Aula Prática 4

Docente: Miguel Tavares Coimbra

1. **Spatial Convolution.**
   - [Optional] Create a test image using a drawing program of your choosing. This image should have a small dimension and should have controlled values. Example:

0	0	0	0	0
0	255	255	255	0
0	255	255	255	0
0	255	255	255	0
0	0	0	0	0

   - Create a function that applies a 3x3 mask over point (2,2) of the image. Remember:

$$g(2,2)=\sum_{i=-1}^{1}\sum_{j=-1}^{1} f(2+i,2+j).mask(i,j)$$

   - [Optional] Test various points of the test image where you already know what results you should obtain. Confirm that you are obtaining correct results.
   - A convolution consists on the application of this mask to the whole image. Remember:

$$g(x,y)=\sum_{i=-1}^{1}\sum_{j=-1}^{1} f(x+i,y+j).mask(i,j)$$

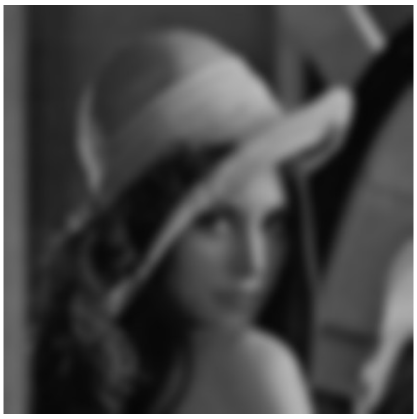
   - Implement a spatial convolution function so that it applies a mask to an image. To handle the border problem, obtain a smaller image that the original one.

2. **Spatial filters**
   - Create a function that applies a *mean filter* to an image. It should work on the image domain, using a 3x3 mask. Ignore the border problem, by not altering any value in the image border.

1	1	1
1	1	1
1	1	1

Mask of a 3x3 *mean filter*

[non-normalized mask!]



   - Consecutively apply the previous filter, noticing the importance of the border problem. A better solution is to create a larger image, filling these extra spaces with new values. Implement the three following alternatives:
     - **i.** New pixels have zero value (black color).
     - **ii.** New pixels have a value equal to its closest neighbor.
     - **iii.** New pixels have a value equal to a 'reflected' version of the original image.

3. **Noise reduction using spatial filters.**
   - Consider image *Imagem_AP4_2*, which has been degraded by salt and pepper noise. Apply the mean filter previously implemented.
   - Create a function that implements a *median filter* of size 3x3. Apply this filtre to the same image. What differences do you observe?
     Google: Java Arrays Sort