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
«Технологии распознавания образов»

Выполнил:
Мизин Глеб Егорович
2 курс, группа ПИЖ-б-о-21-1,
011.03.04 «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

In [247]:
```
import cv2
import numpy as np
import random
from matplotlib import pyplot as plt
```

In [248]:
```
def img_input(img_path, size=(0,0), type=1):
    img = cv2.imread(img_path,type)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    if size != (0,0):
        img = cv2.resize(img, size)
        return img
    else:
        return img
```

Функция вывода изображения

In [249]:
```
def img_print(images, titles):
    num_of_img = len(images)
    for i in range(num_of_img):
        plt.subplot(1,num_of_img,i+1),plt.imshow(images[i])
        plt.title(titles[i])
        plt.xticks([]),plt.yticks([])
    plt.show ()
```

Задание 6.1 Создать файл с зашумлением изображения шумом типа соль-перец.

Создаем 3 возможных цвета - красный, зеленый и синий. Помещаем их в кортеж

In [250]:
```
red, green, blue = (255, 0, 0), (0, 255, 0), (0, 0,255)
rgb = [red, green, blue]
```

Создаем функцию с параметрами (<наше изображение>, <вероятность зашумления>), создаем массив нулей такого же размера и формата как исходное изображение, задаем порог. Пробегаем все столбцы и для каждого из них пробегаем все строки, задаем случайное число от 0 до 1. Если это случайное число выпало больше нашего порога задаем пикселю случайное значение из кортежа Иначе оставляем пиксель без изменения и возвращаем получившееся изображение

In [251]:
```
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
```

Загружаем картинку

In [252]:
```
image = img_input('pic/Cat.jpg',size=(900,600))
```

Применяем к нашему изображению image, созданную функцию sp_noise, где 0.3 - вероятность зашумления пикселя

In [253]:
```
noise_img = sp_noise(image,0.3)
```

Выводим результат, для наглядности так же выводим оригинал

In [254]:
```
title = ['Оригинал', 'Зашумлённое']
res = [image, noise_img]

img_print(res, title)
```

Задание 6.2 Провести сглаживание изображения с помощью функции cv2.filter2D (), используя ядро 6×6.

In [255]:
```
image = img_input('pic/Cat2.jpg')

kernel = np.ones((6,6),np.float32)/25
dst = cv2.filter2D(image,-1,kernel)

title = ['Оригинал', 'Сглаженное']
res = [image, dst]

img_print(res, title)
```

Задание 6.3 Провести усреднение изображения с помощью функции cv2.blur (), используя ядро 6×6.

In [256]:
```
image = img_input('pic/Cat3.jpg')

blur = cv2.blur(image,(6,6))

title = ['Оригинал', 'Заблюренное']
res = [image, blur]

img_print(res, title)
```

Задание 6.4 Добавить к исходному изображению 20–30% шума. Провести фильтрацию изображения по Гауссу, используя ядро 5×5.

In [257]:
```
image = img_input('pic/jojo.jpg')


mash =sp_noise(image, 0.5)
blur = cv2.GaussianBlur(image,(5,5), 0)


title = ['Оригинал', 'Шумы', 'Фильтр по Гауссу']
res = [image, mash, blur]

img_print(res, title)
```

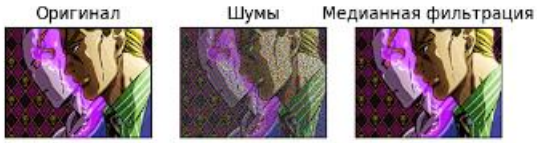
Задание 6.5 Добавить к исходному изображению 20–50% шума. Провести медианную фильтрацию изображения, используя ядро 5x5.

In [258]:
```
image = img_input('pic/jojo2.jpg')


mash =sp_noise(image, 0.5)
median = cv2.medianBlur(image,5)


title = ['Оригинал', 'Шумы', 'Медианная фильтрация']
res = [image, mash, median]

img_print(res, title)
```



Задание 6.6 Создать файл с изображением, в котором обязательно присутствуют вертикальные и горизонтальные линии. С помощью оператора Собеля обнаружить и выделить эти линии.

