МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ»
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)
Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806: «Вычислительная математика и программирование»

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №5

По курсу: «Цифровая обработка изображений»

Тема: «Цифровая обработка изображений в частотной области. Часть 1»

Студент: Чернышев Д.В.
Группа: М8О-107М-22
Вариант: 8
Преподаватель: Гаврилов К.Ю.

Москва
2023

СОДЕРЖАНИЕ

1 Задание к лабораторной работе №1

Часть 1

Исходное изображение представлено в виде двумерной матрицы, записанной в формате Matlab в файле $A5\_08\_1.mat$. Постройте заданное изображение в пространственной области, вычислите и отобразите на экране его амплитудный и фазовый спектр. Используя свойство преобразования Фурье о сдвиге, выполните сдвиг изображения на 120 пикселей вправо и 10 пикселей вниз путем преобразования спектра в частотной области. Вычислите амплитудный и фазовый спектр сдвинутого изображения. Отобразите на экране сдвинутое изображение и его спектр. Проведите анализ и сравнение спектров исходного и сдвинутого изображений. Объясните полученные результаты. Рекомендация. При построении амплитудных спектров изображений целесообразно использовать логарифмическое преобразование яркости.

Сохраните все построенные изображения в формате *jpg* для представления в отчете с комментариями.

Часть 2

Используя в качестве исходного изображения матрицу файла $A5\_08\_1.mat$ из п. 1, постройте центрированный спектр изображения, в котором нулевая частота располагается в центре частотной области. Выполните построение центрированного спектра первым способом – путем преобразования исходного изображения в пространственной области. Сравните два спектра – центрированный и нецентрированный. Выполните построение центрированного спектра вторым способом – путем преобразований полученного в частотной области. Сравните изображения центрированных амплитудных спектров, полученных двумя способами. С помощью обратного преобразования Фурье постройте изображение в пространственной области, соответствующее центрированному спектру изображения в частотной области. Сравните его с исходным изображением.

Сохраните все построенные изображения в формате *jpg* для представления в отчете с комментариями.

Часть 3

Дано текстовое изображение, представленное файлом *A5_08_3.jpg*. Выполните с этим изображением следующие операции:

Рис. 1: Исходное изображение

1. Вычислите спектр изображения и на экране отобразите его абсолютное значение в центрированном виде. 2. Выполните расфокусировку изображения с помощью фильтра нижних частот (ФНЧ), в качестве которого используйте фильтр Баттерворта второго порядка с частотой среза на уровне 50 пикселей. 3. Для расфокусированного изображения с целью повышения резкости выполните фильтрацию с помощью Лапласиана в частотной области. Частотную характеристику фильтра Лапласиана постройте путем двумерного ДПФ от маски Лапласиана

3x3 в пространственной области. Рекомендация. С целью улучшения эффективности повышения резкости изображения при фильтрации используйте двух- или трехкратное последовательное применение фильтра Лапласиана. Конечный результат фильтрации для улучшения контраста изображения подвергните нелинейному преобразованию с помощью гамма-коррекции или логарифмического преобразования яркости в комбинации с эквализацией гистограммы. 4. С целью выделения контуров объектов на изображении выполните фильтрацию исходного изображения с помощью фильтра верхних частот (ФВЧ). В качестве ФВЧ используйте гауссов фильтр. Рекомендация. Частоту среза ФВЧ выберите около 200 пикселей.

Сохраните все построенные изображения в формате *jpg* для представления в отчете с комментариями.

2 Выполнение лабораторной работы №5

Подгрузка библиотек

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

import cv2
from scipy.fft import fft, fftshift, fft2
from scipy.fft import fftfreq,rfft
from scipy.fft import rfftfreq,
                        irfft,
                        ifft,
                        ifftshift,
                        ifft2,
                        dct

from scipy import signal
from scipy import ndimage


from PIL import Image,
                ImageDraw,
                ImageStat,
                ImageOps
import cv2
import bisect
from numba import jit
import IPython.display #import Image
import os
import math
```

Часть 1

Объявляем метод логарифмического преобразования к изображению

```
def logTransformImage(image:Image, c_i=255):
    print(f"
            np.max(np.array(image))
            {np.max(np.array(image))}"
            )
    c = c_i / np.log(1 + np.max(np.array(image)))
    print(f"c={c}")
    log_image_arr = c * (np.log(np.array(image) + 1))
    print(f"log_image={np.max(log_image_arr)}")
    return  Image.fromarray(
                log_image_arr.astype('uint8'),
                'RGB'
                )
```

