

ОТЧЕТ

По лабораторной работе №2

Тема: Основы обработки цифровых изображений.

Вариант: 15

Выполнил: Студент 4 гр. Минковский Виталий

1. Цель работы

Изучение алгоритмов предварительной обработки изображений. Реализация веб-приложения для нелинейной фильтрации и локальной пороговой обработки изображений с возможностью динамической настройки параметров.

2. Постановка задачи (Вариант 15)

Согласно варианту необходимо реализовать следующие методы:

1. **Нелинейные фильтры, основанные на порядковых статистиках:** Реализовать медианный фильтр.
2. **Локальная пороговая обработка:** Реализовать и сравнить два метода (Adaptive Mean и Adaptive Gaussian).

Требования к реализации:

- Разработка графического интерфейса (Web).
- Возможность загрузки собственных изображений.
- Подбор тестовых изображений (зашумленные, с неравномерным освещением).
- Динамическое изменение параметров обработки.

3. Теоретические сведения

3.1. Медианная фильтрация

Медианный фильтр — это нелинейный метод подавления шума. Алгоритм работает с окном (ядром), скользящим по изображению. Значения пикселей, попавших в окно, сортируются по возрастанию, и центральный пиксель заменяется на медиану (средний элемент) этого упорядоченного списка. Этот метод особенно эффективен для устранения импульсного шума (тип «соль и перец»), при этом, в отличие от линейного сглаживания, он лучше сохраняет резкие границы объектов.

3.2. Локальная пороговая обработка (Адаптивная бинаризация)

В отличие от глобального порога, где используется одно значение для всего изображения, адаптивный метод вычисляет порог $T(x,y)$ для каждого пикселя индивидуально на основе небольшой области вокруг него(блока).

$$T(x,y)=Mean(Block)-C$$

Где C — константа.

Реализованы два метода вычисления среднего:

1. **Adaptive Mean:** Среднее арифметическое всех пикселей в блоке.
2. **Adaptive Gaussian:** Взвешенная сумма значений в блоке, где веса распределены по закону Гаусса (центральные пиксели важнее).

Этот метод применяется для обработки изображений с неравномерным освещением (наличие теней, бликов).

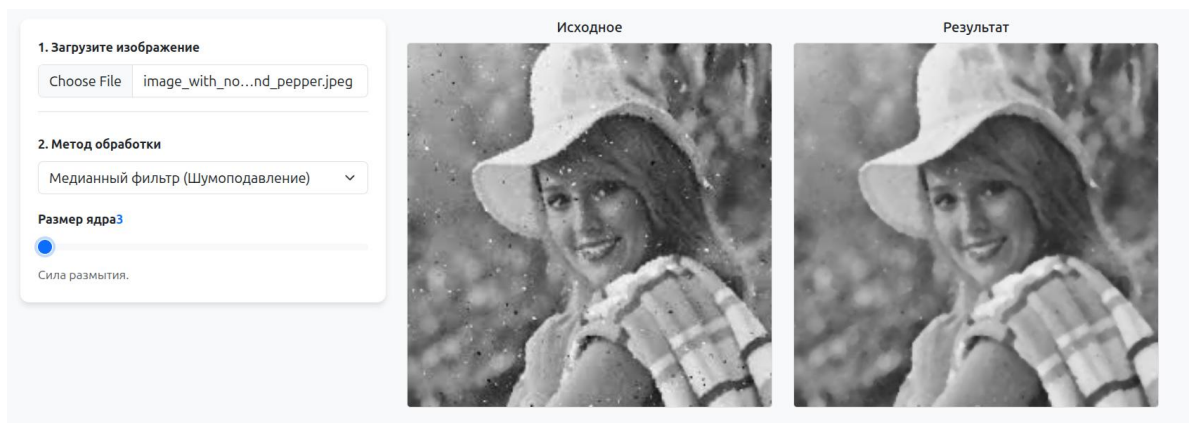
4. Описание реализации

Приложение разработано на языке **Python** с использованием фреймворка **FastAPI**.

- **Бэкенд:** Обработка изображений выполняется с помощью библиотеки **OpenCV**. Изображения хранятся в памяти и конвертируются в формат Base64 для передачи на клиент.
- **Фронтенд:** HTML + JavaScript. Используются ползунки (Range inputs) для изменения параметров (Block Size, Constant C, Kernel Size). Обмен данными происходит асинхронно через fetch API, что обеспечивает мгновенное обновление результата без перезагрузки страницы.

5. Результаты тестирования

5.1. Тестирование Медианного фильтра



Входные данные: Изображение с искусственным импульсным шумом («соль и перец»).

5.2. Тестирование Локальной пороговой обработки

Входные данные: Фотография текста/QR-кода с падающей тенью (сильный перепад освещения).

1. Adaptive Mean:



2. Adaptive Gaussian:



Оба метода показали приемлемый результат

6. Выводы

В ходе лабораторной работы было реализовано веб-приложение для обработки изображений.

1. **Медианная фильтрация** подтвердила свою эффективность в борьбе с импульсным шумом. В отличие от размытия по Гауссу, она не "размазывает" шум, а исключает его.
2. **Локальная пороговая обработка** является необходимым инструментом для сегментации реальных фотографий документов. Глобальный порог в условиях неравномерного освещения неработоспособен.
3. Критически важным является динамический подбор параметров Block Size и C . Слишком маленький блок приводит к потере сплошности объектов (эффект контура), а слишком большая константа C приводит к разрывам в тонких линиях.