	254	254	254
72	44	63	59	48	52	49	74	127	137	148	149	132	103	78	90	134	141	168	165	199	207	204	203	218	193	238	244	251	242	238	243
55	20	69	73	59	80	48	74	117	127	144	161	148	124	105	120	158	187	193	182	189	208	201	205	214	194	174	185	197	188	183	193
65	49	77	89	50	68	43	61	109	127	141	147	113	100	121	145	148	169	181	178	181	201	201	205	202	174	168	169	178	183	188	184
82	78	92	79	54	58	37	47	90	121	132	118	89	78	111	148	163	149	122	124	180	197	197	198	178	149	148	152	155	157	159	168
104	107	122	123	105	79	27	33	68	111	122	120	114	114	147	175	190	198	163	101	170	200	187	185	158	148	145	139	137	141	140	145
117	124	127	133	135	105	21	28	37	88	115	121	128	128	141	142	168	202	212	153	184	188	180	168	154	148	144	149	151	151	147	144
119	118	118	125	128	111	21	29	28	58	100	118	131	140	151	159	188	201	205	192	180	168	149	168	119	144	147	143	140	141	144	148
117	119	125	130	139	108	18	29	44	58	70	102	133	147	168	197	212	215	210	195	177	152	133	195	57	59	128	151	145	143	142	141
115	123	128	134	145	102	27	54	52	38	45	69	105	135	175	189	193	218	208	168	139	111	184	203	74	5	121	151	142	142	143	148
101	108	123	121	132	105	44	40	31	35	57	44	58	101	147	144	138	163	145	94	90	145	198	187	84	48	165	180	142	144	142	145
98	97	97	98	104	78	34	33	30	48	41	49	51	58	74	53	55	68	63	89	150	188	209	158	62	108	140	149	125	133	131	131
102	102	97	88	73	35	30	23	42	50	65	41	90	60	59	51	57	82	123	157	187	205	169	62	98	151	105	101	154	135	130	129



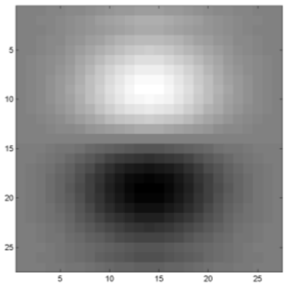
Review: questions

Filtering Operator

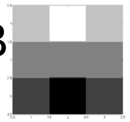
Fill in the blanks:

```
a) _ = D * B
b) A = _ * _
c) F = D * _
d) _ = D * D
```

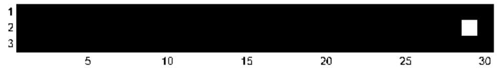
A



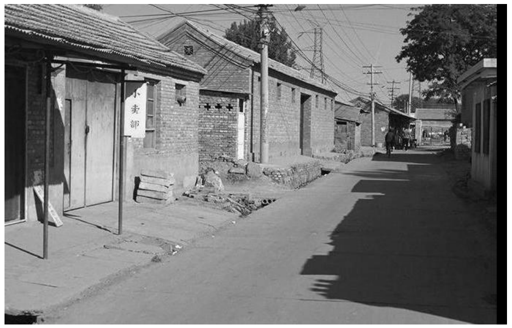
B



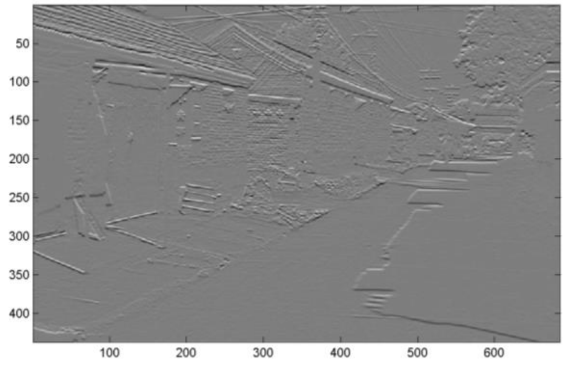
E



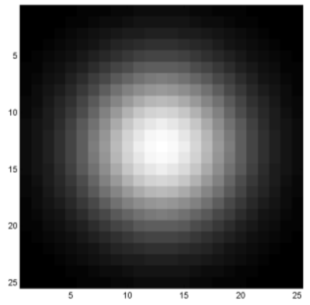
F



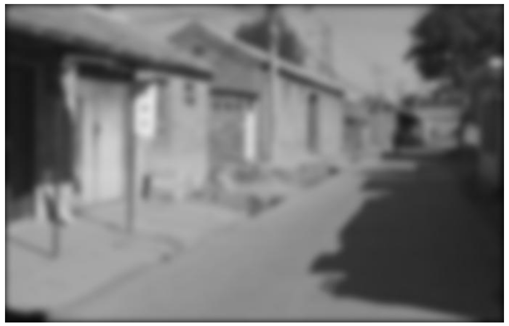
G



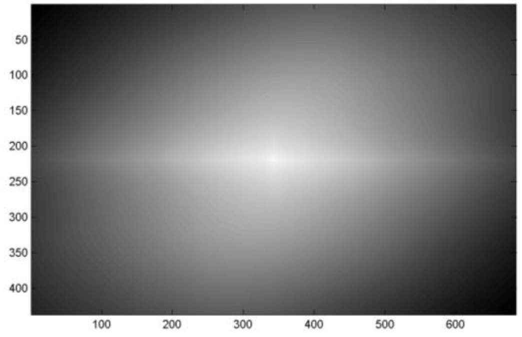
C



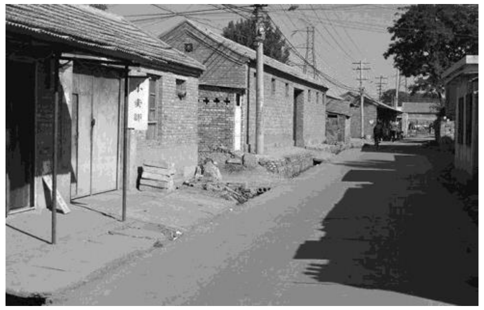
H



I



D

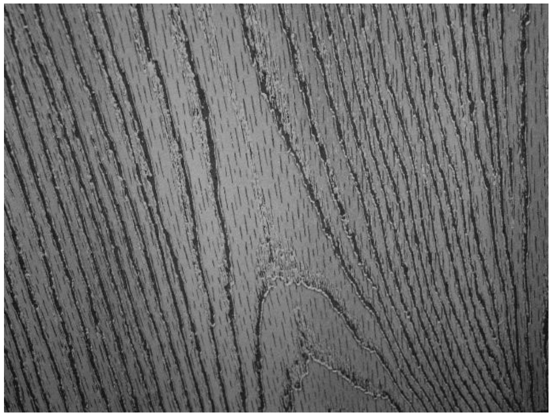
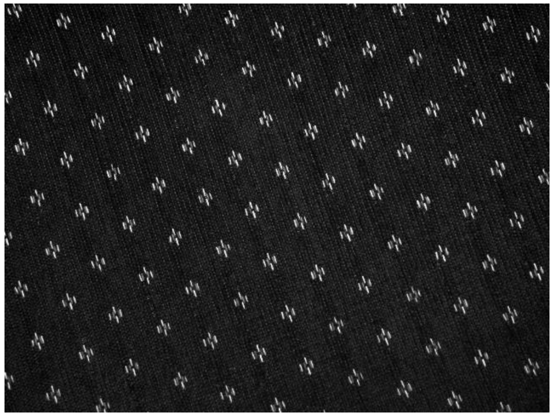


Slide: Hoiem

Application: Representing Texture

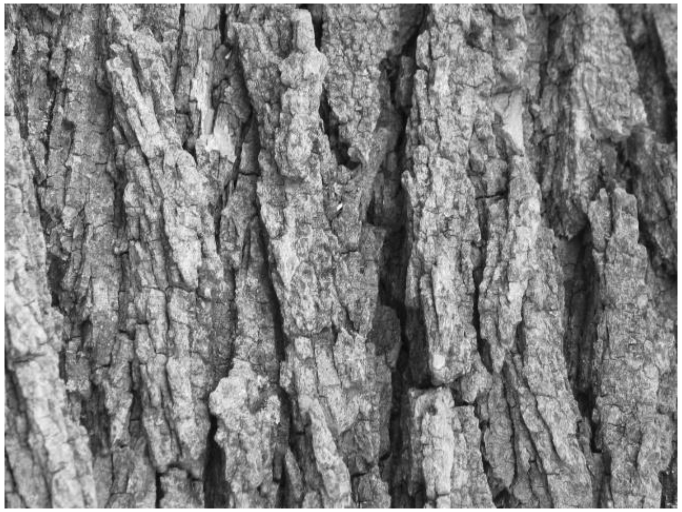
Source: Forsyth

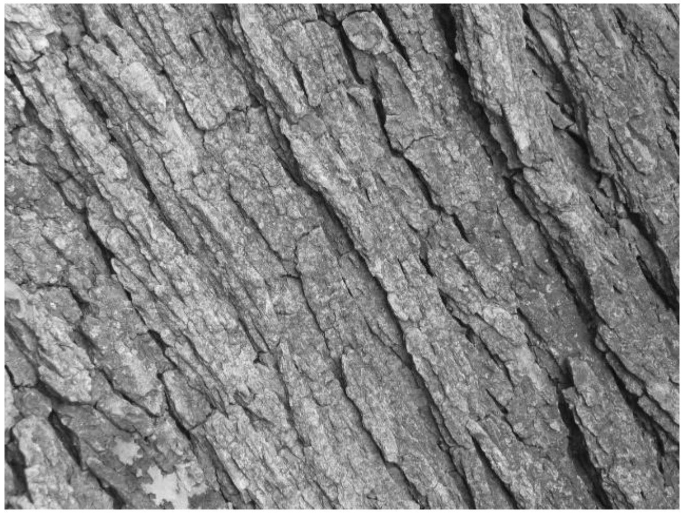
Texture and Material

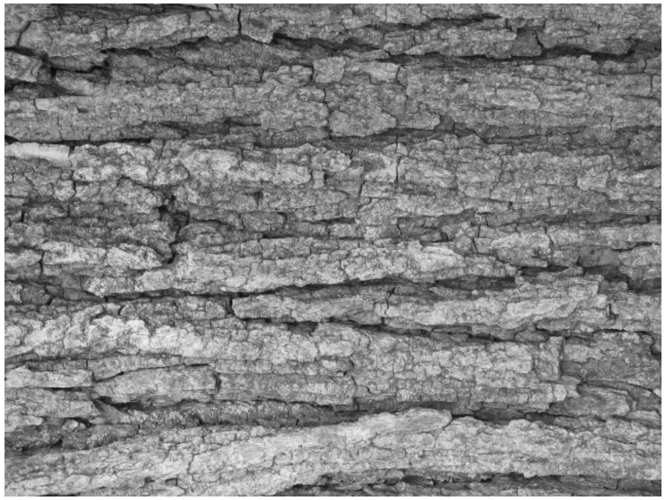



http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Texture and Orientation

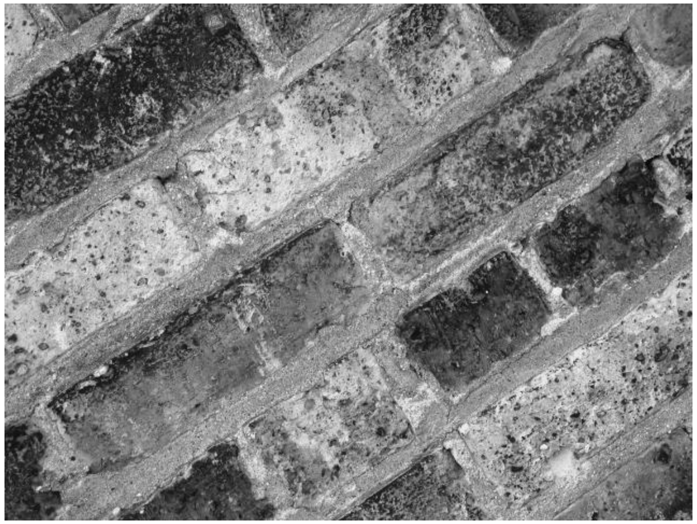


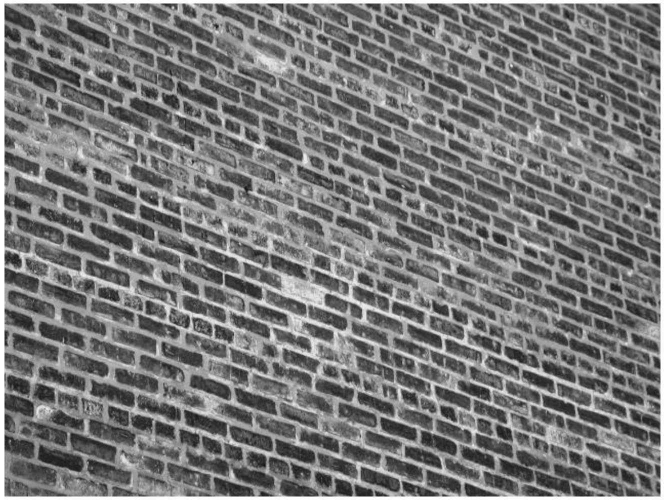




http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Texture and Scale





http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

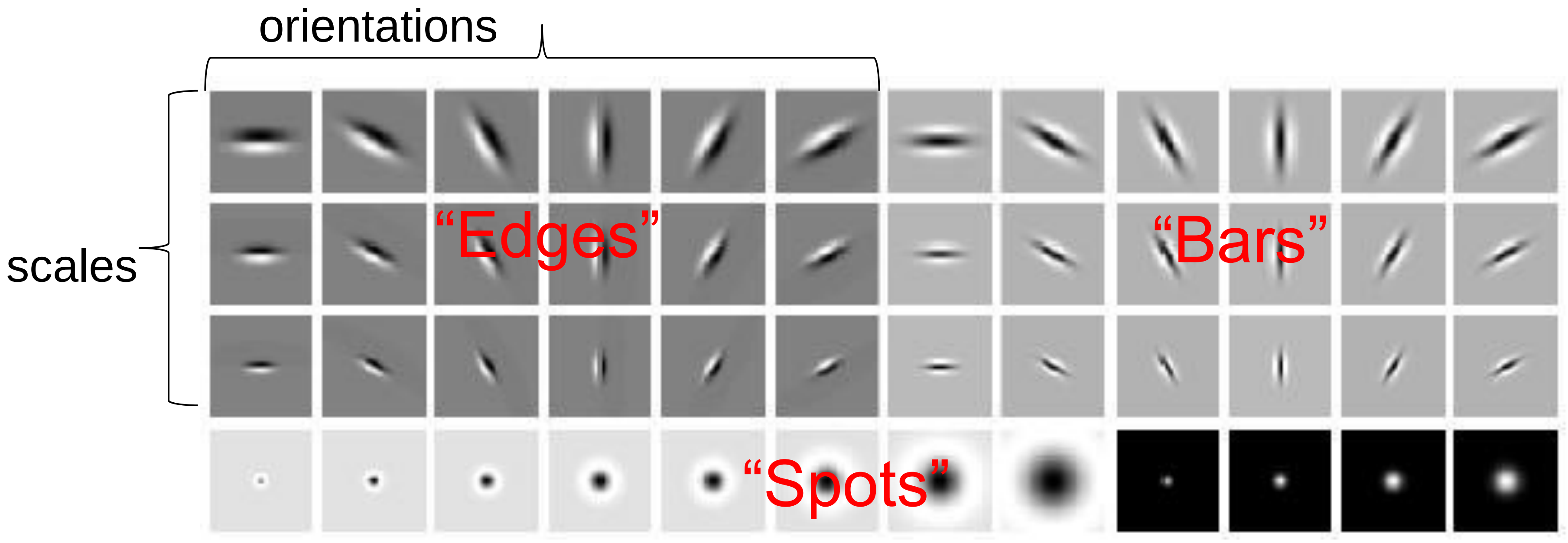
What is texture?

Regular or stochastic patterns caused by bumps, grooves, and/or markings

How can we represent texture?

- Compute responses of blobs and edges at various orientations and scales

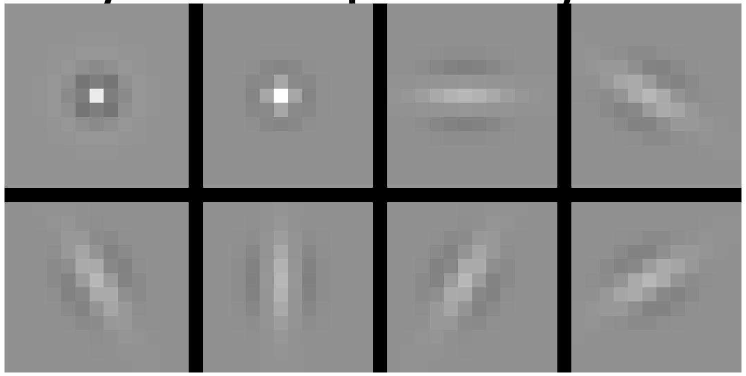
Overcomplete representation: filter banks

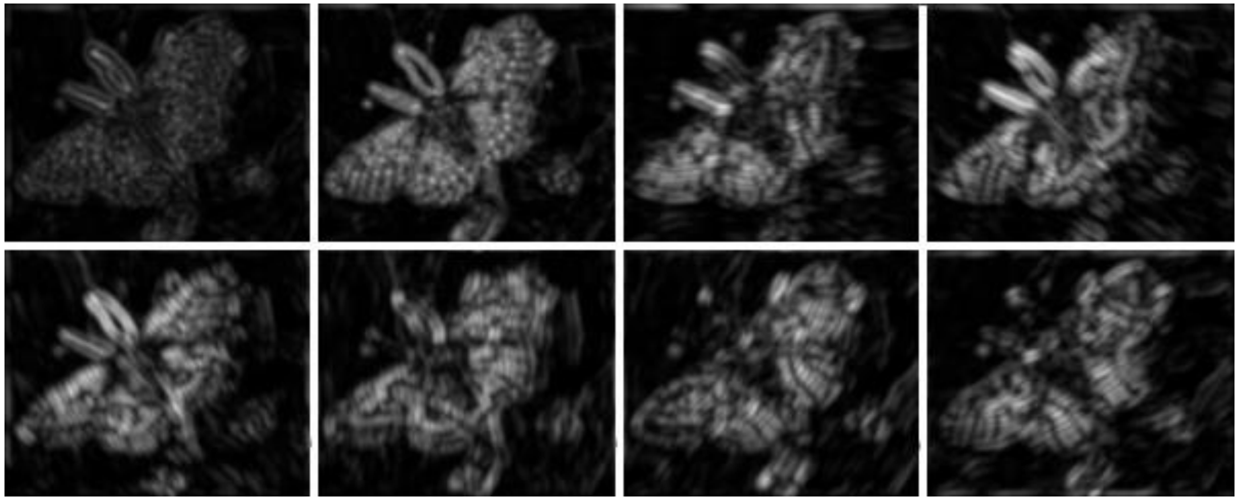


Code for filter banks: *www.robots.ox.ac.uk/~vgg/research/texclass/filters.html*

Filter banks

- Process image with each filter and keep responses (or squared/abs responses)





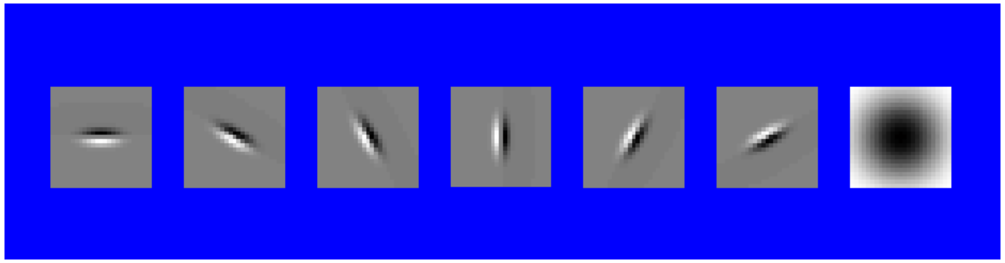
How can we represent texture?

- Measure responses of blobs and edges at various orientations and scales

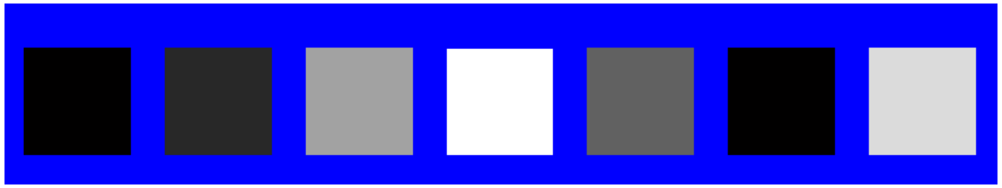
- Idea 1: Record simple statistics (e.g., mean, std.) of absolute filter responses

Can you match the texture to the response?

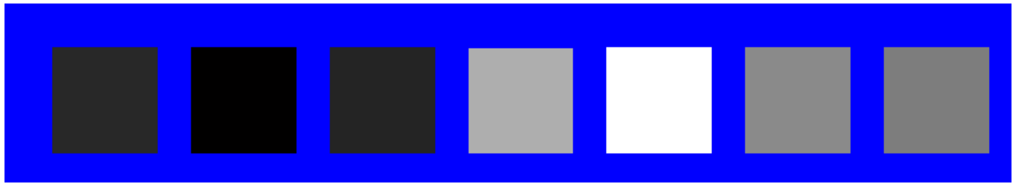
Filters



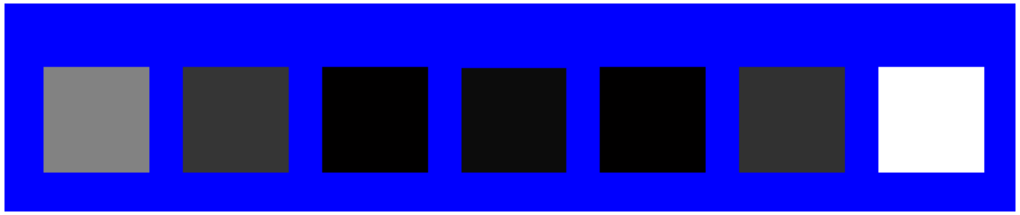
1



2

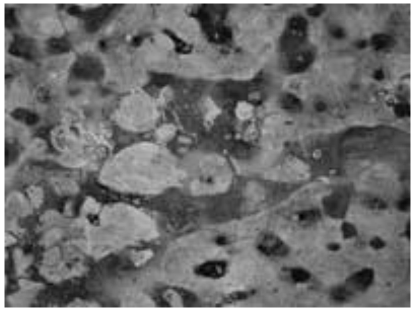


3

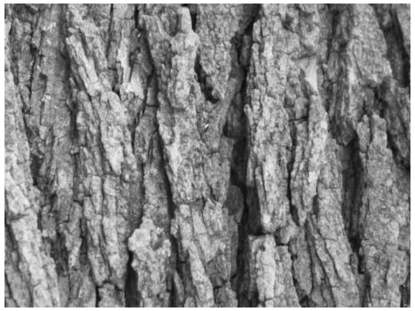


Mean abs responses

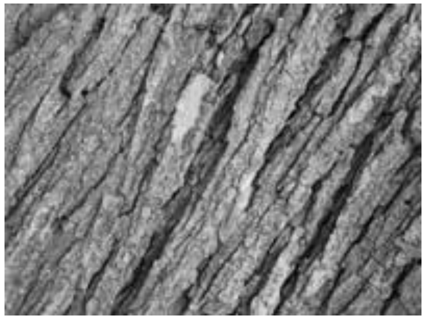
A



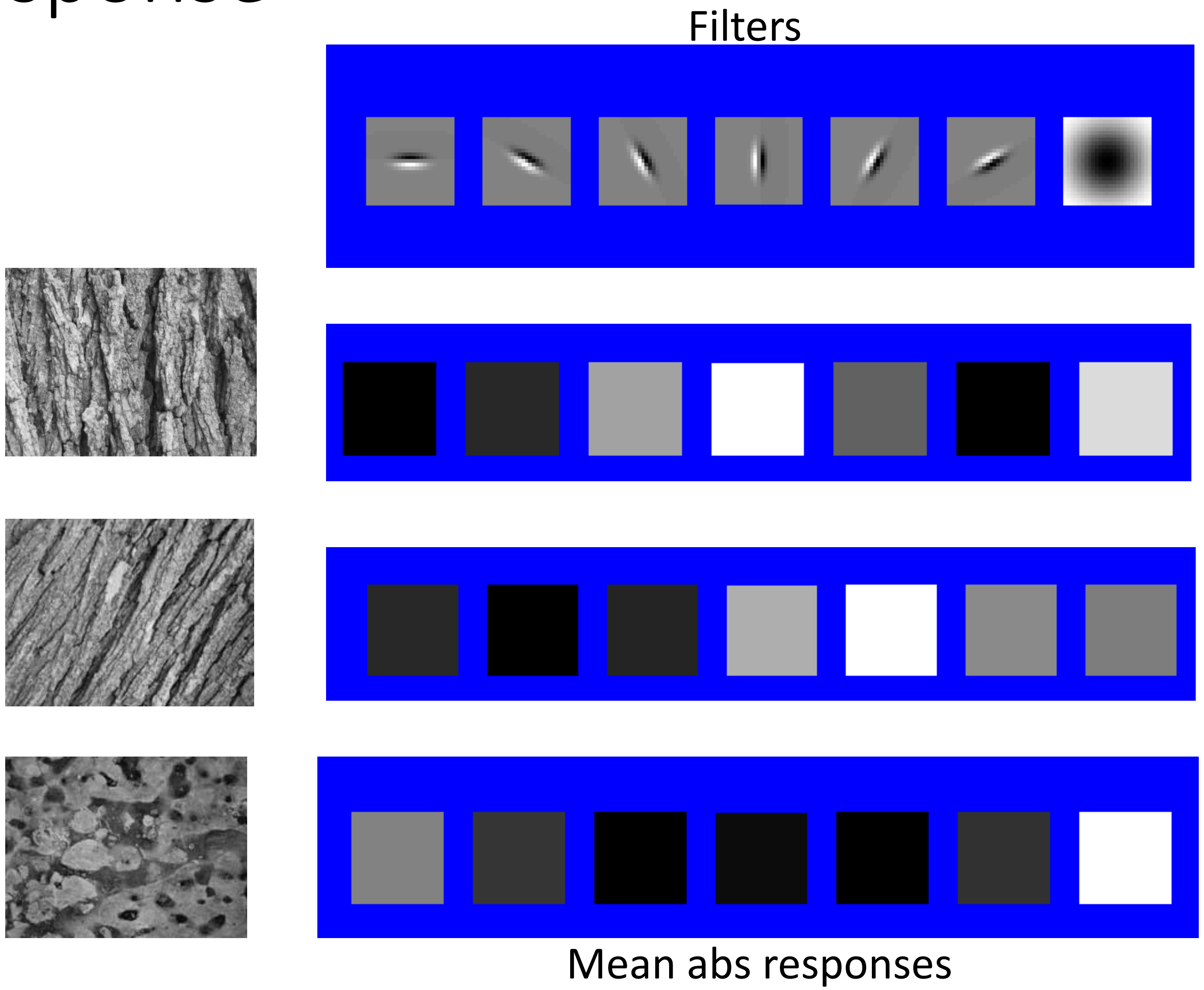
B



C

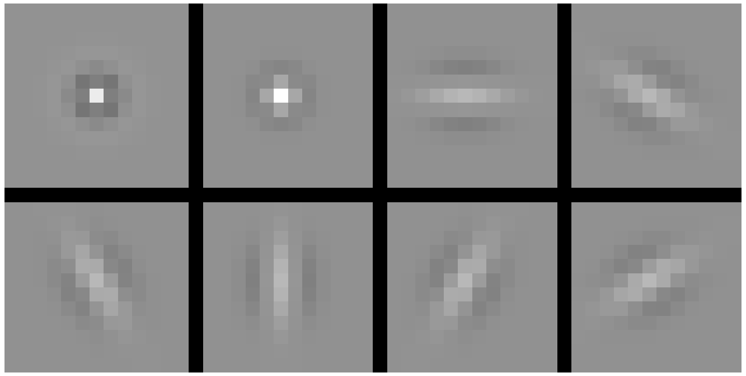


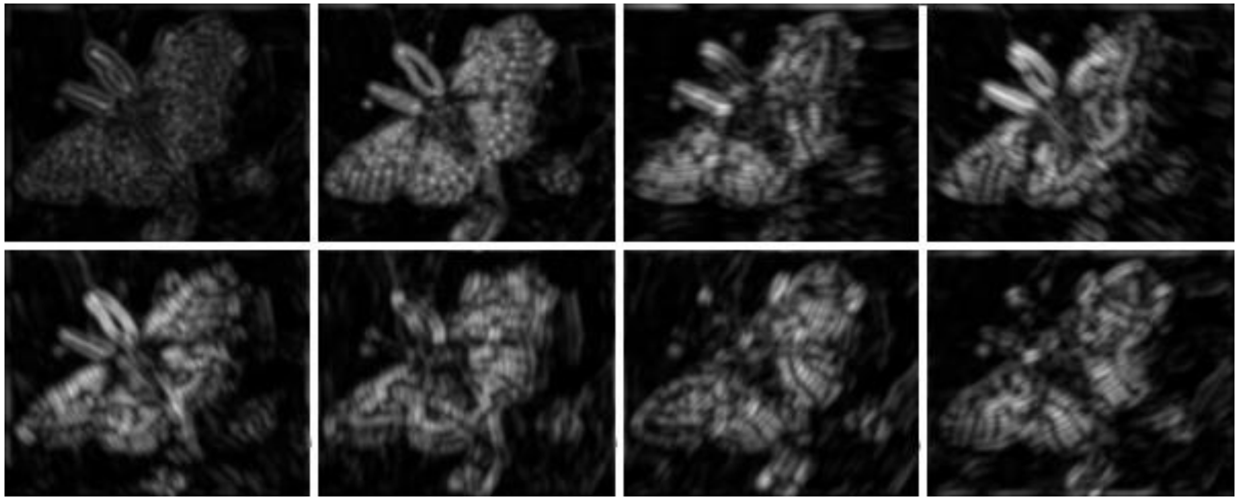
Representing texture by mean abs response



Representing texture

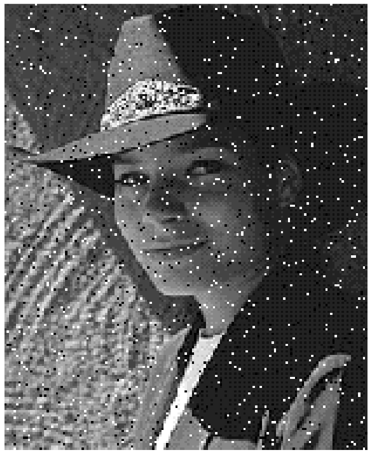
- Idea 2: take vectors of filter responses at each pixel and cluster them, then take histograms (more on this in coming weeks)

