Computervision Lab 2
Filtering

David Van Hamme, Sanne Roegiers

February 8, 2019

1 Filtering

In this lab we look at some common filter operations. In image processing, filtering is mainly used to:

- blur images;
- sharpen images;
- remove noise;
- extract edges.

Filtering means that a window or kernel is moved over the image and each time a pixel is replaced by a value calculated from the entire window. In the case of linear filters, the kernel is a matrix of weights by which the surrounding pixels are multiplied before summing them in the central pixel.

2 Exercise 2

Write a program that reads a PNG image (filename through command line) and blurs it with a Gaussian filter. This filter replaces each pixel with a weighted average of the surrounding pixels. The weights in the kernel are determined by a 2D normal distribution around the central pixel, so nearby pixels have more influence than slightly more distant pixels. This type of filter is often used to remove *white noise* from the image. White noise is a form of noise in which each pixel has undergone a random deviation from its original value. Test your program on **whitenoise.png**. Show the original image and the blurred version side by side in two separate windows. New functions you need: **GaussianBlur**.

3 Exercise 3

Write a program that reads a PNG image (filename through command line) and applies *unsharp masking* to it. This is a technique to sharpen an image and involves the following:

- blur the image;
- determine the (absolute) difference between the original and the blurred image;
- add this difference to the original image.

Make sure you do not get overflow or saturation in your datatype! Use appropriate scaling factors. Test your program on **unsharp.png**. Show the original image and the sharpened version side by side.

4 Exercise 4

Write a program that reads a PNG image (filename through command line) and removes the *salt and pepper noise* with a median filter. Salt and pepper noise is a form of noise in which some pixels turn black or white. The median filter replaces every pixel by the median of the surrounding pixels. As a result, the outliers have no influence anymore, in contrast to a Gaussian filter, where the influence is only reduced. Test your program on **saltandpeppernoise.png**. Show the original image and the filtered version side by side. New features you need: **medianBlur**.

5 Exercise 5

Write a program that reads a PNG image (filename through command line), converts it into grayscale and calculates the horizontal first derivative. You do this by filtering with a horizontal Sobel kernel:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

This gives you a measure of the horizontal change in an image. Note that this filter may generate negative response values and your data type must be chosen to accomodate this. Test your program on **building.png**. New features: **Sobel**.

6 Exercise 6

Write a program that reads a PNG image (filename through command line) and filters it with a 15x15 kernel whose first 7 diagonal elements have the value **1/7**, all others are zero.

$$1/7 \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}$$

Make sure that the anchor point of the kernel (the pixel in which the calculated value ends up) is at the bottom right in the kernel instead of in the middle as usual. What do you expect this filter to do? Test your program on **blots.png**. Show the original and the filtered image next to each other. New functions: **filter2D**.

7 Report

Write a short report about how you solved the exercises. Include in this report your input and output images. Describe every new function that you used. Explain the basic algorithm and purpose of the functions and clarify the parameters and how you selected them. Upload your report in the form of **LabX_name.pdf** to the dropbox on Minerva.