МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное автономное

образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

«Пространственные методы обработки изображений»

Отчет по лабораторной работе № 12

по дисциплине «Технологии распознавания образов»

Выполнил студент группы ПИЖ-б-о-21-1

Халимендик Я. Д. « » 2023г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2023

Цель работы: сглаживание изображений с помощью различных фильтров нижних частот. Усвоение навыков применения 2D-свертки к изображениям; нахождение градиентов изображения, края и т. д. Изучение функций: cv2.Sobel (), cv2.Scharr (), cv2.Laplacian ().

Ход работы:

Задание 6.1.

Создать файл с зашумлением изображения шумом типа соль-перец.

Задание 6.1.

Создать файл с зашумлением изображения шумом типа соль-перец

```
red, green, blue = (255, 0, 0), (0, 255, 0), (0, 0, 255)
rgb = [red, green, blue]
```

```
def sp_noise(image, prob):
    output = np.zeros(image.shape, np.uint8)
    thres = 1- prob
    for i in range(image.shape[0]):
        for j in range(image.shape[1]):
            rnd = random.random()
            if rnd > thres:
                output[i][j] = random.choice(rgb)
            else:
                output[i][j] = image[i][j]
    return output
```

```
image = cv2.imread('img.jpg')
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
image = cv2.resize(image, (900, 600))
```

```
noise_img = sp_noise(image, 0.3)
res = np.hstack((image, noise_img))
```

```
plt.figure(figsize=(20,20))
plt.axis("off")
plt.imshow(res);
```

Рисунок 1 – Результат работы

Задание 6.2.

Провести сглаживание изображения с помощью функции cv2.filter2D (), используя ядро 5×5.

Задание 6.2.

Провести сглаживание изображения с помощью функции cv2.filter2D (), используя ядро 5×5.

```
img = cv2.imread('img.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
kernel = np.ones((5, 5), np.float32) / 25
dst = cv2.filter2D(img, -1, kernel)
```

```
plt.figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.axis("off")
plt.subplot(122),plt.imshow(dst),plt.title('Averaging')
plt.axis("off")
plt.show()
```

Рисунок 2 – Результат работы

Задание 6.3.

Провести усреднение изображения с помощью функции cv2.blur (), используя ядро 5×5.

Задание 6.3.

Провести усреднение изображения с помощью функции cv2.blur (), используя ядро 5×5.

```
img = cv2.imread('img.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
blur = cv2.blur(img, (5, 5))
```

```
plt.figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img), plt.title('Original')
plt.axis("off")
plt.subplot(122),plt.imshow(blur), plt.title('Blurred')
plt.axis("off")
plt.show()
```

Рисунок 3 – Результат работы

Задание 6.4.

Добавить к исходному изображению 20−30% шума. Провести фильтрацию изображения по Гауссу, используя ядро 5×5.

Задание 6.4.

Добавить к исходному изображению 20−30% шума. Провести фильтрацию изображения по Гауссу, используя ядро 5×5.

```
img = cv2.imread('img.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
blur = cv2.GaussianBlur(img, (5 ,5), 0)
```

```
plt.figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img), plt.title('Original')
plt.axis("off")
plt.subplot(122),plt.imshow(blur), plt.title('Blurred')
plt.axis("off")
plt.show()
```

Рисунок 4 – Результат работы

Задание 6.5.

Добавить к исходному изображению 20–50% шума. Провести медианную фильтрацию изображения, используя ядро 5□5.

Задание 6.5.

Добавить к исходному изображению 20–50% шума. Провести медианную фильтрацию изображения, используя ядро 5□5.

```
img = cv2.imread('img.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
median = cv2.medianBlur(img,5)
```

```
plt.figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img), plt.title('Original')
plt.axis("off")
plt.subplot(122),plt.imshow(median), plt.title('Blurred')
plt.axis("off")
plt.show()
```

