МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение   
высшего образования «Самарский национальный исследовательский университет   
имени академика С.П. Королева»

(Самарский университет)

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ**

**РАБОТЕ №1**

Предмет: *Системы обработки изображений*

Выполнил:

студент группы

№ 6131-010402D Музыка А.А.

Проверил: Демин Никита Сергеевич

Самара 2022

**Содержание**

1. **Поэлементные преобразования изображений 3**
   1. **Цель работы 3**
   2. **Задание 3**
   3. **Основная часть 4**

**Поэлементные преобразования изображений**

* 1. **Цель работы**

Целью данной лабораторной работы является изучение и экспериментальное исследование простых поэлементных методов цифровой обработки видеоинформации; получение навыков работы с автоматизированной системой обработки изображений.

* 1. **Задание**

1. Считать цветное RGB изображение;
2. Преобразовать изображение в градации серого;
3. Написать функцию реализации препарирования изображения;
4. По гистограмме изображения определить значение порога яркостей, обеспечивающего оптимальное разделение объекта и фона. Осуществить пороговую обработку входного изображения с найденным пороговым значением;
5. Сделать пороговую обработку методом Otsu(используя библиотеку OpenCV);
6. Определить динамический диапазон входного изображения. Осуществить линейное контрастирование входного изображения в заданный динамический диапазон яркостей;
7. Сделать эквализацию гистограммы изображения;
8. Сделать эквализацию методом CLAHE(используя функцию openCV)
9. Осуществить препарирование изображения с заданной препарирующей функцией.
   1. **Основная часть**

Для работы на языке программирования Python нам понадобятся следующие библиотеки:

1. *Matplotlib* для визуализации графиков и изображений.
2. *NumPy* для комфортной работы с многомерными массивами.
3. *OpenCV* для обработки изображений (по условиям задания)

Разбираемый код можно найти по следующей ссылке на GitHub: <https://github.com/Liptee/Image-processing>

После прочтения изображения и его записи в значение img, следуют функции разложения на каналы (decomp) и из разложенных каналов следующая функция получает изображение в градациях серого используя формулу цветового пространства YCrCb. Передаем img функции grayscale, которая возвращает двумерный массив изображения.



**Рис. 1 Изображение в градациях серого**

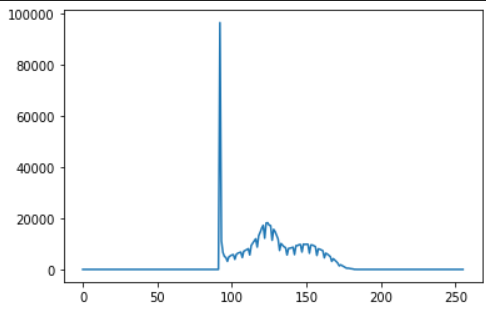
Для преобразования цветного RGB изображения в градации серого существуют несколько формул для вычисления яркостей. В практической части была использована формула цветового пространства YCrCb:

Y = 0.299R + 0.587G + 0.114B (1)

где R, G, B - компоненты системы RGB. Кодирования или вычисления яркости осуществляется с использованием эмпирических коэффициентов при компонентах RGB, учитывающие особенности человеческого зрения, а именно его чувствительность при восприятии основных цветов.

Далее по коду следует функция препарирования (prep). Она принимает в себя только изображение и функцию препарирования и поэлементно применяет ее ко всем элементам оригинального изображения, возвращая новое препарированное изображение.

Далее идет функция make\_hist с говорящим названием. Она принимает в себя изображение, а возвращает массив с 256 значениями гистограммы. В ней изначально создается список с 256 нулями и за тем функция перебирает элементы изображения и прибавляет к соответствующему индексу нового списка 1, для каждого элемента входного изображения.

****

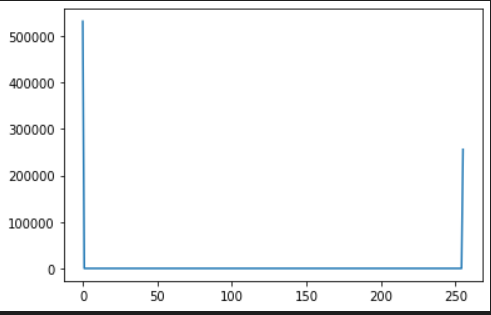
**Рис. 2 Гистограмма оригинального изображения**

Функция бинаризации (или пороговой обработки) изображения получает на входе изображение и индекс порога, возвращая двуцветное изображение.

****

**Рис. 3 Изображение после пороговой обработки**

На рис. 3 представлен результат бинаризации с порогом 137. Порог был выбран на основе двух вершин гистограммы (не считая самой первой) находящихся примерно на 125 и 150 значениях.

