Lecture 11: LoG and DoG Filters

Today’s Topics

- Laplacian of Gaussian (LoG) Filter
  - useful for finding edges
  - also useful for finding blobs!
- approximation using Difference of Gaussian (DoG)

Recall: First Derivative Filters

- Sharp changes in gray level of the input image correspond to “peaks or valleys” of the first-derivative of the input signal.

\[
F(x) \quad \quad \quad \quad F'(x)
\]

(1D example)

Second-Derivative Filters

- Peaks or valleys of the first-derivative of the input signal, correspond to “zero-crossings” of the second-derivative of the input signal.

Numerical Derivatives

See also T&V, Appendix A.2

Taylor Series expansion

\[
f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + O(h^4)
\]

\[
f(x-h) = f(x) - hf'(x) + \frac{h^2}{2}f''(x) - \frac{h^3}{6}f'''(x) + O(h^4)
\]

\[
f(x+h) + f(x-h) = 2f(x) + h^2f''(x) + O(h^4)
\]

\[
f(x-h) - 2f(x) + f(x+h) = f''(x) + O(h^2)
\]

\[
\begin{bmatrix} 1 & -2 & 1 \end{bmatrix} \quad \text{Central difference approx to second derivative}
\]

Example: Second Derivatives

\[
\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}
\]

2nd Partial deriv. wrt x

\[
\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}
\]

2nd Partial deriv. wrt y

\[
I_{xx} = \frac{\partial^2 I(x,y)}{\partial x^2}
\]

\[
I_{yy} = \frac{\partial^2 I(x,y)}{\partial y^2}
\]
Example: Second Derivatives

Ixx  Iyy

Benefit: you get clear localization of the edge, as opposed to the “smear” of high gradient magnitude values across an edge.

Compare: 1st vs 2nd Derivatives

Ixx  Iyy

Finding Zero-Crossings

An alternative approx to finding edges as peaks in first deriv is to find zero-crossings in second deriv.

In 1D, convolve with [1 -2 1] and look for pixels where response is (nearly) zero?

Problem: when first deriv is zero, so is second. i.e. the filter [1 -2 1] also produces zero when convolved with regions of constant intensity.

So, in 1D, convolve with [1 -2 1] and look for pixels where response is nearly zero AND magnitude of first derivative is “large enough”.

Edge Detection Summary

1D  2D

\[ \frac{dI(x)}{dx} \Rightarrow \Theta \]

\[ \frac{d^2I(x)}{dx^2} \Rightarrow 0 \]

\[ \left( I_x(x,y)^2 + I_y(x,y)^2 \right)^{1/2} \Rightarrow \Theta \]

\[ \tan \Theta = I_x(x,y)/I_y(x,y) \]

Finite Difference Laplacian

\[ I_{xx} + I_{yy} = \left( \begin{array}{cc} 1 & -2 \\ -2 & 1 \end{array} \right) * I \]

\[ = \left( \begin{array}{ccc} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{array} \right) * I \]

Laplacian filter \( \nabla^2 I(x,y) \)

Example: Laplacian

Ixx + Iyy
Example: Laplacian

\[ I_{xx} + I_{yy} \]

\[ \nabla^2 I(x,y) \]

Notes about the Laplacian:

- \( \nabla^2 I(x,y) \) is a SCALAR
  - Can be found using a SINGLE mask
  - Orientation information is lost
  - Very noise sensitive!
- It is always combined with a smoothing operation:

\[ I(x,y) \xrightarrow{\text{Smooth}} \nabla^2 I(x,y) \xrightarrow{\text{Laplacian}} O(x,y) \]

LoG Filter

- First smooth (Gaussian filter),
- Then, find zero-crossings (Laplacian filter):
  \[ O(x,y) = \nabla^2 (I(x,y) \ast G(x,y)) \]

Just another linear filter.