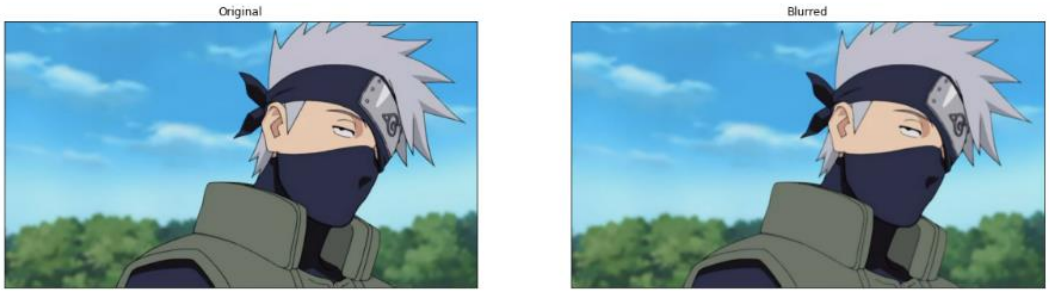

Рисунок 5 – Результат работы

Задание 6.6.

Создать файл с изображением, в котором обязательно присутствуют вертикальные и горизонтальные линии. С помощью оператора Собеля обнаружить и выделить эти линии.

Задание 6.6.

Создать файл с изображением, в котором обязательно присутствуют вертикальные и горизонтальные линии. С помощью оператора Собеля обнаружить и выделить эти линии.

```
img = cv2.imread('img.jpg', 0)
img = cv2.resize(img, (900, 600))
```

```
sobel_vertical = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
sobel_horizontal = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=5)
```

```
plt.figure(figsize=(15,15))
plt.subplot(131),plt.imshow(img), plt.title('Original')
plt.axis("off")

plt.subplot(132),plt.imshow(sobel_vertical), plt.title('sobel_vertical')
plt.axis("off")

plt.subplot(133),plt.imshow(sobel_horizontal), plt.title('sobel_horizontal')
plt.axis("off")

plt.show()
```

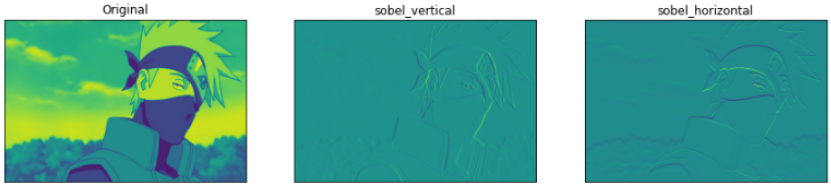

Рисунок 6 – Результат работы

Задание 6.7.

Сравнить оба способа для горизонтального фильтра Собела с преобразованием в cv2.CV_8U и без него.

Задание 6.7.

Сравнить оба способа для горизонтального фильтра Собела с преобразованием в cv2.CV_8U и без него.

```
img = cv2.imread('img.jpg', 0)
```

```
# Output dtype = cv2.CV_8U
sobelx8u = cv2.Sobel(img,cv2.CV_8U,1,0,ksize=5)

# Output dtype = cv2.CV_64F.
sobelx64f = cv2.Sobel(img,cv2.CV_64F,1,0,ksize=5)
abs_sobel64f = np.absolute(sobelx64f)
sobel_8u = np.uint8(abs_sobel64f)
```

```
plt.figure(figsize=(15,15))
plt.subplot(131),plt.imshow(img, cmap = 'gray'), plt.title('Original')
plt.axis("off")

plt.subplot(132),plt.imshow(sobelx8u, cmap = 'gray'), plt.title('Sobel CV_8U')
plt.axis("off")

plt.subplot(133),plt.imshow(sobel_8u, cmap = 'gray'), plt.title('Sobel abs(CV_64F)')
plt.axis("off")

plt.show()
```

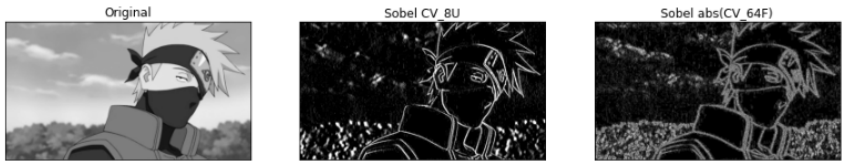

Рисунок 7 – Результат работы

Задание 6.8.

Создать файл с изображением, который обязательно содержит вертикальные и горизонтальные линии. С помощью оператора Превитта обнаружить и выделить эти линии.

