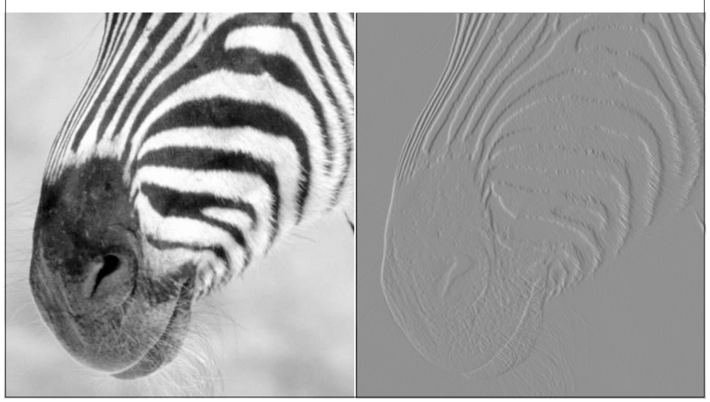
Finding Edges

- Edges reveal much about images
- Edge representations can be seen as information compression (because boundary is fewer pixels than the inside)
- Edges are the result of many different things
  - simple material change (step edge, corners)
  - illumination change (often soft, but not always)
  - shading edges and bar edges in inside corners
- An edge is basically where the images changes---hence finding images is studying changes (differentiation)

Differentiation and convolution

- Recall $\frac{\partial f}{\partial x} = \lim_{\varepsilon \to 0}\left(\frac{f(x+\varepsilon, y)}{\varepsilon} - \frac{f(x, y)}{\varepsilon}\right)$
- Now this is linear and shift invariant, so must be the result of a convolution.
- We could approximate this as $\frac{\partial f}{\partial x} \approx \frac{f(x_{n+1}, y) - f(x_n, y)}{\Delta x}$

Finite differences (x-direction)



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

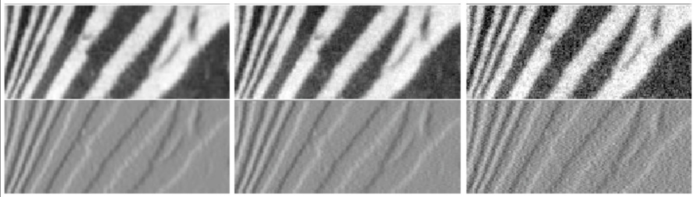
image with added
Gaussian noise
(sigma=1)

image with added
Gaussian noise
(sigma=16)

Finite differences and noise

- Finite difference filters respond strongly to noise
  - Noise is not correlated across adjacent pixels, but the pixels tend to be correlated
  - Thus differences lock onto the noise!
- The larger the noise, the bigger such a response

Finite differences responding to noise



Increasing noise ----->

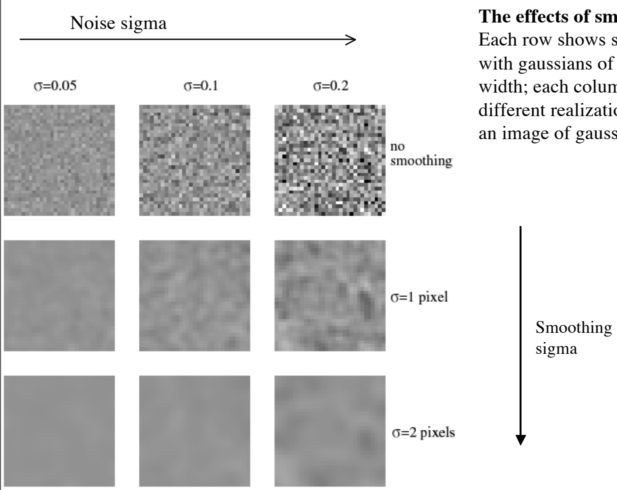
(zero mean additive gaussian noise)

Smoothing reduces noise

- Generally expect pixels to “be like” their neighbours
  - surfaces turn slowly
  - relatively few reflectance changes
- Generally expect noise processes to be **independent** from pixel to pixel
- Implies that some kind of averaging or smoothing suppresses noise, for appropriate noise models

Smoothing reduces noise

- Degree of smoothing <==> 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



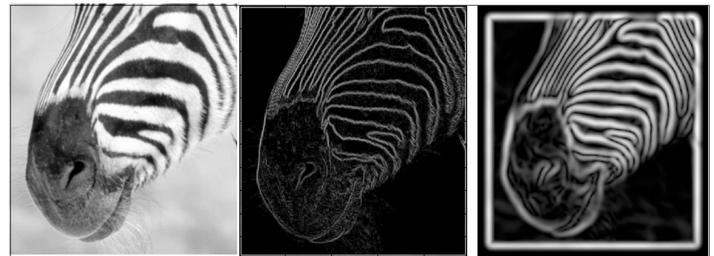
The effects of smoothing Each row shows smoothing with gaussians of different width; each column shows different realizations of an image of gaussian noise.