```
MAT = MAT_RAW_FIX.reshape(*MAT_RAW_FIX.shape).astype(
                                        np.uint8
                                        )

MAT = np.stack((MAT,) * 3, axis=-1)
image_raw = Image.fromarray(MAT)
image_raw.save(os.path.join(OUTPUT,"raw_image.png"))
%matplotlib inline
plt.imshow(image_raw)
```

```
# загрузка изображения в оттенках серого
img = cv2.imread(os.path.join(OUTPUT,"raw_image.png"), 0)

# применение двумерного БПФ (Быстрое преобразование Фурье)
f = np.fft.fft2(img)

# сдвиг низких частот к центру спектра
fshift = np.fft.fftshift(f)

# вычисление амплитудного спектра
magnitude_spectrum = 20*np.log(np.abs(fshift))

# вычисление фазового спектра
phase_spectrum = np.angle(fshift)

plt.subplot(131),plt.imshow(img, cmap='gray')
plt.title(
        'Input Image'
        ),plt.xticks([]),plt.yticks([])

plt.subplot(132),plt.imshow(magnitude_spectrum, cmap='gray')

plt.title(
    'Magnitude Spectrum'
    ), plt.xticks([]), plt.yticks([])
plt.subplot(133),plt.imshow(phase_spectrum, cmap='gray')
plt.title('Phase Spectrum'), plt.xticks([]), plt.yticks([])
```

```
# Расчет быстрого преобразования Фурье
f = np.fft.fft2(img)

# Смещение нулевой частоты в центр
fshift = np.fft.fftshift(f)

# Определите расстояние для смещения вдоль осей x и y
dx = 120
dy = -10

# Вычислить сдвиг вдоль осей x и y
M, N = img.shape
shiftx, shifty = np.meshgrid(np.arange(N), np.arange(M))
shiftx = np.exp(-2j * np.pi * dx * shiftx / N)
shifty = np.exp(-2j * np.pi * dy * shifty / M)

# Сдвиг спектра для выполнения сдвига в пространственной области
fshift = fshift * shiftx * shifty

# Обратный сдвиг для визуализации результирующего изображения
f_ishift = np.fft.ifftshift(fshift)

# Вычислить обратное бпф смещенного спектра
img_back = np.fft.ifft2(f_ishift)

# Возьмите абсолютное значение комплексного результата, чтобы
# получить реальное изображение
img_back = np.abs(img_back)




# Отображение измененного изображения
```

```
33 plt.subplot(121),plt.imshow(img, cmap = 'gray')
34 plt.title('Исходное изображение'), plt.xticks([]), plt.yticks([])
35 plt.subplot(122),plt.imshow(img_back, cmap = 'gray')
36 plt.title('Сдвинутое изображение'), plt.xticks([]), plt.yticks([])
37 plt.show()
```

```
# применение двумерного БПФ (Быстрое преобразование Фурье)
f = np.fft.fft2(img_back)

# сдвиг низких частот к центру спектра
fshift = np.fft.fftshift(f)

# вычисление амплитудного спектра
magnitude_spectrum = 20*np.log(np.abs(fshift))

# вычисление фазового спектра
phase_spectrum = np.angle(fshift)

plt.subplot(131),plt.imshow(img_back, cmap='gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(magnitude_spectrum, cmap='gray')
plt.title('Magnitude Spectrum'), plt.xticks([]), plt.yticks([])
plt.subplot(133),plt.imshow(phase_spectrum, cmap='gray')
plt.title('Phase Spectrum'), plt.xticks([]), plt.yticks([])
```

Часть 2

Создадим вспомогательную функцию построения графиков

```
def make_subplots(args:list,titles:list ):
    assert len(args) == len(titles)

    fig = plt.figure(figsize=(10,10))
    for i in range(len(args)):
        ax_i = fig.add_subplot(1,len(args),i+1)
        ax_i.imshow(args[i], cmap="Spectral")
        ax_i.title.set_text(titles[i])
    plt.show()
```

Построение центрированного спектра в пространственной области с использованием ядра Гаусса:

```
# Загрузка изображения
img  = Image.fromarray(MAT_RAW_FIX)

filtered_gausss_img = ndimage.gaussian_filter(img, sigma=0.75)

# Построение центрированного спектра
magnitude_spectrum =  np.abs((filtered_gausss_img))

make_subplots(
    [
        magnitude_spectrum,
        20*np.log(abs(fft2(filtered_gausss_img))),
        20*np.log(abs(fft2(img))),
        img], # plots
    [
        "Gauss Magnitude Spectrum",
        "Gauss  Spectrum",
        "input image spectrum",
        "Input Image"] # titles
)
```

Второй способ преобразования: Преобразование Фурье

Преобразование Фурье преобразует изображение из пространственной области в частотную область, строя спектр изображения.

