МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
кафедра систем штучного інтелекту

Лабораторна робота №1

на тему:

«Попередня обробка зображень»

з дисципліни

«Обробка зображень методами машинного навчання»

Виконав:
ст. групи КН-408
Романьчук Максим

ЛЬВІВ – 2022

Мета - вивчити просторову фільтрацію зображень, методи мінімізації шуму, морфології, виділення країв і границь та елементи бібліотеки OpenCV для розв’язання цих завдань.

Теоретичні відомості

Фільтри поділяються на **лінійні** та **нелінійні**. Перевагою перших є асоціативність, комутативність та легша (обчислювально теж) реалізація. Останні ж, призначені для фільтрації специфічних шумів, що може суттєво покращити результати в певних задачах комп’ютерного зору.

До лінійних відносяться:

- Одновимірна лінійна фільтрація
- Двовимірна лінійна фільтрація
- Box фільтрація

До нелінійних:

- Фільтр Гауса
- Медіанний фільтр
- Вирівнювання гістограм

Хід роботи

Варіант 13. Виконати детекцію границь на зображеннях за допомогою операторів Sobel, Prewitt. Провести порівняльний аналіз.

1) Для цієї роботи було обрано 2 пари фото з різними контрастністю та деталізацією.

Рис. 1-2 Пара зображень з низькою (зліва) та високою (справа) деталізаціями.

Рис. 3-4 Пара зображень з низькою (зліва) та високою (справа) контрастністю.

2) Наступним кроком було написано код накладання фільтра на фото, приведене до відтінків сірого.

Код програми:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import image as img

# Sobel edge operator
kernel_Sobel_x = np.array([
    [-0.5, 0, 0.5],
    [-1, 0, 1],
```

```
    [-0.5, 0, 0.5]
])
kernel_Sobel_y = np.array([
    [0.5, 1, 0.5],
    [0, 0, 0],
    [-0.5, -1, -0.5]
])

# Prewitt edge operator
kernel_Prewitt_x = np.array([
    [-1, 0, 1],
    [-1, 0, 1],
    [-1, 0, 1]
])
kernel_Prewitt_y = np.array([
    [1, 1, 1],
    [0, 0, 0],
    [-1, -1, -1]
])

# Plot few images in a row for comparison
def plot_images(images, labels) :
    if(len(images) != len(labels)):
        raise RuntimeError(f'Cannot assign {len(labels)} labels to {len(images)}
images!')

    fig, axes = plt.subplots(1, len(images))
    for idx, ax in enumerate(axes) :
        ax.imshow(images[idx], cmap=plt.get_cmap('gray'))
        ax.set_title(labels[idx], fontsize=12)
        ax.axis('off')
    plt.show()

# Convert RGB image to grayscale
def rgb2gray(rgb):
    return np.dot(rgb[...,:3], [0.2989, 0.5870, 0.1140])

# Warning appliable only for 3x3 filters
def apply_filter(image: np.ndarray, kernel_x: np.ndarray, kernel_y: np.ndarray) :
    if(len(image.shape) != 2 or len(kernel_x.shape) != 2 or len(kernel_y.shape) != 2):
        raise RuntimeError('Image and kernels should have 2 dimentions')
    result = image.copy()

    # Add padding for image to avoid index out of bounds
    image = np.vstack((image[-1], image, image[0]))
    image = np.hstack((image[:,-1][:, np.newaxis], image, image[:,0][:, np.newaxis]))

    # Iterate through image
    for i in range(1, image.shape[0] - 2) :
        for j in range(1, image.shape[1] - 2) :
            result[i, j] = np.sqrt(
                np.sum(kernel_x * image[i-1:i+2, j-1:j+2])**2 +
                np.sum(kernel_y * image[i-1:i+2, j-1:j+2])**2
```

```
            )
    return result

# Read images
images = [
    rgb2gray(img.imread('hand_compressed.jpg')),
    rgb2gray(img.imread('shepherd_compressed.jpg')),
    rgb2gray(img.imread('high_contrast.jpg')),
    rgb2gray(img.imread('low_contrast.jpg'))
]

labels = [
    'Original image (grayscale)',
    'After Sobel filtration',
    'After Prewitt filtration'
]

