Image Enhancement in Spatial Domain(Neighbourhood Processing)

Dr. Su Su Maung
CEIT Department
Yangon Technological University

Outline of Lecture3

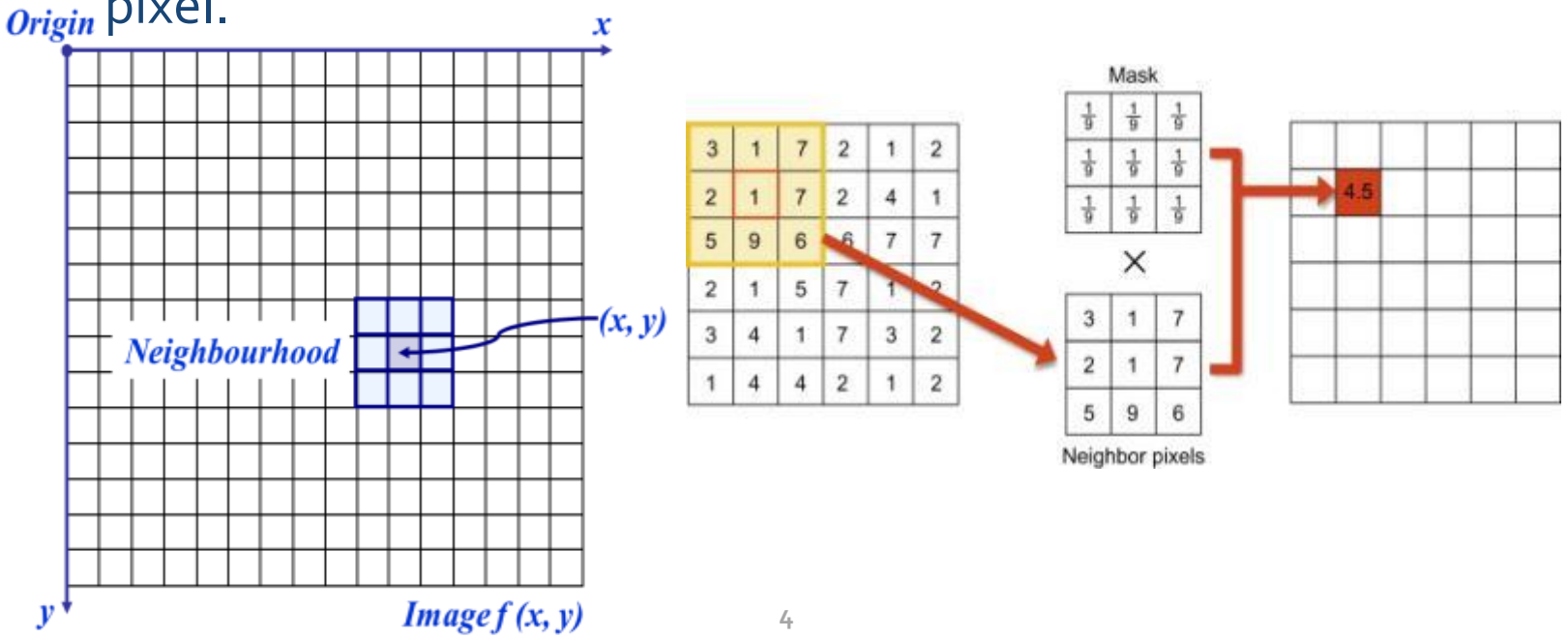
- Introduction
- Neighbourhood operations
- Linear filter
- Frequencies; low and high pass filters
- Nonlinear filters

Introduction

- In point processing, an image can be modified by applying a particular function to each pixel value.
- Neighbourhood processing , a function is applied to a neighbourhood of each pixel.
- The idea is to move a "mask": a rectangle (sides of odd length) or other shape over the given image. By creating a new image whose pixels have grey values calculated from the grey values under the mask.
- The combination of mask and function is called a filter.
- If the function is a linear function of all the grey values in the mask, then the filter is called a linear filter.

Neighbourhood operations

- Neighbourhood operations simply operate on a larger neighbourhood of pixels
- Neighbourhoods are mostly a rectangle around a central pixel.

