**Дата: 12.06.23**

**ФИО: Козлов Евгений Юрьевич**

**Группа: 224-322**

**ЛАБОРАТОРНАЯ РАБОТА №****1**  
**Выбор параметров градационной коррекции на основе требований к конечному изображению**

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

**2. Содержание работы**

1. Проанализировать предложенное изображение по следующим параметрам: глубина цвета, разница между максимальной и минимальной светлотой,

2. Построить гистограмму изображения

3. Рассмотреть возможные варианты коррекции изображения которые позволят подчеркнуть детали изображения, содержащие важную информацию о переломе

4. Выбрать оптимальный вариант коррекции из рассмотренных в п. 4

5. Построить гистограмму изображения после коррекции и сравнить с гистограммой, полученной в п. 2

6. Оценить контраст откорректированного изображения

**3. Исходные данные и программное обеспечение**

Используемая среда программирования: Visual Studio Code Используемый язык программирования: Python 3.11.1 64-bit Используемые библиотеки: numpy, scipy, skimage, matplotlib

X-ray of a hand and knee joint

Description automatically generated with low confidenceA cat in a tree

Description automatically generated with low confidenceA picture containing outdoor, land vehicle, vehicle, road

Description automatically generated

­

**4. Выполнение работы**

1. Анализ входных данных.

|  |  |  |  |
| --- | --- | --- | --- |
| **Изоб-ражение** | **Перелом** | **Закат** | **Кот** |
| Разре-шение | 528 px× 996px | 3872 px× 2592px | 3872px × 2592px |
| Глубина цвета | 24 бита | 24 бита | 24 бита |
| Гистог-рамма  светлости |  |  |  |
| Контраст | 126 | 255 | 255 |

1. Перелом: гистограмма почти целиком занимает уровни 50-150, что свидетельствует о низком контрасте, недостатке уровня чёрного и белого.

Степенной и логарифмический метод могут повысить контраст, но это приведёт к потере деталей. Гистограмму необходимо эквализировать, чтобы сделать различимыми наибольшее число уровней светлот.

|  |  |  |
| --- | --- | --- |
| Преобразование | **Эквализация гистограммы** | Нормализация гистограммы |
| Параметры | — | (−128; 3) |
| Результат |  |  |
| Контраст | 255 | 250 |

Для контроля изображение также подверглось нормализации гистограммы. Видно, что эквализированное изображение получилось максимально контрастным, при этом все детали различимы, пересвета и потери информации не наблюдается, в области перелома чётко различимы все контуры.

|  |  |
| --- | --- |
| Исходная гистограмма | Гистограмма после нормализации |
|  |  |

|  |  |
| --- | --- |
| Исходная гистограмма | Гистограмма после эквализации |
|  |  |

1. Закат: гистограмма отражает недоэкспонирование нижней половины снимка и большое количество информации о средних и средне-светлых тонах, что свидетельствует о том, что городской пейзаж не утерян и может быть восстановлен. Поскольку на гистограмме так или иначе представлен весь спектр, можно попробовать «выровнять» его логарифмическим или степенным преобразованием (γ < 1), либо же попробовать устраниить пик в районе уровня 100 эквализацией.

|  |  |  |  |
| --- | --- | --- | --- |
| Преобра-зование | Эквализация гистограммы | Степенное преобразование | **Логарифмическое преобразование** |
| Параметры | — | γ = 0.5 | ­— |
| Результат |  |  |  |
| Гисто-грамма |  |  |  |
| Контраст | 255 | 255 | 255 |

Эквализация даёт более насыщенные тёмные тона, но, при этом теряются детали: машины на дальнем плане слаборазличимы. Степенное преобразование устранило тени на снимке.

Логарифмическое дало средний между этими преобразованиями результат: чуть сильнее выраженные, чем при степенном, тени и больше различимых деталей.

1. Кот: слишком низкая доля тёмных пикселей относительно большого пика яркостей в районе уровней 220-255. Гистограмма довольно ровная, не считая большого пика светлот. Будут использоваться логарифмическое и степенное преобразование.

