Computer Vision Assignment 2

Wolf Vos 10197923 & Kasper Bouwens 6137547

February 2016

1 Gaussian Filters

1.1 1D Gaussian Filter

To build a kernel we used the built in function linspace with three arguments. Linspace is a function that generates a number of points (equal to the third argument) between an upper and lower boundary. For the upper boundary we chose a number between the kernel size minus one, divided by two. The lower boundary was the same as the negative upper boundary. Next, The values in the array we created were inputted in the gaussian function. In our case this resulted in a gaussian kernel that was exactly the same as the built in fspecial function with parameters 'gaussian', [1, kernelLength] and sigma, No matter what sigmas or kernel lengths were chosen. Another way of doing this would be to have an n number of points (egual to the kernel length) between -3σ and 3σ because 99.7 % of the data lies between three standard deviations.

1.2 Convolving an image with a 2D Gaussian

We create a function that takes an image path, sigma in the x direction and sigma in the y direction as input. The kernel length is set to 11, 5 on each side of the origin. We used the matlab built-in *conv2()*. *conv2()* has three different padding settings:

Full is a setting that will produce a convoluted image which has rows added to the image matrix. The amount of rows depends on the length of the kernel, for a kernel of odd length it will be $\frac{1}{2}(kernelLength - 1)$. For an even length it will be half the kernel length. This is the full convolution.

Same is a setting that will produce a convoluted image which has no rows added to the image and thus it will return an convoluted image of the original size of the image.

Valid is a setting that will produce a convoluted image which has rows removed from the image. Any row that is involved by the zero padding around the image will be removed, but this results in a smaller image in that direction of the convolution.

2D vs 1D As is assumed, it is true that a convolving an image with kernel A and kernel B subsequently (figures 1) yields the same result as convolving kernel A with kernel B and convolving the image afterwards (figure 3).

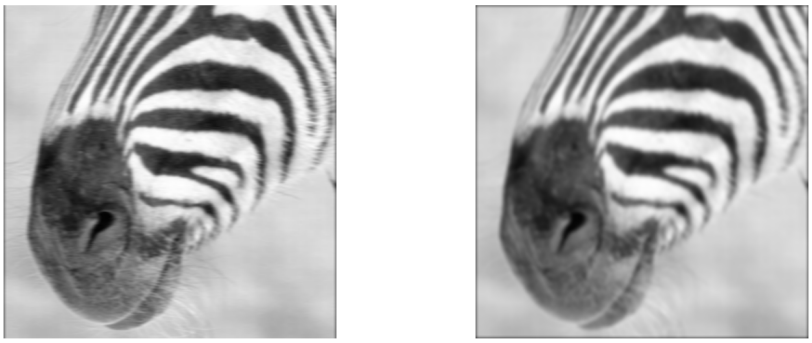

Figure 1: Convoluted image in x direction of the left. X and Y on the right.

Figure 2: 2D convolution

Kernel seperability is a very important aspect of the convolution because it can decrease the computational complexity of convoluting images with complex

kernels. the decrease of computational time will be very important when you are working with video's for instance.

1.3 Gaussian Derivative

Trying a range of different values for sigma results in thicker edges. Also many smaller edges will be blurred out. We tried sigma 1 which shows much richer detail on some of the smaller edges up to sigma 5 which showed much thicker and blury edges. The differences between σ and 5σ are plotted in figure 4 to 9. Edge detection can be performed by taking both convoluted images, squaring them, adding them together and taking the root.