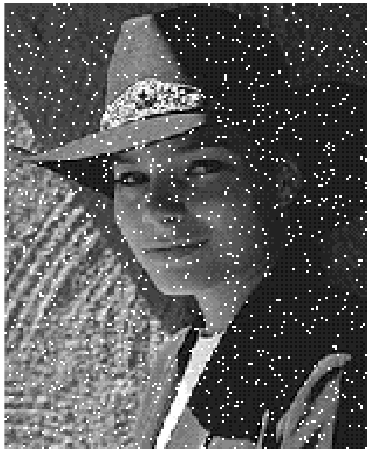


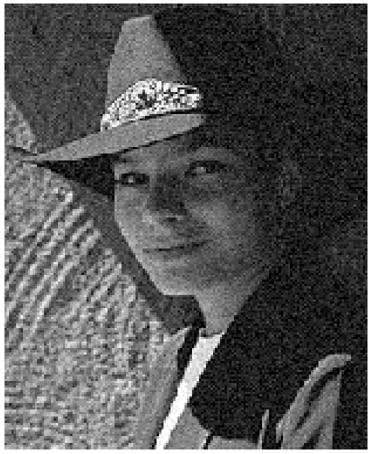


Denoising and Nonlinear Image Filtering

Original


Salt and pepper noise


Impulse noise

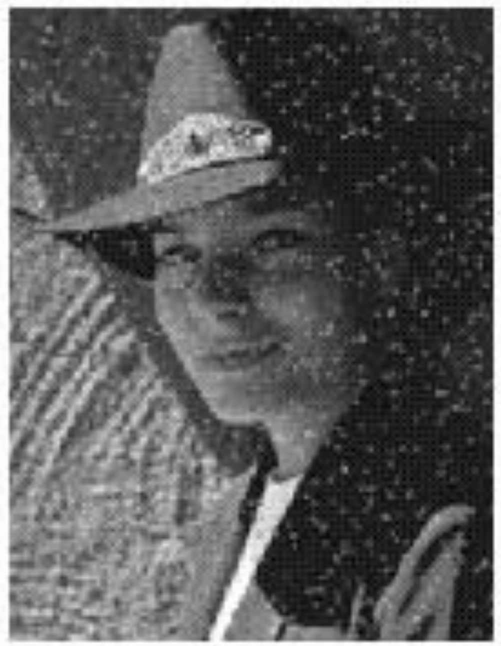

Gaussian noise

- **Salt and pepper noise**: contains random occurrences of black and white pixels
- **Impulse noise:** contains random occurrences of white pixels
- **Gaussian noise**: variations in intensity drawn from a Gaussian normal distribution

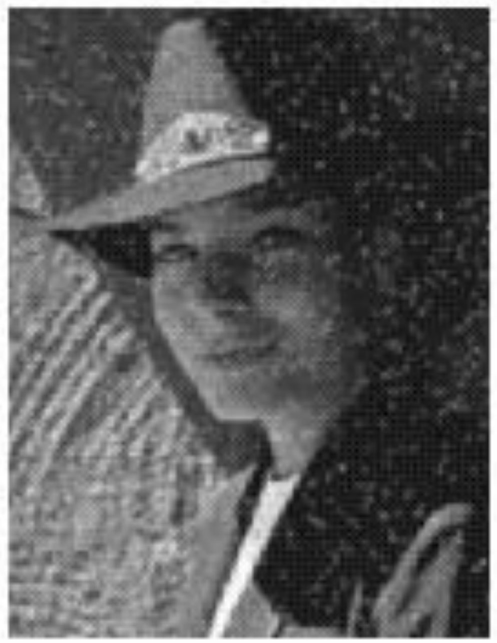
S

Reducing salt-and-pepper noise

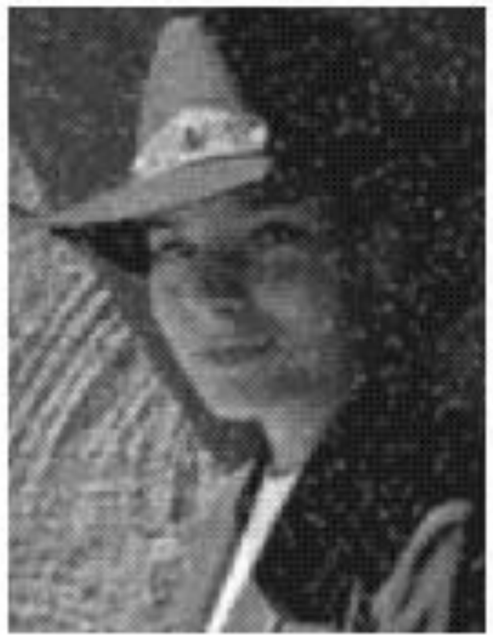
3x3



5x5



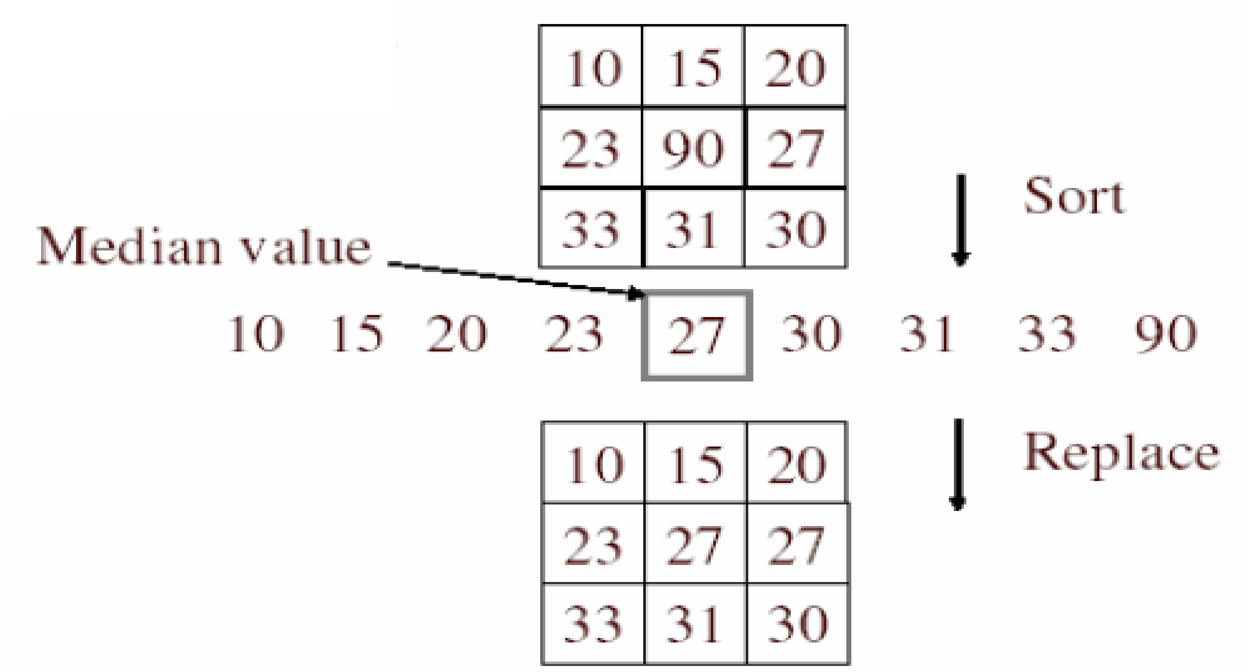
7x7



- What’s wrong with Gaussian filtering?

Alternative idea: Median filtering

- A **median filter** operates over a window by selecting the median intensity in the window



- Is median filtering linear?

Source: K. Grauman

Median filter

- Is median filtering linear?
- Let’s try filtering

$$\begin{array}{ccc} A & & B \\ \begin{bmatrix}1&1&1\\1&1&2\\2&2&2\end{bmatrix} & + & \begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}\end{array}$$

Median(A) = 1 Median(B) = 0

Median(A+B) = 2

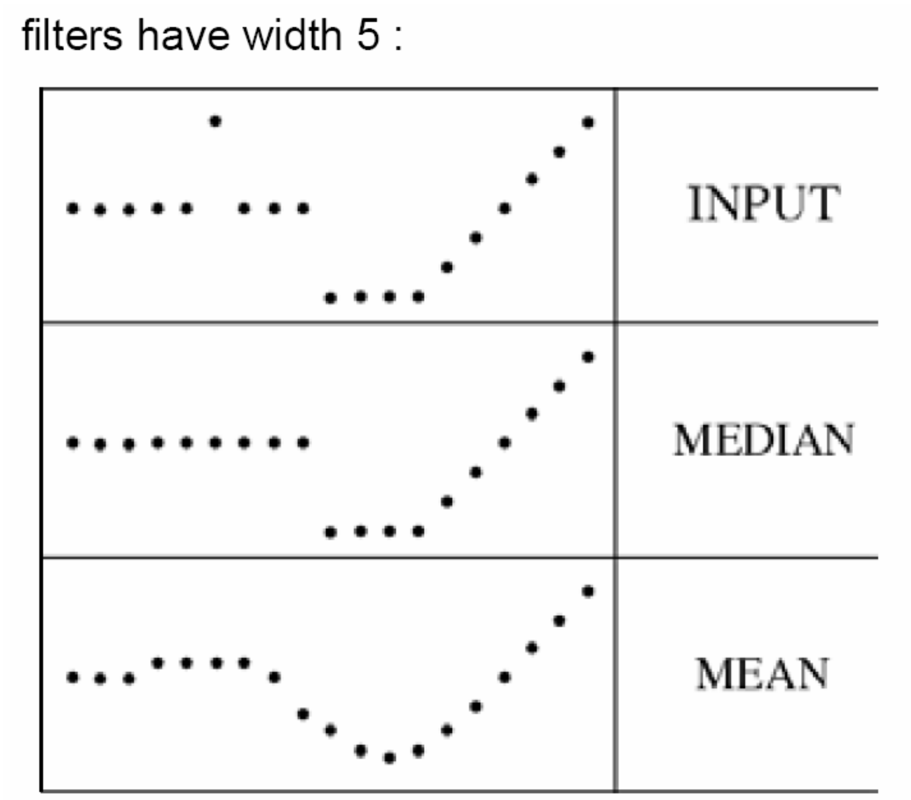
Violate **linearity**

```
filter(f1 + f2) = filter(f1) + filter(f2)
```

$\texttt{filter}(f_1 + f_2) = \texttt{filter}(f_1) + \texttt{filter}(f_2)$

Median filter

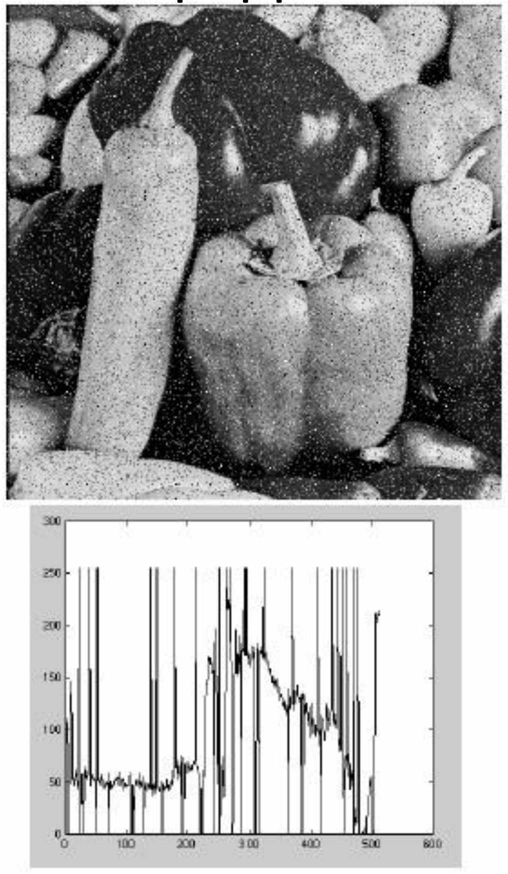
- What advantage does median filtering have over Gaussian filtering?
  - Robustness to outliers



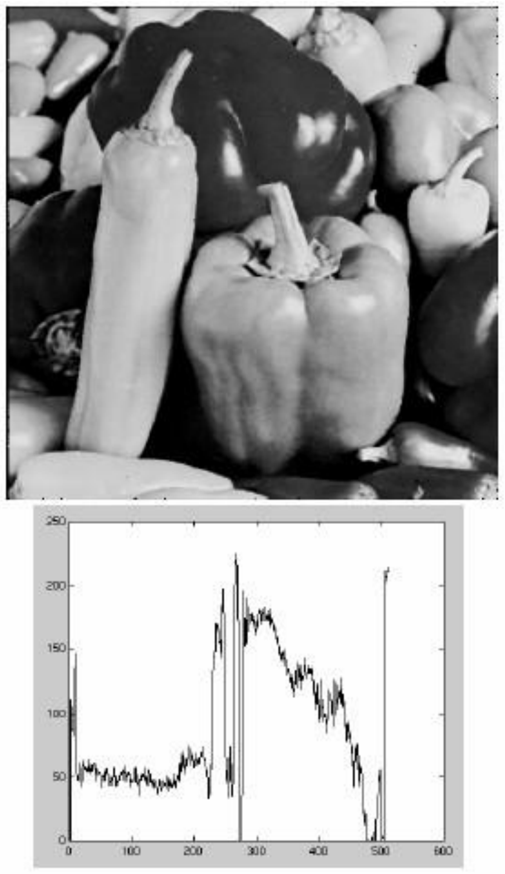
Source: K. Grauman

Median filter

Salt-and-pepper noise



Median filtered



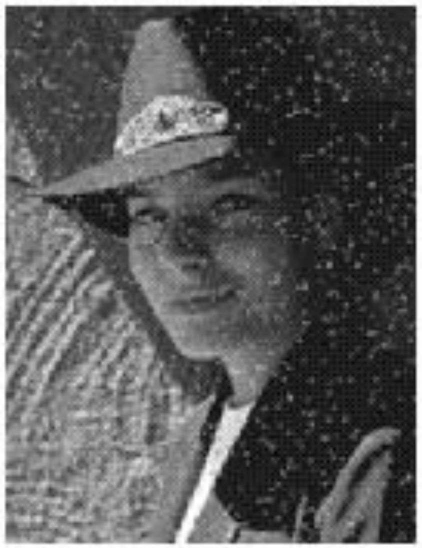
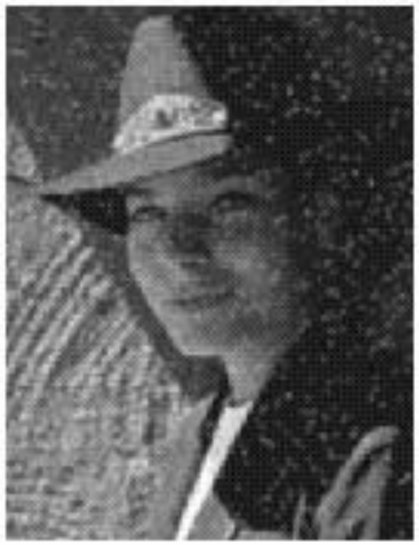
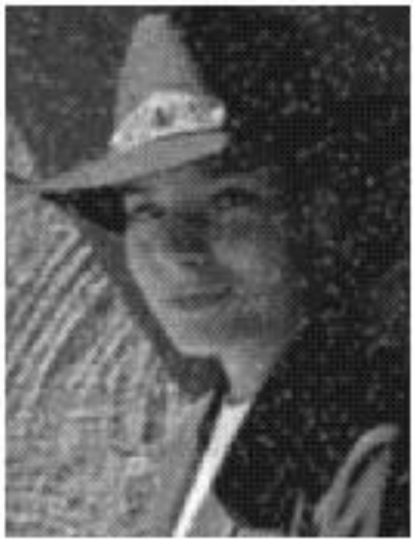
- MATLAB: medfilt2(image, [h w])

Source: M. Hebert

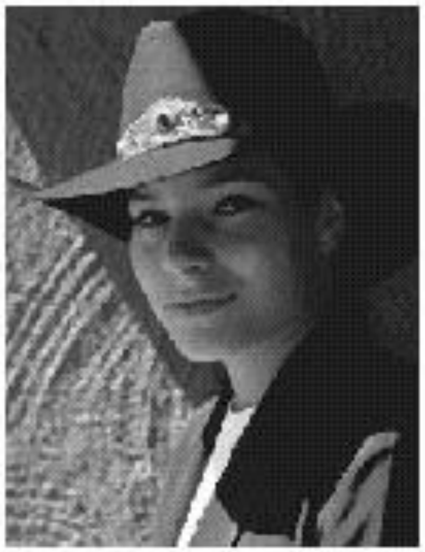
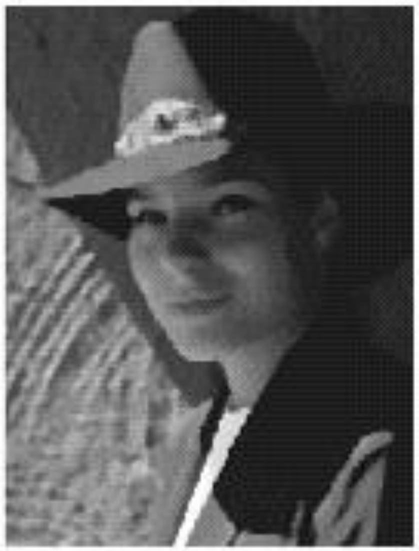
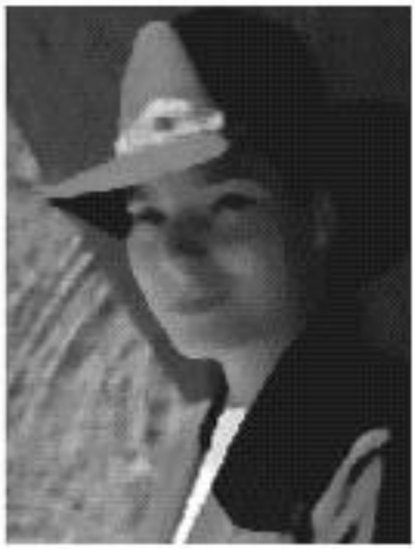
Gaussian vs. median filtering

3x3 5x5 7x7

Gaussian

Median

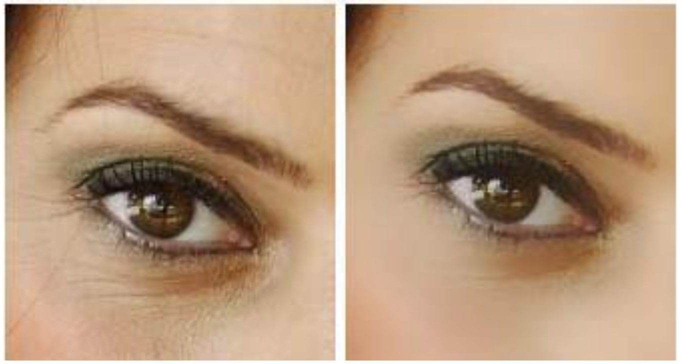




Other non-linear filters

- Weighted median (pixels further from center count less)

- Clipped mean (average, ignoring few brightest and darkest pixels)

- Bilateral filtering (weight by spatial distance *and* intensity difference)

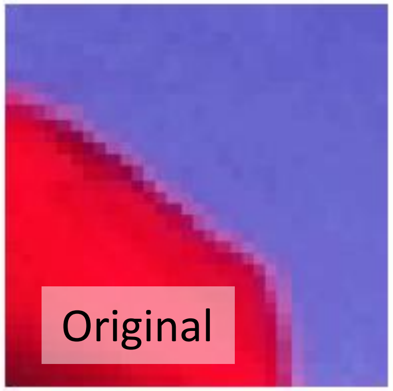


Bilateral filtering

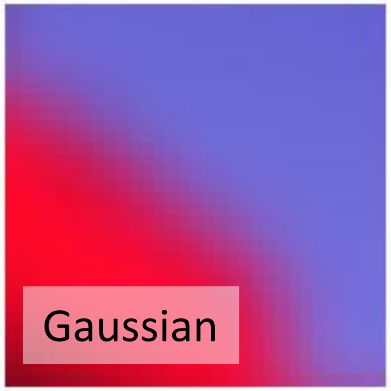
Image: http://vision.ai.uiuc.edu/?p=1455

Bilateral Filters

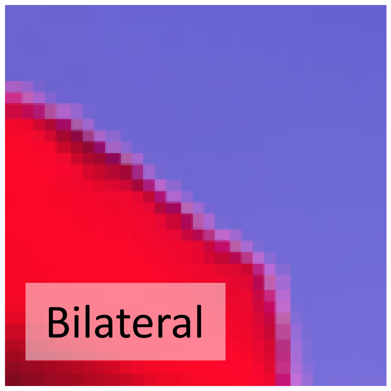
- Edge preserving: weights similar pixels more



(a)



(b)



(c)

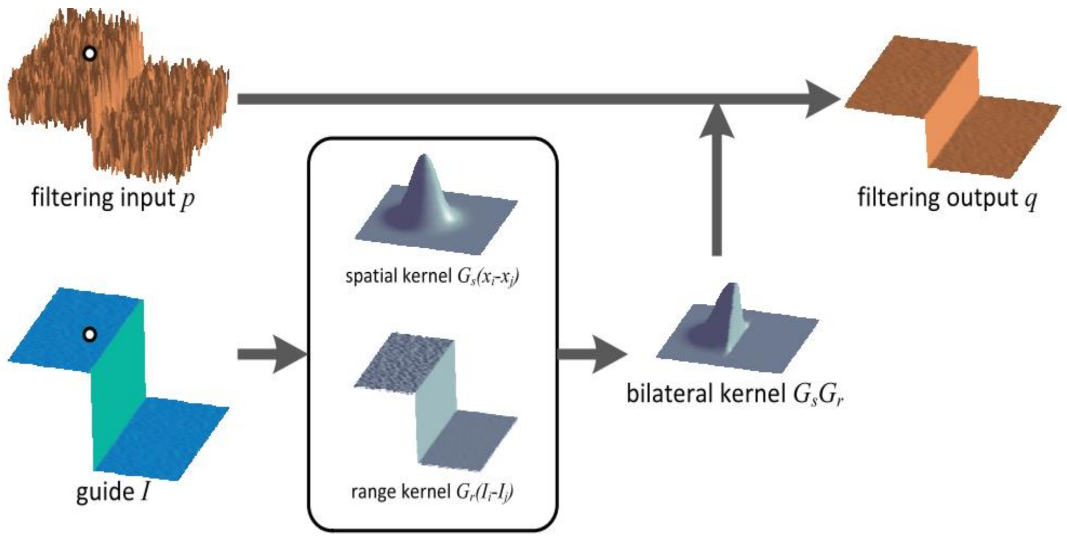
spatial similarity (e.g., intensity)

$$I^{\mathrm{b}}_{\mathbf{p}} = \frac{1}{W^{\mathrm{b}}_{\mathbf{p}}} \sum_{\mathbf{q} \in \mathcal{S}} G_{\sigma_{\mathrm{s}}}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_{\mathrm{r}}}(|I_{\mathbf{p}} - I_{\mathbf{q}}|)\, I_{\mathbf{q}}$$

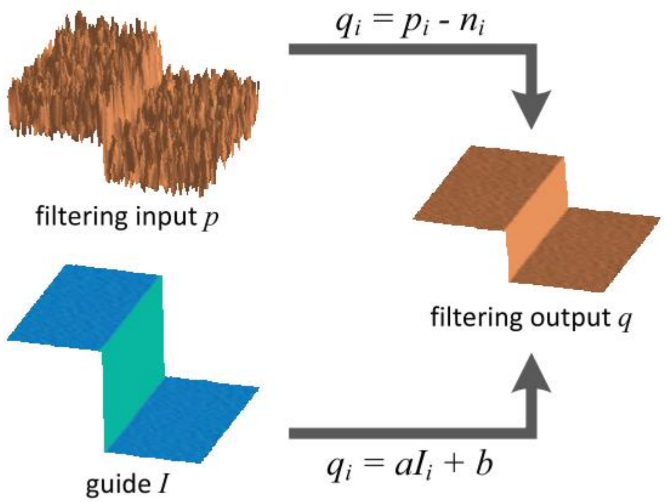
$$\text{with} \quad W^{\mathrm{b}}_{\mathbf{p}} = \sum_{\mathbf{q} \in \mathcal{S}} G_{\sigma_{\mathrm{s}}}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_{\mathrm{r}}}(|I_{\mathbf{p}} - I_{\mathbf{q}}|)$$

Carlo Tomasi, Roberto Manduchi, Bilateral Filtering for Gray and Color Images, ICCV, 1998.

Guided Image Filters


filtering input p
filtering output q
spatial kernel $G_s(x_i\text{-}x_j)$
range kernel $G_r(I_i\text{-}I_j)$
bilateral kernel G_sG_r
guide I

Bilateral filters

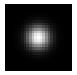


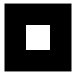
Guided filters

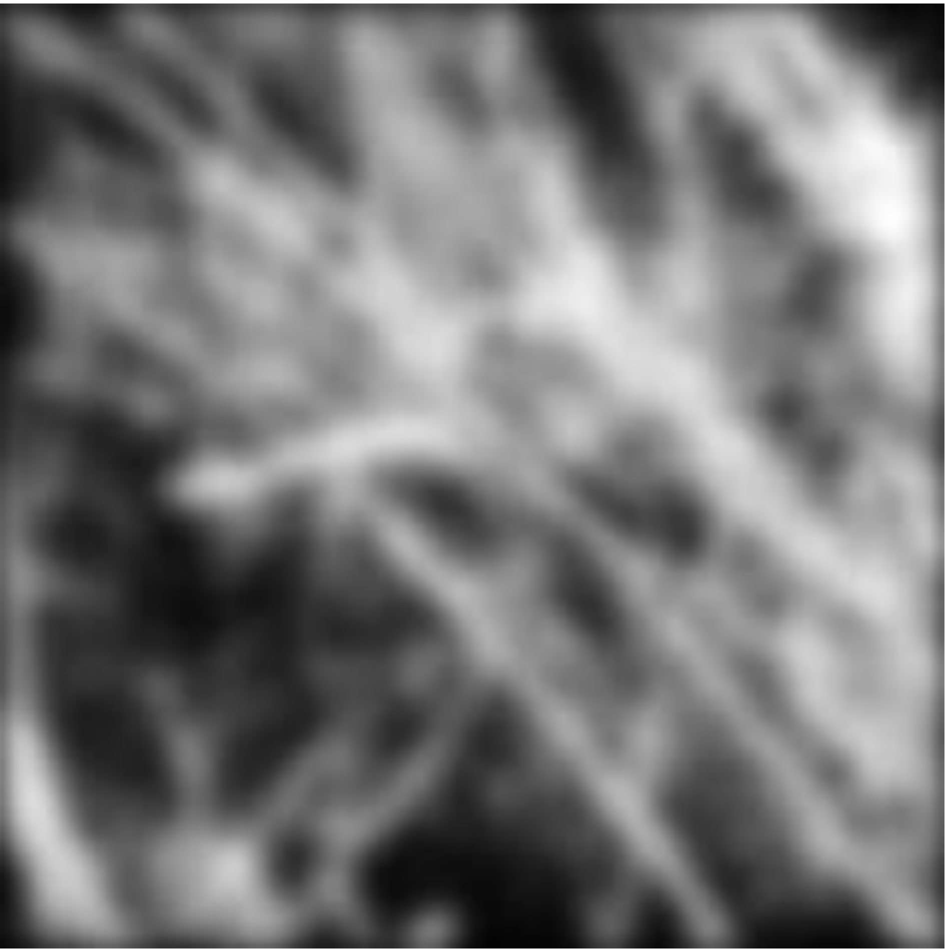
```
B = imguidedfilter(A,G);
```

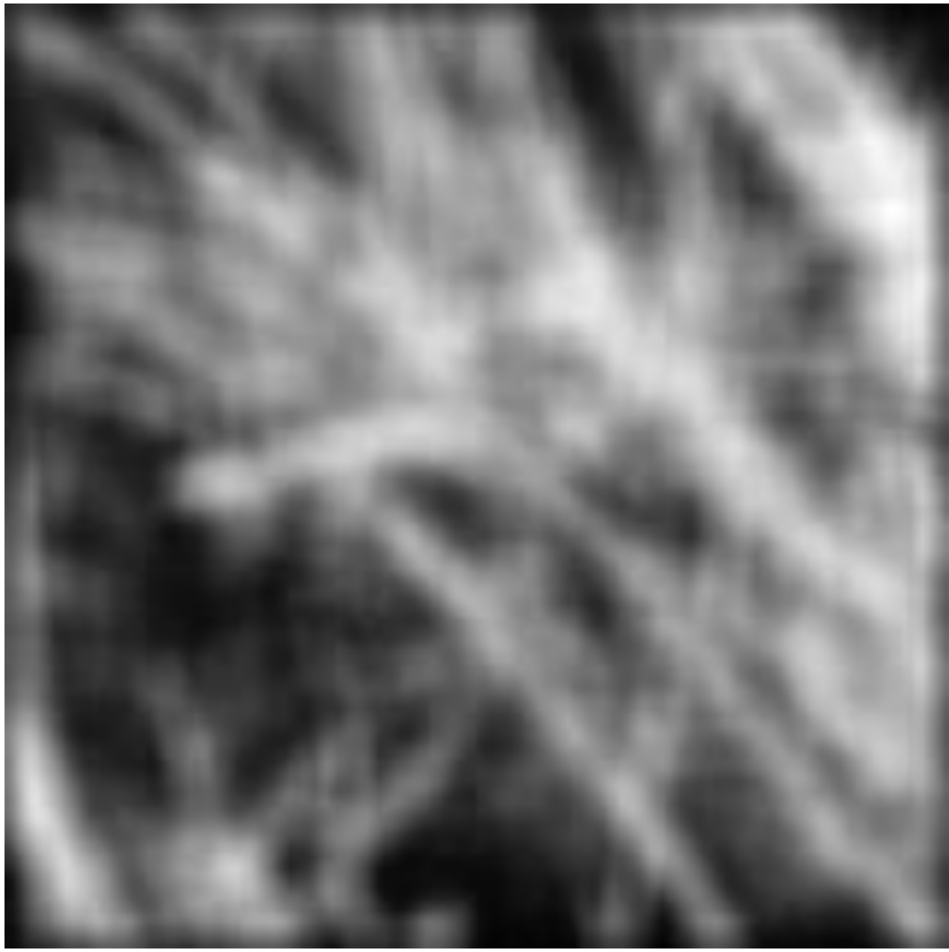
Kaiming He, Jian Sun, Xiaou Tang, Guided Image Filtering. PAMI 2013

