Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №3**

**Дисциплина: Алгоритмы цифровой обработки мультимедиа**

Работу выполнила: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Пинский Д.А.

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Крамаренко А. А.

**Цель работы:** Рассмотрение фильтра Гаусса, как базовой части алгоритма выявления контуров объектов на изображении.

**Ход работы:**

Задание 1. Выполнить пункты 1 и 2 алгоритма, то есть построить матрицу Гаусса. Просмотреть итоговую матрицу для размерностей 3, 5, 7.

Задание 2. Нормировать полученную матрицу Гаусса. Протестировать результаты на матрицах из предыдущего пункта.

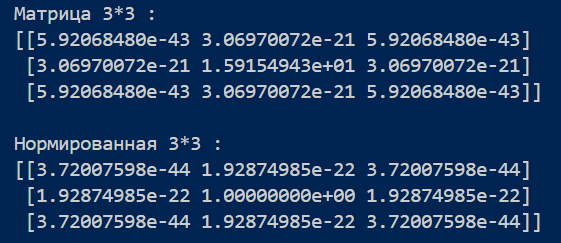


Рисунок 1 – Матрица 3х3 и нормированная матрица 3х3.

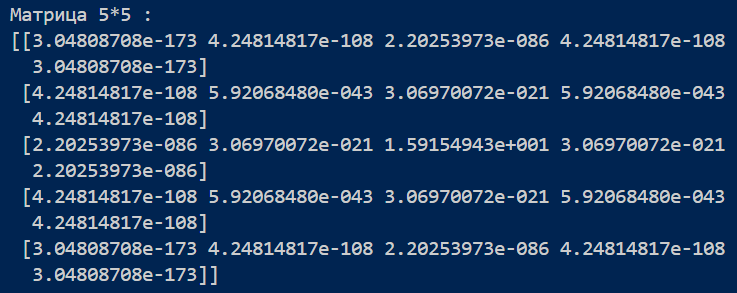


Рисунок 2 – Матрица 5х5.

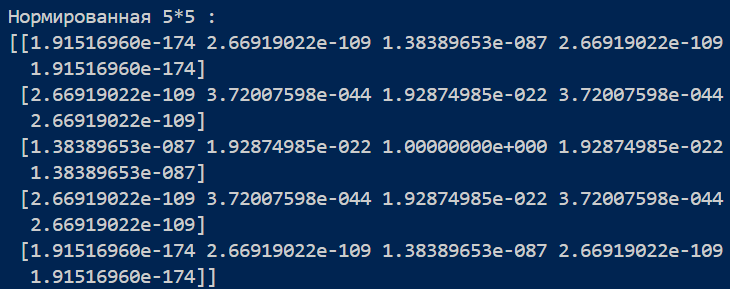


Рисунок 3 –Нормированная матрица 5х5.

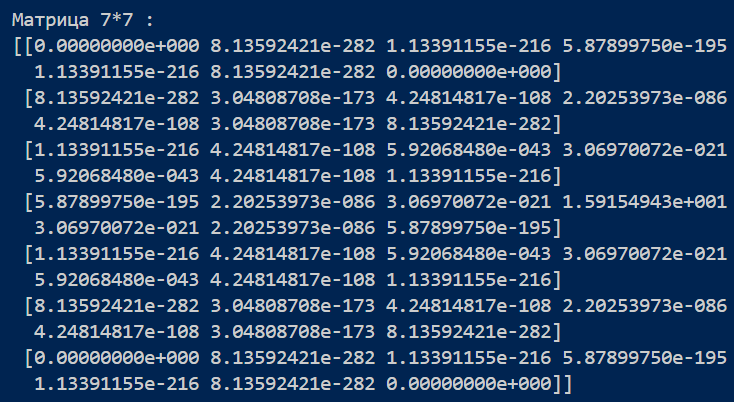


Рисунок 4 – Матрица 7х7.

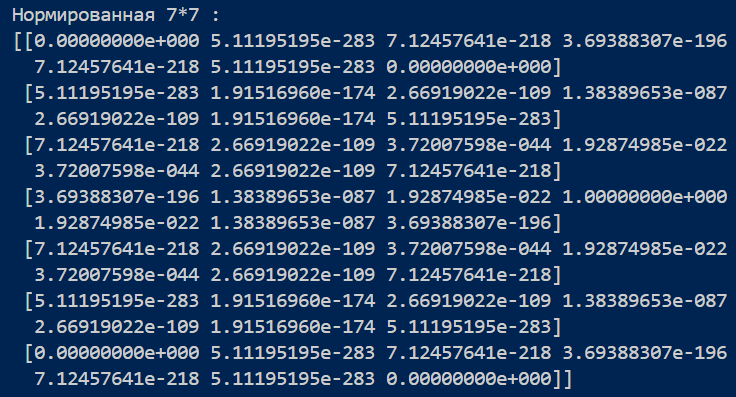


Рисунок 5 – Нормированная матрица 7х7.

Задание 3. Реализовать фильтр Гаусса средствами языка python.

Задание 4. Применить данный фильтр для двух разных значений среднего квадратичного отклонения и двух разных размерностей матрицы свертки, сравнить результаты для ОДНОГО изображения.

