

Linear Filters

- Example: smoothing by averaging
 - form the average of pixels in a neighbourhood
- Example: smoothing with a Gaussian
 - form a weighted average of pixels in a neighbourhood
- Example: finding a derivative
 - form a weighted average of pixels in a neighbourhood

Smoothing by Averaging

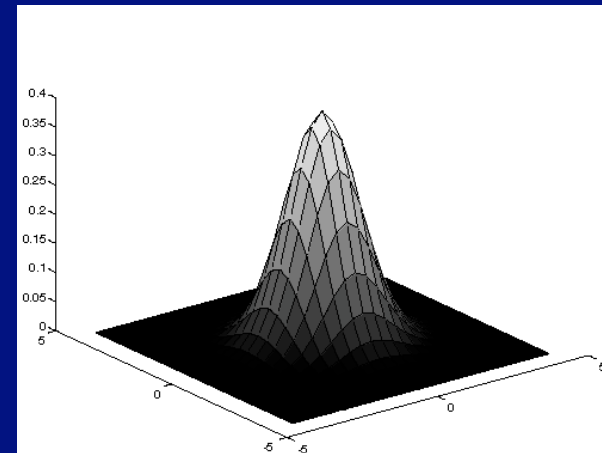


$$N_{ij} = \frac{1}{N} \sum_{uv} O_{i+u, j+v}$$

where u, v , is a window of N pixels in total centered at $0, 0$

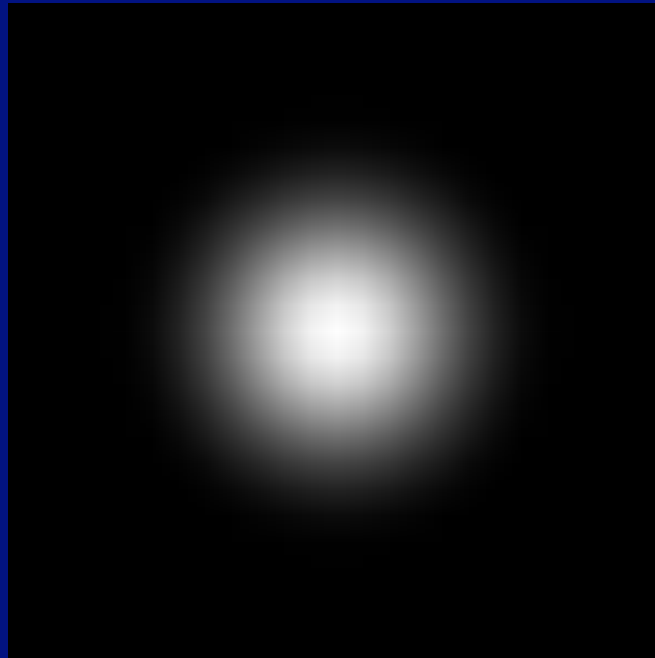
Smoothing with a Gaussian

- Notice “ringing”
 - apparently, a grid is superimposed
- Smoothing with an average actually doesn't compare at all well with a defocussed lens
 - what does a point of light produce?



- A Gaussian gives a good model of a fuzzy blob

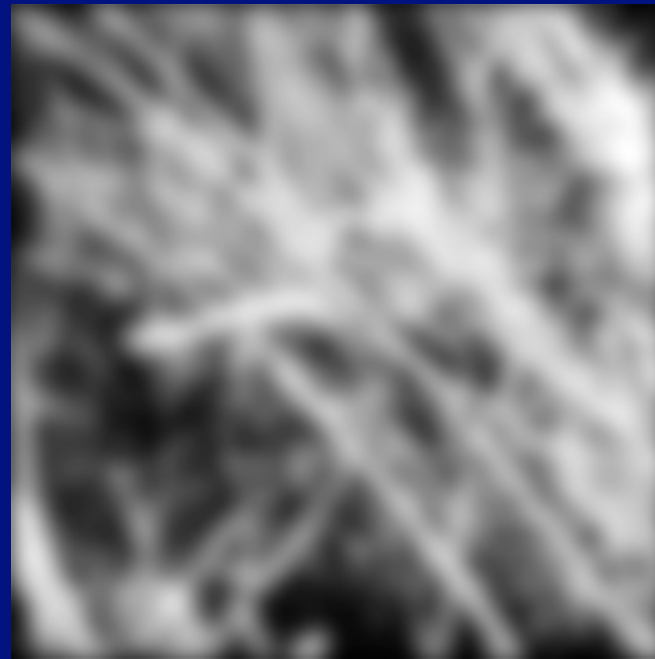
Gaussian filter kernel



$$K_{uv} = \left(\frac{1}{2\pi\sigma^2} \right) \exp \left(\frac{-[u^2 + v^2]}{2\sigma^2} \right)$$

We're assuming the index can take negative values

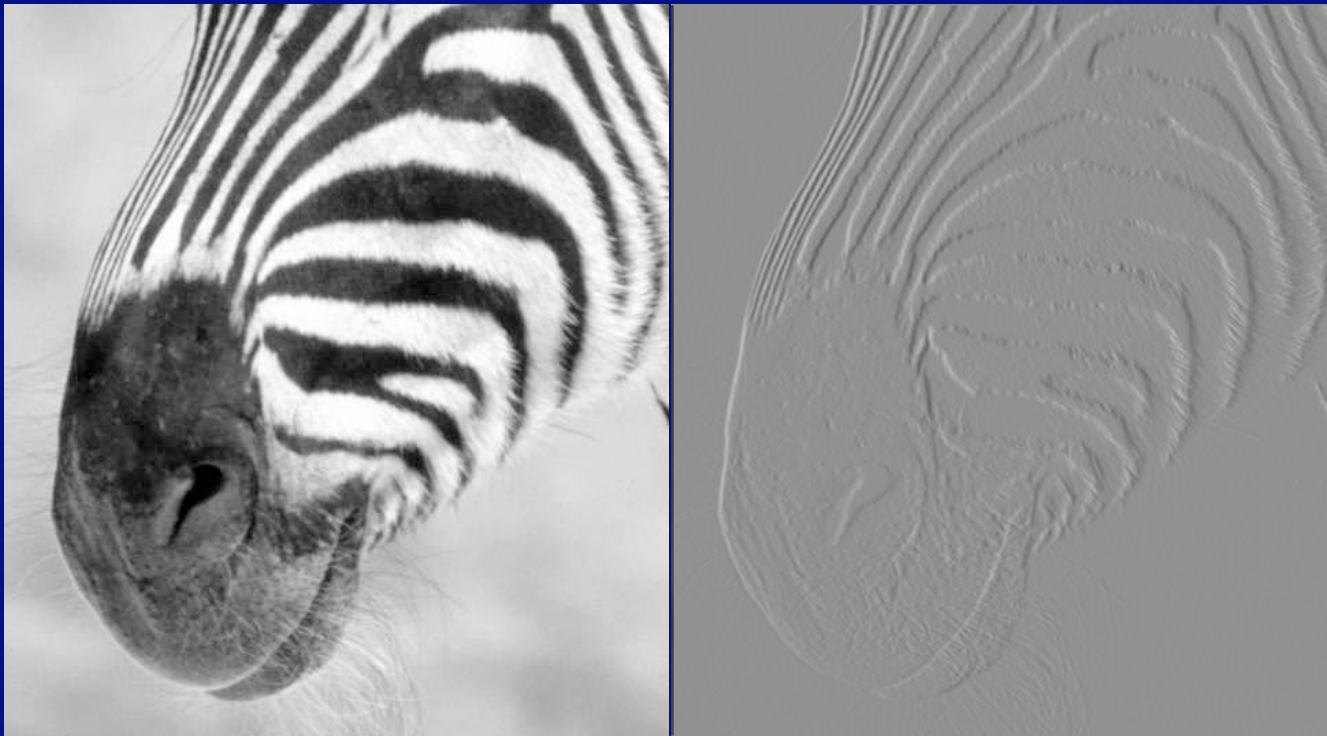
Smoothing with a Gaussian



$$N_{ij} = \sum_{uv} O_{i-u, j-v} K_{uv}$$

Notice the curious looking form

Finding derivatives



$$N_{ij} = \frac{1}{\Delta x} (I_{i+1,j} - I_{ij})$$

Convolution

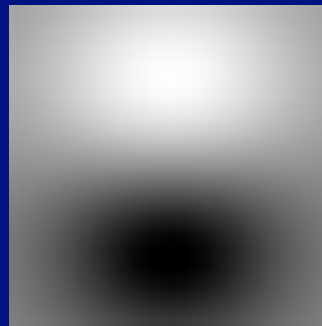
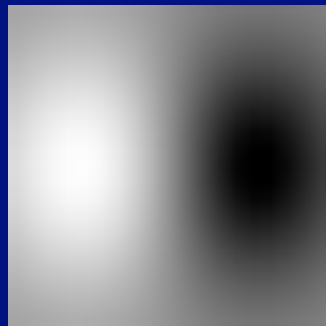
- Each of these involves a weighted sum of image pixels
- The set of weights is the same
 - we represent these weights as an image, H
 - H is usually called the kernel
- Operation is called convolution
 - it's associative
- Any linear shift-invariant operation can be represented by convolution
 - linear: $G(k f)=k G(f)$
 - shift invariant: $G(\text{Shift}(f))=\text{Shift}(G(f))$
 - Examples:
 - smoothing, differentiation, camera with a reasonable, defocussed lens system

$$N_{ij} = \sum_{uv} H_{uv} O_{i-u, j-v}$$

Filters are templates

$$N_{ij} = \sum_{uv} H_{uv} O_{i-u, j-v}$$

- At one point
 - output of convolution is a (strange) dot-product
- Filtering the image involves a dot product at each point
- Insight
 - filters look like the effects they are intended to find
 - filters find effects they look like