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

Gaussian 

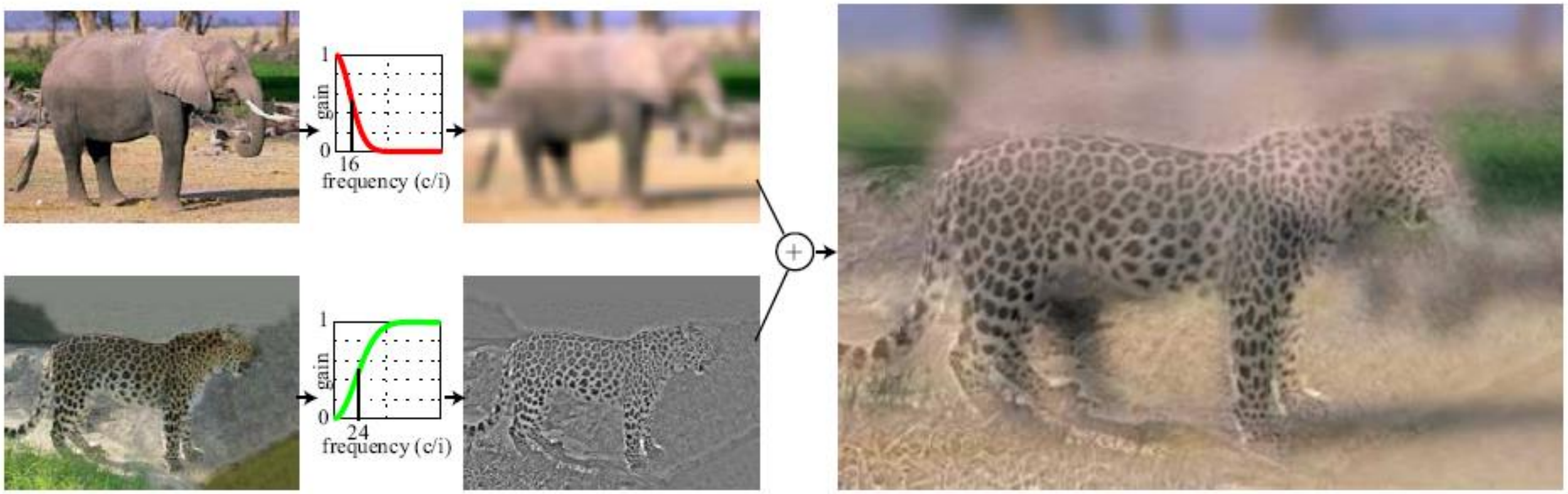
Box filter 





Slide credit: Derek Hoiem

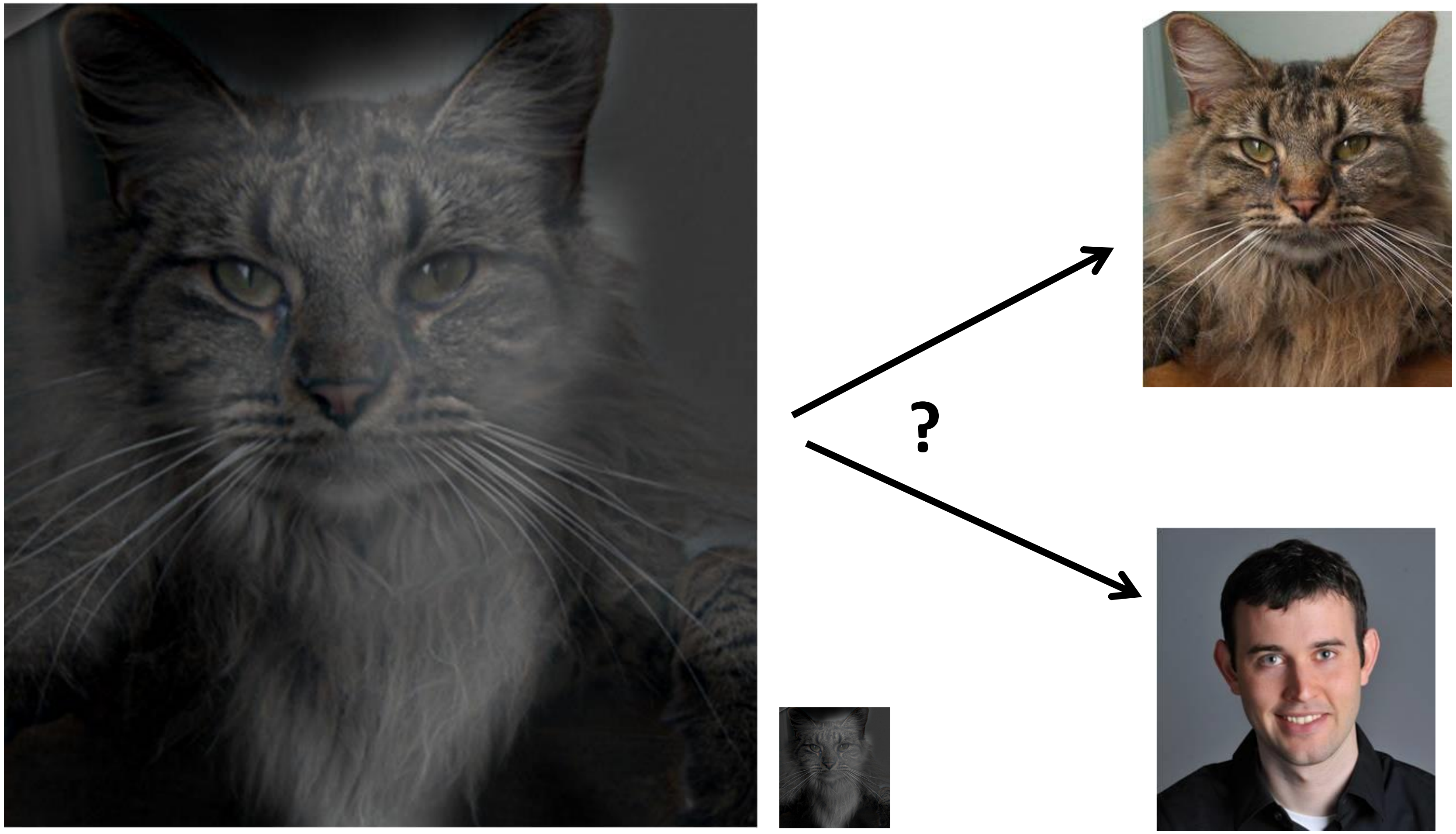
Hybrid Images



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

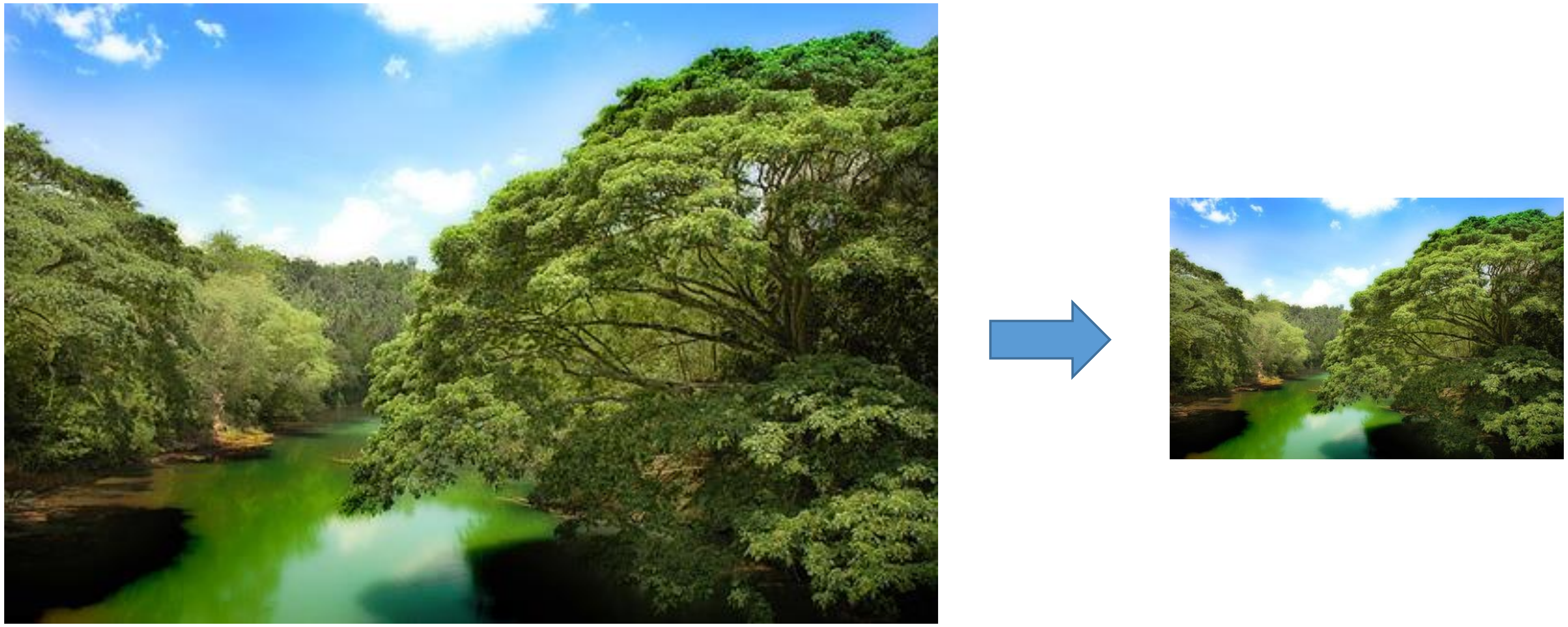
Slide credit: Derek Hoiem

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/

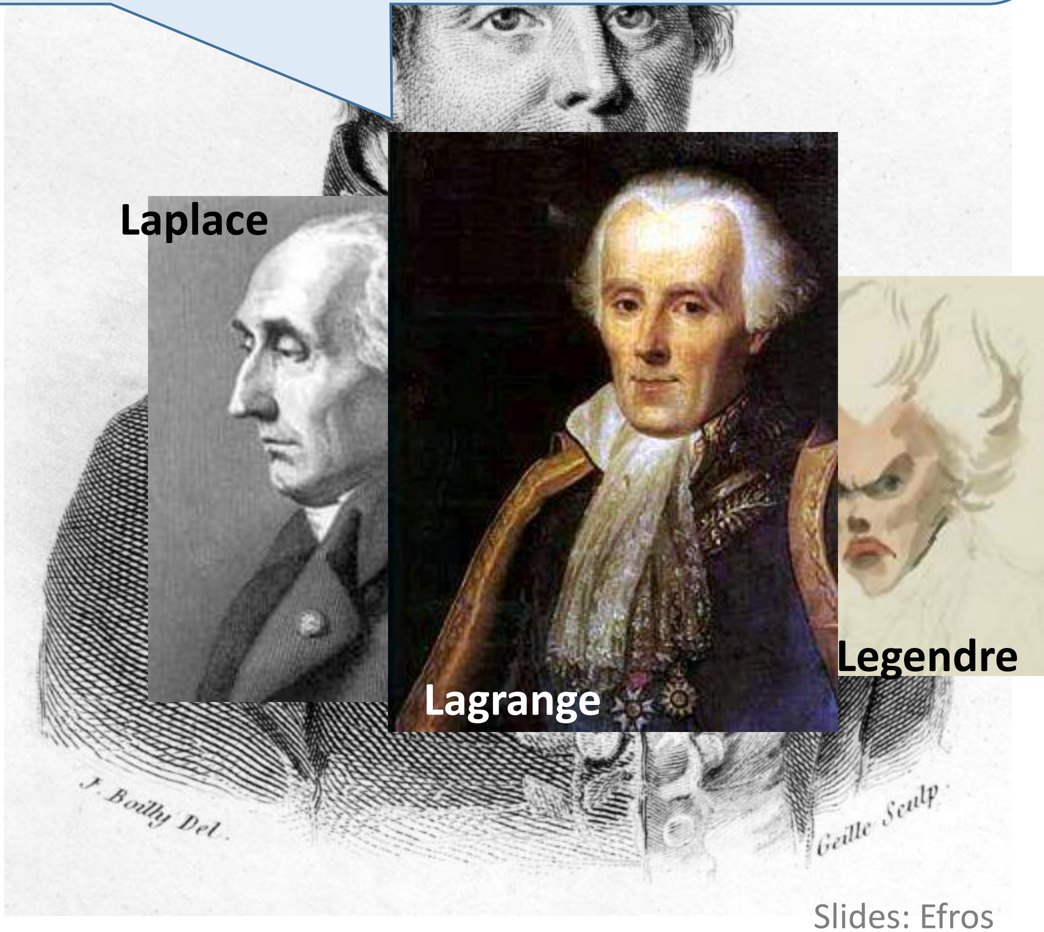
Thinking in terms of frequency

Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

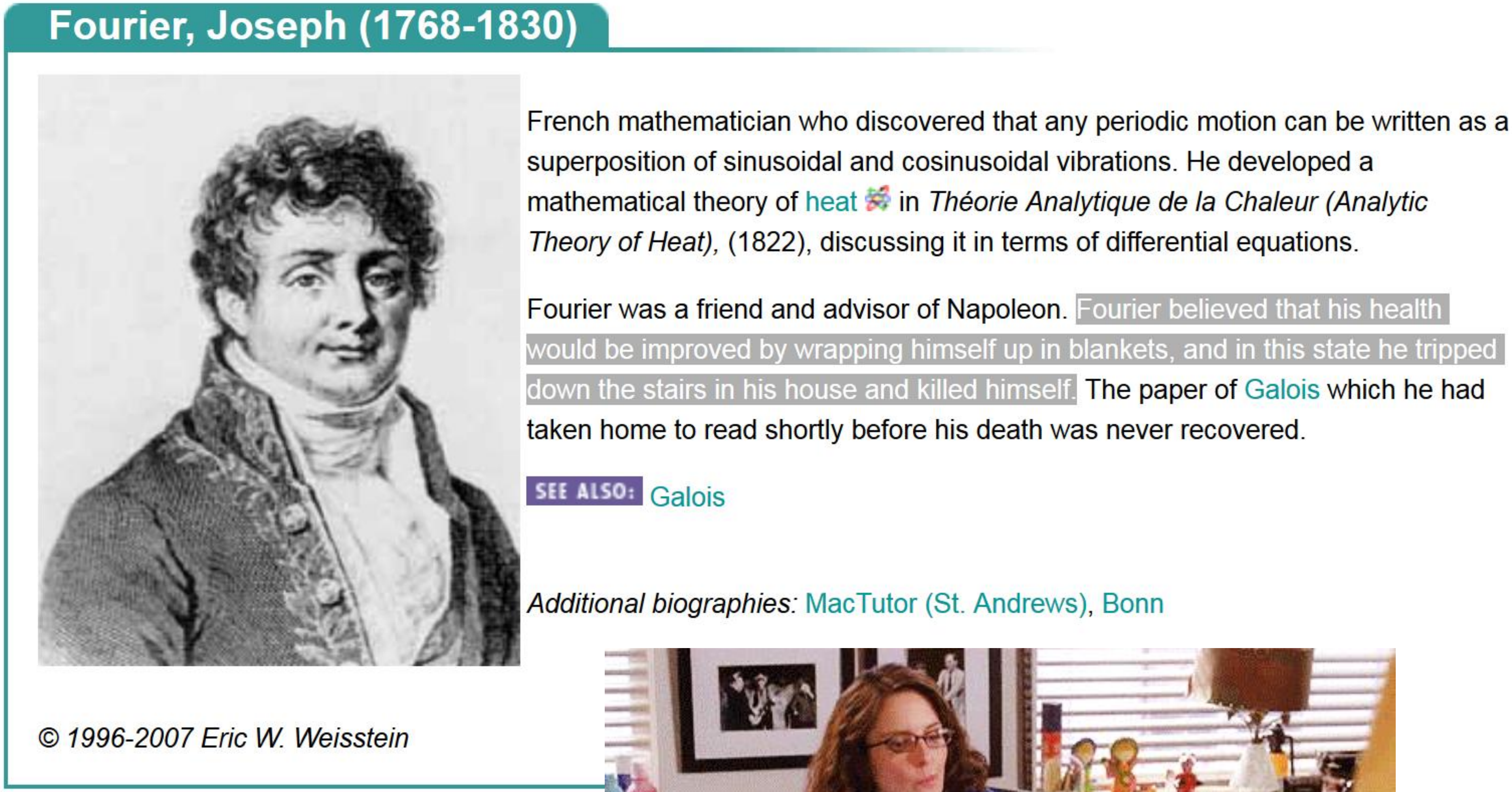
__Any__ univariate function can be rewritten as a weighted sum of sines and cosines of different frequencies.

- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it's (mostly) true!
  - called Fourier Series
  - there are some subtle restrictions



...the manner in which the author arrives at these equations is not exempt of difficulties and...his analysis to integrate them still leaves something to be desired on the score of generality and even rigour.

Slides: Efros



Fourier, Joseph (1768-1830)

French mathematician who discovered that any periodic motion can be written as a superposition of sinusoidal and cosinusoidal vibrations. He developed a mathematical theory of heat in *Théorie Analytique de la Chaleur (Analytic Theory of Heat)*, (1822), discussing it in terms of differential equations.

Fourier was a friend and advisor of Napoleon. Fourier believed that his health would be improved by wrapping himself up in blankets, and in this state he tripped down the stairs in his house and killed himself. The paper of Galois which he had taken home to read shortly before his death was never recovered.

SEE ALSO: Galois

Additional biographies: MacTutor (St. Andrews), Bonn

© 1996-2007 Eric W. Weisstein

How would math have changed if the Slanket or Snuggie had been invented?

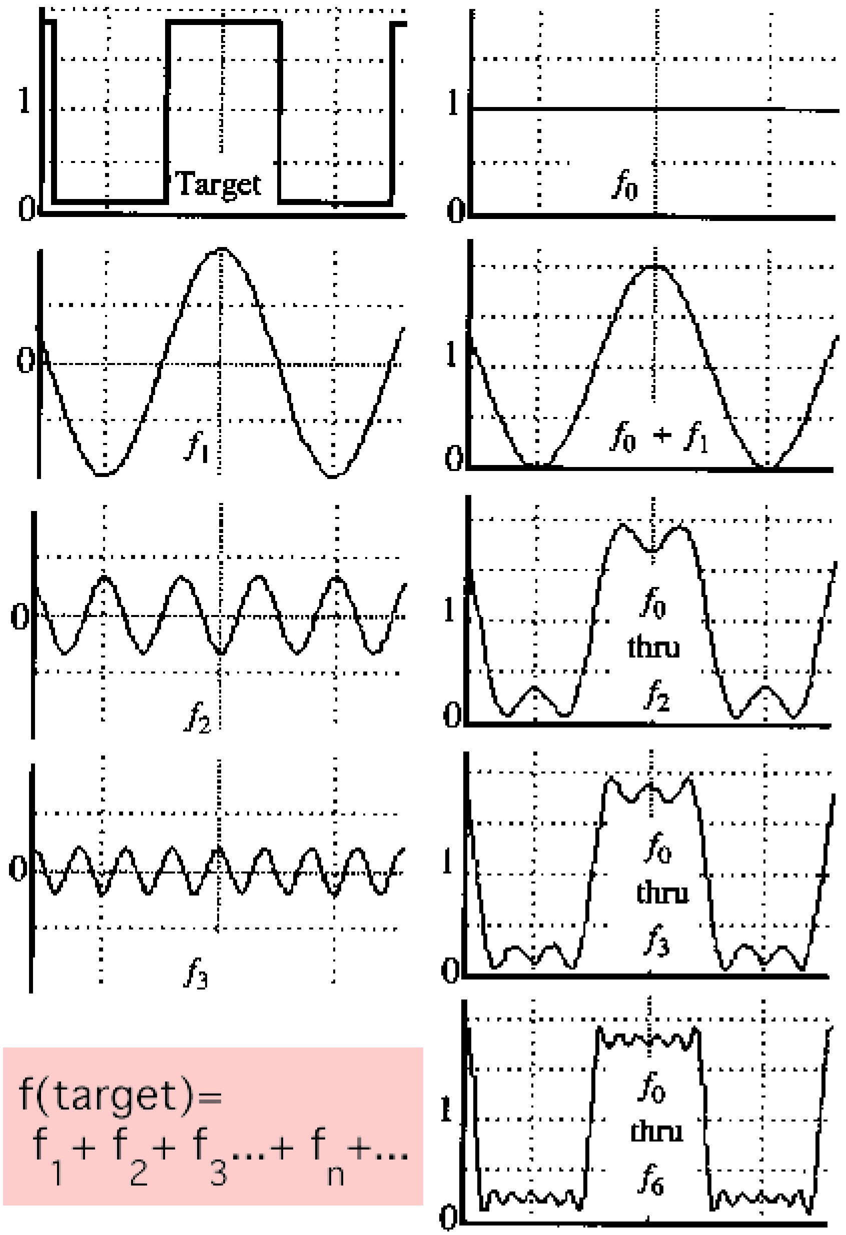
Slide credit: James Hays

A sum of sines

Our building block:

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

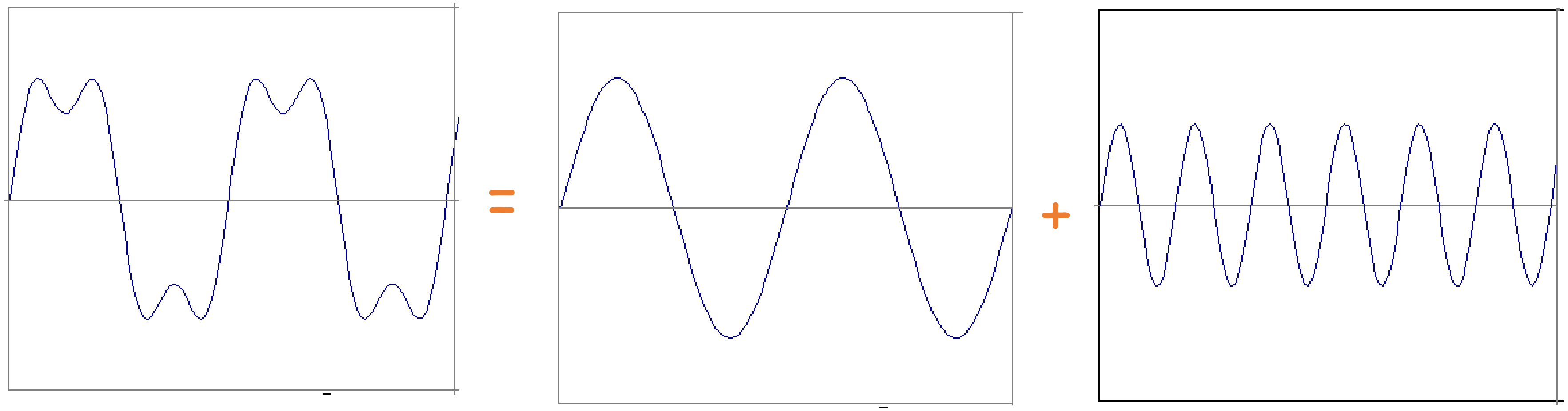
Add enough of them to get any signal *f(x)* you want!

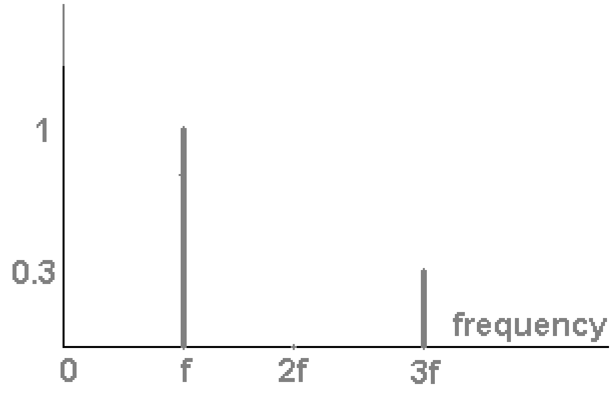


f(target)=
$f_1 + f_2 + f_3 \ldots + f_n + \ldots$

Frequency Spectra

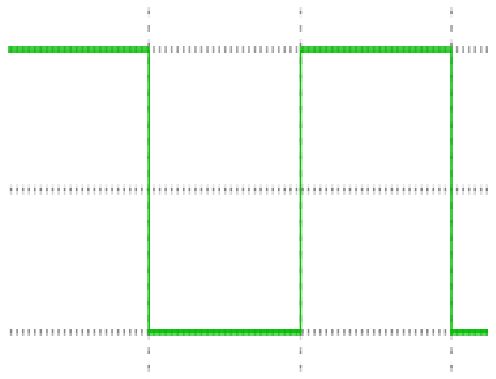
- example : $g(t) = \sin(2\pi f\, t) + (1/3)\sin(2\pi(3f)\, t)$