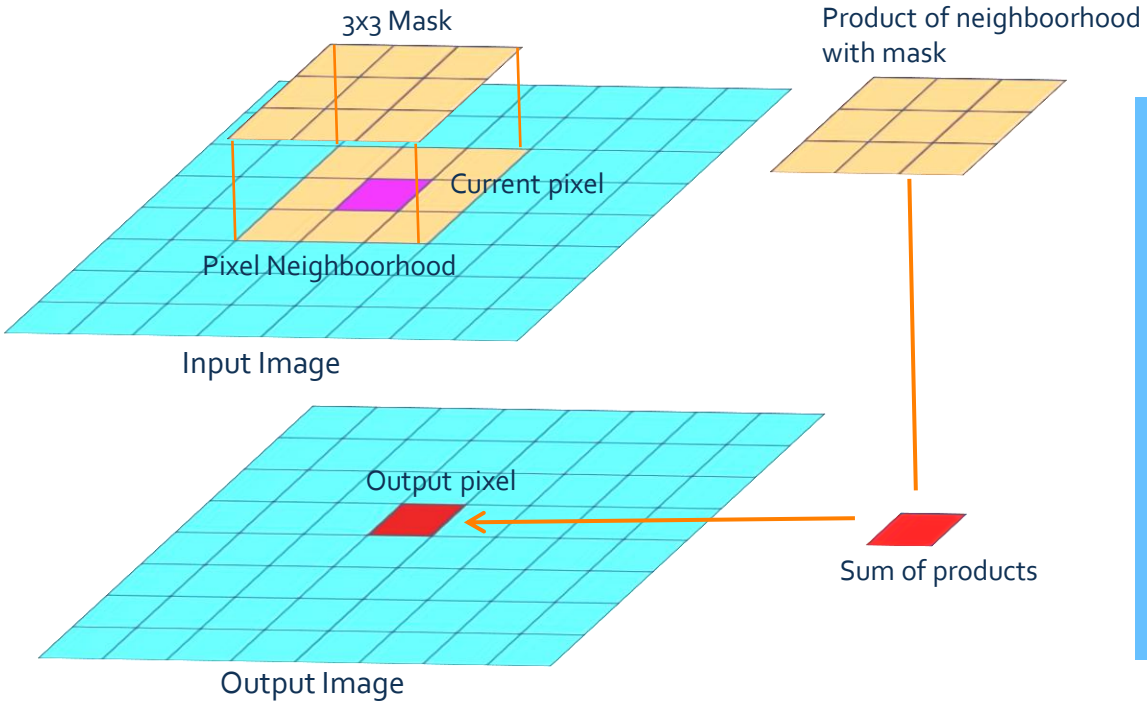


Filter Parameters

- Filter parameters (size, weights, function, etc.)
- Filter size (size of neighbourhood):
  - 3x3, 5x5, 7x7, …,15x15
- Filter shape
  - not necessarily square. Can be rectangle, circle, etc.
- Filter weights
  - may apply unequal weighting to different pixels
- Filters function:
  - can be linear (a weighted summation) or nonlinear

Linear filter

- A linear filter can be implemented by multiplying all elements in the mask by corresponding elements in the neighbourhood, and adding up all these products.



Spatial filtering requires three steps:
1. position the mask over the current pixel,
2. form all products of filter elements with the corresponding elements of the neighbourhood,
3. add up all the products.
This must be repeated for every pixel in the image.

Performing spatial filtering

Example of linear filter

- Linear filter takes the average of all nine values within the mask. This value becomes the grey value of the corresponding pixel in the new image. This operation may be described as follows:

a	b	c
d	e	f
g	h	i

$$\frac{1}{9}(a+b+c+d+e+f+g+h+i)$$

- where e is grey value of the current pixel in the original image, and the average is the grey value of the corresponding pixel in the new image .
- A linear filter can be described simply in terms of the coefficients of all the grey values of pixels within the mask. This can be written as a matrix.

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Example: What does this Filter Do?