Задание 6.8.

Создать файл с изображением, который обязательно содержит вертикальные и горизонтальные линии. С помощью оператора Превитта обнаружить и выделить эти линии.

```
img = cv2.imread('img.jpg', 0)
img = cv2.resize(img, (900, 600))
```

```
xkernel = np.array([[-1, -1, -1], [0, 0, 0], [1, 1, 1]])
ykernel = np.array([[-1, 0, 1], [-1, 0, 1], [-1, 0, 1]])
```

```
img_prewittx = cv2.filter2D(img, -1, xkernel)
img_prewitty = cv2.filter2D(img, -1, ykernel)
```

```
plt.figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img_prewittx, cmap = 'gray'), plt.title('img_prewittx')
plt.axis("off")

plt.subplot(122),plt.imshow(img_prewitty, cmap = 'gray'), plt.title('Sobel img_prewitty')
plt.axis("off")

plt.show()
```

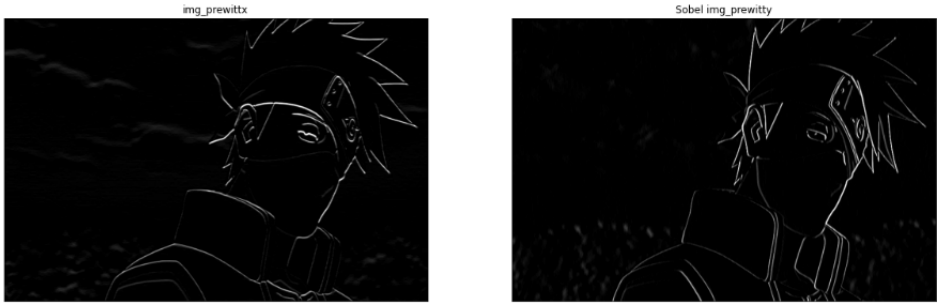

Рисунок 8 – Результат работы

Задание 6.9.

Используя оператор Робертса, выделить линии на изображении.

Задание 6.9.

Используя оператор Робертса, выделить линии на изображении.

```
kernel1 = np.array([[1, 0], [0, 1]])
kernel2 = np.array ([[0, 1],[0, 1]])
```

```
img_robx = cv2.filter2D(img, -1, kernel1)
img_roby = cv2.filter2D(img, -1, kernel2)

output_image = img_robx + img_roby
```

```
plt.imshow(output_image, cmap = 'gray'), plt.title('output_image')
plt.axis("off")
plt.show()
```

output_image

Рисунок 9 – Результат работы

Задание 6.10.

Создать файл с изображением, в котором присутствуют перепады изображения. С помощью оператора Лапласа обнаружить и выделить эти перепады.

Задание 6.10.

Создать файл с изображением, в котором присутствуют перепады изображения. С помощью оператора Лапласа обнаружить и выделить эти перепады.

```
laplacian = cv2.Laplacian(img, cv2.CV_64F)
```

```
plt.figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img, cmap = 'gray'), plt.title('Original')
plt.axis("off")

plt.subplot(122),plt.imshow(laplacian, cmap = 'gray'), plt.title('Laplacian')
plt.axis("off")

plt.show()
```

