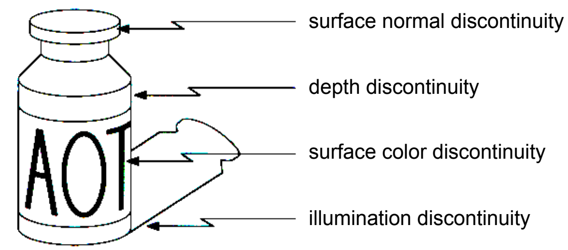
Edges and Scale

From Sandlot Science

Today's reading

- Cipolla & Gee on edge detection (available online)
- Szeliski 3.4.1 – 3.4.2

Origin of Edges



Edges are caused by a variety of factors

Detecting edges

What's an edge?

- intensity discontinuity (= rapid change)

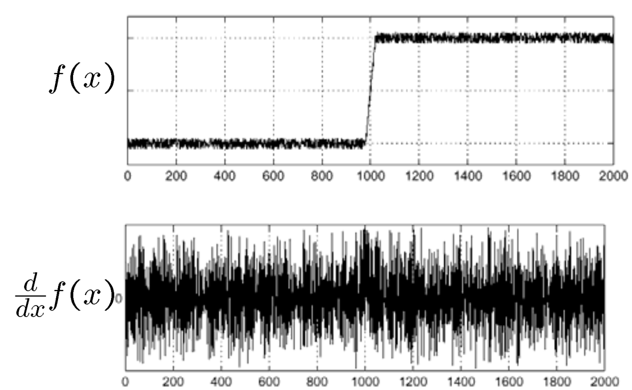
How can we find large changes in intensity?

- gradient operator seems like the right solution

Effects of noise

Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal



Where is the edge?

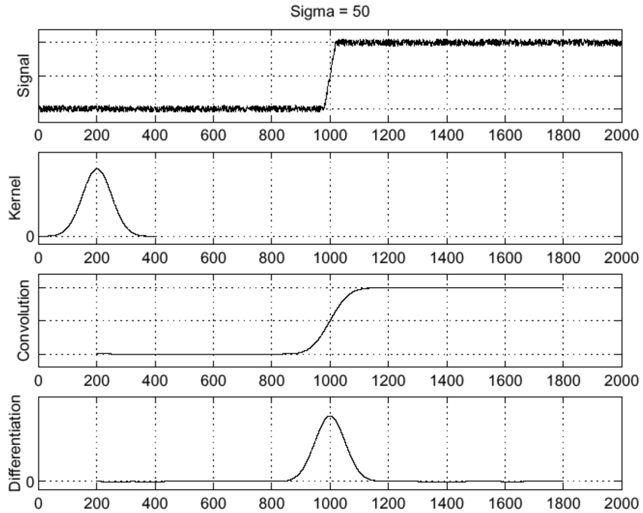
Solution: smooth first

f

h

$h \star f$

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



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

Associative property of convolution

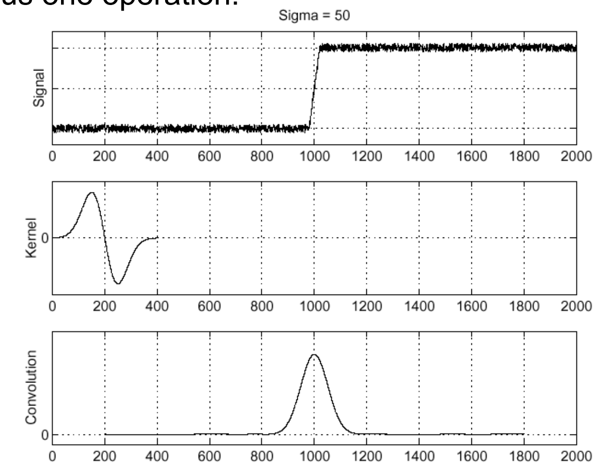
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

This saves us one operation:

f

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

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



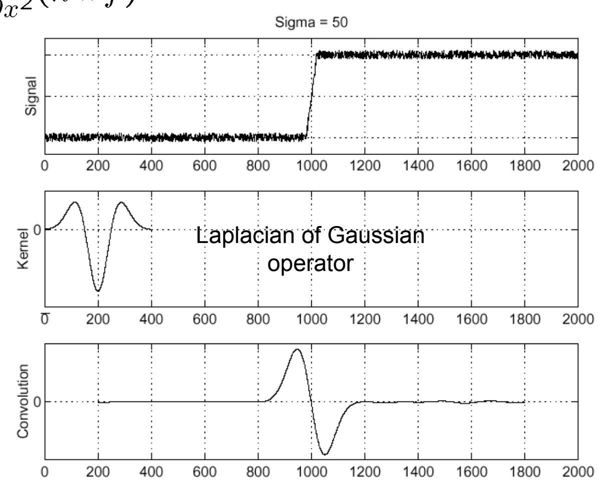
Laplacian of Gaussian

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

f

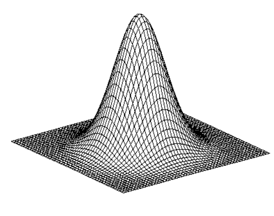
$\frac{\partial^2}{\partial x^2}h$

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



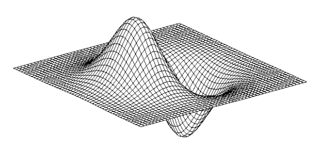
Where is the edge? Zero-crossings of bottom graph