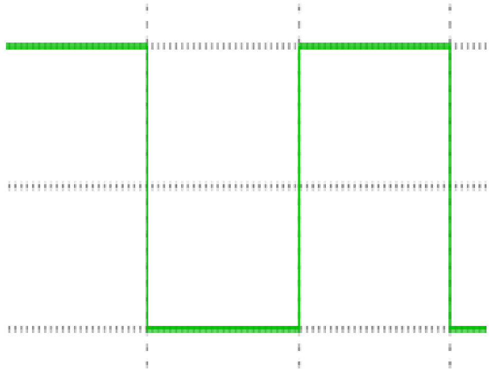


Slides: Efros

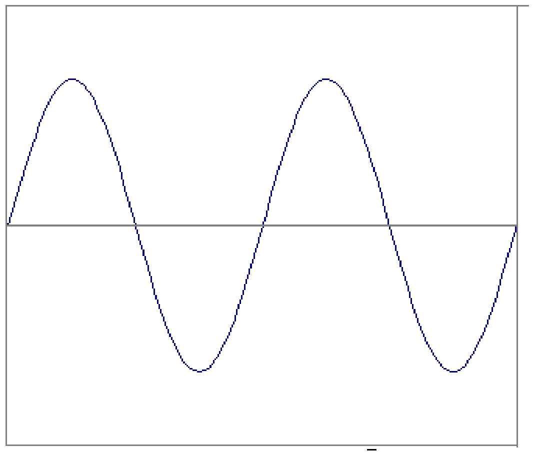
Frequency Spectra



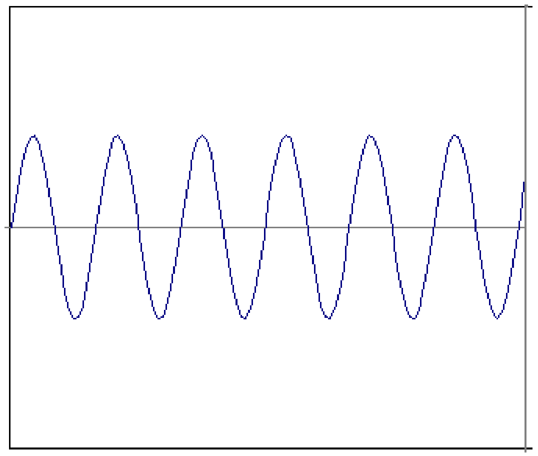
Frequency Spectra



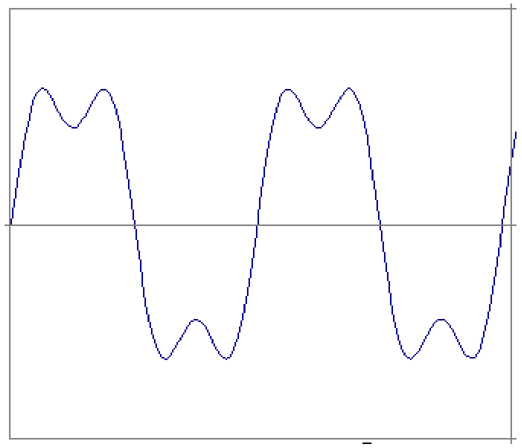
=



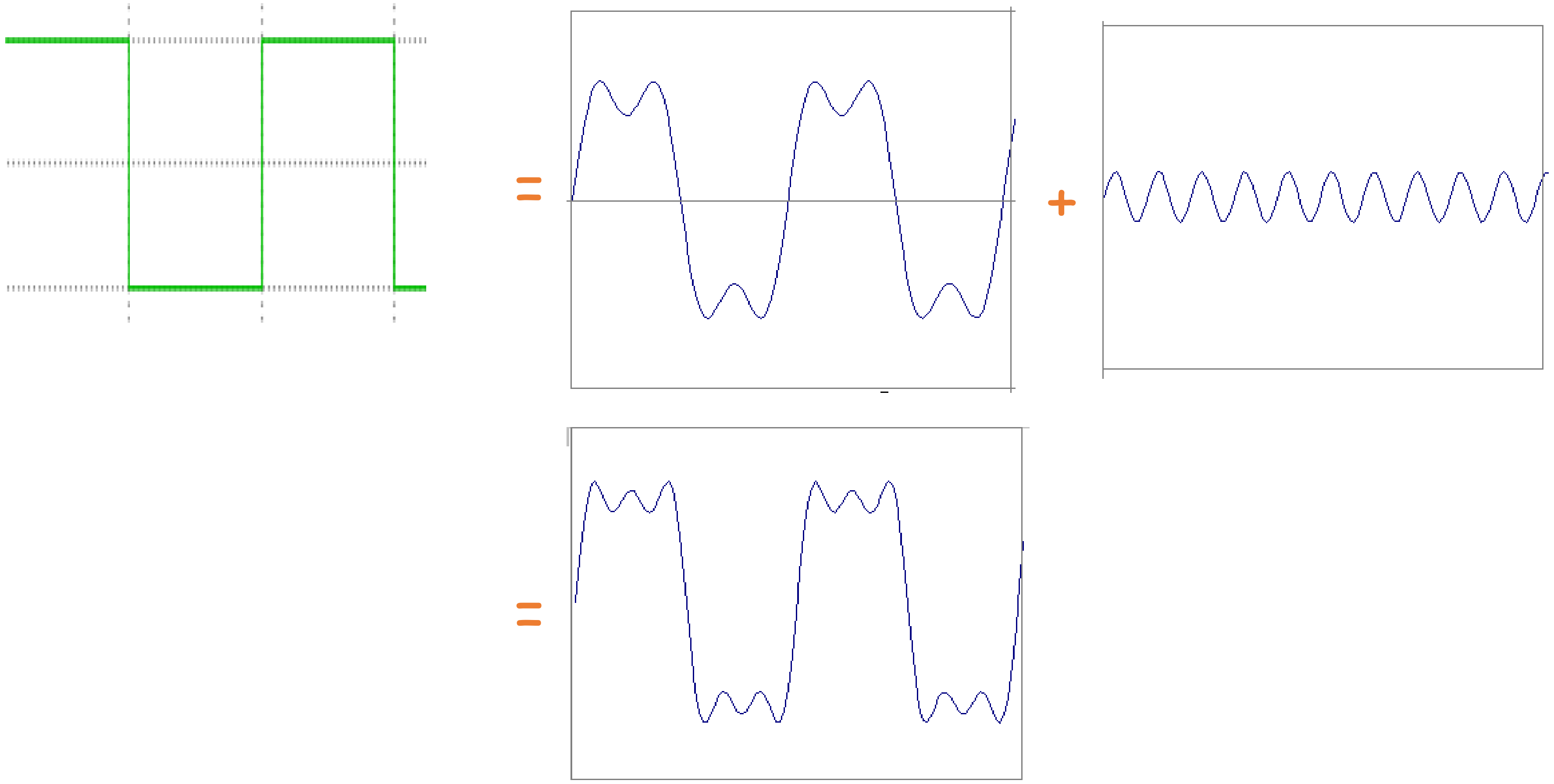
+



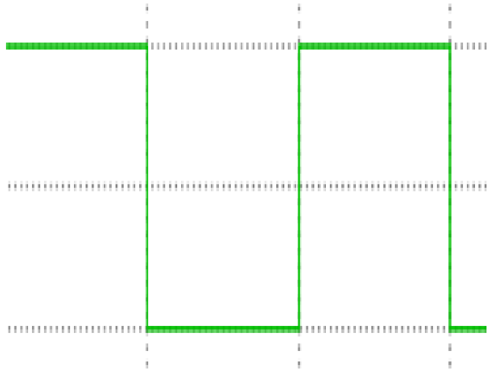
=



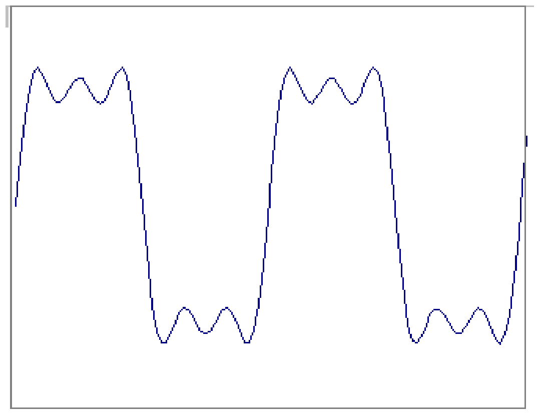
Frequency Spectra



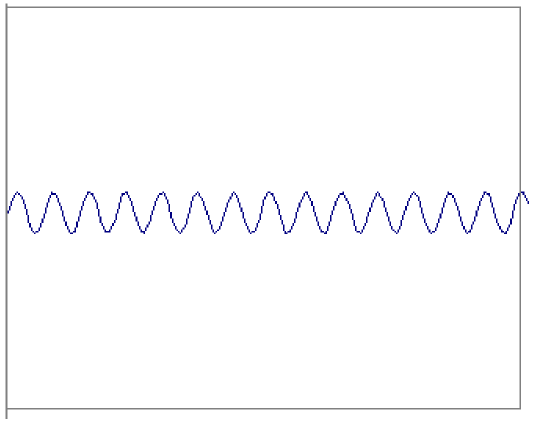
Frequency Spectra



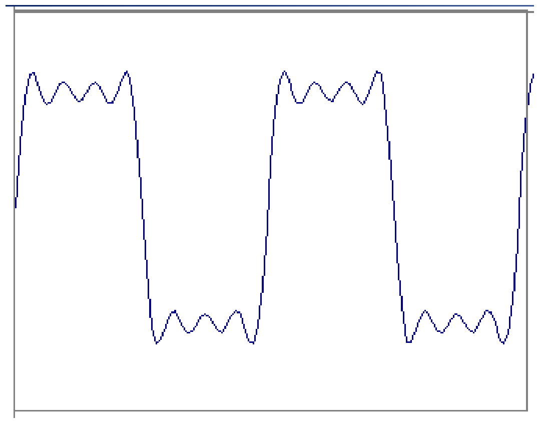
=



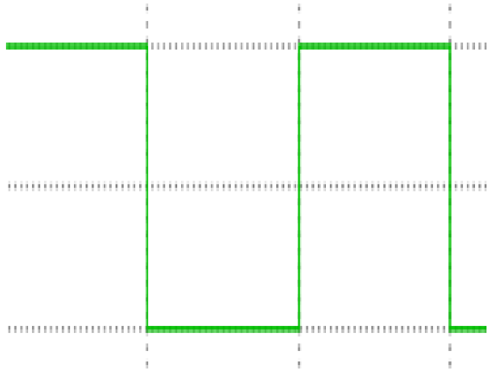
+



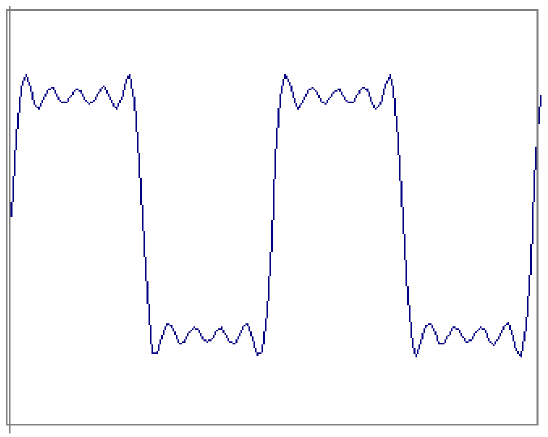
=



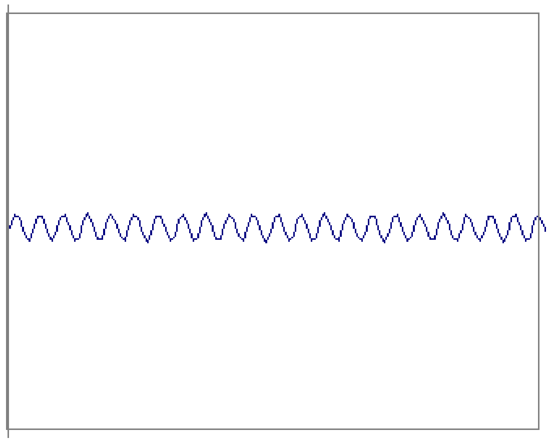
Frequency Spectra



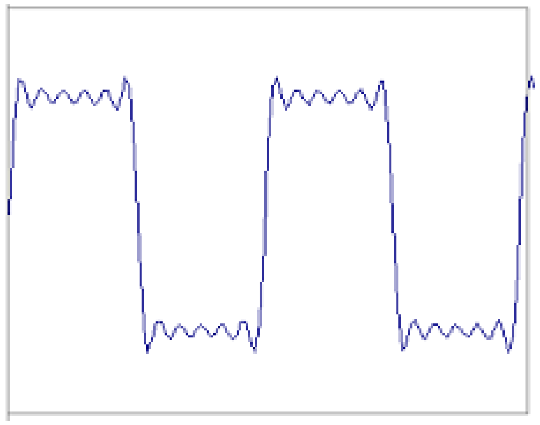
=



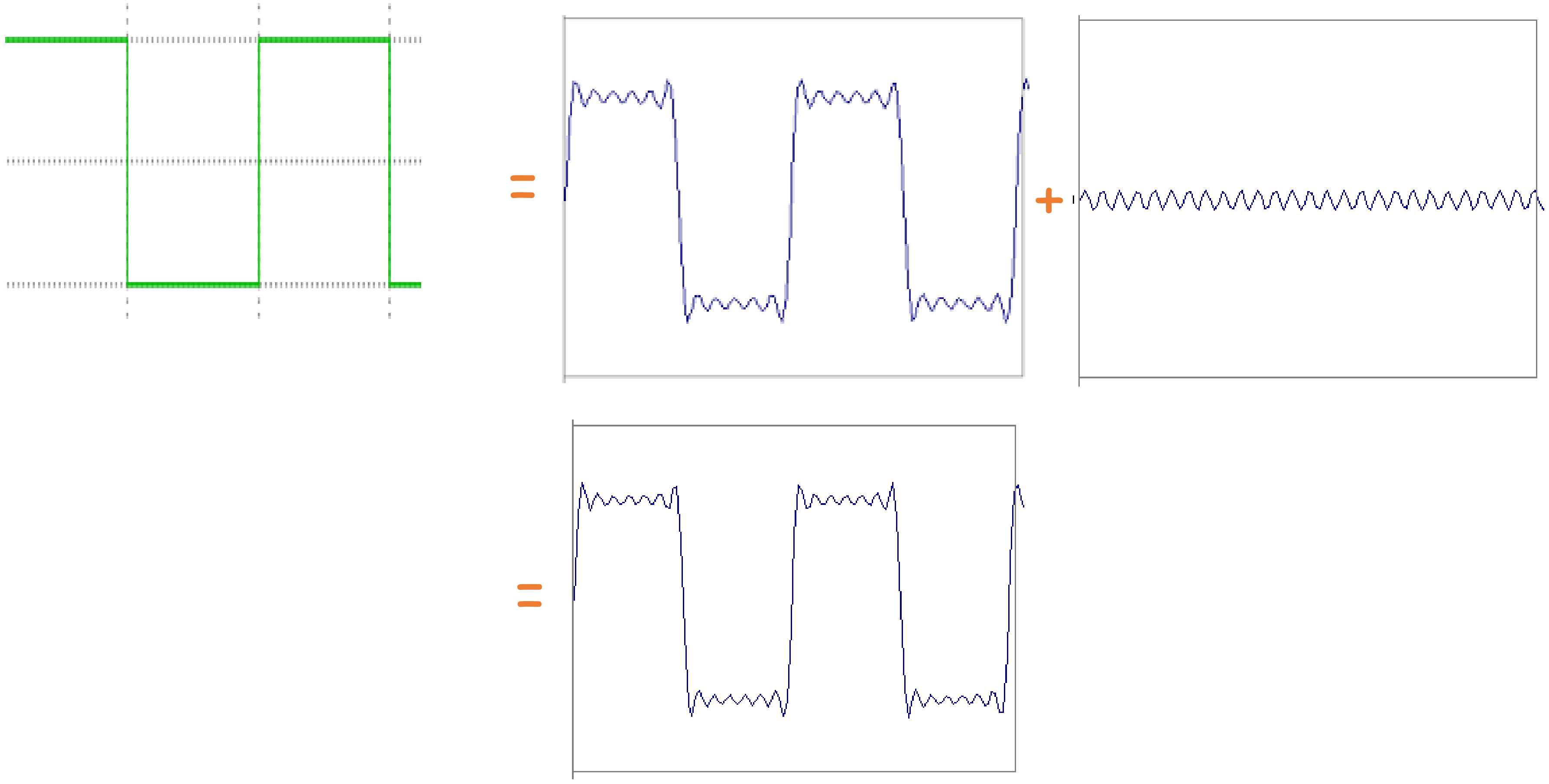
+



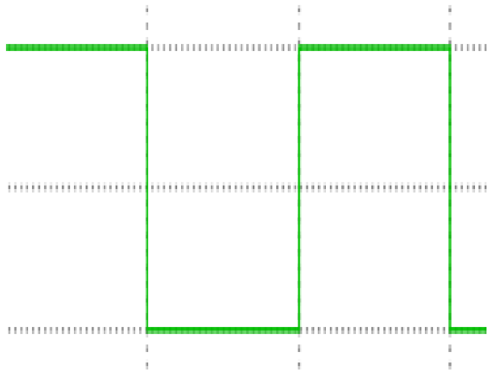
=



Frequency Spectra

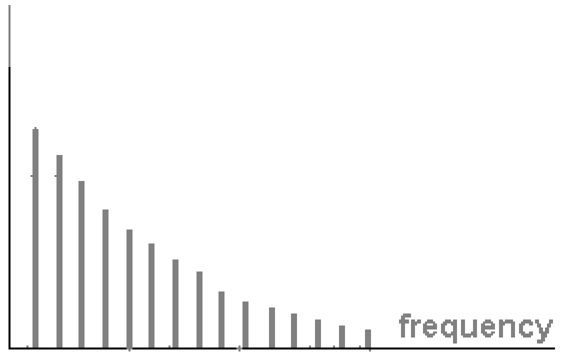


Frequency Spectra



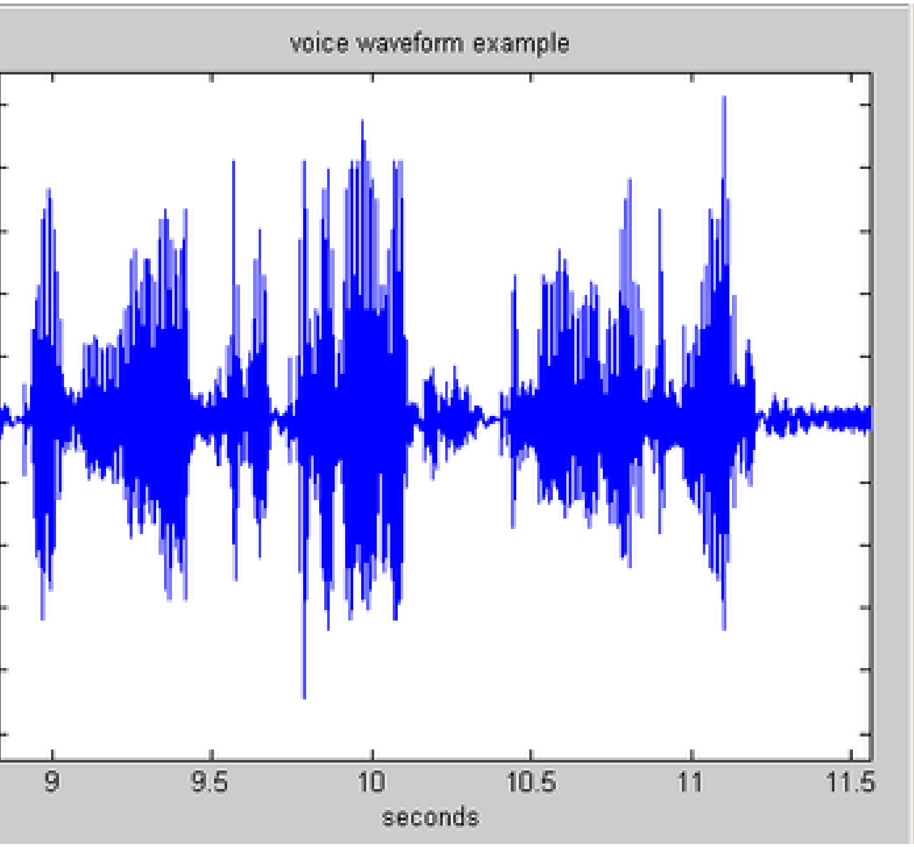
=

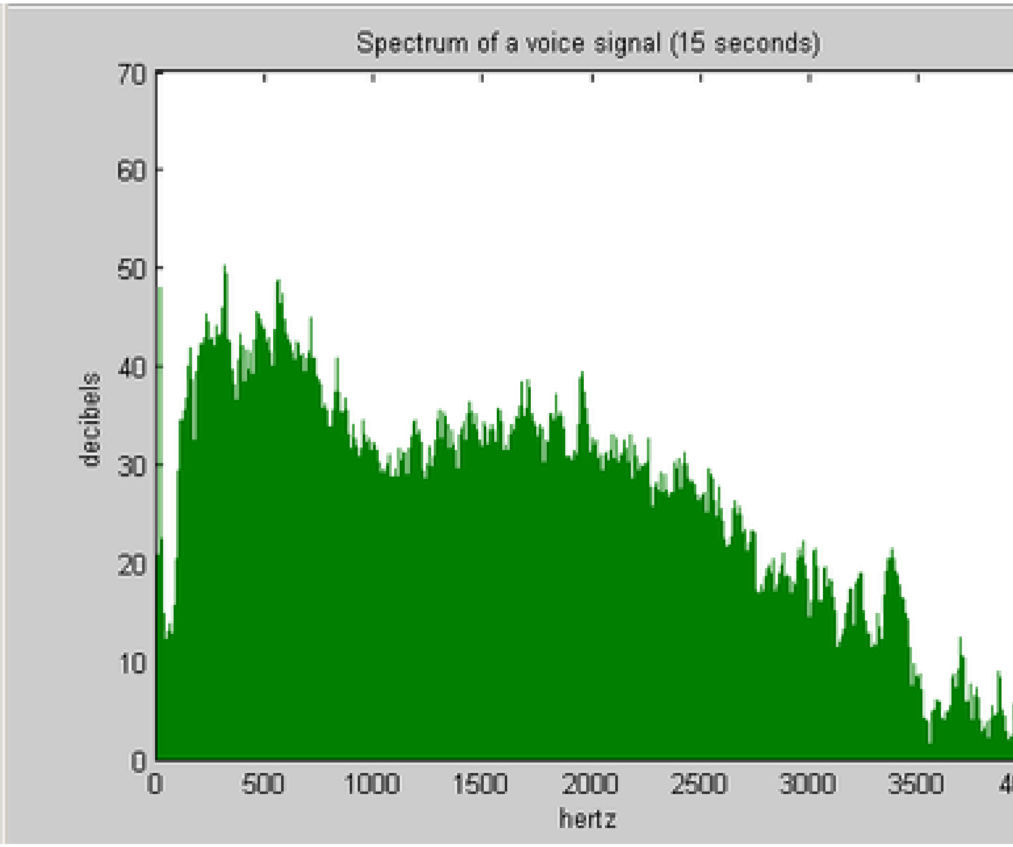
$$\boxed{A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)}$$



Example: Music

- We think of music in terms of frequencies at different magnitudes

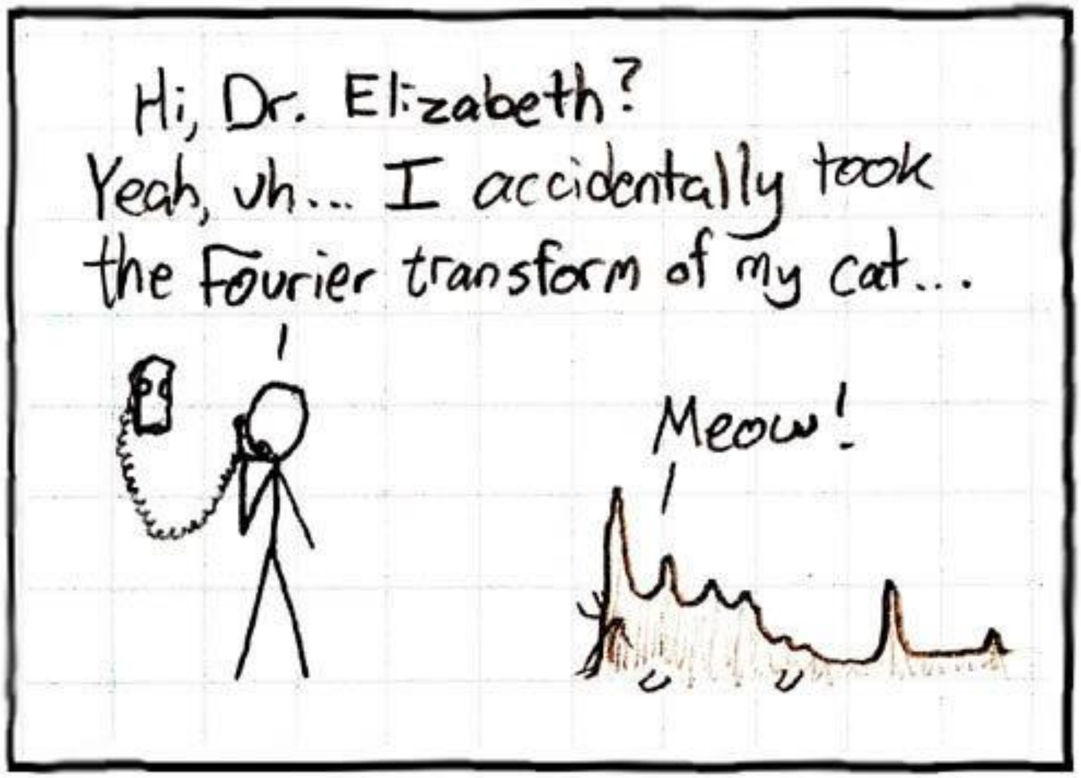




Other signals

- We can also think of all kinds of other signals the same way

Cats(?)

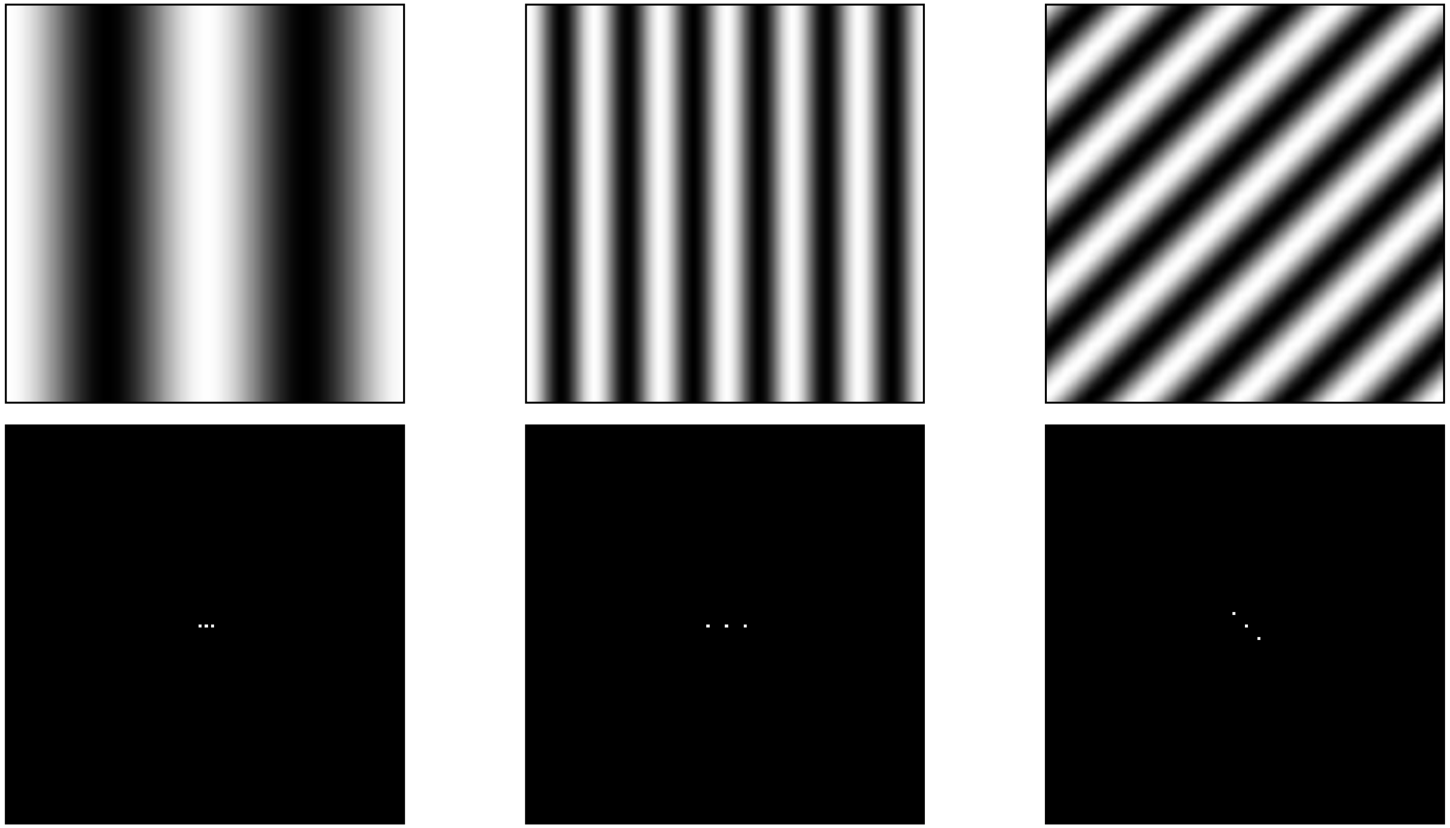

Hi, Dr. Elizabeth?
Yeah, uh... I accidentally took the Fourier transform of my cat...
Meow!

xkcd.com

Fourier analysis in images

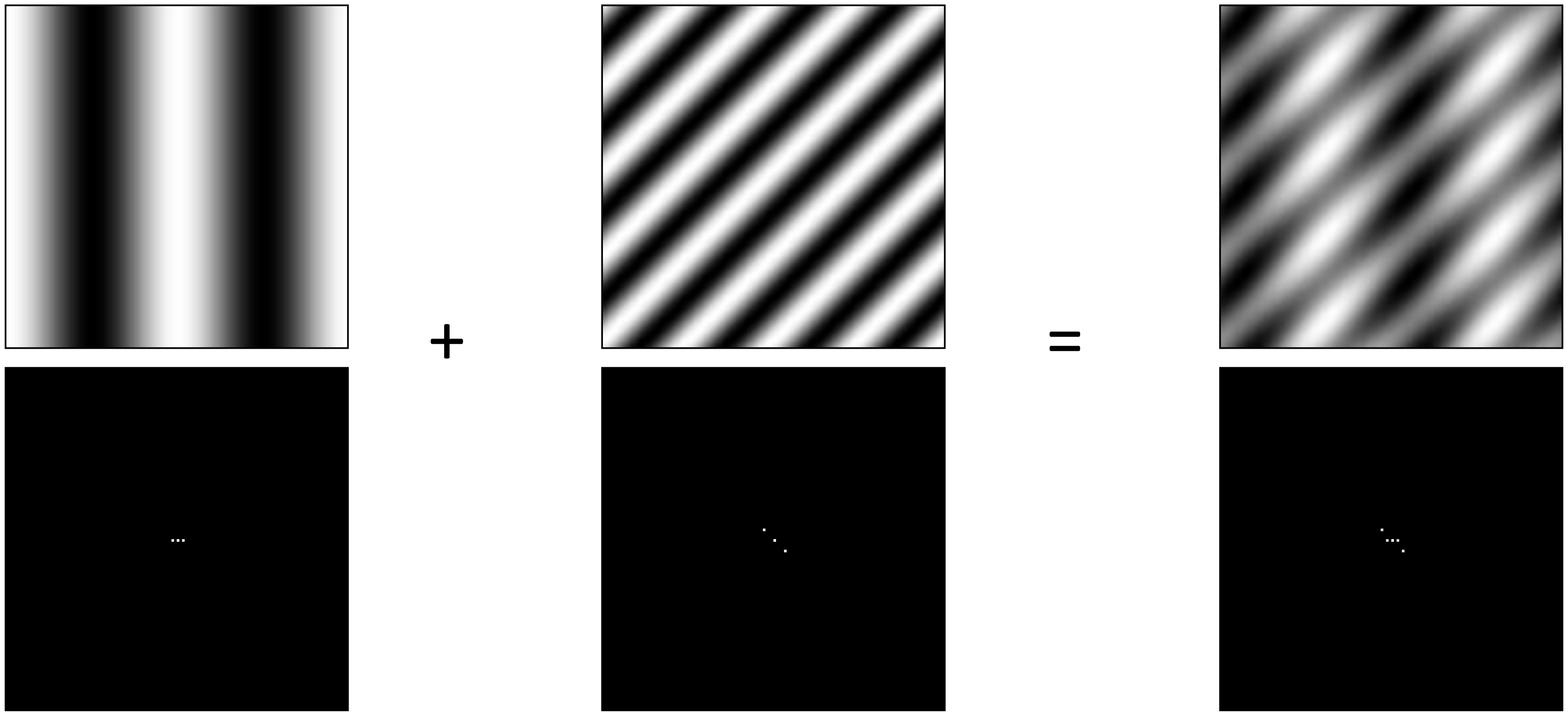
Intensity Image

Fourier Image



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

Fourier Transform

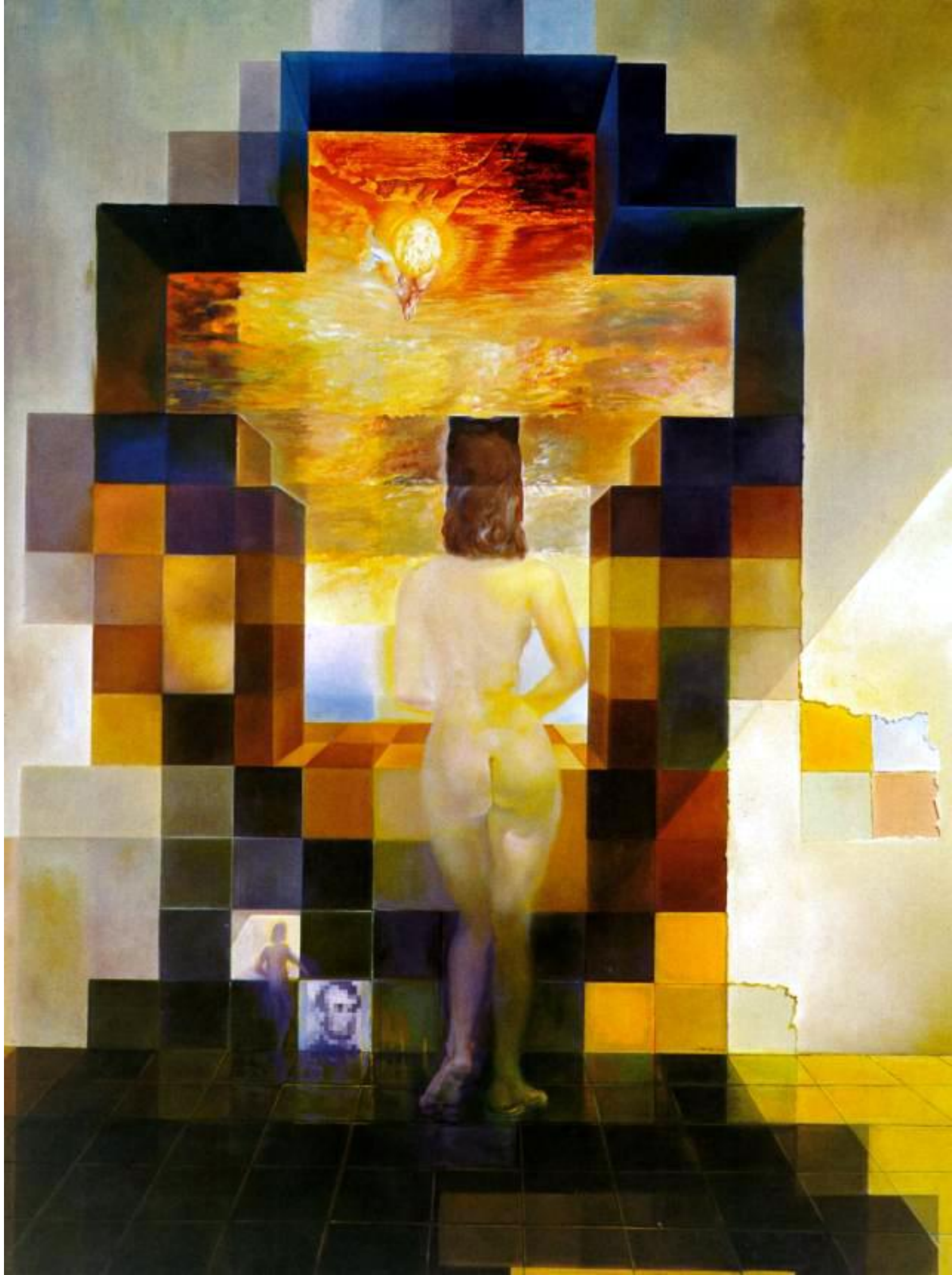
- Fourier transform stores the magnitude and phase at each frequency
    - Magnitude encodes how much signal there is at a particular frequency
    - Phase encodes spatial information (indirectly)
    - For mathematical convenience, this is often notated in terms of real and complex numbers

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

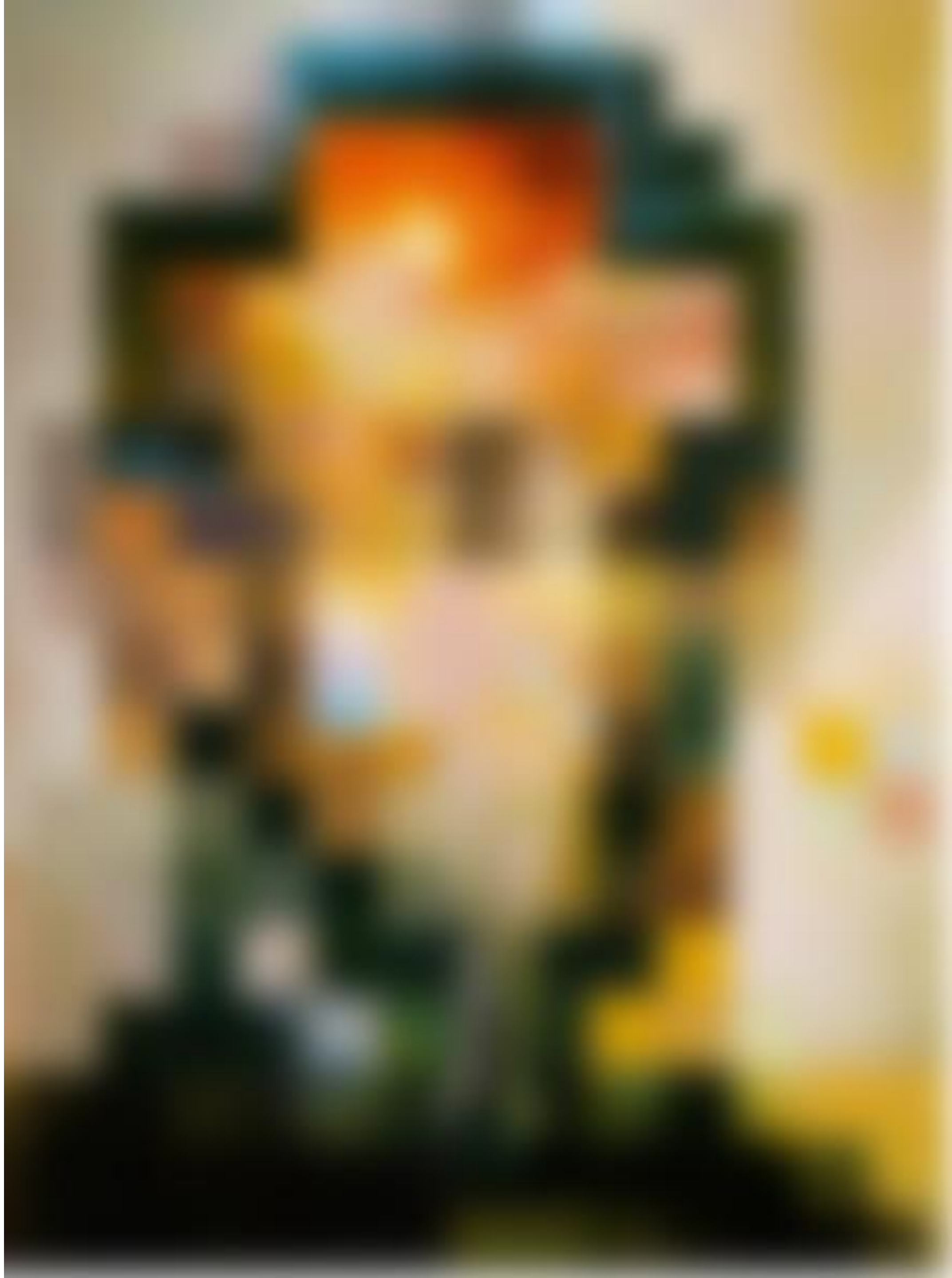
Phase: $\phi = \tan^{-1} \dfrac{I(\omega)}{R(\omega)}$

