Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Отчет по лабораторной работе №1

«Гистограммы, профили и проекции»

По дисциплине «Техническое зрение»

Выполнил: Курчавый В. В.,

студент группы R3338

Преподаватель: Шаветов С. В.,

канд. техн. Наук, доцент ФСУ и Р

Санкт-Петербург, 2022

Цель работы

Освоение основных яркостных и геометрических характеристик изображений и их использование для анализа изображений.

Теоретическое обоснование применимых методов

Фотография описывается двумерным массивом значений интенсивностей (яркостей) - $I.I(x, y)$ – означает значение интенсивности в координатах x, y. Элемент массива соответствует пикселю.

Гистограмма — это распределение частоты встречаемости пикселей одинаковой яркости на изображении.

Яркость — это среднее значение интенсивности сигнала. Контраст — это интервал значений между минимальной и максимальной яркостями изображения

Профиль вдоль линии — это функция интенсивности изображения, распределенного вдоль данной линии (прорезки).

Проекция на ось — это сумма интенсивностей пикселей изображения, взятая в направлении перпендикулярном данной оси.

Различные способы выравнивания гистограммы:

Линейный сдвиг:

$$I_{new}(x, y) = I(x, y) + 60$$

Растяжение динамического диапазона:

$$I_{new} = \left(\frac{I - I_{min}}{I_{max} - I_{min}}\right)^{\alpha}$$

Равномерное преобразование:

$$I_{new} = (I_{max} - I_{min}) * P(I) + I_{min}$$
$$P(I) = \sum_{m=0}^{I} Hist(I)$$

Экспоненциальное преобразование:

$$I_{new} = I_{min} - \frac{1}{\alpha}\ln(1 - P(I))$$

Преобразование по закону Рэлея:

$$I_{new} = I_{min} + \left(2\alpha^2 \ln\frac{1}{1 - P(I)}\right)^{\frac{1}{2}}$$

Преобразование по закону $^2/_3$:

$$I_{new} = (P(I))^{2/3}$$

Преобразование по гиперболическому преобразованию:

$$I_{new} = \alpha^{P(I)}$$

LUT:

$$\mathrm{I}_{new} = \mathrm{LUT[I]}$$

Профиль строки изображения:

$Profile\ i(x) = I(x, i)$, где i — номер строки изображения I.

Профиль столбца изображения:

$Profile\ j(x) = I(j, y)$, где j — номер столбца изображения I.

Вертикальная проекция на ось Ox:

$$Proj\ X(y) = \sum_{y\ =\ 0}^{\dim Y - 1} I(x, y)$$

Горизонтальная проекция на ось Oy:

$$Proj\ Y(x) = \sum_{x\ =\ 0}^{\dim X - 1} I(x, y)$$

Проекция на ось Oe, где $e = (e_x,\ e_y)$ – вектор единичной длинны:

$$Proj\ E(t) = \sum_{xe_x + ye_y\ =\ t} I(x, y)$$

Ход выполнения работы

1. Гистограммы

1.1 Исходные данные

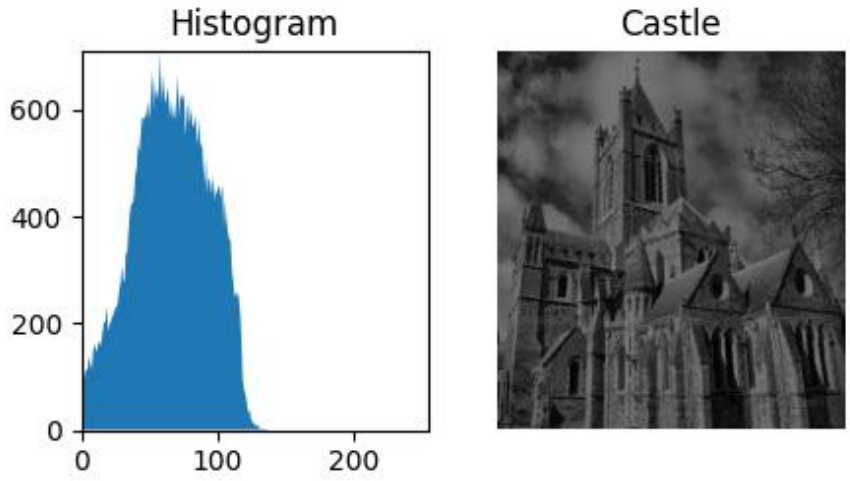

Рисунок 1. Исходное изображение.