Логарифмическое преобразование дало более детальное изображение, что делает это преобразование более подходящим, чем степенное. Однако, при этом изменился естественный цвет заднего фона.

Также была выполнена эквализация, которая дала чуть менее различимый нос, но более насыщенные чёрные тона, высокий визуальный контраст и лучшее разделение объекта и фона.

|  |  |  |  |
| --- | --- | --- | --- |
| Преобра-зование | Степенное преобразование | Логарифмическое преобразование | **Эквализация** |
| Параметры | γ = 0.5 | ­— | — |
| Результат |  |  |  |
| Гисто-граммы |  |  |  |
| Контраст | 255 | 255 |  |

Изображения размещены на гугл-диске по адресу:

<https://drive.google.com/drive/folders/1eWwBdnTe72SEMsqDvlygdFTBjz6ySzyh?usp=sharing>

**Код работы**

import numpy as np

import cv2

import scipy as sp

import matplotlib.pyplot as plt

from skimage.io import imread, imshow, imsave

from skimage import data, img\_as\_float

# Изображения

# 1. Подобрать изображения (из практической 2)

INIT\_IMG\_1 = cv2.imread('img/init/01.jpg')

INIT\_IMG\_2 = cv2.imread('img/init/02.tif')

INIT\_IMG\_3 = cv2.imread('img/init/03.jpg')

# 2. Перевести изображения в черно-белые

GRAY\_IMG\_1 = cv2.cvtColor(INIT\_IMG\_1.copy(), cv2.COLOR\_BGR2GRAY)

GRAY\_IMG\_2 = cv2.cvtColor(INIT\_IMG\_2.copy(), cv2.COLOR\_BGR2GRAY)

GRAY\_IMG\_3 = cv2.cvtColor(INIT\_IMG\_3.copy(), cv2.COLOR\_BGR2GRAY)

cv2.imwrite('img/dist/1/GRAY\_IMG\_1.png', GRAY\_IMG\_1)

cv2.imwrite('img/dist/1/GRAY\_IMG\_2.png', GRAY\_IMG\_2)

cv2.imwrite('img/dist/1/GRAY\_IMG\_3.png', GRAY\_IMG\_3)

# 3. Вычислить гистограммы

def save\_fig(fig, path=''):

    plt.plot(fig)

    plt.savefig(path)

    plt.close()

# Построить гистограмму изображения

GIST\_GRAY\_1 = cv2.calcHist([GRAY\_IMG\_1], [0], None, [256], [0, 256])

GIST\_GRAY\_2 = cv2.calcHist([GRAY\_IMG\_2], [0], None, [256], [0, 256])

GIST\_GRAY\_3 = cv2.calcHist([GRAY\_IMG\_3], [0], None, [256], [0, 256])

save\_fig(GIST\_GRAY\_1, 'img/dist/1/GIST\_GRAY\_1.png')

save\_fig(GIST\_GRAY\_2, 'img/dist/1/GIST\_GRAY\_2.png')

save\_fig(GIST\_GRAY\_3, 'img/dist/1/GIST\_GRAY\_3.png')

# Провести нормализацию

def normalize(in\_img):

    hist\_before\_1, bins\_before\_1 = np.histogram(in\_img, 256)

    cdf\_1 = hist\_before\_1.cumsum()

    cdf\_1 = (cdf\_1-cdf\_1[0])\*255/(cdf\_1[-1]-1)

    normalized = np.zeros((384, 495, 1), dtype =np.uint8)

    normalized = cdf\_1[GRAY\_IMG\_1]

    return normalized

NORMALIZED\_1 = normalize(GRAY\_IMG\_1)

GIST\_NORM\_1, bins\_after\_1 = np.histogram(NORMALIZED\_1, 256)

cv2.imwrite('img/dist/2/NORMALIZED\_1.png', NORMALIZED\_1)

save\_fig(GIST\_NORM\_1, 'img/dist/2/GIST\_NORM\_1.png.png')

# Эквализация

EQUALIZED\_1 = cv2.equalizeHist(GRAY\_IMG\_1.copy())

cv2.imwrite('img/dist/2/EQUALIZED\_1.png', EQUALIZED\_1)

