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»

**Лабораторная работа №1**

«Выбор параметров градационной коррекции на основе требований к конечному изображению»

Преподаватель:

Пухова   
Екатерина Александровна

Студентка:

Кириленко М.А.  
гр. 224-322

Москва 2023

1. Цель работы

Провести градационную коррекцию с учетом исходных параметров изображения и заданных параметров к откорректированному изображению. Сравнить разные методы коррекции, выбрать оптимальный.

1. Содержание работы

* Проанализировать предложенное изображение по следующим параметрам: глубина цвета, разница между максимальной и минимальной светлотой,
* Построить гистограмму изображения
* Рассмотреть возможные варианты коррекции изображения которые позволят подчеркнуть детали изображения, содержащие важную информацию о переломе
* Выбрать оптимальный вариант коррекции из рассмотренных в п. 4
* Построить гистограмму изображения после коррекции и сравнить с гистограммой, полученной в п. 2
* Оценить контраст откорректированного изображения

1. Изображение для коррекции (рис. 1)

  
Рис.1. Изображение для коррекции

1. Теоретическая часть

В данной работе рассматривается возможность изменения светлот в изображении для визуализации наиболее важных участков изображения, за счет изменения скорости приращения светлоты (L) на определенных участках тонового диапазона изображения. Тоновый диапазон изображения зависит от глубины цвета k, измеряемой в количестве ,бит на пиксель [bpp]. Если для изображения в градациях серого k=8, то тоновый диапазон [0:255]. При этом пиксели изображения могут занимать не весь этот диапазон, а только его часть. Разница между Lmax и Lmin в изображении определяет его контраст K. Чем больше эта разница, тем выше контраст. Контраст можно оценивать не только относительно всего изображения, но и на локализованных участках тонового диапазона, сопоставив этот участок с конкретной областью изображения

Методы коррекции распределения светлот в изображении, это пространственные методы коррекции. К базовым методам относятся:

1. преобразования по переходным кривым

a. преобразование полярности

b. логарифмические преобразования c. степенные преобразования

d. кусочно-линейные преобразования

2. гистограммные преобразования

a. нормализаци

b. эквализация c. приведение гистограммы по заданной функции

d. вырезание уровней

Эти методы позволяют изменить скорость приращения светлоты как по всему изображению так и на отдельных его участках, что в свою очередь приведет к увеличению контраста и проработке деталей. Однако, следует учитывать, что чрезмерное повышение контраста может привести к потере деталей (рис. 2)

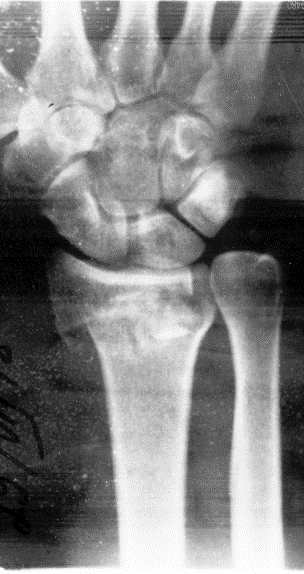


Рис.1.

а б в

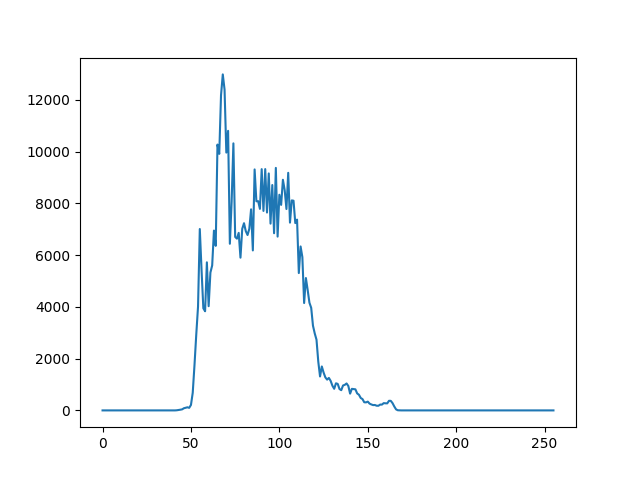
а – исходное изображение; б – откорректированное изображение с проработкой необходимых деталей; в – откорректированное изображение с потерей деталей

1. Выполнение работы
2. Анализ исходного изображения

Таблица 1

|  |  |
| --- | --- |
| изображение | test |
| k, bpp | 24 |
| m x n, pix | 528 x 996 |
| Гистограмма | C:\Users\teacher\.vscode\project\home_work\obrab _izobr\lab\img\bc_test.png |
| Контраст | 126 |

1. Гистограммы исходного изображения



1. Выбрать для коррекции 4 метода преобразования

Нормализация, эквализация, логарифмическое и степенное.

1. Выбрать параметры преобразований

Параметры преобразовани представлены в таблице 2

1. Применить к исходному изображению выбранные преобразования, внести в таблицу 1