Контрастность изображения:

$$K = I_{max} - I_{min} = 137 - 0 = 137$$

Изображение темное с небольшим контрастом, из-за этого гистограмма смещена влево.

Листинг 1. Считывание изображения и построение гистограммы.

```
img = cv.imread(PATH_TO_PHOTO, cv.IMREAD_GRAYSCALE)
hist = cv.calcHist([image], [0], None, [256], [0, 256])
```

Листинг 2. Совместное изображение гистограммы и изображения

```
fig, (h, i) = plt.subplots(1, 2, figsize=(5,2.5))
h.set_title('Histogram')
h.set_xlim([0, 255])
h.set_ylim([0, int(np.max(hist))])
h.fill_between(list(range(256)), 0, list(map(int, hist)))
i.set_title('Castle')
i.imshow(img, cmap='gray', vmin=0, vmax=255, aspect='auto')
i.axis('off')
plt.show()
```

1.2 Линейное преобразование

Увеличим каждое значение интенсивности на 60, чтобы изображение стало более светлым.

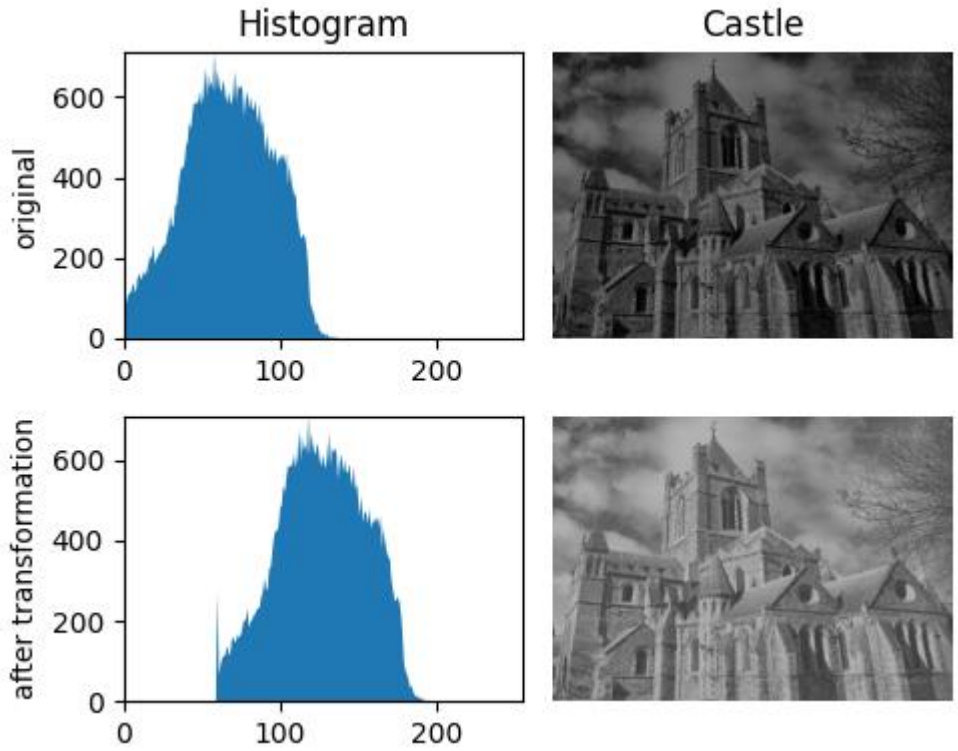

Рисунок 2. Влияние линейного сдвига.

Как и ожидалось изображение стало более светлым, но при этом не осталось очень темных тонов.

Листинг 3. Функция линейного сдвига.

```
def linear_transform(img, shift=0):
    return img + shift
```

1.3 Растяжение динамического диапазона

Применим растяжение динамического диапазона с параметром с различными α.