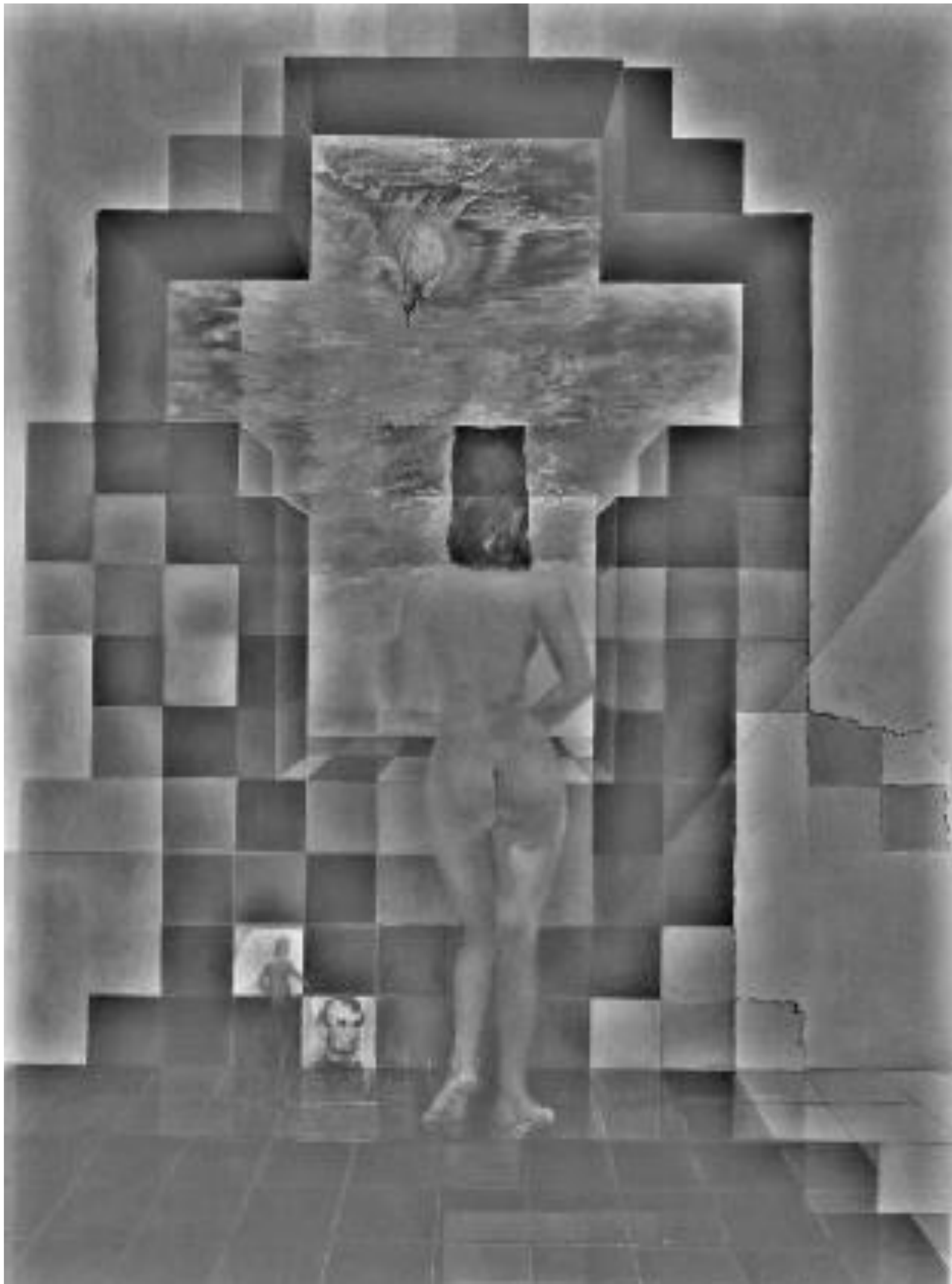
Euler's formula: $e^{inx} = \cos(nx) + i\sin(nx)$

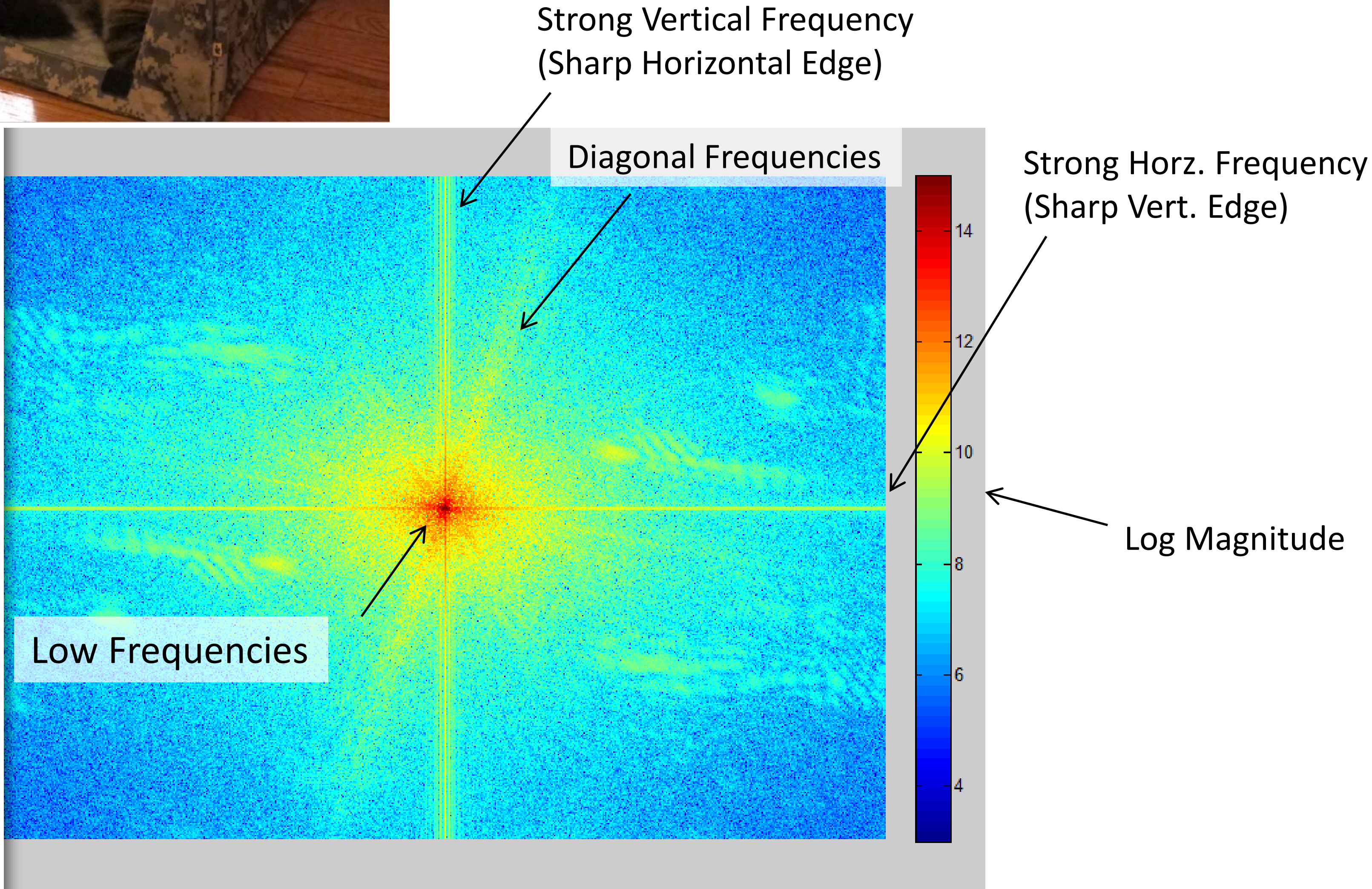
Salvador Dali invented Hybrid Images?



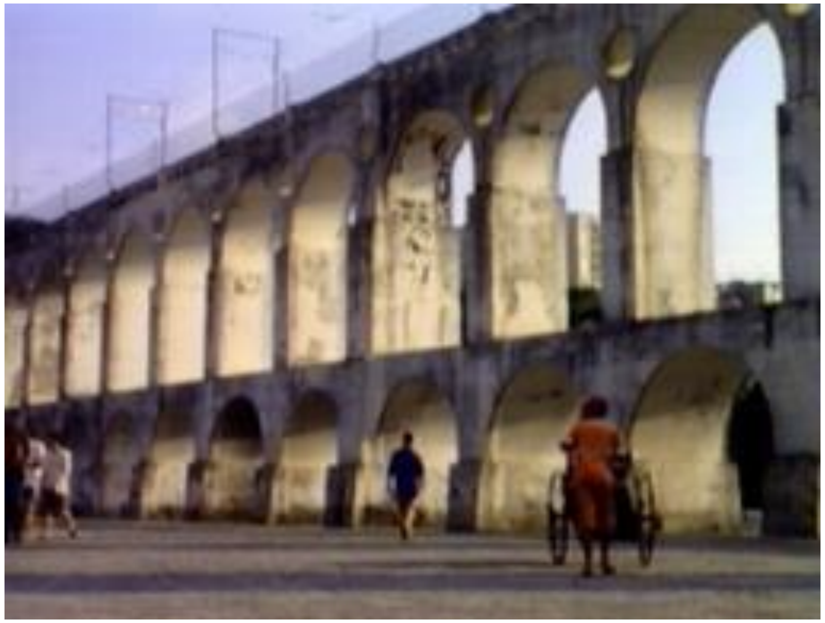
Salvador Dali
“Gala Contemplating the Mediterranean Sea, which at 30 meters becomes the portrait of Abraham Lincoln”, 1976

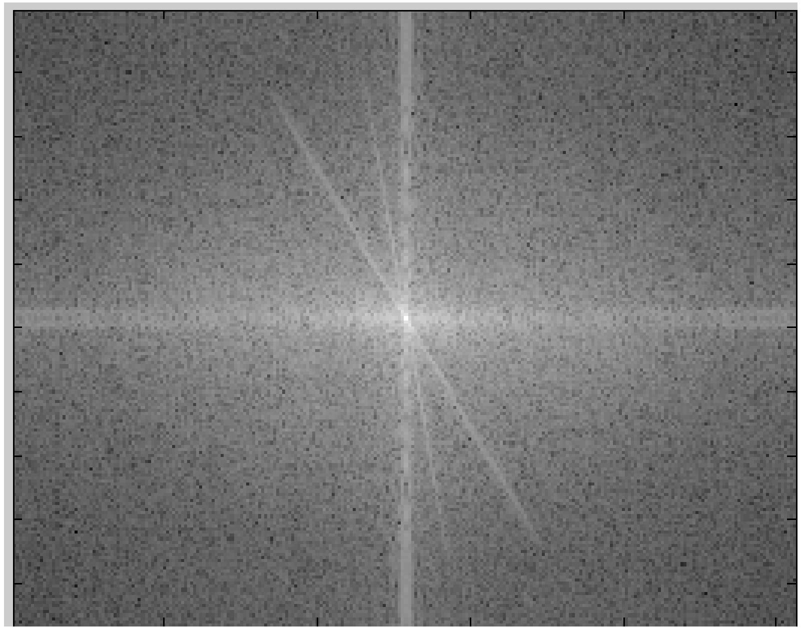




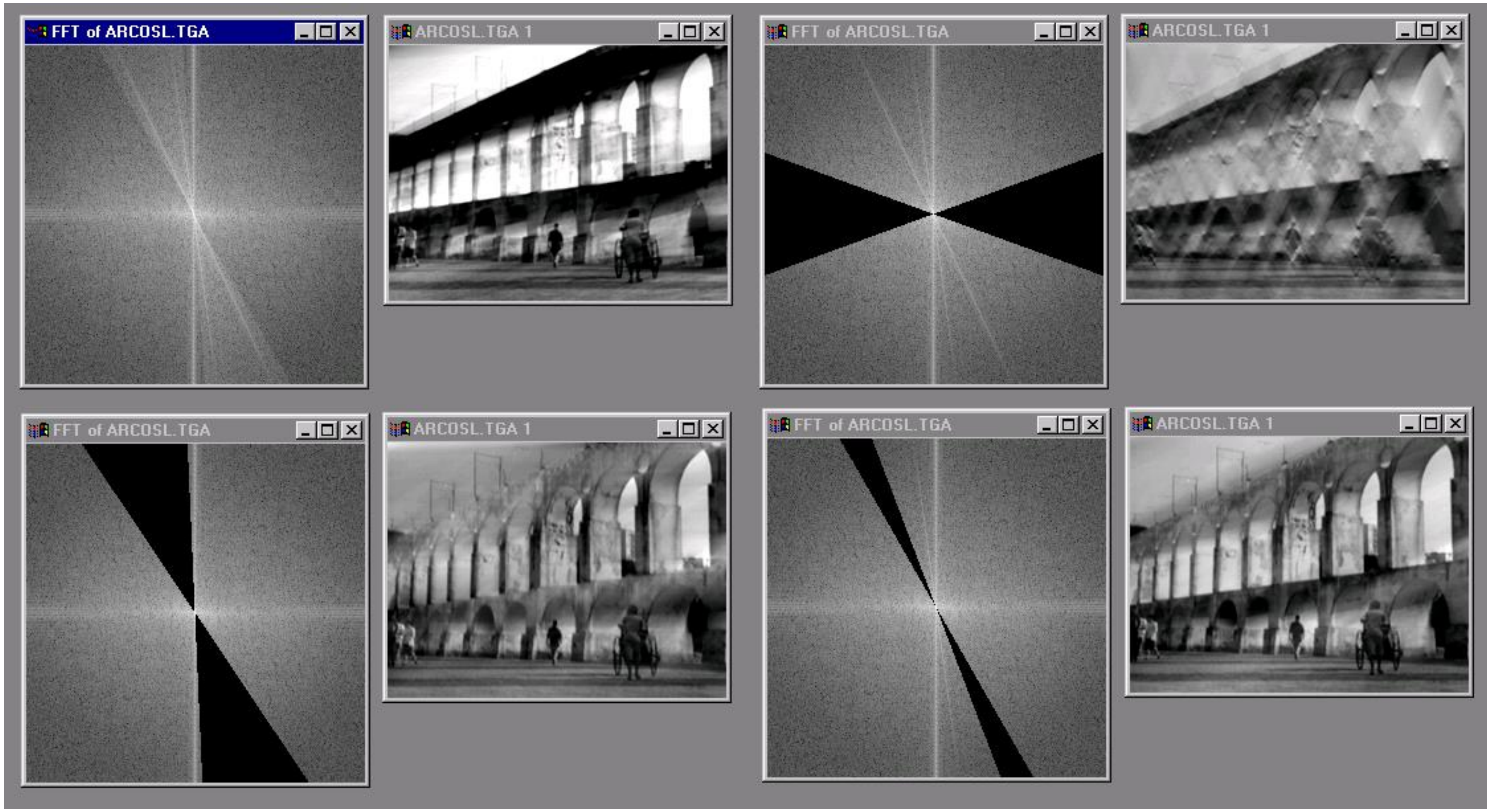

Strong Vertical Frequency
(Sharp Horizontal Edge)
Diagonal Frequencies
Strong Horz. Frequency
(Sharp Vert. Edge)
14
12
10
8
6
4
Log Magnitude
Low Frequencies

Man-made Scene

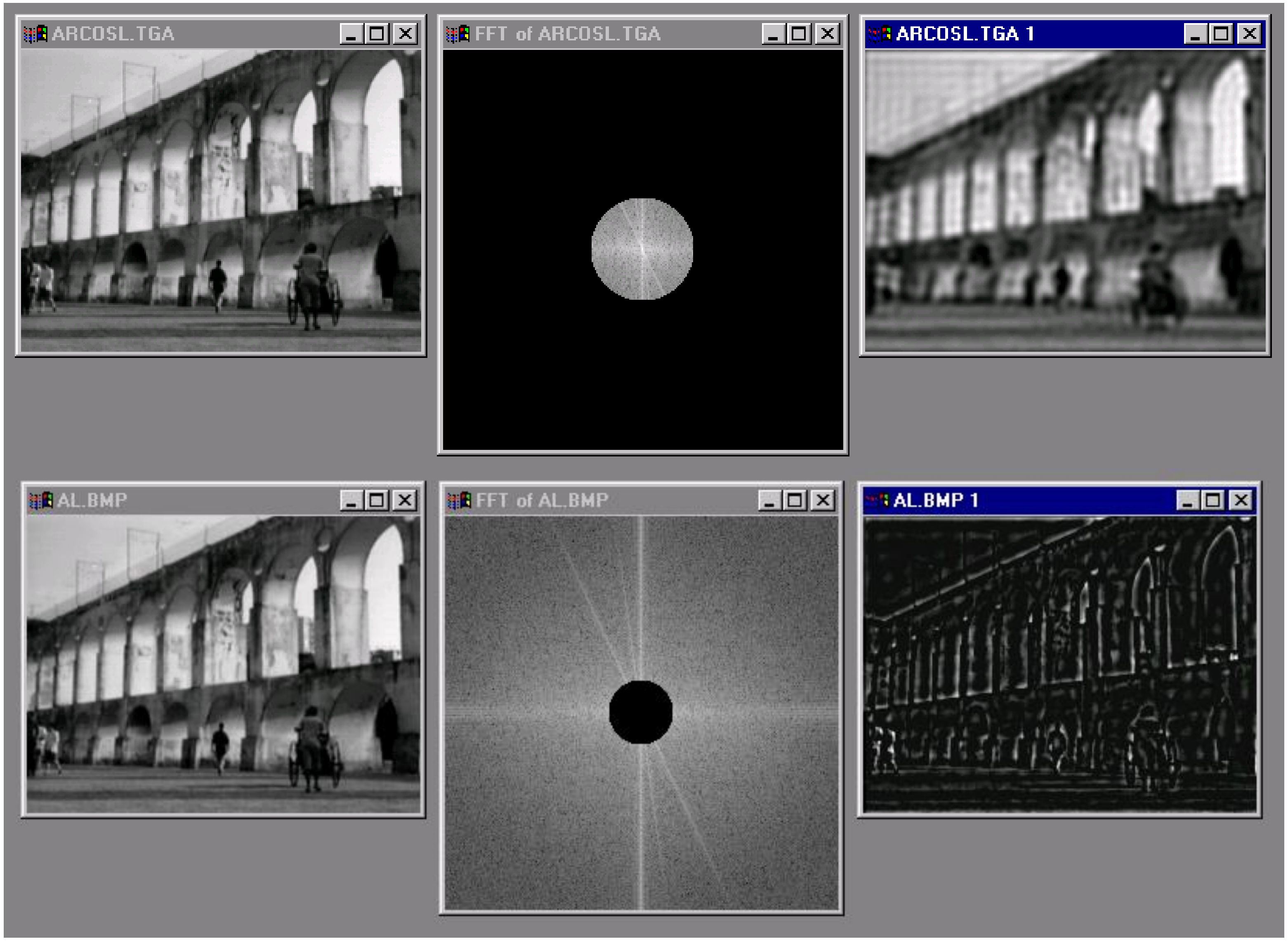




Can change spectrum, then reconstruct



Low and High Pass filtering



Computing the Fourier Transform

$$H(\omega) = \mathcal{F}\{h(x)\} = Ae^{j\phi}$$

Continuous

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

Discrete

$$H(k) = \frac{1}{N} \sum_{x=0}^{N-1} h(x) e^{-j\frac{2\pi kx}{N}}$$ *k = -N/2..N/2*

Fast Fourier Transform (FFT): NlogN

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

$$\mathrm{F}^{-1}[gh] = \mathrm{F}^{-1}[g] * \mathrm{F}^{-1}[h]$$

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

Properties of Fourier Transforms

- Linearity $\mathcal{F}[ax(t) + by(t)] = a\mathcal{F}[x(t)] + b\mathcal{F}[y(t)]$

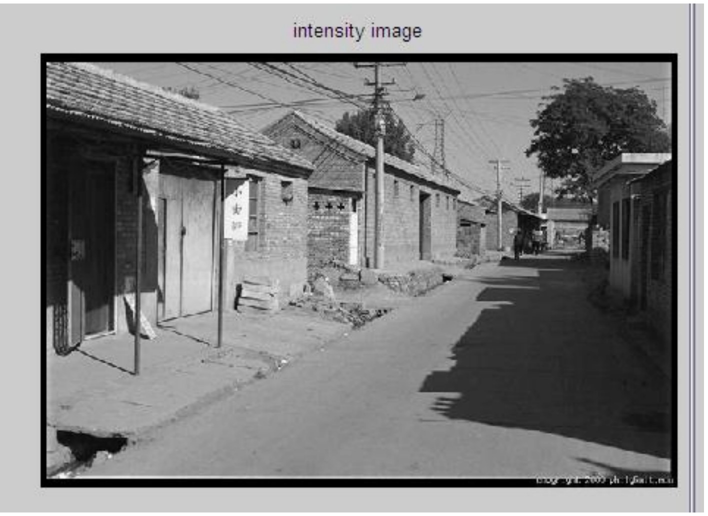
- Fourier transform of a real signal is symmetric about the origin

- The energy of the signal is the same as the energy of its Fourier transform

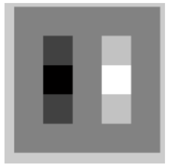
See Szeliski Book (3.4)

Filtering in spatial domain

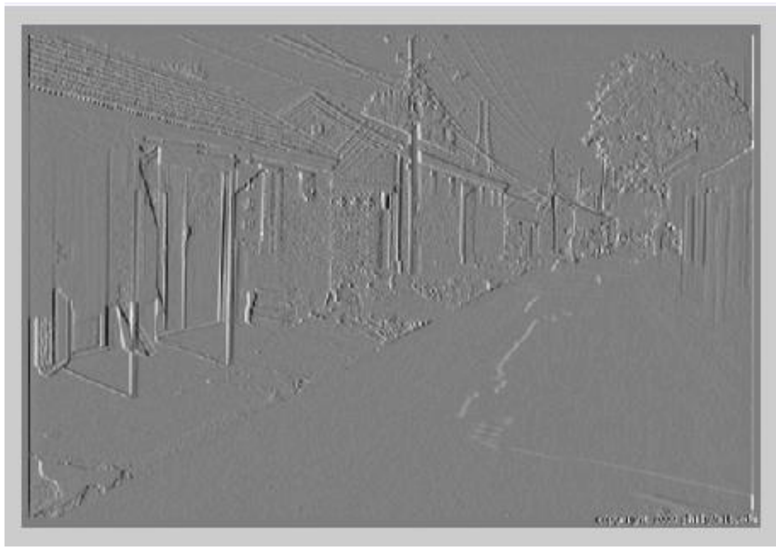
1	0	-1
2	0	-2
1	0	-1



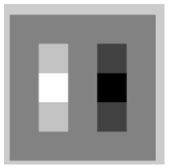
$*$

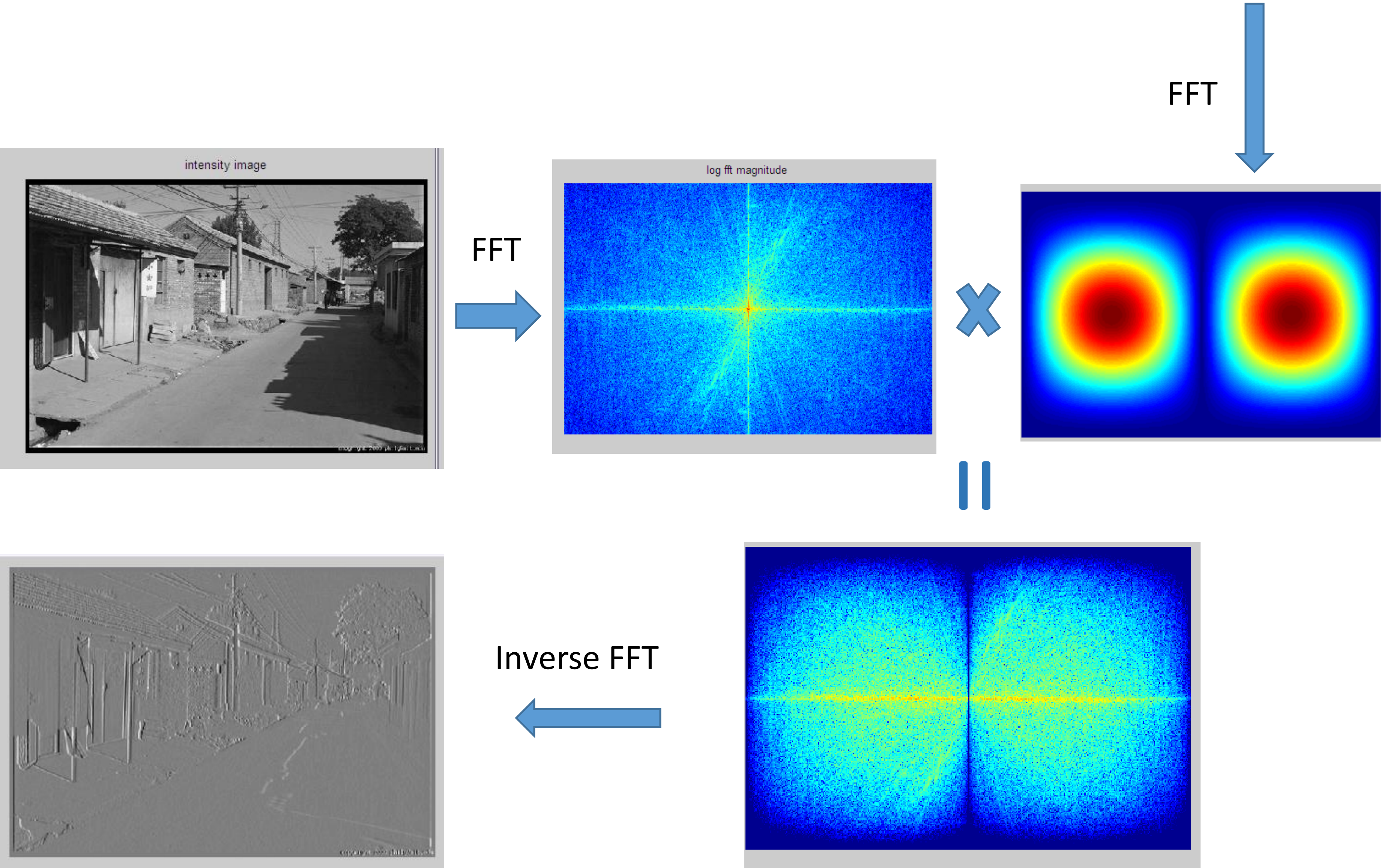


$=$



Filtering in frequency domain





FFT in Matlab

- Filtering with fft

```
im = ... % “im” should be a gray-scale floating point image
[imh, imw] = size(im);
fftsize = 1024; % should be order of 2 (for speed) and include padding
im_fft = fft2(im, fftsize, fftsize); % 1) fft im with padding
hs = 50; % filter half-size
fil = fspecial('gaussian', hs*2+1, 10);
fil_fft = fft2(fil, fftsize, fftsize); % 2) fft fil, pad to same size as image
im_fil_fft = im_fft .* fil_fft; % 3) multiply fft images
im_fil = ifft2(im_fil_fft); % 4) inverse fft2
im_fil = im_fil(1+hs:size(im,1)+hs, 1+hs:size(im, 2)+hs); % 5) remove padding
```

- Displaying with fft

```
figure(1), imagesc(log(abs(fftshift(im_fft)))), axis image, colormap jet
```

Questions

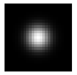
Which has more information, the phase or the magnitude?