In [259]:
```
image = img_input('pic/jojo3.jpg');
```

Функция Собеля для вычисления вертикальных линий

In [260]:
```
sobel_vertical = cv2.Sobel(image, cv2.CV_64F, 1, 0, ksize=5);
```

Функция Собеля для вычисления горизонтальных линий

In [261]:
```
Sobel_horizontal = cv2.Sobel(image, cv2.CV_64F, 0, 1, ksize=5);
```

In [259]:
```
image = img_input('pic/jojo3.jpg');
```

Функция Собеля для вычисления вертикальных линий

In [260]:
```
sobel_vertical = cv2.Sobel(image, cv2.CV_64F, 1, 0, ksize=5);
```

Функция Собеля для вычисления горизонтальных линий

In [261]:
```
Sobel_horizontal = cv2.Sobel(image, cv2.CV_64F, 0, 1, ksize=5);
```

Вывод

In [262]:
```
res = [image, sobel_vertical, Sobel_horizontal]
title = ['Оригинал', 'Вертикальные', 'Горизонтальные']

img_print(res, title)
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```

Задание 6.7 Сравнить оба способа для горизонтального фильтра Собела с преобразованием в cv2.CV_8U и без него.

In [263]:
```
image = img_input('pic/jojo3.jpg')
```

На выходе: dtype = cv2.CV_8U

In [264]:
```
sobelx8u = cv2.Sobel(image,cv2.CV_8U,1,0,ksize=5)
```

На выходе: dtype = cv2.CV_64F

In [265]:
```
sobelx64f = cv2.Sobel(image,cv2.CV_64F,1,0,ksize=5)
abs_sobel64f = np.absolute(sobelx64f)
sobel_8u = np.uint8(abs_sobel64f)
```

Вывод

In [266]:
```
res = [image ,sobelx8u ,sobel_8u]
title = ['Оригинал', 'CV_8U', 'CV_64F']

img_print(res, title)
```

In [266]:
```
res = [image ,sobelx8u ,sobel_8u]
title = ['Оригинал', 'CV_8U', 'CV_64F']

img_print(res, title)
```

Задание 6.8 Создать файл с изображением, который обязательно содержит вертикальные и горизонтальные линии. С помощью оператора Превитта обнаружить и выделить эти линии.

In [267]:
```
image = img_input('pic/dom.jpg')
```

Создаем ядро (маску) для x

In [268]:
```
x_mask = np.array([[-1, -1, -1], [0, 0, 0], [1, 1, 1]])
```

Создаем ядро (маску) для y

In [269]:
```
y_mask = np.array([[-1, 0, 1], [-1, 0, 1], [-1, 0, 1]])
```

Функция соединения изображения с ядром (в данном случае с X, то есть выбраны будут горизонтальные линии), здесь -1 –это глубина изображения (если значение отрицательное, то глубина соответствует исходному изображению, как и cv2.CV_64F)

In [270]:
```
img_prewittx = cv2.filter2D(image, -1, x_mask)
```

Соединение изображения с ядром Y, выбор вертикальных линий

In [271]:
```
img_prewitty = cv2.filter2D(image, -1, y_mask)
```

Выведем полученный результат

In [272]:
```
res = [image, img_prewittx, img_prewitty]
title = ['Оригинал', 'Превитт по X', 'Превитт по Y']

img_print(res, title)
```

Задание 6.9 Используя оператор Робертса, выделить линии на изображении.

In [273]:
```
image = img_input('pic/avtolines.kpg.jpg', type=0)
```

Создадим маски для ядра X(kernel1) и Y(kernel2)

In [274]:
```
kernel1 = np.array([[1, 0], [0, 1]])
kernel2 = np.array ([[0, 1],[0, 1]])
```

Выделим линии

In [275]:
```
img_robx = cv2.filter2D(image, -1, kernel1)
img_roby = cv2.filter2D(image, -1, kernel2)
```

Выведем:

In [276]:
```
res = [image,img_robx, img_roby]
title = ['Оригинал', 'Робертс по X', 'Робертс по Y']

img_print(res, title)
```

Задание 6.10 Создать файл с изображением, в котором присутствуют перепады изображения. С помощью оператора Лапласа обнаружить и выделить эти перепады.

In [277]:
```
image = img_input('pic/jojo5.jpg')
```

Используем оператор Лапласа для обнаружения и выделения перепадов на изображении

In [278]:
```
laplacian = cv2.Laplacian(image, cv2.CV_64F)
```

Выведем результат

In [279]:
```
res = [image, laplacian]
title = ['Оригинал', 'Оператор Лапласа']

img_print(res, title)
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```

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.