

**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.

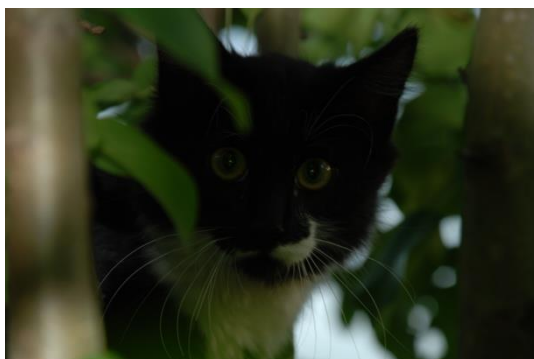
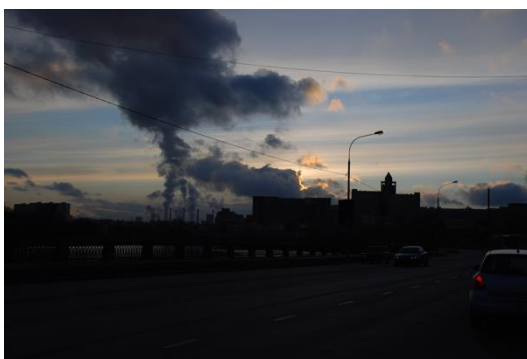
Сравнить разные методы коррекции, выбрать оптимальный

#### **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

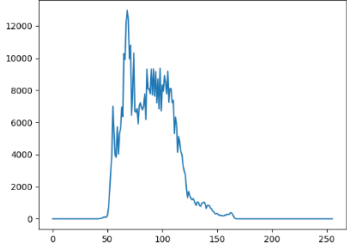
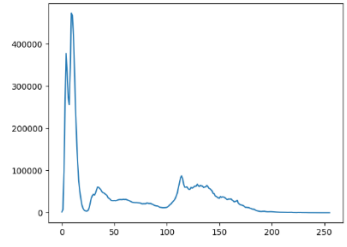
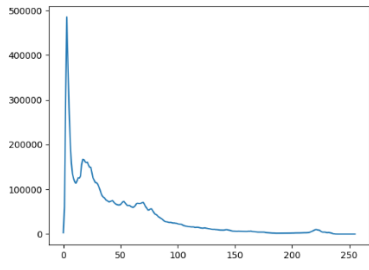
#### **3. Исходные данные и программное обеспечение**

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





## 4. Выполнение работы

### 1. Анализ входных данных.

Изображение	Перелом	Закат	Кот
Разрешение	528px×996px	3872px×2592px	3872px×2592px
Глубина цвета	24 бита	24 бита	24 бита
Гистограмма светлости			
Контраст	126	255	255