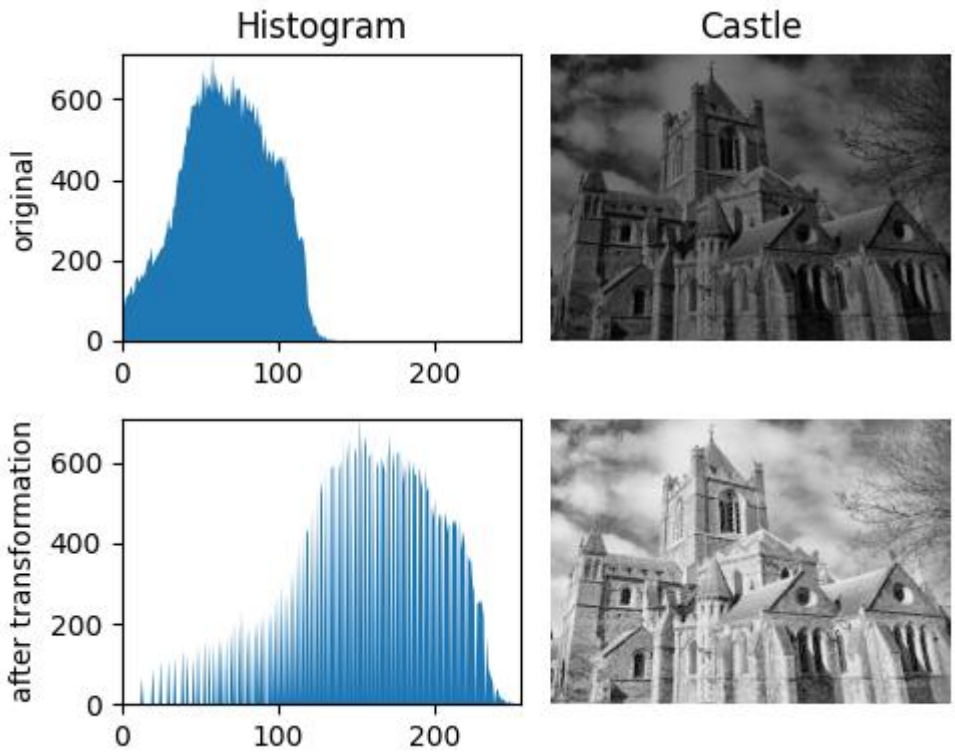

Рисунок 3. Влияние динамического растяжение с α =0.6.

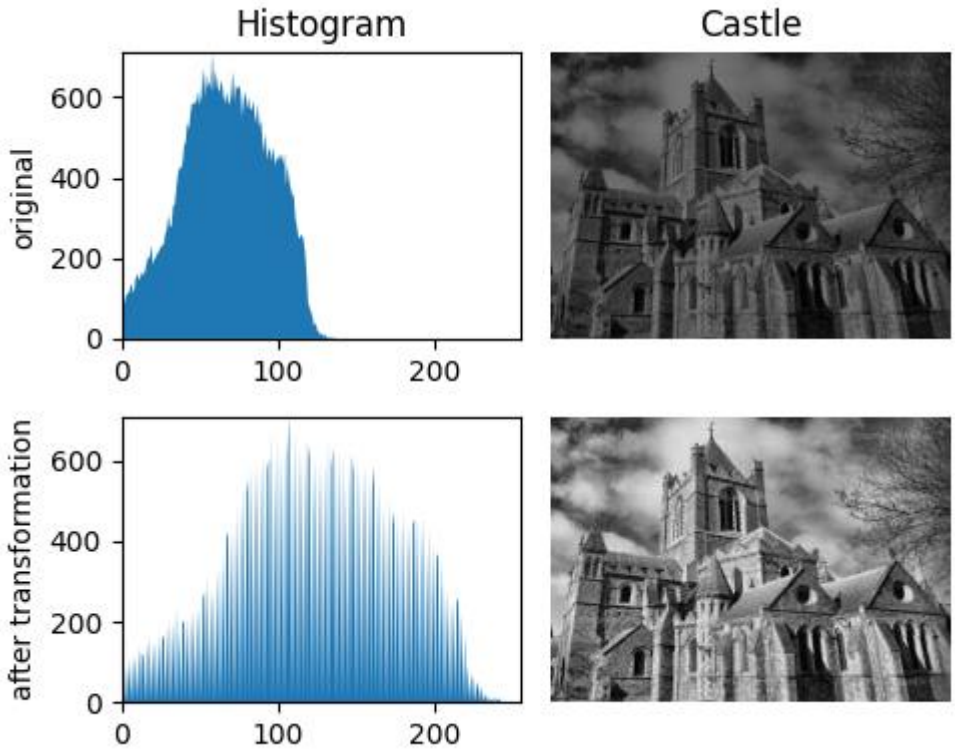

Рисунок 4. Влияние динамического растяжение с α =1.

