Exercise Session 4

Image processing

Rolf Ingold, Anna Scius-Bertrand
DIVA Group, University of Fribourg, Switzerland

UNI FR
UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

Reminder: assignment 2 et 3

- Tonight: assignment 2, indexed colors

- Next week: assignment 3, histogram equalization
  - Deadline: October 17th, end of the day

- Submit your solution via ILIAS:
  - Number of results images asked (no more, no less)
  - A text file with your name, surname, GitHub link and a brief description of your algorithm.

Assignment 1

- Open an image in RGB mode
- Create an empty image to store the resulting pixel values

- Iterate through the image:

```
for y in range(height):
    for x in range(width):
        image_result.putpixel((x, y), image.getpixel((width - x - 1, y)))
```

- numpy:

  h: image[:,::-1]

  v: image[::-1]

  h&v: image[::-1, ::-1]

Take away – python image basis

- Pillow
  - Size: width, height = image.size
  - Getpixel: image.getpixel((x, y))
  - Putpixel: result.putpixel((x, y), (r, g, b))
  - Image_result: Image.new('RGB', (image_width, image_height))
  - Image to array: array = np.array(img)
  - Array to image: image = Image.fromarray(numpydata)

- Opencv and numpy (as np)
  - Size: width, height, channels = image.shape
  - Getpixel in cv2: (b, g, r) = image[x, y], in np: (r, g, b) = image[x, y]
  - Putpixel: image[x, y] = (b, g, r)
  - Image_result: np.zeros((width, height, channels), dtype=np.uint8)

Assignment 4: Local operators

- Goal: implement and apply different local filters with different sizes:
  - Observe the impact of the filter size
  - Compare two edge detection filters.
  - Implement statistical filters.
- Input and output images should have the same size.

Convolution in practice

h is the output, m the kernel size, x the input and w the convolution kernel:

$$h_{i,j} = \sum_{k=1}^{m} \sum_{l=1}^{m} w_{k,l} x_{i+k-1,j+l-1}$$

$$\begin{pmatrix} 0 & 1 & 2 \\ 2 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix}$$

h at 0,0 = 0*3 + 1*3 + 2*2 +
2*0 + 2*0 + 0*1 +
0*3 + 1*1 + 2*2

h at 0,0 = 12

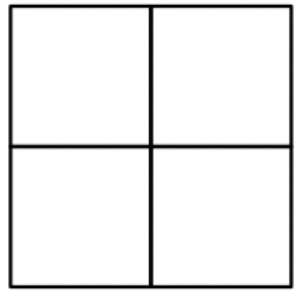
Which output?

3	1	1	0
4	1	0	2
5	0	1	2
1	0	2	1

Input image

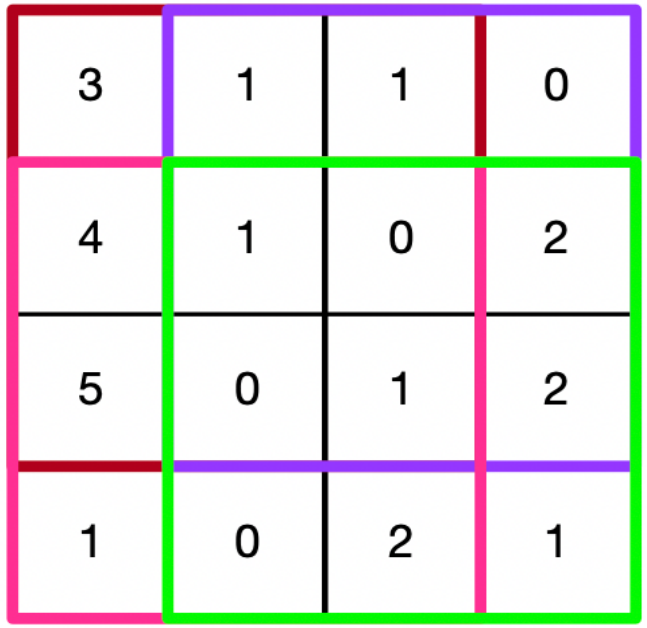
0	0	2
1	1	0
0	0	0

Kernel

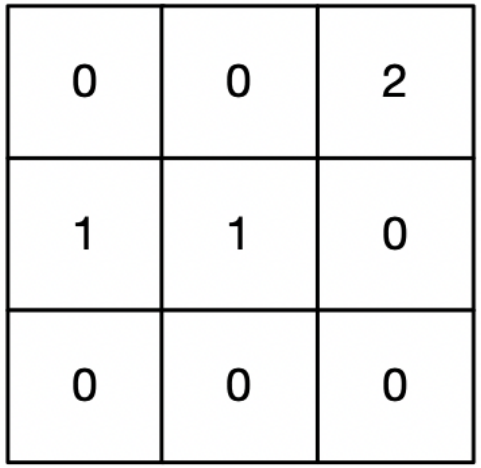


Output image

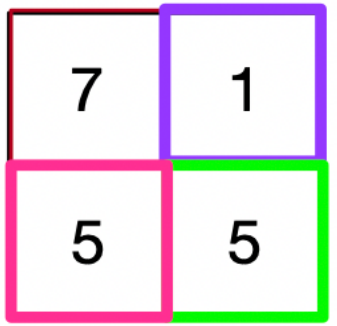
Result



Input image



Kernel



Output image

$$7 = 0*3 + 0*1 + 2*1 + 1*4 + 1*1 + 0*0 + 0*5 + 0*0 + 0*2$$
$$1 = 2*0 + 1*1 + 1*0$$
$$5 = 2*0 + 1*5 + 1*0$$
$$5 = 2*2 + 1*0 + 1*1$$

Part 1 and 2

1. Blurring by convolutions

- (a) Implement your own convolutional mean filter with the following sizes: 3*3, 5*5, 9*9 pixels (example kernels provided in the lecture).
- (b) Apply each filter on the greyscale images provided on ILIAS.
- (c) Comment on the results obtained with the different filter sizes.

2. Edge detection

- (a) Implement your own Laplacian of Gaussian filter (example provided in the lecture).
- (b) Implement your own gradient filters for edge detection (examples for vertical and horizontal filters provided in the lecture).
- (c) Apply both edge detection methods to the greyscale images provided on ILIAS.
- (d) Comment on the results obtained with the two methods.

Part 3 and Hand-in

3. Statistical filters

- (a) Implement your own minimum and maximum filters (formulas provided in the lecture), with size 3*3 and 5*5 pixels.
- (b) Apply both filters to the greyscale images provided on ILIAS, as well as their combination (max - min).
- (c) Comment on the results obtained with the individual filters and their combination.

Hand-in

- Submit on ILIAS nine and only nine files:
  - 3 images after using a 3*3, 5*5 and 9*9 pixels mean filter,
  - 2 images after using a gradient filter and a Laplacian of Gaussian filter.
  - 3 images after using a min, max and a (max - min) filter.
  - A text file with your name, surname, the link to your GitHub, explain how you get the same input and output size, the link to your GitHub, a short description of you gradient filter and the Laplacian of Gaussian filter, and the responses to the questions 1. c), 2. d) et 3. c).

Questions?