2D edge detection filters



Gaussian

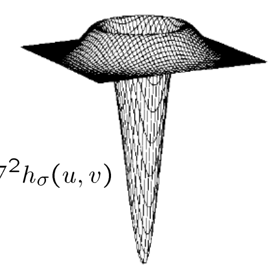
$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2}e^{-\frac{u^2+v^2}{2\sigma^2}}$



derivative of Gaussian

$\frac{\partial}{\partial x}h_\sigma(u,v)$

Laplacian of Gaussian



$\nabla^2 h_\sigma(u,v)$

∇^2 is the **Laplacian** operator:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

The Sobel operator

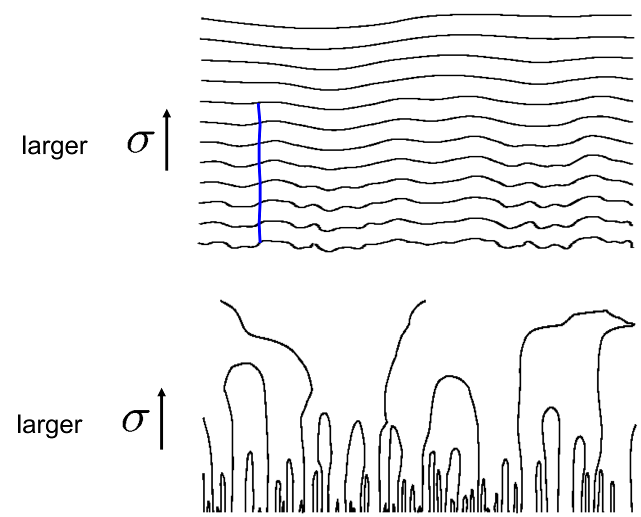
Common approximation of derivative of Gaussian

$$\frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$s_x \qquad\qquad s_y$$

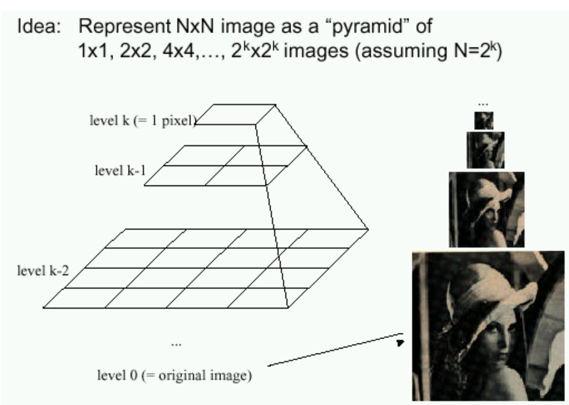
- The standard defn. of the Sobel operator omits the 1/8 term
  - doesn't make a difference for edge detection
  - the 1/8 term **is** needed to get the right gradient value, however

The effect of scale on edge detection



Scale space (Witkin 83)

Some times we want many resolutions

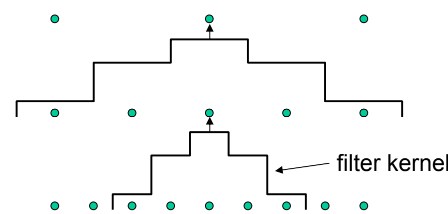


Known as a **Gaussian Pyramid** [Burt and Adelson, 1983]

- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*

Gaussian Pyramids have all sorts of applications in computer vision

Gaussian pyramid construction

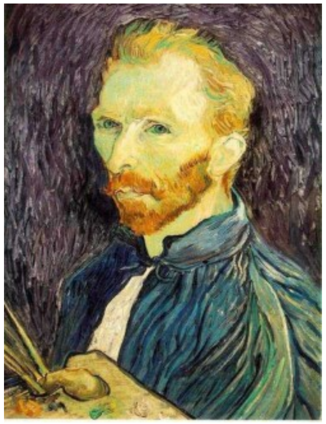


Repeat
- Filter
- Subsample

Until minimum resolution reached
- can specify desired number of levels (e.g., 3-level pyramid)

The whole pyramid is only 4/3 the size of the original image!

