МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Систем автоматического управления

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Техническое зрение»

Тема: «Сглаживание изображений»

Студент гр. 1492 Старцев Н.А.

Преподаватель Федоркова А.О.

Санкт-Петербург

2024

Цель работы: научиться применять фильтры для сглаживания изображений

Задание 1

Для выполнения этого задания используйте изображения 3-1 и 3-2. Примените к ним все известные фильтры для сглаживания изображений. Определите, какие из них больше всего подходят для того, чтобы максимально убрать все неровности на изображениях. Попробуйте применить фильтры как к цветному изображению, так и к изображению в оттенках серого.

Задание 2

Отфильтровать файл 3-3 так, чтобы можно было распознать содержимое изображения.

Задание 3

Применить к изображению 3-4 все возможные рамки, проанализировать их особенности и различия.

.

Примеры работы программы:

Рисунок 1 – Blur фильтр

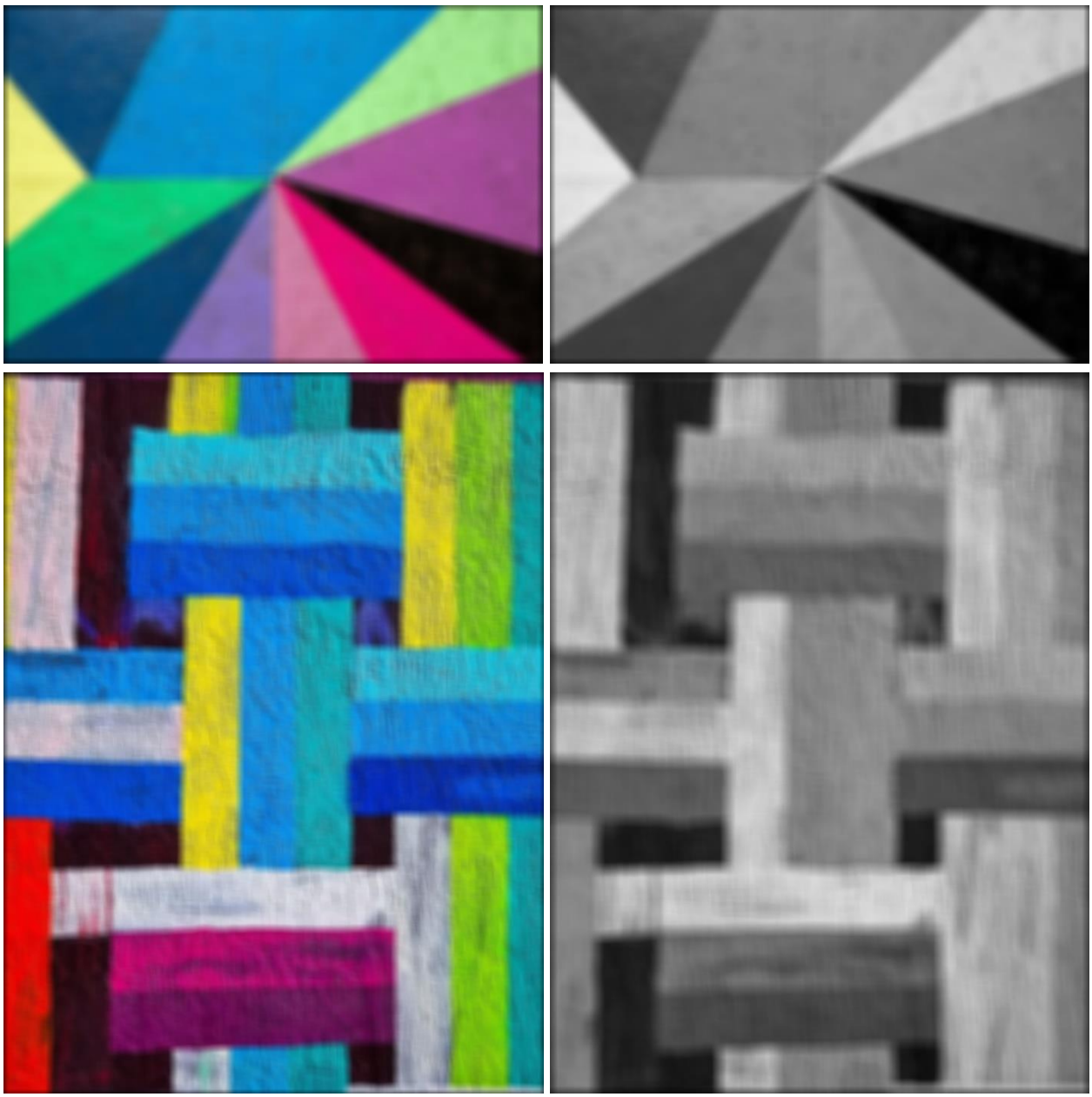

Рисунок 2 – boxFilter

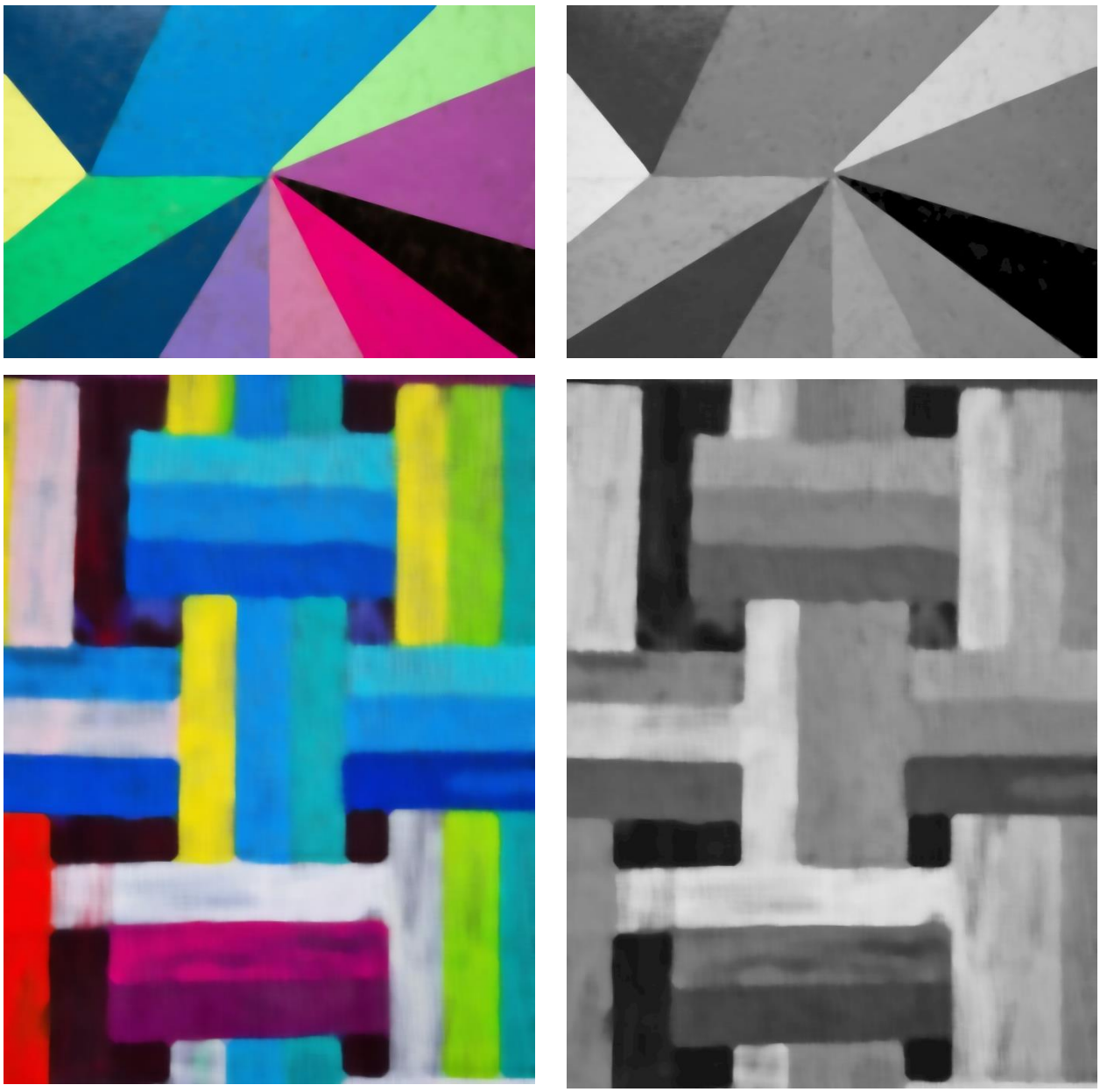

Рисунок 3 – medianBlur

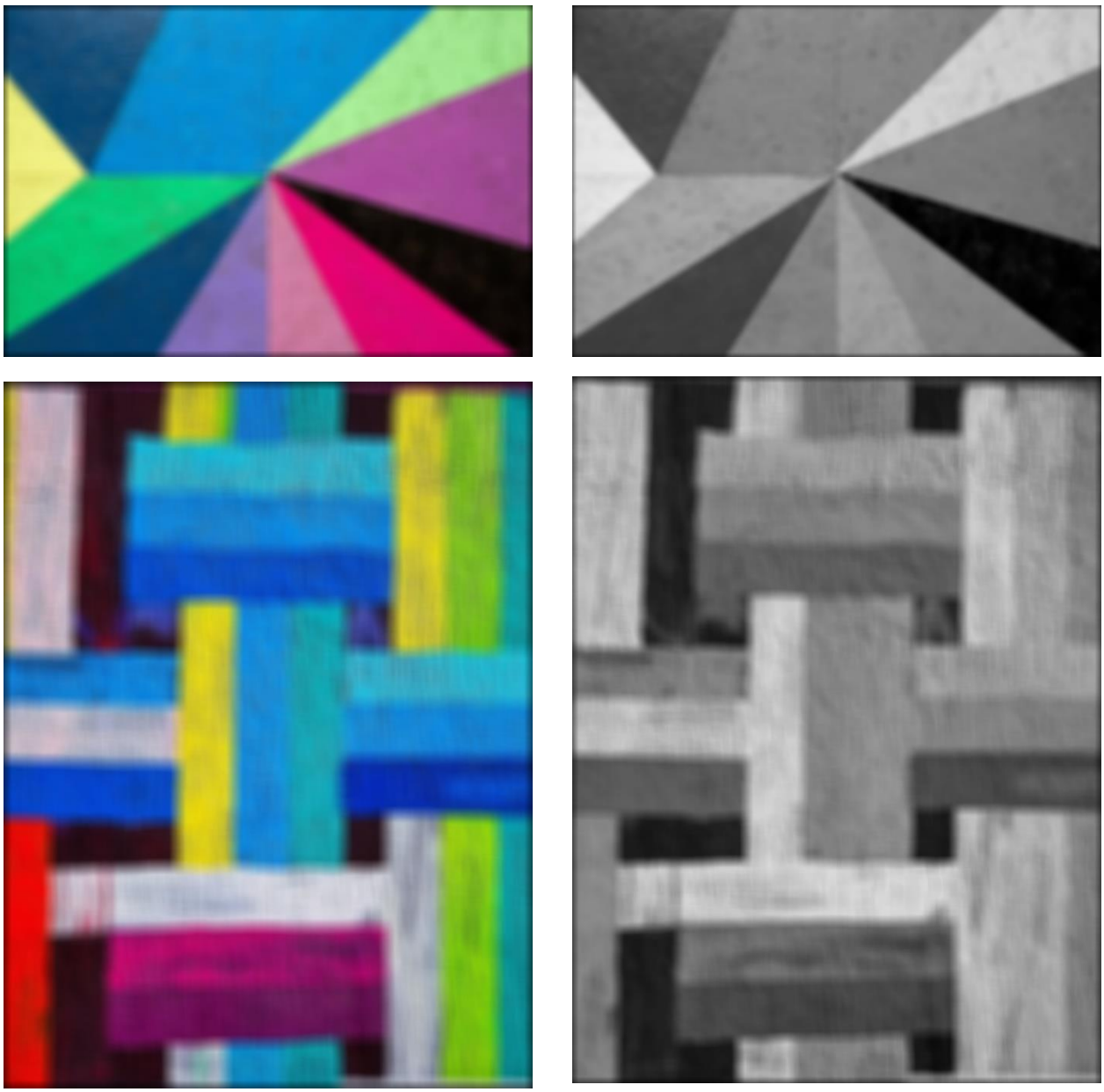

Рисунок 4 – GaussianBlur

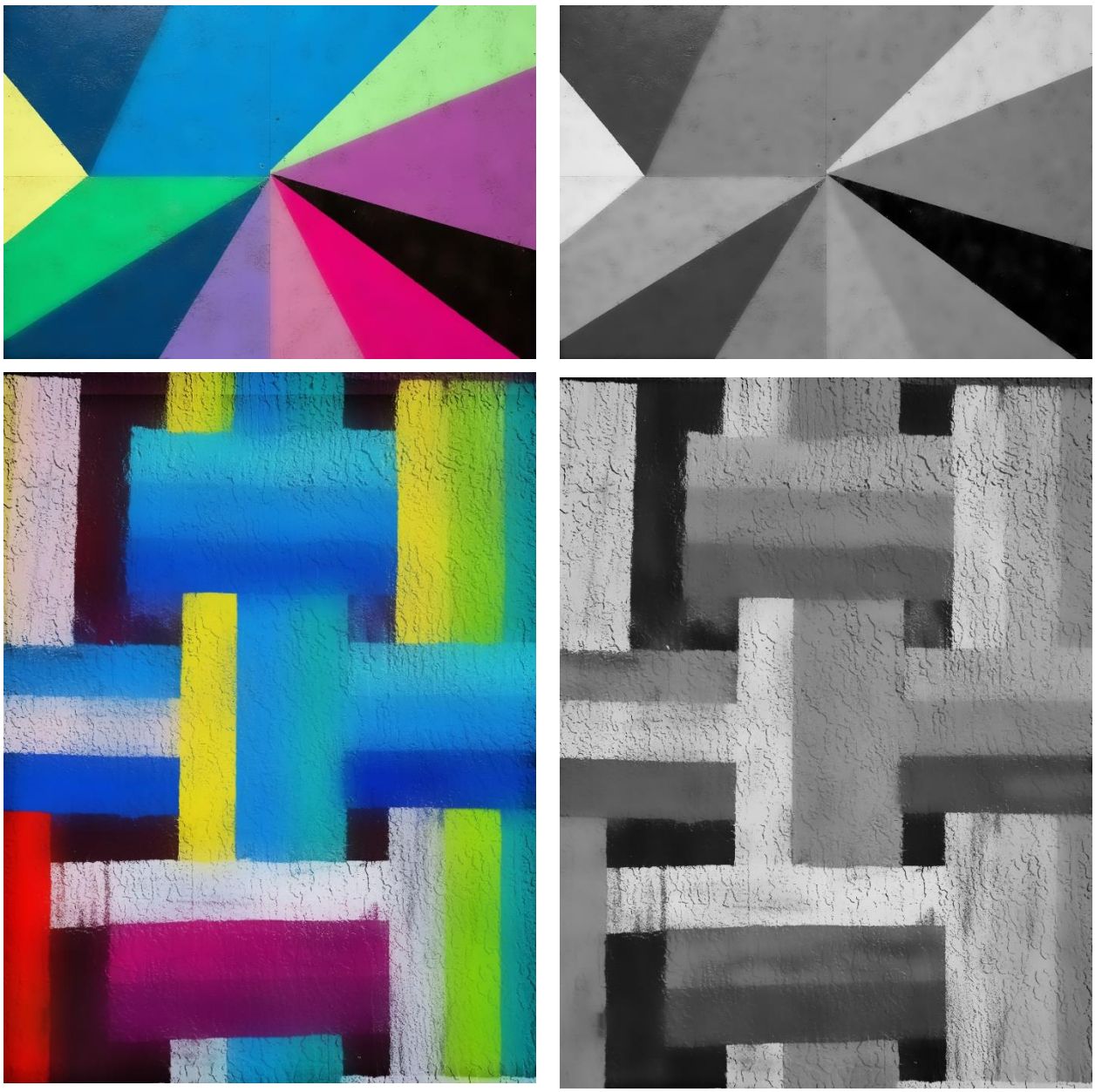

Рисунок 5 - bilateralFilter

.