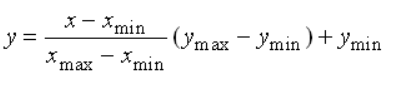


**Рис. 4 Гистограмма бинарного изображения**

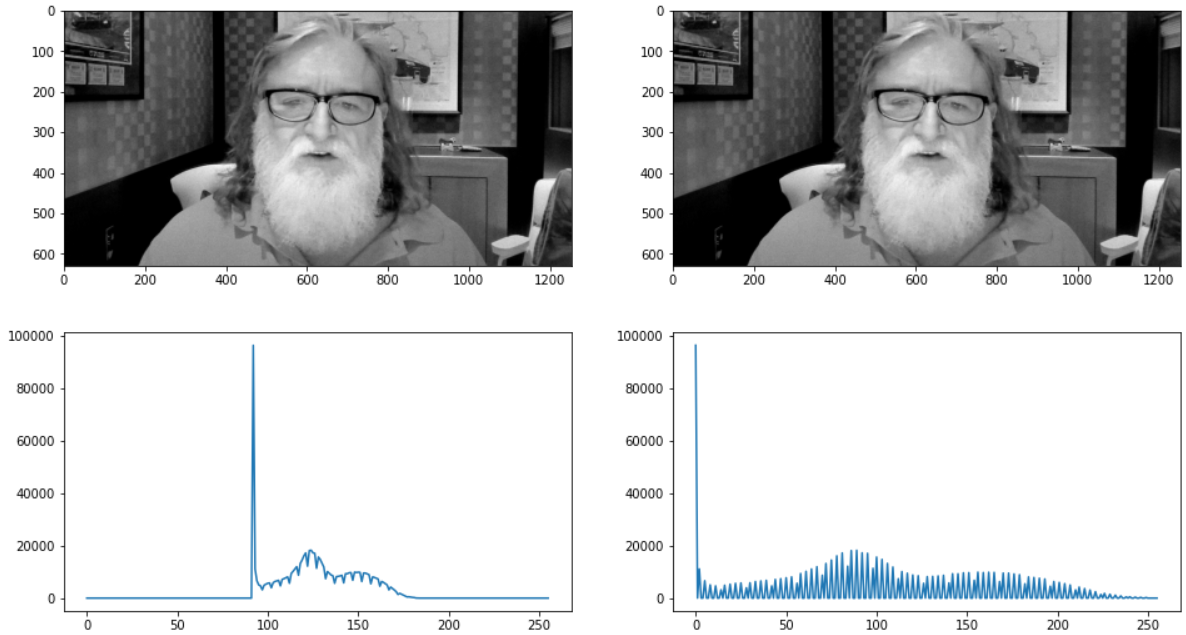
Далее реализована функция Otsu для нахождения порогового значения по гистограмме изображения. В ней мы для каждого значения тона t гистограммы вычисляем их значения ω1, ω2, μ1, μ2 (в коде t = h, а ω1, ω2, μ1, μ2 = w1, w2, m1, m2 соответственно). Далее мы вычисляем сигму. Наша цель найти максимальную сигму(t). Код возвращает t с максимальной сигмой. В нашем случае данный код возвращает значение 130. Далее в коде идет применение реализованного метода Otsu и его визуальное сравнение с методом Otsu из библиотеки OpenCV.

Далее следует функция find\_edged, которая определяет динамический диапазон гистограммы. Возвращает она первые индексы списка hist, значения которых не равны 0 с каждого края списка.

Формула линейного контрастирования:

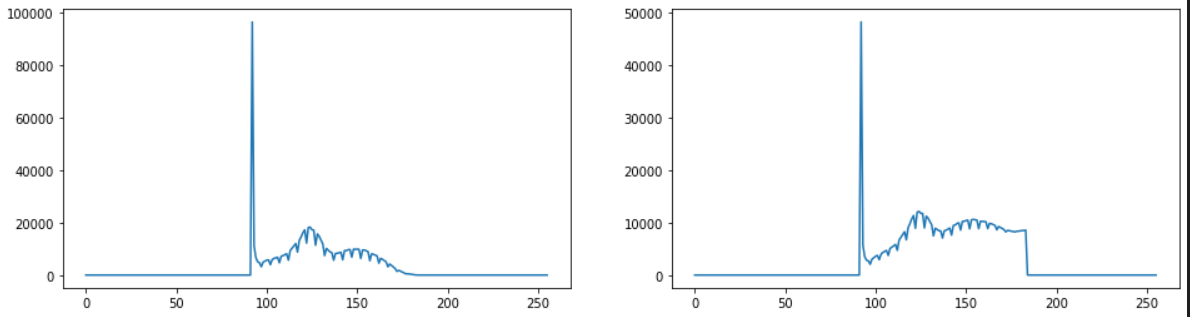
 (2)

Далее идет функция линейного контрастирования с фиксированными значениями ymin и ymax равными 0 и 255 соответственно, что дает в результате изображение с максимальным динамическим диапазоном.