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Mean (averages neighbourhood)

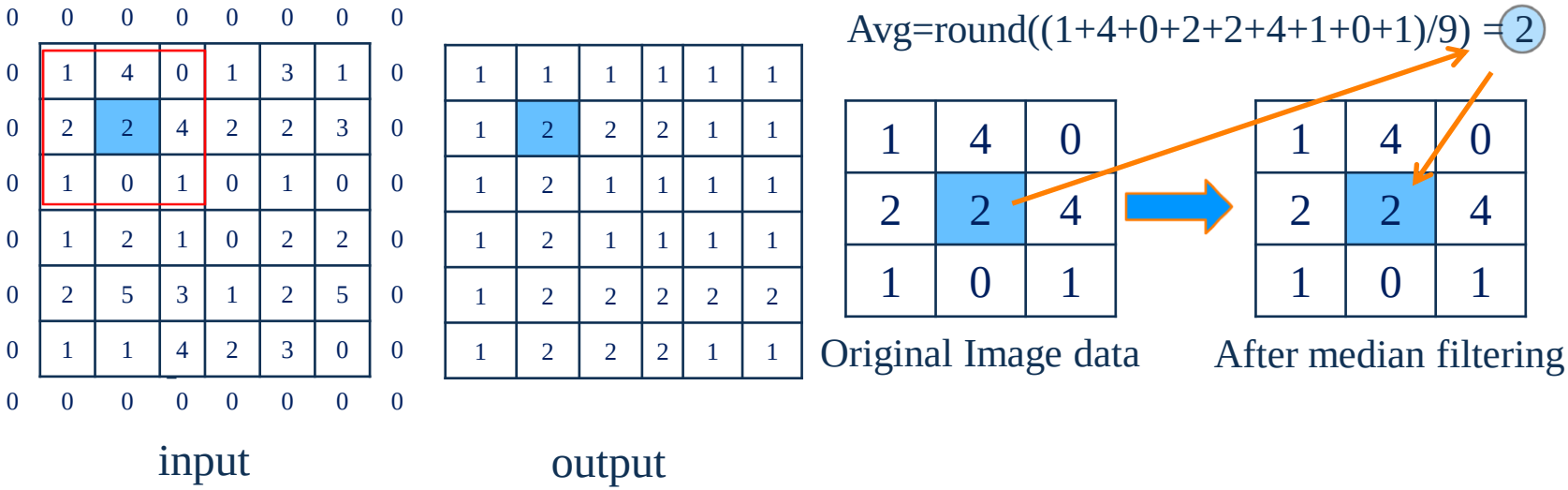
Original

0	0	0
0	1	0
0	0	0

Filtered
(no change)

Identity filter

Averaging filter



2D Averaging filtering example using a 3x3 filter

Filtering in Python

- Linear filtering can be performed by using convolve or correlate from the ndimage module of the library, or generic_filter.
- The mode parameter 'constant' tells the function that the image is to be padded with constant values, the default value of which is zero

```
#apply 9 × 9 averaging to the cameraman image

import scipy.ndimage as ndi
import matplotlib.pyplot as plt
import skimage.io as io
from skimage import data
import numpy as np

img=data.camera()
img1=ndi.convolve(img,np.ones((9,9))/81,mode='
constant')
io.imshow(img1)
```

Cameraman image and image by average filtering using 9x9 filter

Filtering in Python (cont.)

- Alternately, you could use the built in uniform_filter function:

```
img1=ndi.uniform_filter(img,25)
```

- The averaging filter blurs the image; the edges in particular are less distinct than in the original. The image can be further blurred by using an averaging filter of larger size.

Average filtering using 25x25 filter

Separable filters

- the averaging filter can be implemented by first applying a 3x1 averaging filter, and then applying a 1x3 averaging filter to the result. The 3x3 averaging filter is thus separable into two smaller filters.
- Separability can result in great time savings.
- The application of an nx1 filter only requires n multiplications and n-1 additions. Since this must be done twice, the total number of multiplications and additions are 2n and 2n-2 respectively. If n is large the savings in efficiency can be dramatic.
- All averaging filters are separable; another separable filter is the laplacian .

