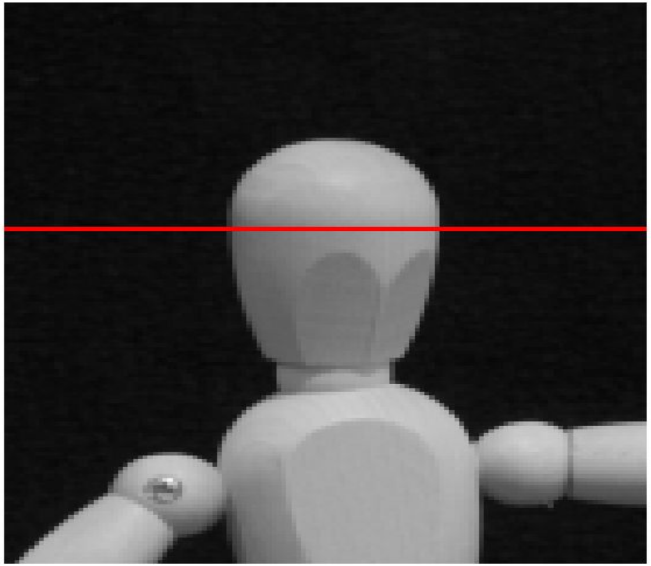
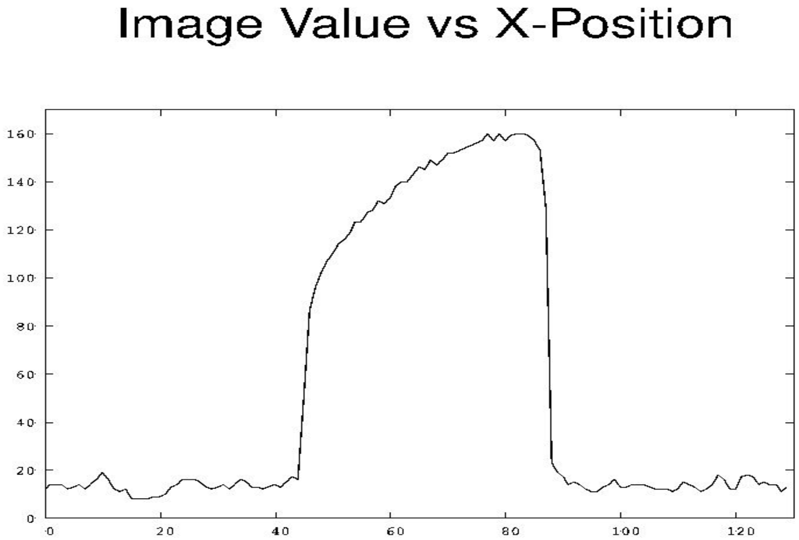
Low Level Features