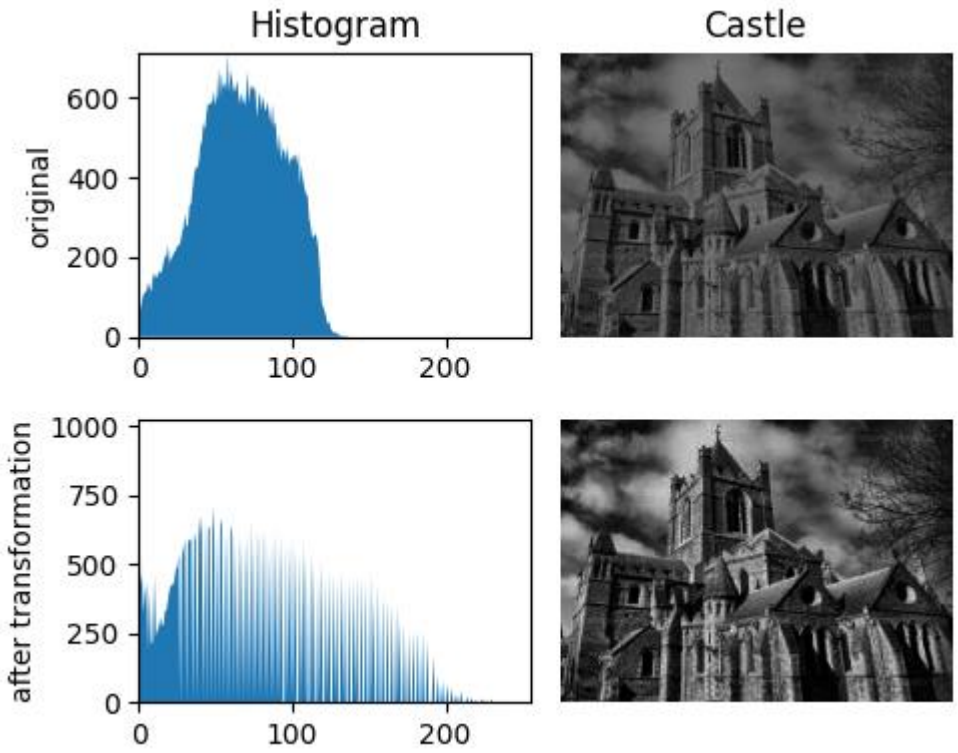

Рисунок 5. Влияние динамического растяжение с α =1.9.

Можно заметить, что при увеличении α картинка становится темнее, так как значения интенсивностей становятся меньше. При этом изображения становятся контрастнее.

Листинг 4. Функция динамического растяжения.

```
def stretching_tranform(img: ndarray, a: float = 1) -> ndarray:
    i_max = np.max(img)
    i_min = np.min(img)
    return (255*(np.power((img - i_min)/(i_max - i_min),
a))).astype(np.uint8)
```

1.4 Равномерное преобразование

Применим равномерное преобразование для исходного изображения:

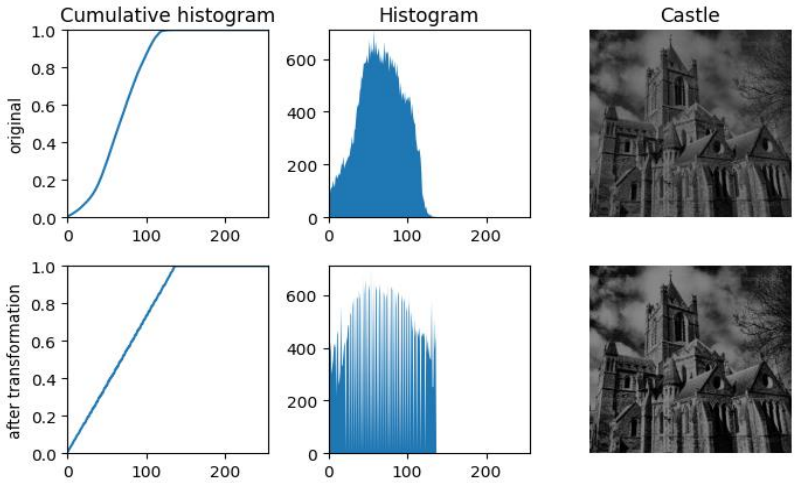

Рисунок 6. Влияние равномерного преобразования.