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

Frequencies; low and high pass filters

- The frequencies of an image are a measure of the amount by which grey values change with distance.
- High frequency components are characterized by large changes in grey values over small distances. E.g. edges and noise.
- Low frequency components, on the other hand, are parts of the image characterized by little change in the grey values. E.g. backgrounds, skin textures.
- high pass filter : if it "passes over" the high frequency components, and reduces or eliminates low frequency components,

Frequencies; low and high pass filters

- low pass filter : if it "passes over" the low frequency components, and reduces or eliminates high frequency components.
- For example, the averaging filter is low pass filter, as it tends to blur edges. The following filter is a high pass filter.

$$\begin{bmatrix} 1 & -2 & 1 \\ 2 & 4 & 2 \\ 1 & -2 & 1 \end{bmatrix}$$

- The sum of the coefficients (that is, the sum of all e elements in the matrix), in the high pass filter is zero.
- High pass filters are of particular value in edge detection and edge enhancement .

High pass filters

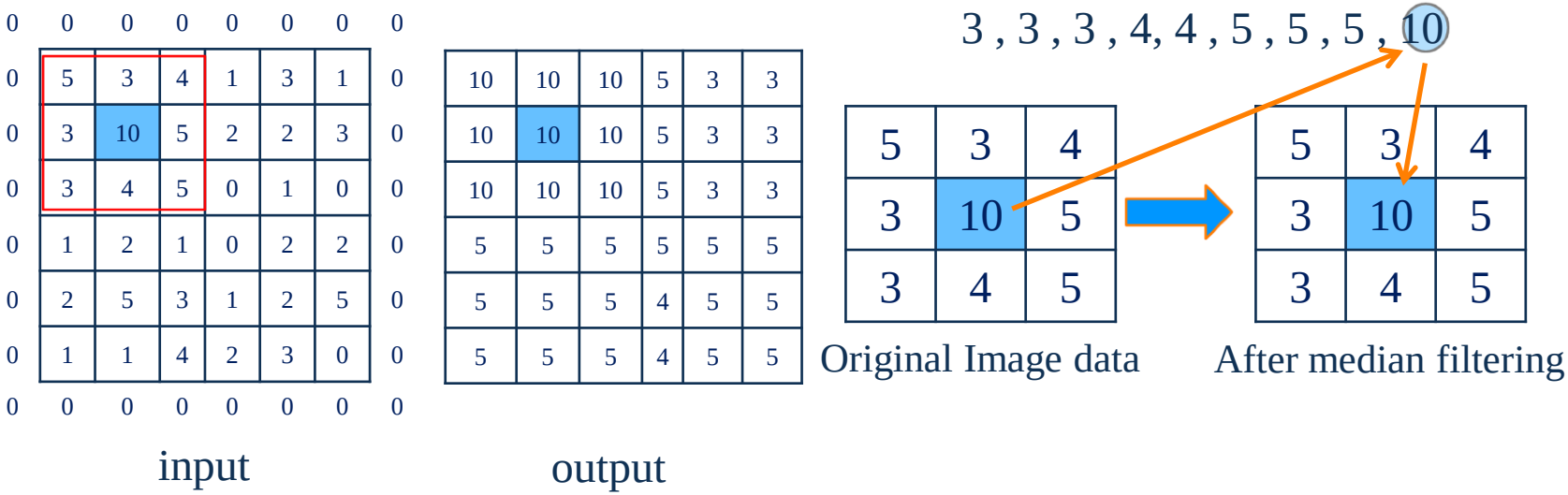
High pass filtering

```
f=np.array([[1, 4, 1],[4, -20, 4],[1, 4, 1]])
img1=ndi.convolve(img,f)
plt.figure()
io.imshow(img1)
```

