**Міністерство освіти і науки України**

**Львівський національний університет імені Івана Франка**

**Факультет електроніки та комп’ютерних технологій**

**ЗВІТ**

**З лабораторної роботи № 2**

На тему «Градаційні перетворення»

**Виконала:**

Студентка 3 курсу

групи ФеП-32

Галабурда Є.

**Перевірив:**

Половинко І.І.

**Львів – 2025**

**Мета:** дослідження та застосування методів градаційних перетворень для поліпшення якості зображень.

**Завдання:** 1. Ознайомитись з основними способами градаційних перетворень.

2. Використовуючи представлений фрагмент коду перетворення зображення в негатив, запустити програму для градаційного перетворення.

3. Скласти аналогічну програму для градаційних перетворень описаних у пунктах 1-3.

4. Використовуючи оригінальне зображення, здійснити його зашумлення, після цього провести градаційне перетворення. Використовуючи формули 6-7, оцінити у дБ ступінь покращення зображення.

5. Проаналізувати для якого типу зображень ці перетворення є найбільш ефективні.

import cv2

import numpy as np

import matplotlib.pyplot as plt

def negative\_transformation(image):

    return 255 - image

def log\_transformation(image, c=1):

    image = np.uint8(image)

    return np.uint8(c \* np.log1p(image))

def power\_law\_transformation(image, gamma, c=1):

    image = np.uint8(image)

    return np.uint8(c \* (image / 255) \*\* gamma \* 255)

def add\_gaussian\_noise(image, mean=0, sigma=25):

    noise = np.random.normal(mean, sigma, image.shape).astype(np.uint8)

    noisy\_image = cv2.add(image, noise)

    return noisy\_image

def mse(original, processed):

    return np.mean((original - processed) \*\* 2)

def psnr(original, processed):

    mse\_value = mse(original, processed)

    if mse\_value == 0:

        return 100

    return 10 \* np.log10((255 \*\* 2) / mse\_value)

def plot\_images(images, titles, cmap='gray'):

    plt.figure(figsize=(15, 5))

    for i in range(len(images)):

        plt.subplot(1, len(images), i + 1)

        plt.imshow(images[i], cmap=cmap)

        plt.title(titles[i], fontsize=10)

        plt.axis('off')

    plt.show()

gray\_image = cv2.imread('image.jpg', cv2.IMREAD\_GRAYSCALE)

if gray\_image is None:

    print("Помилка: зображення не завантажено. Перевірте шлях до файлу.")

else:

    negative\_image = negative\_transformation(gray\_image)

    log\_image = log\_transformation(gray\_image, c=45)

    power\_image = power\_law\_transformation(gray\_image, gamma=0.5)

    noisy\_image = add\_gaussian\_noise(gray\_image)

    denoised\_image = log\_transformation(noisy\_image, c=45)

    psnr\_value = psnr(gray\_image, denoised\_image)

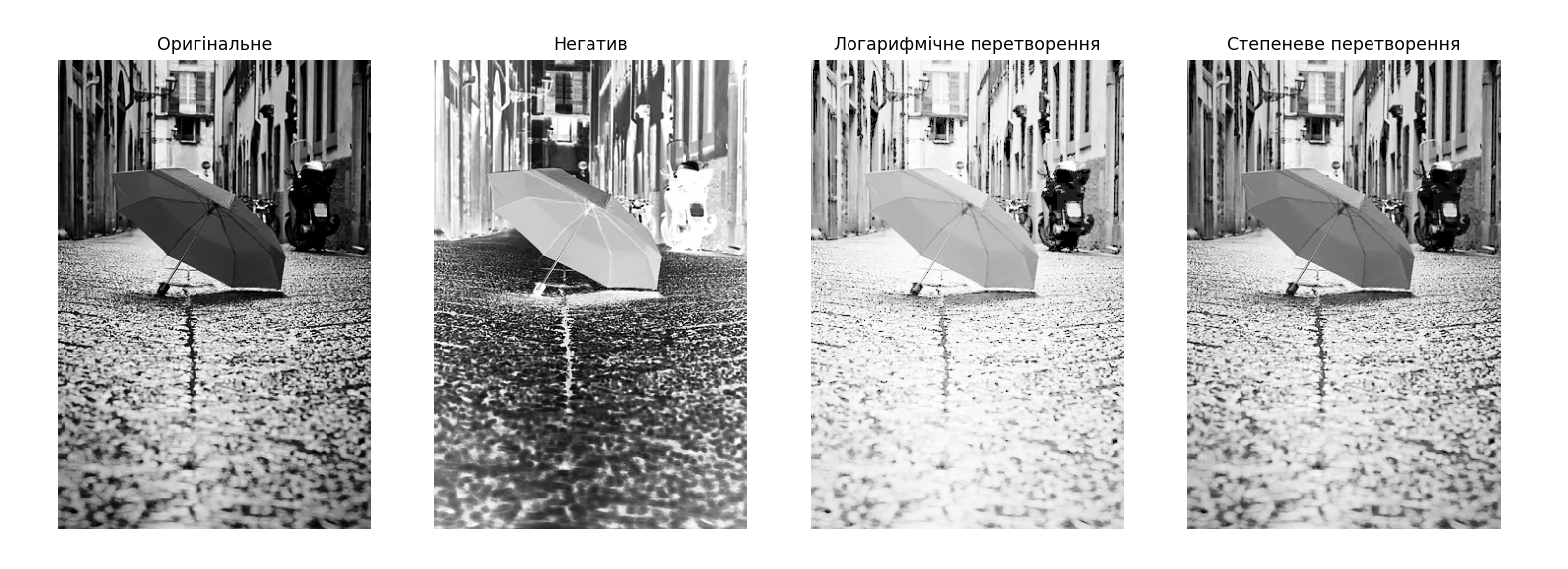
    print(f'PSNR після обробки: {psnr\_value:.2f} дБ')

