

Дата: 12.06.23

ФИО: Козлов Евгений Юрьевич

Группа: 224-322

ПРАКТИЧЕСКАЯ РАБОТА №4

Применение пространственных фильтров размытия и повышения резкости

1. Цель работы

Познакомится с пространственными методами фильтрации.

2. Исходные данные и программное обеспечение

Используемая среда программирования: Visual Studio Code

Используемый язык программирования: Python 3.11.1 64-bit

Используемые библиотеки: numpy, scipy, skimage, matplotlib

3. Выполнение работы

Формирование функции периодического прямоугольного сигнала.

Периодический прямоугольный сигнал формируется из значений $\{0.0, 1.0\}$ замещается с помощью библиотеки NumPy.

Получение изображения, заданного функцией из п. 1

Изображение 200×200 px с глубиной цвета 32 bpp (float32). Период 25px, и ширина полосы 4px:

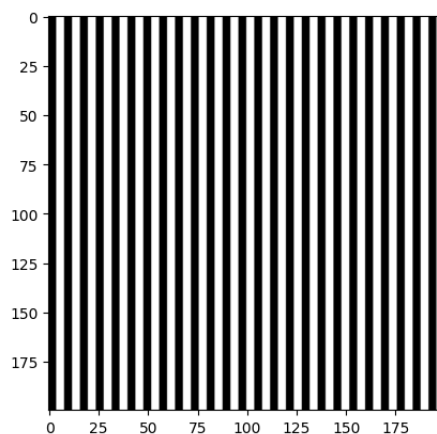
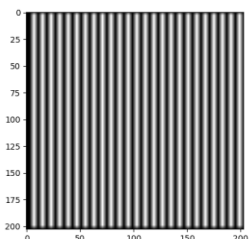
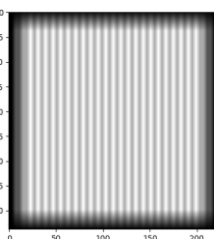
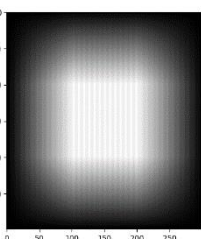


