Fourier, filtering, smoothing, and noise

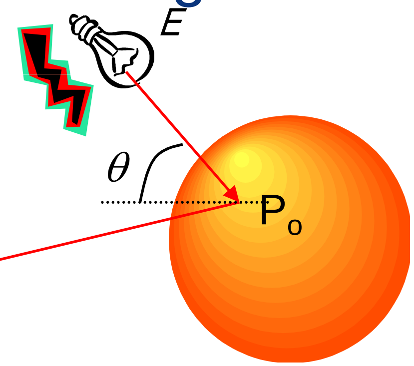
Nuno Vasconcelos

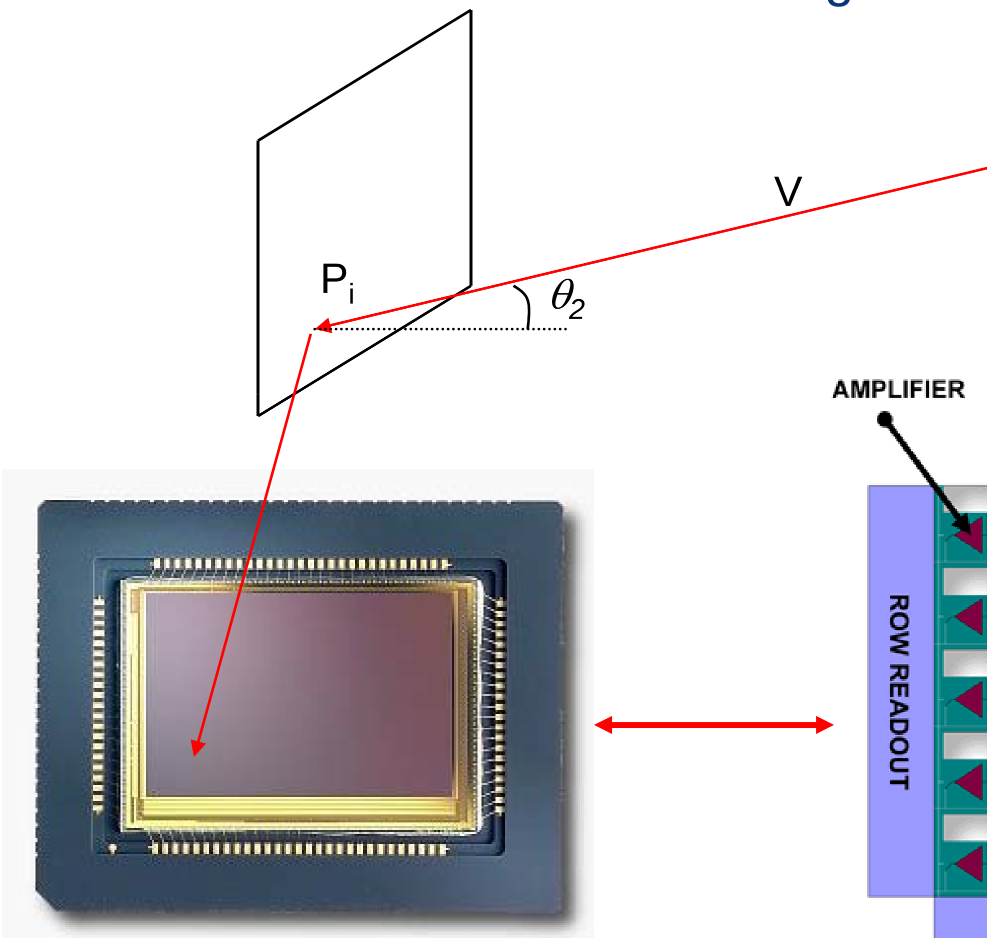
ECE Department, UCSD

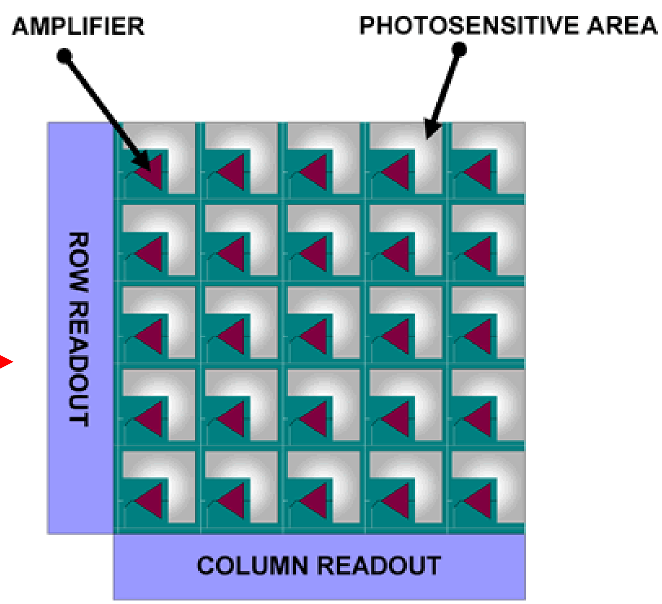
(with thanks to David Forsyth)

Images

- the incident light is collected by an image sensor
  - that transforms it into a 2D signal







2D-DSP

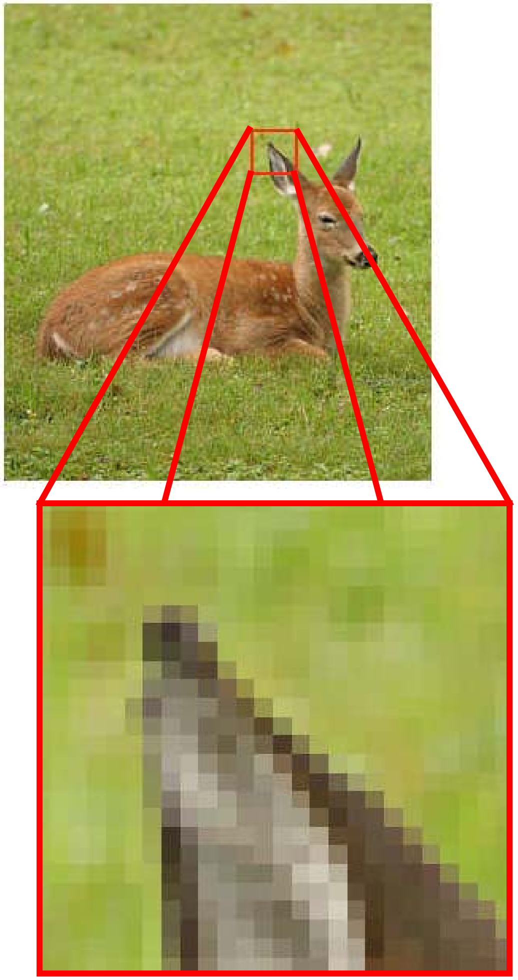
- in summary:
  - image is a *N x M* array of pixels
  - each pixel contains three colors
  - overall, the image is a 2D discrete-space signal
  - each entry is a 3D vector

$$x[n_1, n_2] = (r, g, b), \quad n_1 \in \{0, \ldots, N\}$$
$$n_2 \in \{0, \ldots, M\}$$

  - for simplicity, we consider only single channel images

$$x[n_1, n_2], \quad n_1 \in \{0, \ldots, N\}$$
$$n_2 \in \{0, \ldots, M\}$$

  - but everything extends to color in a straightforward manner



2D convolution

- the operation

$$y[n_1,n_2] = \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} x[k_1,k_2]\, h[n_1-k_1, n_2-k_2]$$

  is the 2D convolution of x and h

  - we will denote it by

$$y[n_1,n_2] = x[n_1,n_2] * h[n_1,n_2]$$

- this is of great practical importance:
  - for an LSI system the response to any input can be obtained by the convolution with this impulse response
  - the IR fully characterizes the system
  - it is all that I need to measure

Separable systems

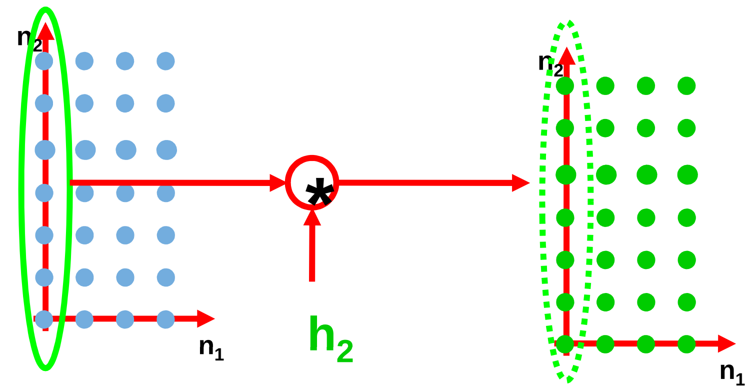
- **Definition:** a system is separable if and only if its impulse response is a separable sequence

$$h[n_1, n_2] = h_1[n_1] \times h_2[n_2]$$

- in this case the convolution simplifies
- **step1)** for every k_1,
  - $f[k_1,n_2]$ is 1D convolution of $x[k_1,n_2]$ and $h_2[n_2]$

$$f[k_1, n_2] = x[k_1, n_2] * h_2[n_2]$$

  - which means: "convolve the columns of x with h_2 to obtain columns of f"



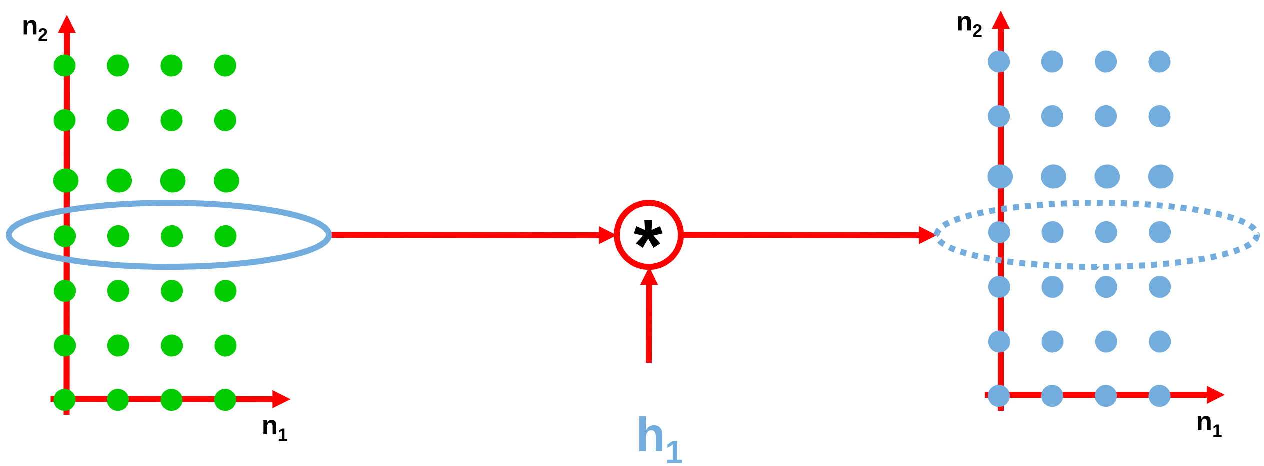
$x[n_1, n_2]$ $f[n_1, n_2]$

Separable systems

- **step2)** for every n_2,
  - $y[n_1,n_2]$ is 1D convolution of $f[n_1,n_2]$ and $h_1[n_1]$

$$y[n_1,n_2] = f[n_1,n_2] * h_1[n_1]$$

  - which means: "convolve the rows of f with h_1 to obtain rows of y"