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

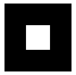
Filtering

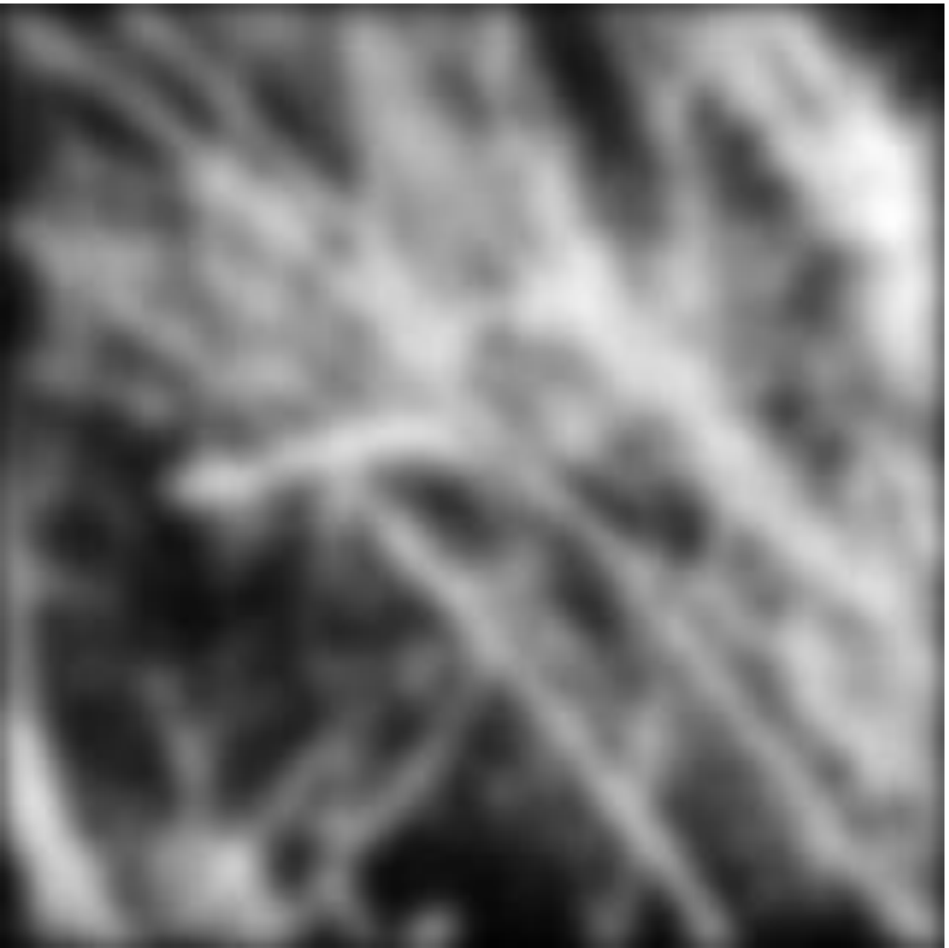
Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

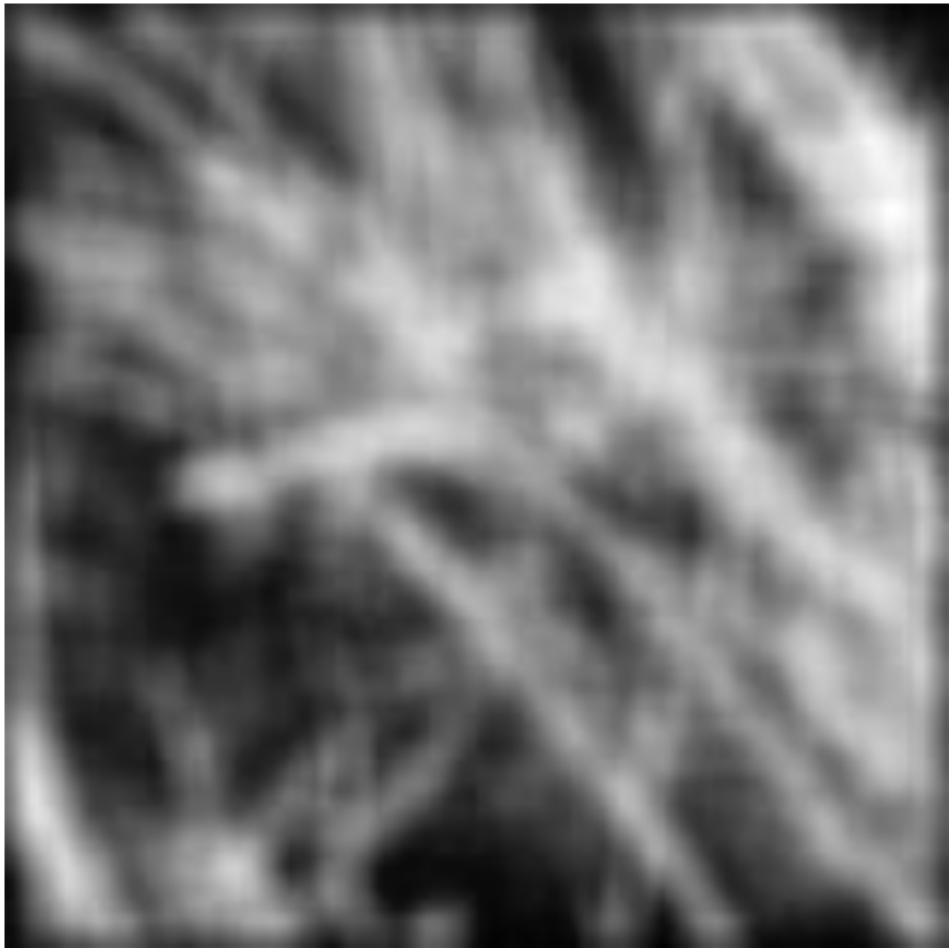
Gaussian



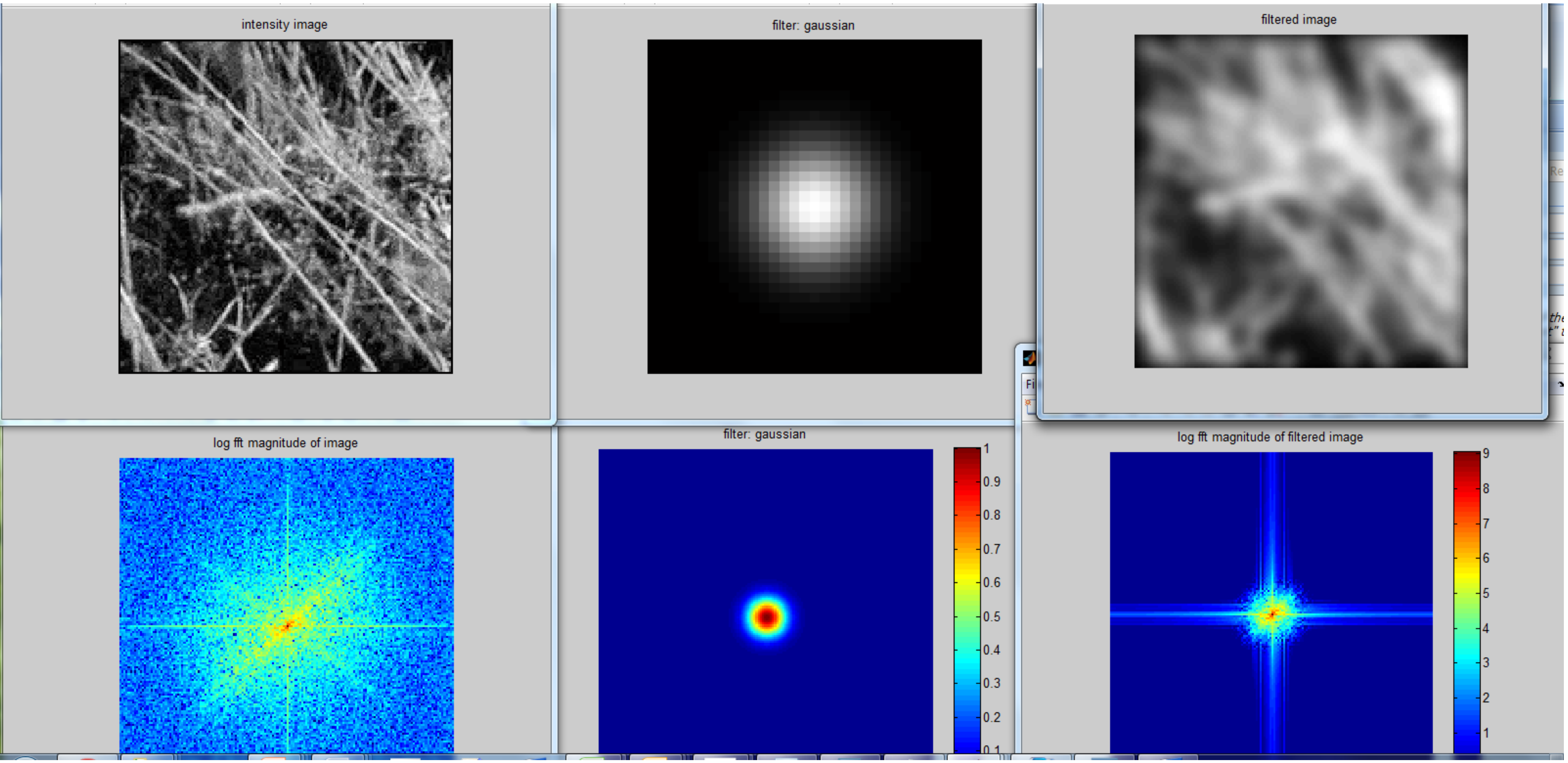
Box filter



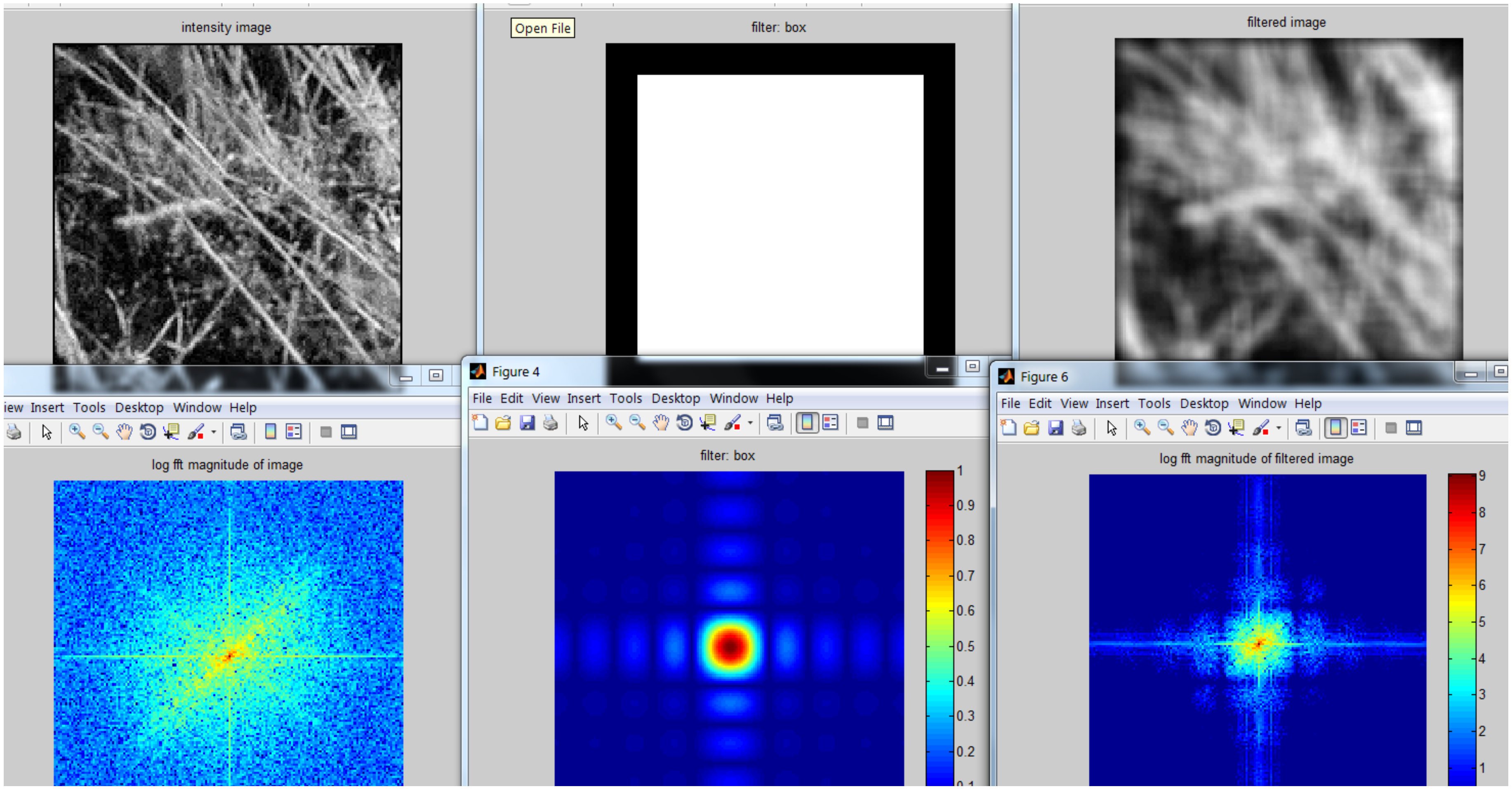




Gaussian


intensity image
filter: gaussian
filtered image
log fft magnitude of image
filter: gaussian
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
log fft magnitude of filtered image
9
8
7
6
5
4
3
2
1

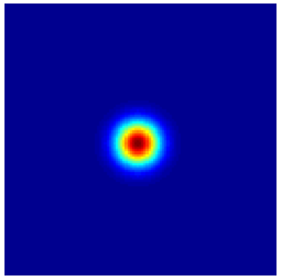
Box Filter


intensity image
Open File
filter: box
filtered image
Figure 4
Figure 6
File Edit View Insert Tools Desktop Window Help
log fft magnitude of image
filter: box
log fft magnitude of filtered image
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
9
8
7
6
5
4
3
2
1

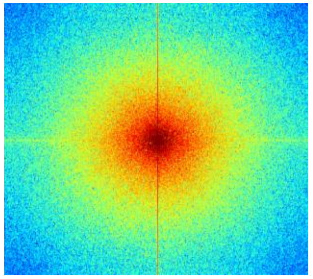
Question

Match the spatial domain image to the Fourier magnitude image

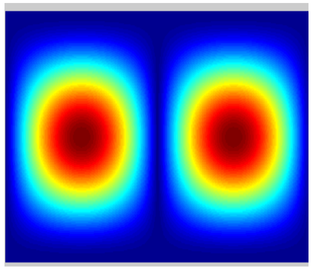
1



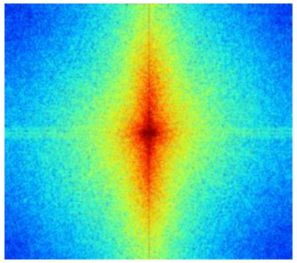
2



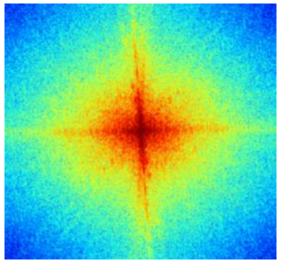
3



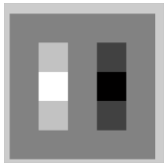
4



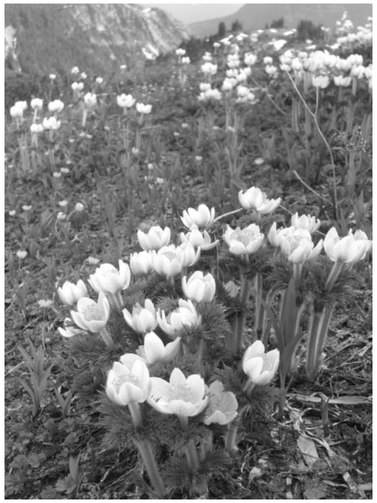
5



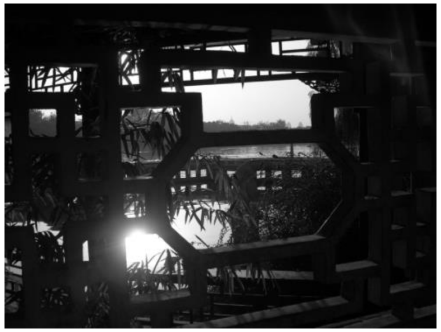
A



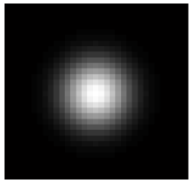
B



C



D



E

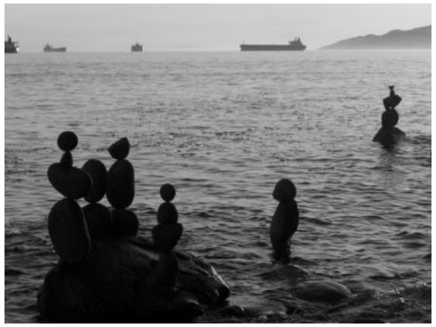
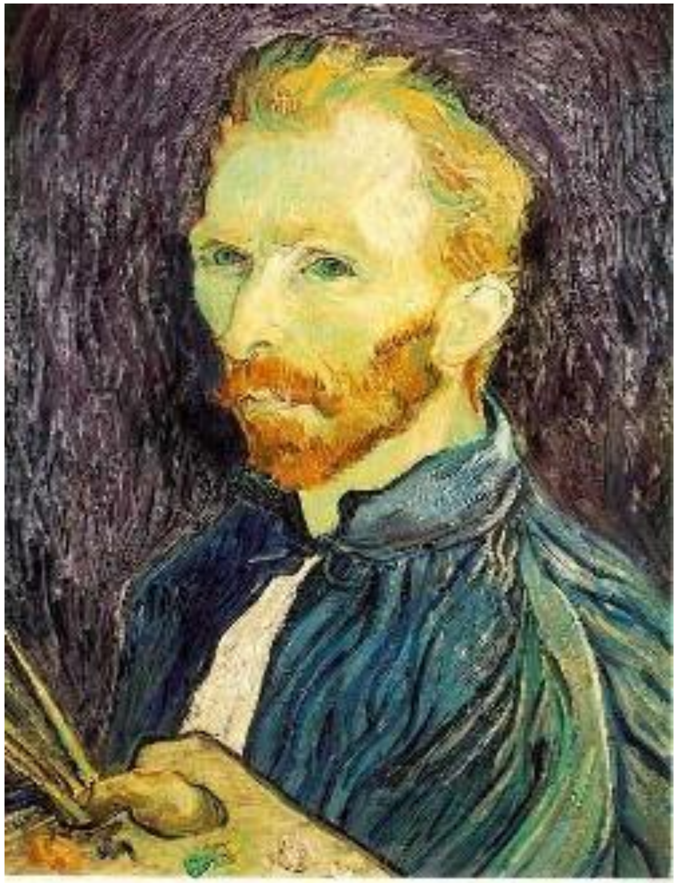


Image half-sizing

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



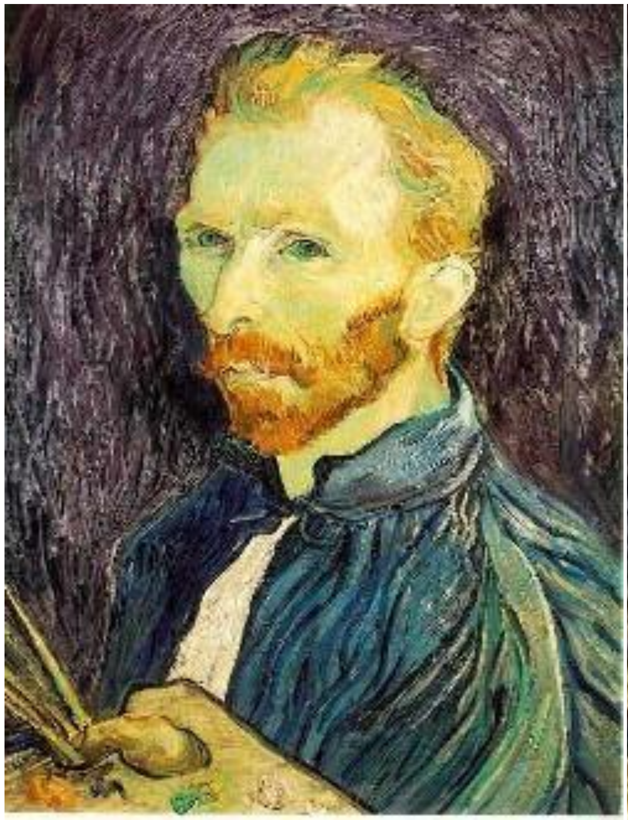
Throw away every other row and column to create a 1/2 size image
- called *image sub-sampling*

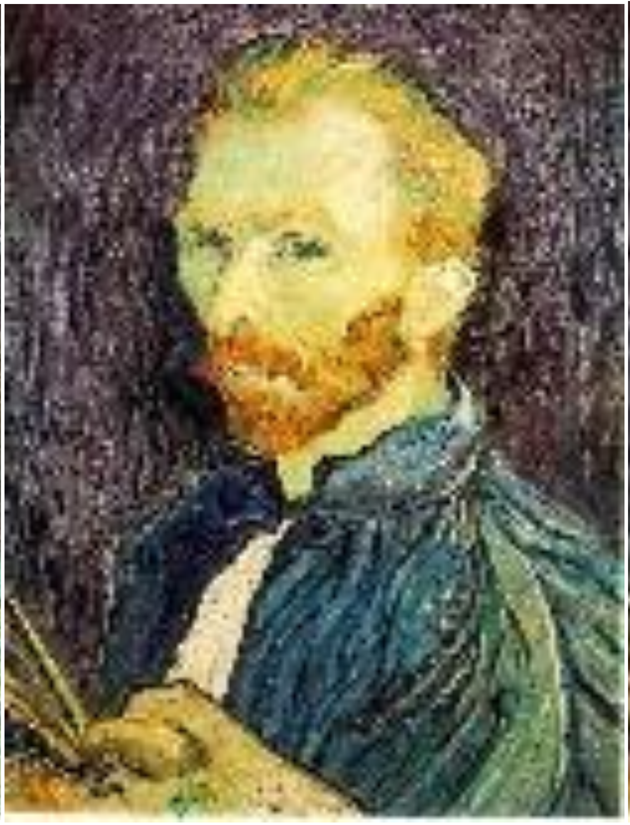
1/4

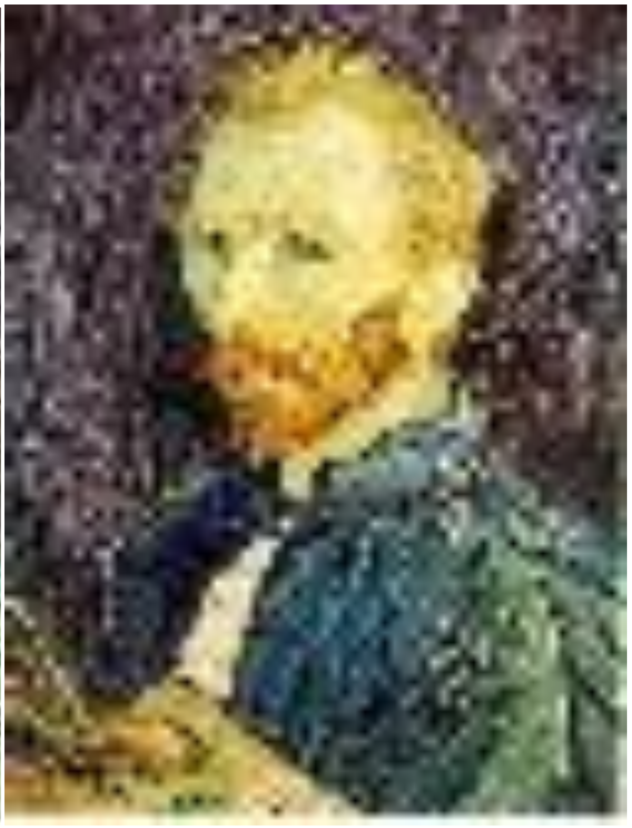
1/8

Slide by Steve Seitz

Image sub-sampling


1/2

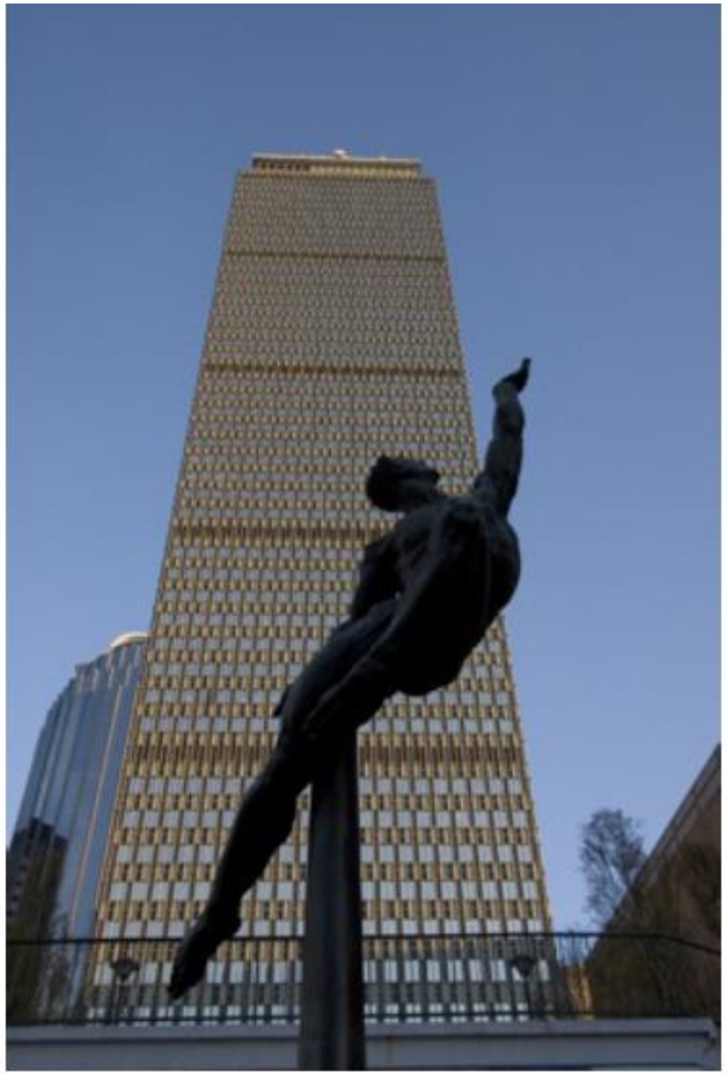

1/4 (2x zoom)

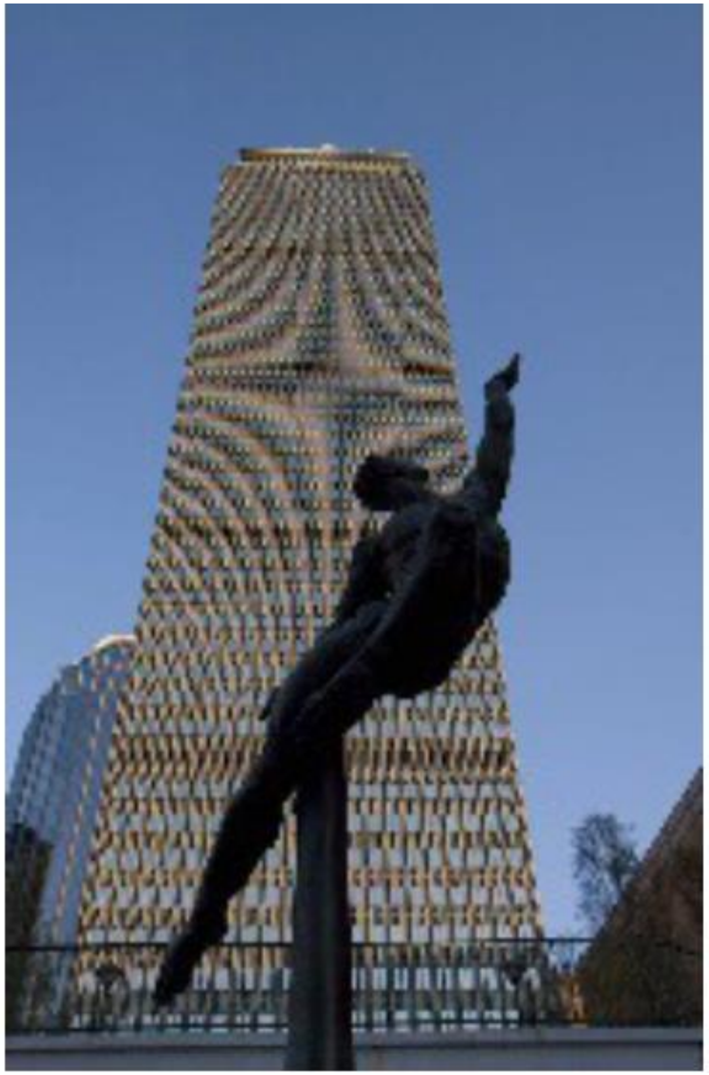

1/8 (4x zoom)

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

Slide by Steve Seitz

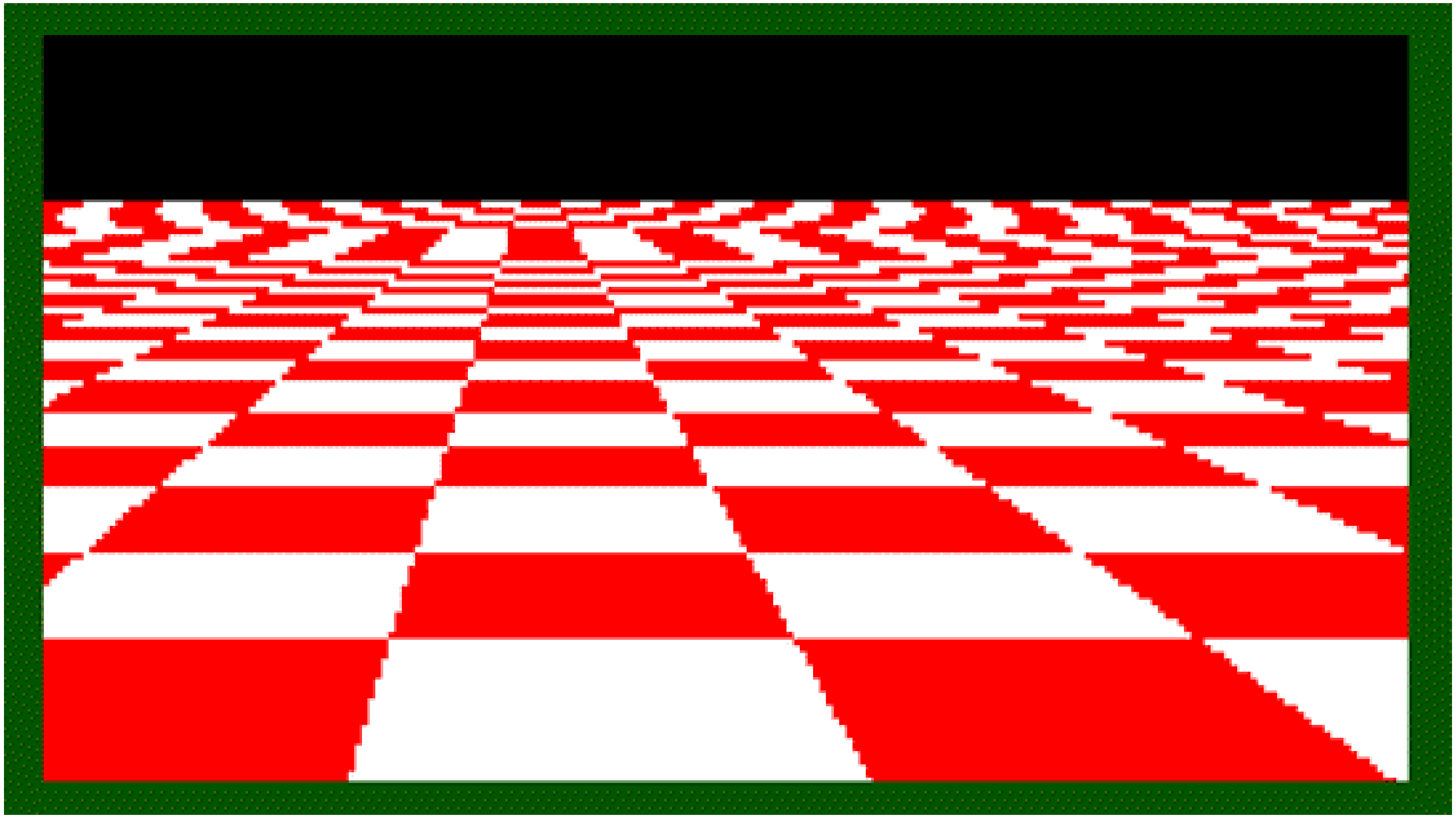
Image sub-sampling





Source: F. Durand

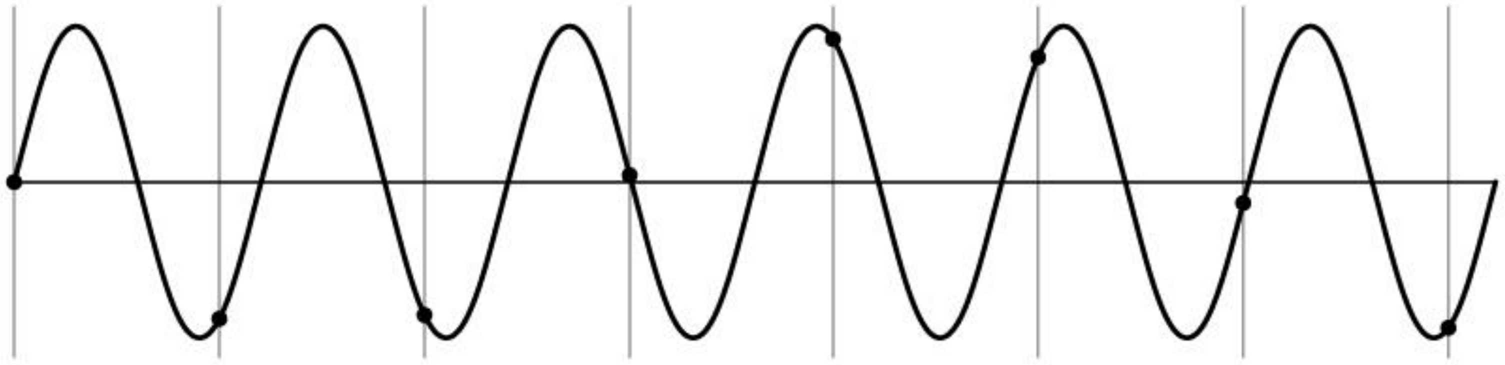
Even worse for synthetic images



Source: L. Zhang

Aliasing problem

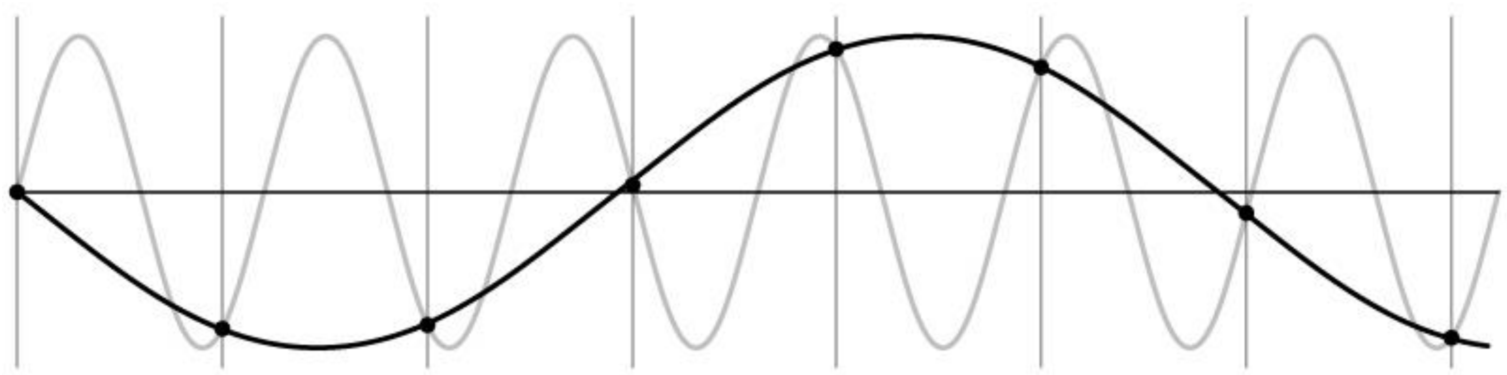
- 1D example (sinewave):



