МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

«Основы работы с библиотекой NumPy»

Отчет по лабораторной работе № 12 (6)

по дисциплине «Технологии распознавания образов»

Выполнил студент группы ПИЖ-б-о-21-1

Кучеренко С. Ю. « » 2023 г.

Подпись студента_____

Работа защищена « »_____2023 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 ().

Выполнение работы:

Примеры лабораторной работы

Задание 6.1.

Создать файл с зашумлением изображения шумом типа соль-перец.

In [38]:
```
import cv2
import numpy as np
import random
from matplotlib import pyplot as plt
```

In [39]:
```
red, green, blue = (255, 0, 0), (0, 255, 0), (0, 0, 255)
rgb = [red, green, blue]
```

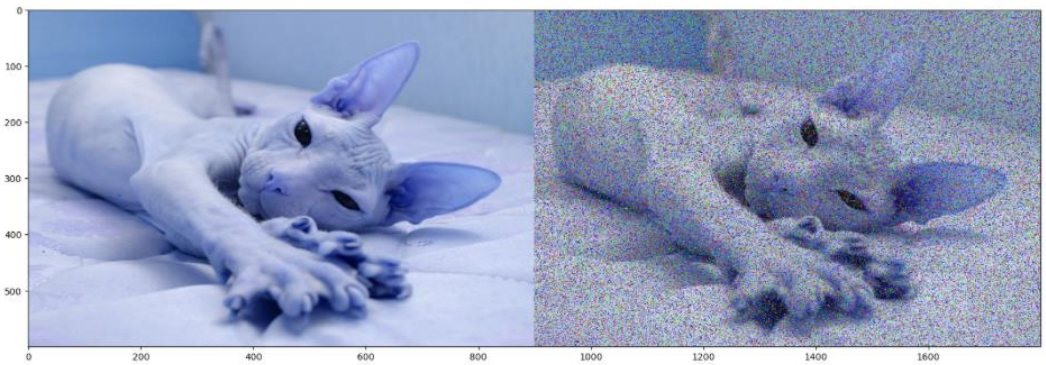
In [40]:
```
def sp_noise(image, prob):
    output = np.zeros(image.shape, np.uint8)
    thres = 1- prob
    for i in range(image.shape[0]):
        for j in range(image.shape[1]):
            rnd = random.random()
            if rnd > thres:
                output[i][j] = random.choice(rgb)
            else:
                output[i][j] = image[i][j]
    return output
```

In [41]:
```
image = cv2.imread('img/cat.jpg')
image = cv2.resize(image, (900, 600))
```

In [42]:
```
noise_img = sp_noise(image, 0.3)
res = np.hstack((image, noise_img))
```

In [43]:
```
figure(figsize=(20,20))
plt.imshow(res)
```

Out[43]:
```
<matplotlib.image.AxesImage at 0x13442cd30>
```



Задание 6.2.

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

In [61]:
```
img = cv2.imread('img/cat.jpg')
```

In [62]:
```
kernel = np.ones((5, 5), np.float32) / 25
dst = cv2.filter2D(img, -1, kernel)
```

In [63]:
```
figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(dst),plt.title('Averaging')
plt.xticks([]), plt.yticks([])
plt.show()
```

Задание 6.3.

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

In [64]:
```
img = cv2.imread('img/cat.jpg')
```

In [70]:
```
blur = cv2.blur(img, (5, 5))
```

In [71]:
```
figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img), plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur), plt.title('Blurred')
plt.xticks([]), plt.yticks([])
plt.show()
```

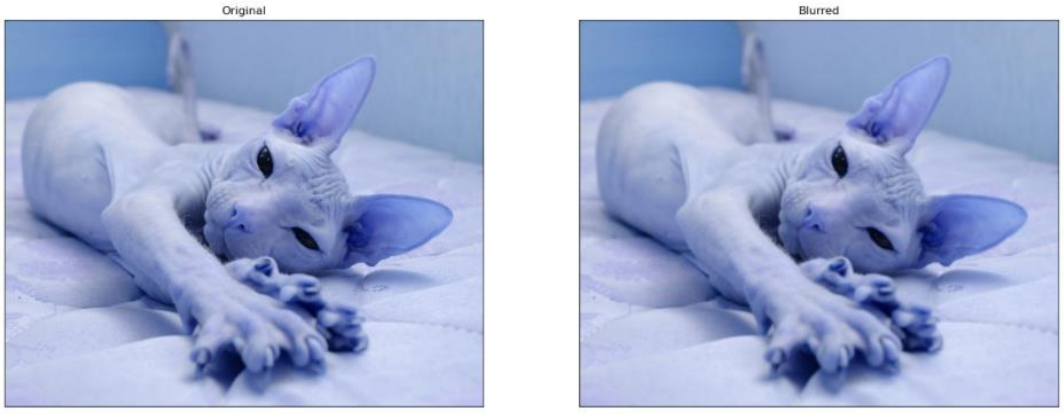
Задание 6.4.