$f[n_1,n_2]$ $y[n_1,n_2]$

The Discrete-Space Fourier Transform

- is, once again, a straightforward extension of the 1D Discrete-Time Fourier Transform

$$X(\omega_1, \omega_2) = \sum_{n_1} \sum_{n_2} x[n_1, n_2] e^{-j\omega_1 n_1} e^{-j\omega_2 n_2}$$

$$x[n_1, n_2] = \frac{1}{(2\pi)^2} \iint X(\omega_1, \omega_2) e^{j\omega_1 n_1} e^{j\omega_2 n_2} d\omega_1 d\omega_2$$

- properties:
  - basically the same as in 1D (see table in Lim, page 25)
  - only novelty is separability (homework)

$$x[n_1, n_2] = x_1[n_1] x_2[n_2] \leftrightarrow X(\omega_1, \omega_2) = X_1(\omega_1) X_2(\omega_2)$$

Properties of the DSFT

$$x(n_1, n_2) \longleftrightarrow X(\omega_1, \omega_2)$$
$$y(n_1, n_2) \longleftrightarrow Y(\omega_1, \omega_2)$$

Property 1. *Linearity*

$$ax(n_1, n_2) + by(n_1, n_2) \longleftrightarrow aX(\omega_1, \omega_2) + bY(\omega_1, \omega_2)$$

Property 2. *Convolution*

$$x(n_1, n_2) * y(n_1, n_2) \longleftrightarrow X(\omega_1, \omega_2)Y(\omega_1, \omega_2)$$

Property 3. *Multiplication*

$$x(n_1, n_2)y(n_1, n_2) \longleftrightarrow X(\omega_1, \omega_2) \circledast Y(\omega_1, \omega_2)$$

$$= \frac{1}{(2\pi)^2} \int_{\theta_1=-\pi}^{\pi} \int_{\theta_2=-\pi}^{\pi} X(\theta_1, \theta_2)Y(\omega_1 - \theta_1, \omega_2 - \theta_2)\, d\theta_1\, d\theta_2$$

Property 4. *Separable Sequence*

$$x(n_1, n_2) = x_1(n_1)x_2(n_2) \longleftrightarrow X(\omega_1, \omega_2) = X_1(\omega_1)X_2(\omega_2)$$

Property 5. *Shift of a Sequence and a Fourier Transform*

(a) $x(n_1 - m_1, n_2 - m_2) \longleftrightarrow X(\omega_1, \omega_2)e^{-j\omega_1 m_1}e^{-j\omega_2 m_2}$

(b) $e^{j\nu_1 n_1}e^{j\nu_2 n_2}x(n_1, n_2) \longleftrightarrow X(\omega_1 - \nu_1, \omega_2 - \nu_2)$

Property 6. *Differentiation*

(a) $-jn_1x(n_1, n_2) \longleftrightarrow \dfrac{\partial X(\omega_1, \omega_2)}{\partial \omega_1}$

(b) $-jn_2x(n_1, n_2) \longleftrightarrow \dfrac{\partial X(\omega_1, \omega_2)}{\partial \omega_2}$

Properties of the DSFT

Property 7. *Initial Value and DC Value Theorem*

(a) $$x(0,0) = \frac{1}{(2\pi)^2} \int_{\omega_1=-\pi}^{\pi} \int_{\omega_2=-\pi}^{\pi} X(\omega_1, \omega_2)\, d\omega_1\, d\omega_2$$

(b) $$X(0,0) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} x(n_1, n_2)$$

Property 8. *Parseval's Theorem*

(a) $$\sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} x(n_1, n_2) y^*(n_1, n_2) = \frac{1}{(2\pi)^2} \int_{\omega_1=-\pi}^{\pi} \int_{\omega_2=-\pi}^{\pi} X(\omega_1, \omega_2) Y^*(\omega_1, \omega_2)\, d\omega_1\, d\omega_2$$

(b) $$\sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} |x(n_1, n_2)|^2 = \frac{1}{(2\pi)^2} \int_{\omega_1=-\pi}^{\pi} \int_{\omega_2=-\pi}^{\pi} |X(\omega_1, \omega_2)|^2\, d\omega_1\, d\omega_2$$

Properties of the DSFT

Property 9. <u>*Symmetry Properties*</u>

(a) $x(-n_1, n_2) \longleftrightarrow X(-\omega_1, \omega_2)$

(b) $x(n_1, -n_2) \longleftrightarrow X(\omega_1, -\omega_2)$

(c) $x(-n_1, -n_2) \longleftrightarrow X(-\omega_1, -\omega_2)$

(d) $x^*(n_1, n_2) \longleftrightarrow X^*(-\omega_1, -\omega_2)$

(e) $x(n_1, n_2)$: real $\longleftrightarrow X(\omega_1, \omega_2) = X^*(-\omega_1, -\omega_2)$

$X_R(\omega_1, \omega_2)$, $|X(\omega_1, \omega_2)|$: even (symmetric with respect to the origin)

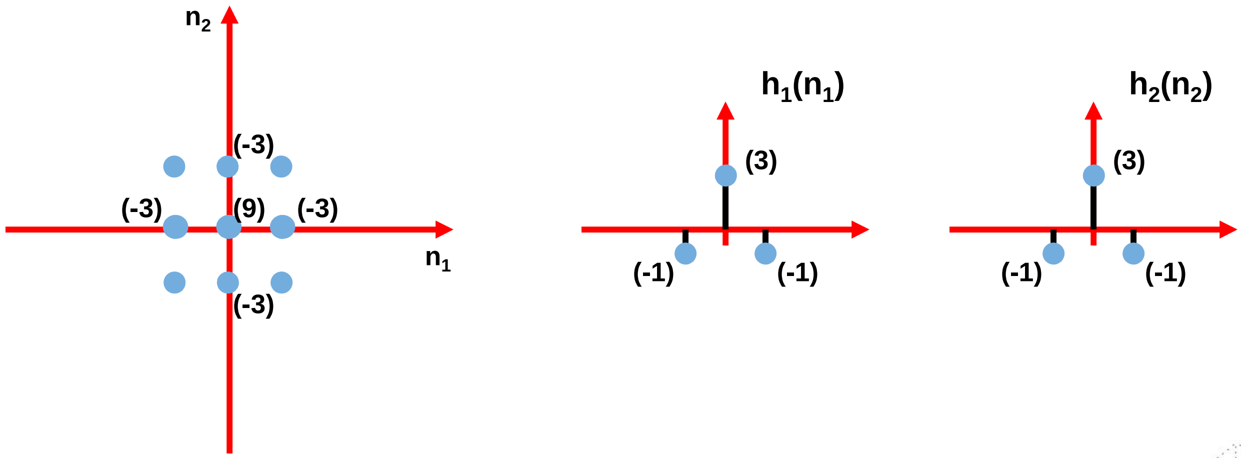
$X_I(\omega_1, \omega_2)$, $\theta_x(\omega_1, \omega_2)$: odd (antisymmetric with respect to the origin)

(f) $x(n_1, n_2)$: real and even $\longleftrightarrow X(\omega_1, \omega_2)$: real and even

(g) $x(n_1, n_2)$: real and odd $\longleftrightarrow X(\omega_1, \omega_2)$: pure imaginary and odd

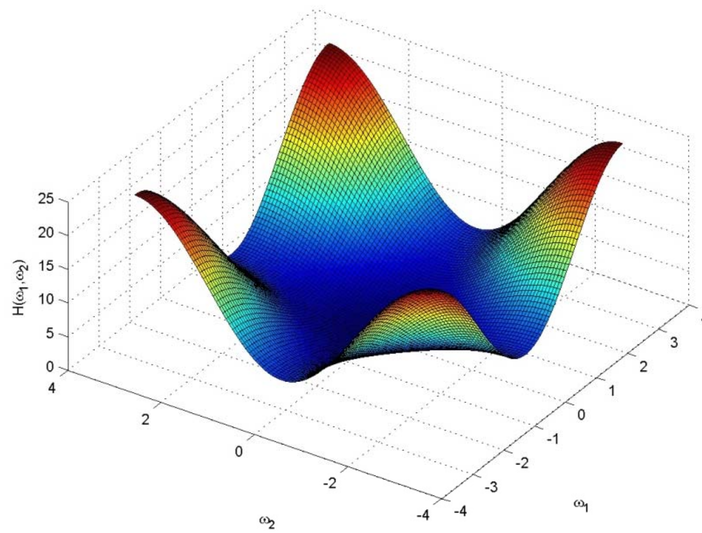
Example

- consider the separable impulse response



- frequency response

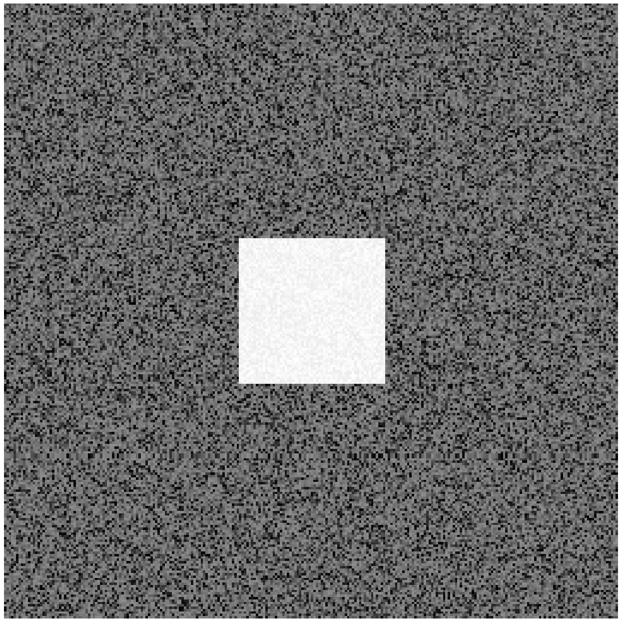
$$H(\omega_1,\omega_2) = H_1(\omega_1)H_2(\omega_2)$$
$$= (3 - 2\cos\varpi_1)(3 - 2\cos\varpi_2)$$

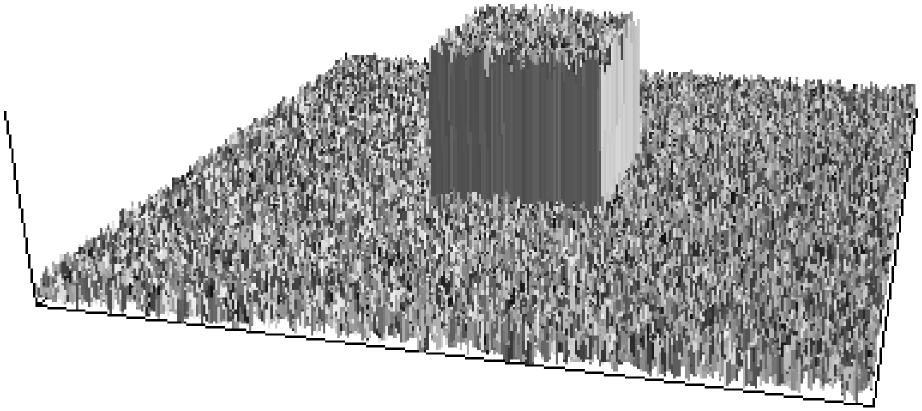


- note that:
  - this system is a high-pass filter
  - “diagonal” frequencies are enhanced

