Computer Vision Assignment 1: Filtering

Robrecht Jurriaans (5887380), Taco Cohen (6394590)

November 7, 2012

1 Gaussian Filters

1.1 1D Gaussian Filter

We implemented the 1D Gaussian in `gaussian.m`. We made sure the kernel size is about $3 * \sigma$ and is always odd. For this purpose we used the formula $2 * \lfloor 1.5 * \sigma \rfloor + 1$.

Because the filter has a finite size, the sum of the filter values will not be one in a naive implementation. For this reason, we must normalize the kernel after calculating the values of the Gaussian at each kernel entry. To save computation, we used the following equality:

$$\begin{aligned}\frac{G_\sigma(x)}{\sum_{x'=-h}^{h} G_\sigma(x')} &= \frac{\frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{x^2}{2\sigma^2})}{\sum_{x'=-h}^{h}\frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{x'^2}{2\sigma^2})}\\ &= \frac{\exp(-\frac{x^2}{2\sigma^2})}{\sum_{x'=-h}^{h}\exp(-\frac{x'^2}{2\sigma^2})}\end{aligned}$$

That is, we leave out the normalization of each index, because it falls out in the normalization of the entire kernel anyway.

1.2 Convolving an image with a 2D Gaussian

Look at all the programming-fucks I'm giving

1.3 Comparing with Matlab's Gaussian Filter

Thanks to the inherent separability of Gaussians, the implementation that convolves on one dimension at the time is slightly faster than the normal 2D implementation. However, when taking a difference image, it becomes clear that there is a difference between both implementations. The sum of squared error between both resulting images is in the order of 10^{-17}. This difference is also found in the kernels as produced

1.4 Gaussian Derivative

1.5 Gradient Magnitude and Orientation

1.5.1 Quiver before my magnitude

1.5.2 Magnitude and orientation for different σ

1.5.3 Threshold

1.5.4 Second Order Derivative

1.5.5 Impulse

1.6 Convolving an Image with a 2D Gaussian