Добавить к исходному изображению 20–30% шума. Провести фильтрацию изображения по Гауссу, используя ядро 10×10.

In [84]:
```
img = cv2.imread('img/cat.jpg')
```

In [85]:
```
blur = cv2.GaussianBlur(img, (5 ,5), 0)
```

In [86]:
```
figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img), plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur), plt.title('Blurred')
plt.xticks([]), plt.yticks([])
plt.show()
```



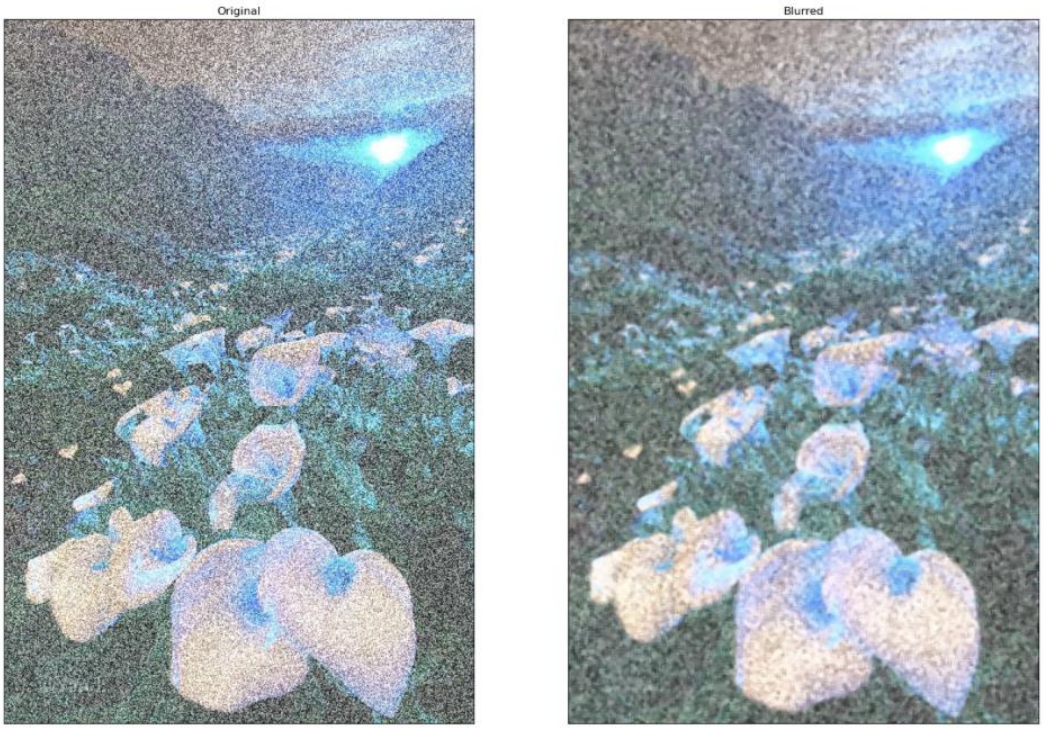
Задание 6.5.

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

In [89]:
```
img = cv2.imread('img/median.png')
```

In [90]:
```
median = cv2.medianBlur(img,5)
```

In [92]:
```
figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img), plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(median), plt.title('Blurred')
plt.xticks([]), plt.yticks([])
plt.show()
```



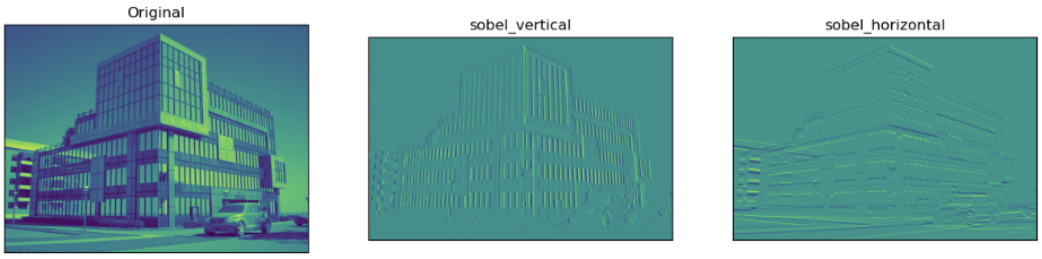
Задание 6.6.