\[ \nabla^2 (f(x,y) \ast G(x,y)) = \nabla^2 G(x,y) \ast f(x,y) \]

1D Gaussian and Derivatives

\[ g(x) = e^{-\frac{x^2}{2\sigma^2}} \]

\[ g'(x) = -\frac{x}{\sigma^2}e^{-\frac{x^2}{2\sigma^2}} \]

\[ g''(x) = \left( \frac{x^2}{\sigma^4} - \frac{1}{\sigma^2} \right)e^{-\frac{x^2}{2\sigma^2}} \]

Second Derivative of a Gaussian

\[ g''(x) = \left( \frac{x^2}{\sigma^4} - \frac{1}{\sigma^2} \right)e^{-\frac{x^2}{2\sigma^2}} \]

Effect of LoG Operator

LoG "Mexican Hat"

Band-Pass Filter (suppresses both high and low frequencies)
Why? Easier to explain in a moment.
Zero-Crossings as an Edge Detector

Raw zero-crossings (no contrast thresholding)

LoG sigma = 2, zero-crossing

LoG sigma = 4, zero-crossing

LoG sigma = 8, zero-crossing

Note: Closed Contours

You may have noticed that zero-crossings form closed contours. It is easy to see why…

Think of equal-elevation contours on a topo map.

Each is a closed contour.

Zero-crossings are contours at elevation = 0.

remember that in our case, the height map is of a LoG filtered image - a surface with both positive and negative “elevations”

Other uses of LoG: Blob Detection


Pause to Think for a Moment:

How can an edge finder also be used to find blobs in an image?
Example: LoG Extrema

LoG filter extrema locates “blobs”
- maxima = dark blobs on light background
- minima = light blobs on dark background

Scale of blob (size; radius in pixels) is determined by the sigma parameter of the LoG filter.

LoG Blob Finding

Observe and Generalize

LoG looks a bit like an eye.
and it responds maximally in the eye region!

Observe and Generalize

LoG sigma = 2
LoG sigma = 10

Observe and Generalize

LoG Extrema, Detail

LoG sigma = 2

Observe and Generalize

LoG
Derivative of Gaussian

Looks like dark blob on light background
Looks like vertical and horizontal step edges

Recall: Convolution (and cross correlation) with a filter can be viewed as comparing a little “picture” of what you want to find against all local regions in the image.
Observe and Generalize

Key idea: Cross correlation with a filter can be viewed as comparing a little "picture" of what you want to find against all local regions in the image.

Maximum response: dark blob on light background
Minimum response: light blob on dark background

Efficient Implementation

Approximating LoG with DoG

LoG can be approximate by a Difference of two Gaussians (DoG) at different scales.

Separability of and cascadability of Gaussians applies to the DoG, so we can achieve efficient implementation of the LoG operator.

DoG approx also explains bandpass filtering of LoG (think about it. Hint: Gaussian is a low-pass filter)

Maximum response:
vertical edge; lighter on left
Minimum response:
vertical edge; lighter on right

Back to Blob Detection

Lindeberg: blobs are detected as local extrema in space and scale, within the LoG (or DoG) scale-space volume.

Efficient Implementation

LoG can be approximate by a Difference of two Gaussians (DoG) at different scales.

Separability of and cascadability of Gaussians applies to the DoG, so we can achieve efficient implementation of the LoG operator.

DoG approx also explains bandpass filtering of LoG (think about it. Hint: Gaussian is a low-pass filter)

Other uses of LoG:

Blob Detection

Gesture recognition for the ultimate couch potato

Other uses for LOG:

Image Coding

- Coarse layer of the Gaussian pyramid predicts the appearance of the next finer layer.
- The prediction is not exact, but means that it is not necessary to store all of the next fine scale layer.
- Laplacian pyramid stores the difference.
Other uses for LOG: Image Coding

The Laplacian Pyramid as a Compact Image Code  
Burt, P., and Adelson, E. H.,  