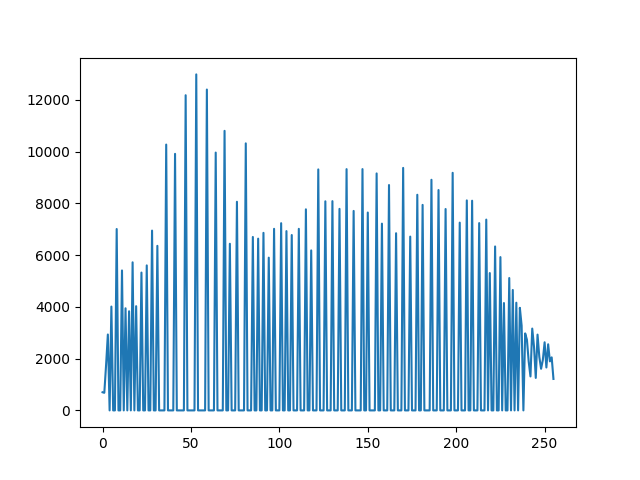
Таблица 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Преобразование | Нормализация | Эквализация | Логарифмическое | Степенное |
| Параметры преобразования | (−128; 3) | - | 255 и 100 | 0,5 и 1,5 |
| Изображение после преобразования | C:\Users\teacher\.vscode\project\home_work\obrab _izobr\lab\img\test_img_norm.png | C:\Users\teacher\.vscode\project\home_work\obrab _izobr\lab\img\test__img_ekv.png | C:\Users\teacher\.vscode\project\home_work\obrab _izobr\lab\img\test_log_y.pngC:\Users\teacher\.vscode\project\home_work\obrab _izobr\lab\img\test_log_x.png | C:\Users\teacher\.vscode\project\home_work\obrab _izobr\lab\img\test_deg_x.pngC:\Users\teacher\.vscode\project\home_work\obrab _izobr\lab\img\test_deg_y.png |
| Контраст изображения, к | 250 | 255 | 100 | 100  78 |

1. Выбрать оптимальный вариант преобразования и обосновать его

Оптимальный метод – это эквализация. При этом методе, изображение является максимально контрастным, при этом все детали изображения различимы, потеря информация отсутствует.

1. Гистограмма изображения после выбранного варианта преобразования



1. Вывод по работе

Проведена градационная коррекция с учетом исходных параметров изображения и заданных параметров к откорректированному изображению. Проведено сравнение разных методов коррекции, выбран оптимальный - эквализация

1. Используемый язык программирования и код программы с комментариями

Язык: Python 3.11.1

Код программы:

import numpy as np

import cv2

import matplotlib.pyplot as plt

#исходное изображение

test\_img = cv2.imread('img/test.jpg')

#Чрно-белое изображение

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

#запись черно-белого изображения в папку

cv2.imwrite('img/bw\_img.png', bw\_img)

# Вычислить гистограмму

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

#Отобразим график

plt.plot(bc\_test)

plt.show()

#Запишем график в папку

plt.plot(bc\_test)

plt.savefig('img/bc\_test.png')

plt.close()

#Нормализация гистограмм

#Вычисление гистограммы набора данных

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

#Вычисление cdf

cdf\_test = hist\_test.cumsum()

#Формула распределения

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

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

#Генерируем изображение после нормализации гистограммы

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

img\_norm\_test = cdf\_test[bw\_img]

#Записываем нормализированные изображения в папку

cv2.imwrite('img/test\_img\_norm.png', img\_norm\_test)

#вычисляем гистограмму

bc\_norm\_test, bins\_test1 = np.histogram(img\_norm\_test, 256)

#Вывод графика

plt.plot(bc\_norm\_test)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_norm\_test)

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

plt.close()

#Эквализация

ekv\_test = cv2.equalizeHist(bw\_img.copy())

#сохранения изображений в папку

cv2.imwrite('img/test\_\_img\_ekv.png', ekv\_test)

#вычисление гистограмм

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

#Вывод графика

plt.plot(bc\_ekv\_test)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_ekv\_test)

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

plt.close()

#Логарифмические преобразование

#Функция логарифмического преобразования

def log(r, c):

    max = np.max(r)

    log\_max = np.log(1 + max)

    s = c \* (np.log(1 + r)/log\_max)

    s = np.array(s, dtype=np.uint8)

    return s

#Применение функции к изображению

deg\_test\_x = log(test\_img.copy(), 255)

log\_test\_y = log(test\_img.copy(), 100)

#запись изображений в папку

cv2.imwrite('img/test\_log\_x.png', deg\_test\_x)

cv2.imwrite('img/test\_log\_y.png', log\_test\_y)

#рассчитаем гистограммы

bc\_deg\_x = cv2.calcHist([deg\_test\_x], [0], None, [256], [0, 256])

bc\_log\_y = cv2.calcHist([log\_test\_y], [0], None, [256], [0, 256])

#Вывод графика

plt.plot(bc\_deg\_x)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_deg\_x)

plt.savefig('img/bc\_deg\_x.png')

#Вывод графика

plt.plot(bc\_log\_y)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_log\_y)

plt.savefig('img/bc\_log\_y.png')

#Степенное преобразование

#Функция степенного преобразования

def degree(r, y):

    table = np.array([((i / 255.0) \*\* y) \* 255 for i in np.arange(0,256)]).astype("uint8")

    corrected\_gamma = cv2.LUT(r, table)

    return corrected\_gamma

#Применение функции к изображениям

#γ>1

deg\_test\_x = degree(test\_img.copy(), 1.5)

#γ<1

deg\_test\_y= degree(test\_img.copy(), 0.5)

#запись изображений в папку

cv2.imwrite('img/test\_deg\_x.png', deg\_test\_x)

cv2.imwrite('img/test\_deg\_y.png', deg\_test\_y)

#рассчитаем гистограммы

bc\_deg\_x = cv2.calcHist([deg\_test\_x], [0], None, [256], [0, 256])

bc\_deg\_y = cv2.calcHist([deg\_test\_y], [0], None, [256], [0, 256])

#Вывод графика

plt.plot(bc\_deg\_x)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_deg\_x)

plt.savefig('img/bc\_deg\_x.png')

#Вывод графика

plt.plot(bc\_deg\_y)

plt.show()

#Сохранения графика в папку

plt.plot(bc\_deg\_y)

plt.savefig('img/bc\_deg\_y.png')