```
img  = Image.fromarray(MAT_RAW_FIX)

f = fft2(img)
f_shift = fftshift(np.abs(f))

#
```

```
make_subplots(
    [
        20 * np.log(fftshift(np.abs(fft2(img)))),
        20*np.log(np.abs(fft2(img))),img], # plots
    [
        "Centered Spectrum (Frequency Domain)",
        "Input Image Spectrum","Input Image"] # titles
)
```

С помощью обратного преобразования Фурье постройте изображение в пространственной области, соответствующее центрированному спектру изображения в частотной области. Сравните его с исходным изображением.

```
# Создаем полосовой фильтр для удаления шума
rows, cols = img.size
crow, ccol = rows//2, cols//2
mask = np.zeros((rows, cols), np.uint8)
v = 30
mask[crow-v:crow+v, ccol-v:ccol+v] = 1

f_shift = fshift * mask
#20*np.log(abs(fft2(filtered_gausss_img)))
f_ishift = ifftshift(f_shift)
img_back = np.abs(ifft2(f_ishift))

make_subplots(
    [
        img,20*np.log(abs(fft2(img))),
        img_back,20*np.log(abs(fft2(img_back)))
        ], # plots
    [
        "Input Image","Input Spectrum",
        f"IFFT2 Image",f"IFFT2 Spectrum (mask v={v})"
```

```
        ] # titles
)
```

Часть 3

```
# Aply fft2 to image
f = fft2(img_source)
# Shift
f_shift = fftshift(np.abs(f))

make_subplots(
    [
        img_source,20 * np.log(f_shift)
    ],

    [
        "A5_08_3.jpg" ,"Spectrum Centered"
    ]
)
```

Для выполнения п.2 необходима функция ‘*scipy.signal.butter*‘ и оптимизурем её:

```
def ButterworthLowpass(cutoff, fs, order=2):
    nyquist = 0.5 * fs
    normal_cutoff = cutoff / nyquist
    b, a = signal.butter(
        order,
        normal_cutoff,
        btype='low',
        analog=False)
    return b, a

```

```
def ApplyButter(image, N, W):
    #
    gray  = ImageOps.grayscale(image)
    f = fft2(gray)
    # Shift
    f_shift = fftshift((f))
    # Определение размера изображения и центра
    rows, cols = gray.size
    crow, ccol = rows // 2, cols // 2

    # Создание ФНЧ-фильтра Баттерворта
    b, a = ButterworthLowpass(order=N,cutoff=W,fs=cols)

    # Применение фильтра к спектру изображения
    fshift_filtered = signal.lfilter(b, a, f_shift)

    # Обратное сдвигание
    f_filtered = ifftshift(fshift_filtered)
    img_filtered = ifft2(f_filtered)
    img_filtered = np.abs(img_filtered)

    # Приведение результатов к диапазону от 0 до 255
    img_filtered = img_filtered / np.max(img_filtered) * 255
    img_filtered = img_filtered.astype(np.uint8)

    return img_filtered
```

```
W = 50  # Частота среза фильтра в пикселях
N  = 2 # порядок
filtered_image = ApplyButter(img_source, N,W)
make_subplots(
    [
        img_source,
```

```
        filtered_image
    ],

    [
        "A5_08_3.jpg" ,
        f"Filtered A5_08_3.jpg (freq={W},Order={N})"
    ]
)
```

```
iterations = 3
lapl_img = filtered_image
# # Вариант 1
ksize = 5
depth = 3
scale = 1
delta = 0
# Применение фильтра Лапласиана
for i in range(iterations):
    lapl_img = cv2.Laplacian(
        lapl_img,
        depth, ksize=ksize,
        scale=scale, delta=delta,
        borderType=cv2.BORDER_DEFAULT
        )

lapl_img_fft2 = fft2(lapl_img)

# Преобразование изображения в частотную область
image_fft = fft2(filtered_image)

# Применение фильтра Лапласиана в частотной области
filtered_fft = image_fft - lapl_img_fft2

```

```
# Обратное преобразование из частотной области
# в пространственную
filtered_img = np.real(ifft2(filtered_fft))

