Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

«Пространственные методы обработки изображений»

ОТЧЕТ
по лабораторной работе №12
дисциплины
«Основы распознавания образов»

Выполнил:
Луценко Дмитрий Андреевич
2 курс, группа ПИЖ-б-о-21-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

_____
(подпись)

Проверил:
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Пространственные методы обработки изображений

Цель работы: изучение алгоритмов пространственной обработки изображений.

Ход работы:

Задание 6.1

Создать файл с зашумлением изображения шумом типа соль-перец.

Сначала создадим кортеж RGB с красным, зелёным, сисним цветом и функцию зашумления, принимающую на вход изображение и вероятность зашумления.

```
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]):
            rdn = random.random()
            if rdn > thres:
                output[i][j] = random.choice(rgb)
            else:
                output[i][j] = image[i][j]

    return output
```

Загрузим изображение и применим к нему нашу функцию

```
image = cv2.imread('images/jaguar.jpg')
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
image = cv2.resize(image, (900, 900))

noise_img = sp_noise(image,0.3)

res = np.hstack((image, noise_img))
noise_img = cv2.cvtColor(noise_img, cv2.COLOR_BGR2RGB)
cv2.imwrite('images/Noise.jpg', noise_img)
plt.axis('off')
plt.imshow(res);
```

Рисунок 1 – Задания

Задание 6.2

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

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

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

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()
```

Задание 6.3

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

```
img = cv2.imread ('images/dom.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
blur = cv2.blur(img, (5, 5))

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.axis('off')
plt.subplot(122),plt.imshow(blur),plt.title('Averaging')
plt.axis('off')
plt.show()
```

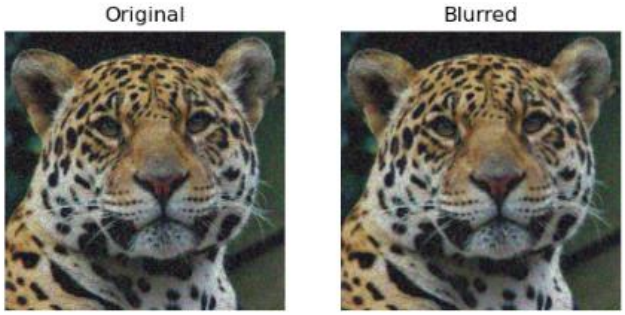
Рисунок 2 – Задания

Задание 6.4

Провести фильтрацию изображения по Гауссу, используя ядро 5×5.

```
img = cv2.imread ('images/Noise.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
blur = cv2.GaussianBlur(img, (5, 5), 0)

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()
```



Задание 6.5

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

```
img = cv2.imread ('images/median.png')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
median = cv2.medianBlur(img,5)

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 3 – Задания

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 ('images/dom.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (900, 600))
```

Теперь применим оператор Собеля для горизонтальных и вертикальных линий.

```
sobel_vertical = cv2.normalize(img, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U)
sobel_vertical = cv2.Sobel(sobel_vertical, cv2.CV_64F, 1, 0, ksize=5)
sobel_vertical = cv2.normalize(sobel_vertical, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U)

sobel_horizontal = cv2.normalize(img, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U)
sobel_horizontal = cv2.Sobel(sobel_horizontal, cv2.CV_64F, 0, 1, ksize=5)
sobel_horizontal = cv2.normalize(sobel_horizontal, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U)