Non-linear filters

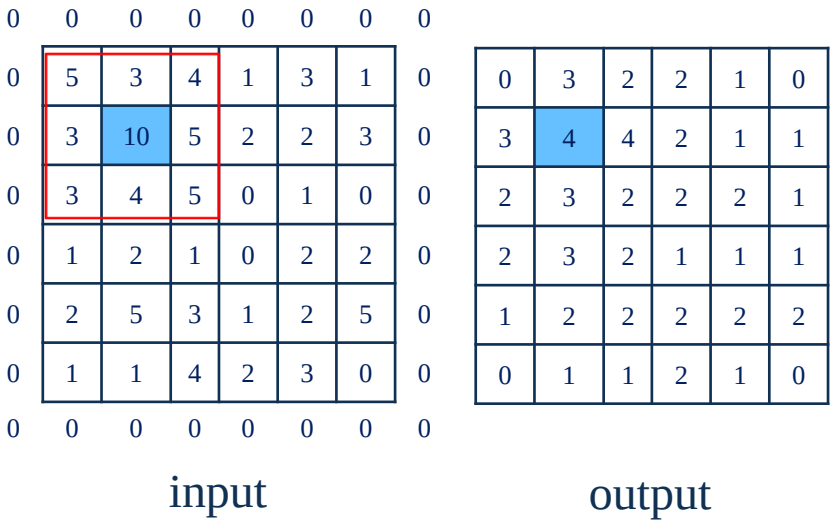
- A non-linear filter is obtained by a non-linear function of the greyscale values in the mask.
- Simple examples are the maximum filter, which has as its output the maximum value under the mask, the corresponding minimum filter, which has as its output the minimum value under the mask, and median filter which takes the central value of the ordered list.
- Both the maximum and minimum filters are examples of rank-order filters. In such a filter, the elements under the mask are ordered, and a particular value returned as output.
- So if the values are given in increasing order, the minimum filter is a rank-order filter for which the first element is returned, and the maximum filter is a rank-order filter for which the last element is returned