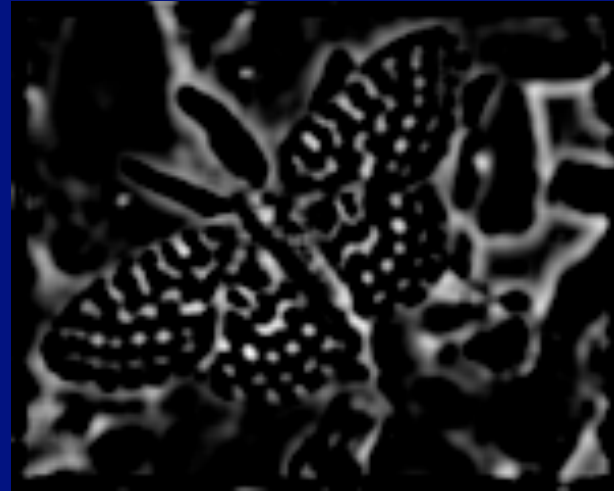
Normalised correlation

- Think of filters of a dot product
 - now measure the **angle**
 - i.e normalised correlation output is filter output, divided by root sum of squares of values over which filter lies
 - Tricks:
 - ensure that filter has a zero response to a constant region
 - helps reduce response to irrelevant background
 - subtract image average when computing the normalising constant
 - absolute value deals with contrast reversal



normalised correlation
with non-zero mean filter

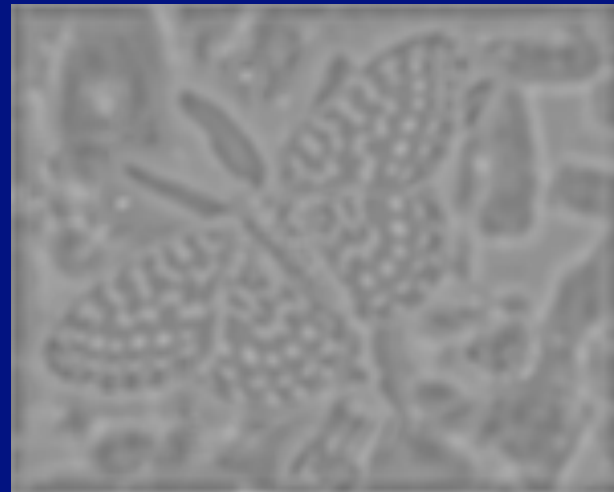


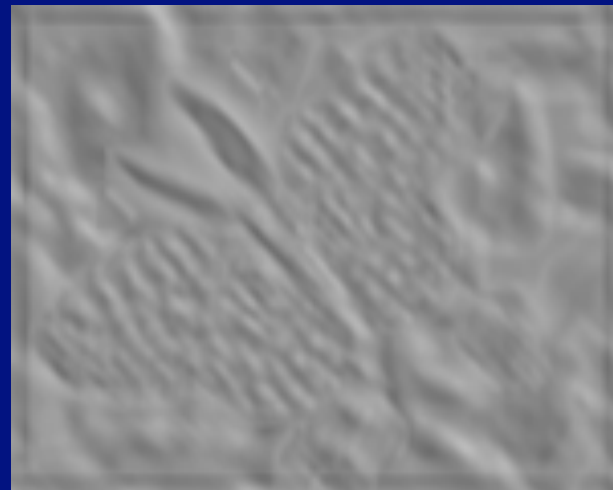
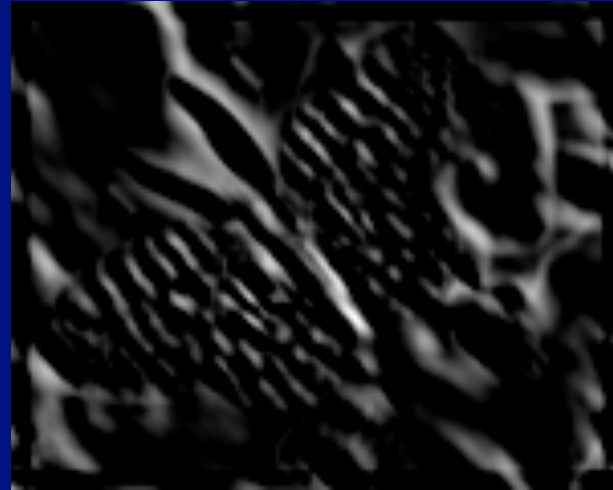