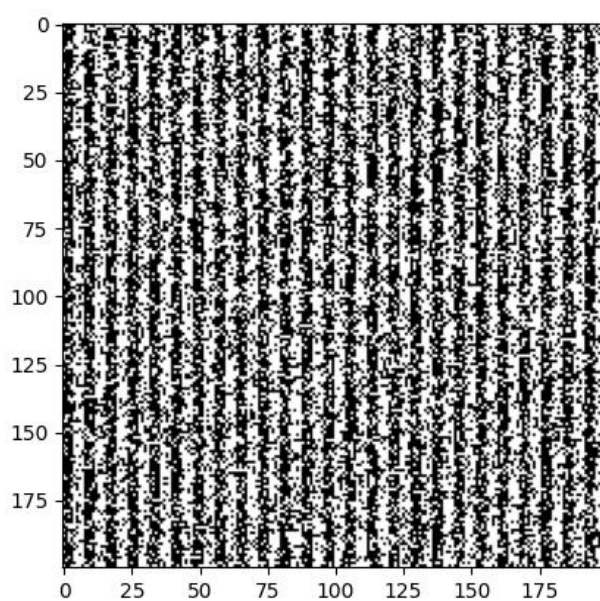
Рисунок 1 – исходное изображение

Применение к изображению линейных сглаживающих фильтров

Размер маски	4x4	20x20	100x100
Изображение после фильтрации			

Изображение после добавления импульсного шума

$P=.25$

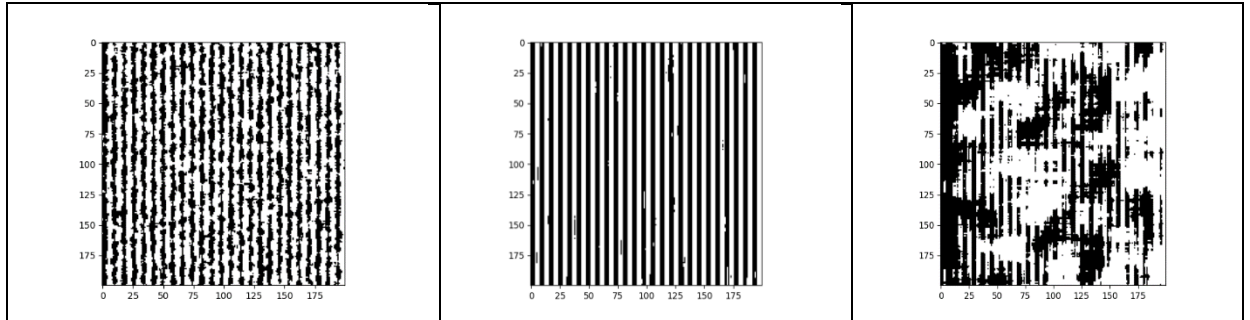


Параметры медианной фильтрации

Фильтрация реализована с помощью метода `median_filter` из библиотеки `scipy`

```
def median_fn(a,b):  
    return sp.ndimage.median_filter(NOISED, size=(a, b))
```

Изображения после медианной фильтрации



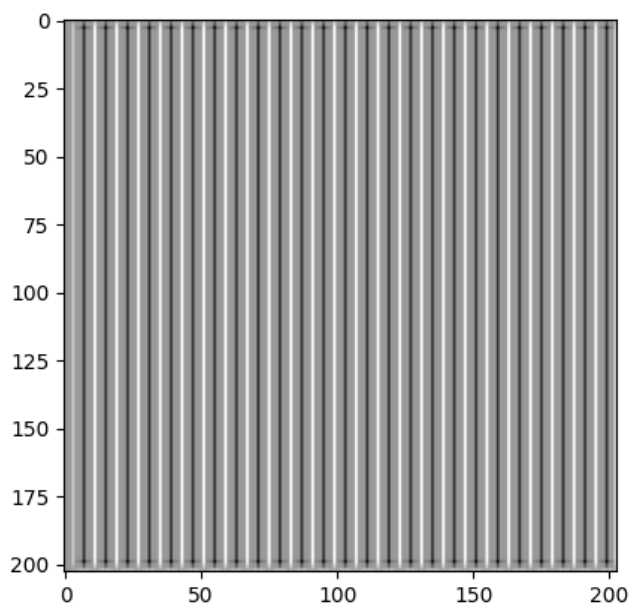
Параметры $(a, b) = (4, 4)$, $(25, 1)$ и $(25, 25)$ соответственно

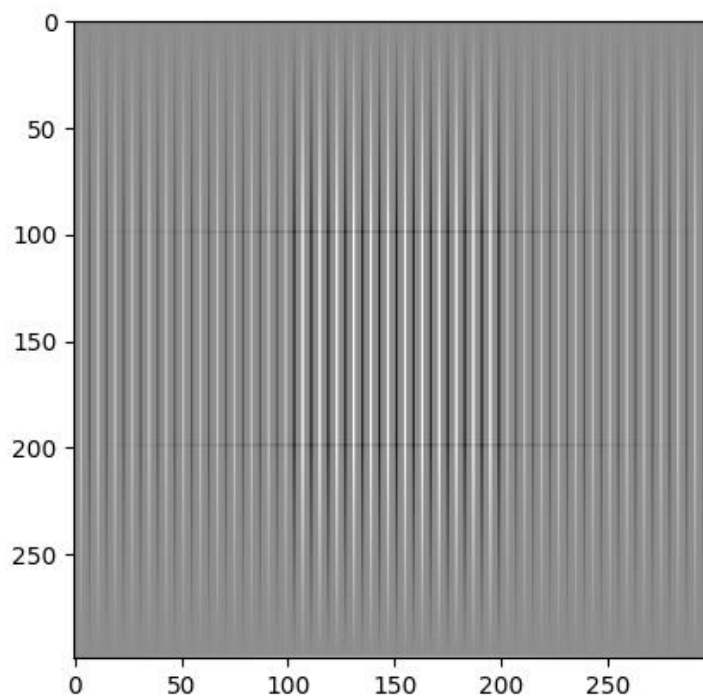
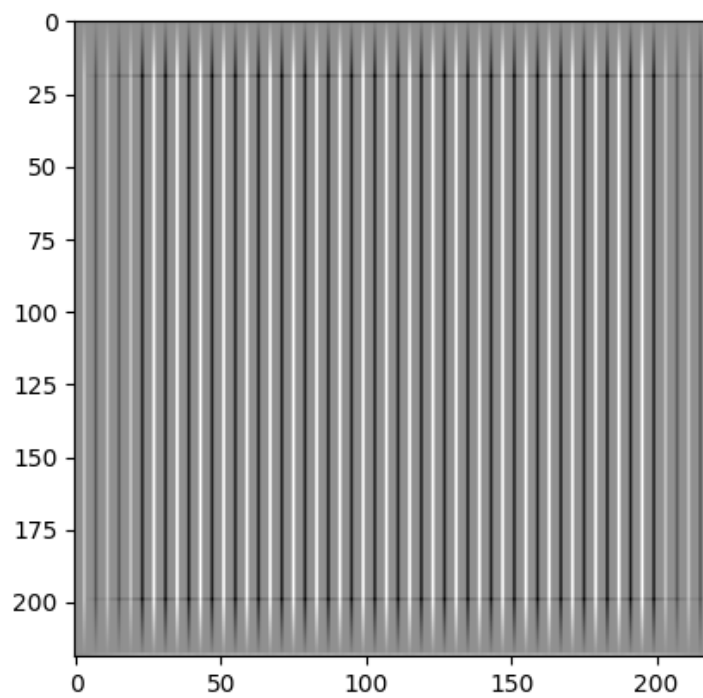
Параметры фильтра повышения резкости на примере лапласиана