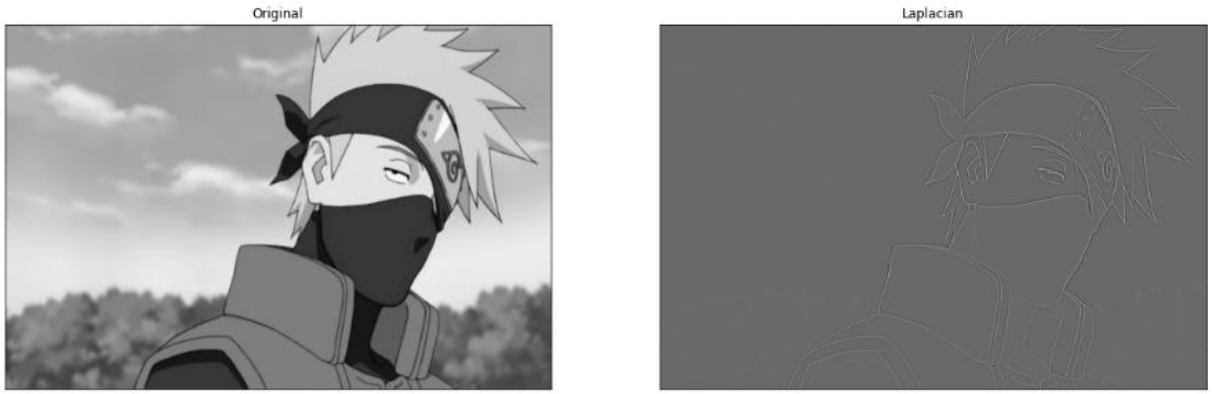

Рисунок 10 – Результат работы

Индивидуальное задание.

Провести усреднение изображения с помощью функции cv2.blur(), и примененить различные способы обнаружения и выделения линий и перепадов изображения

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

```
img = cv2.imread('img.jpg', 0)
```

```
blur = cv2.blur(img,(20,20))
```

Оператор Собеля

```
sobel_vertical = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
sobel_horizontal = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=5)
```

```
sobel_vertical_blur = cv2.Sobel(blur, cv2.CV_64F, 1, 0, ksize=5)
sobel_horizontal_blur = cv2.Sobel(blur, cv2.CV_64F, 0, 1, ksize=5)
```

```
images = [img, sobel_vertical, sobel_horizontal,blur, sobel_vertical_blur, sobel_horizontal_blur]
title = ['Original', 'sobel_vertical', 'sobel_horizontal', 'blur', 'sobel_vertical', 'sobel_horizontal']

plt.figure(figsize=(15,15))
for i in range(6):
    plt.subplot(3,3,i+1),plt.imshow(images[i],'gray')
    plt.title(title[i])
    plt.xticks([]),plt.yticks([])
plt.show()
```