Positive responses

Zero mean image, -1:1 scale

Zero mean image, -max:max scale





Finding hands

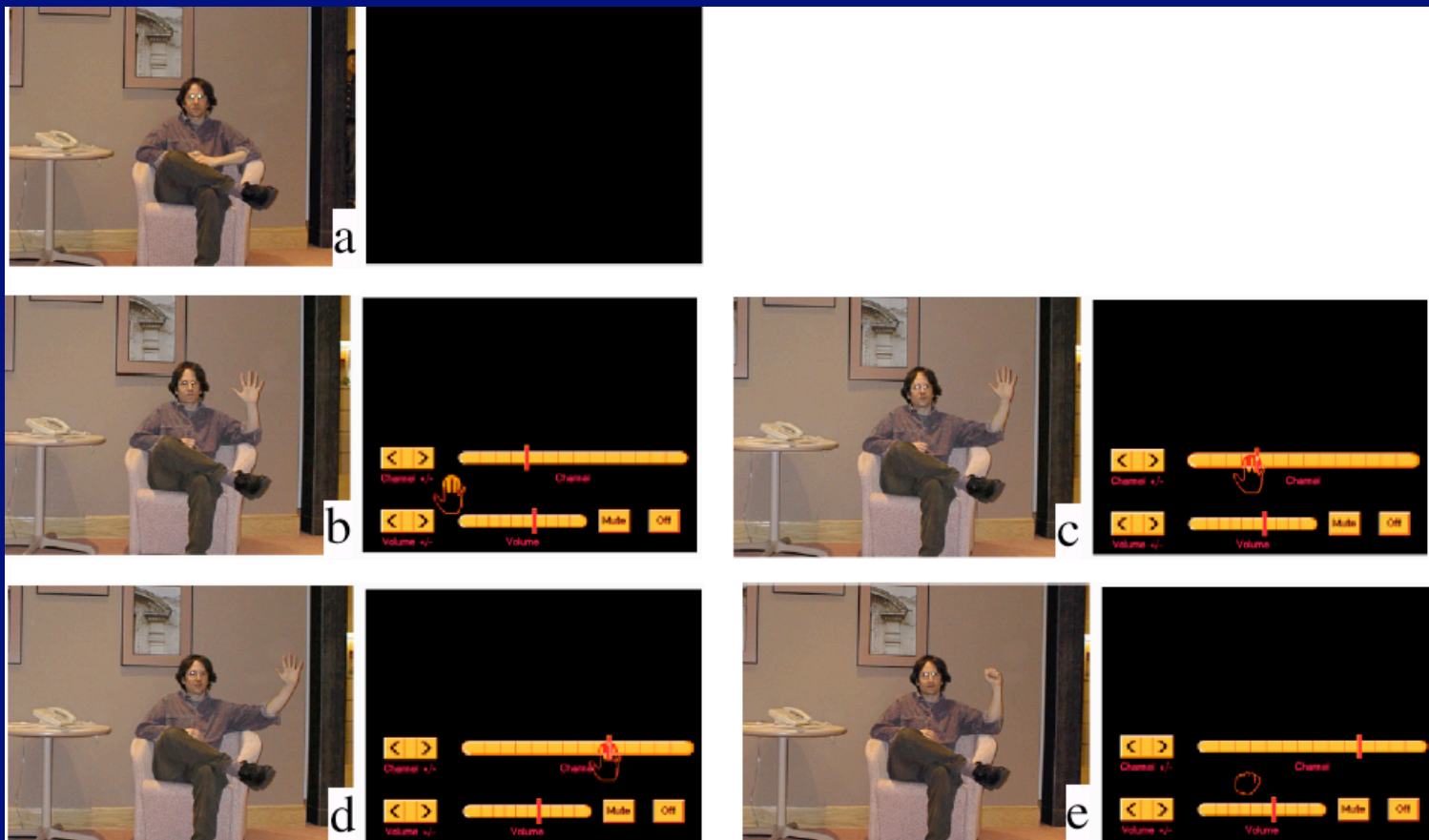


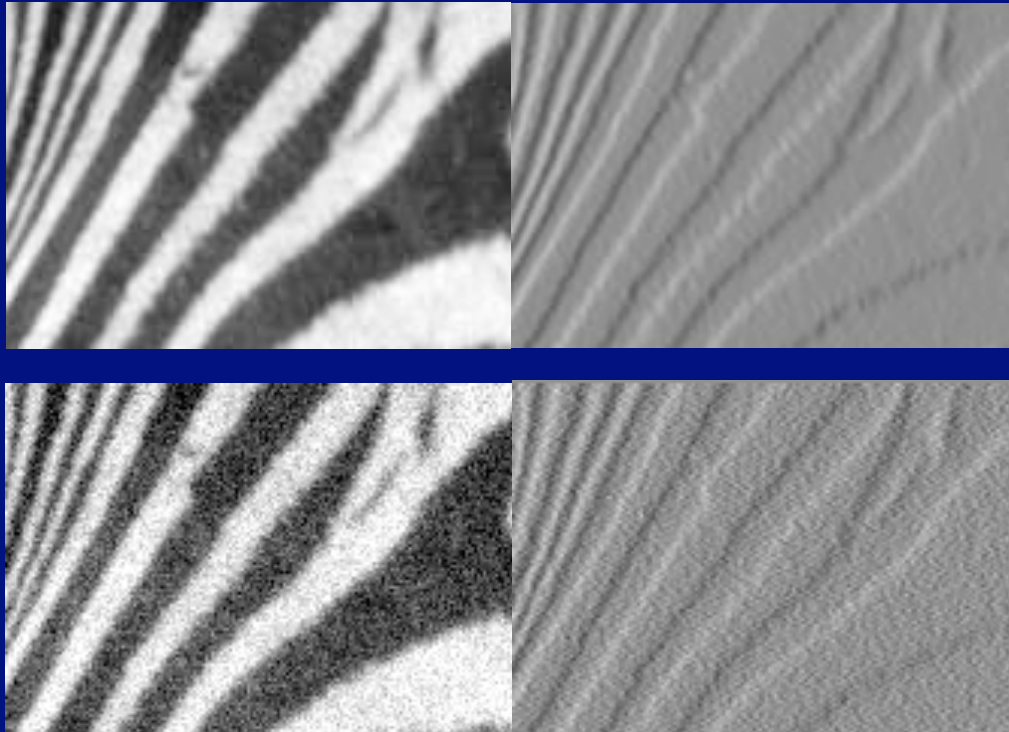
Figure from “Computer Vision for Interactive Computer Graphics,” W.Freeman et al, IEEE Computer Graphics and Applications, 1998

Gradients and edges

- Points of sharp change in an image are interesting:
 - change in reflectance
 - change in object
 - change in illumination
 - noise
- Sometimes called **edge points**
- General strategy
 - determine image gradient
 - now mark points where gradient magnitude is particularly large wrt neighbours

Differentiation and noise

- Simple derivative filters respond strongly to noise
 - obvious reason: noise is associated with strong changes, as above
- Generally, the larger the noise the stronger the response



Noise

- Simplest noise model
 - independent stationary additive Gaussian noise
 - the noise value at each pixel is given by an independent draw from the same normal probability distribution
- Issues
 - allows values greater than maximum camera output or less than zero
 - for small standard deviations, this isn't too much of a problem
 - independence may not be justified (e.g. damage to lens)
 - may not be stationary (e.g. thermal gradients in the ccd)

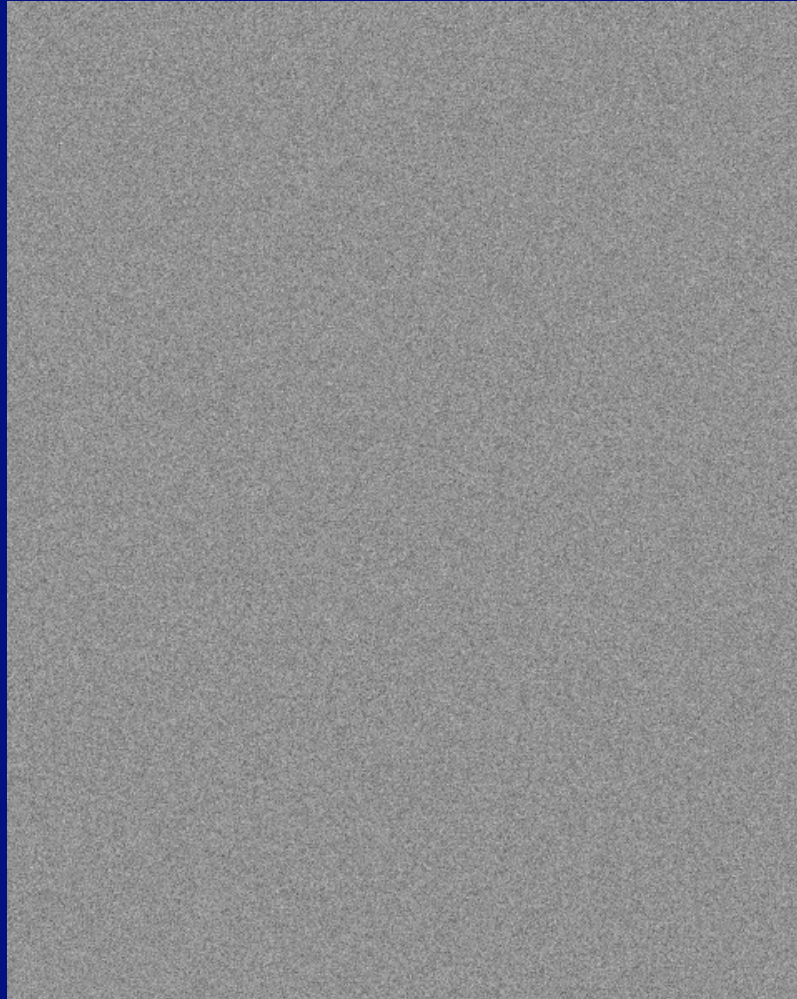
$\sigma=1$

A diagram consisting of a dark blue square background. In the center of this square is a smaller, solid gray square. To the left of the gray square, the text "sigma=1" is written in a white, serif font.