Задание 2

Рисунок 6 – распознанное изображение

Задание 3

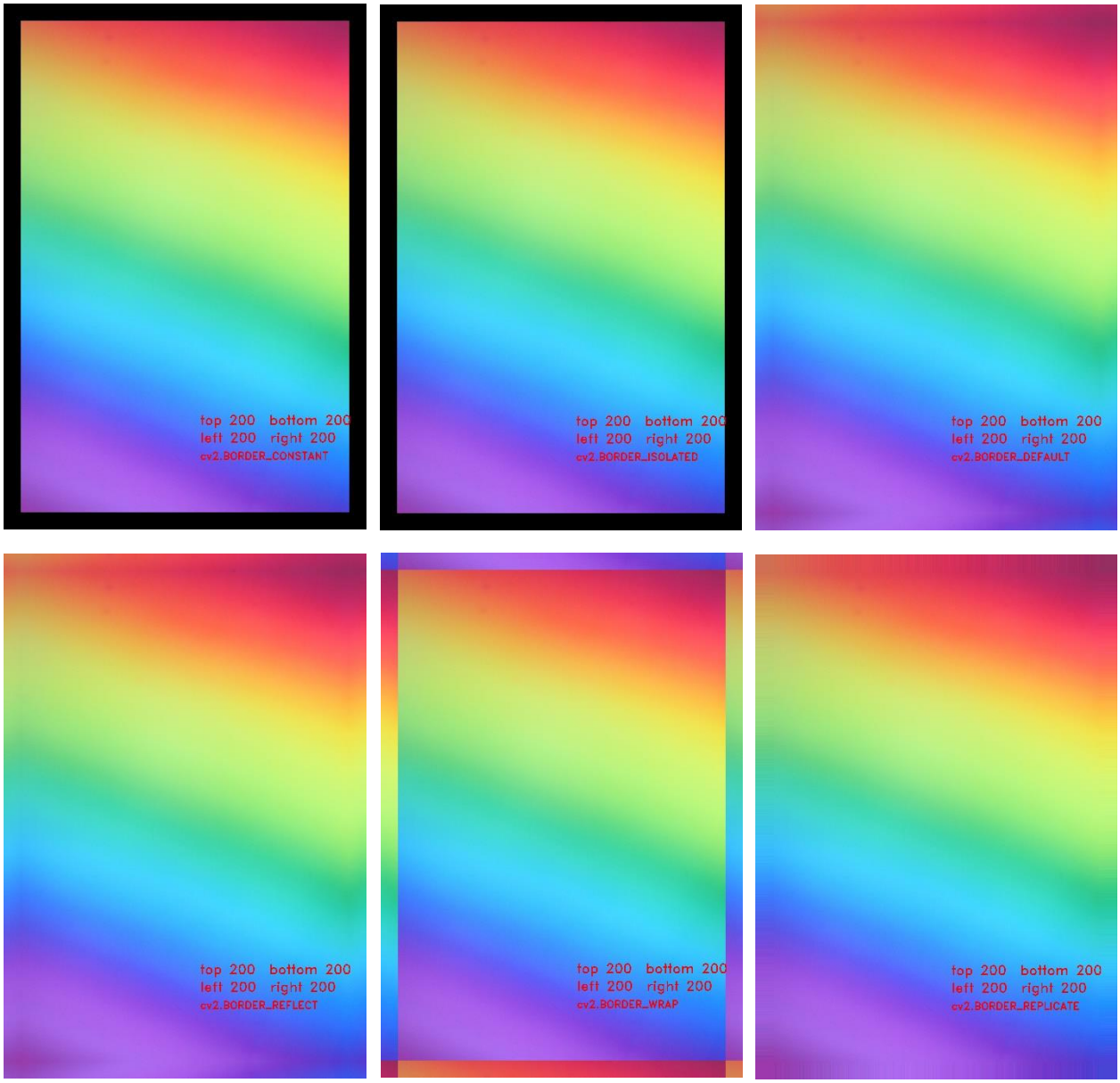

Рисунок 7 – пример использования рамок на изображение

Все рамки добавляют границы изображению с заданной толщиной, цвет добавляемых пикселей зависит от выбранного типа рамки, есть рамки добавляющие постоянный заданный цвет, добавляют часть отраженного изображения, растягивают границу исходного изображения с сохранением цвета. Рамки позволяют избежать ошибки при обработке фильтрами, которые рассматривают каждый пиксель как центр квадрата.

Вывод.

В ходе лабораторной работы были изучены различные виды фильтров, позволяющие убрать шум и мелкие детали мешающие дальнейшей обработке изображения.