plt.subplot(131),plt.imshow(img),plt.title('Original')
plt.axis('off')
plt.subplot(132),plt.imshow(sobel_vertical),plt.title('Vertical')
plt.axis('off')
plt.subplot(133),plt.imshow(sobel_horizontal),plt.title('Horizontal')
plt.axis('off')
plt.show()
```

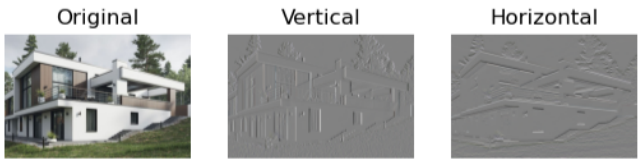


Рисунок 4 – Задания

Задание 6.7.

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

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

sobelx8u = cv2.Sobel(img, cv2.CV_8U, 1, 0, ksize=5)
sobelx64f = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
abs_sobel64f = np.absolute(sobelx64f)
sobel_8u = np.uint8(abs_sobel64f)

plt.subplot(131),plt.imshow(img),plt.title('Original')
plt.axis('off')
plt.subplot(132),plt.imshow(sobelx8u),plt.title('Sobel (CV8U)')
plt.axis('off')
plt.subplot(133),plt.imshow(sobel_8u),plt.title('Sobel abs (CV64F)')
plt.axis('off')
plt.show()
```

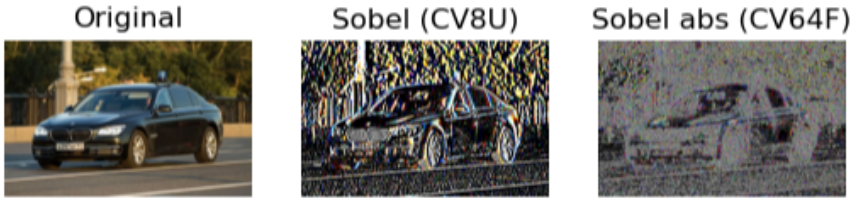


Рисунок 5 – Задания

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('images/dom.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (900, 600))
```

Создадим маски для X и Y, а затем нарисуем линии функцией cv2.filter2D - Функция соединения изображения с ядром, здесь 1 – это глубина изображения (если значение отрицательное, то глубина соответствует исходному изображению, как и cv2.CV_64F)

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

img_prewittx = cv2.filter2D(img, -1, kernel1)
img_prewitty = cv2.filter2D(img, -1, kernel2)

plt.subplot(131), plt.imshow(img), plt.title('Original')
plt.axis('off')
plt.subplot(132),plt.imshow(img_prewittx), plt.title('Prewitt X')
plt.axis('off')
plt.subplot(133),plt.imshow(img_prewitty), plt.title('Prewitt Y')
plt.axis('off')
plt.show()
```

Рисунок 6 – Задания

Задание 6.9

Используя оператор Робертса, выделить линии на изображении.

```
img = cv2.imread('images/bmw.jpg', 0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (900, 600))
```

Создадим маски ядер оператора Робертса для осей X и Y, а затем применим их для выделения линий с помощью функции cv2.filter2D

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

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

output_image = img_robx + img_roby

plt.axis('off')
plt.imshow(output_image);
```

Рисунок 7 – Задания

Задание 6.10 ¶

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

Оператор Лапласа подчеркивает разрывы, скачки яркости и ослабляет плавное изменение яркости. Для повышения резкости изображение, обработанное оператором Лапласа, накладывают на исходное изображение.

```
img = cv2.imread('images/o001oo.png')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

laplacian = cv2.Laplacian(img, cv2.CV_64F)
laplacian = cv2.normalize(laplacian, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U)

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.axis('off')
plt.subplot(122),plt.imshow(laplacian),plt.title('Laplacian')
plt.axis('off')
plt.show()
```

Рисунок 8 - Задания

Примение операторов Лапласа, Собеля, Робертса и Превитта

```
img = cv2.imread ('images/Titanic.jpg')
```

Сначала составим маски(ядра) для оператора Превитта (kernel1 и kernel2) и для оператора Робертса (kernel3 и kernel4)

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

kernel3 = np.array([[-1, 0], [0, 1]])
kernel4 = np.array ([[0, -1], [1, 0]])
```

Теперь применим их к изображению функцией cv2.filter2D() - функцией соединения изображения с ядром, здесь 1 – это глубина изображения (если значение отрицательное, то глубина соответствует исходному изображению)

```
img_prewittx = cv2.filter2D(img, -1, kernel1)
img_prewitty = cv2.filter2D(img, -1, kernel2)

img_robx = cv2.filter2D(img, -1, kernel3)
img_roby = cv2.filter2D(img, -1, kernel4)
```