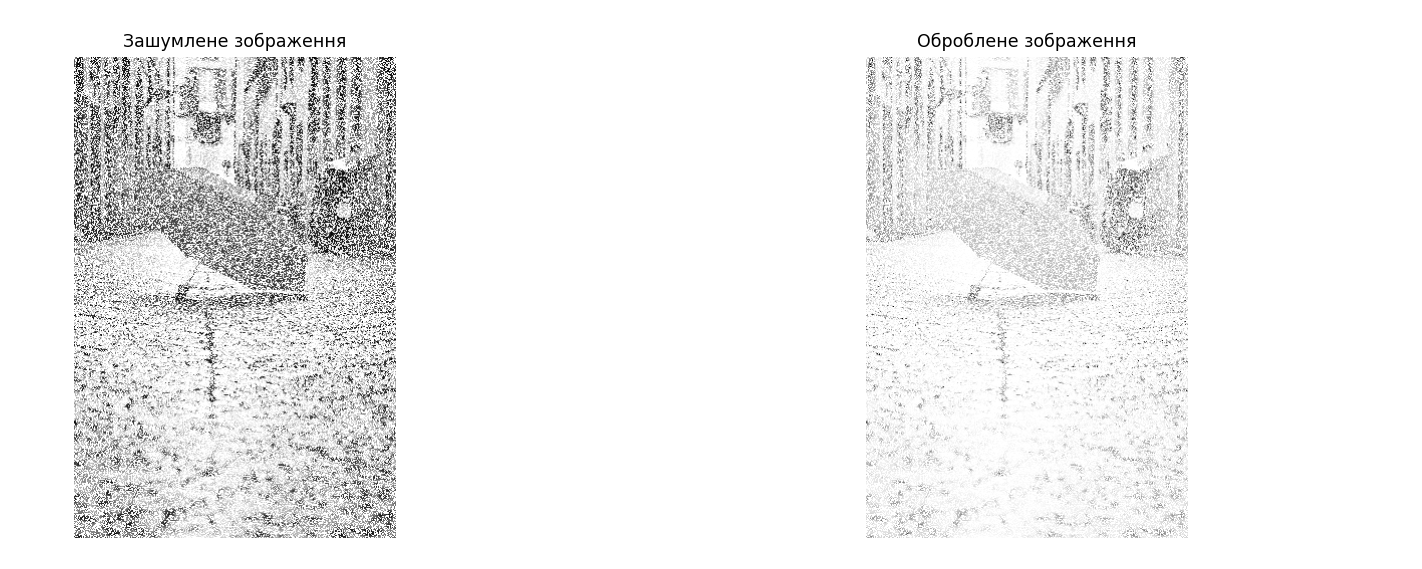
    plot\_images([gray\_image, negative\_image, log\_image, power\_image],

                ['Оригінальне', 'Негатив', 'Логарифмічне перетворення', 'Степеневе перетворення'])

    plot\_images([noisy\_image, denoised\_image], ['Зашумлене зображення', 'Оброблене зображення'])

**Результат:**

****

****

**  
Висновок:** під час виконання цієї лабораторної роботи, я навчилась, як застосовувати методи градаційних перетворень для поліпшення якості зображень.