N20DCCN031

Nguyễn Nhật Kha

D20CQCNPM01-N

**Homework 6**

# MSSV: N20DCCN031

# Nguyễn Nhật Kha

# Lớp: D20DCCNPM01-N

import numpy as np

import os

from matplotlib import pyplot as plt

def get\_file(name):

    current\_file\_path = os.path.abspath(\_\_file\_\_)

    parent\_directory = os.path.dirname(current\_file\_path)

    return np.fromfile(f"{parent\_directory}/{name}", dtype=np.uint8)

def show\_group\_result(image\_title, original, median, opened, closed):

    plt.figure(figsize=(10,8))

    plt.subplot(2,2,1)

    plt.title(image\_title)

    plt.imshow(original, cmap="gray")

    plt.axis("off")

    plt.subplot(2,2,2)

    plt.title("Median Filter")

    plt.imshow(median, cmap="gray")

    plt.axis("off")

    plt.subplot(2,2,3)

    plt.title("Morphological Opening")

    plt.imshow(opened, cmap="gray")

    plt.axis("off")

    plt.subplot(2,2,4)

    plt.title("Morphological Closing")

    plt.imshow(closed, cmap="gray")

    plt.axis("off")

    plt.show()

def compute(image):

    wsize = 3

    wsizeo2 = wsize // 2

    filter\_window = np.zeros((3, 3))

    size = 256

    median\_img = np.zeros((size, size))

    eroded\_img = np.zeros((size, size))

    dilated\_img = np.zeros((size, size))

    opened\_img = np.zeros((size, size))

    closed\_img = np.zeros((size, size))

    # Apply median, erode, and dilate

    for row in range(wsizeo2, size - wsizeo2):

        for col in range(wsizeo2, size - wsizeo2):

            filter\_window = image[row - wsizeo2:row + wsizeo2 + 1, col - wsizeo2:col + wsizeo2 + 1]

            median\_img[row, col] = np.median(filter\_window)

            eroded\_img[row, col] = np.min(filter\_window)

            dilated\_img[row, col] = np.max(filter\_window)

    # Apply dilate to eroded\_img and erode to dilated\_img

    for row in range(wsizeo2 + 1, size - wsizeo2 - 1):

        for col in range(wsizeo2 + 1, size - wsizeo2 - 1):

            filter\_window = eroded\_img[row - wsizeo2:row + wsizeo2 + 1, col - wsizeo2:col + wsizeo2 + 1]

            opened\_img[row, col] = np.max(filter\_window)

            filter\_window = dilated\_img[row - wsizeo2:row + wsizeo2 + 1, col - wsizeo2:col + wsizeo2 + 1]

            closed\_img[row, col] = np.min(filter\_window)

    return [median\_img, opened\_img, closed\_img]

camera9 = get\_file("camera9.bin").reshape(256,256)

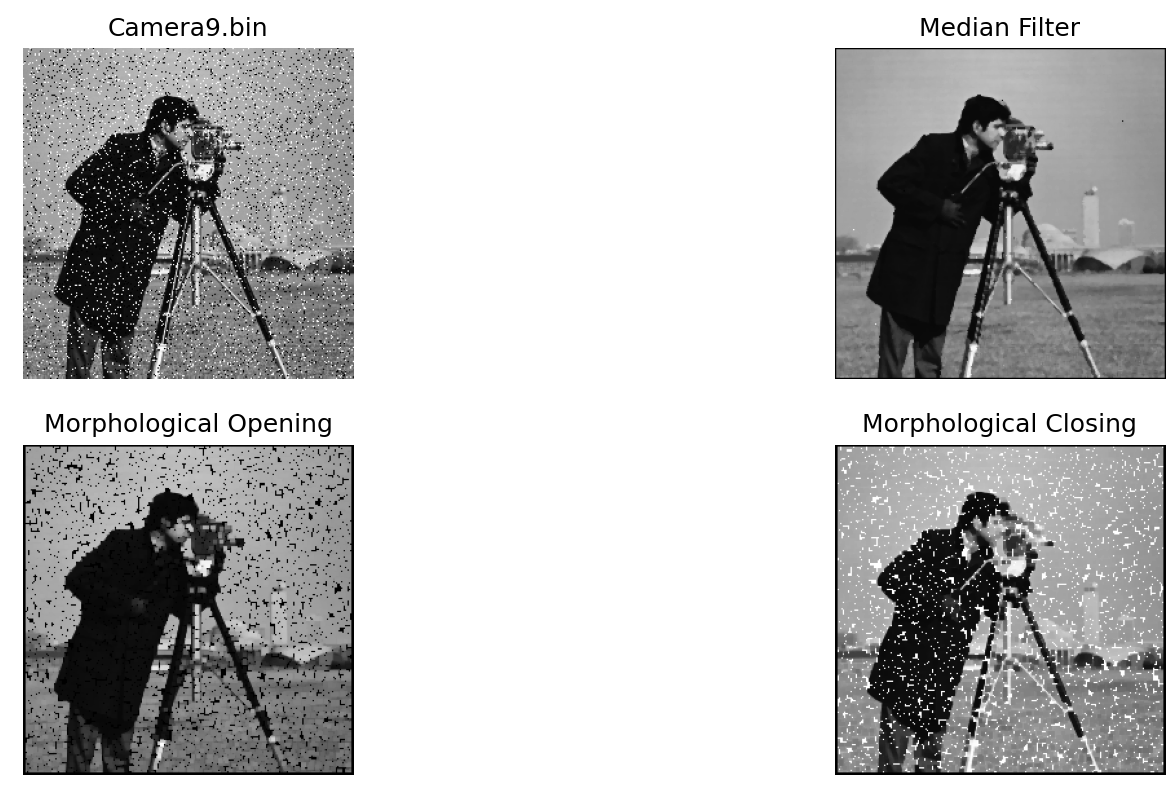
camera99 = get\_file("camera99.bin").reshape(256,256)