Examples

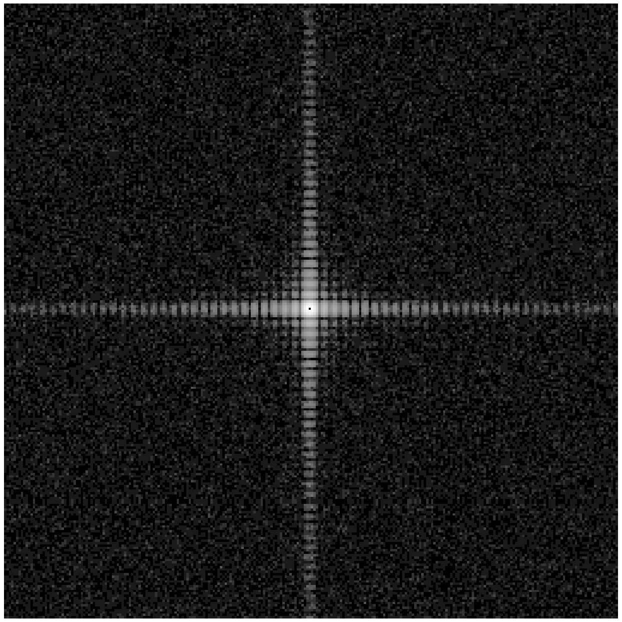
- what do filtered images look like?
  - here is a noisy image
  - a light square against dark background, plus noise

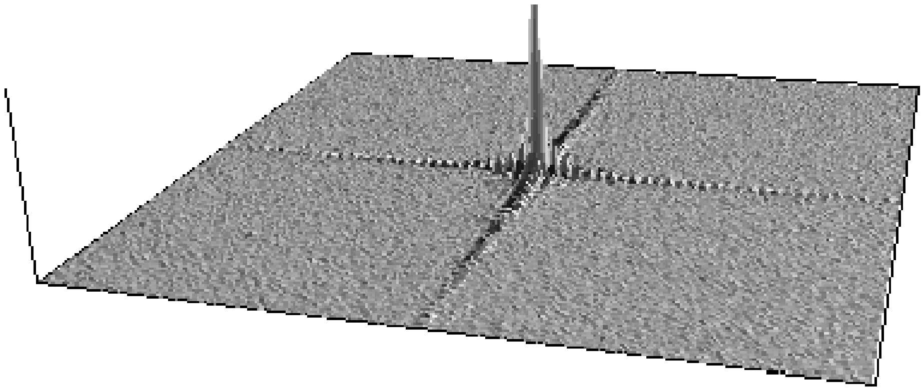




Examples

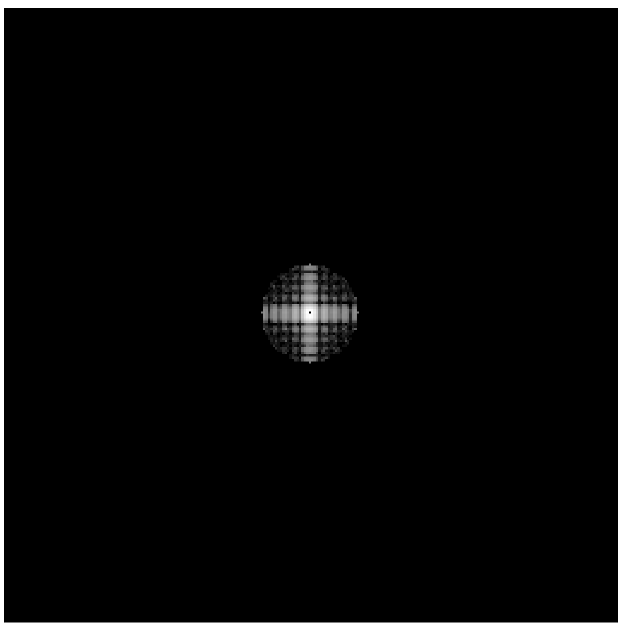
- what do filtered images look like?
  - here is the magnitude of its DSFT (origin at center), it contains:
  - a peak at the center,
  - some background signal at all frequencies,
  - a cross-like pattern that goes from low to high frequencies
  - why does it look like this?

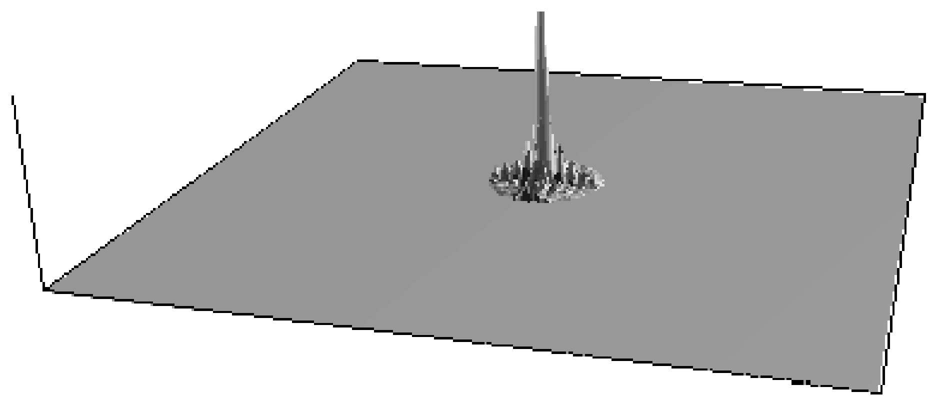




Examples

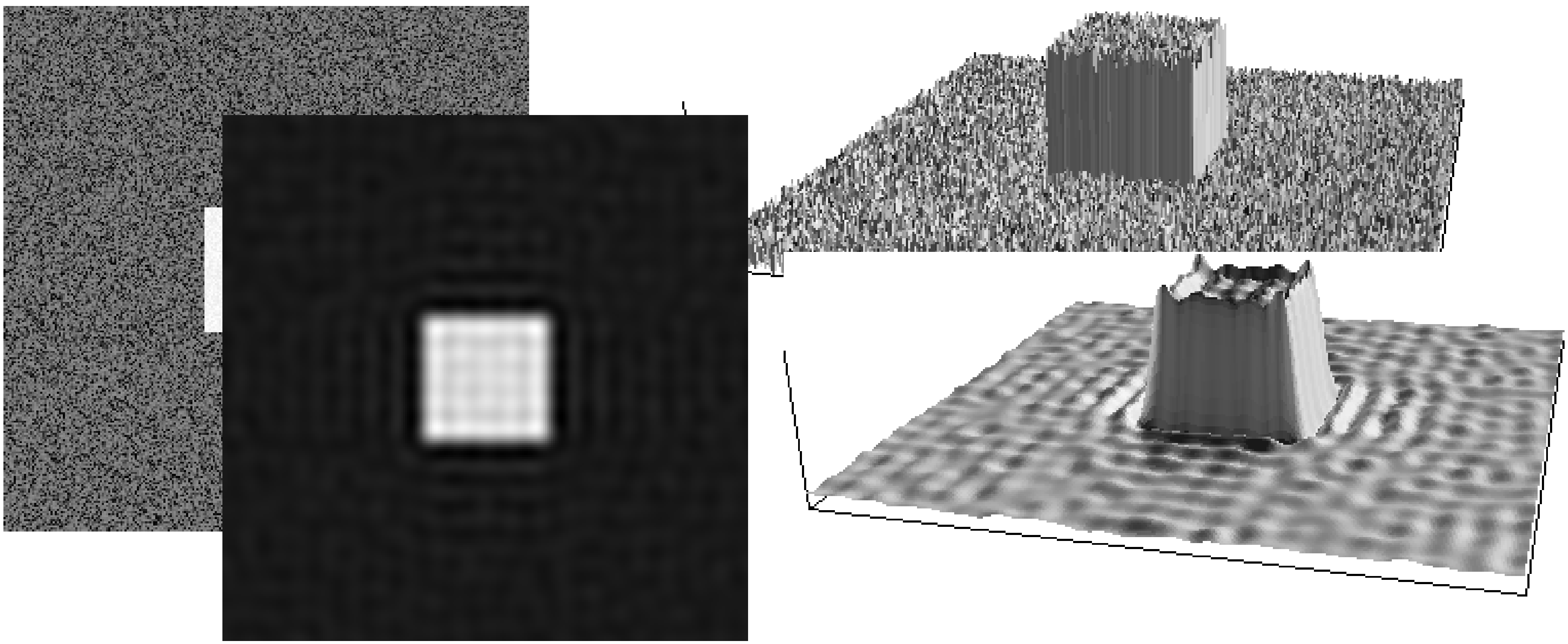
- one way to find out is to filter and reconstruct the image
  - we simulate the ideal low-pass filter by
  - removing all signal components outside a circle in the frequency domain
  - this is what the spectrum looks like
  - this gets rid of the background signal that covers all frequencies





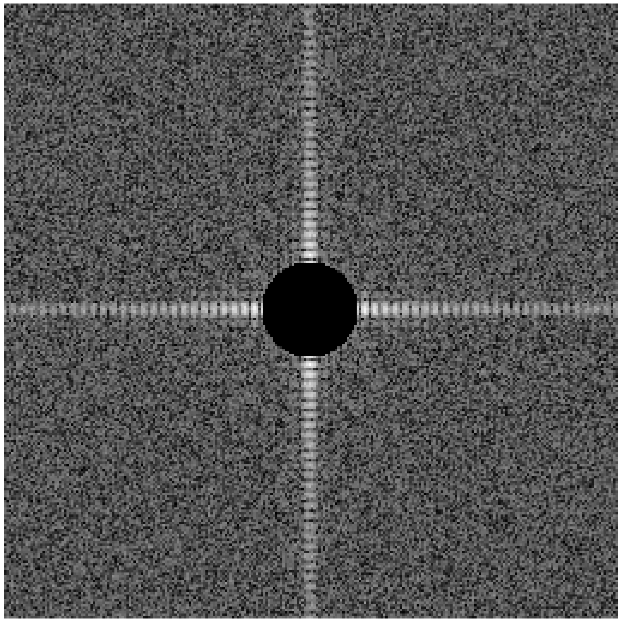
Examples

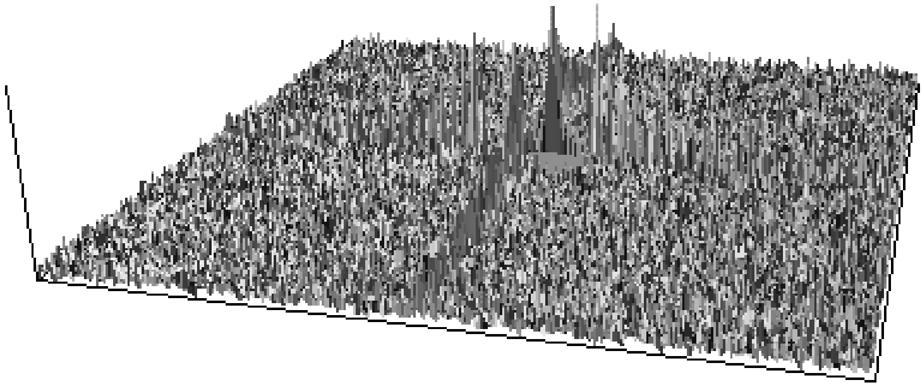
- this is the resulting image
  - the component we removed was due to the noise
  - “white” noise has energy at all frequencies
  - notice that there are some artifacts (i.e. ringing) in the reconstructed image