Результаты работы

Код программы:

Задание 1.

```
import cv2
import numpy

def nothing(x):
    pass

pathImg1 = "./lab3/3-1.PNG"
pathImg2 = "./lab3/3-2.PNG"
winName = "Border"

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)
img1 = cv2.imread(
    pathImg2, flags=cv2.IMREAD_COLOR)
img1 = cv2.imread(
    pathImg2, flags=cv2.IMREAD_GRAYSCALE)

height, weight = img1.shape[0:2]

All_border = [cv2.BORDER_CONSTANT, cv2.BORDER_DEFAULT, cv2.BORDER_ISOLATED, cv2.BORDER_REFLECT,
              cv2.BORDER_REFLECT101, cv2.BORDER_REFLECT_101, cv2.BORDER_REPLICATE,  cv2.BORDER_WRAP]
All_border_name = ["cv2.BORDER_CONSTANT", "cv2.BORDER_DEFAULT", "cv2.BORDER_ISOLATED",
"cv2.BORDER_REFLECT",
              "cv2.BORDER_REFLECT101", "cv2.BORDER_REFLECT_101", "cv2.BORDER_REPLICATE",
"cv2.BORDER_WRAP"]

winName = "Blur"

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("size_H", winName, 0, 200, nothing)
cv2.createTrackbar("size_W", winName, 0, 200, nothing)
cv2.createTrackbar("anchor_H", winName, 0, 200, nothing)
cv2.createTrackbar("anchor_W", winName, 0, 200, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-2, nothing)

while (1):
    sizeH = cv2.getTrackbarPos("size_H", winName)
    sizeW = cv2.getTrackbarPos("size_W", winName)
    anchorH = cv2.getTrackbarPos("anchor_H", winName)
    anchorW = cv2.getTrackbarPos("anchor_W", winName)
    border = cv2.getTrackbarPos("border", winName)
    if anchorH > sizeH-1:
        anchorH = sizeH
    if anchorW > sizeW-1:
        anchorW = sizeW
    img_new = cv2.blur(
        img1,
        (sizeW+1, sizeH+1),  # размер ядра
        anchor=(anchorW-1, anchorH-1),  # положение якорной точки
        borderType=All_border[border],  # тип рамки
    )
    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break

cv2.destroyWindow(winName)
winName = "BoxFilter"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("ddepth", winName, 0, 200, nothing)
cv2.createTrackbar("size", winName, 0, 200, nothing)
cv2.createTrackbar("normalize", winName, 0, 1, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-2, nothing)

while (1):
    ddepth = cv2.getTrackbarPos("ddepth", winName)
    size = cv2.getTrackbarPos("size", winName)+2
    f = cv2.getTrackbarPos("normalize", winName)
    border = cv2.getTrackbarPos("border", winName)
```

```
    img_new = cv2.boxFilter(
    img1,  # входное изображение
    -1,  # глубина изображения-результата
    (size,size),  # размер ядра
    (size//2,size//2),  # положение якорной точки
    normalize=f,  # нормирование (если True)
    borderType=All_border[border],   # тип рамки
)
    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break

cv2.destroyWindow(winName)
winName = "medianBlur"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("size", winName, 0, 100, nothing)

while (1):
    size = cv2.getTrackbarPos("size", winName)


    img_new = cv2.medianBlur(
    img1,  # входное изображение
    size*2+1  # размер ядра
)

    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break

cv2.destroyWindow(winName)
winName = "GaussianBlur"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("size", winName, 0, 100, nothing)
cv2.createTrackbar("sigmaX", winName, 0, 200, nothing)
cv2.createTrackbar("sigmaY", winName, 0, 200, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-1, nothing)

while (1):
    size = cv2.getTrackbarPos("size", winName)
    sigmaX = cv2.getTrackbarPos("sigmaX", winName)
    sigmaY = cv2.getTrackbarPos("sigmaY", winName)
    border = cv2.getTrackbarPos("border", winName)

    img_new = cv2.GaussianBlur(
    img1,  # входное изображение
    (size*2+1,size*2+1),  # размер ядра
    sigmaX,  # сигма по оси X
    sigmaY,  # сигма по оси Y
    borderType=All_border[border]  # тип рамки
)

    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break

cv2.destroyWindow(winName)
winName = "bilateralFilter"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)


cv2.createTrackbar("size", winName, 0, 100, nothing)
cv2.createTrackbar("sigmaColor", winName, 0, 200, nothing)
cv2.createTrackbar("sigmaSpace", winName, 0, 200, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-1, nothing)

while (1):
    size = cv2.getTrackbarPos("size", winName)
    sigmaColor = cv2.getTrackbarPos("sigmaColor", winName)
    sigmaSpace = cv2.getTrackbarPos("sigmaSpace", winName)
    border = cv2.getTrackbarPos("border", winName)
```

```
    img_new = cv2. bilateralFilter(
    img1,  # входное изображение
    size,  # размер окрестности пикселя
    sigmaColor,  # ширина второго компонента весовой функции
    sigmaSpace,  # ширина первого компонента весовой функции
    borderType=All_border[border]  # тип рамки
)

    cv2.imshow(winName, img_new)
    key = cv2.waitKey(10)
    if key == 27:
        break
```