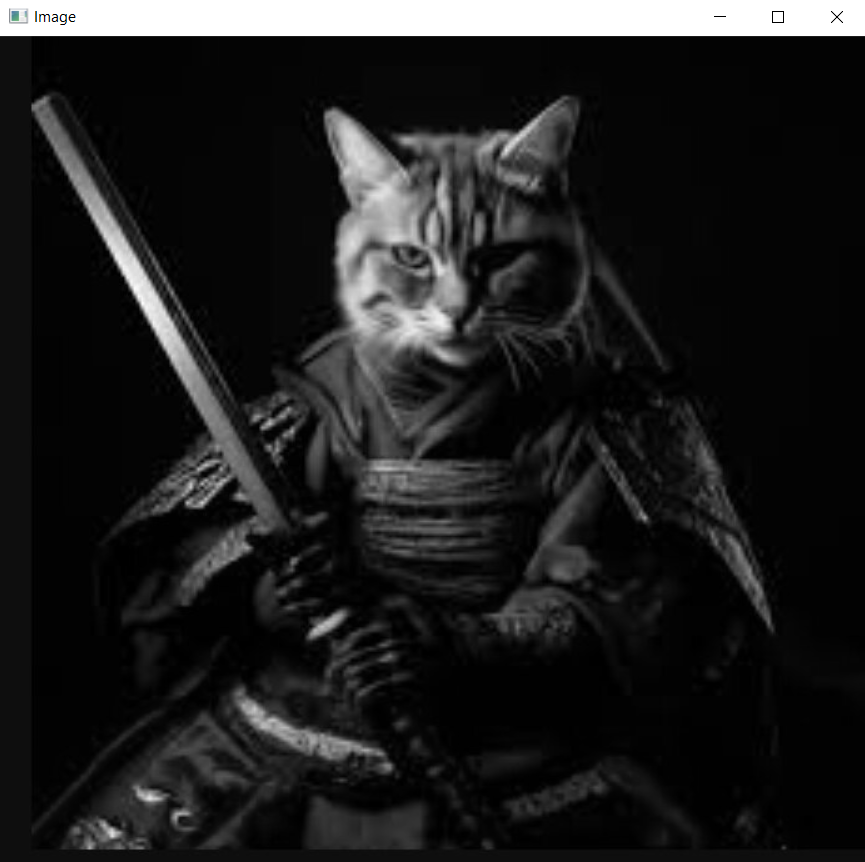


Рисунок 6 – Изображение в черно-белом формате



Рисунок 7 – Отфильтрованное изображение

Задание 5. Реализовать размытие Гаусса встроенным методом библиотеки OpenCV, сравнить результаты с Вашей реализацией.

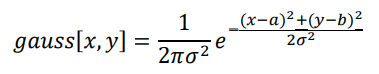


Рисунок 8 – Отфильтрованное изображение при помощи встроенного метода.

**Ответы на вопросы:**

1. Опишите, в чем заключается принцип операции размытия изображения и зачем ее применяют?

Размытие по Гауссу использует гауссову функцию для создания ядра свертки, которое определяет, как значения пикселей будут смешиваться. Формула гауссовой функции выглядит следующим образом:

​

Результатом применения гауссового фильтра является размытое изображение, где детали и шумы сглаживаются. Это помогает уменьшить резкость и визуальные артефакты.

2. Что такое операция свертки матриц.

Свертка используется для вычисления нового значения пикселя на основе значений окружающих его пикселей.

Свертка начинается с определения ядра — матрицы весов.

Ядро “скользит” по изображению, начиная с верхнего левого угла. Для каждого положения ядра выполняется поэлементное умножение значений пикселей на соответствующие значения из ядра.

Результаты поэлементного умножения суммируются, чтобы получить новое значение для центрального пикселя. Это значение заменяет исходное значение пикселя в выходном изображении.

3. Каким образом строится матрица свертки в размытии по Гауссу?

Матрица свертки, создается на основе функции Гаусса, которая описывает нормальное распределение. В этом процессе учитываются координаты пикселей и стандартное отклонение, которое определяет степень размытия.

4. Опишите алгоритм размытия по Гауссу.

Определение ядра свертки.

Создание гауссового ядра.

Свертка.

5. Укажите известные Вам параметры размытия по Гауссу.

Размер ядра

Стандартное отклонение (сигма)

6. Каким образом производить фильтрацию по Гауссу средствами библиотеки OpenCV?

cv2.GaussianBlur(), чтобы применить размытие. Нужно указать размер ядра и стандартное отклонение (сигма).

**Листинг программ**

import cv2

import numpy as np

image = cv2.imread('C:/ACOM/LR3/kot.jpg')

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

def gauss(size, sigma):

    ker = np.zeros((size, size))

    a = size // 2

    b = size // 2

    for x in range(size):

        for y in range(size):

            ker[x, y] = (1 / (2 \* np.pi \* sigma \*\* 2)) \* np.exp(-((x - a) \*\* 2 + (y - b) \*\* 2) / (2 \* sigma \*\* 2))

    return ker

# Вывод матриц

# val = [3, 5, 7]

# for size in val:

#     ker = gauss(size, sigma)

#     norm\_ker = ker / np.sum(ker)

#     print(f"Матрица {size}\*{size} :\n{ker}\n")

#     print(f"Нормированная {size}\*{size} :\n{norm\_ker}\n")

def filter(image, size, sigma):

    ker = gauss(size, sigma)

    norm\_ker = ker / np.sum(ker)

    image\_copy = image.copy()

    mean = size // 2

    h = image\_copy.shape[0]

    w = image\_copy.shape[1]

    for i in range(mean, h - mean):

        for j in range(mean, w - mean):

            sum = 0

            for k in range(-mean, mean + 1):

                for l in range(-mean, mean + 1):

                    sum += image\_copy[i + k, j + l] \* norm\_ker[k + mean, l + mean]

            image\_copy[i, j] = sum

    return image\_copy

result = filter(gray, 5, 1.0)

gauss\_cv = cv2.GaussianBlur(gray, (5, 5), 1.0)

cv2.imshow('Filter', result)

cv2.imshow('FilterCV2', gauss\_cv)

cv2.imshow('Image', gray)

cv2.waitKey(0)

cv2.destroyAllWindows()