$\sigma=4$



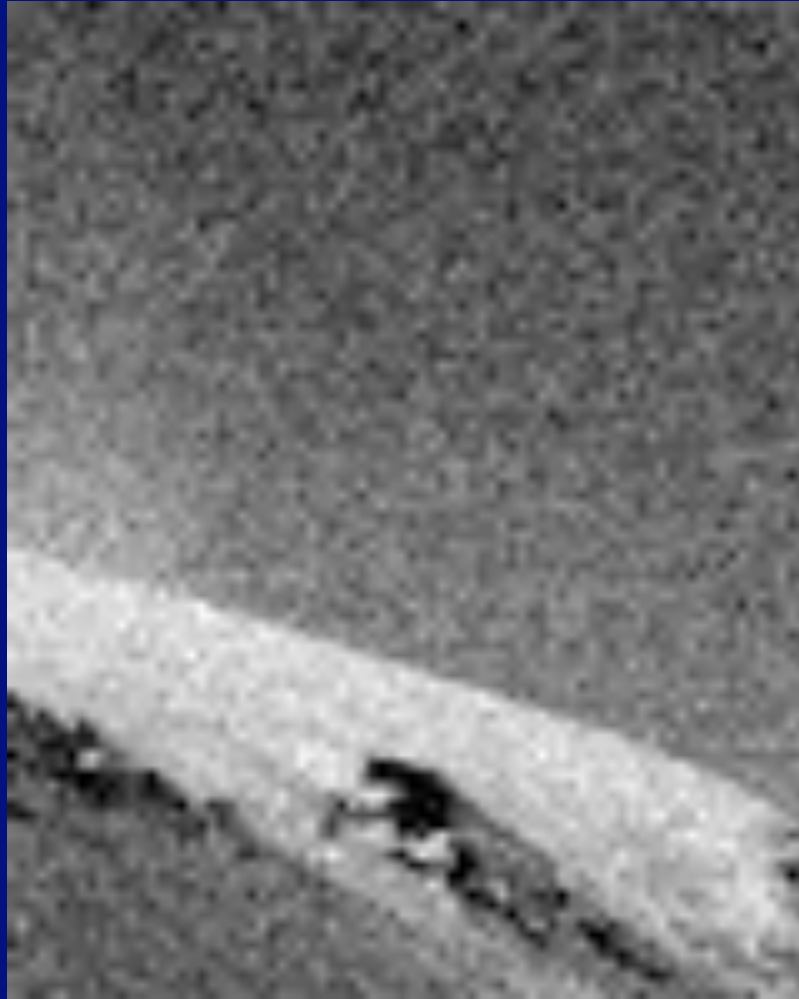
sigma=
16



$\sigma=1$



$\sigma=16$

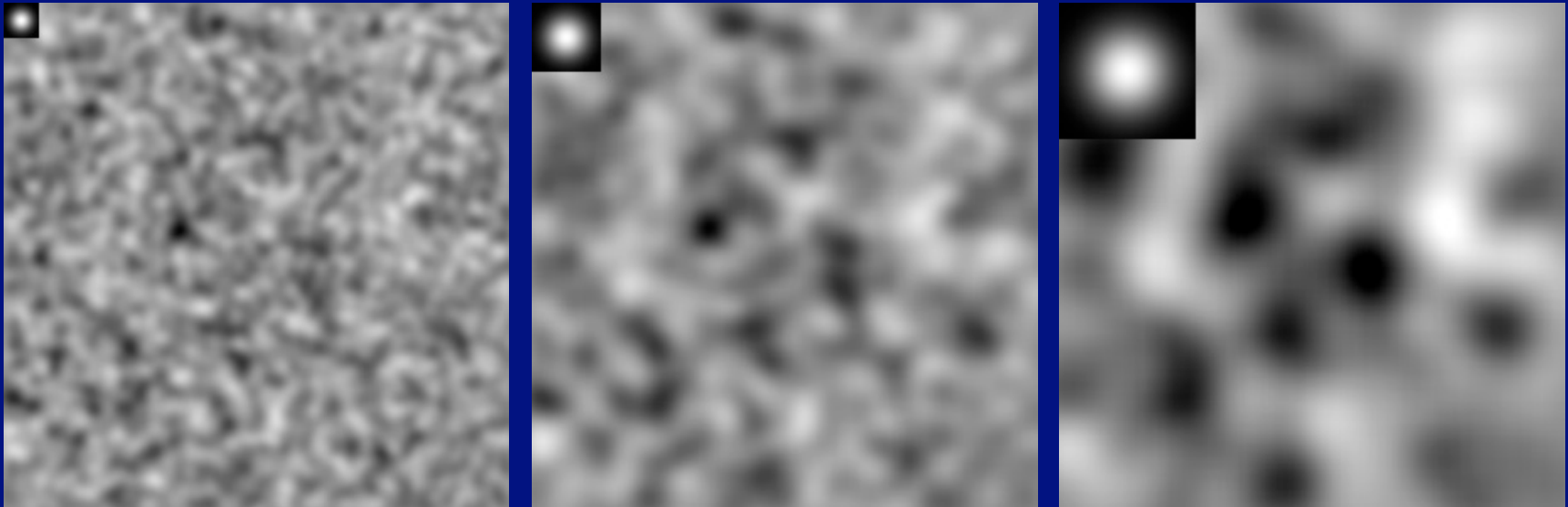


The response of a linear filter to noise

- Do only stationary independent additive Gaussian noise
 - get mean and variance of response by pattern matching
- Note that outputs are quite strongly correlated
 - useful trick for constructing simple textures

Filter responses are correlated

- (Fairly obviously) over scales similar to the scale of the filter



Smoothing reduces noise

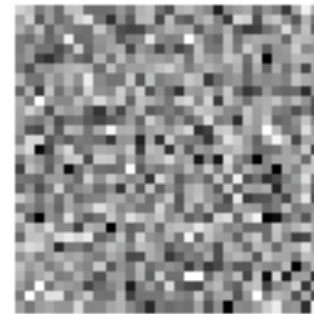
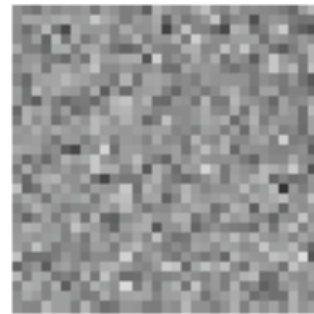
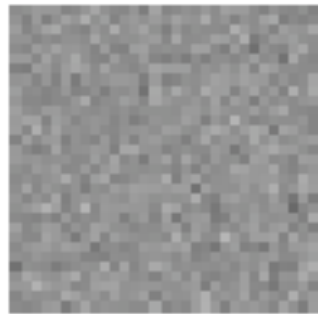
- Generally expect pixels to “be like” their neighbours
 - surfaces turn slowly
 - relatively few reflectance changes
- Expect noise to be independent from pixel to pixel
 - Implies that smoothing suppresses noise, for appropriate noise models
- Scale
 - the parameter in the symmetric Gaussian
 - as this parameter goes up, more pixels are involved in the average
 - and the image gets more blurred
 - and noise is more effectively suppressed

$$K_{uv} = \left(\frac{1}{2\pi\sigma^2} \right) \exp \left(\frac{-[u^2 + v^2]}{2\sigma^2} \right)$$

$\sigma=0.05$

$\sigma=0.1$

$\sigma=0.2$



no
smoothing



$\sigma=1$ pixel



$\sigma=2$ pixels

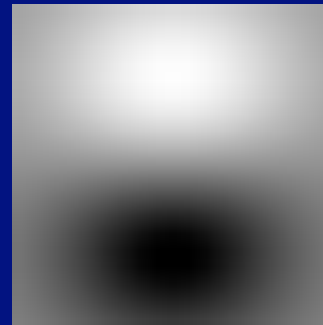
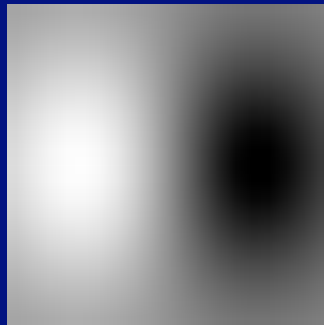


Edge detection

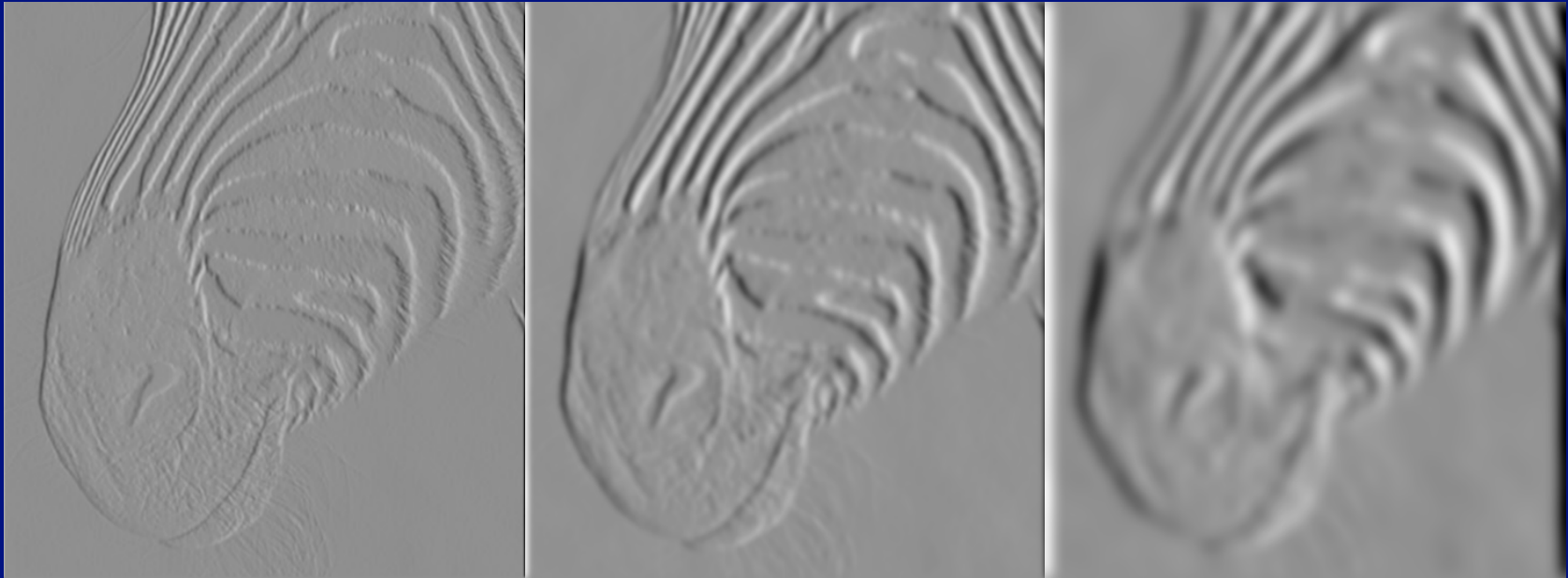
- Find points where image value changes sharply
- Strategy:
 - Estimate gradient magnitude using appropriate smoothing
 - Mark points where gradient magnitude is
 - Locally biggest and
 - big

