Computer Vision 1 - AI Assignment2
Linear Filters: Gaussians and Derivatives

September 15, 2014

All the files should be zipped and sent to **uva.computervision1(at)gmail.com** before **22-09-2014** , **23.59** (Amsterdam Time).

1 Gaussian Filters

In this section, you will be working with Gaussian filters and applying them to images, exploiting their separability property.

1.1 1D Gaussian Filter

In this first part, you will be implementing a 1D discrete Gaussian filter:

$$G_\sigma = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{x^2}{2\sigma^2}), \tag{1}$$

where σ is the variance of the Gaussian. The Matlab fucntion should look like this:

```
function G = gaussian(sigma)
...
end
```

1.2 Convolving an image with a 2D Gaussian

You will write a function to convolve an image with a 2D Gaussian. One of the most important properties of Gaussian kernels is separability. Therefore, convolving an image with a 2D Gaussian is equivalent to convolving the image along the x-axis and the y-axis **separately** with a 1D Gaussian. The Matlab function should use the previous function and should look like this:

```
function imOut = gaussianConv(image_path,sigma_x,sigma_y)
...
end
```

Hint: use Matlab function conv2.

1.3 Comparing with Matlab's Gaussian Filter

Matlab already has a built-in function implementing 2D Gaussian kernels. The Matlab code for using it is:

```
G = fspecial('gaussian',n,sigma)
imOut = conv2(imIn,G,'same')
```

In this part, you can compare your implementation of 2D Gaussian convolution, through two 1D operations, to Matlab's. You can read any image and check the difference between your outputted image and Matlab's.

1.4 Gaussian Derivative

The Gaussian first order derivative is given by:

$$\begin{aligned} \frac{d}{dx}G_\sigma &= \frac{d}{dx}\left(\frac{1}{\sigma\sqrt{2\pi}}\exp(\frac{x^2}{2\sigma^2})\right) \\ &= -\frac{1}{\sigma^3\sqrt{2\pi}}x\exp(-\frac{x^2}{2\sigma^2}) \\ &= -\frac{x}{\sigma^2}G_\sigma \end{aligned} \tag{2}$$

Write a function which implements this first order derivative and applies it on a given image. Your function should look like this:

```
function [imOut Gd] = gaussianDer(image_path,G,sigma)
...
end
```