Edges

01/27/16

Edge detection

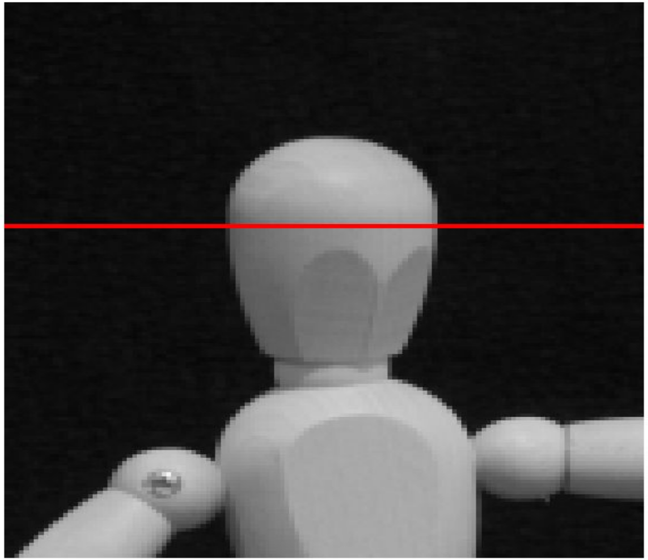




Abrupt transition in intensity between two regions

From http://www.coe.utah.edu/~cs4640/

Edge detection



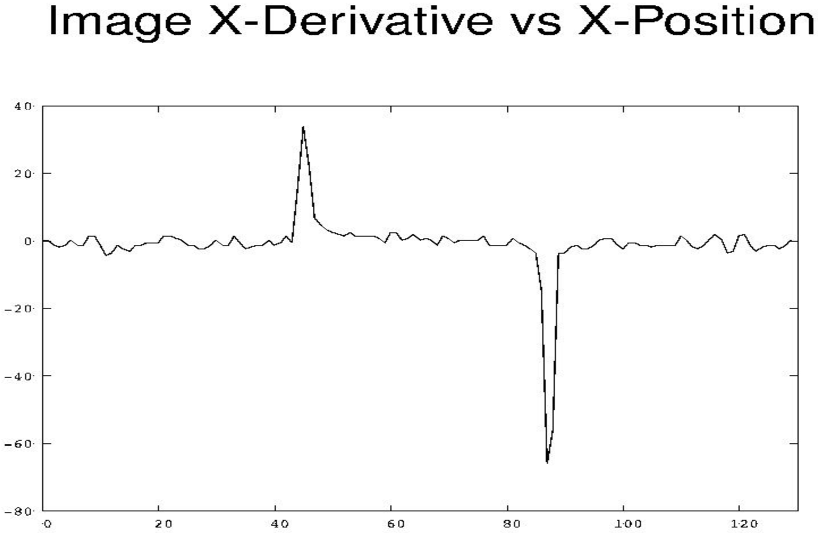
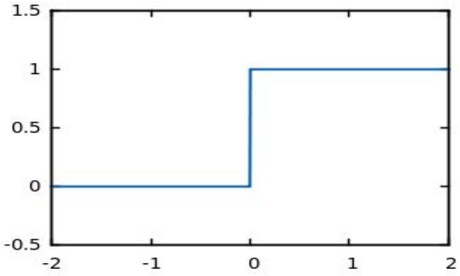


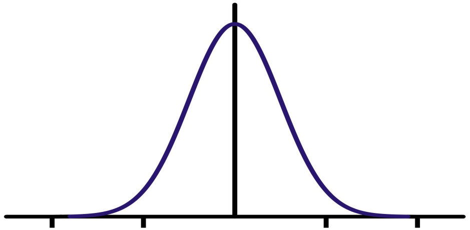
Image derivatives are high (or low) at edges

