МИНИСТЕРСТВО НАУКИ и ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Московский политехнический университет»**

Направление: Информатика и вычислительная техника

Дисциплина «Обработка изображений»

**Практическая работа №3**

«Применение гистограммных методов коррекции»

Преподаватель:

Пухова   
Екатерина Александровна

Студентка:

Кириленко М.А.  
гр. 224-322

Москва 2023

*Цель:* познакомится с пространственными методами коррекции на примере гистограммной коррекции.

*Этапы выполнения*

1. Подобрать 2 изображения для коррекции (можно взять из работы 2)

2. Перевести изображения в черно-белое

3. Получить гистограммы изображений

4. Провести нормализацию гистограмм

5. Провести эквализацию гистограмм

6. Провести преобразование гистограммы по произвольно заданной функции распределения

1. *Используемый язык программирования*

Python 3.11.1

1. *Параметры исходных изображений*

* *

01:

a. глубина цвета - 24, bpp

b. размер - 5734 x 3819, pix

02:

a. глубина цвета - 24, bpp

b. размер - 1060 x 1484, pix

1. *Черно-белые изображения*



1. *Гистограммы изображений*

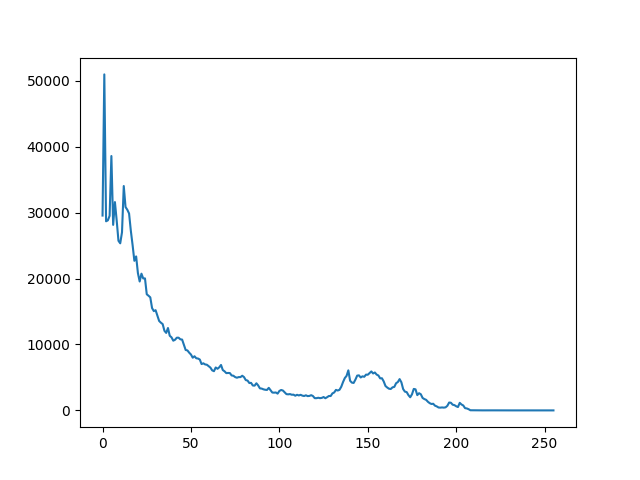
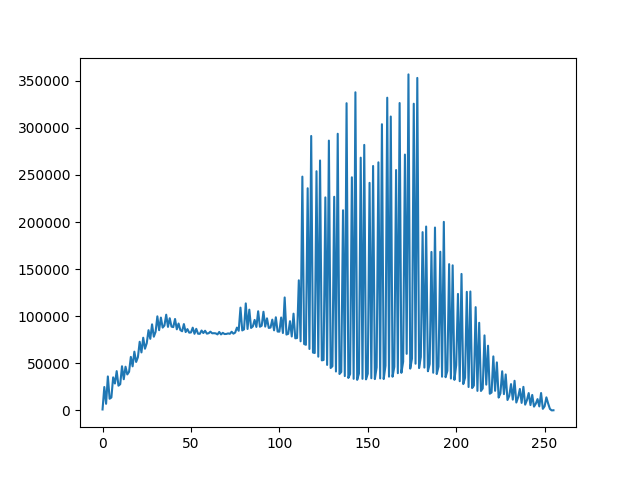
**

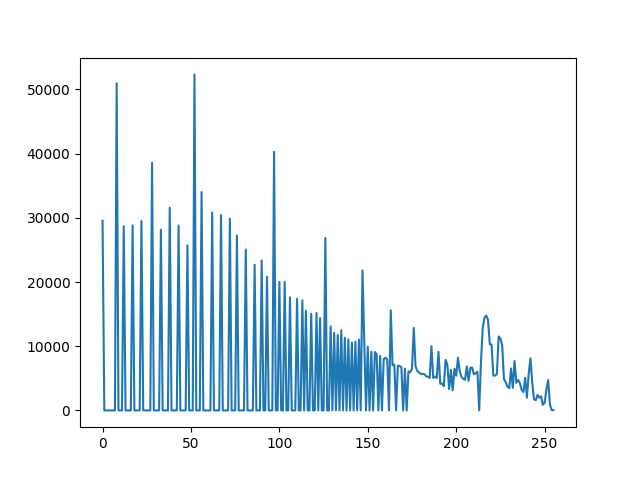
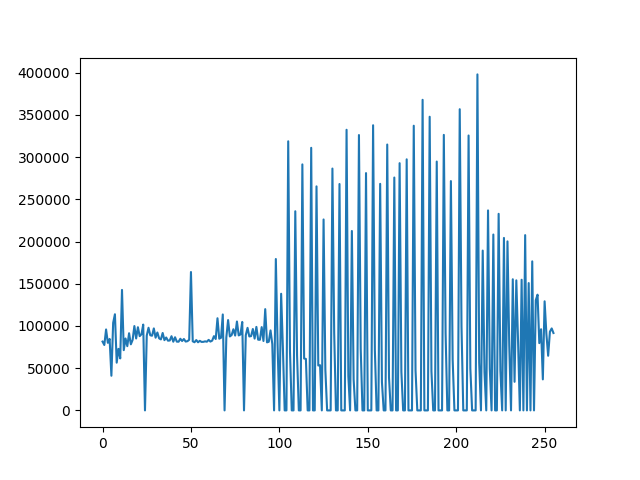
Таблица 1