Рисунок 11 – Результат работы

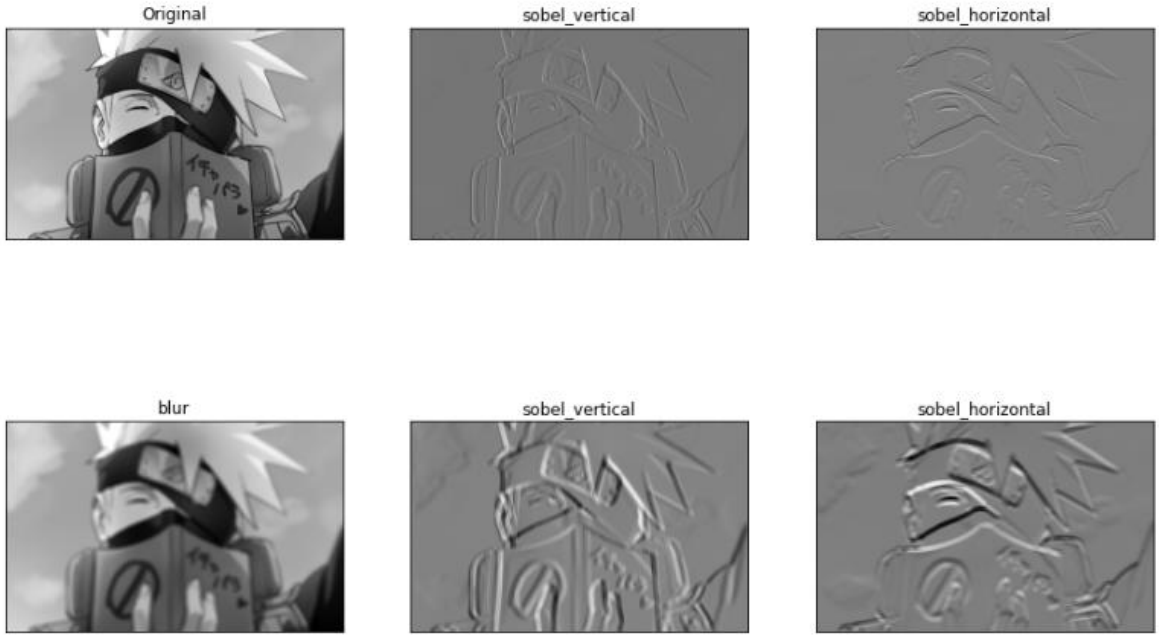

Рисунок 12 – Результат работы

Оператор Превитта

```
xkernel = np.array([[-1, -1, -1], [0, 0, 0], [1, 1, 1]])
ykernel = np.array([[-1, 0, 1], [-1, 0, 1], [-1, 0, 1]])
```

```
img_prewittx = cv2.filter2D(img, -1, xkernel)
img_prewitty = cv2.filter2D(img, -1, ykernel)
```

```
img_prewittx_blur = cv2.filter2D(blur, -1, xkernel)
img_prewitty_blur = cv2.filter2D(blur, -1, ykernel)
```

```
images = [img, img_prewittx, img_prewitty, blur, img_prewittx_blur, img_prewitty_blur]
title = ['Original', 'sobel_vertical', 'sobel_horizontal', 'blur', 'sobel_vertical', 'sobel_horizontal']

plt.figure(figsize=(15,15))
for i in range(6):
    plt.subplot(3,3,i+1),plt.imshow(images[i],'gray')
    plt.title(title[i])
    plt.xticks([]),plt.yticks([])
plt.show()
```

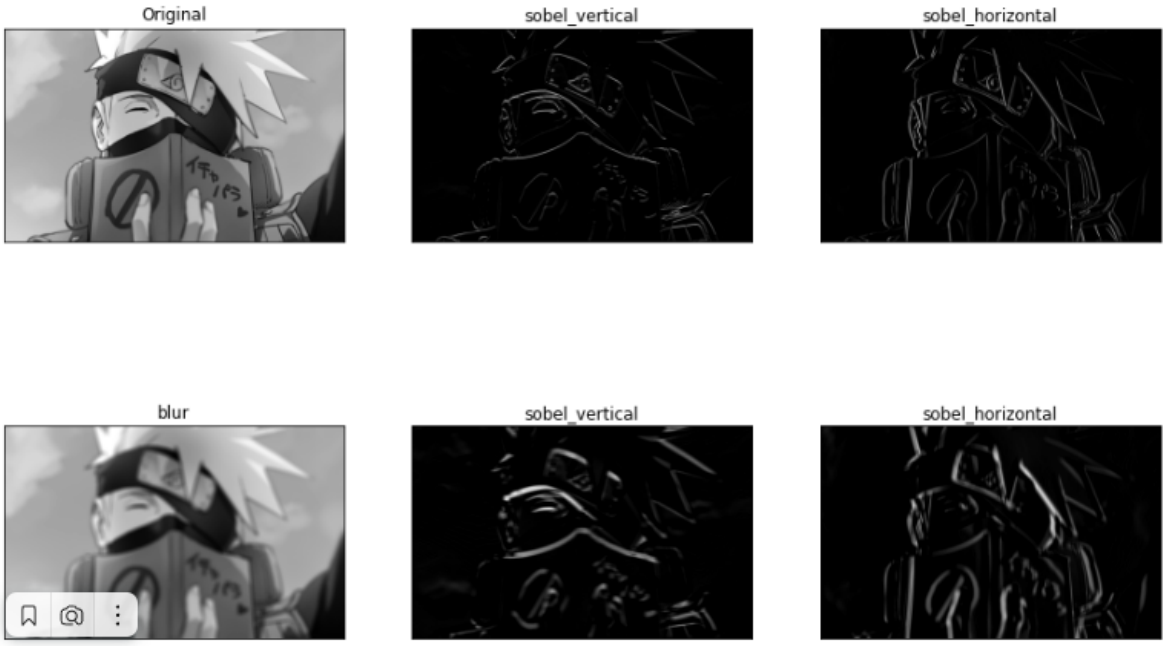

Рисунок 13 – Результат работы

Оператора Лапласа

```
laplacian = cv2.Laplacian(img, cv2.CV_64F)
laplacian_blur = cv2.Laplacian(blur, cv2.CV_64F)
```

```
images = [img, laplacian, blur, laplacian_blur]
title = ['Original', 'Laplacian', 'blur', 'Laplacian']

plt.figure(figsize=(15,15))
for i in range(4):
    plt.subplot(2,2,i+1),plt.imshow(images[i],'gray')
    plt.title(title[i])
    plt.xticks([]),plt.yticks([])
plt.show()
```

