Manara university
Engineering college

Faculty of Informatics engineering

Computer vision

2025 - 2026

Lab session 6:
Min – Max- Median filters

Tasks:

1. Read the image in the gray scale.
2. Define a function called sort, that takes a [3 by 3] window , and return a sorted vector created from the same window .
3. Define 3 zeros arrays (Minimum, Maximum and Median)as the same size of the original image .
4. Scan the original gray image using a [3 by 3] empty window, and find the values (min , max and med) for all windows cropped from the original image starting from the upper left corner of the image .
5. Fill in the previous arrays (Minimum, Maximum and Median) with the values (min, max and med) starting from the 2nd rows and 2nd columns .
6. Display the results.

Code:

```
import cv2 as cv

import numpy as np

import matplotlib.pyplot as plt



def sort(window):

    flat_list = []

    for row in window:

        flat_list.extend(row)


    # Bubble algorithm

    n = len(flat_list)  # 9
```

```python
    for i in range(n - 1):
        for j in range(n - 1 - i):
            if flat_list[j] > flat_list[j + 1]:
                flat_list[j], flat_list[j + 1] = flat_list[j + 1], flat_list[j]

    return flat_list


path = r" \\"

img = cv.imread(path+'noisy.jpg', 0)
height, width = img.shape
minimum_array = np.zeros((height, width))
maximum_array = np.zeros((height, width))
median_array = np.zeros((height, width))

for i in range(1, height - 1):
    for j in range(1, width - 1):
        window = img[i-1:i+2, j-1:j+2]
        sorted_window = sort(window)
        min =  sorted_window[0]
        max =  sorted_window[8]
        med =  sorted_window[4]
        minimum_array[i, j] = min
        maximum_array[i, j] = max
        median_array[i, j] = med
```

```
fig, axes = plt.subplots(2, 2)

axes[0][0].imshow(img, cmap='gray')

axes[0][1]. imshow(minimum_array, cmap='gray')

axes[1][0].imshow(maximum_array, cmap='gray')

axes[1][1].imshow(median_array, cmap='gray')

plt.show()
```

Results :