Для примения оператора Собеля воспользуемся функцией cv2.Sobel(). Для предотварщения возникновения ошибок нормализуем предварительно изображение функцией cv2.normalize(). Нормализация - это процесс, который изменяет диапазон значения интенсивности пикселей.

```
sobel_vertical = cv2.normalize(
    img, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U
)
sobel_vertical = cv2.Sobel(
    sobel_vertical, cv2.CV_64F, 1, 0, ksize=5
)
sobel_vertical = cv2.normalize(
    sobel_vertical, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U
)

sobel_horizontal = cv2.normalize(
    img, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U
)
sobel_horizontal = cv2.Sobel(
    sobel_horizontal, cv2.CV_64F, 0, 1, ksize=5
)
sobel_horizontal = cv2.normalize(
    sobel_horizontal, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U
)
```

Рисунок 9 – Индивидуальное задание

```
sobel_horizontal = cv2.normalize(
    img, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U
)
sobel_horizontal = cv2.Sobel(
    sobel_horizontal, cv2.CV_64F, 0, 1, ksize=5
)
sobel_horizontal = cv2.normalize(
    sobel_horizontal, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U
)
```

Аналогично воспользуемся функцией cv2.Laplacian

```
]: laplacian = cv2.Laplacian(img, cv2.CV_64F)
laplacian = cv2.normalize(laplacian, None, 0, 255, cv2.NORM_MINMAX, cv2.CV_8U)
```

Осталось отобразить результаты с помощью matplotlib.pyplot

```
]: plt.subplot(331),plt.imshow(img),plt.title('Original')
plt.axis('off')
plt.subplot(332),plt.imshow(sobel_vertical),plt.title('Sobel vertical')
plt.axis('off')
plt.subplot(333),plt.imshow(sobel_horizontal),plt.title('Sobel horizontal')
plt.axis('off')
plt.subplot(334),plt.imshow(laplacian),plt.title('Laplasian')
plt.axis('off')
plt.subplot(335),plt.imshow(img_roby),plt.title('Roberts vertical')
plt.axis('off')
plt.subplot(336),plt.imshow(img_robx),plt.title('Roberts horizontal')
plt.axis('off')
plt.subplot(337),plt.imshow(img_prewitty),plt.title('Previtt vertical')
plt.axis('off')
plt.subplot(338),plt.imshow(img_prewittx),plt.title('Previtt horizontal')
plt.axis('off')
plt.show()
```

