**Минобрнауки РФ**

**ЮЗГУ**

**ФФиПИ**

Кафедра программной инженерии

**ЛАБОРАТОРНАЯ РАБОТА №9**

**ВВЕДЕНИЕ В ФИЛЬТРАЦИЮ ИЗОБРАЖЕНИЙ**

Выполнил: ст. гр. ПО-22б

Алешин К.Р.

Проверил: Ефремов В. В.

Курск, 2024 г

**Задание:**  
Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание

**Листинг программы:**import cv2

import numpy as np

from matplotlib import pyplot as plt

def apply\_low\_pass\_filter(image, radius):

"""Применяет фильтр низких частот: размытие за пределами круга радиуса R."""

# Создаем маску: область внутри круга - 1, за пределами - 0

rows, cols = image.shape[:2]

center = (cols // 2, rows // 2)

mask = np.zeros((rows, cols), dtype=np.uint8)

cv2.circle(mask, center, radius, 255, -1)

# Применяем Гауссово размытие

blurred = cv2.GaussianBlur(image, (15, 15), 0)

# Комбинируем: область вне круга - размытая, внутри - оригинал

result = np.copy(image)

result[mask == 0] = blurred[mask == 0]

return result

def apply\_high\_pass\_filter(image, threshold):

"""Применяет фильтр высоких частот: повышает резкость в районе пикселей с яркостью > T."""

# Преобразуем изображение в оттенки серого

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

# Находим области с яркостью выше порога

mask = cv2.threshold(gray, threshold, 255, cv2.THRESH\_BINARY)[1]

# Применяем фильтр резкости (ядро повышения резкости)

kernel = np.array([[0, -1, 0],

[-1, 5, -1],

[0, -1, 0]], dtype=np.float32)

sharpened = cv2.filter2D(image, -1, kernel)

# Комбинируем: для маски (> T) используем резкое изображение, остальное оставляем оригинальным

result = np.copy(image)

result[mask > 0] = sharpened[mask > 0]

return result

def main():

# Загружаем изображение

image\_path = "input.jpg"

image = cv2.imread(image\_path)

if image is None:

print("Ошибка: изображение не найдено.")

return

# Ввод параметров

radius = int(input("Введите радиус R для фильтра низких частот: "))

threshold = int(input("Введите порог T для фильтра высоких частот: "))

# Применяем фильтры

low\_pass\_result = apply\_low\_pass\_filter(image, radius)

high\_pass\_result = apply\_high\_pass\_filter(image, threshold)

# Отображаем результаты

plt.figure(figsize=(12, 6))

plt.subplot(1, 3, 1)

plt.imshow(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

plt.title("Исходное изображение")

plt.axis('off')

plt.subplot(1, 3, 2)

plt.imshow(cv2.cvtColor(low\_pass\_result, cv2.COLOR\_BGR2RGB))

plt.title(f"Фильтр низких частот (R={radius})")

plt.axis('off')

plt.subplot(1, 3, 3)

plt.imshow(cv2.cvtColor(high\_pass\_result, cv2.COLOR\_BGR2RGB))

plt.title(f"Фильтр высоких частот (T={threshold})")

plt.axis('off')

plt.tight\_layout()

plt.show()

# Сохранение результатов

cv2.imwrite("low\_pass\_result.png", low\_pass\_result)

cv2.imwrite("high\_pass\_result.png", high\_pass\_result)

print("Результаты сохранены как 'low\_pass\_result.png' и 'high\_pass\_result.png'.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Пример работы программы:  
Изображение выглядит как снимок экрана, дерево, на открытом воздухе, пейзаж

Автоматически созданное описание**

**Изображение выглядит как снимок экрана, дерево, на открытом воздухе, восход солнца

Автоматически созданное описание**