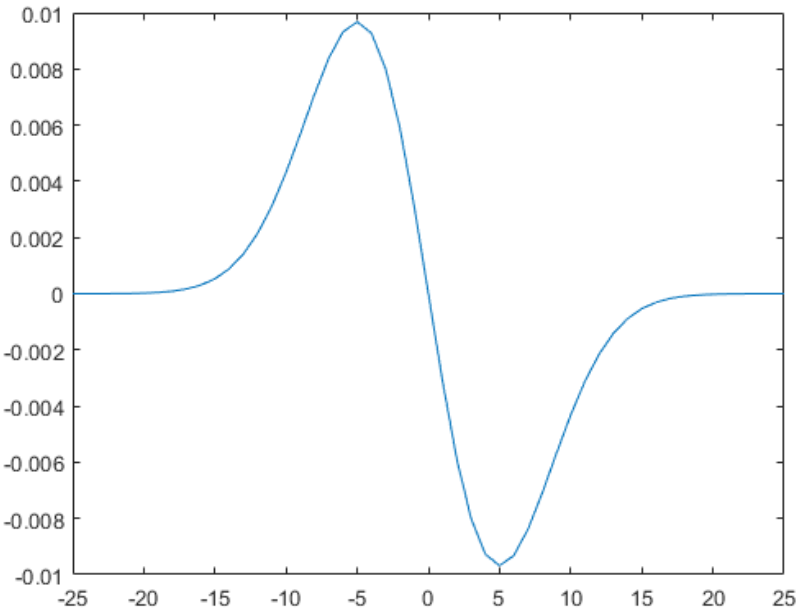

Figure 3: $\sigma = 5$, gaussian derivative

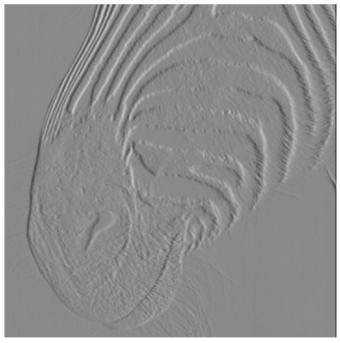

Figure 4: $\sigma = 1$, kernel in x direction

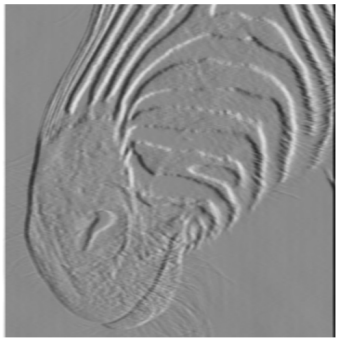

Figure 5: $\sigma = 5$, kernel in x direction

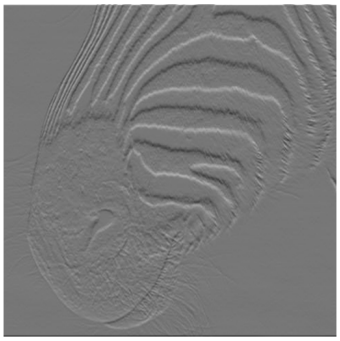

Figure 6: $\sigma = 1$, kernel in Y direction

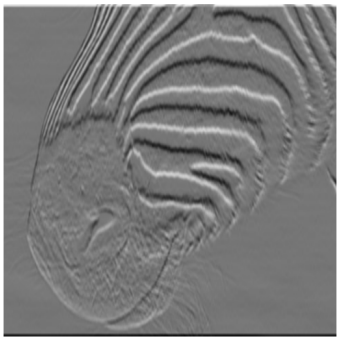

Figure 7: $\sigma = 5$, kernel in Y direction

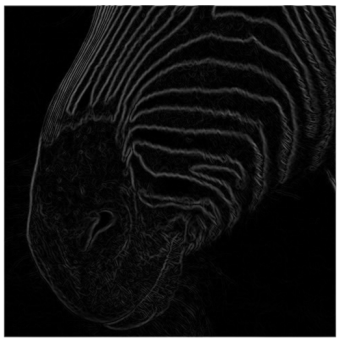

Figure 8: $\sigma = 1, \sqrt{(x^2 + y^2)}$

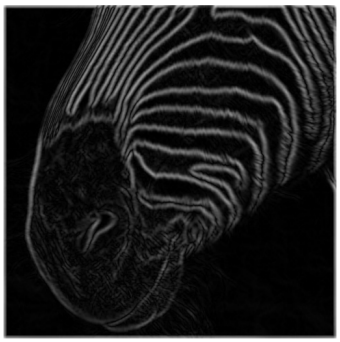

Figure 9: $\sigma = 5, \sqrt{(x^2 + y^2)}$