Задание 2.

```
import cv2
import numpy

def nothing(x):
    pass

pathImg1 = "./lab3/3-3.jpg"
winName = "Border"

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)
img1 = cv2.imread(pathImg1, flags=cv2.IMREAD_COLOR)

height, weight = img1.shape[0:2]

All_border = [cv2.BORDER_CONSTANT, cv2.BORDER_DEFAULT, cv2.BORDER_ISOLATED, cv2.BORDER_REFLECT,
              cv2.BORDER_REFLECT101, cv2.BORDER_REFLECT_101, cv2.BORDER_REPLICATE,  cv2.BORDER_WRAP]
All_border_name = ["cv2.BORDER_CONSTANT", "cv2.BORDER_DEFAULT", "cv2.BORDER_ISOLATED",
"cv2.BORDER_REFLECT",
              "cv2.BORDER_REFLECT101", "cv2.BORDER_REFLECT_101", "cv2.BORDER_REPLICATE",
"cv2.BORDER_WRAP"]

winName = "Blur"

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("size_H", winName, 0, 200, nothing)
cv2.createTrackbar("size_W", winName, 0, 200, nothing)
cv2.createTrackbar("anchor_H", winName, 0, 200, nothing)
cv2.createTrackbar("anchor_W", winName, 0, 200, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-2, nothing)

while (1):
    sizeH = cv2.getTrackbarPos("size_H", winName)
    sizeW = cv2.getTrackbarPos("size_W", winName)
    anchorH = cv2.getTrackbarPos("anchor_H", winName)
    anchorW = cv2.getTrackbarPos("anchor_W", winName)
    border = cv2.getTrackbarPos("border", winName)
    if anchorH > sizeH-1:
        anchorH = sizeH
    if anchorW > sizeW-1:
        anchorW = sizeW
    img_new = cv2.blur(
        img1,
        (sizeW+1, sizeH+1),  # размер ядра
        anchor=(anchorW-1, anchorH-1),  # положение якорной точки
        borderType=All_border[border],  # тип рамки
    )
    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break

cv2.destroyWindow(winName)
winName = "BoxFilter"

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("ddepth", winName, 0, 200, nothing)
cv2.createTrackbar("size", winName, 0, 200, nothing)
cv2.createTrackbar("normalize", winName, 0, 1, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-2, nothing)

while (1):
    ddepth = cv2.getTrackbarPos("ddepth", winName)
    size = cv2.getTrackbarPos("size", winName)+2
    f = cv2.getTrackbarPos("normalize", winName)
    border = cv2.getTrackbarPos("border", winName)

    img_new = cv2.boxFilter(
    img1,  # входное изображение
    -1,  # глубина изображения-результата
    (size,size),  # размер ядра
    (size//2,size//2),  # положение якорной точки
    normalize=f,  # нормирование (если True)
    borderType=All_border[border],   # тип рамки
```

```
)
    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break

cv2.destroyWindow(winName)
winName = "medianBlur"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("size", winName, 0, 100, nothing)

while (1):
    size = cv2.getTrackbarPos("size", winName)


    img_new = cv2.medianBlur(
    img1,  # входное изображение
    size*2+1  # размер ядра
)

    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break

cv2.destroyWindow(winName)
winName = "GaussianBlur"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("size", winName, 0, 100, nothing)
cv2.createTrackbar("sigmaX", winName, 0, 200, nothing)
cv2.createTrackbar("sigmaY", winName, 0, 200, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-1, nothing)

while (1):
    size = cv2.getTrackbarPos("size", winName)
    sigmaX = cv2.getTrackbarPos("sigmaX", winName)
    sigmaY = cv2.getTrackbarPos("sigmaY", winName)
    border = cv2.getTrackbarPos("border", winName)

    img_new = cv2.GaussianBlur(
    img1,  # входное изображение
    (size*2+1,size*2+1),  # размер ядра
    sigmaX,  # сигма по оси X
    sigmaY,  # сигма по оси Y
    borderType=All_border[border]  # тип рамки
)
    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27: break

cv2.destroyWindow(winName)
winName = "bilateralFilter"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

cv2.createTrackbar("size", winName, 0, 100, nothing)
cv2.createTrackbar("sigmaColor", winName, 0, 200, nothing)
cv2.createTrackbar("sigmaSpace", winName, 0, 200, nothing)
cv2.createTrackbar("border", winName, 0, len(All_border)-1, nothing)

while (1):
    size = cv2.getTrackbarPos("size", winName)
    sigmaColor = cv2.getTrackbarPos("sigmaColor", winName)
    sigmaSpace = cv2.getTrackbarPos("sigmaSpace", winName)
    border = cv2.getTrackbarPos("border", winName)

    img_new = cv2. bilateralFilter(
    img1,  # входное изображение
    size,  # размер окрестности пикселя
    sigmaColor,  # ширина второго компонента весовой функции
    sigmaSpace,  # ширина первого компонента весовой функции
    borderType=All_border[border]  # тип рамки
)
    cv2.imshow(winName, img_new)
    key = cv2.waitKey(10)
    if key == 27: break
```