Smoothing and Differentiation

- Issue: noise
 - smooth before differentiation
 - two convolutions to smooth, then differentiate?
 - actually, no - we can use a derivative of Gaussian filter



Scale affects derivatives



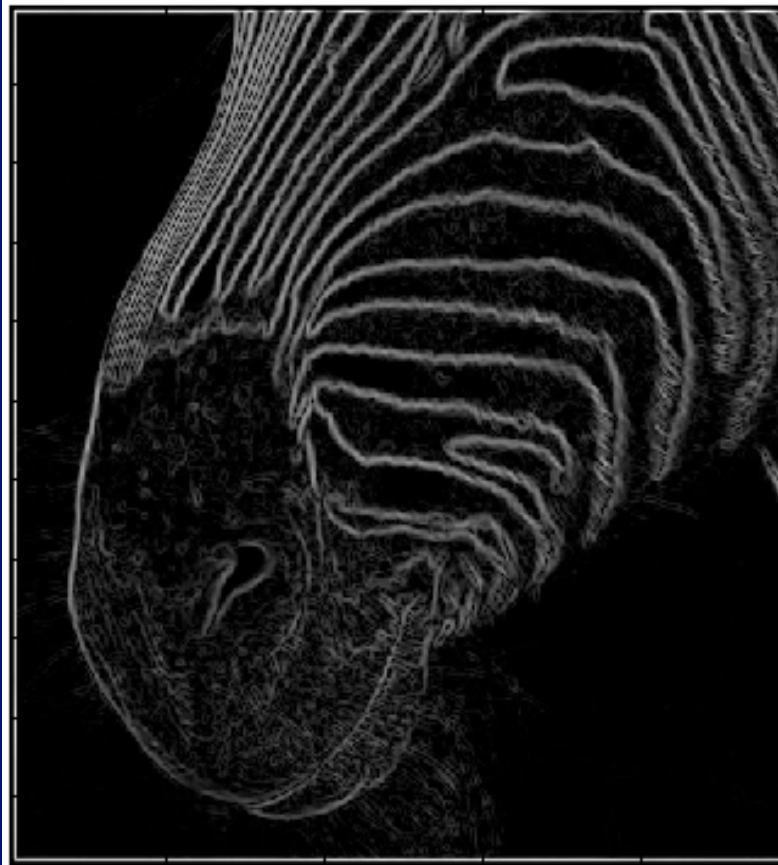
1 pixel

3 pixels

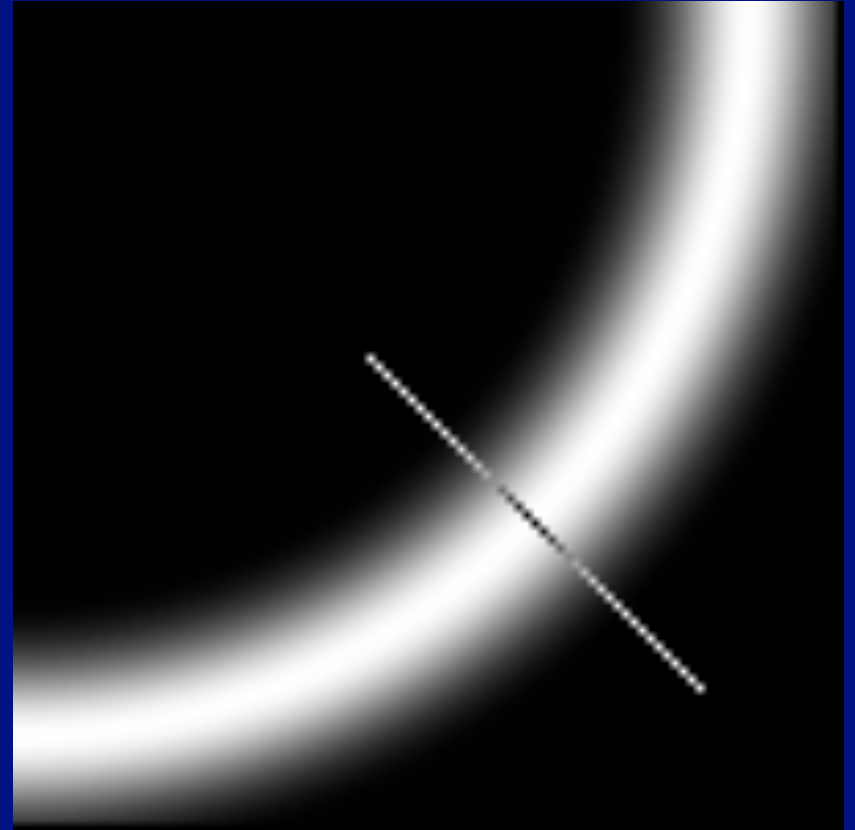
7 pixels



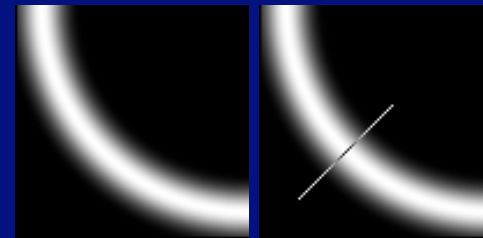
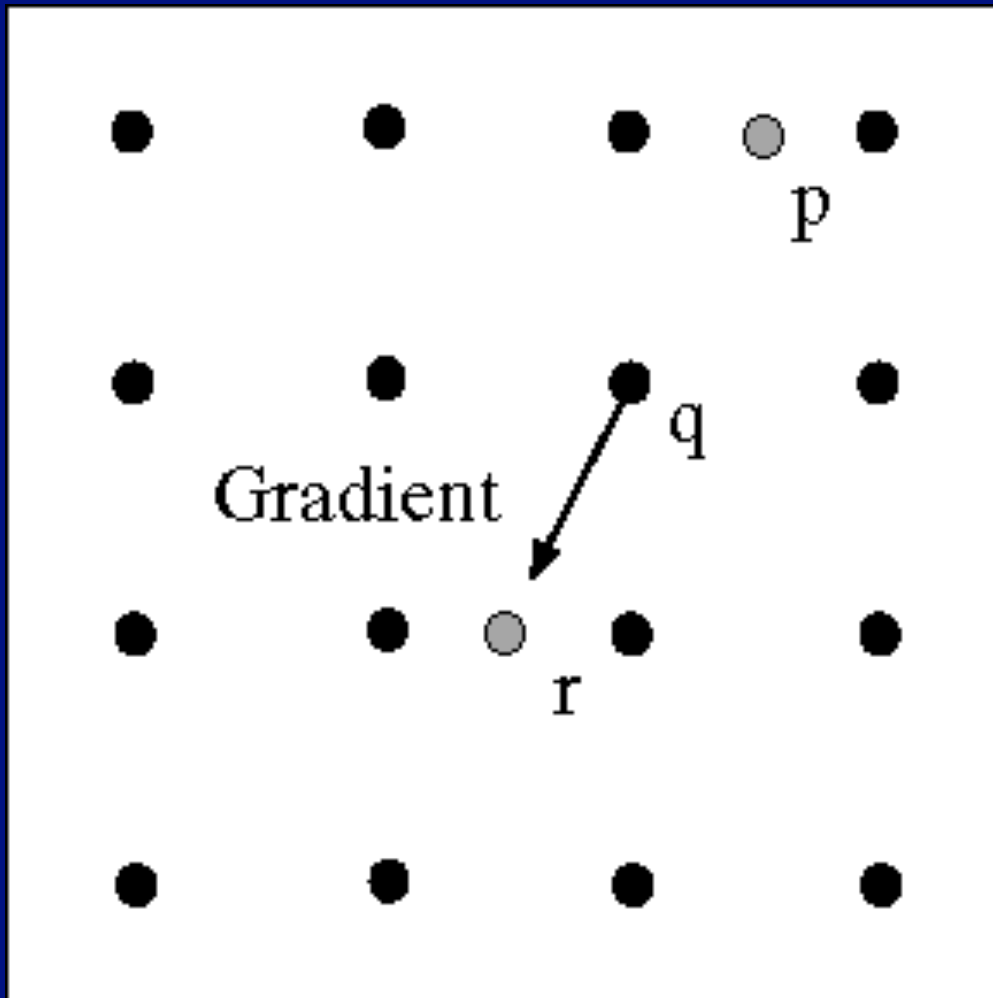
Scale affects gradient magnitude