Median Filtering

- Using a Gaussian to remove noise assumes a well behaved noise process (sensitive to outliers).
- A more robust method is to replace a pixel with the median of the ones in a window (median filtering)
- This filter is non-linear!
  - We give up lots of nice properties

Gradients and edges

- Sources of points of sharp change in an image:
  - change in reflectance
  - change in object
  - change in illumination
  - noise!
- Sometimes called **edge points**
- General strategy
  - determine image gradient
  - mark points where gradient magnitude is particularly large compared to that of neighbours
  - attempt to promote linked edge points



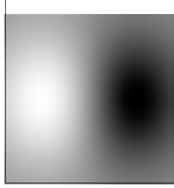
Gradient magnitudes of zebra at two different scales

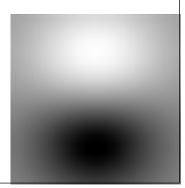
Three major issues:

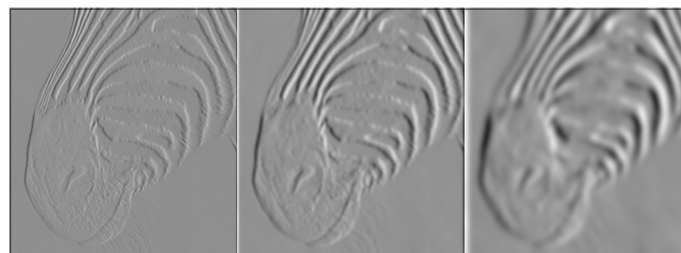
1) The gradient magnitude at different scales is different; which one should we choose?
2) The gradient magnitude is large throughout thick trail; how do we identify the actual edge location?
3) How do we link the relevant points up into curves?

Smoothing and Differentiation

- Issue: noise
  - so we smooth before differentiation
  - this suggests a convolution to smooth, then a convolution to differentiate
  - but we can use a derivative of Gaussian filter
    - because differentiation is convolution, and convolution is associative







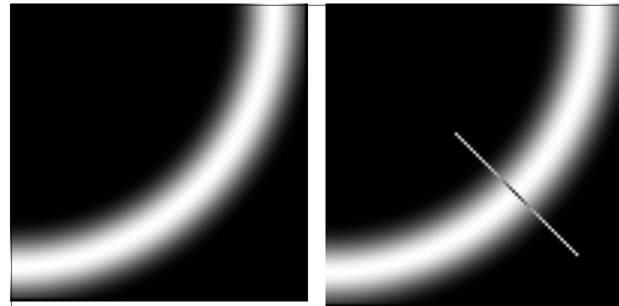
1 pixel   3 pixels   7 pixels

Horizontal derivative magnitude at different scales

The scale affects the estimates and the semantics of the edges recovered.

Non-maximal suppression (alg 8.2)

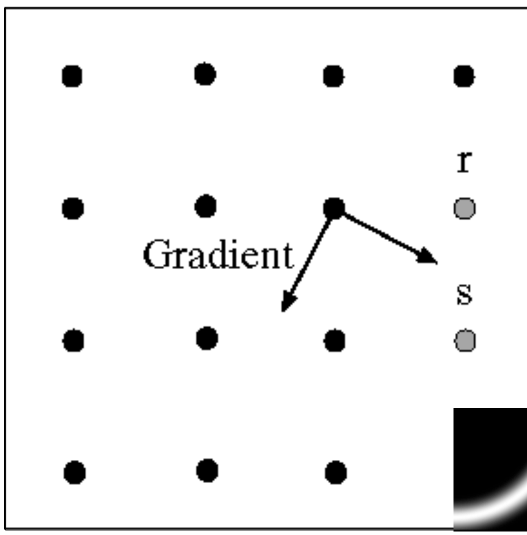
- Given a scale, how do we find edge points?
- If we set a threshold, then either lots of points near the edge are accepted, or none are.



We wish to mark points along the curve where the gradient magnitude is biggest in the gradient direction (best edge points).

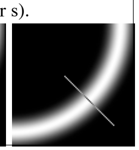
We can do this by looking for a maximum along a slice normal to the curve (non-maximum suppression).

These points should form a curve.



Predicting the next edge point

Assume the marked point is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (here either r or s).



Non-maximal suppression (alg 8.2)

(See book, page 180)

For non-marked points with sufficiently large gradient

Find a maximum along gradient, marking max as edge point, others as non edge.

Follow chain by looking perpendicular to gradient for points which are local max in gradient direction, and marking them as edges if their gradient magnitude is big enough, and marking other visited points as non-edge.

Remaining issues

- Check that maximum value of gradient value is sufficiently large
  - **hysteresis** method
    - use a high threshold to start edge curves and a low threshold to continue them.