# Повышение контраста изображения с помощью гамма-коррекции
# и эквализации гистограммы
c = 255 / np.log(1 + np.max(filtered_img))
log_image = c * (np.log(filtered_img + 1))
log_image = np.array(log_image, dtype = np.uint8)


enhanced_image = cv2.equalizeHist(log_image.astype(np.uint8))

make_subplots(
    [
        # img_source,
        filtered_image,
        enhanced_image
        ],
    [
        # "A5_08_3.jpg" ,
        f"Filtered(f={W},Or={N})",
        f"Lapl(it={iterations})_Eq_Log"
        ]
)
```

```
# Примените фильтр Гаусса к исходному изображению:
ksize = 0
delta = 200
filtered_image = cv2.GaussianBlur(
    enhanced_image,
    (ksize,ksize),
    delta
```

```
    )

# Вычтите от исходного изображения фильтрованное изображение:
filtered_image = np.uint8(filtered_image)
high_pass_filtered = cv2.subtract(
    enhanced_image,
    filtered_image
    )

# Получите контуры объектов на фильтрованном изображении:
contours, _ = cv2.findContours(
    high_pass_filtered,
    cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE
    )

# Создайте копию исходного изображения
image_contours = enhanced_image.copy()

# Рисование контуров на копии изображения
cv2.drawContours(
            image_contours,
            contours,
            -1,
            (0, 255, 0),
            2
        )

make_subplots(
    [
        enhanced_image,
        image_contours
        ],
```