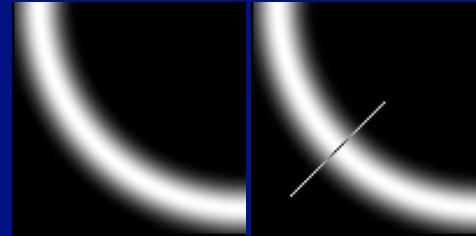
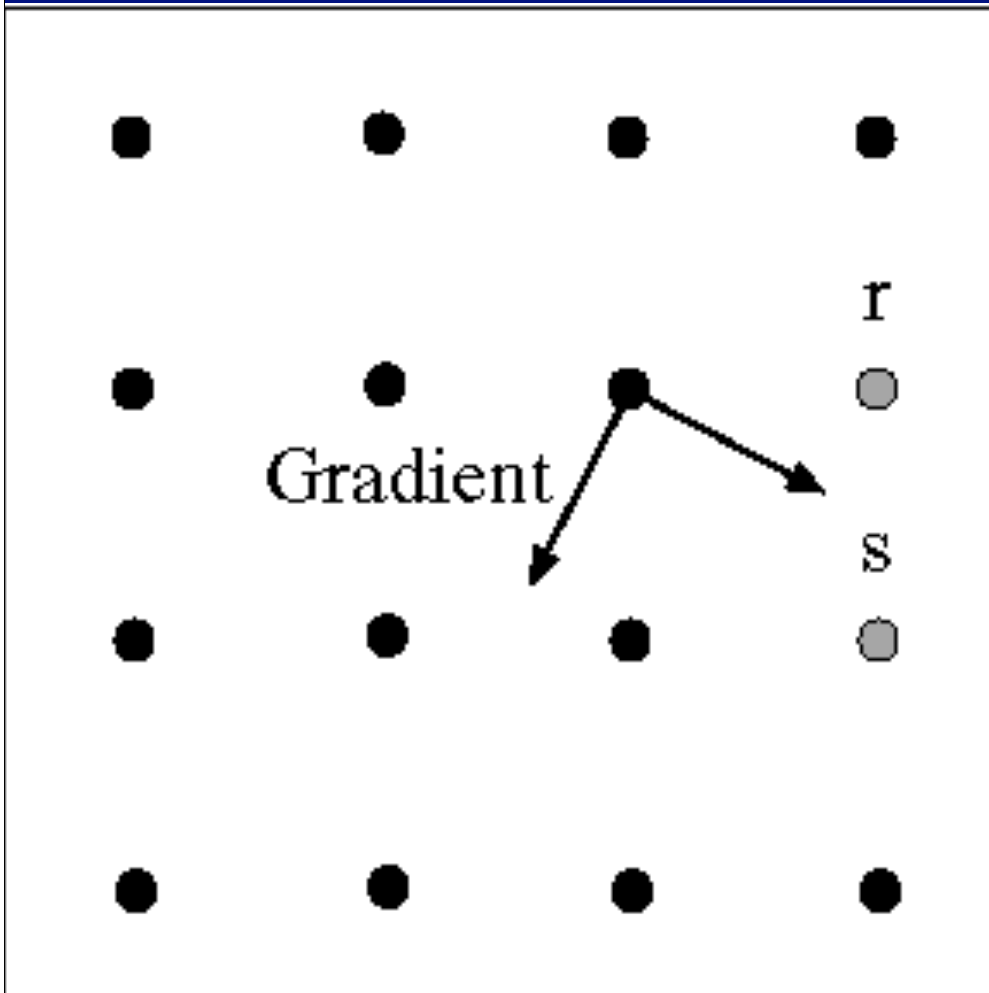
Marking the points



Non-maximum suppression



Predicting the next edge point



r

s

Remaining issues

- Check maximum value of gradient value is sufficiently large
 - drop-outs?
 - use hysteresis

Notice

- Something nasty is happening at corners
- Scale affects contrast
- Edges aren't bounding contours



The Laplacian of Gaussian

- Another way to detect an extremal first derivative is to look for a zero second derivative
- Appropriate 2D analogy is rotation invariant
- Zero crossings of Laplacian
 - Bad idea to apply a Laplacian without smoothing
 - smooth with Gaussian, apply Laplacian
 - this is the same as filtering with a Laplacian of Gaussian filter
 - Now mark the zero points where
 - there is a sufficiently large derivative,
 - and enough contrast