Фильтрация реализована с помощью метода `laplace` из библиотеки `scipy`

```
LAPLAS_1 = sp.ndimage.laplace(BLURRED_1)
```

Изображения после повышения резкости





Все изображения хранятся на гугл-диске:

<https://drive.google.com/drive/folders/1Tb3JQ6p5sTJNme5eF--8MToBTefKQRKw?usp=sharing>

Код программы:

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

from skimage.io import imread, imshow, imsave
from skimage import data, img_as_float

# Формирование функции периодического прямоугольного сигнала.

def square_wave(period, w=200, h=200):
    freq = w // period
    tile = np.repeat((0.0, 1.0), freq // 2)
    line = np.tile(tile, period)
    fig = np.repeat(line, h).reshape(w, h).T
    return fig

# Получение изображения, заданного функцией из п. 1

def show_n_save_figure(fig, path='', cmap='gray'):
    plt.imshow(fig, cmap=cmap)
    plt.savefig(path)
    plt.show()
    plt.close()

INIT_FIGURE = square_wave(25)

# Применение к изображению линейных сглаживающих фильтров
# Воспользуемся методом для свёртки из библиотеки scipy

def blur(fig, x, y):
    return sp.signal.convolve2d(fig, np.ones((x, y)) / (x * y))

BLURRED_1 = blur(INIT_FIGURE, 4, 4)
BLURRED_2 = blur(INIT_FIGURE, 20, 20)
BLURRED_3 = blur(INIT_FIGURE, 100, 100)

# Изображения после размытия
show_n_save_figure(BLURRED_1, 'img/1/BLURRED_1.png')
show_n_save_figure(BLURRED_2, 'img/1/BLURRED_2.png')
show_n_save_figure(BLURRED_3, 'img/1/BLURRED_3.png')

# Изображение после добавления импульсного шума

def noise(fig, p):
```

```

mask = np.random.choice((0, 255), size=fig.shape, p=(1-p, p))
noised = fig.astype(np.uint8) * 255 ^ mask
return (noised // 255).astype(np.float32)

NOISED = noise(INIT_FIGURE, .25)
show_n_save_figure(NOISED, 'img/2/NOISED.png')

# Параметры медианной фильтрации

def median_fn(a,b):
    return sp.ndimage.median_filter(NOISED, size=(a, b))

MEDIAN_1 = median_fn(4, 4)
MEDIAN_2 = median_fn(25, 1)
MEDIAN_3 = median_fn(25, 25)

# Изображение после медианной фильтрации
show_n_save_figure(MEDIAN_1, 'img/2/MEDIAN_1.png')
show_n_save_figure(MEDIAN_2, 'img/2/MEDIAN_2.png')
show_n_save_figure(MEDIAN_3, 'img/2/MEDIAN_3.png')

# Параметры фильтра повышения резкости на примере лапласиана

LAPLAS_1 = sp.ndimage.laplace(BLURRED_1)
LAPLAS_2 = sp.ndimage.laplace(BLURRED_2)
LAPLAS_3 = sp.ndimage.laplace(BLURRED_3)

# Изображения после повышения резкости

show_n_save_figure(LAPLAS_1, 'img/3/LAPLAS_1.png')
show_n_save_figure(LAPLAS_2, 'img/3/LAPLAS_2.png')
show_n_save_figure(LAPLAS_3, 'img/3/LAPLAS_3.png')

```