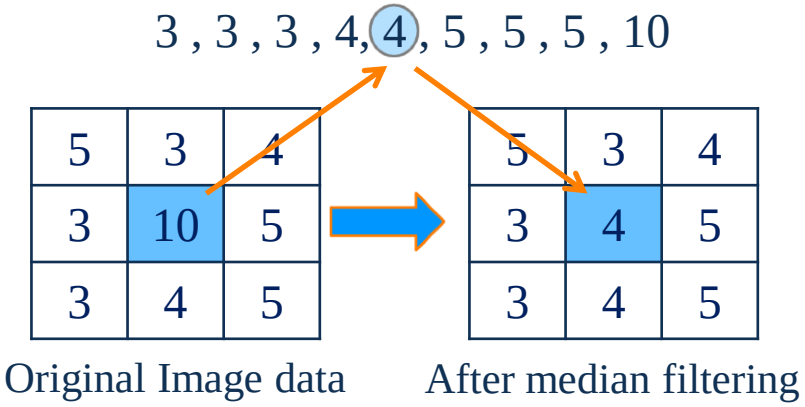
Maximum filter



2D Maxmum filtering example using a 3x3 filter

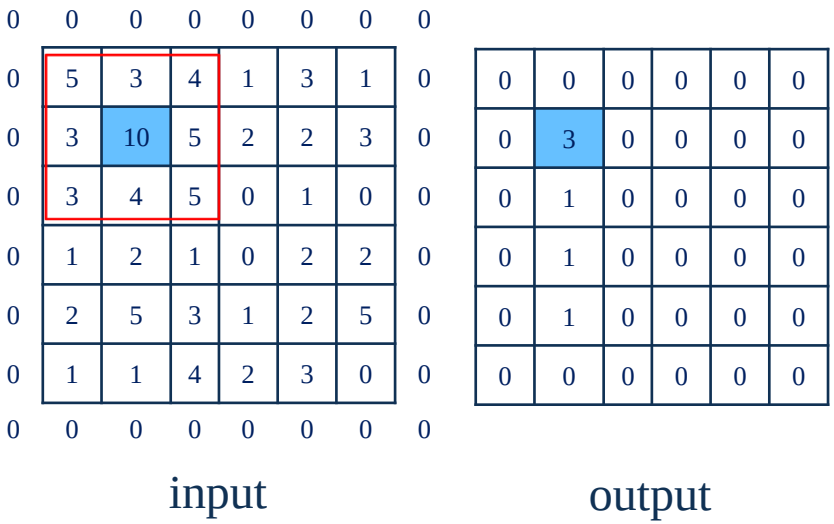
Medium filter

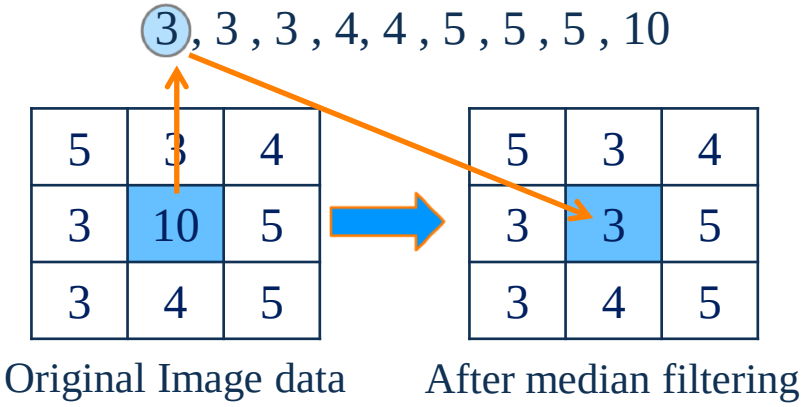




2D Median filtering example using a 3x3 filter

Minimum filter





2D Minimum filtering example using a 3x3 filter

Non-linear filters

- For implementing a general non-linear filter in Python, the function to use is generic_filter, which applies a filter to an image according to a pre-defined function.
- Here are some examples; first to implement a maximum filter over a 3x3 neighbourhood:

```
cmax=ndi.generic_filter(c,max,[3,3])
```

- The generic_filter function requires three arguments: the image matrix, the size of the filter, and the function to be applied. The function must be a matrix function which returns a scalar value.
- A corresponding implementation of the minimum filter is:

```
cmin=ndi.generic_filter(c,min,[3,3])
```

Non-linear filters

- Python has maximum,median and minimum filters in the scipy.ndimage module and also in the skimage.filter.rank module. As an example of each:

```
cmax=ndi.maximum_filter(c,size=(3,3))
cmin=ndi.minimum_filter(c,size=(3,3))
cmin=ndi.median_filter(c,size=(3,3))
```

```
cmax=rk.maximum(c,ones((3,3)))
cmin=rk.minimum(c,ones((3,3)))
cmin=rk.median(c,ones((3,3)))
```

Non-linear filters

- Note that in each case the image has lost some sharpness, and has been brightened by the maximum filter, and darkened by the minimum filter.

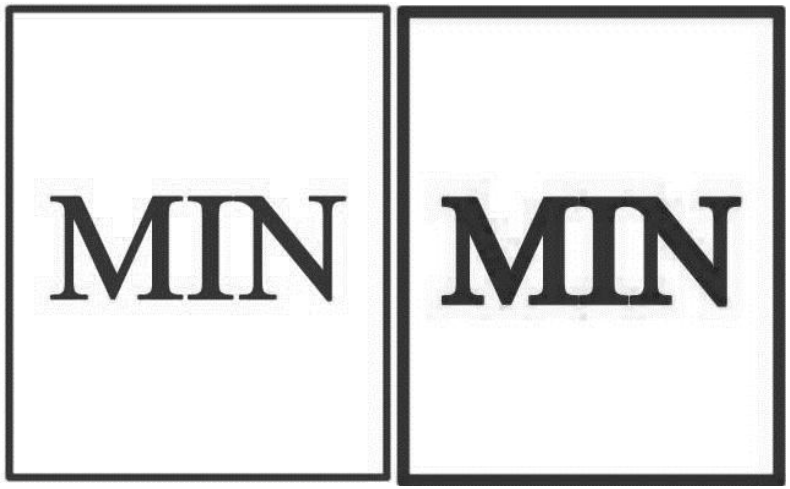
Original image

Using a maximum filter

Using a minimum filter

Using non-linear filters

Non-linear filters (cont.)



Original Image and after minimum Filtering

```
import scipy.ndimage as ndi
import skimage.io as io
from skimage.color import rgb2gray
img=rgb2gray(io.imread('images/min.jpg'))
io.imsave('mingrey.jpg',img)
minimg=ndi.minimum_filter(img,5)
io.imsave('minimg.jpg',minimg)
```

Original Image and after maximum Filtering

```
import scipy.ndimage as ndi
import skimage.io as io
from skimage.color import rgb2gray
img=rgb2gray(io.imread('images/max.jpg'))
io.imsave('maxgrey.jpg',img)
maximg=ndi.maximum_filter(img,5)
io.imsave('maximg.jpg',maximg)
```

Next Week Lecture (Week4)

- Lecture4:Image Enhancement in Frequency Domain

Thank You