Source: S. Marschner

Aliasing problem

- 1D example (sinewave):



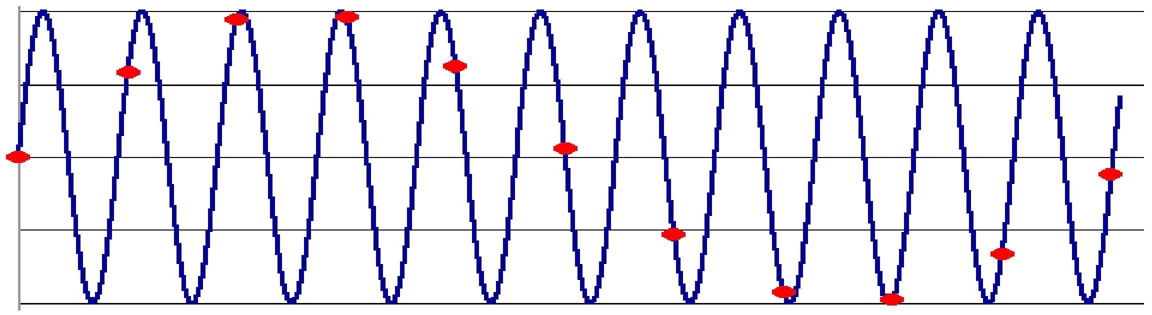
Source: S. Marschner

Aliasing problem

- Sub-sampling may be dangerous….
- Characteristic errors may appear:
  - “Wagon wheels rolling the wrong way in movies”
  - “Checkerboards disintegrate in ray tracing”
  - “Striped shirts look funny on color television”

Source: D. Forsyth

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*

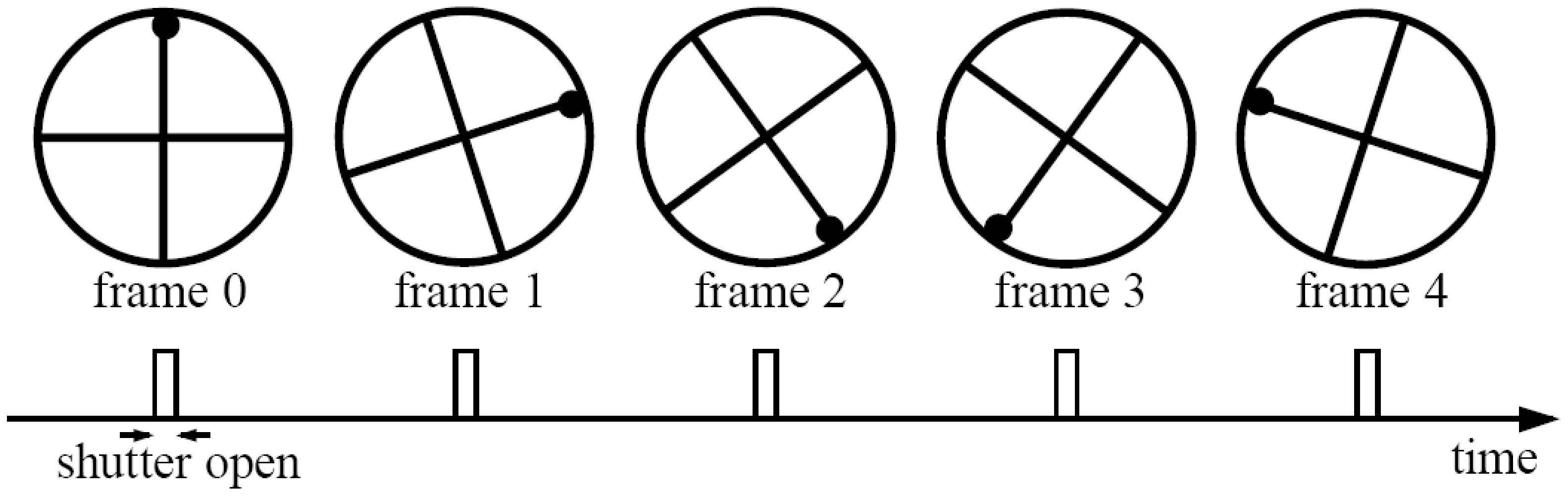
- To do sampling right, need to understand the structure of your signal/image
- To avoid aliasing:
  - sampling rate ≥ 2 * max frequency in the image
    - said another way: ≥ two samples per cycle
  - This minimum sampling rate is called the **Nyquist rate**

Source: L. Zhang

Wagon-wheel effect

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)

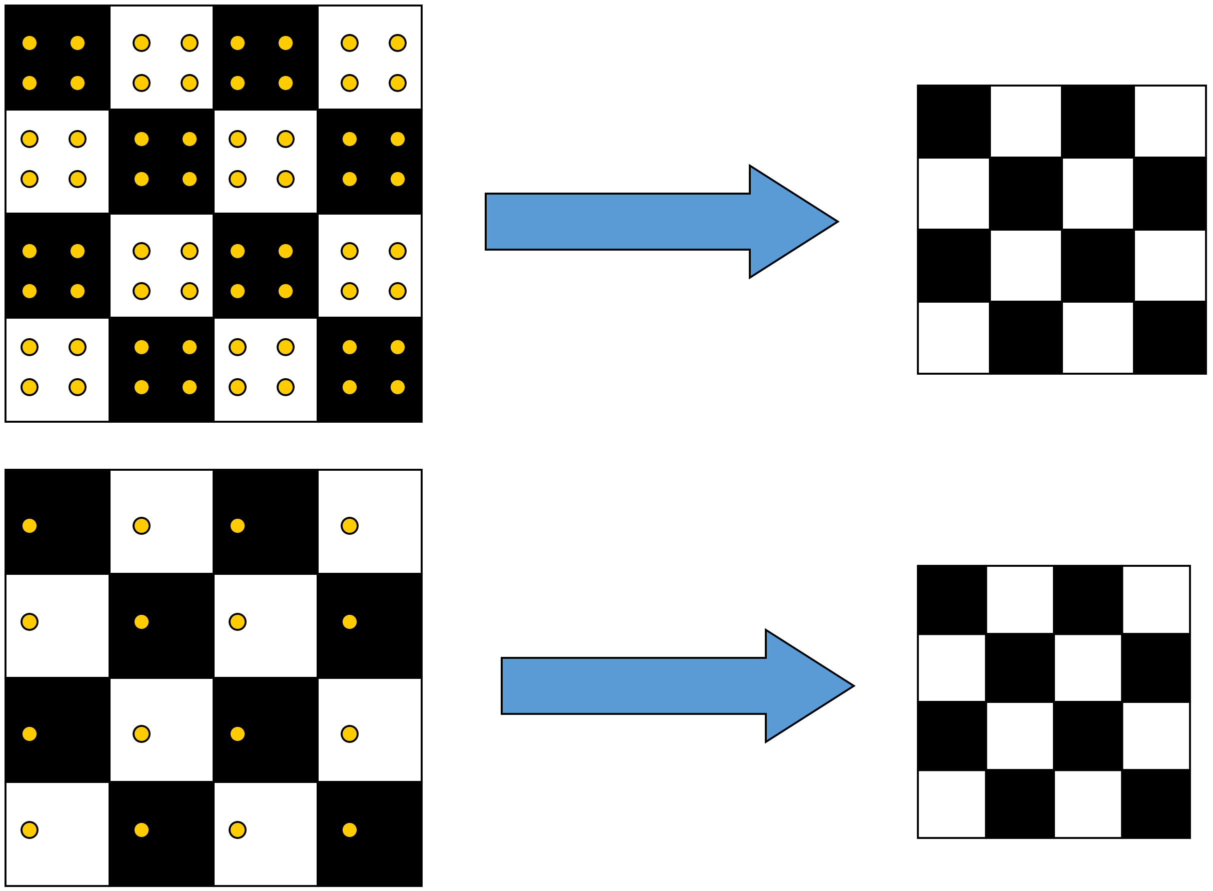
(See http://www.michaelbach.de/ot/mot_wagonWheel/index.html)

Source: L. Zhang

Wagon-wheel effect

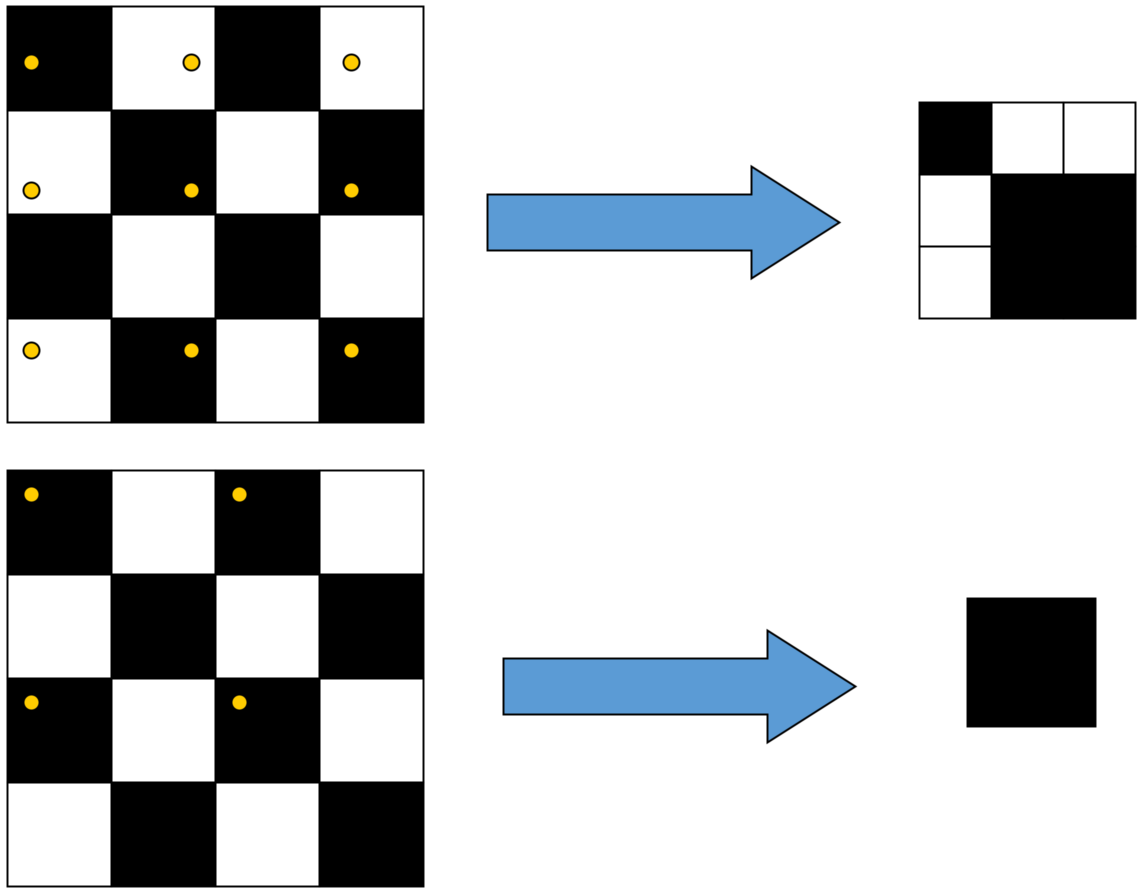
https://www.youtube.com/watch?v=QOwzkND_ooU

Sampling an image



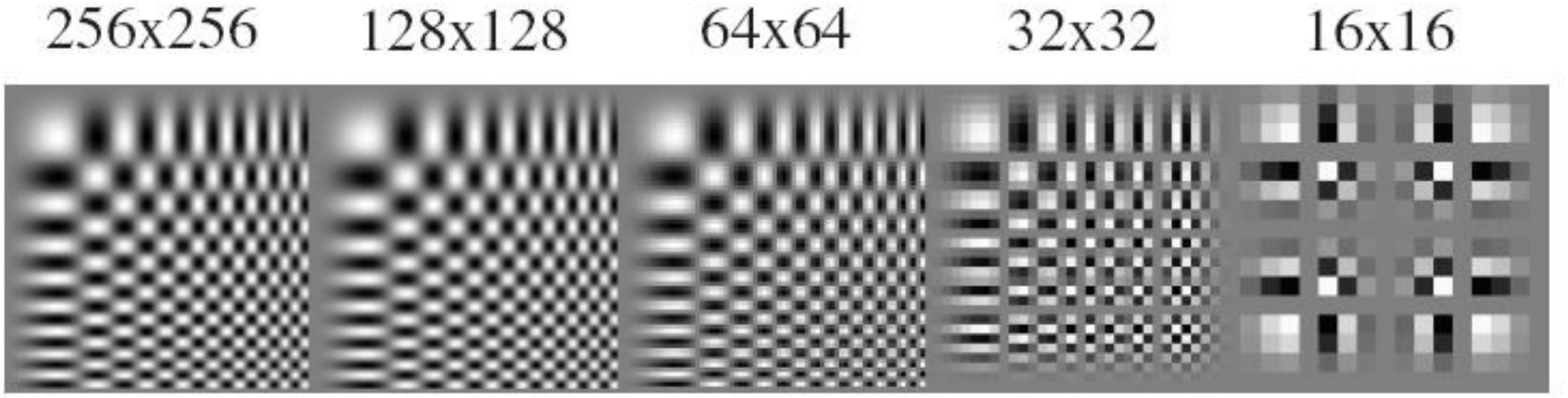
Examples of GOOD sampling

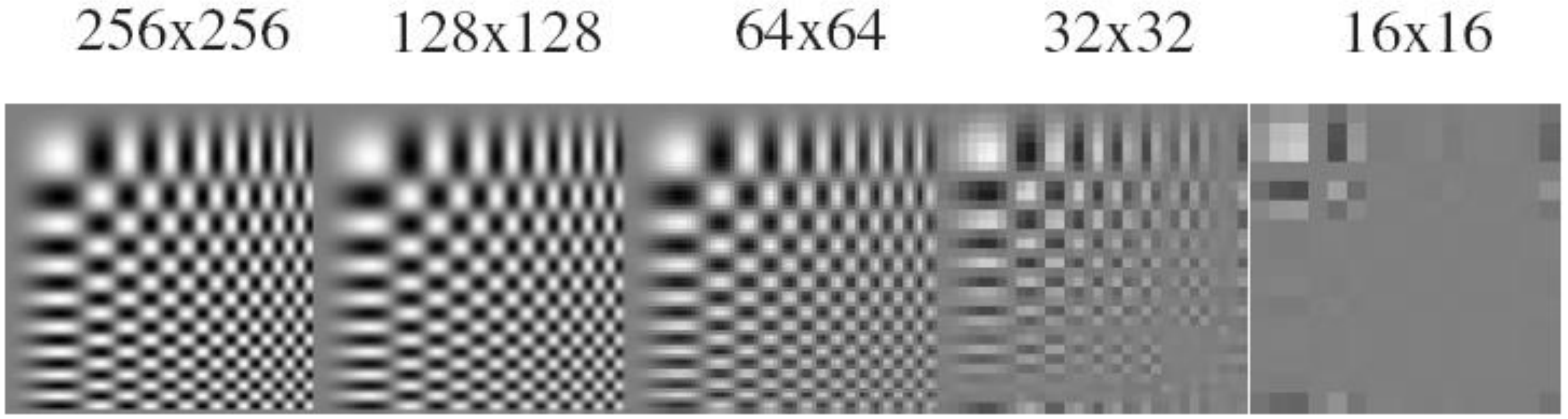
Undersampling



Examples of BAD sampling -> Aliasing

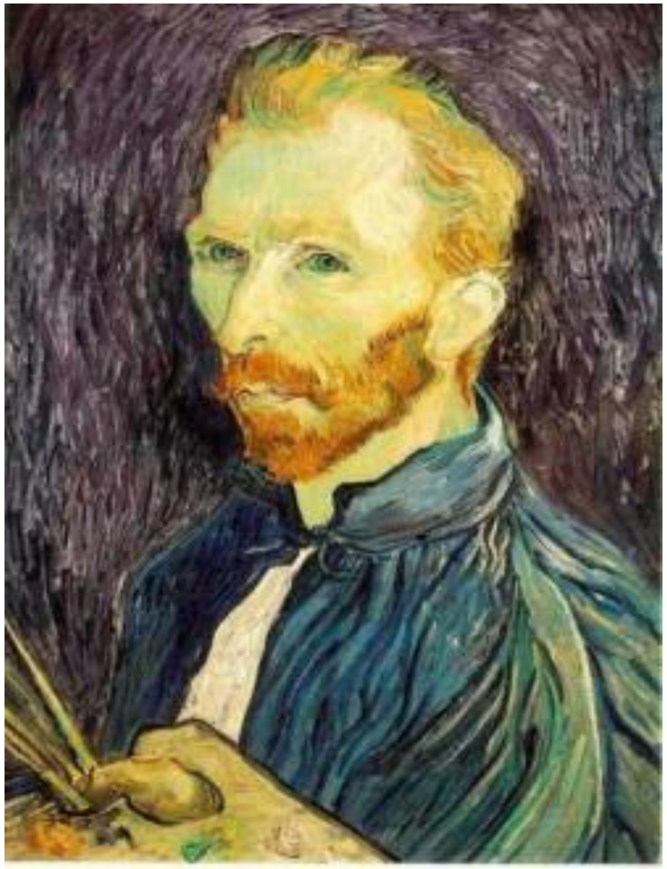
Anti-aliasing

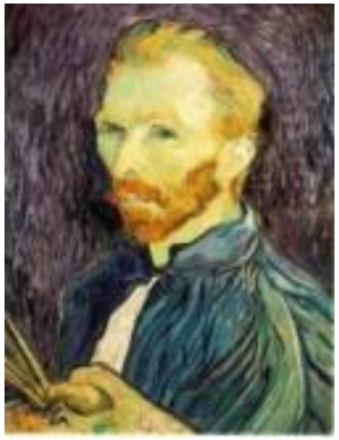


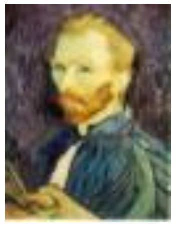


Forsyth and Ponce 2002

Gaussian (low-pass) pre-filtering


Gaussian 1/2

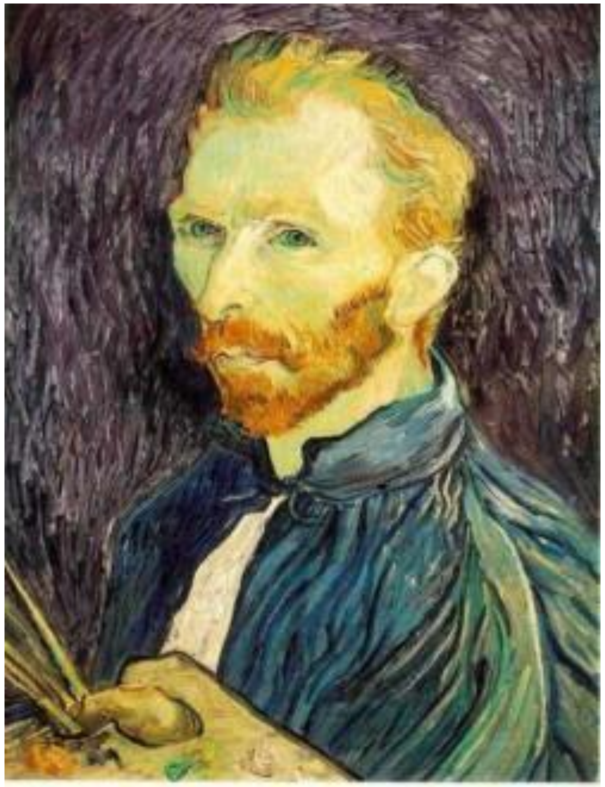

G 1/4

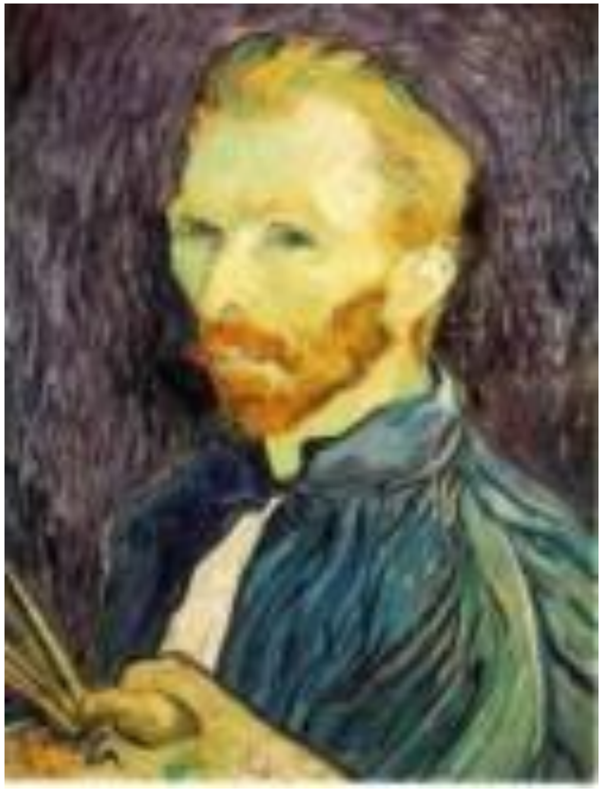

G 1/8

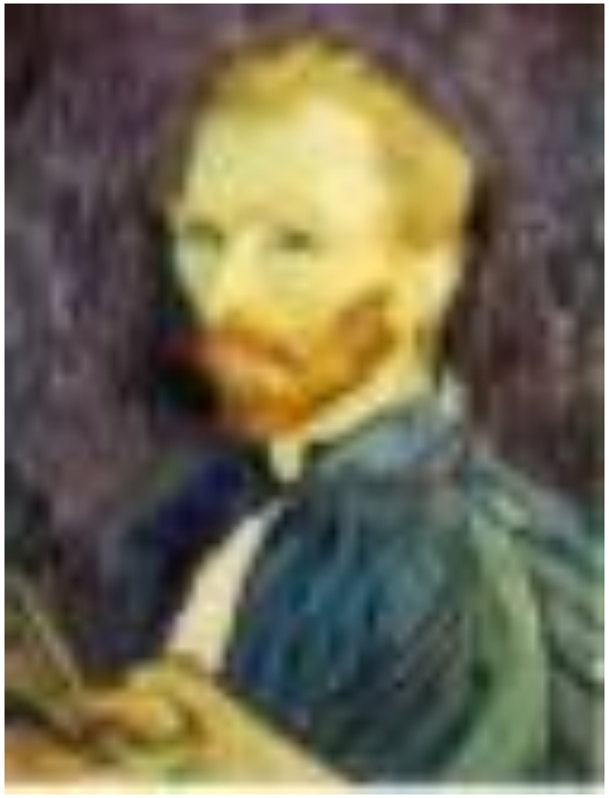
- Solution: filter the image, *then* subsample

Source: S. Seitz

Subsampling with Gaussian pre-filtering


Gaussian 1/2

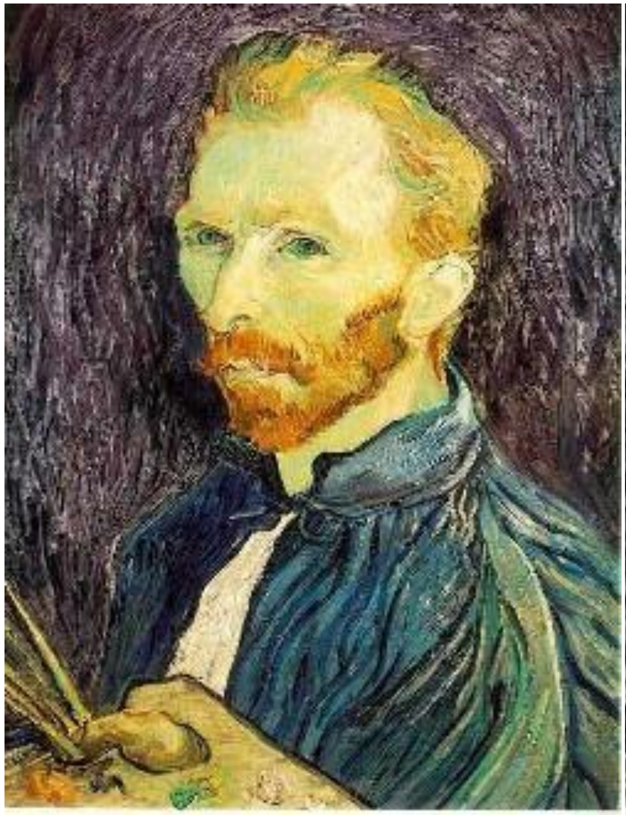

G 1/4

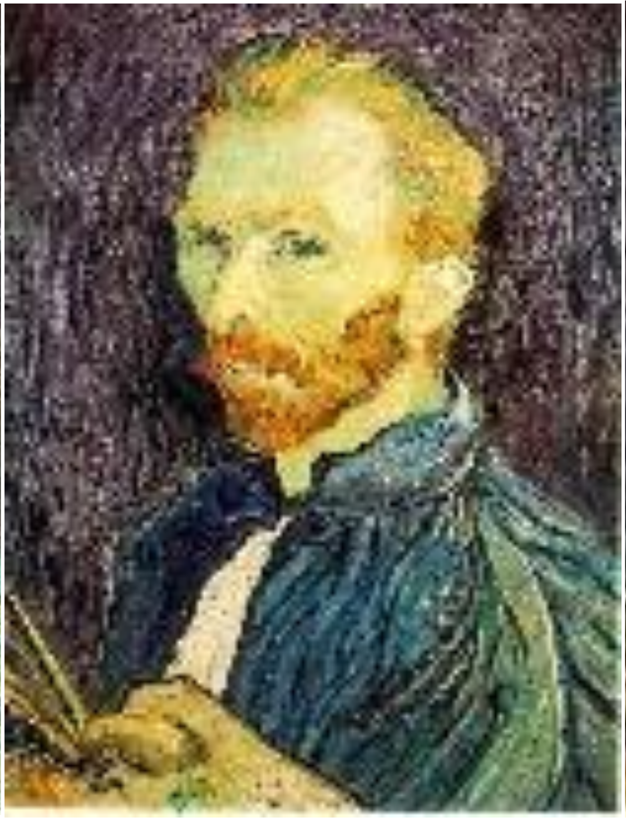

G 1/8

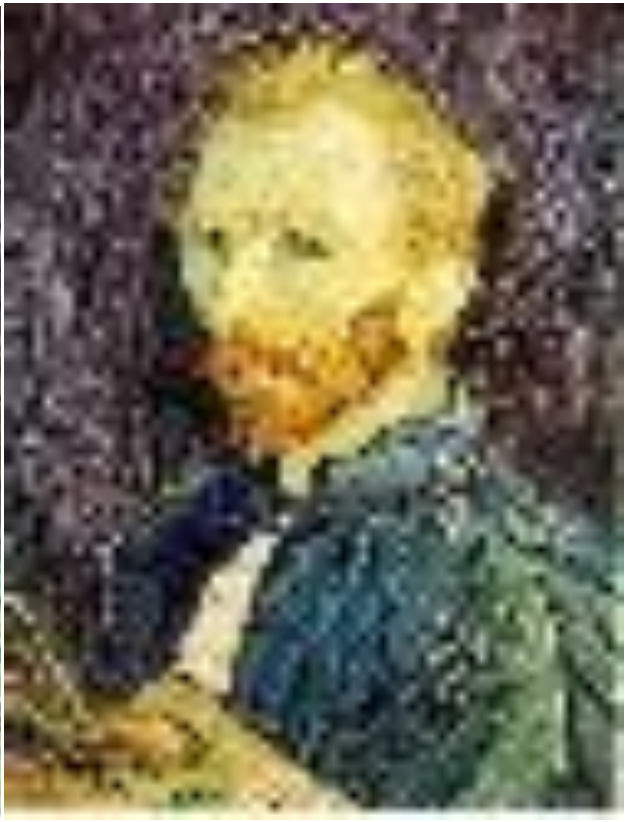
- Solution: filter the image, *then* subsample

Source: S. Seitz

Compare with...


1/2


1/4 (2x zoom)


1/8 (4x zoom)

Source: S. Seitz

Why does a lower resolution image still make sense to us? What do we lose?