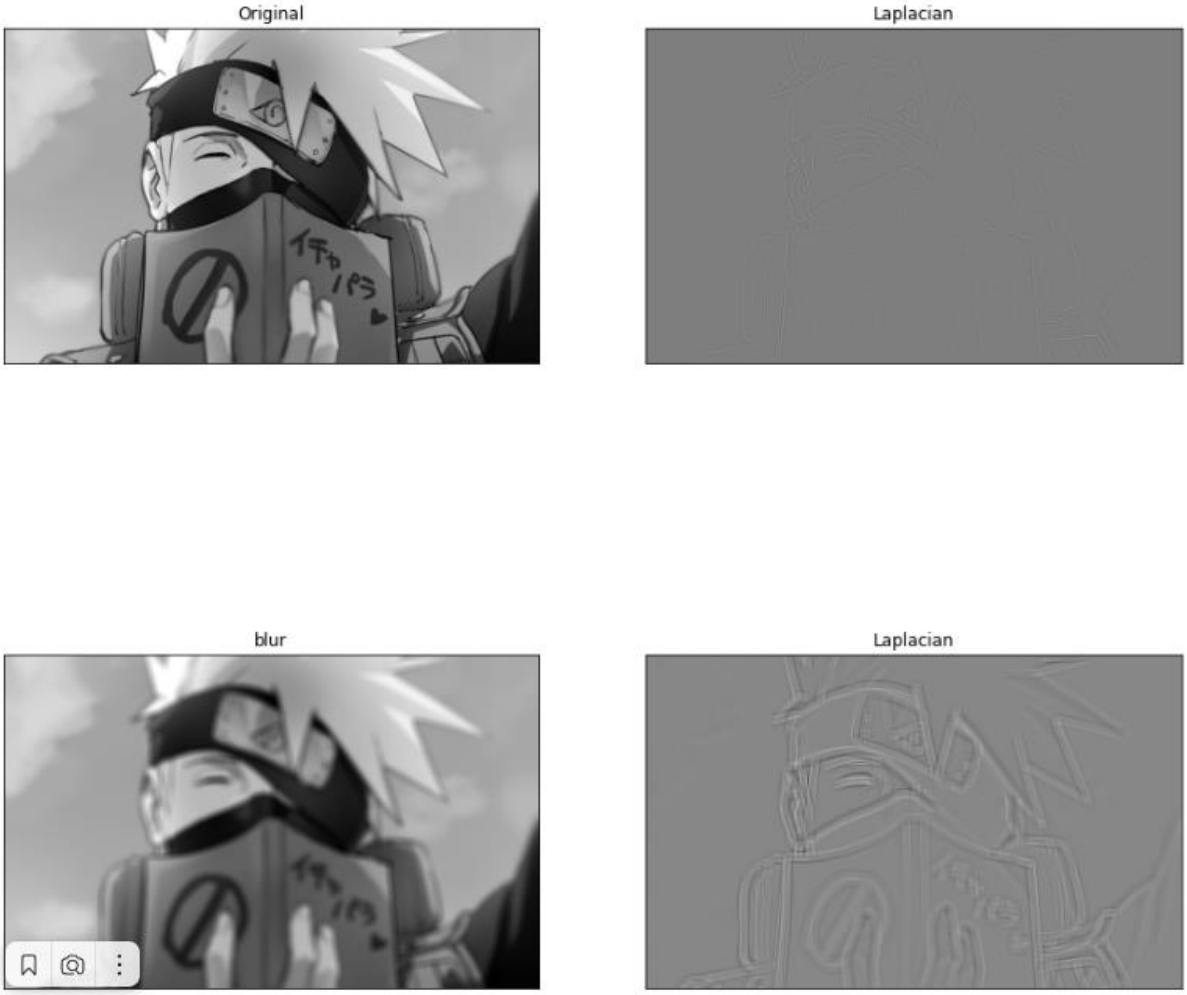

Рисунок 14 – Результат работы