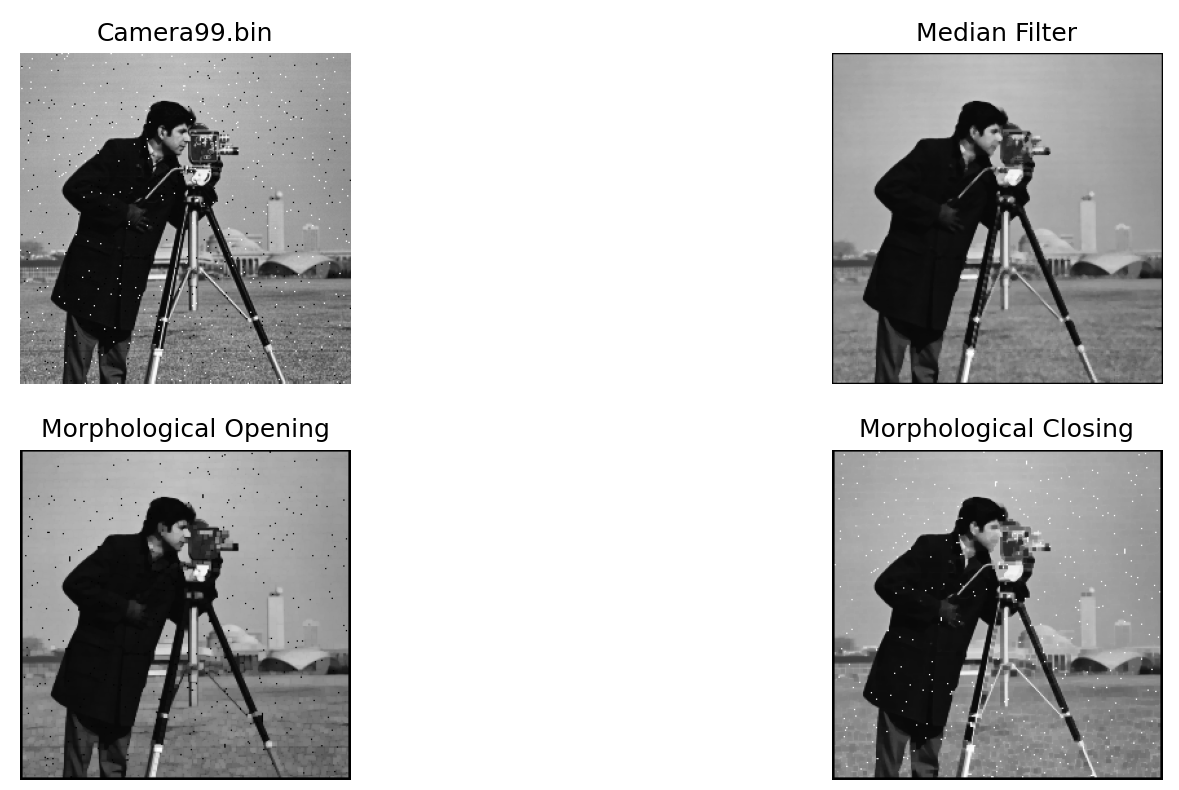
[median9,opened9,closed9] = compute(camera9)

show\_group\_result("Camera9.bin", camera9, median9, opened9, closed9)

[median99,opened99,closed99] = compute(camera99)

show\_group\_result("Camera99.bin", camera99, median99, opened99, closed99)

****

****

Qua cả 2 trường hợp camera9.bin và camera99.bin, tuy mỗi hình ảnh chịu mức độ nhiễu muối tiêu (salt and peper noise) là khác nhau (camera9 nhiễu nhiều hơn camera99), nhưng với Median Filter, các nhiễu này đều được lọc đi rất tốt. Với Morphological Opening, các nhiễu sáng (có giá trị 255) được lọc đi trong khi đó các nhiễu tối (có giá trị 0) thì vẫn còn giữ, ngược lại với Morphological Closing, các nhiễu tối được lọc đi trong khi vẫn còn các nhiễu sáng.

Median Filter hoạt động tốt đối với cả nhiễu sáng và tối vì nó giảm nhiễu bằng cách lấy giá trị trung vị của các pixel lân cận

Morphological Opening thì giúp loại bỏ các nhiễu sáng do nó thực thiện erode – co (lấy giá trị min) các pixel lân cận trước rồi sau đó mới thực hiện dilate – giãn (lấy giá trị max)

Ngược lại, Morphological Closing thì giúp loại bỏ các nhiễu tối do nó thực hiện dilate trước và erode sau