```
41      [
42          f"Lapl(it={iterations})_Eq_Log",
43          "Countours"
44          ]
45  )
```

3 Результаты выполнения лабораторной работы №5

В результате выполнения работы были изучены быстрое преобразование Фурье, смещение а также обратные операции. Были зашумлены, а затем обработанны фотографии.

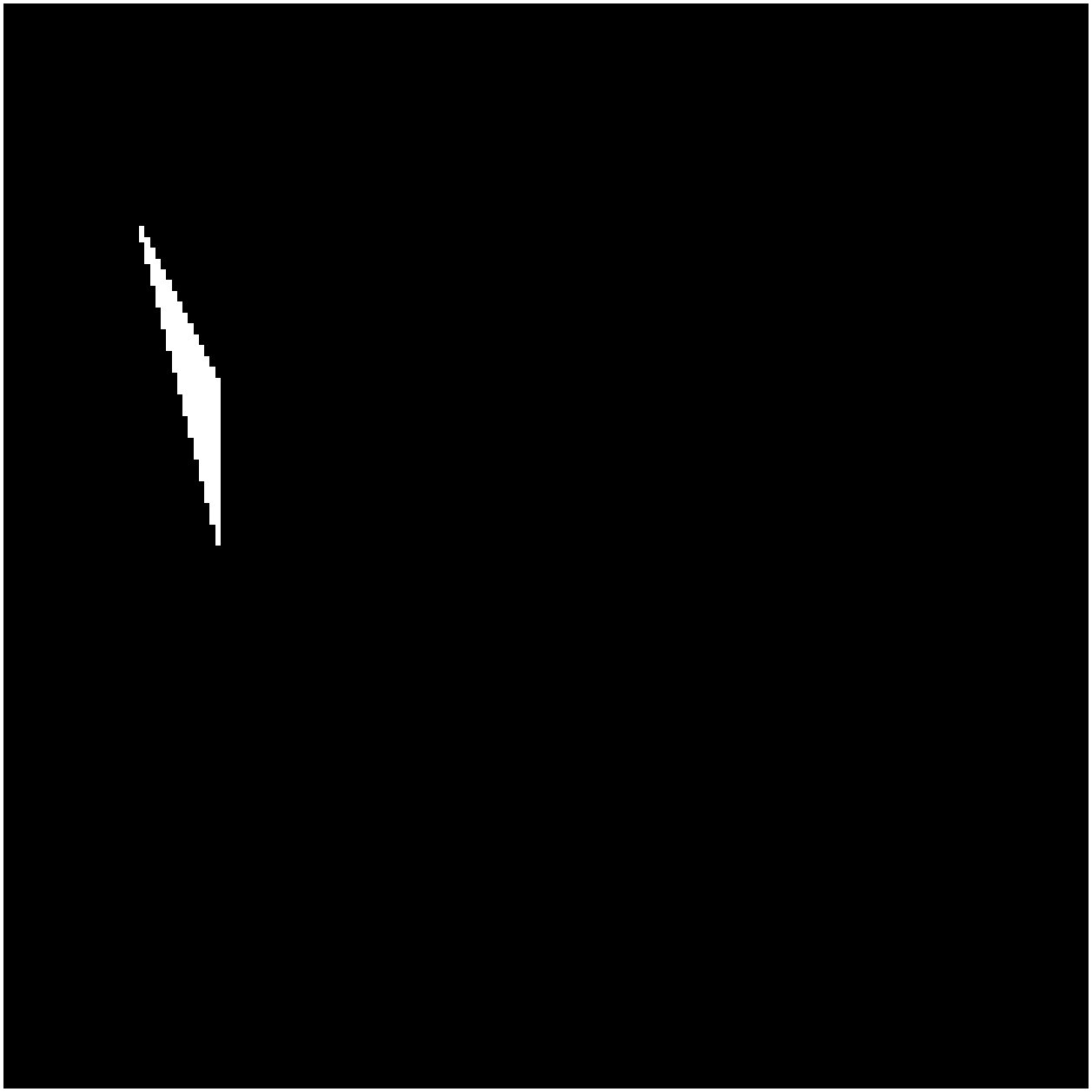

Рис. 2: Изображение полученное из матрицы

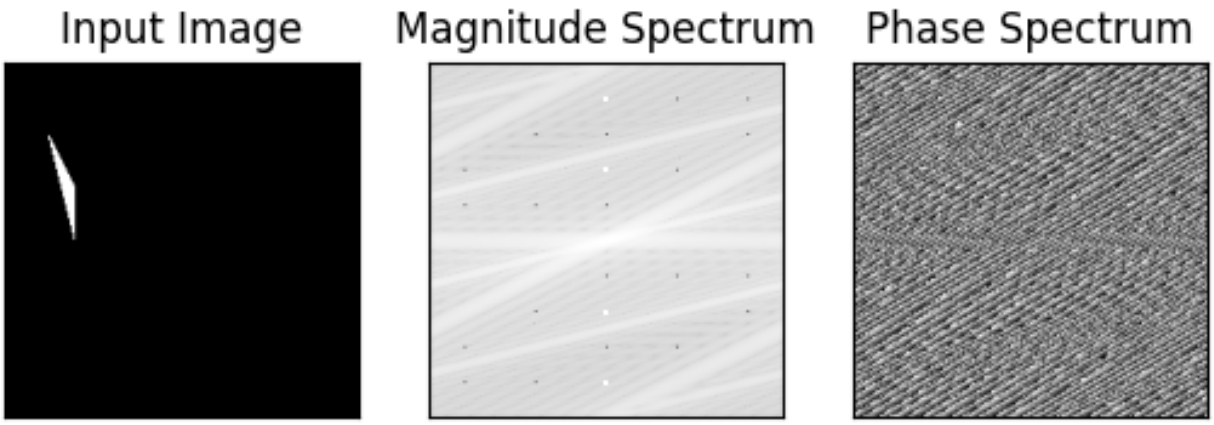

Рис. 3: Магнитуда и спектр исходного изображения

Рис. 4: Старое и новое изображение

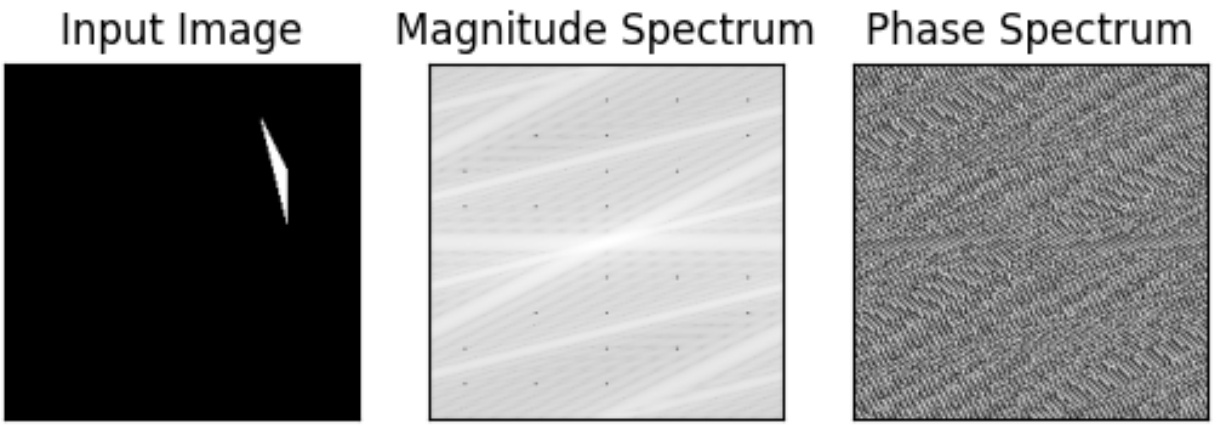

Рис. 5: Магнитуда и спектр нового изображения