|  |  |  |
| --- | --- | --- |
| № изображения | 01 | 02 |
| k, bpp | 24 | 24 |
| m x n, pix | 5734 x 3819 | 1060 x 1484 |
| Изображение | 01\_bw | 02\_bw |
| гистограмма | 01\_ bw\_bc | 02\_ bw\_bc |

1. *Изображения после нормализации*

**

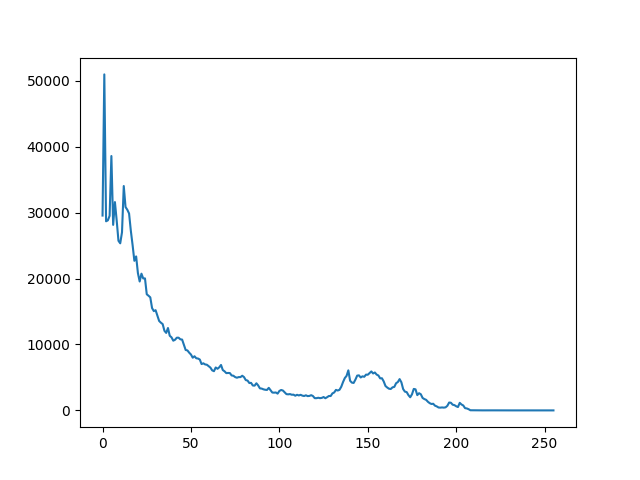
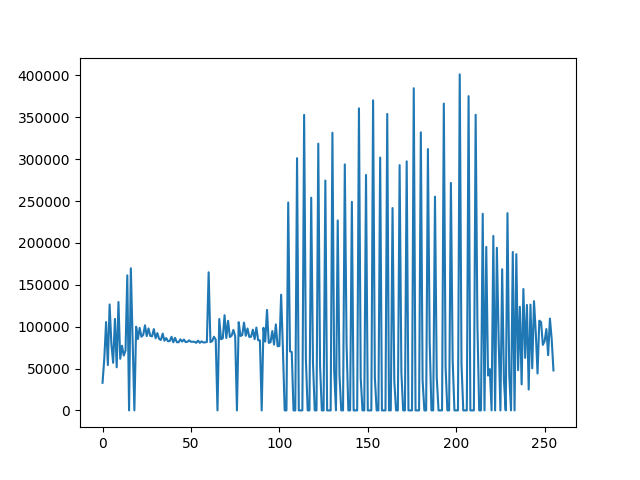
1. *Гистограммы нормализованных изображений*

**

1. *Изображения после эквализации*

**

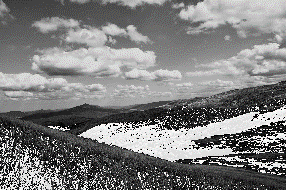
1. *Гистограммы эквализированных изображений*