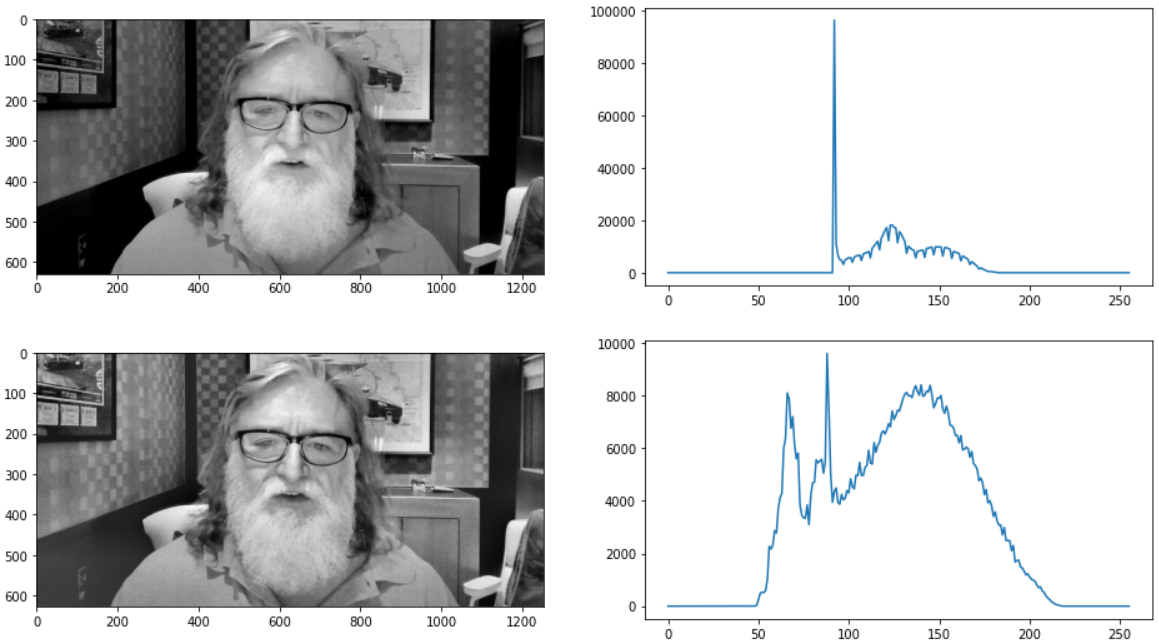
**Рис. 5 Сравнение оригинального изображения с обработанным после функции линейного контрастирования**

Далее идет эквализация гистограммы изображения и эквализация встроенным методом CLAHE (OpenCV). Для эквализация гистограммы сначала высчитывается прямолинейное возрастающее распределение для заданного числа элементов (суммы элементов гистограммы). Затем берется среднее значение элемента гистограммы и его соответсвующего элемента из распределения.