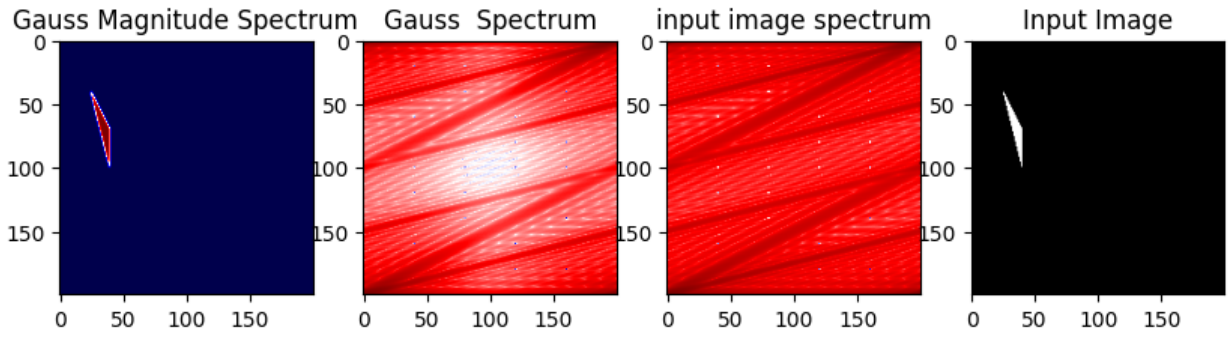

Рис. 6: Примениен Гауссовского ядра в пространственной области

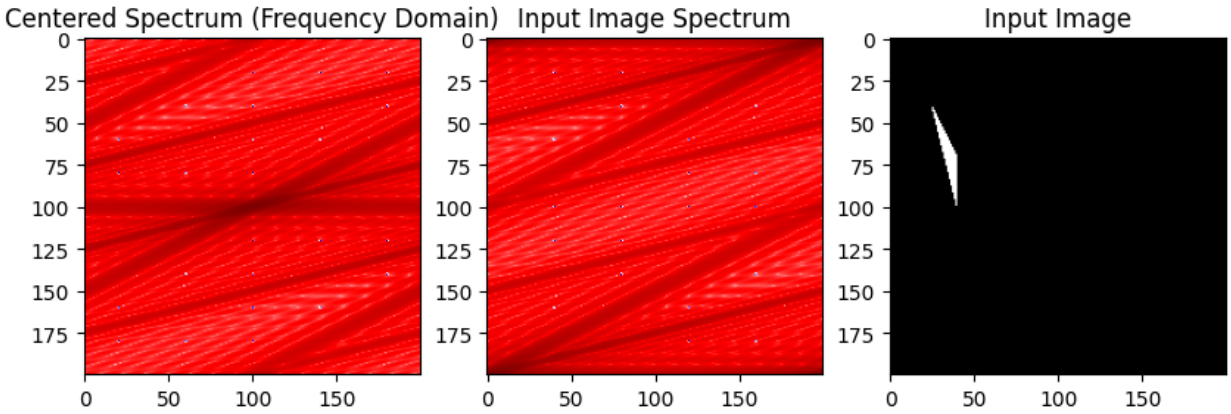

Рис. 7: Операция $FFTShift$

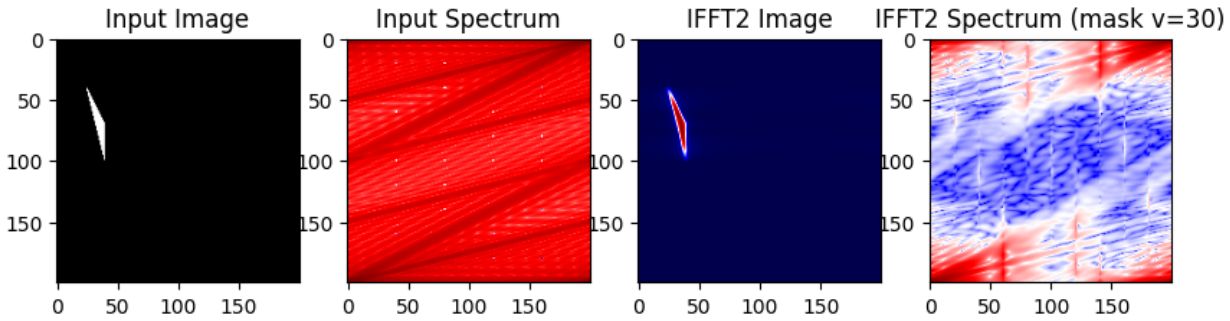

Рис. 8: Результаты обратного преобразования

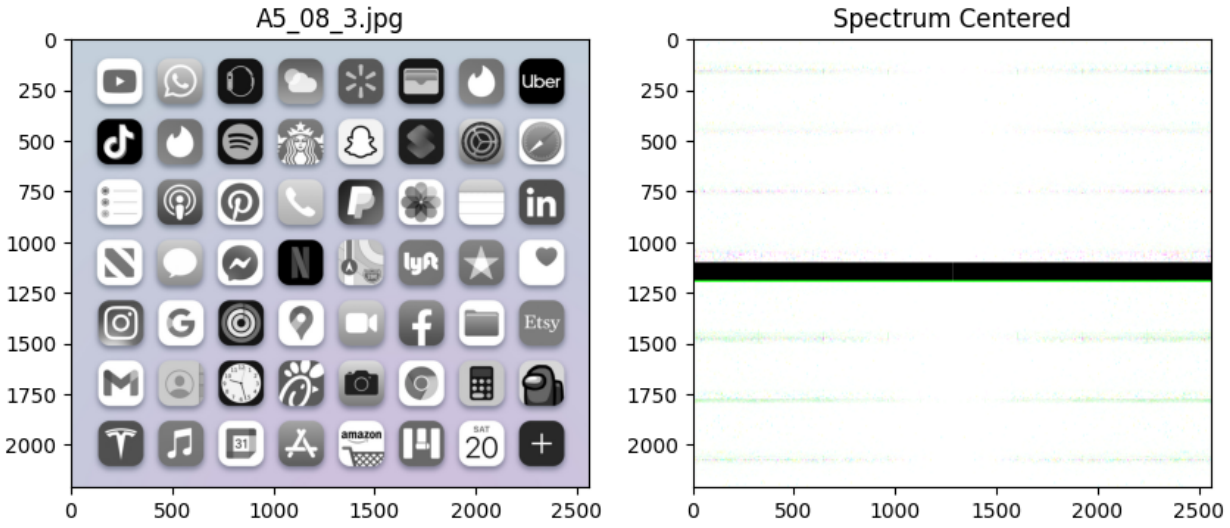