for img in images:
    imgs = [
        img,
        apply_filter(img, kernel_Sobel_x, kernel_Sobel_y),
        apply_filter(img, kernel_Prewitt_x, kernel_Prewitt_y),
    ]
    plot_images(imgs, labels)
```

3) Опісля було виконано програму та здійснено пошук відмінностей результатів накладання фільтрів (рис. 5-8).

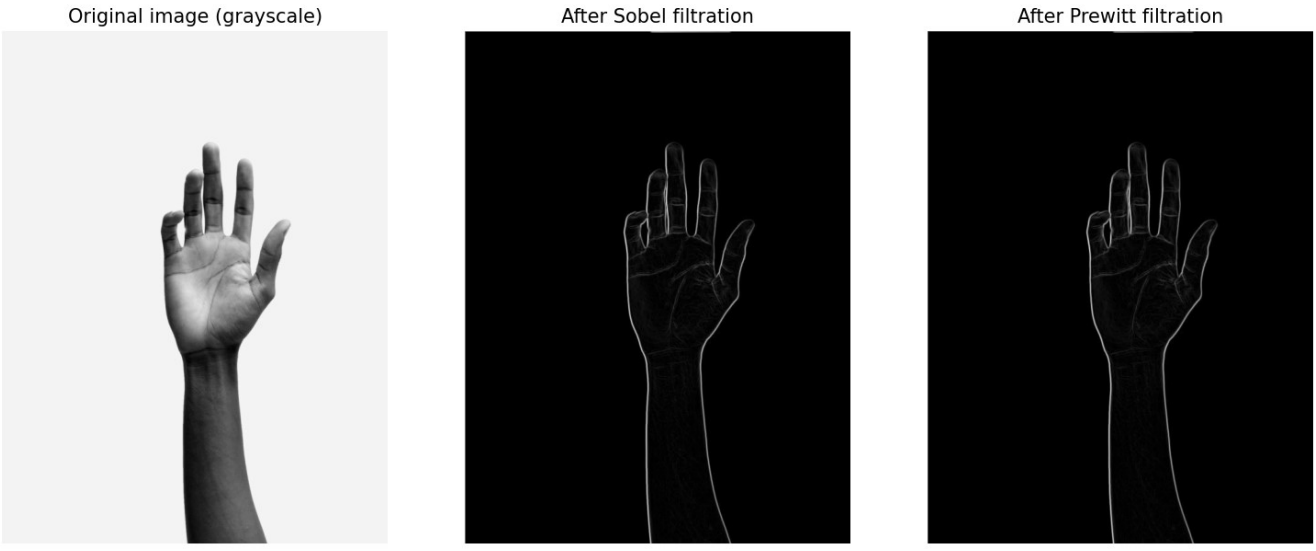

Рис. 5 Результат визначення границь зображення з низькою деталізацією.

Рис. 6 Результат визначення границь зображення з високою деталізацією.

Рис. 7 Результат визначення границь зображення з високою контрастністю.

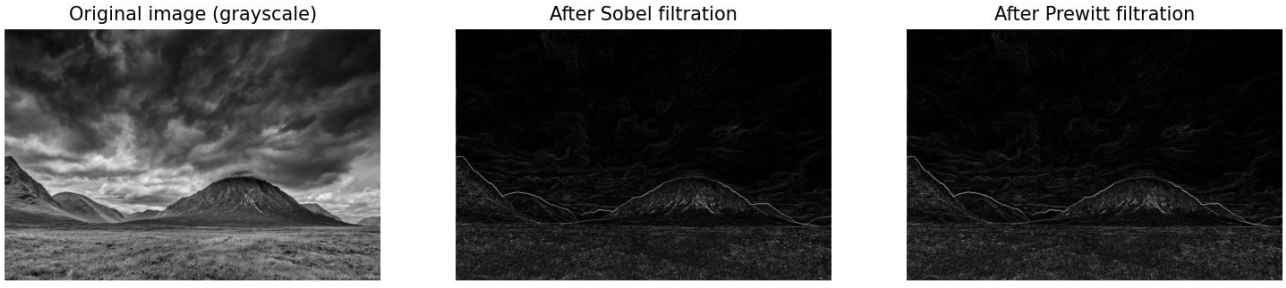

Рис. 8 Результат визначення границь зображення з низькою контрастністю.

З даних зображень складно зробити висновок щодо ефективності даних операторів та їх переваг. Втім, за рахунок помітної різниці між ядрами фільтрів, а саме, збільшеної ваги оператора Sobel по середніх точках можна зробити висновок, що при масштабуванні він дозволяє детальніше описувати границі, що

може бути як перевагою при пошукує незначних деталей на фото, так і недоліком при роботі з незвичайними шумами/дефектами/артефактами.

Висновок

При виконанні цієї лабораторної роботи було досліджено принципи накладання фільтрів на зображення та визначення границь об’єктів на ньому на прикладі операторів Sobel та Prewitt. Також було здійснено спробу знайти суттєві відмінності між операторами на парах зображень з суттєво різними контрастністю та деталізованістю, але значних відмінностей знайдено не було, тому було висунуто припущення, згідно з відмінностями ядер фільтрів кожного з операторів.