From http://www.coe.utah.edu/~cs4640/

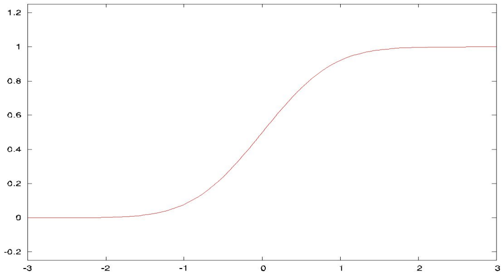
Edge in 1D



*



=



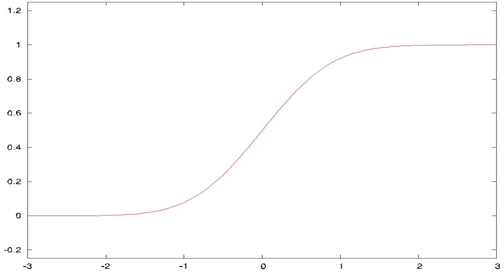
1D step

1D Gaussian blur

Edge model

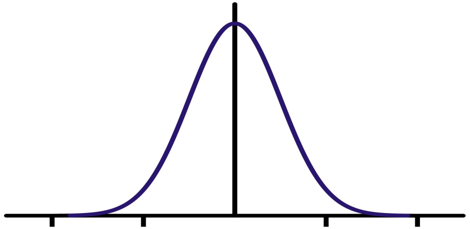
Derivative of an Edge

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



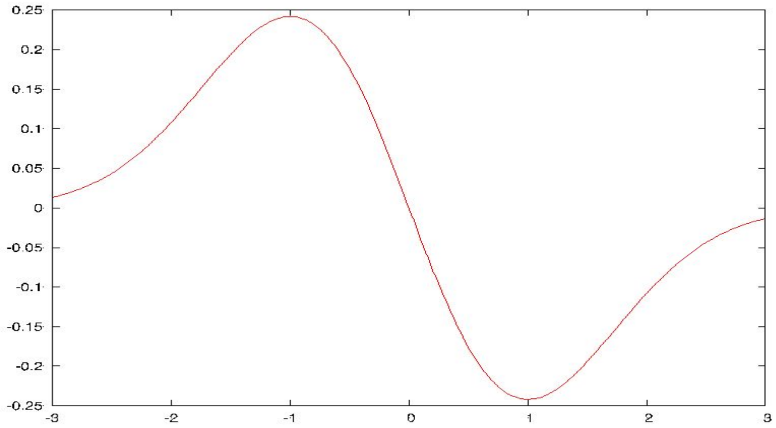
Edge model

=

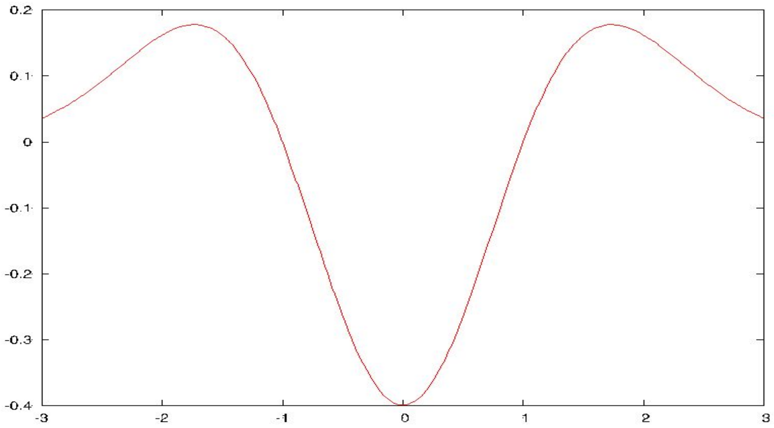


Gaussian

Derivatives of Gaussians (DOG)