Рис. 9: Исходное изображение

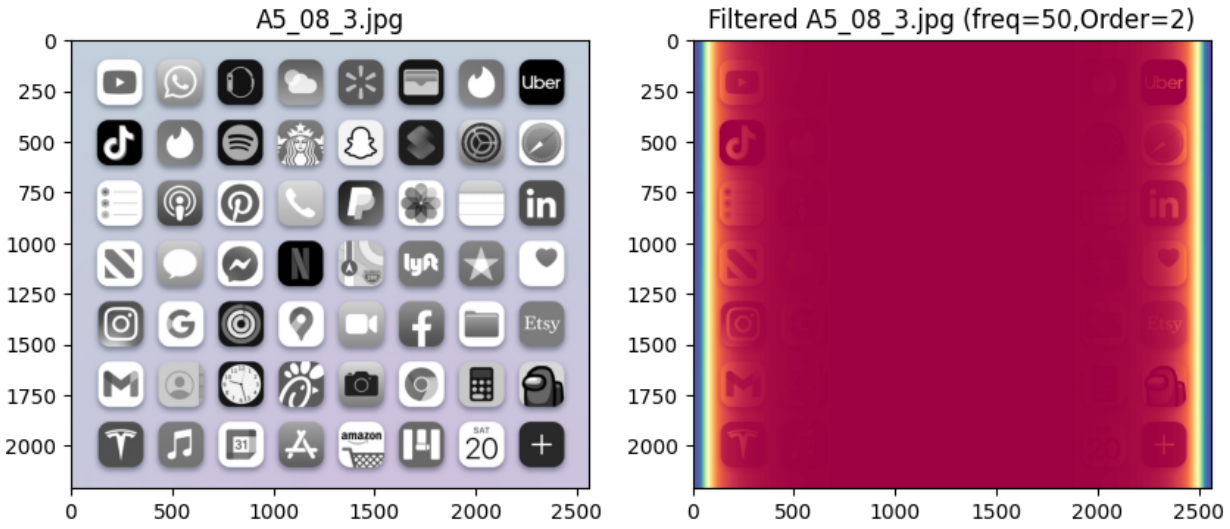

Рис. 10: Применение фильтра Баттерворта

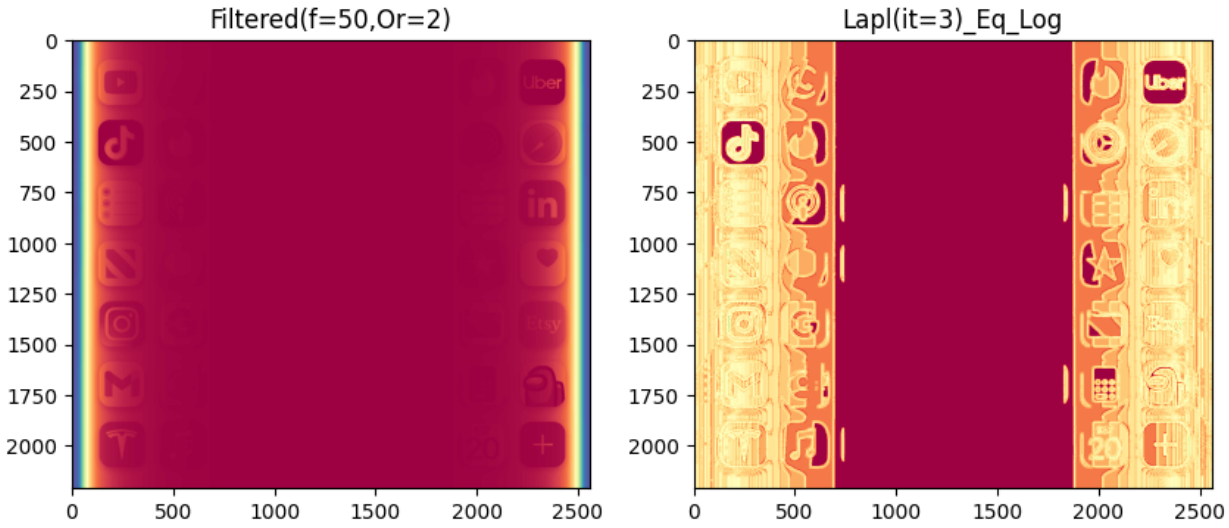

Рис. 11: Применение Лапласиана, еквализации и логорифмического преобразования

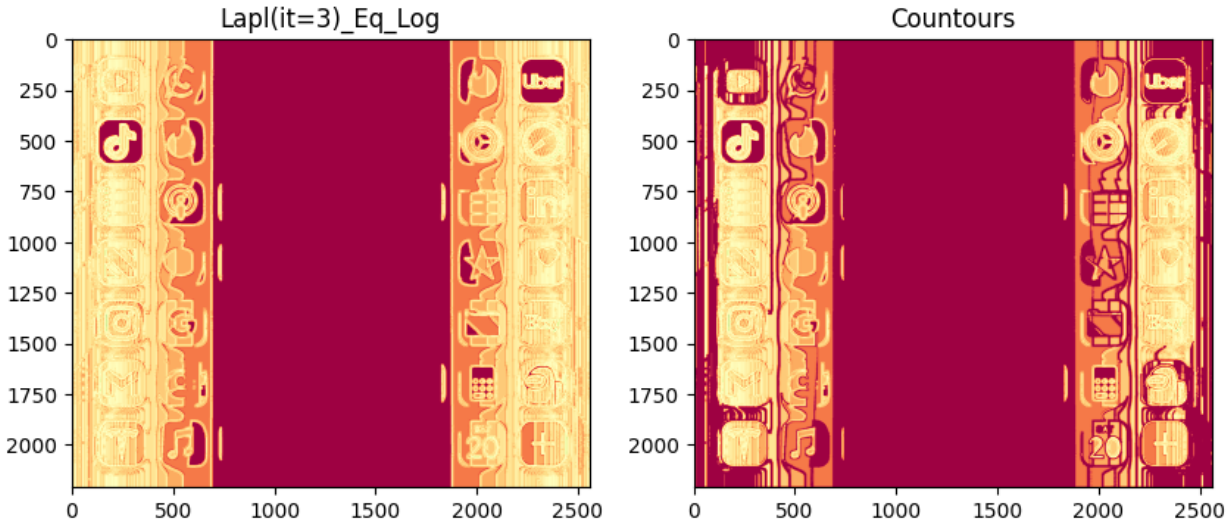

Рис. 12: Выделение контуров