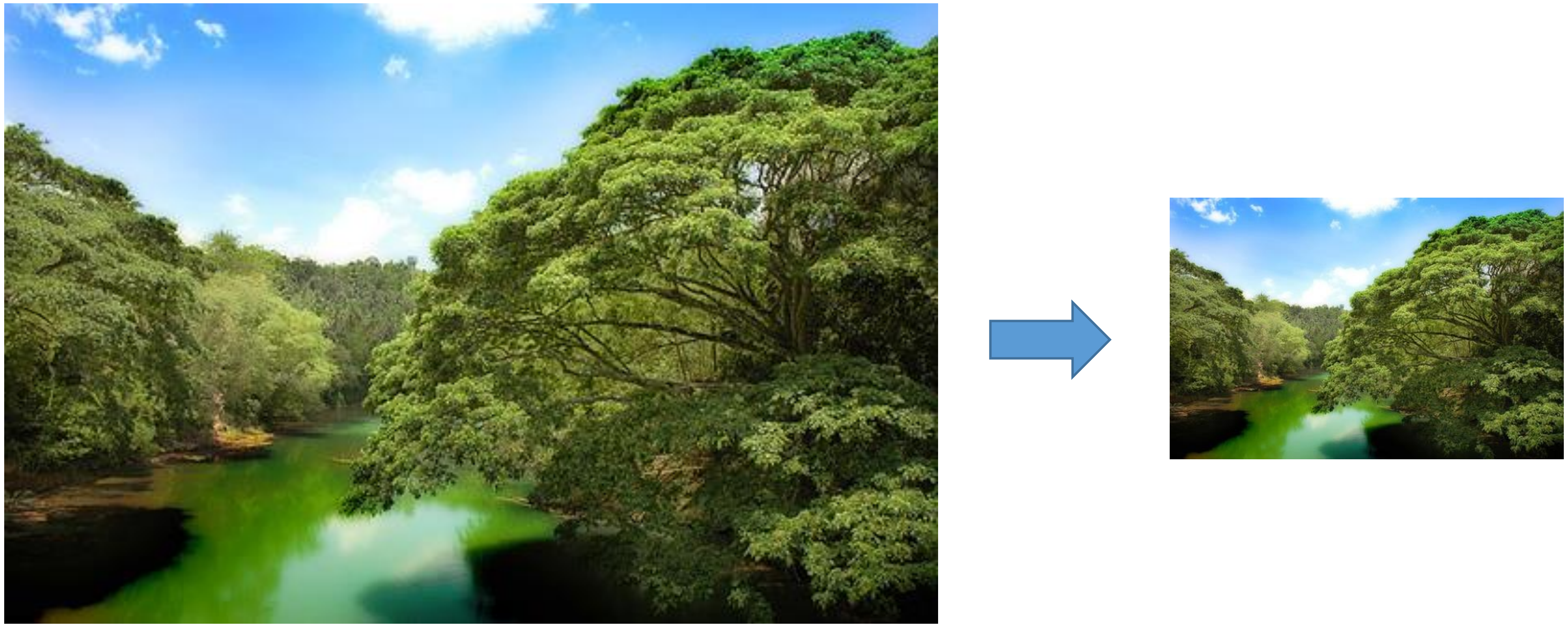
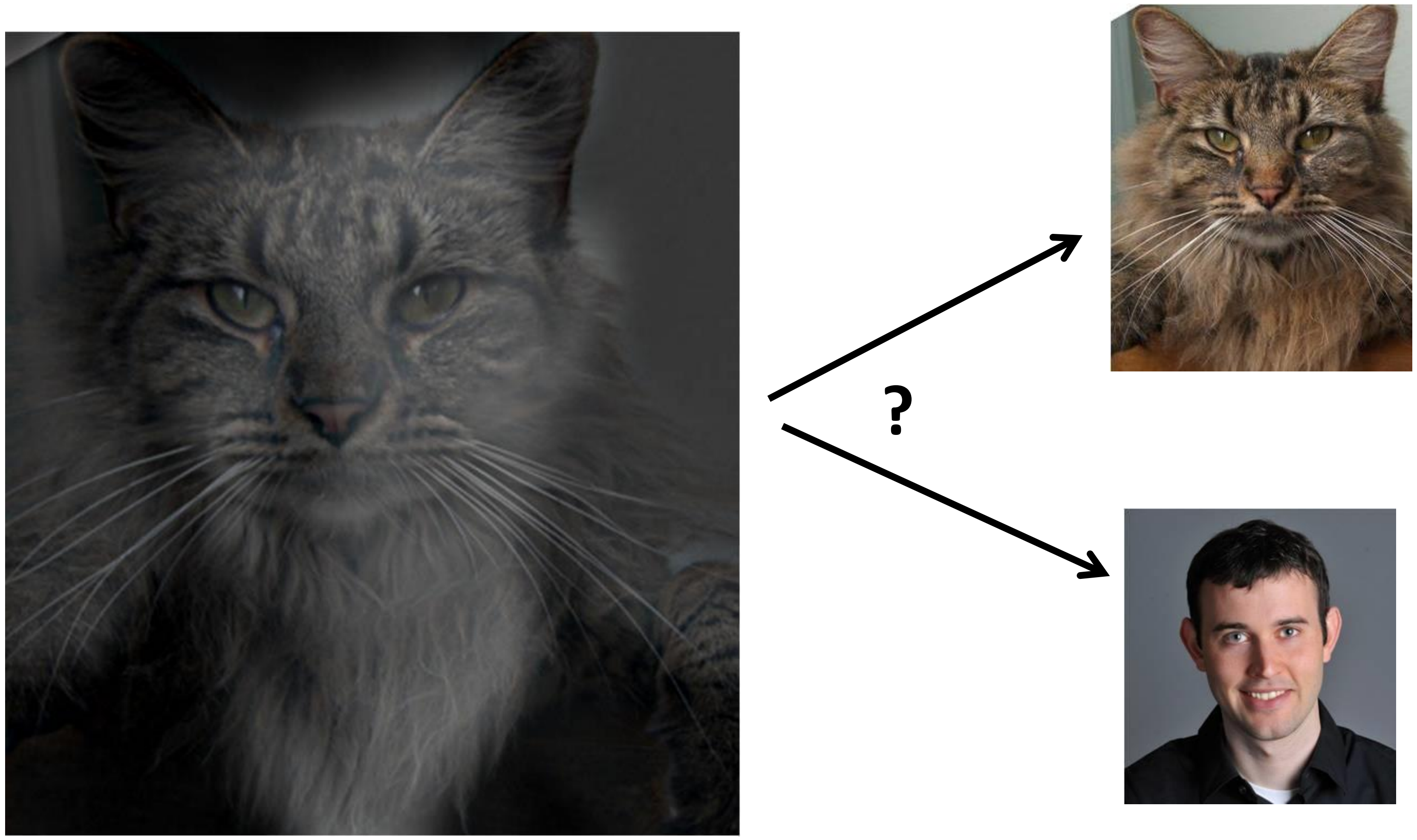


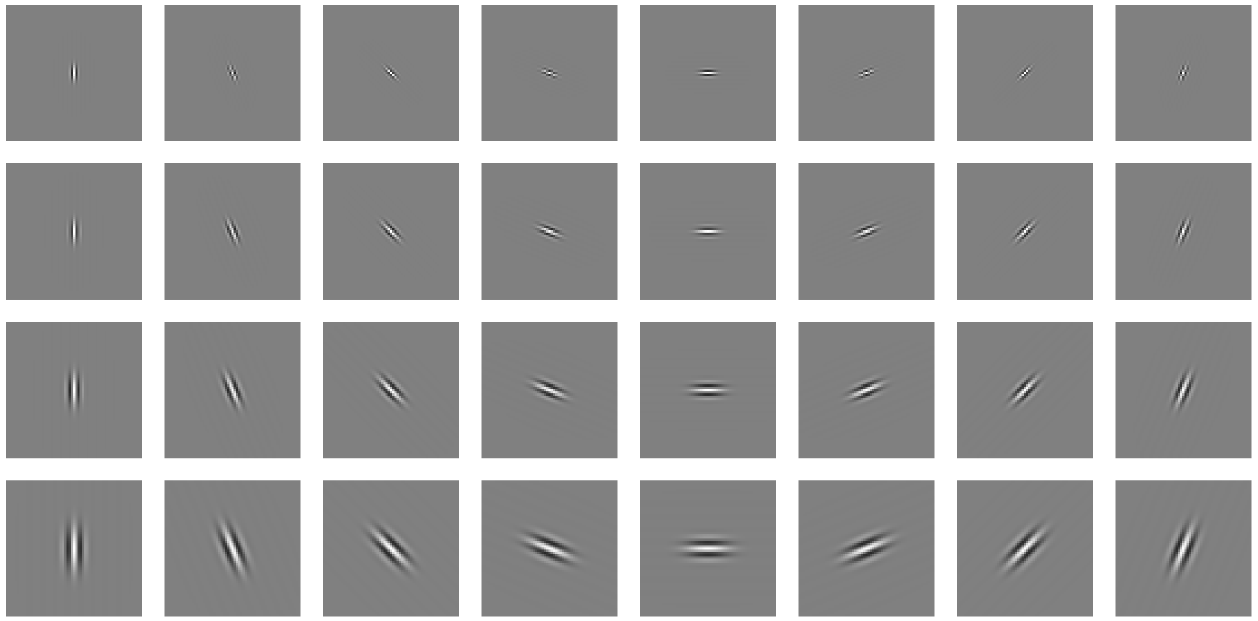
Image: http://www.flickr.com/photos/igorms/136916757/

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



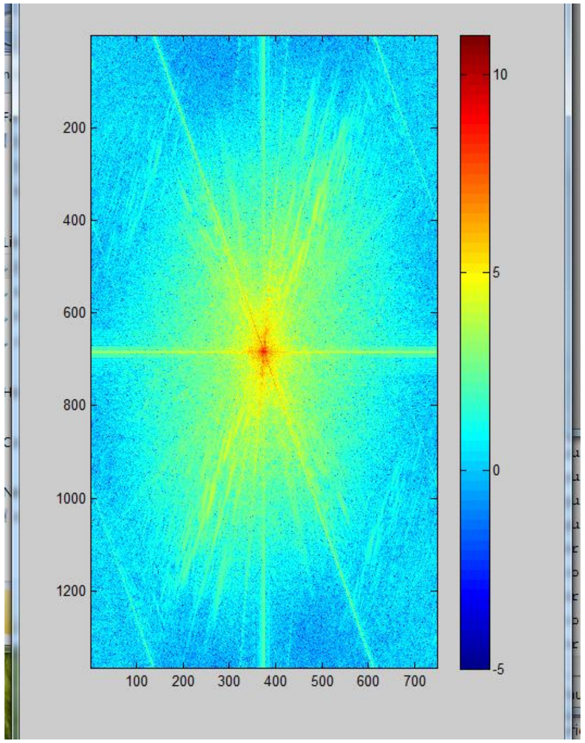
Clues from Human Perception

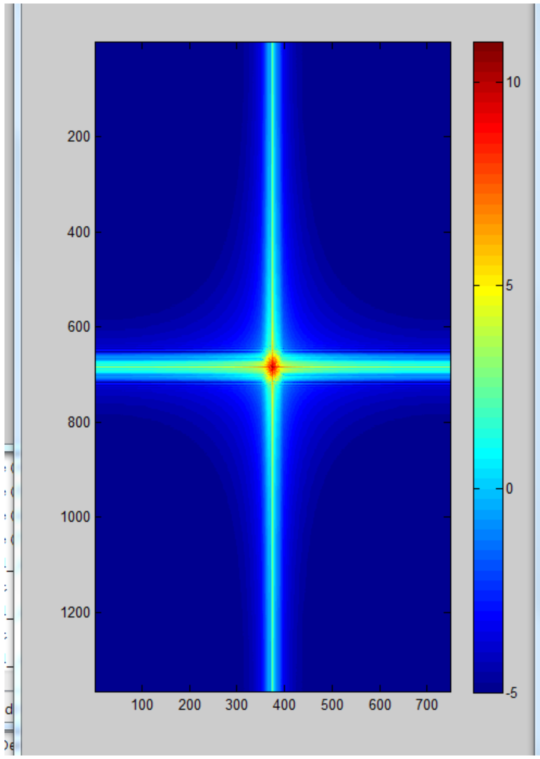
- Early processing in humans filters for various orientations and scales of frequency
- Perceptual cues in the mid-high frequencies dominate perception
- When we see an image from far away, we are effectively subsampling it

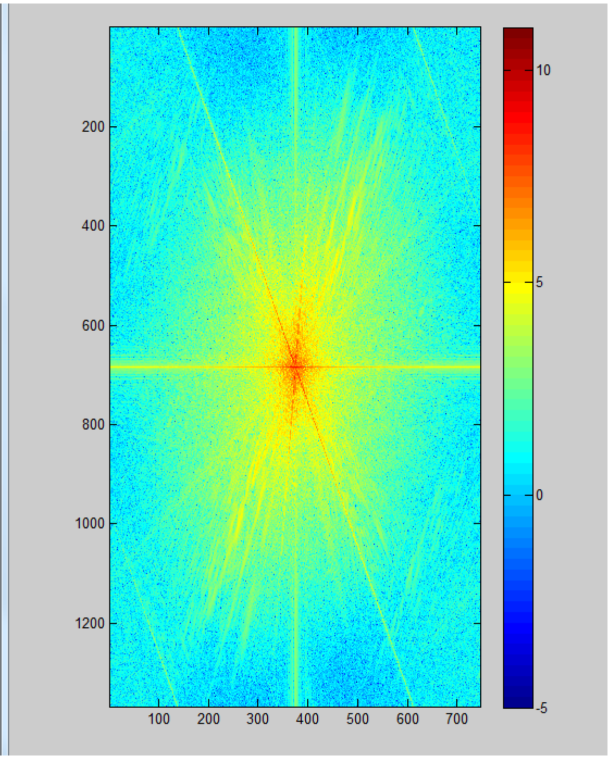

Early Visual Processing: Multi-scale edge and blob filters

Hybrid Image in FFT

Hybrid Image = Low-passed Image + High-passed Image







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")

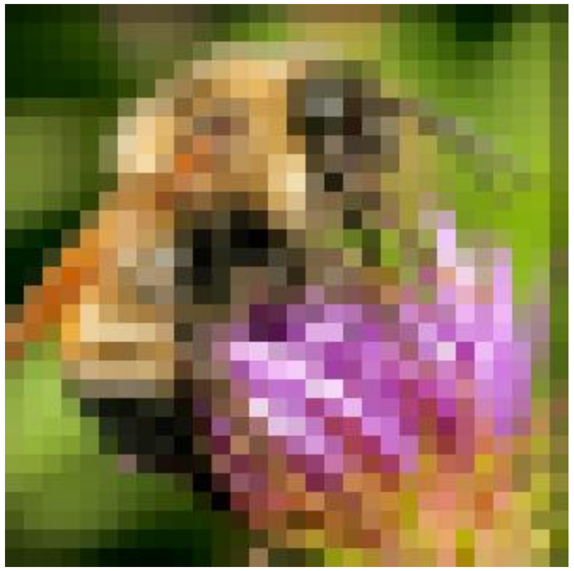
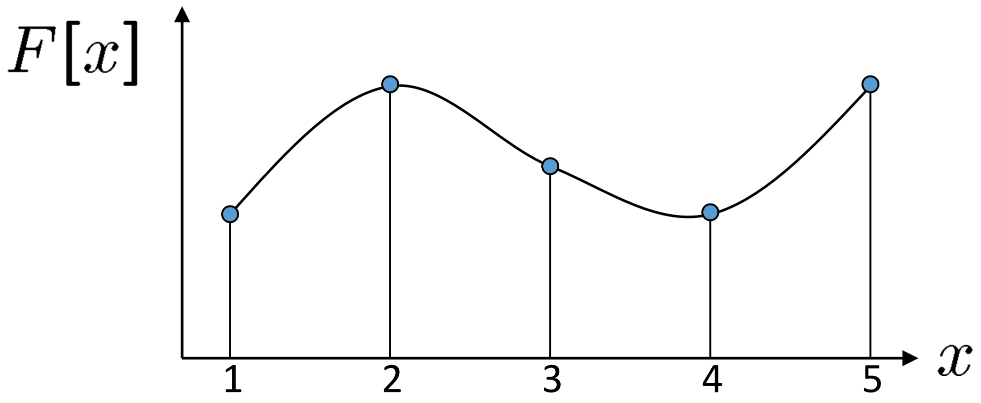


Image interpolation



d = 1 in this example

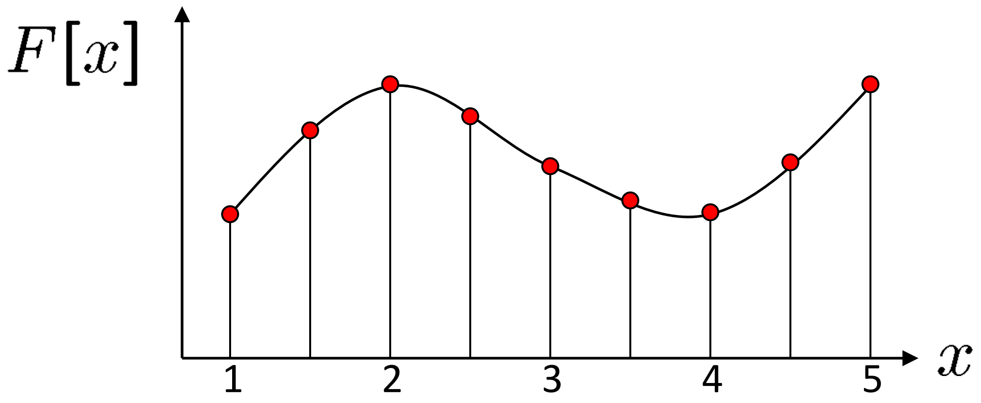
Recall how a digital image is formed

$$F[x,y] = \text{quantize}\{f(xd, yd)\}$$

- 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

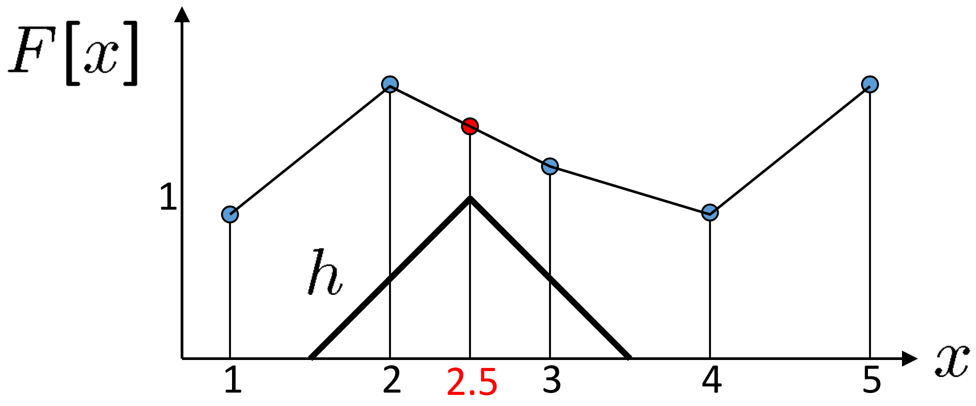
Recall how a digital image is formed

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- 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 ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

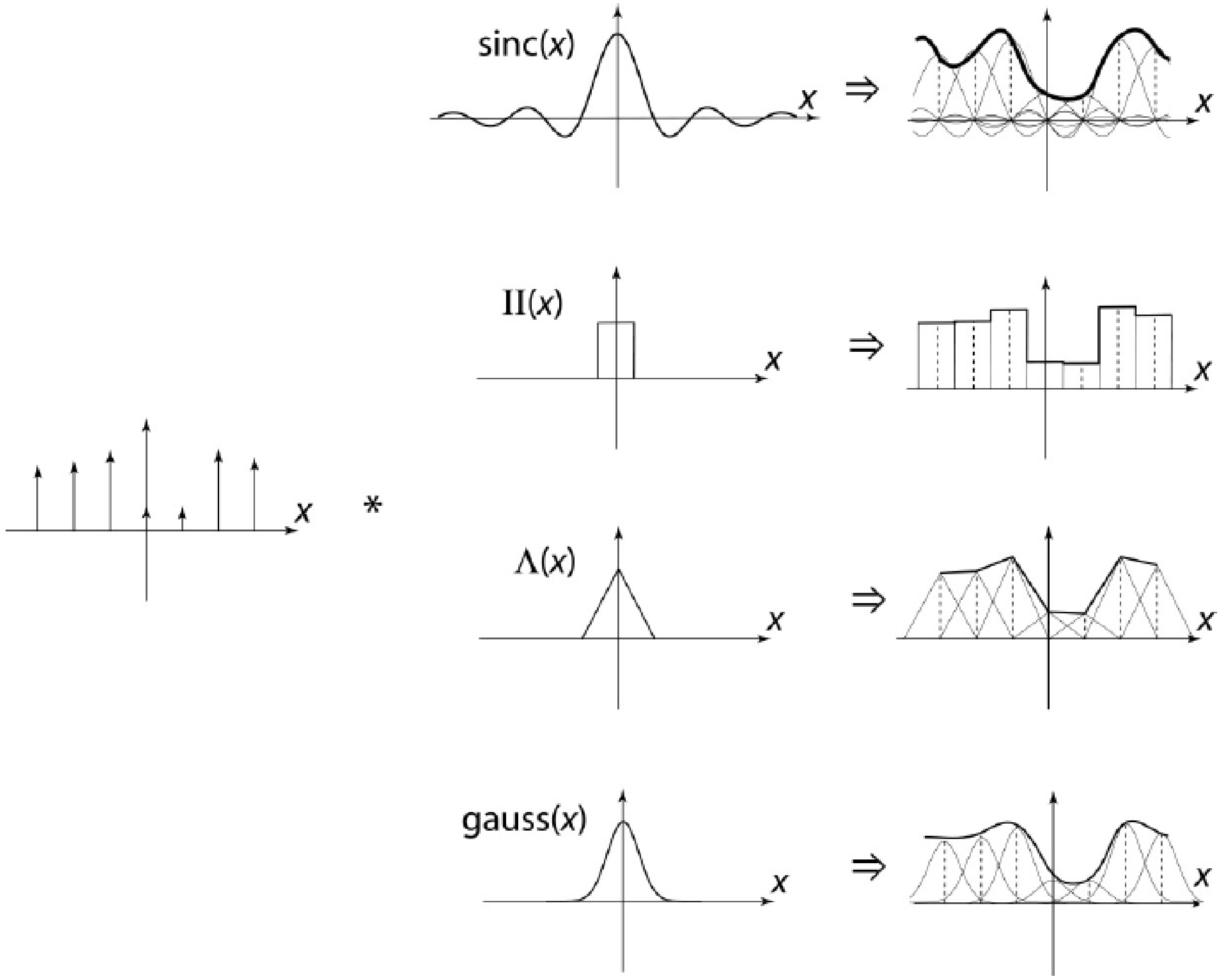
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

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

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

Adapted from: S. Seitz

Image interpolation



"Ideal" reconstruction

Nearest-neighbor interpolation

Linear interpolation

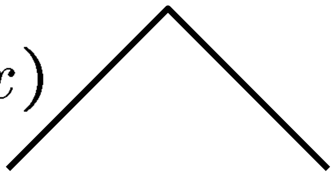
Gaussian reconstruction

Source: B. Curless

Reconstruction filters

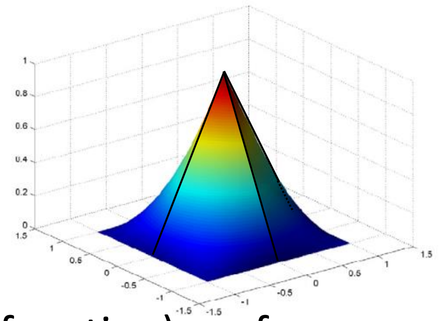
- What does the 2D version of this hat function look like?

$h(x)$



performs
linear interpolation

$h(x,y)$



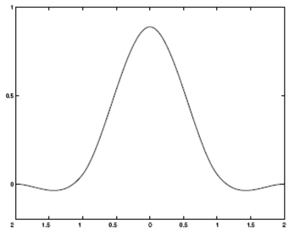
(tent function) performs
bilinear interpolation

Often implemented without cross-correlation

- E.g., http://en.wikipedia.org/wiki/Bilinear_interpolation

Better filters give better resampled images

- **Bicubic** is common choice

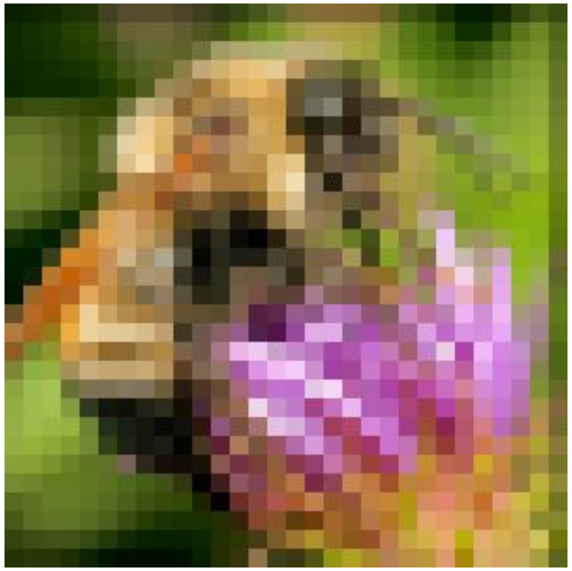


Cubic reconstruction filter

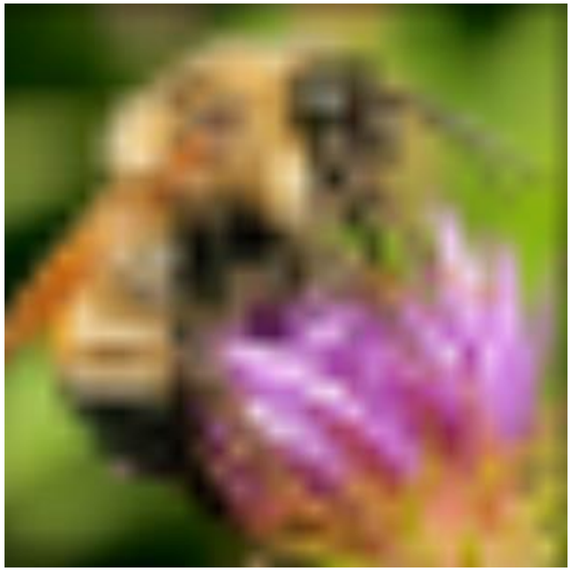
$$r(x)=\frac{1}{6}\begin{cases}(12-9B-6C)|x|^3+(-18+12B+6C)|x|^2+(6-2B) & |x|<1\\((-B-6C)|x|^3+(6B+30C)|x|^2+(-12B-48C)|x|+(8B+24C) & 1\le|x|<2\\0 & \textit{otherwise}\end{cases}$$

Image interpolation

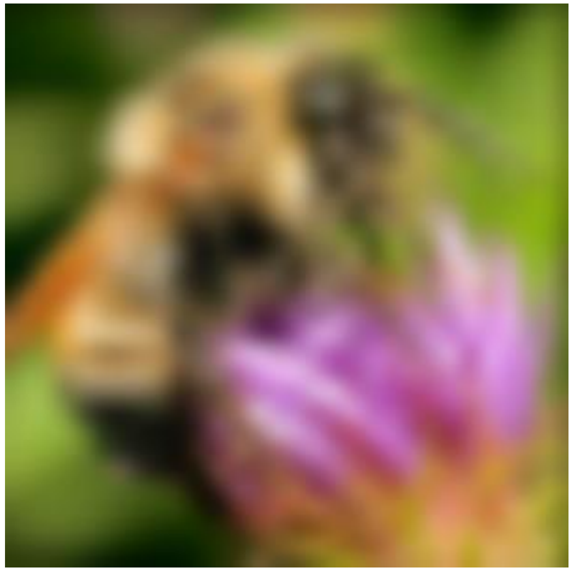
Original image: x 10



Nearest-neighbor interpolation



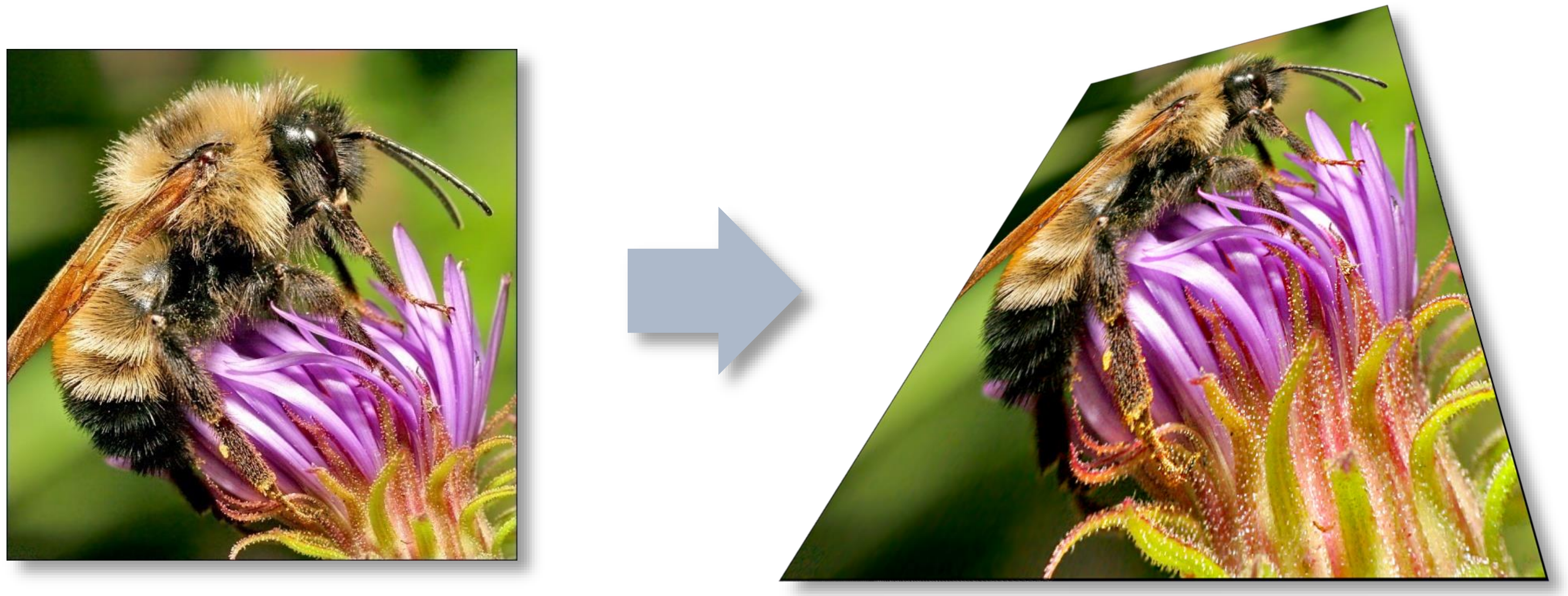
Bilinear interpolation



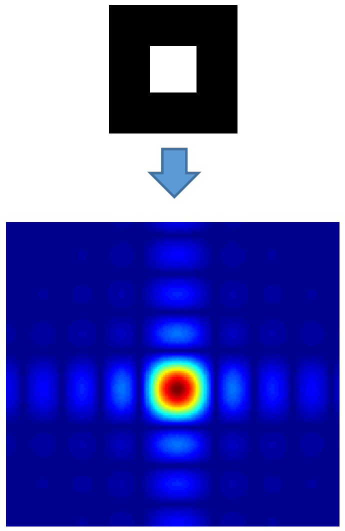
Bicubic interpolation

Image interpolation

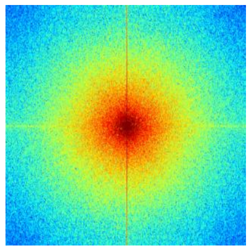
Also used for *resampling*



Things to Remember

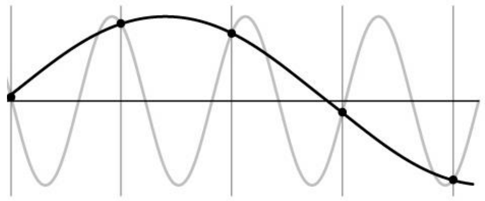


- Sometimes it makes sense to think of images and filtering in the frequency domain
  - Fourier analysis



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

- Images are mostly smooth
  - Basis for compression



- Remember to low-pass before sampling

Thank you

- Enjoy your long weekend!
- Next class:
  - Pyramid, template matching