$$D * G$$



$$D * D * G$$

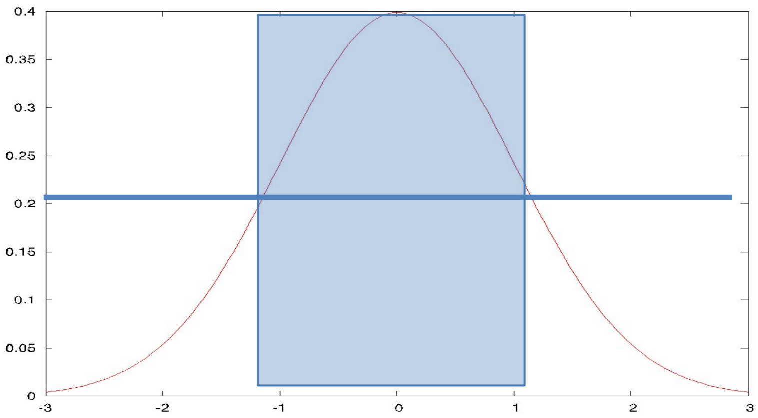
G = Gaussian

$$D = \begin{bmatrix} 0.5 & 0 & -0.5 \end{bmatrix}$$

Images from http://www.coe.utah.edu/~cs4640/

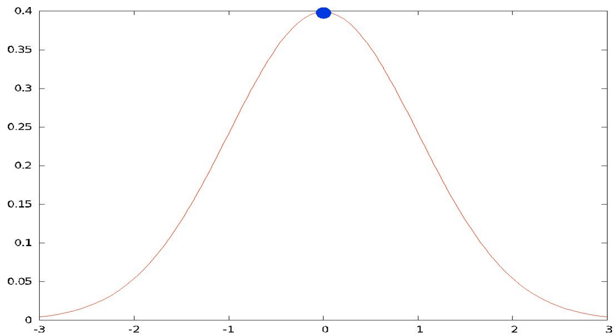
Thresholding to detect Edges

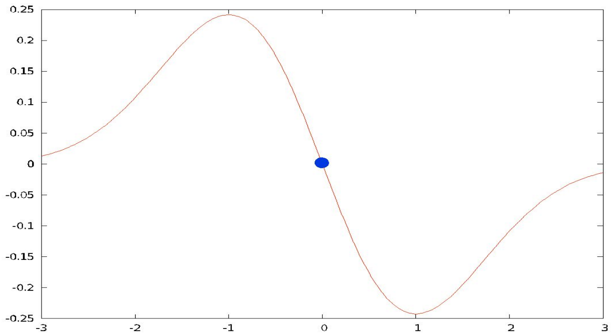
- At an edge, the gradient looks like a Gaussian.
- We can threshold to detect edges.
- Threshold leaves a “fat” edge!



Zero-crossing of the Second Derivative of Gaussians

- To avoid having a “fat” edge, we can calculate a single local maximum.
- This happens when the second derivative is zero.





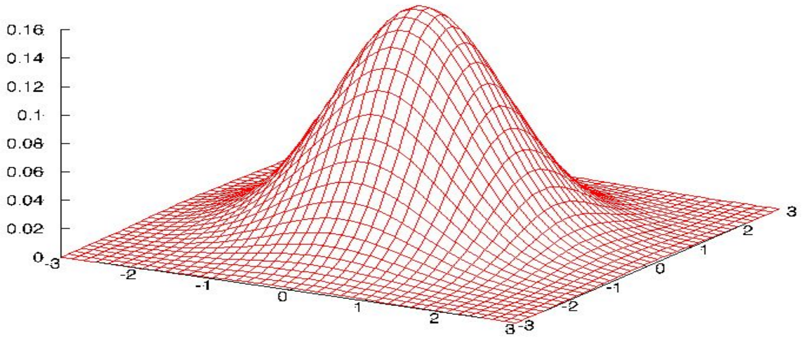
2D extension

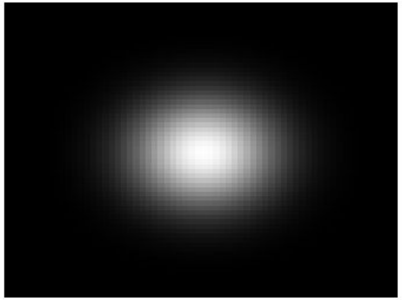
- In 2D we have x and y derivatives.
- We can compute the gradient of the Image, but using DOG kernels for x and y directions.
- Zero-crossings of the second derivative now are calculated using the Laplacian of the image.

$$\Delta I(x, y) = \frac{\partial^2}{\partial x^2} I(x, y) + \frac{\partial^2}{\partial y^2} I(x, y)$$

