Computer Vision - Assignment 2

Arthur Bražinskas
11138904

24 February 2016

1 1D Gaussian Filter

Our implementation differs from the Matlab's, see eq. 1 and 2, in several ways. First of all, their Gaussian filter is 2 dimensional, and computation of filter values differ from our approach, and therefore we decided to prove that our filters differ non-linearly. First, we produced Gaussian filters both with our function and with Matlab's. Then as we know that a Gaussian filter is a rank 1 matrix we performed a decomposition of a matrix into an outer product of a column and a row, and noticed that vectors are identical, and thus the column space is the same as the row space as matrix is in fact symmetric. Finally, to prove that our 1D filter is not a linear combination of the basis of the Matlab's filter, we tried to solve a linear equation: $Ax = b$, where A is the Matlab's filter, and b is our 1D filter, and observed that it's not possible to solve (without performing projection, i.e. least squares). So we conclude that filters are non-linearly different.

$$h_g(n_1, n_2) = \exp(\frac{-(n_1^2 + n_2^2)}{2\sigma^2}) \tag{1}$$

$$h(n_1, n_2) = \frac{h_g(n_1, n_2)}{\sum_{n_1} \sum_{n_2} h_g} \tag{2}$$

2 Convolving an image with a 2D Gaussian

We convolved an image with two 1D filters and compared to the result of applying 2 D filter, and it was equivalent with 0.001 tolerance. It's important to note that it's equivalent while two 1 D filters are equivalent (same σ). We show pipeline snapshots in fig .1. We also have tried to perform first horizontal filtering and then vertical filters, and the results are different from 2D filtering.

We have tried different parameters to *conv2* function, and 'full' makes the function to return full two-dimensional convolution (with padding i.e. border), 'same' return only the central part of the convolution of the same size as the image, and 'valid' returns only those parts that are computed without the zero-padded edges. The benefits of the separability are that it's possible to perform convolution in $O(n)$ time instead of $O(n^2)$.

3 Gaussian derivative

From the experiments we observe that standard deviation (σ) affects the sensitivity (output visibility) for edges, see fig. 3. For example, $\sigma = 1$ makes most edges, even minor ones, sharp and visible, while with $\sigma = -5$ and 5, we can see only the main edges. It's easy to see from the equation of the filter that we divide by σ^2, thus the larger is the absolute value of σ, the smaller will be the output pixel values for edges. To produce the previously mentioned plot it was necessary to upscale intensity of pixels by a factor of 10. If we use $\sigma = 0.5$ we can observe from fig. 2 that even without upscaling, edges are detected very well but with some noise detected as edges too. To conclude the experiment results, we can say that σ controls sensitivity to edges, and if chosen incorrectly can make the procedure detect noise as edges or detect only the main edges.

The obvious application of such filter is edge detection that can be used to better understand images and produce informative features.

(a) Original image

(b) 1D horizontal filtering

(c) 1D vertical filtering

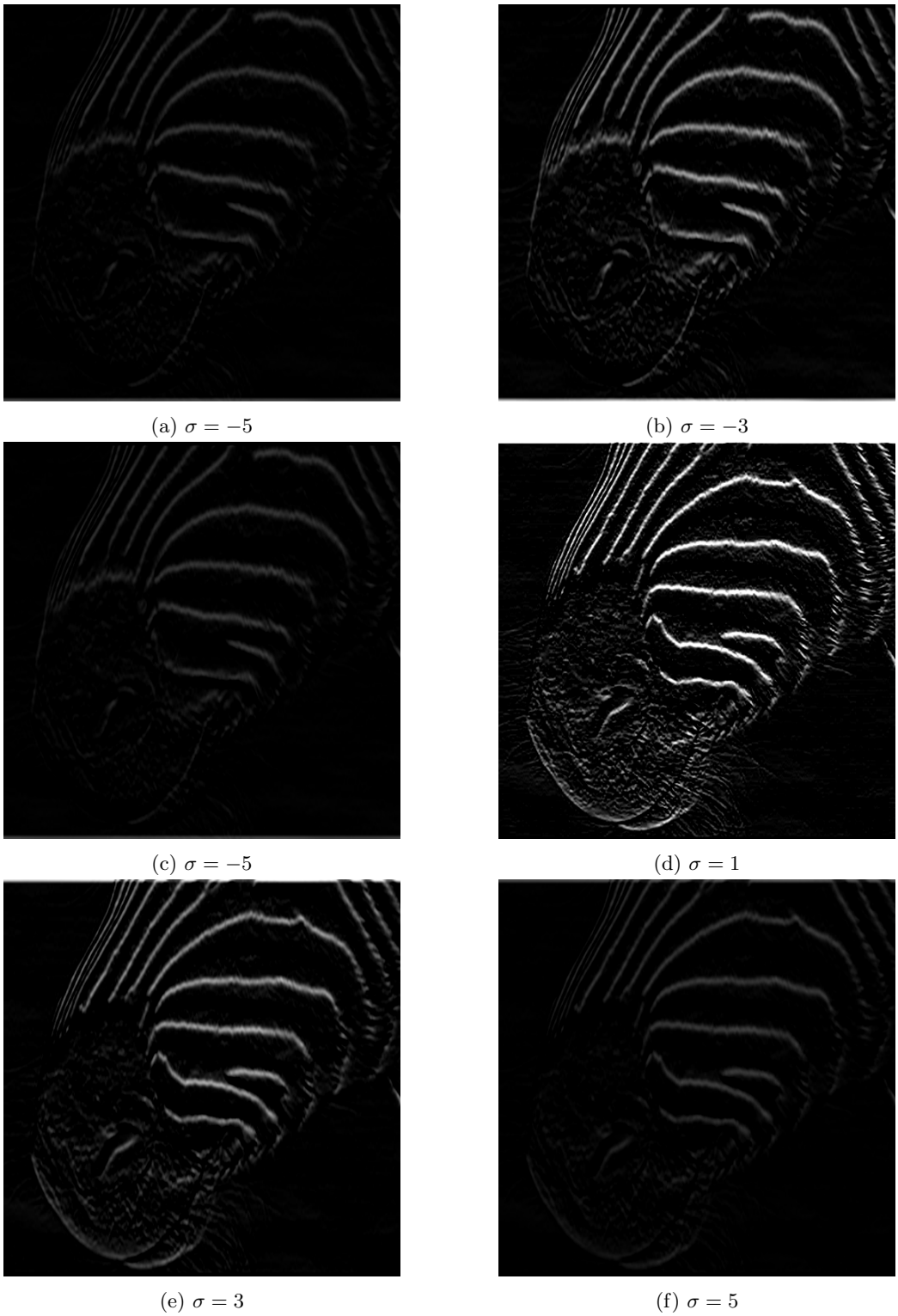
(d) 1D vertical and then horizontal filtering

(e) 1D vertical and horizontal filtering

(f) 2D filtering

Figure 1: Different filtering pipelines with $\sigma = 2$

Figure 2: Edge detection with $\sigma = 0.5$



(a) $\sigma = -5$ (b) $\sigma = -3$

(c) $\sigma = -5$ (d) $\sigma = 1$

(e) $\sigma = 3$ (f) $\sigma = 5$

Figure 3: First order derivative of Gaussian filter w.r.t y application under different standard deviations