Examples

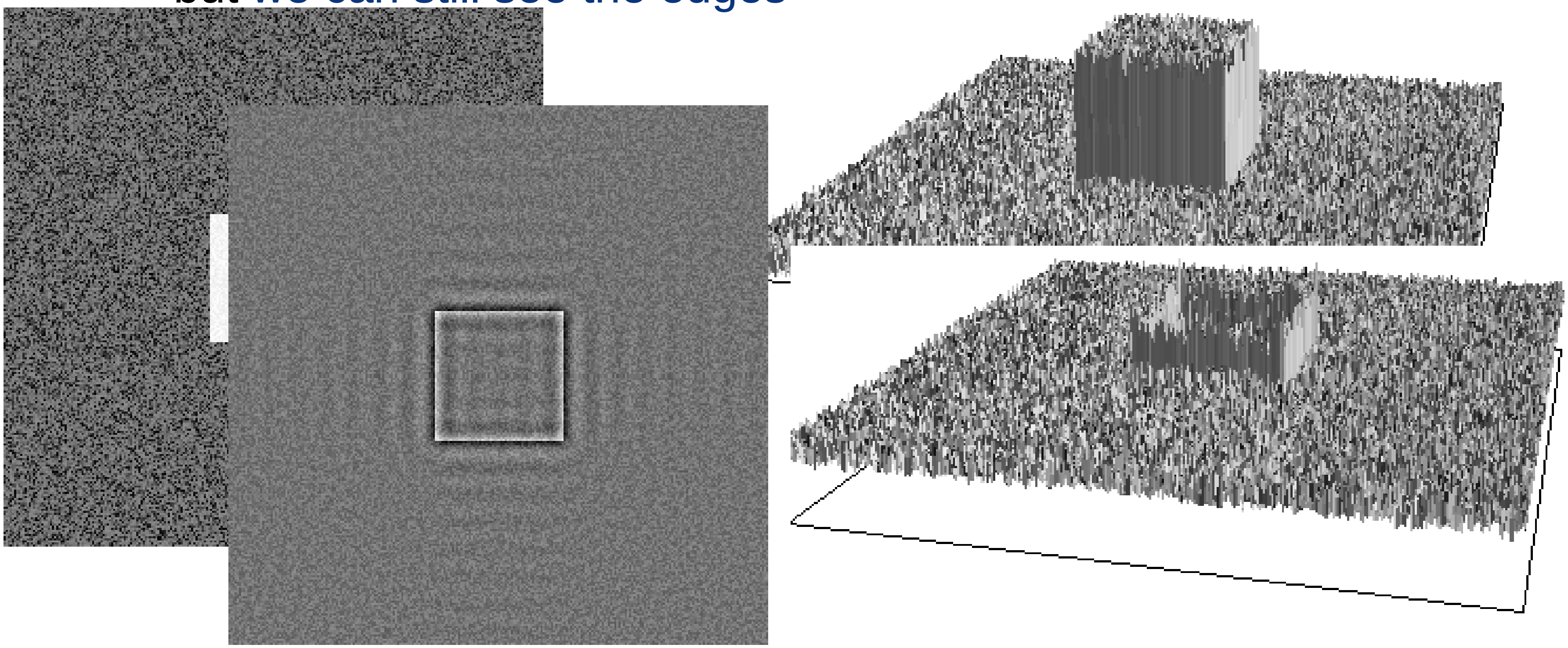
- what about the stuff other than noise?
  - let's high-pass by removing everything inside the circle





Examples

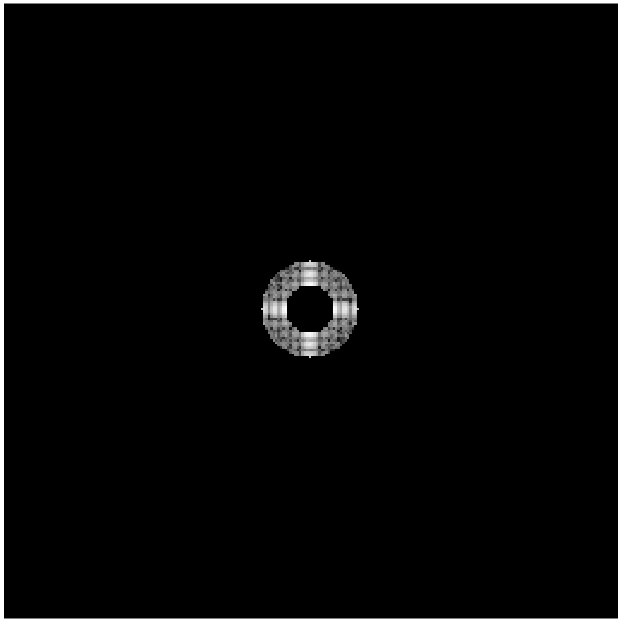
- this is the resulting image
  - we now get mostly noise, as expected
  - note that the square has mostly gone away
  - this means that the flat part is low-frequency
  - but we can still see the edges

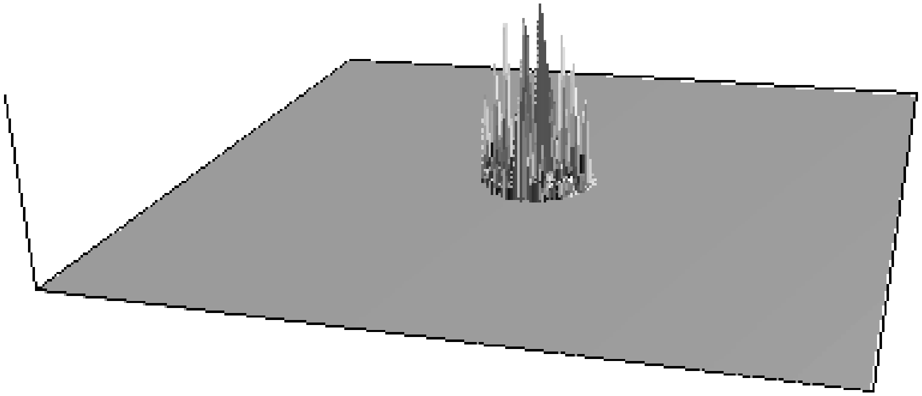


Examples

▸ this is interesting

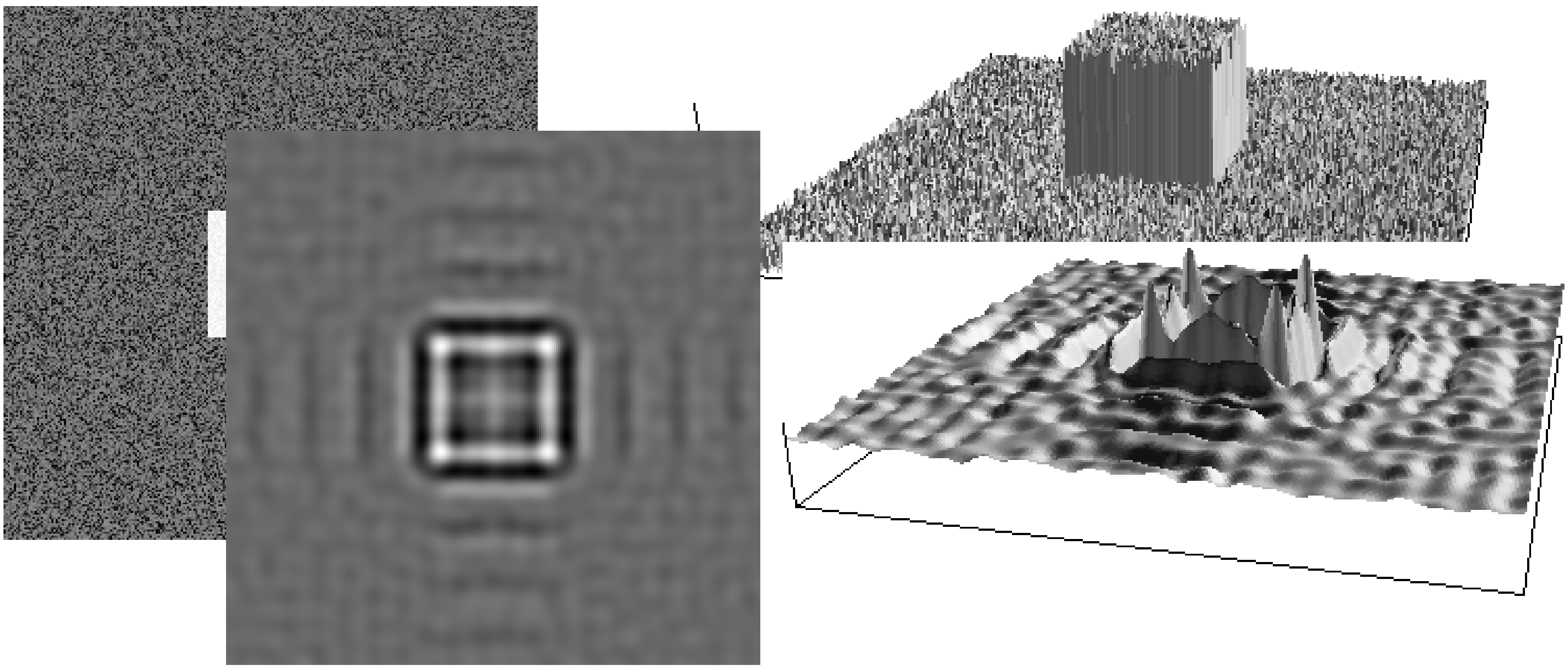
- the edges are not only low-pass
- maybe they are the reason for the cross-shaped pattern
- to check we band-pass





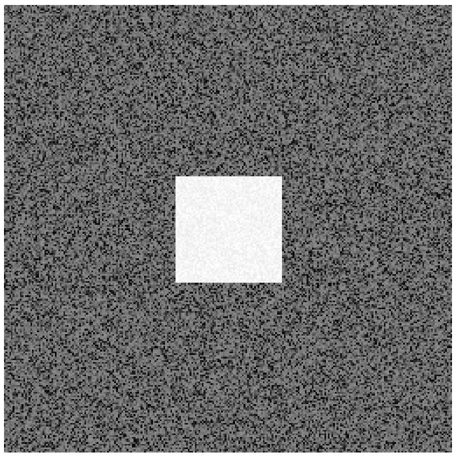
Examples

- this is the resulting image
  - we now get mostly the edges
  - we were right, the edges cause the cross-shaped pattern
  - note that the edges are very hard to filter out



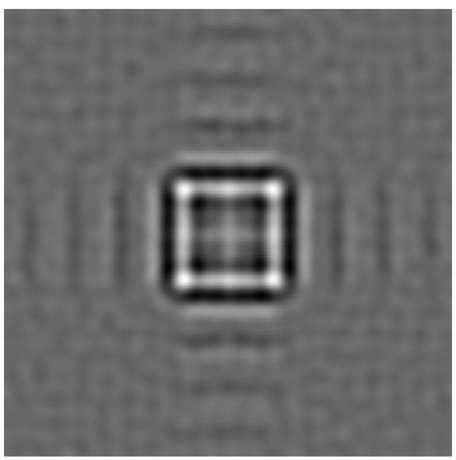
Examples

- this is one of the fundamental properties of images:
  - edges have energy at all frequencies