From http://www.coe.utah.edu/~cs4640/

2D Gaussian filter

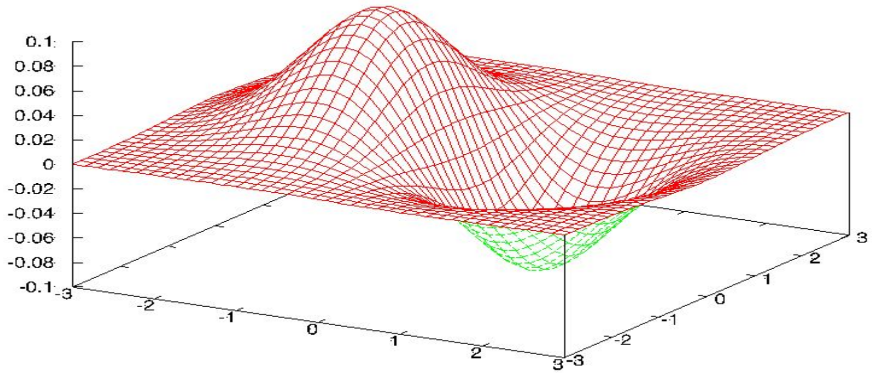


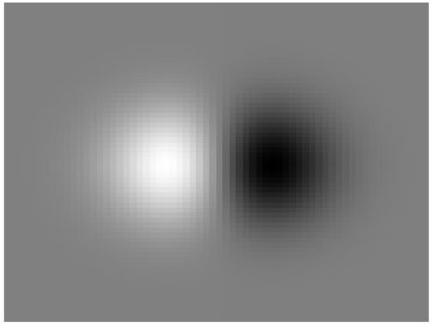


$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Images from http://www.coe.utah.edu/~cs4640/

2D Gaussian X-Derivative

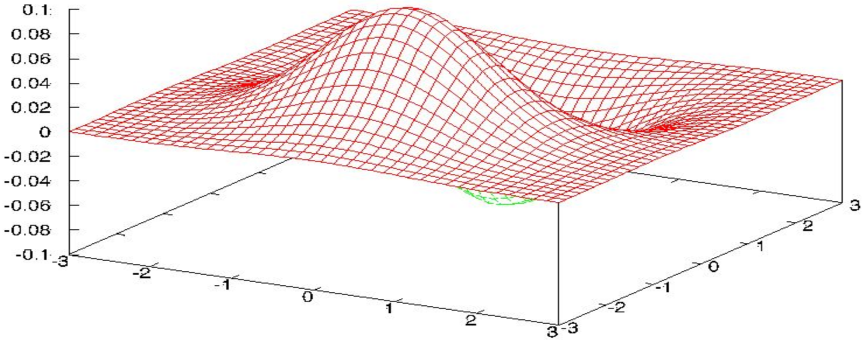


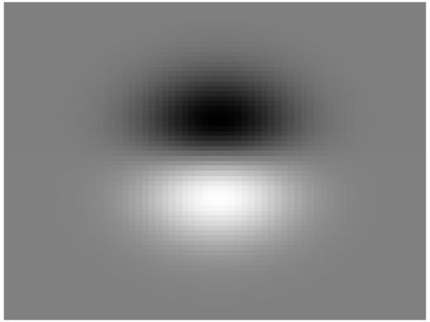


$$\frac{\partial}{\partial x} G_\sigma(x, y) = \frac{-x}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Images from http://www.coe.utah.edu/~cs4640/