GIST\_EQ\_1 = cv2.calcHist([EQUALIZED\_1], [0], None, [256], [0, 256])

save\_fig(GIST\_EQ\_1, 'img/dist/2/GIST\_EQ\_1.png')

# Город

# Эквализация

def run\_histogram\_equalization(image, path\_img, path\_gist):

    # convert from RGB color-space to YCrCb

    ycrcb\_img = cv2.cvtColor(image, cv2.COLOR\_BGR2YCrCb)

    # equalize the histogram of the Y channel

    ycrcb\_img[:, :, 0] = cv2.equalizeHist(ycrcb\_img[:, :, 0])

    # convert back to RGB color-space from YCrCb

    equalized\_img = cv2.cvtColor(ycrcb\_img, cv2.COLOR\_YCrCb2BGR)

    cv2.imwrite(path\_img, equalized\_img)

    gist\_eq = cv2.calcHist([equalized\_img], [2], None, [256], [0, 256])

    save\_fig(gist\_eq, path\_gist)

run\_histogram\_equalization(INIT\_IMG\_2, 'img/dist/3/EQUALIZED\_2.png', 'img/dist/3/GIST\_EQ\_2.png')

# Степенное преобразование

def deg\_rgb(img, y):

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

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

    return corrected\_gamma

DEG\_LESS\_SAT\_2 = deg\_rgb(INIT\_IMG\_2.copy(), 0.5)

cv2.imwrite('img/dist/3/DEG\_LESS\_SAT\_2.png', DEG\_LESS\_SAT\_2)

GIST\_DEG\_LESS\_SAT\_2 = cv2.calcHist([DEG\_LESS\_SAT\_2], [2], None, [256], [0, 256])

save\_fig(GIST\_DEG\_LESS\_SAT\_2, 'img/dist/3/GIST\_DEG\_LESS\_SAT\_2.png')

# Логарифмическое преобразование

LOG\_IMG = INIT\_IMG\_2

LOG\_IMG[LOG\_IMG==255]=254 # делаем такое преобразование, чтобы 0 не попал под логарифм

LOG\_IMG\_2=np.asarray(np.rint(255\*np.log(1+LOG\_IMG)/np.log(255)),dtype=np.uint8)

cv2.imwrite('img/dist/3/GIST\_NORM\_1.png', LOG\_IMG\_2) #результат логарифмического преобразования

GIST\_LOG\_2, bins\_log\_2 = np.histogram(LOG\_IMG\_2, 256)

save\_fig(GIST\_LOG\_2, 'img/dist/3/GIST\_LOG\_2.png')

# Кот

# Эквализация

run\_histogram\_equalization(INIT\_IMG\_3, 'img/dist/4/EQUALIZED\_3.png', 'img/dist/4/GIST\_EQ\_3.png')

# Степенное преобразование

DEG\_LESS\_SAT\_3 = deg\_rgb(INIT\_IMG\_3.copy(), 0.5)

cv2.imwrite('img/dist/4/DEG\_LESS\_SAT\_3.png', DEG\_LESS\_SAT\_3)

GIST\_DEG\_LESS\_SAT\_3 = cv2.calcHist([DEG\_LESS\_SAT\_3], [2], None, [256], [0, 256])

save\_fig(GIST\_DEG\_LESS\_SAT\_3, 'img/dist/4/GIST\_DEG\_LESS\_SAT\_3.png')

# Логарифмическое преобразование

LOG\_IMG = INIT\_IMG\_3

LOG\_IMG[LOG\_IMG==255]=254 # делаем такое преобразование, чтобы 0 не попал под логарифм

LOG\_IMG\_3=np.asarray(np.rint(255\*np.log(1+LOG\_IMG)/np.log(255)),dtype=np.uint8)

cv2.imwrite('img/dist/4/GIST\_NORM\_1.png', LOG\_IMG\_3) #результат логарифмического преобразования

GIST\_LOG\_3, bins\_log\_2 = np.histogram(LOG\_IMG\_3, 256)

save\_fig(GIST\_LOG\_3, 'img/dist/4/GIST\_LOG\_3.png')