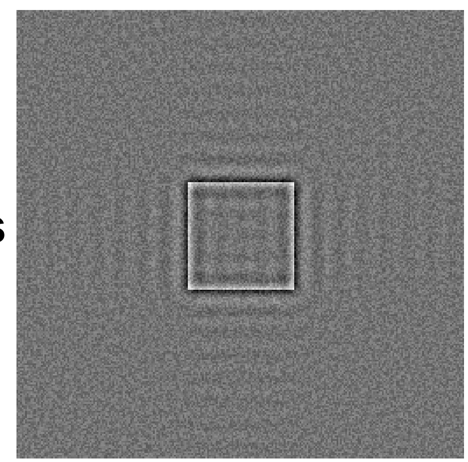
original

low-pass



band-pass

high-pass



Linear Filtering

- image smoothing is implemented with linear filters
- given an image $x(n_1,n_2)$, filtering is the process of convolving it with a kernel $h(n_1,n_2)$

$$y(n_1,n_2) = \sum_{k_1 k_2} x(k_1,k_2)h(n_1-k_1,n_2-k_2)$$

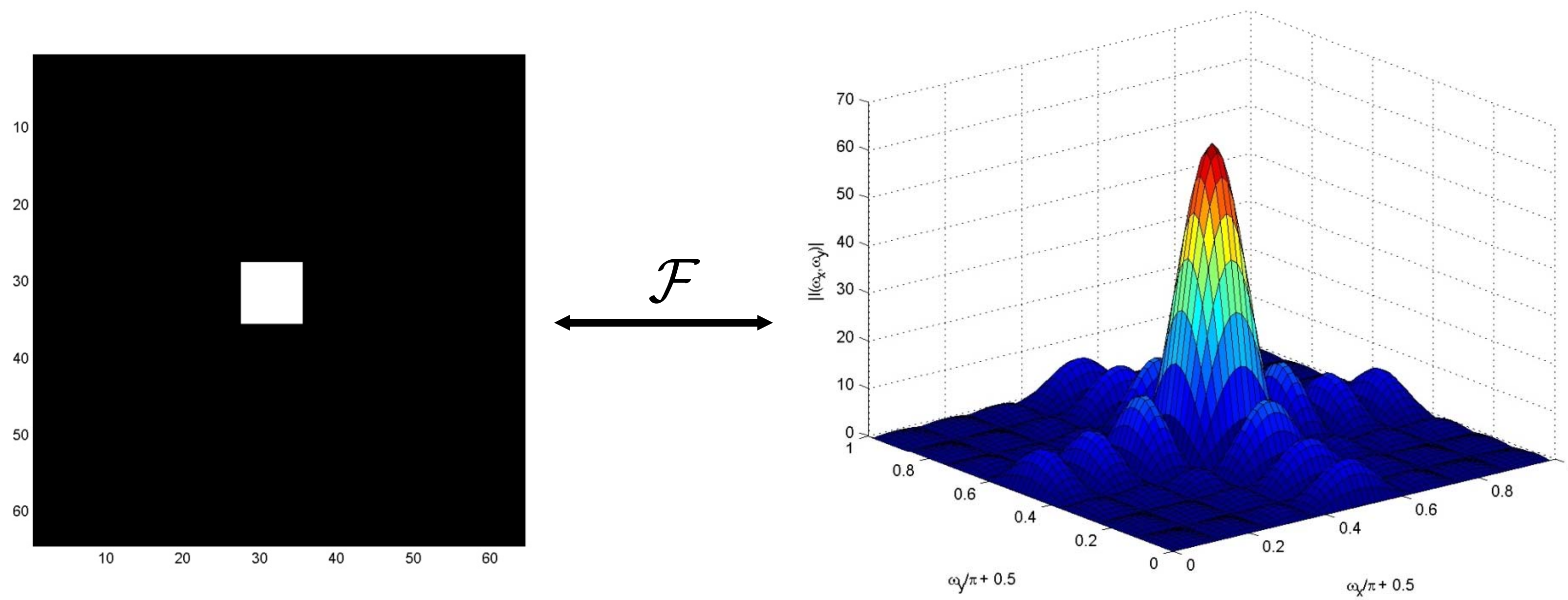
- some very common operations in image processing are nothing but filtering, e.g.
  - smoothing an image by low-pass filtering
  - contrast enhancement by high pass filtering
  - finding image derivatives
  - noise reduction

Popular filters

- box function

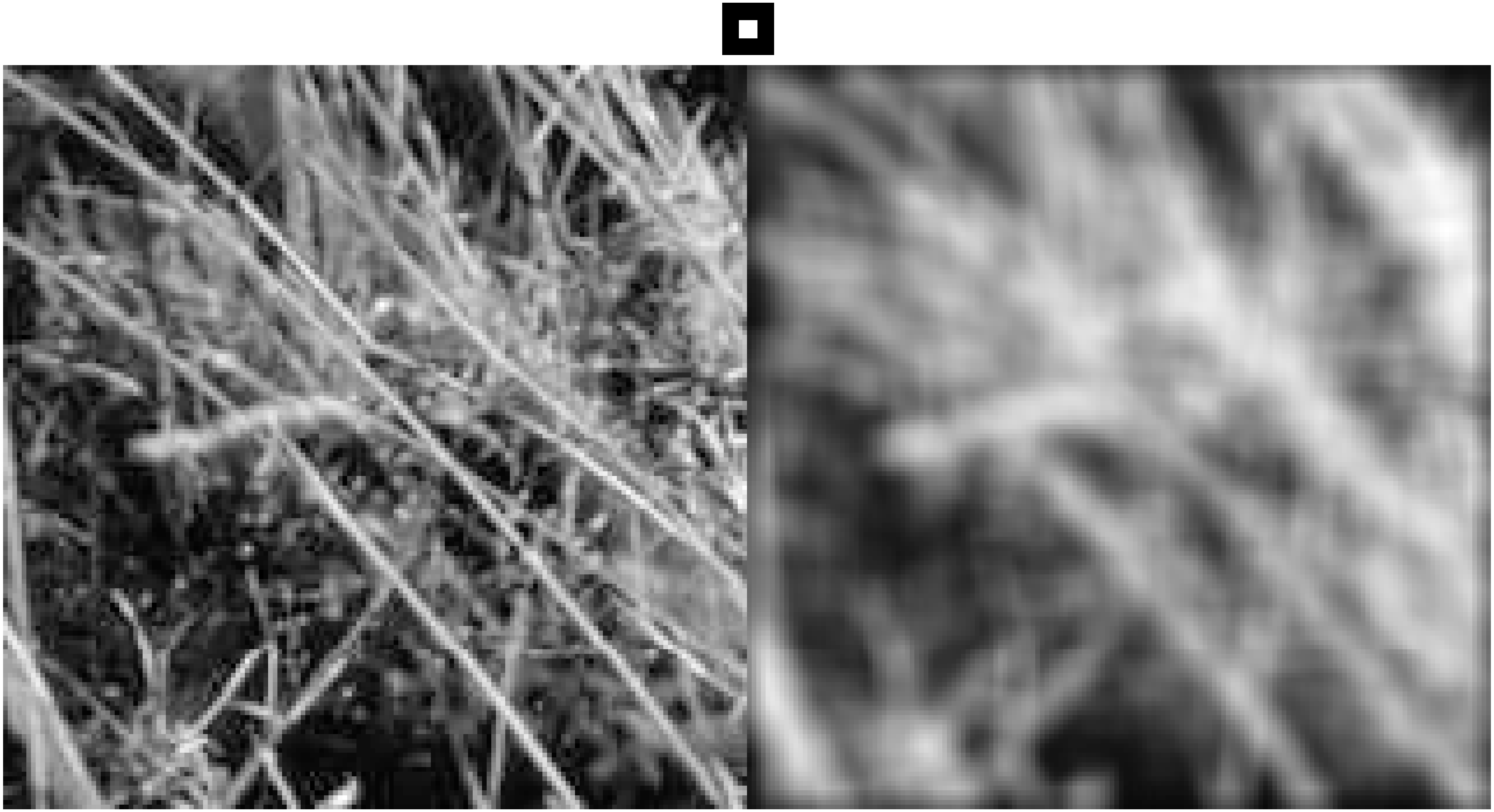
$$R_{N_1 x N_2}(n_1, n_2) = \begin{cases} 1, & 0 \le n_1 \le N_1 - 1, 0 \le n_2 \le N_2 - 1 \\ 0 & otherwise \end{cases}$$

- Fourier transform of a box is the sinc, low-pass filter



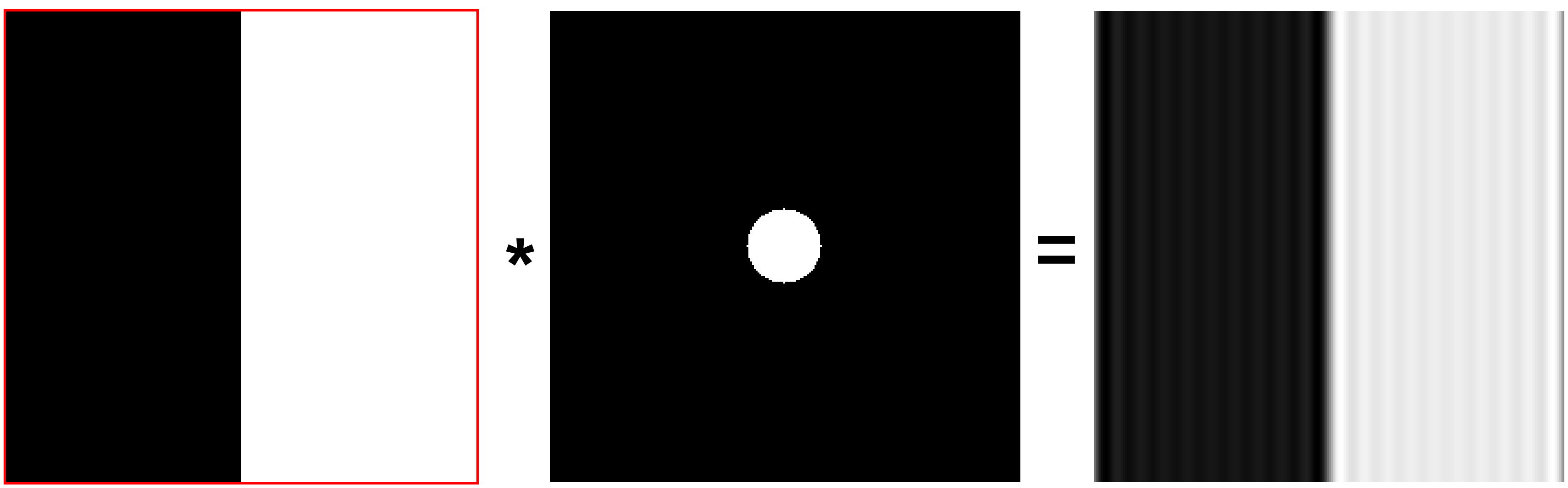
- side-lobes produce artifacts, smoothed image does not look like the result of defocusing