**

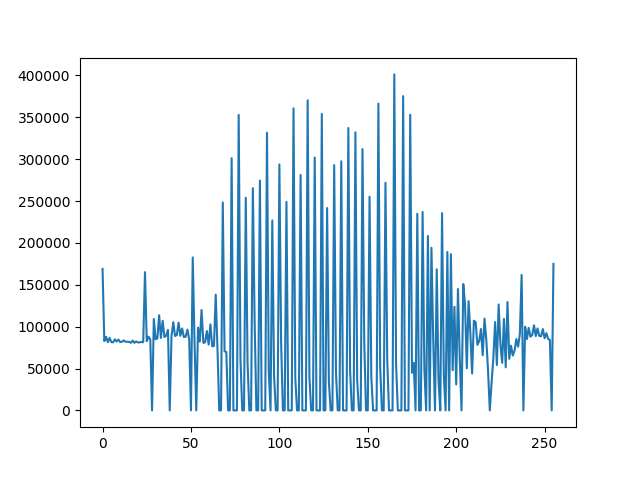
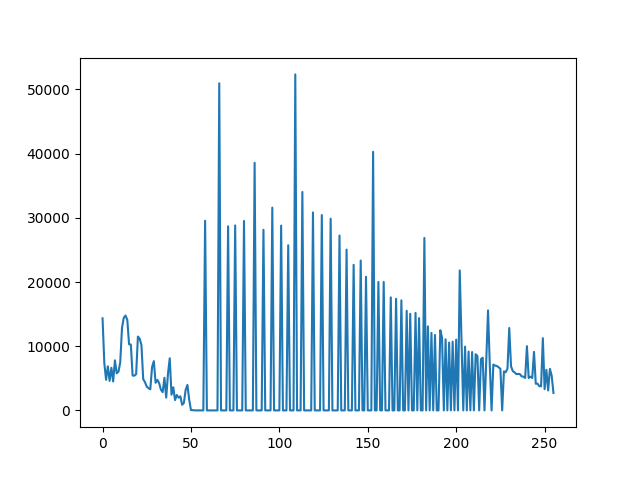
1. *Функция распределения, по которой будет проведено преобразование гистограммы*

func\_cdf\_i = (func\_cdf\_i - func\_cdf\_i[random.randint(0, 256)] + random.randint(0, 256))\*255/(func\_cdf\_i[-1]-1)

1. *Изображения после применения к гистограмме заданной функции*

**

1. *Гистограммы изображений после применения к гистограмме заданной функции*

|  |  |  |
| --- | --- | --- |
| № изображения | 01 | 02 |
| Нормализованная гистограмма | 01\_bc\_norm | 02\_bc\_norm |
| Эквализованная гистограмма | 01\_bc\_ekv | 02\_bc\_ekv |
| Функция распределения | func\_cdf\_01 = (func\_cdf\_01 - func\_cdf\_01[random.randint(0, 256)] + random.randint(0, 256))\*255/(func\_cdf\_01[-1]-1) | func\_cdf\_02 = (func\_cdf\_02 - func\_cdf\_02[random.randint(0, 256)] + random.randint(0, 256))\*255/(func\_cdf\_01[-1]-1) |
| Гистограмма, преобразованная по  заданной функции | 01\_func\_bc | 02\_func\_bc |

1. *Ссылка на изображения*

https://drive.google.com/drive/folders/1t44J1C8zhezC\_XWO9yj3NPiMxc-FGgRG?usp=sharing

*11. Код программы с комментариями*

import matplotlib.pyplot as plt

import numpy as np

import random

import cv2

from matplotlib import pyplot as plt

#Исходные изображения

orig\_img\_01 = cv2.imread('img/01.jpg')

orig\_img\_02 = cv2.imread('img/02.jpg')

#Перевести изображения в черно-белые

bw\_01 = cv2.cvtColor(orig\_img\_01.copy(), cv2.COLOR\_BGR2GRAY)

bw\_02 = cv2.cvtColor(orig\_img\_02.copy(), cv2.COLOR\_BGR2GRAY)

#Запись в папку черно-белых изображений

cv2.imwrite('img/01\_bw.png', bw\_01)

cv2.imwrite('img/02\_bw.png', bw\_02)

#Вычислить гистограммы

bc\_01 = cv2.calcHist([bw\_01], [0], None, [256], [0, 256])

bc\_02 = cv2.calcHist([bw\_02], [0], None, [256], [0, 256])

#Отобразим график

plt.plot(bc\_01)

plt.show()

#Запишем график в папку

plt.plot(bc\_01)

plt.savefig('img/01\_bw\_bc.png')

plt.close()

#Отобразим график

plt.plot(bc\_02)

plt.show()

#Запишем график в папку

plt.plot(bc\_02)

plt.savefig('img/02\_bw\_bc.png')

plt.close()

#Нормализация гистограмм

#Вычисление гистограммы набора данных

hist\_01, bins\_01 = np.histogram(bw\_01, 256)

hist\_02, bins\_02 = np.histogram(bw\_02, 256)