Рисунок 10 – Индивидуальное задание

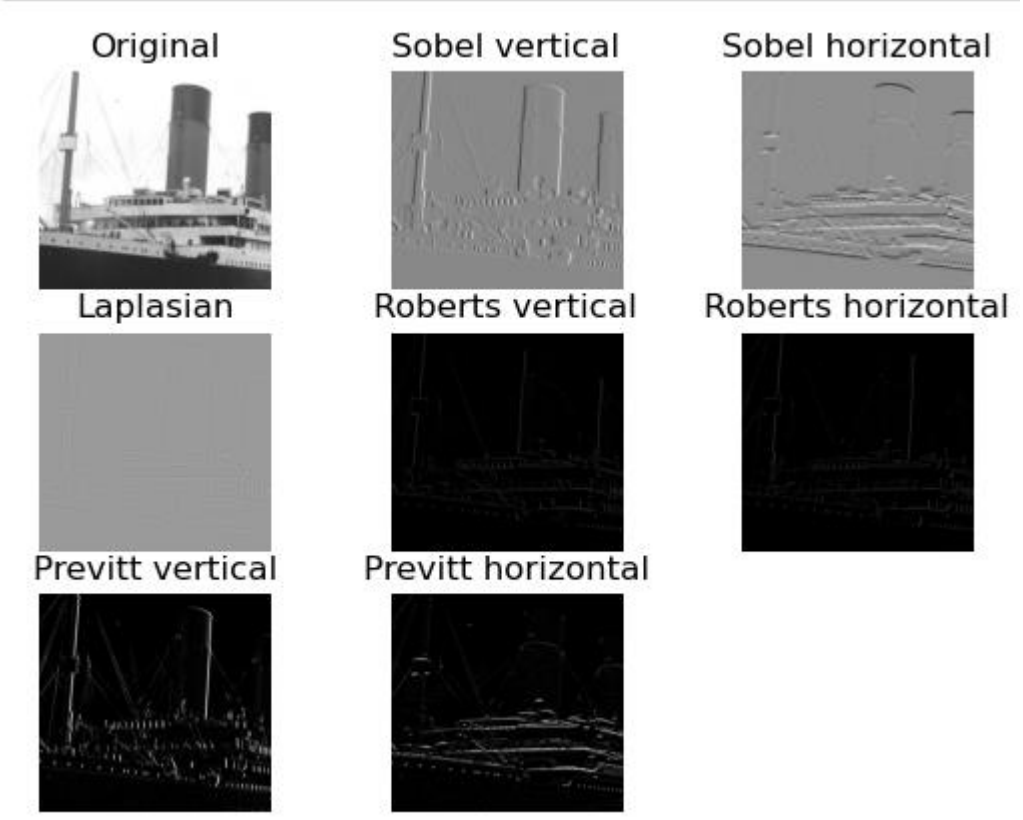


Рисунок 12 – Индивидуальное задание

Вывод: в ходе лабораторной рабоыт изучены пространственные алгоритмы обработки изображений.

Ответы на контрольные вопросы:

1. Что такое маска? Матрицу локального преобразования некоторого пикселя называют фильтром, маской, шаблоном или окном.

2. Что используют для сглаживания изображения? Для сглаживания изображений используют усредняющий фильтр. Усреднение участка изображения проводится с помощью ядра, например, 5×5. Для сглаживания (размытия) изображения используют также свертку с маской, при этом размерность маски должна быть равна сумме весов

3. Какие функции есть для сглаживания изображения? cv2.filter2D и cv2.blur () или cv2.boxFilter ()

4. Как работают функции из вопроса №3? Мы берем скользящее окно, попиксельно умножаем яркость каждого пикселя этого окна на коэффициент в матрице, складываем и результат записываем в центральную точку окна. Потом окно сдвигаем на один пиксель и делаем то же самое. И так пока не пройдем по всему изображению.

5. Как работает Гауссова фильтрация? Фильтрация с ядром Гаусса выполняется маской, веса которой находятся с помощью функции Гаусса. Рассматриваемая фильтрация осуществляется с помощью функции cv2.GaussianBlur (). В скобках второй аргумент – это ширина и высота ядра, которые должны быть положительными и нечетными. Указывается также стандартное отклонение в направлениях X и Y, sigmaX и sigmaY соответственно.

6. Как работает медианная фильтрация? В процессе этой фильтрации функция cv2.medianBlur () вычисляет медианное значение всех пикселей, окружающих центральный пиксель, и его значение заменяется медианным значением. Это очень эффективно для устранения шума соли и перца. Размер ядра должен быть положительным нечетным целым числом.

7. Какие есть функции для обнаружения перепадов? cv2.Sobel (), cv2.Scharr (), cv2.Laplacian ()

8. В чём различия метода Собеля и Превитта при обнаружении перепадов? Различие между операторами Превитта и Собеля в разных весовых коэффициентах

9. В чём заключается отличительная особенность в выделении границ методом Робертса? Метод Робертса использует маску 2x2, из-за чего в данном методе нет чётко выраженного центрального элемента из-за чего нет такого сильно эффекта как от других методов, однако, данный метод отличается своим быстродействием

10. Что делает оператор Лапласа? Оператор Лапласа подчеркивает разрывы, скачки яркости и ослабляет плавное изменение яркости. Для повышения резкости изображение, обработанное оператором Лапласа, накладывают на исходное изображение.