Задание 3.

```
import cv2
import numpy

def nothing(x):
    pass

pathImg1 = "./lab3/3-4.jpg"

winName = "Border"

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

img1 = cv2.imread(pathImg1, flags=cv2.IMREAD_COLOR)

height, weight = img1.shape[0:2]

All_border = [cv2.BORDER_CONSTANT, cv2.BORDER_DEFAULT, cv2.BORDER_ISOLATED, cv2.BORDER_REFLECT,
              cv2.BORDER_REFLECT101, cv2.BORDER_REFLECT_101, cv2.BORDER_REPLICATE,  cv2.BORDER_WRAP]
All_border_name = ["cv2.BORDER_CONSTANT", "cv2.BORDER_DEFAULT", "cv2.BORDER_ISOLATED",
"cv2.BORDER_REFLECT",
              "cv2.BORDER_REFLECT101", "cv2.BORDER_REFLECT_101", "cv2.BORDER_REPLICATE",
"cv2.BORDER_WRAP"]

cv2.createTrackbar("border", winName, 0, len(All_border)-1, nothing)
cv2.createTrackbar("top", winName, 0, 200, nothing)
cv2.createTrackbar("bot", winName, 0, 200, nothing)
cv2.createTrackbar("left", winName, 0, 200, nothing)
cv2.createTrackbar("right", winName, 0, 200, nothing)
while (1):
    top = cv2.getTrackbarPos("top", winName)
    bottom = cv2.getTrackbarPos("bot", winName)
    left = cv2.getTrackbarPos("left", winName)
    right = cv2.getTrackbarPos("right", winName)
    border = cv2.getTrackbarPos("border", winName)
    img_new = cv2.copyMakeBorder(
        img1,  # входное изображение
        top,  # пиксели сверху
        bottom,  # пиксели снизу
        left,  # пиксели слева
        right,  # пиксели справа
        borderType=All_border[border] # тип рамки

    )
    color_text = (0,0,255)
    cv2.putText(img_new, "top "+str(top) + "  bottom " + str(bottom), (weight-1500, height-800),
                cv2.FONT_HERSHEY_DUPLEX, fontScale=5, color=color_text, thickness=10)
    cv2.putText(img_new, "left "+str(left) + "  right " + str(right), (weight-1500, height-600),
                cv2.FONT_HERSHEY_DUPLEX, fontScale=5, color=color_text, thickness=10)
    cv2.putText(img_new, All_border_name[border], (weight-1500, height-400),
                cv2.FONT_HERSHEY_DUPLEX, fontScale=4, color=color_text, thickness=10)
    cv2.imshow(winName, img_new)
    key = cv2.waitKey(1)
    if key == 27:
        break
cv2.destroyWindow(winName)
```