#Вычисление cdf

cdf\_01 = hist\_01.cumsum()

cdf\_02 = hist\_02.cumsum()

#Формула распределения

cdf\_01 = (cdf\_01-cdf\_01[0])\*255/(cdf\_01[-1]-1)

cdf\_02 = (cdf\_02-cdf\_02[0])\*255/(cdf\_02[-1]-1)

cdf\_01 = cdf\_01.astype(np.uint8)

cdf\_02 = cdf\_02.astype(np.uint8)

#Генерируем изображение после нормализации гистограммы

img\_norm\_01 = np.zeros((384, 495, 1), dtype =np.uint8)

img\_norm\_02 = np.zeros((384, 495, 1), dtype =np.uint8)

img\_norm\_01 = cdf\_01[bw\_01]

img\_norm\_02 = cdf\_02[bw\_02]

#Записываем нормализированные изображения в папку

cv2.imwrite('img/01\_img\_norm.png', img\_norm\_01)

cv2.imwrite('img/02\_img\_norm.png', img\_norm\_02)

bc\_norm\_01, binsa\_01 = np.histogram(img\_norm\_01, 256)

bc\_norm\_02, binsa\_02 = np.histogram(img\_norm\_02, 256)

#Вывод графика

plt.plot(bc\_norm\_01)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_norm\_01)

plt.savefig('img/01\_bc\_norm.png')

plt.close()

#Вывод графика

plt.plot(bc\_norm\_02)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_norm\_02)

plt.savefig('img/02\_bc\_norm.png')

plt.close()

#Эквализация

ekv\_01 = cv2.equalizeHist(bw\_01.copy())

ekv\_02 = cv2.equalizeHist(bw\_02.copy())

#сохранения изображений в папку

cv2.imwrite('img/01\_\_img\_ekv.png', ekv\_01)

cv2.imwrite('img/02\_img\_ekv.png', ekv\_02)

#вычисление гистограмм

bc\_ekv\_01 = cv2.calcHist([ekv\_01], [0], None, [256], [0, 256])

bc\_ekv\_02 = cv2.calcHist([ekv\_02], [0], None, [256], [0, 256])

#Вывод графика

plt.plot(bc\_ekv\_01)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_ekv\_01)

plt.savefig('img/01\_bc\_ekv.png')

plt.close()

#Вывод графика

plt.plot(bc\_ekv\_02)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_ekv\_02)

plt.savefig('img/02\_bc\_ekv.png')

plt.close()

# Преобразование гистограм по произвольной функции распределения

func\_cdf\_01 = hist\_01.cumsum()

func\_cdf\_02 = hist\_02.cumsum()

# Произвольная функция распредления

func\_cdf\_01 = (func\_cdf\_01 - func\_cdf\_01[random.randint(0, 256)] + random.randint(0, 256))\*255/(func\_cdf\_01[-1]-1)

func\_cdf\_02 = (func\_cdf\_02 - func\_cdf\_02[random.randint(0, 256)] + random.randint(0, 256))\*255/(func\_cdf\_02[-1]-1)

func\_cdf\_01 = func\_cdf\_01.astype(np.uint8)

func\_cdf\_02 = func\_cdf\_02.astype(np.uint8)

#Генерируем изображение после нормализации гистограммы

func\_norm\_01 = np.zeros((384, 495, 1), dtype =np.uint8)

func\_norm\_02 = np.zeros((384, 495, 1), dtype =np.uint8)

func\_norm\_01 = func\_cdf\_01[bw\_01]

func\_norm\_02 = func\_cdf\_02[bw\_02]

#записываем изображения в папку

cv2.imwrite('img/01\_func\_norm.png', func\_norm\_01)

cv2.imwrite('img/02\_func\_norm.png', func\_norm\_02)

bc\_func\_norm\_01, bins\_func\_01 = np.histogram(func\_norm\_01, 256)

bc\_func\_norm\_02, bins\_\_func\_02 = np.histogram(func\_norm\_02, 256)

#вывод графика

plt.plot(bc\_func\_norm\_01)

plt.show()

#запись графика в папку

plt.plot(bc\_func\_norm\_01)

plt.savefig('img/01\_func\_bc.png')

plt.close()

#вывод графика

plt.plot(bc\_func\_norm\_02)

plt.show()

#запись графика в папку

plt.plot(bc\_func\_norm\_02)

plt.savefig('img/02\_func\_bc.png')

plt.close()