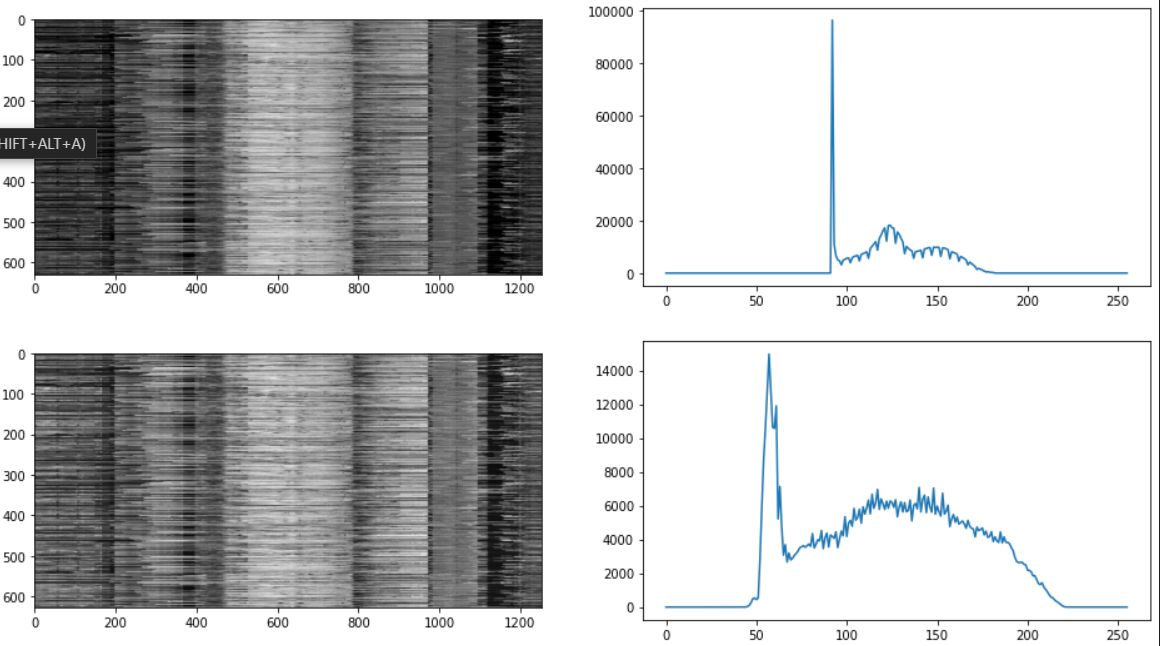


**Рис. 6 Гистограмма оригинального изображения и эквализированной гистограммы**

При работе с эквализацией гистограммы шла оглядка на результаты эквализации CLAHE, однако после эксперимента с перемешиванием элементов изображения, стало ясно, что эквализация CLAHE работает не только с гистограммой, но и использует данные о положении пикселей в изображении для коррекции изображения.



**Рис. 7 Результаты эквализации CLAHE(OpenCV)**



**Рис. 8 Результаты эквализации CLAHE для исходного изображения с перемешенными элементами**

Как мы видим на рисунках 7 и 8, при одинаковых входных гистограммах, CLAHE получил разные результаты на выводе.

Далее была реализована препарирующая функция c использованием ранее написанной функцией препарирования изображения.

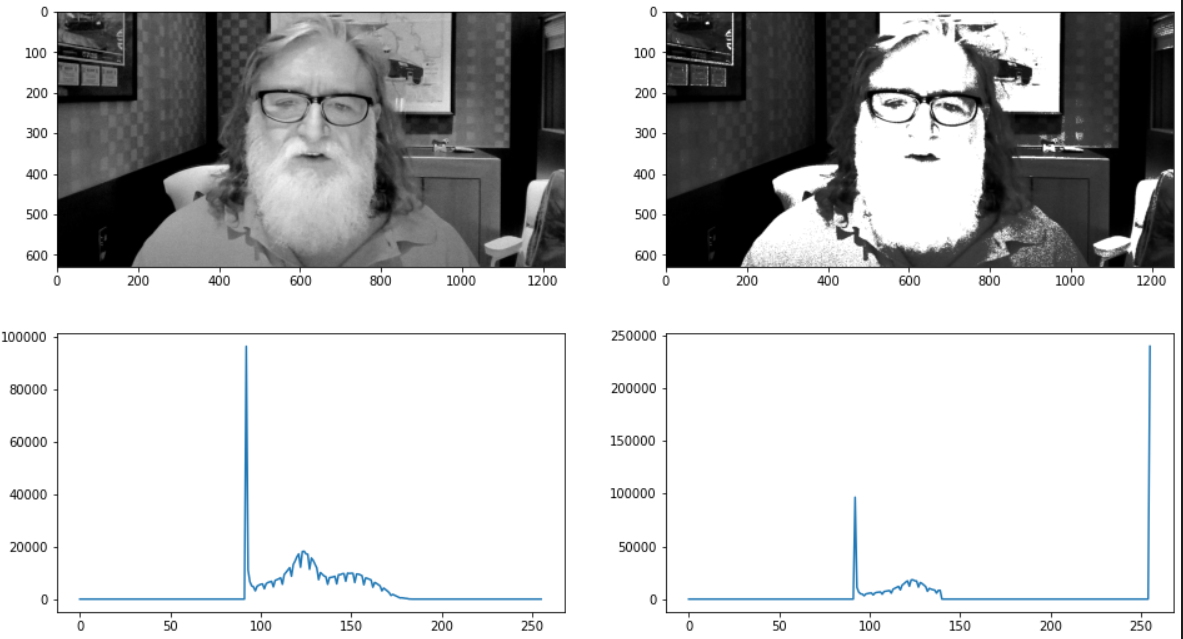


Рис. 8 Результаты препарирования изображения