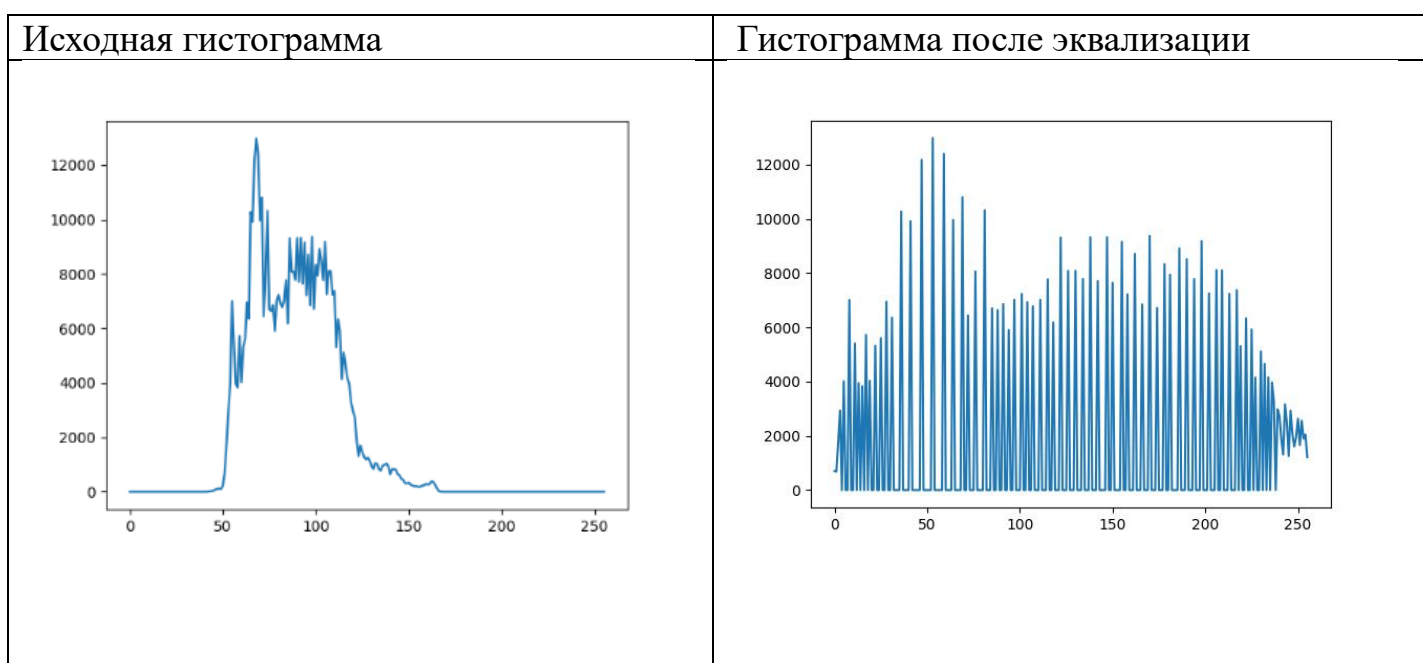
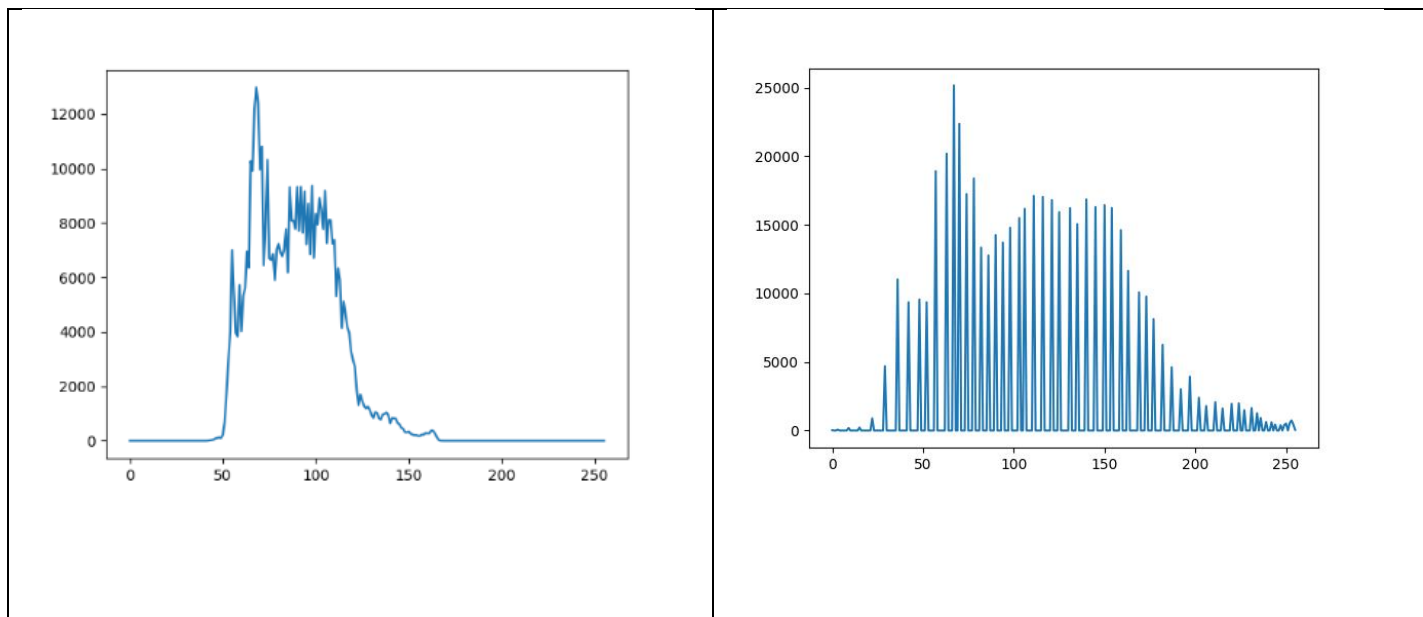
2. Перелом: гистограмма почти целиком занимает уровни 50-150, что свидетельствует о низком контрасте, недостатке уровня чёрного и белого.