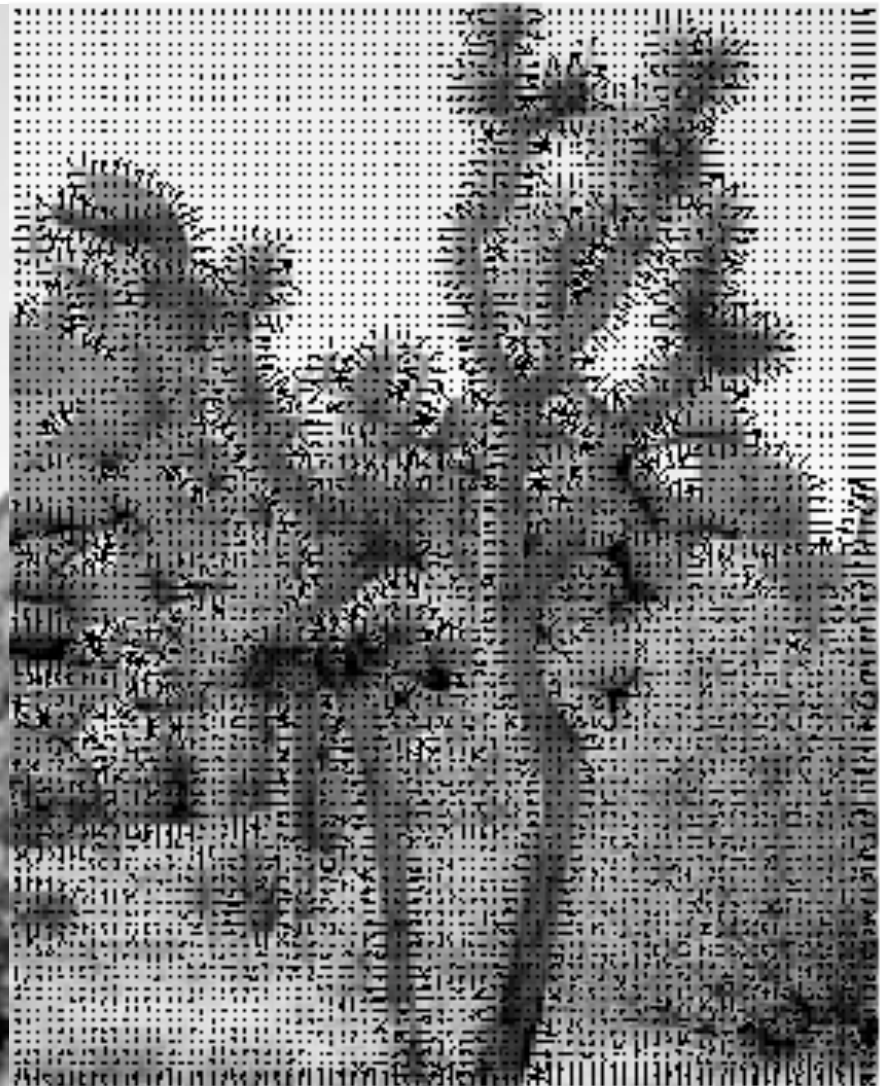
Orientation representations

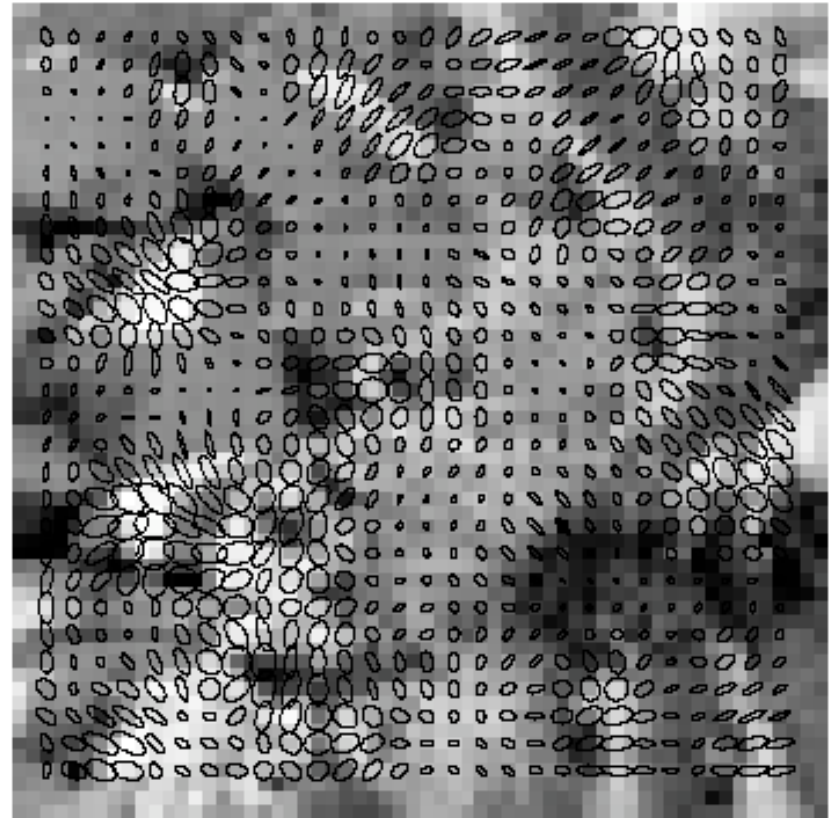
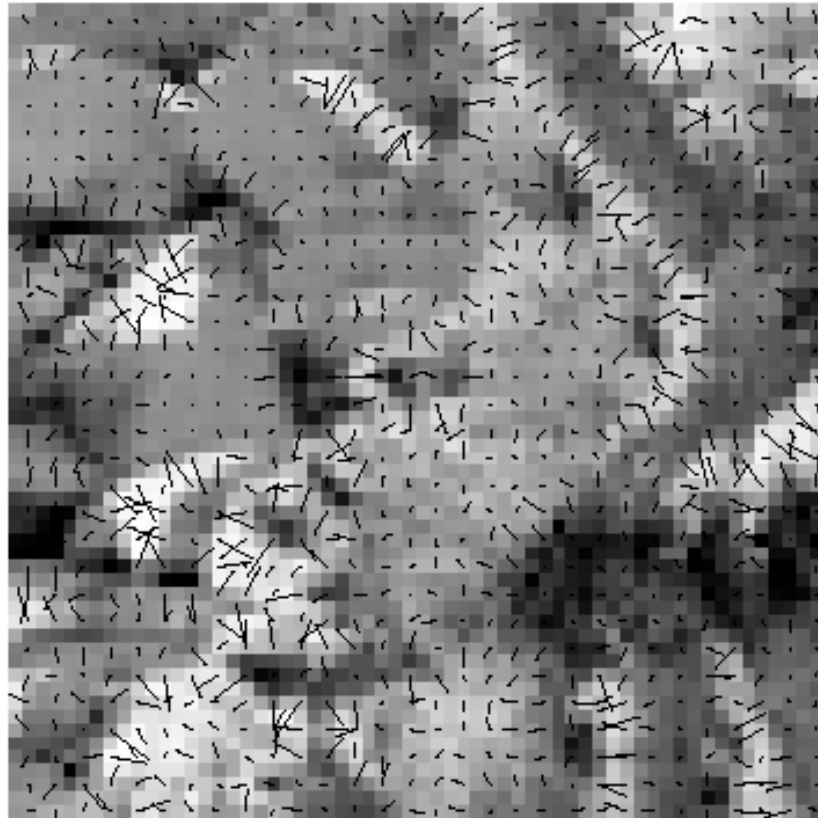
- Gradient magnitude is affected by illumination changes
 - but it's direction isn't
- Describe image patches by gradient direction
- Important types:
 - constant window
 - small gradient mags
 - edge window
 - few large gradient mags in one direction
 - flow window
 - many large gradient mags in one direction
 - corner window
 - large gradient mags that swing

Representing Windows

- Types
 - constant
 - small eigenvalues
 - Edge
 - one medium, one small
 - Flow
 - one large, one small
 - corner
 - two large eigenvalues

$$H = \sum_{\text{window}} (\nabla I)(\nabla I)^T$$



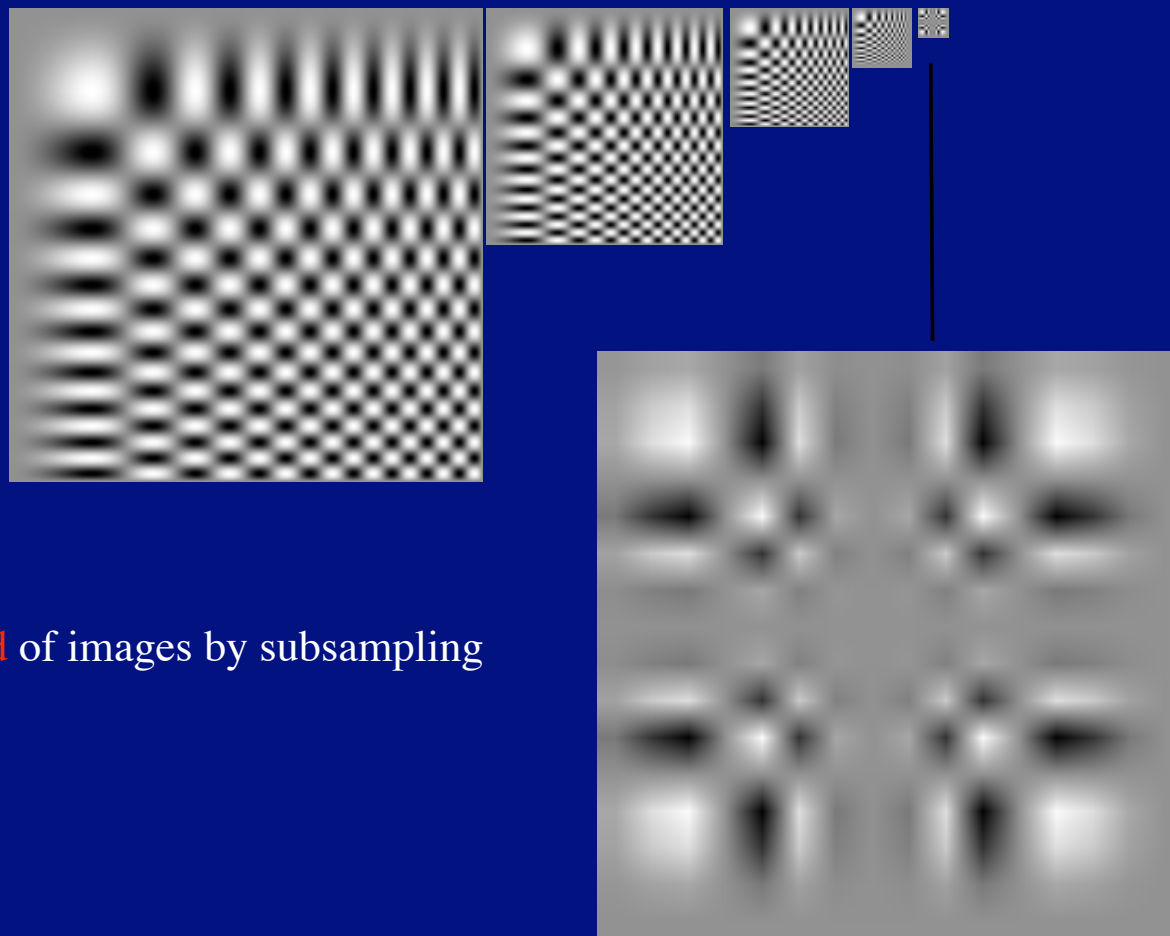


Plots of $\mathbf{x}H^{-1}\mathbf{x} = 0$

Scaled representations

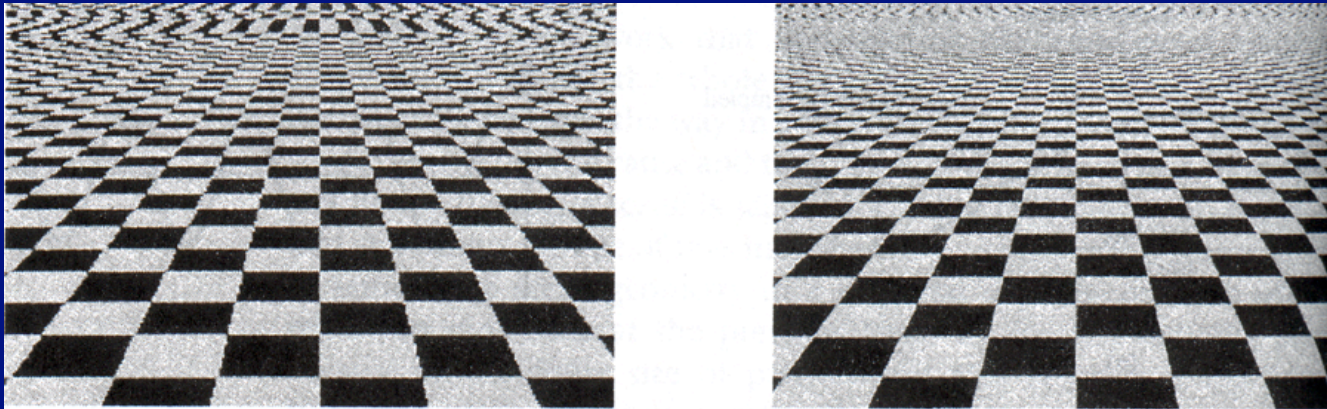
- Represent one image with many different resolutions
- Why?
 - Search for correspondence
 - look at coarse scales, then refine with finer scales
 - Edge tracking
 - a “good” edge at a fine scale has parents at a coarser scale
 - Control of detail and computational cost in matching
 - e.g. finding stripes
 - terribly important in texture representation