Example: Smoothing by Averaging



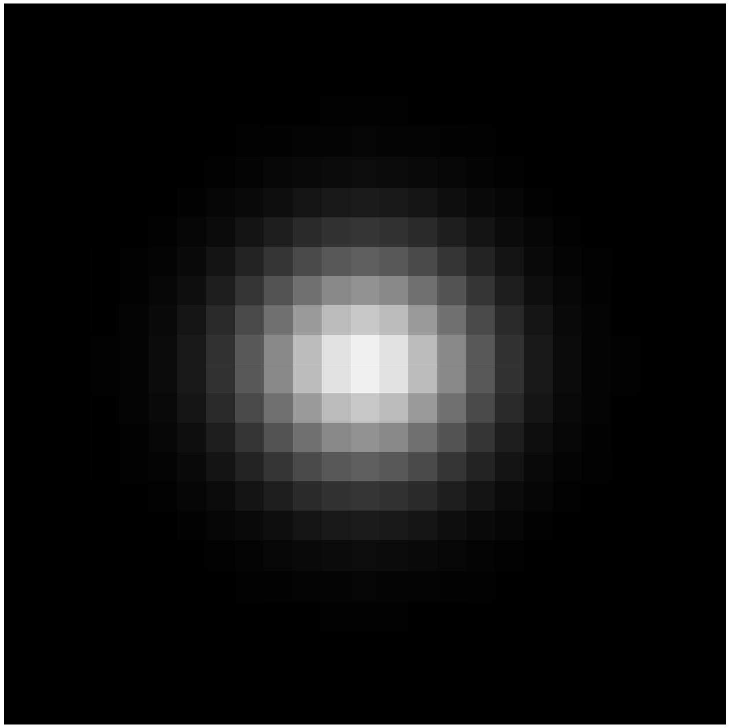
Smoothing by averaging

- the filtered image has a lot of ringing
- this is due to the very sharp edges of the filter
  - the example below shows this more clearly by convolving a synthetic image with a sharp filter
  - note that the problem is not the shape of the filter but the sharpness of the edges



Camera defocusing

- if you point an out-of-focus camera at a very small white light (e.g. a light-bulb) at night, you get something like this
- the light can be thought of as an impulse
- this must be the impulse response
- well approximated by a Gaussian
- more natural filter for image blur than the box



$$h(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

The Gaussian

- the discrete space version is

$$h(n_1, n_2) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{n_1{}^2 + n_2{}^2}{2\sigma^2} \right)$$

- obviously separable

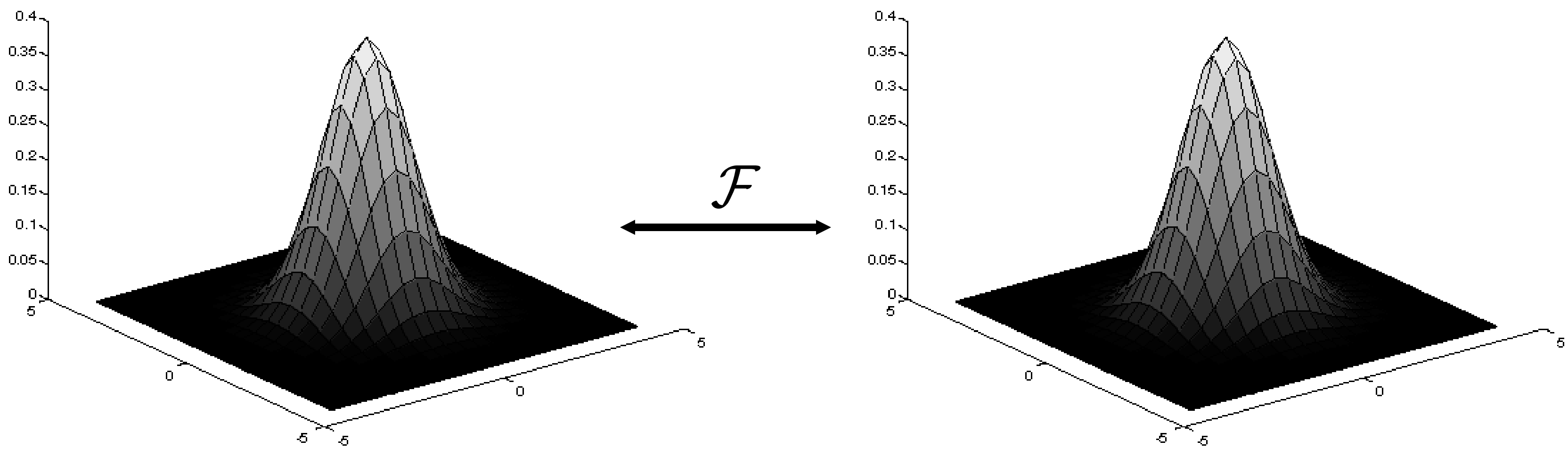
$$h(n_1, n_2) = \underbrace{\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{n_1^2}{2\sigma^2}}}_{h(n_1)} \times \underbrace{\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{n_2^2}{2\sigma^2}}}_{h(n_2)}$$

- $h(n_1,n_2)$ has Fourier transform

$$H(\varpi_1, \varpi_2) = \exp\left(-\frac{\sigma^2 (\varpi_1{}^2 + \varpi_2{}^2)}{2} \right)$$

The Gaussian filter

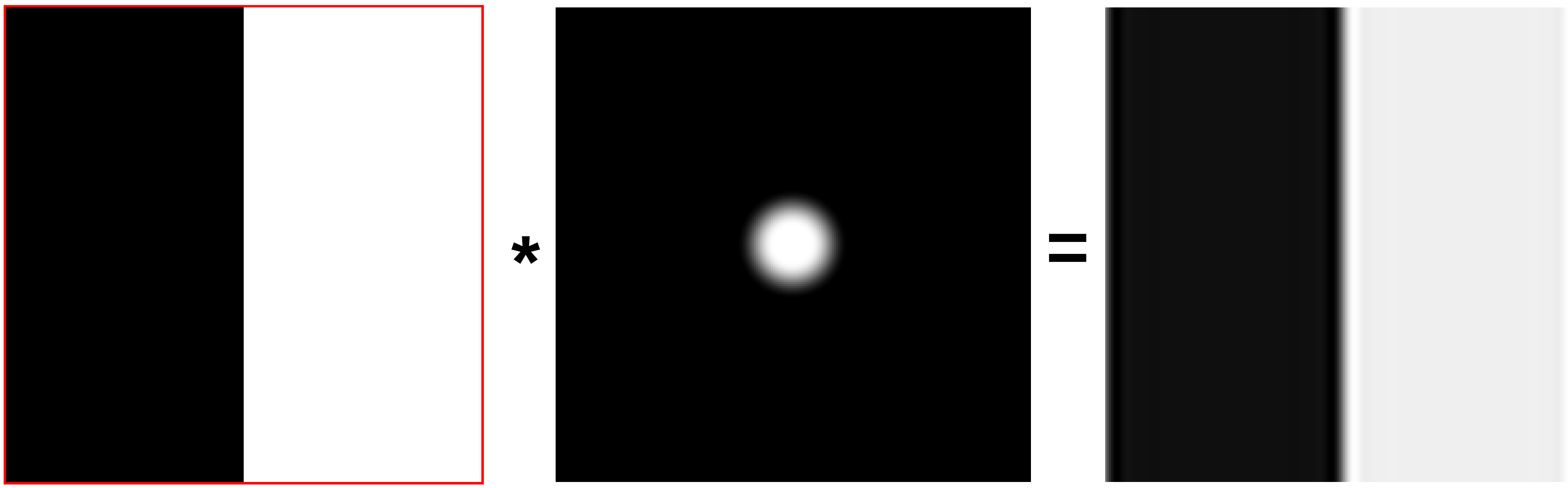
- the Fourier transform of a Gaussian is a Gaussian $(\sigma_x,\sigma_y) \propto (1/\sigma_{w1},1/\sigma_{w2})$



- note that there are no annoying side-lobes

Smoothing by averaging

- when the image is convolved with the Gaussian filter
- the output has very little ringing



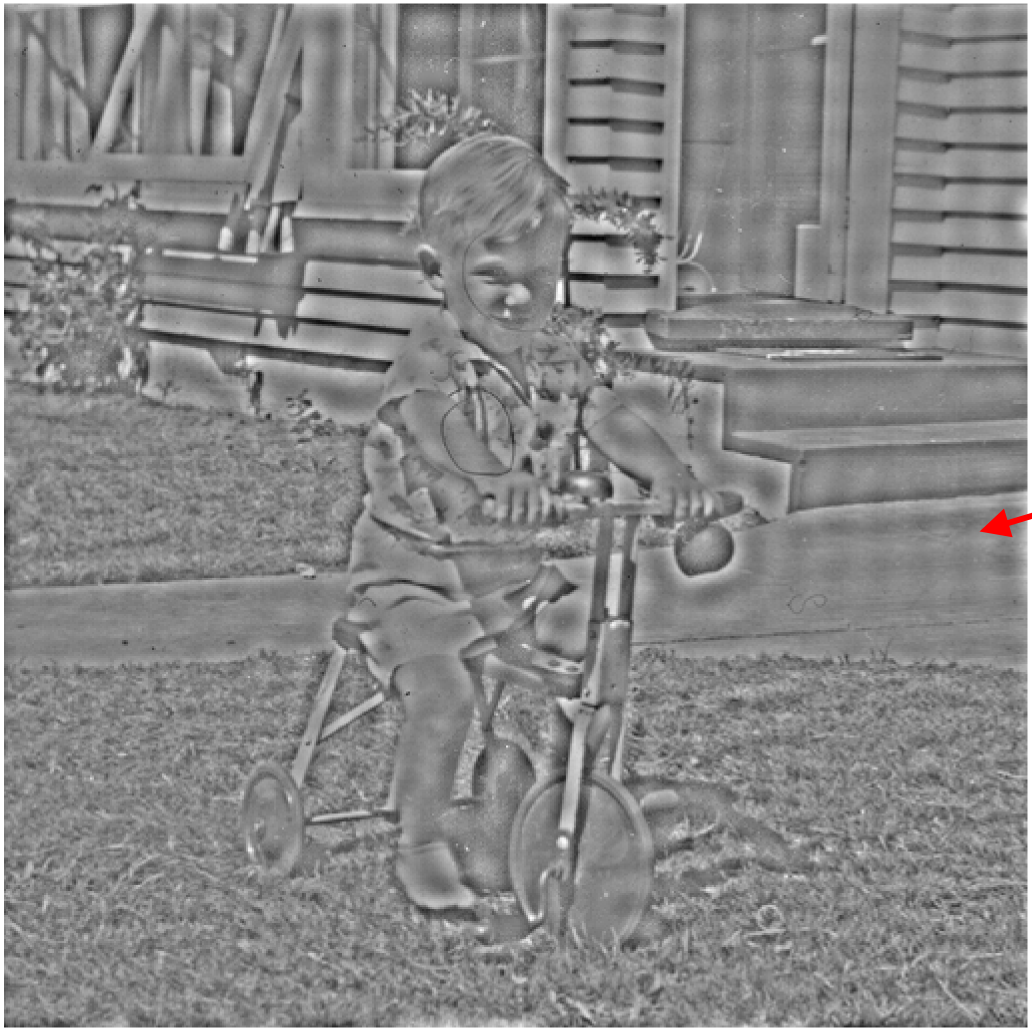
- note:
  - the effects of ringing are most noticeable in the flat image regions