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




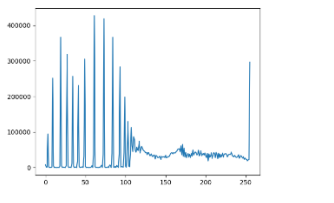
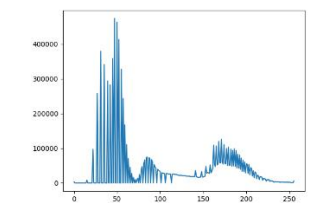
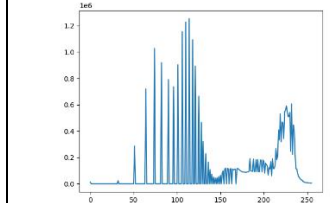
Преобразование	Эквализация гистограммы	Нормализация гистограммы
Параметры	—	(−128; 3)
Результат		
Контраст	255	250

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
----------------------	--------------------------------



3. 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 ( $\gamma < 1$ ), либо же попробовать устранить пик в районе уровня 100 эквализацией.

Преобразование	Эквализация гистограммы	Степенное преобразование	Логарифмическое преобразование
Параметры	—	$\gamma = 0.5$	—
Результат			
Гистограмма			
Контраст	255	255	255

Эквализация даёт более насыщенные тёмные тона, но, при этом теряются детали: машины на дальнем плане слабо различимы.

Степенное преобразование устранило тени на снимке.




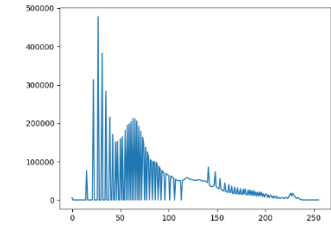
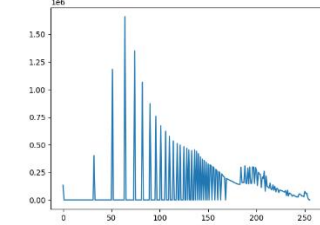
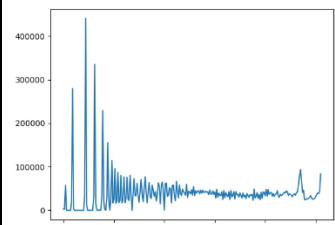
Логарифмическое дало средний между этими преобразованиями результат: чуть сильнее выраженные, чем при степенном, тени и больше различимых деталей.

4. 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.

Преобразование	Степенное преобразование	Логарифмическое преобразование	Эквализация
Параметры	$\gamma = 0.5$	—	—
Результат			
Гистограммы			
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')

# Эквиализация

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

```

```
# Эквиализация
```

```
def run_histogram_equalization(image, path_img, path_gist):  
    # convert from RGB color-space to YCrCb  
    ycrb_img = cv2.cvtColor(image, cv2.COLOR_BGR2YCrCb)  
    # equalize the histogram of the Y channel  
    ycrb_img[:, :, 0] = cv2.equalizeHist(ycrb_img[:, :, 0])  
    # convert back to RGB color-space from YCrCb  
    equalized_img = cv2.cvtColor(ycrb_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 по-  
пал 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 логариф-  
мического 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
```

```
# Эквиализация
```

```
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 по-
пал 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 логариф-
мического 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')
```