


浙江大学爱丁堡大学联合学院

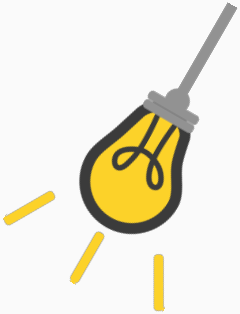
ZJU-UoE Institute

Filters

Nicola Romanò - nicola.romano@ed.ac.uk

Learning objectives

- Define convolutional filters
- Explain their use in image analysis
- Implement basic filters in Python



Types of pixel operations

Operations for manipulating pixel intensities

Two types of operations:

- Point operations - Change pixel intensity based only on its value - $I'_{(x,y)} = f(I_{x,y})$ (see Lecture 3)
- **Neighborhood operations** - Change pixel intensity based on the intensity of the pixel and its neighbours.

Filters

Neighborhood operations, often called **filters** allow to modify an image in a way that is not possible with point operations.

- Detect simple structures such as edges, corners, lines, etc.
- Perform operations such as smoothing, sharpening, etc.
- Noise reduction

Filters

Neighborhood operations, often called **filters** allow to modify an image in a way that is not possible with point operations.

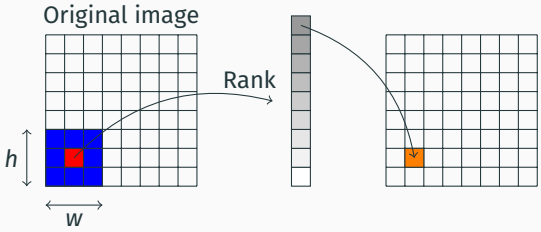
- Detect simple structures such as edges, corners, lines, etc.
- Perform operations such as smoothing, sharpening, etc.
- Noise reduction

Today we will look at:

- **Rank filters** - the new pixel value is a function of the rank of the pixel values of the neighborhood
- **Convolutional filters** - the new pixel value is a weighted sum of the pixel values of the neighborhood

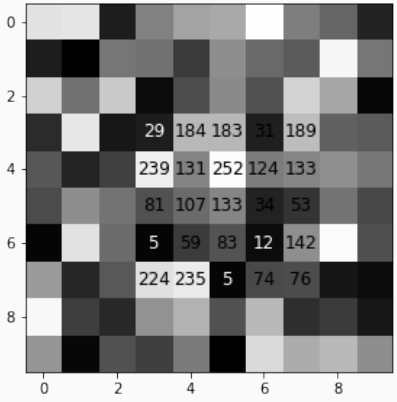
Rank filters

Rank filters



- We decide on a window size $w \times h$ (other non-rectangular shapes are possible)
- We traverse each pixel in the image and take its $w \times h$ neighborhood
- We rank the intensity of each pixel in the neighborhood
- We take a specific value (e.g. minimum, maximum, mean, median) and set the output value to this value

Example - median filter



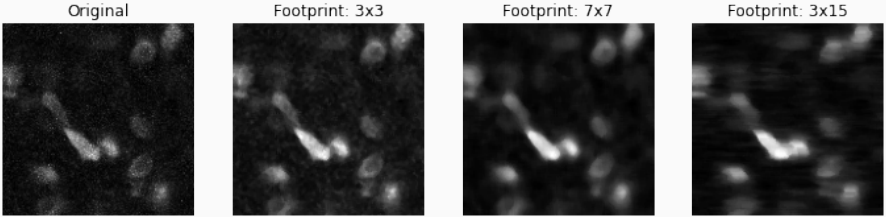
Rank filters in Scikit Image

Rank filters are implemented in Scikit Image in the ‘skimage.rank‘ module.
These filters require a *footprint* of the pixel neighborhood, as a matrix of 0 and 1.

For example

```
 from skimage.rank import median

# 3x3 neighborhood
footprint = np.ones(3, 3)
img_median = median(img, selem=footprint)
```



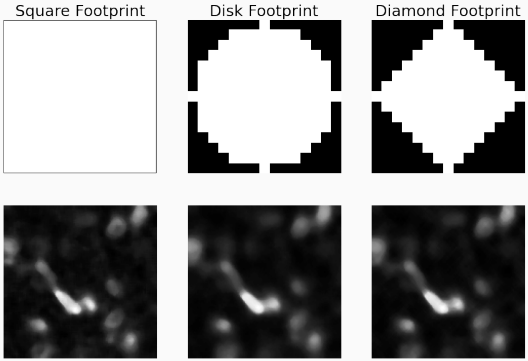
Non-rectangular footprints

The footprint can be a *non-rectangular* matrix. For example, you can generate a footprint with a **circular** shape using the `skimage.morphology.disk` function or one with a **diamond** shape using the `skimage.morphology.diamond` function.

```
from skimage.morphology import disk,
diamond

dsk = disk(7) # A disk, radius 7
dia = diamond(7) # A diamond, radius 7

fig, ax = plt.subplots(1, 2)
ax[0].imshow(dsk, cmap="gray")
ax[1].imshow(dia, cmap="gray")
```



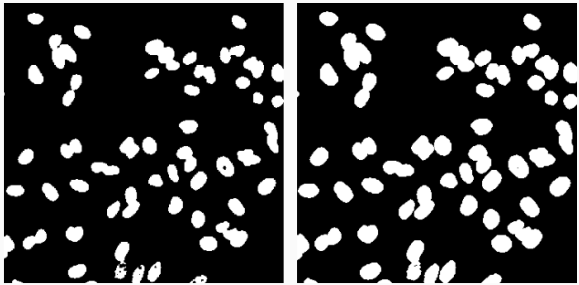
Rank filters - use cases

Rank filters are very simple, but have useful applications.
The **median and mean filter** are used to remove noise and to smooth images.

Rank filters - use cases

Rank filters are very simple, but have useful applications.
The **median and mean filter** are used to remove noise and to smooth images.

The **maximum filter** can be used in binary images to remove small "holes". It is also a very common filter used in modern neural networks for image analysis.



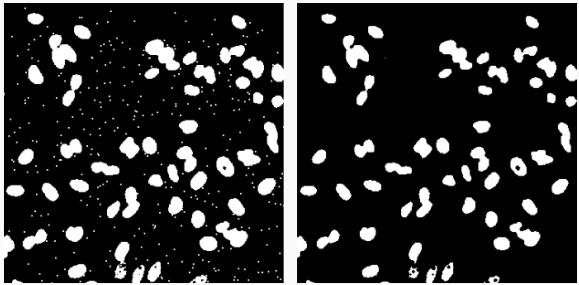
The maximum filter can remove small holes in binary masks

Rank filters - use cases

Rank filters are very simple, but have useful applications.
The **median and mean filter** are used to remove noise and to smooth images.

The **maximum filter** can be used in binary images to remove small "holes". It is also a very common filter used in modern neural networks for image analysis.

The **minimum filter** can be used to remove small bright spots.



The minimum filter can remove small bright spots in binary masks