Carelessness causes aliasing



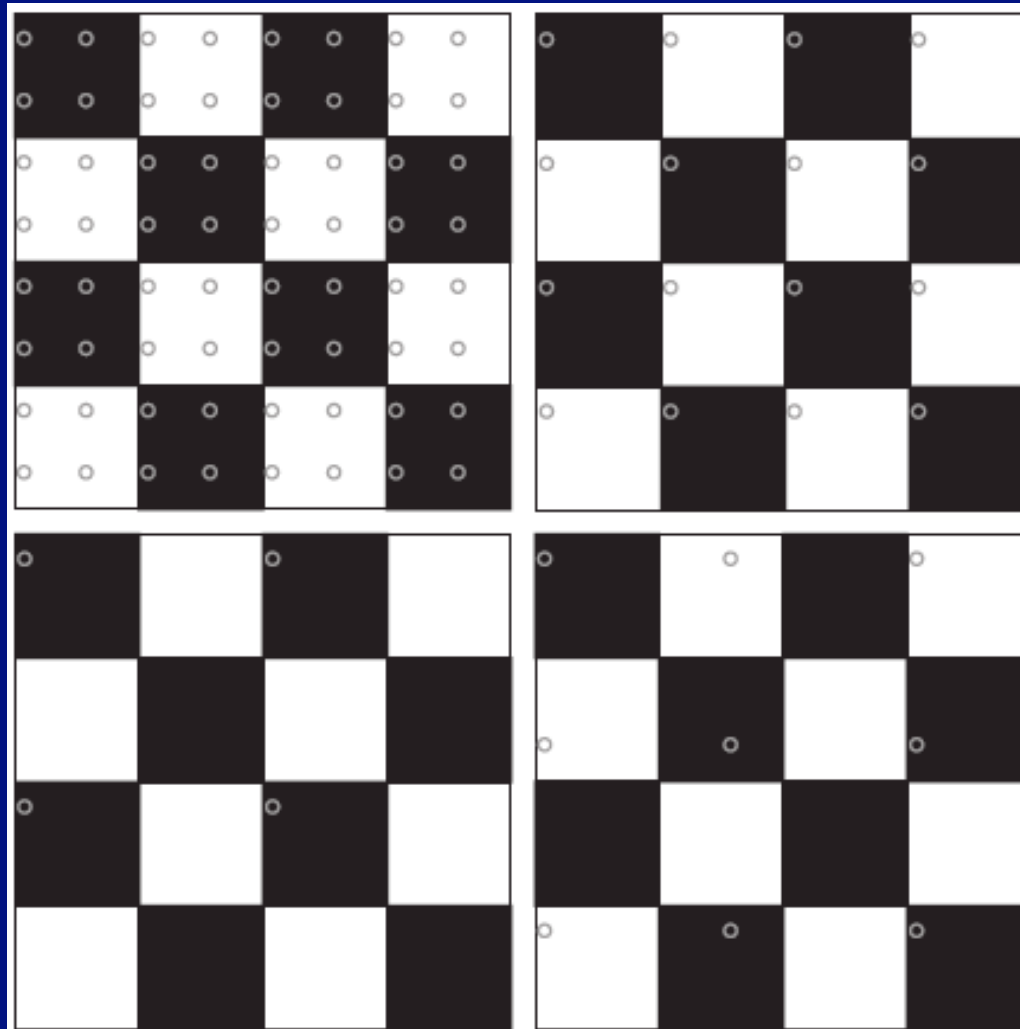
Obtained **pyramid** of images by subsampling

Aliasing



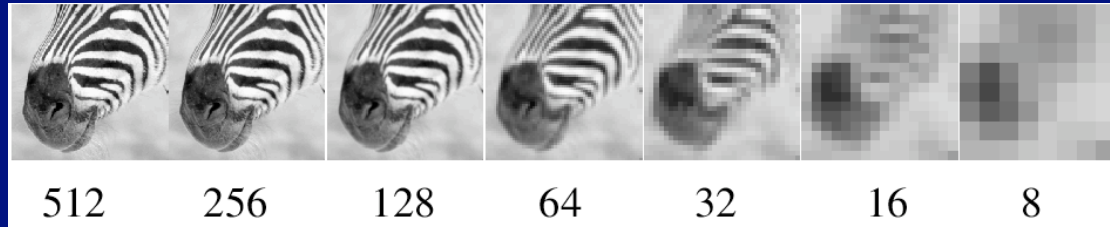
from Watt and Policarpo, *The Computer Image*

Aliasing - smoothing helps



The Gaussian pyramid

- Smooth with gaussians, because
 - a gaussian*gaussian=another gaussian
- Synthesis
 - (making a pyramid from an image)
 - smooth and sample
- Analysis
 - (making an image from a pyramid)
 - take the top image
- Gaussians are low pass filters, so repn is redundant



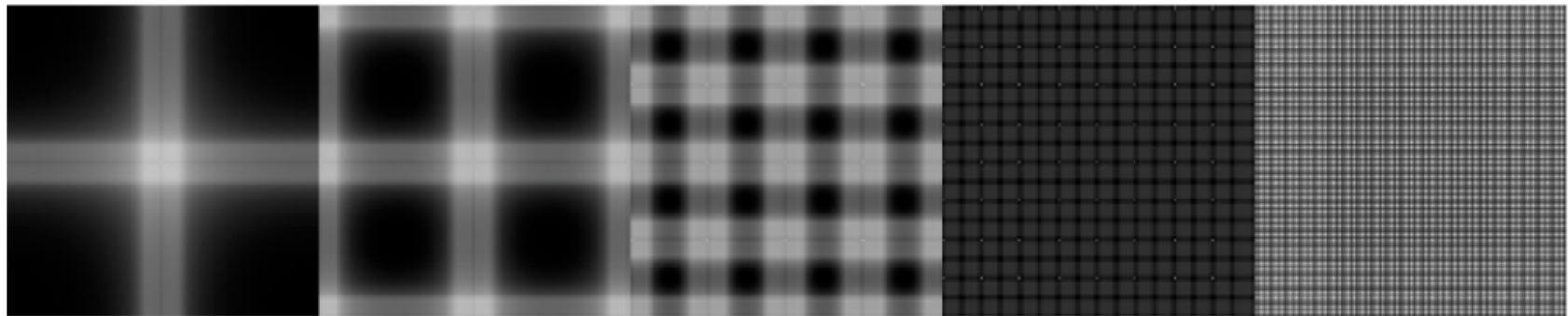
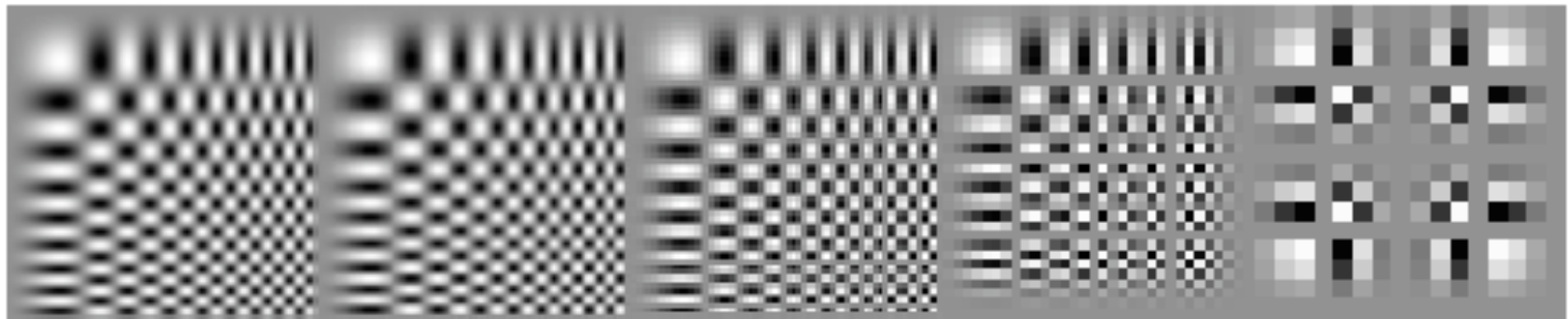
256x256

128x128

64x64

32x32

16x16



256x256

128x128

64x64

32x32

16x16