Subsampling with Gaussian pre-filtering



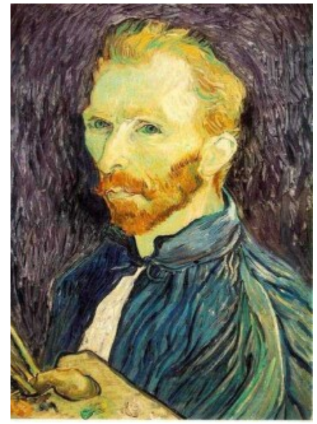
Gaussian 1/2

G 1/4

G 1/8

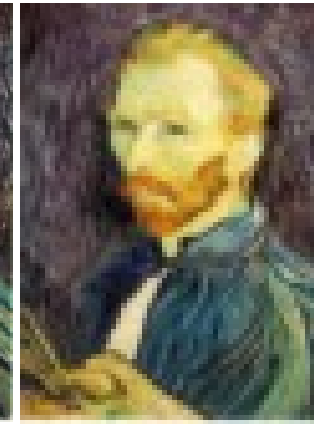
Filter the image, *then* subsample

Subsampling with Gaussian pre-filtering



Gaussian 1/2

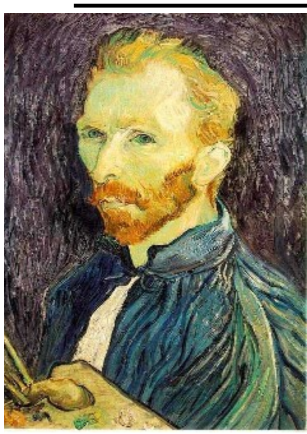
G 1/4



G 1/8

Filter the image, *then* subsample

Subsampling *without* pre-filtering

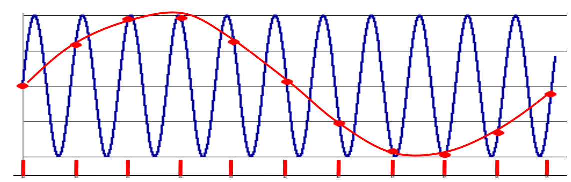


1/2

1/4 (2x zoom)

1/8 (4x zoom)

Sampling and the Nyquist rate



Aliasing can arise when you sample a continuous signal or image

- 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*
- formally, the image contains structure at different scales
  - called "frequencies" in the Fourier domain
- the sampling rate must be high enough to capture the highest frequency in the image

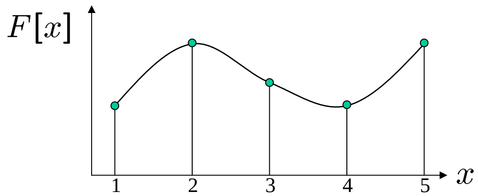
To avoid aliasing:

- sampling rate ≥ 2 * max frequency in the image
  - said another way: ≥ two samples per cycle
- This minimum sampling rate is called the **Nyquist rate**

Image resampling

So far, we considered only power-of-two subsampling

- What about arbitrary scale reduction?
- How can we increase the size of the image?



d = 1 in this example

Recall how a digital image is formed

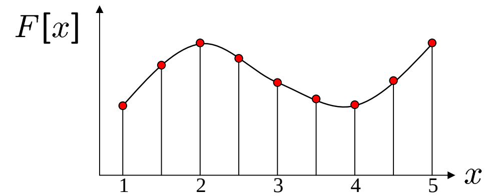
$$F[x,y] = \text{quantize}\{f(xd, yd)\}$$

- 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

Image resampling

So far, we considered only power-of-two subsampling

- What about arbitrary scale reduction?
- How can we increase the size of the image?



d = 1 in this example

Recall how a digital image is formed

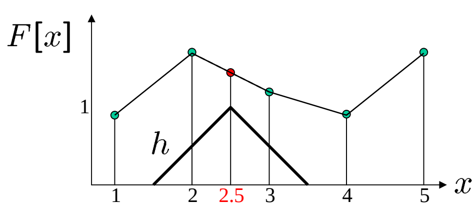
$$F[x,y] = \text{quantize}\{f(xd, yd)\}$$

- 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

Image resampling

So what to do if we don't know f

- Answer: guess an approximation $\tilde{f}$
- Can be done in a principled way: filtering



d = 1 in this example

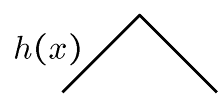
Image reconstruction

- Convert F to a continuous function
  $f_F(x) = F(\frac{x}{d})$ when $\frac{x}{d}$ is an integer, 0 otherwise
- Reconstruct by cross-correlation:

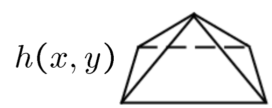
$$\tilde{f} = h \otimes f_F$$

Resampling filters

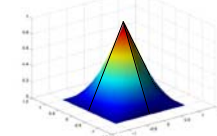
What does the 2D version of this hat function look like?



performs
linear interpolation



(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
  - fit 3rd degree polynomial surface to pixels in neighborhood