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

In [104]:
```
img = cv2.imread('img/house.jpeg', 0)
img = cv2.resize(img, (900, 600))
```

In [105]:
```
sobel_vertical = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
sobel_horizontal = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=5)
```

In [110]:
```
figure(figsize=(15,15))
plt.subplot(131),plt.imshow(img), plt.title('Original')
plt.xticks([]), plt.yticks([])

plt.subplot(132),plt.imshow(sobel_vertical), plt.title('sobel_vertical')
plt.xticks([]), plt.yticks([])

plt.subplot(133),plt.imshow(sobel_horizontal), plt.title('sobel_horizontal')
plt.xticks([]), plt.yticks([])

plt.show()
```



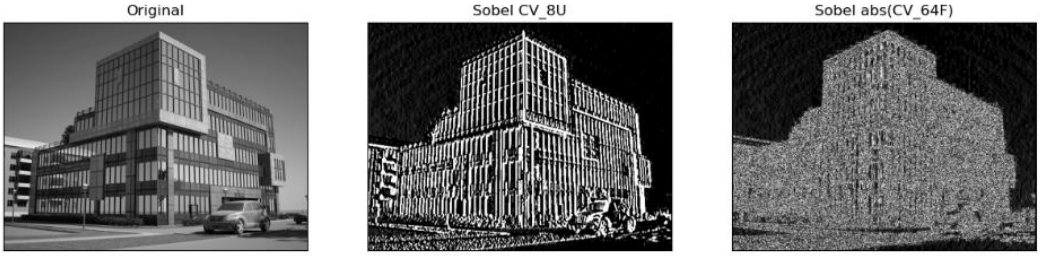
Задание 6.7.

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

In [111]:
```
img = cv2.imread('img/house.jpeg', 0)
```

In [112]:
```
# 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)
```

In [115]:
```
figure(figsize=(15,15))
plt.subplot(131),plt.imshow(img, cmap = 'gray'), plt.title('Original')
plt.xticks([]), plt.yticks([])

plt.subplot(132),plt.imshow(sobelx8u, cmap = 'gray'), plt.title('Sobel CV_8U')
plt.xticks([]), plt.yticks([])

plt.subplot(133),plt.imshow(sobel_8u, cmap = 'gray'), plt.title('Sobel abs(CV_64F)')
plt.xticks([]), plt.yticks([])

plt.show()
```



Задание 6.8.

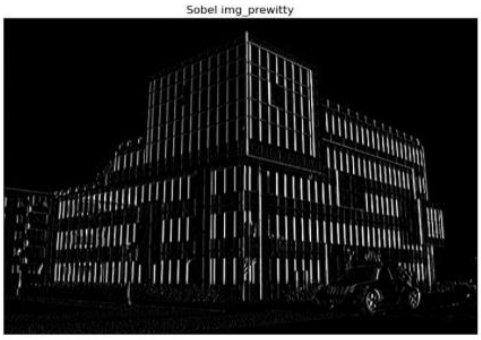
Создать файл с изображением, который обязательно содержит вертикальные и горизонтальные линии. С помощью оператора Превитта обнаружить и выделить эти линии.

In [117]:
```
img = cv2.imread('img/house.jpeg', 0)
img = cv2.resize(img, (900, 600))
```

In [125]:
```
xkernel = np.array([[-1, -1, -1], [0, 0, 0], [1, 1, 1]])
ykernel = np.array([[-1, 0, 1], [-1, 0, 1], [-1, 0, 1]])
```

In [126]:
```
img_prewittx = cv2.filter2D(img, -1, xkernel)
img_prewitty = cv2.filter2D(img, -1, ykernel)
```

In [127]:
```
figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img_prewittx, cmap = 'gray'), plt.title('img_prewittx')
plt.xticks([]), plt.yticks([])

plt.subplot(122),plt.imshow(img_prewitty, cmap = 'gray'), plt.title('Sobel img_prewitty
plt.xticks([]), plt.yticks([])

plt.show()
```



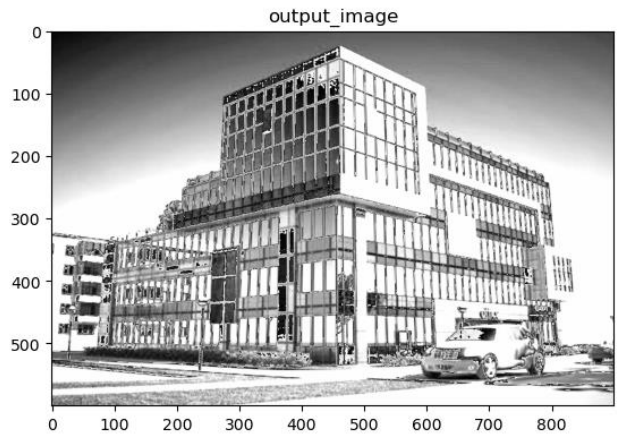
Задание 6.9.