Smoothing by averaging

- e.g. consider the result of filtering this image with the two filters

Smoothing by averaging

- this is the result for the sharper filter



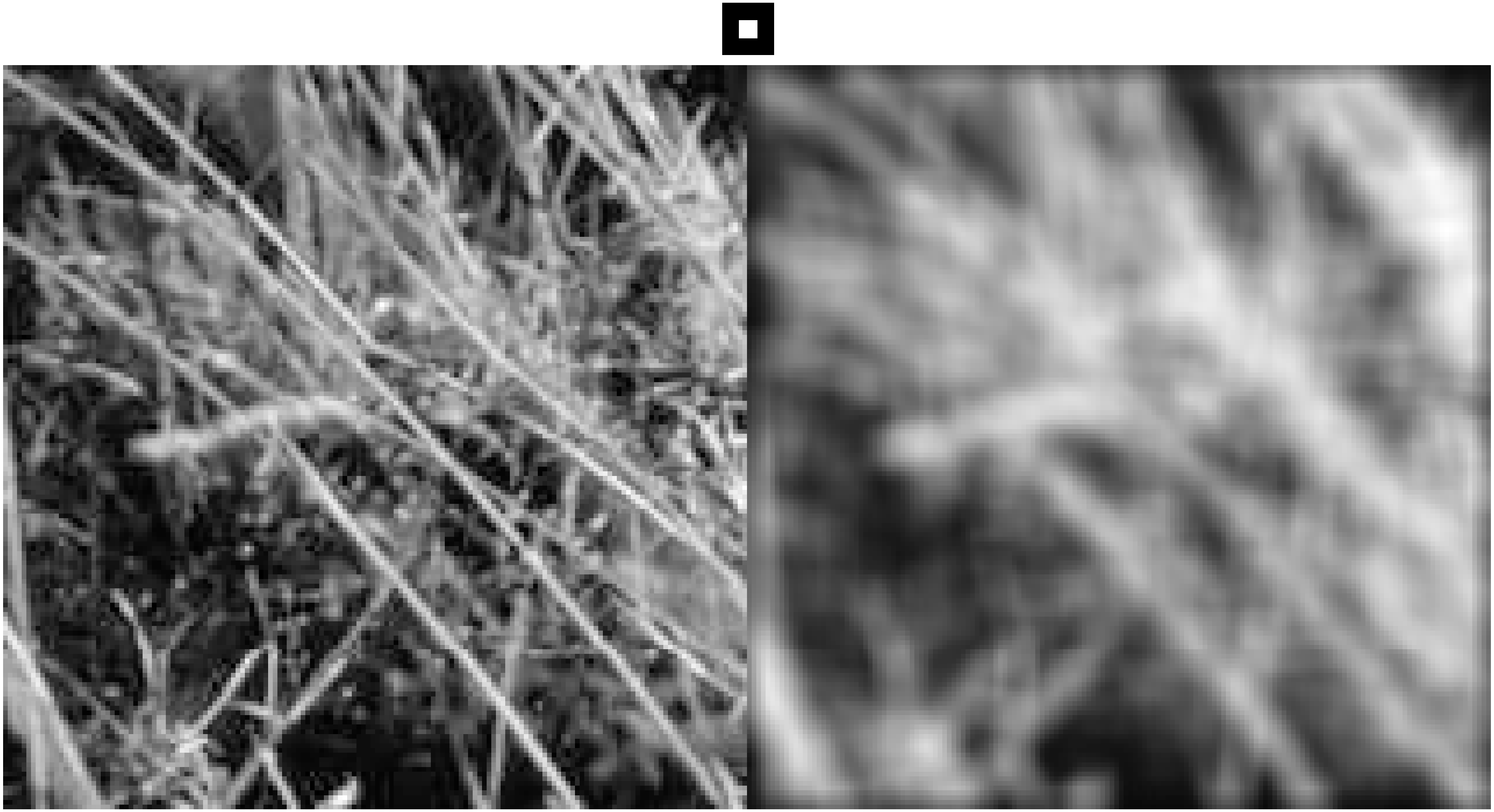
ringing

Smoothing by averaging

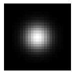
- this is the result for the Gaussian filter