2D Gaussian Y-Derivative

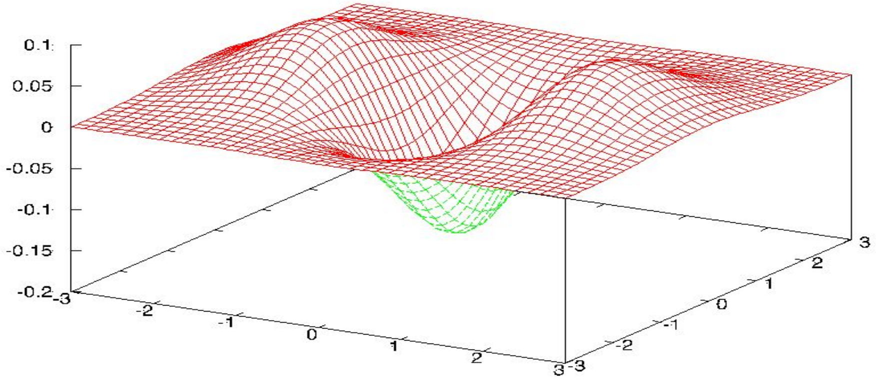


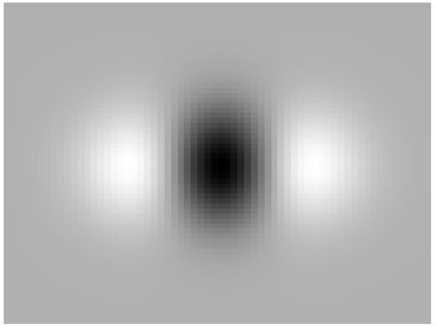


$$\frac{\partial}{\partial y} G_\sigma(x, y) = \frac{-y}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Images from http://www.coe.utah.edu/~cs4640/

2D Gaussian Second X-Derivative

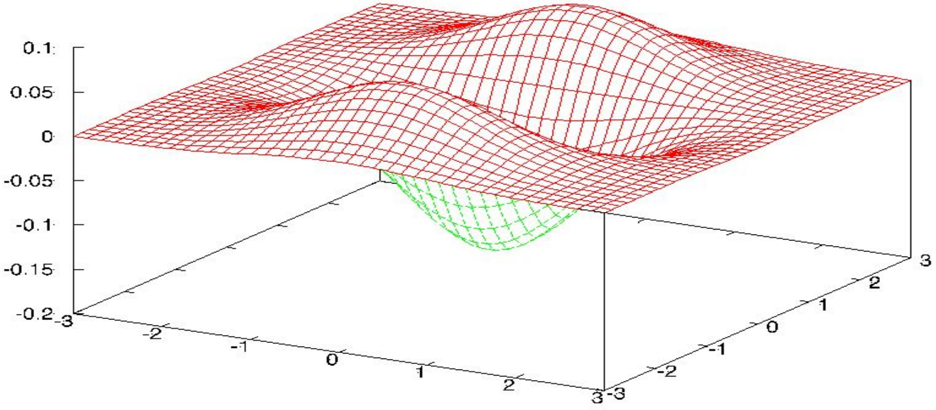


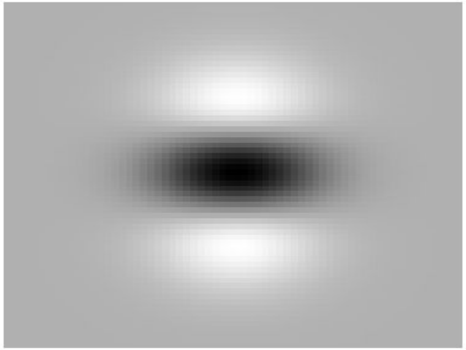


$$\frac{\partial^2}{\partial x^2} G_\sigma(x, y) = \frac{x^2 - \sigma^2}{2\pi\sigma^6} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Images from http://www.coe.utah.edu/~cs4640/

2D Gaussian Second Y-Derivative





$$\frac{\partial^2}{\partial y^2} \mathbf{G}_\sigma(x, y) = \frac{y^2 - \sigma^2}{2\pi\sigma^6} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Images from http://www.coe.utah.edu/~cs4640/

2D Edge Detection with Laplacian

1. Compute gradient magnitude using DOG kernels.
2. Threshold the gradient magnitude; label 1 above threshold, and 0 otherwise.
3. Compute Laplacian image using second-DOG kernels.
4. Find zero-crossings; set 1 at zero-crossing, 0 elsewhere.
5. Combine images from step 2 and 4 (using AND operation).

Compute Zero Crossings

- For every pixel do:
  - Look at your four neighbors (left, right, up, and down).
  - If they have the same sign as the current pixel, continue; not a zero crossing.
  - If the current pixel has the smallest absolute value compared to the four neighbors with opposite sign, then current pixel is a zero crossing.

From http://www.coe.utah.edu/~cs4640/

Canny Edge detector

- In Matlab use the ‘edge’ function.
- In OpenCV use the ‘Canny’ function.