Используя оператор Робертса, выделить линии на изображении.

In [129]:
```
kernel1 = np.array([[1, 0], [0, 1]])
kernel2 = np.array ([[0, 1],[0, 1]])
```

In [132]:
```
img_robx = cv2.filter2D(img, -1, kernel1)
img_roby = cv2.filter2D(img, -1, kernel2)

output_image = img_robx + img_roby
```

In [134]:
```
plt.imshow(output_image, cmap = 'gray'), plt.title('output_image')
plt.show()
```



Задание 6.10.

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

In [135]:
```
laplacian = cv2.Laplacian(img, cv2.CV_64F)
```

In [136]:
```
figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img, cmap = 'gray'), plt.title('Original')
plt.xticks([]), plt.yticks([])

plt.subplot(122),plt.imshow(laplacian, cmap = 'gray'), plt.title('Laplacian')
plt.xticks([]), plt.yticks([])

plt.show()
```

Самостоятельное задание

Создание файлов с изображениями и применение операторов обнаружения и выделения линий и перепадов изображения на этих изображениях с использованием операторов Собеля, Превитта, Робертса и Лапласа.

In [21]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

In [41]:
```
img = cv2.imread('img/build.jpeg', 0)
```

При решении задачи будем использвать операторы Собеля, Превитта и Робертса, для обнаружения и выделения линий, и оператор Лапласа для обнаружения и выделения перепадов изображения.

Применение оператора Собеля

In [42]:
```
sobel_x = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
sobel_y = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=5)
sobel_xy = cv2.bitwise_or(sobel_x, sobel_y)
```

Применение оператора Превитта

In [43]:
```
prewitt_x = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=3)
prewitt_y = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=3)
prewitt_xy = cv2.bitwise_or(prewitt_x, prewitt_y)
```

Применение оператора Робертса

In [52]:
```
roberts_cross = np.array([[1, 0], [0, -1]])

roberts_x = cv2.filter2D(img, -1, roberts_cross)
roberts_y = cv2.filter2D(np.flip(img, axis=1), -1, roberts_cross)
roberts_y = np.flip(roberts_y, axis=1)
roberts_xy = cv2.bitwise_or(roberts_x, roberts_y)
```

Создание изображения с перепадами изображения

In [48]:
```
img_grad = cv2.imread('img/build.jpeg', 0)
```

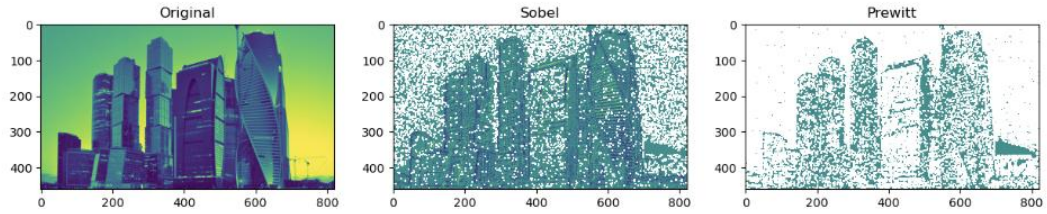
Создание изображения с перепадами изображения

In [48]:
```
img_grad = cv2.imread('img/build.jpeg', 0)
```

Применение оператора Лапласа для обнаружения и выделения перепадов изображения

In [49]:
```
laplacian = cv2.Laplacian(img_grad, cv2.CV_64F)
```

In [50]:
```
plt.figure(figsize=(15,15))
plt.subplot(231), plt.imshow(img), plt.title('Original')
plt.subplot(232), plt.imshow(sobel_xy), plt.title('Sobel')
plt.subplot(233), plt.imshow(prewitt_xy), plt.title('Prewitt')

plt.show();
```



In [51]:
```
plt.figure(figsize=(15,15))
plt.subplot(234), plt.imshow(roberts_xy), plt.title('Roberts')
plt.subplot(235), plt.imshow(img_grad, cmap='gray'), plt.title('Gradient')
plt.show();
```