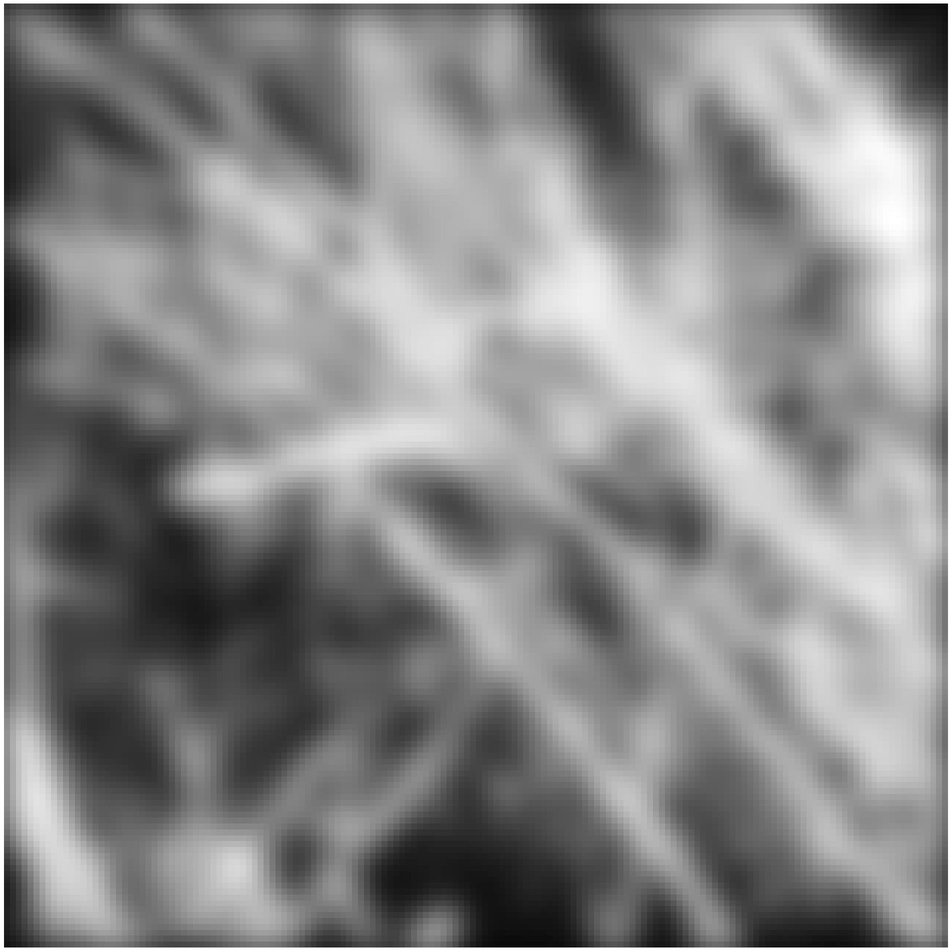
no ringing

Smoothing by Averaging

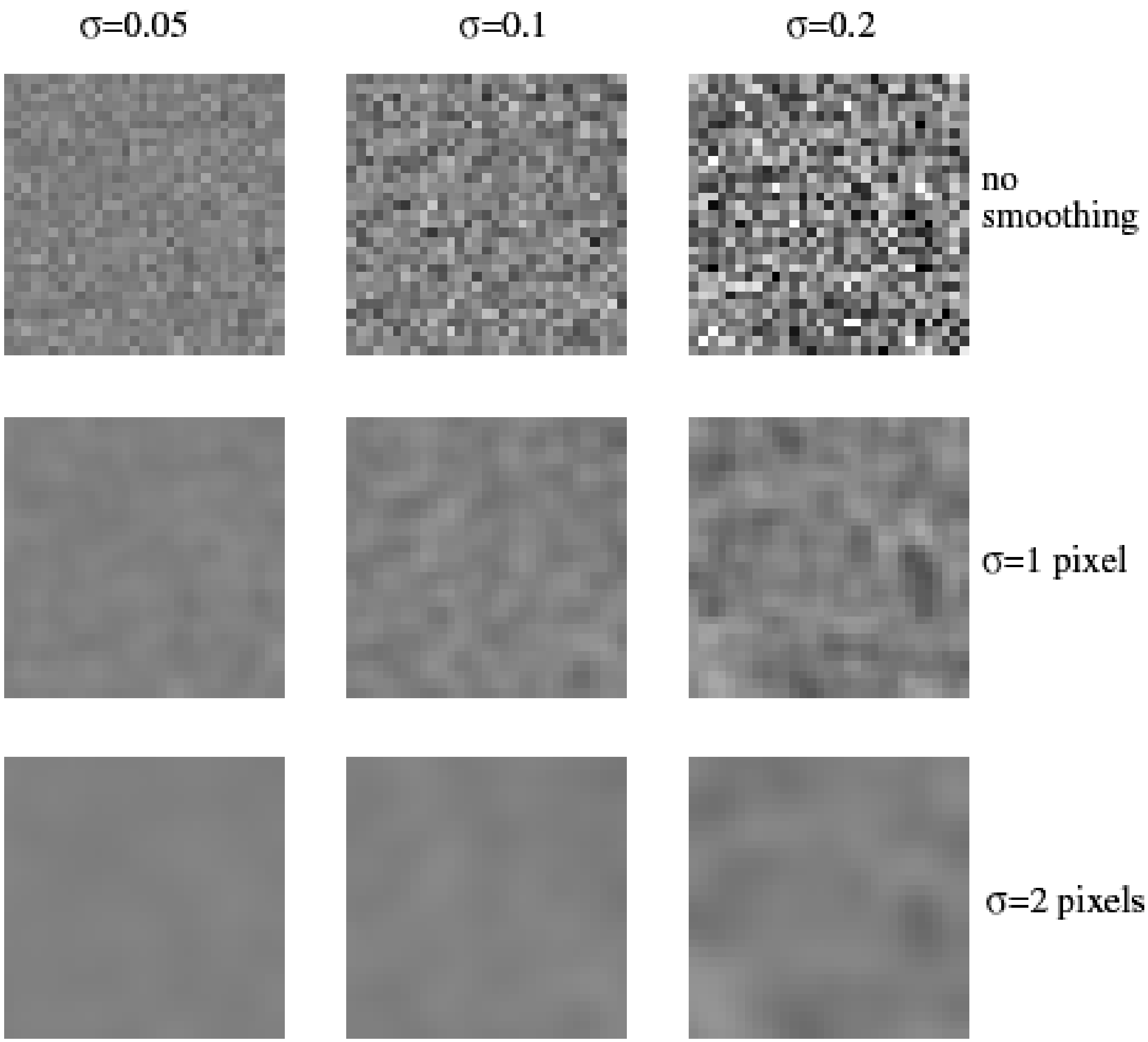


Smoothing with a Gaussian





Role of the variance

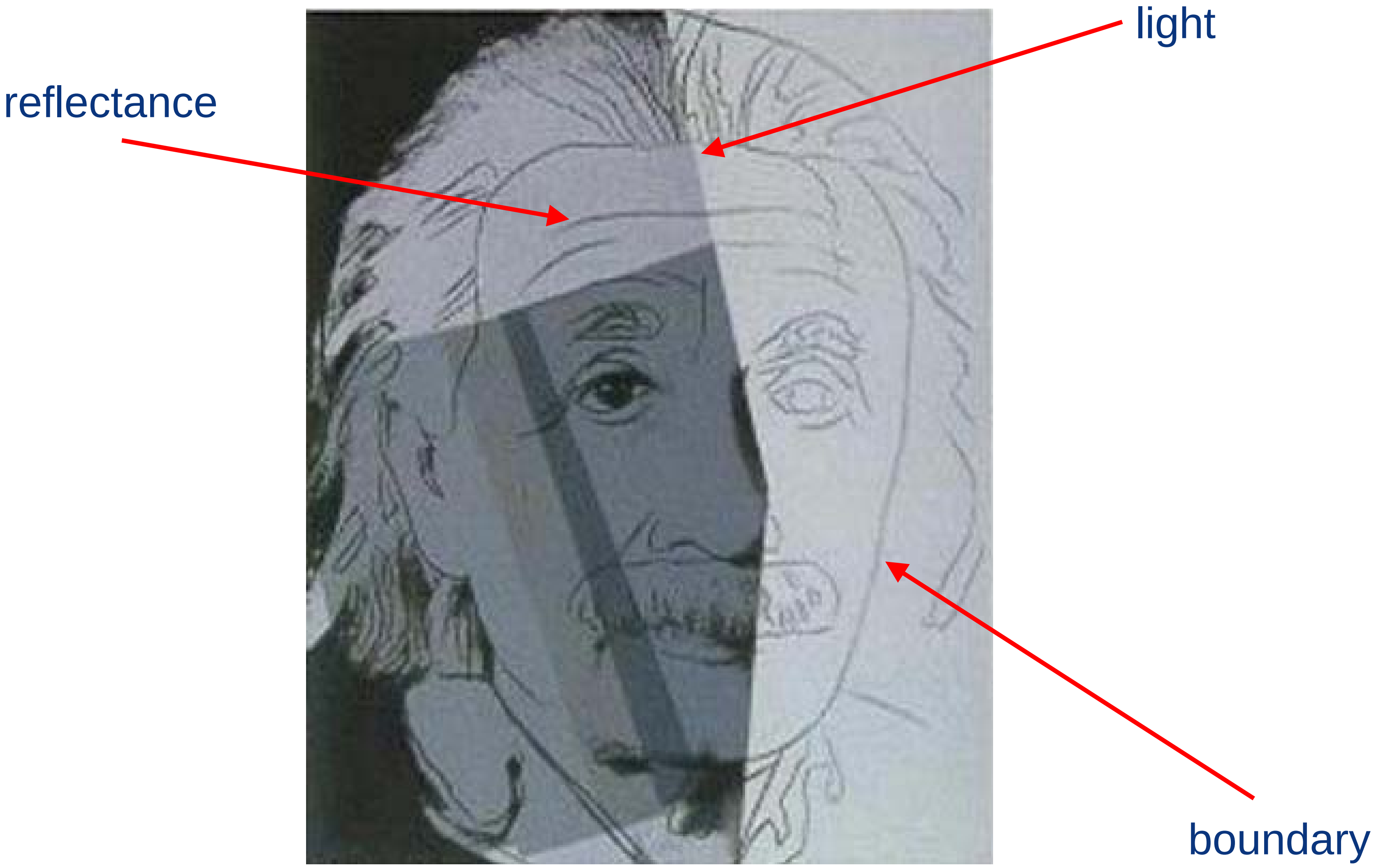


- the variance controls the amount of smoothing
- each column shows different realizations of an image of gaussian noise
- each row shows smoothing with gaussians of different σ

Gradients and edges

- for image understanding, one of the problems is that there is too much information in an image
- just smoothing is not good enough
- how to detect important (most informative) image points?
- note that derivatives are large at points of great change
  - changes in reflectance (e.g. checkerboard pattern)
  - change in object (an object boundary is different from background)
  - change in illumination (the boundary of a shadow)
- these are usually called **edge points**
- detecting them could be useful for various problems
  - segmentation: we want to know what are object boundaries
  - recognition: cartoons are easy to recognize and terribly efficient to transmit

The importance of edges



Gradients

- for a 2D function, *f(x,y)* the gradient at a point (x_0,y_0)

$$\nabla f(x_0,y_0) = \left(\frac{\partial f}{\partial x}(x_0,y_0), \frac{\partial f}{\partial y}(x_0,y_0) \right)^T$$

$$= \left(f_x(x_0,y_0), f_y(x_0,y_0) \right)^T$$

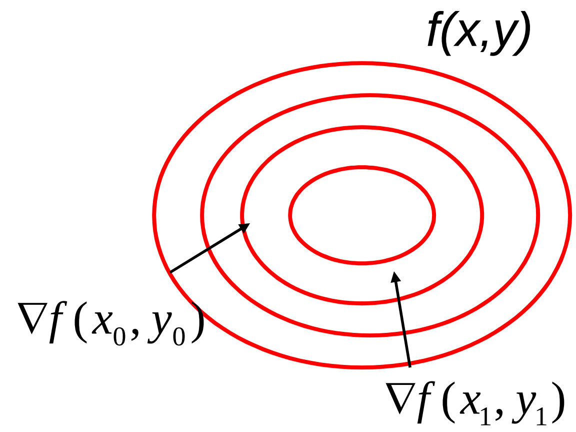
  is the direction of greatest increase at that point

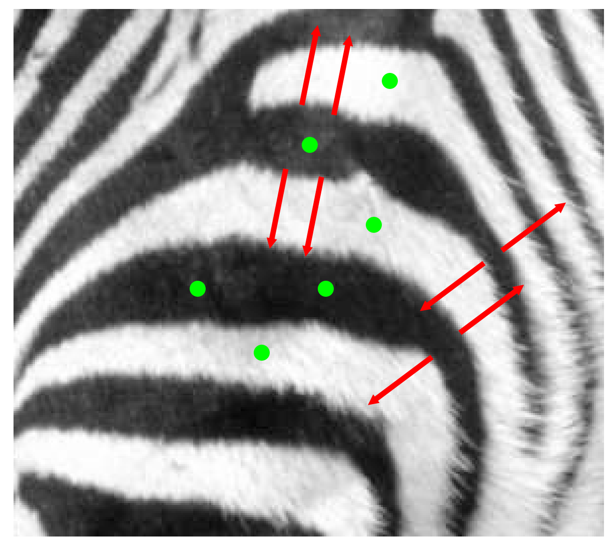
- the gradient magnitude

$$\left\| \nabla f(x_0,y_0) \right\|^2 = \left(\frac{\partial f}{\partial x}(x_0,y_0) \right)^2 + \left(\frac{\partial f}{\partial y}(x_0,y_0) \right)^2$$

  measures the rate of change

- it is large at edges!





– large gradient magnitude

– small gradient magnitude

Derivatives and convolution

- recall that a derivative is defined as

$$\frac{\partial f(x)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

- linear and shift invariant, so must be the result of a convolution.
- we could approximate as

$$\frac{\partial f(n)}{\partial n} = \frac{f(n+1) - f(n)}{1} = f(n+1) - f(n) = f * h(n)$$

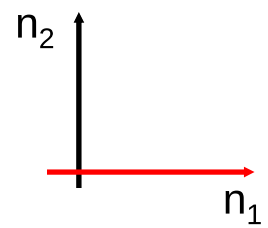
- where the derivative kernel is

$$h(n) = \delta(n+1) - \delta(n)$$

Finite difference kernels

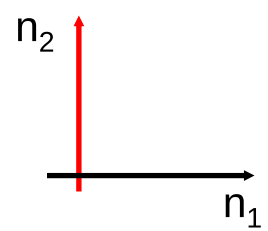
- in two dimensions we have various possible kernels
- e.g. , $N_1=2$, $N_2=3$, derivative along n_1, *(line $n_2=k$)* (horizontal)

$$\begin{array}{cc} 0 & 0 \\ 1 & -1 \\ 0 & 0 \end{array} \qquad \begin{array}{cc} 1 & -1 \\ 1 & -1 \\ 1 & -1 \end{array}$$



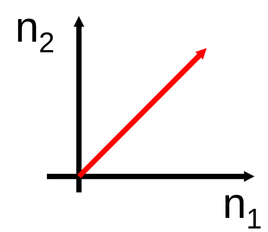
- derivative along $n_{2,}$ *(line $n_1=k$)* (vertical)

$$\begin{array}{ccc} 0 & -1 & 0 \\ 0 & 1 & 0 \end{array} \qquad \begin{array}{ccc} -1 & -1 & -1 \\ 1 & 1 & 1 \end{array}$$



- derivative along line $n_1=n_2$ (diagonal)

$$\begin{array}{ccc} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{array} \qquad \begin{array}{ccc} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{array}$$



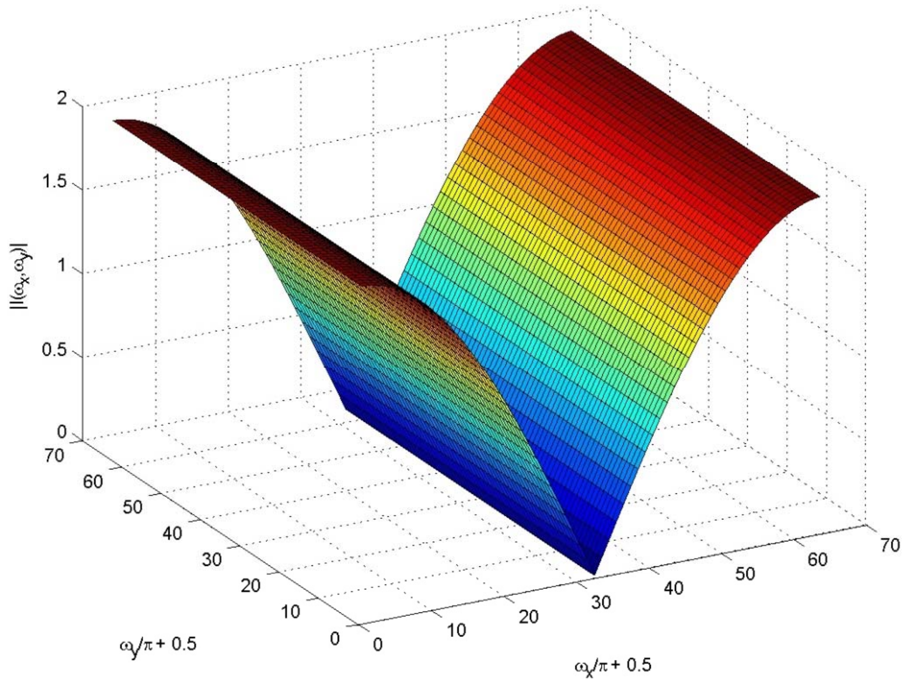
Finite difference kernels

- note that, when

$$h(n_1,n_2)= \begin{matrix} 0 & 0 \\ 1 & -1 \\ 0 & 0 \end{matrix}$$

- we have

$$H(\varpi_1,\varpi_2) = e^{j\varpi_1} - 1$$
$$= \left(e^{j\frac{\varpi_1}{2}} - e^{-j\frac{\varpi_1}{2}}\right)e^{j\frac{\varpi_1}{2}}$$
$$= 2\sin\left(\frac{\varpi_1}{2}\right)e^{j\frac{\varpi_1}{2}}$$



- derivative is a high-pass filter
- hw: check that this holds for all others
- intuitive, because a derivative is a measure of the rate of change of a function

Any questions?