Можно заметить, что кумулятивная гистограмма стала линейно расти, а количество различных значений интенсивностей стало менее разбросанным. Контраст изображения не поменялся.

Листинг 5. Функция расчета нормированной кумулятивной гистограммы.
```
def cum_histogram(hist, num_rows, num_column):
    return np.cumsum(hist) / (num_rows * num_column)
```

Листинг 6. Функция расчета равномерного проебразования.
```
def uniform_transform(img: ndarray, cum_hist: ndarray) -> ndarray:
    i_max, i_min = np.max(img), np.min(img)
    new_img = ndarray(img.shape)
    for x in range(img.shape[0]):
        for y in range(img.shape[1]):
            new_img[x][y] = (i_max - i_min) * cum_hist[img[x][y]] + i_min
    return new_img.astype(np.uint8)
```

1.5 Экспоненциальное преобразование

Выполним экспоненциальное преобразование.

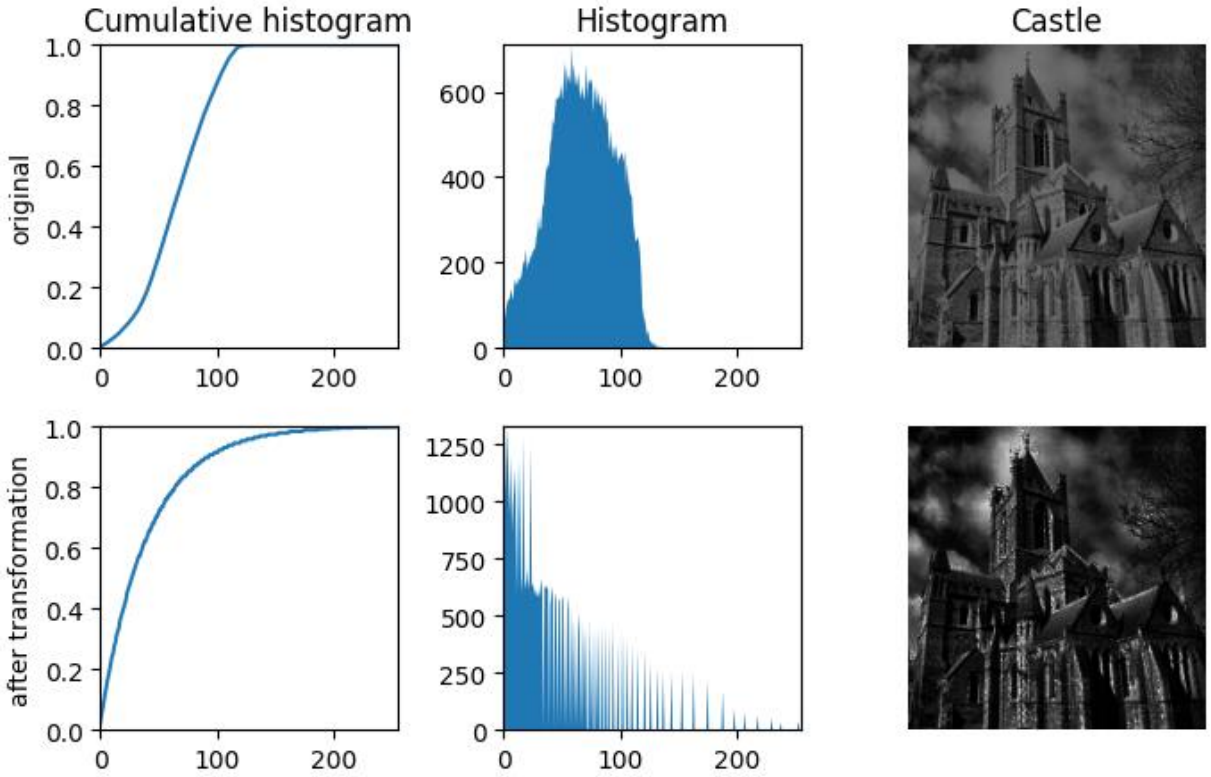

Рисунок 7. Влияние экспоненциального преобразования при $\alpha = 0.025$.

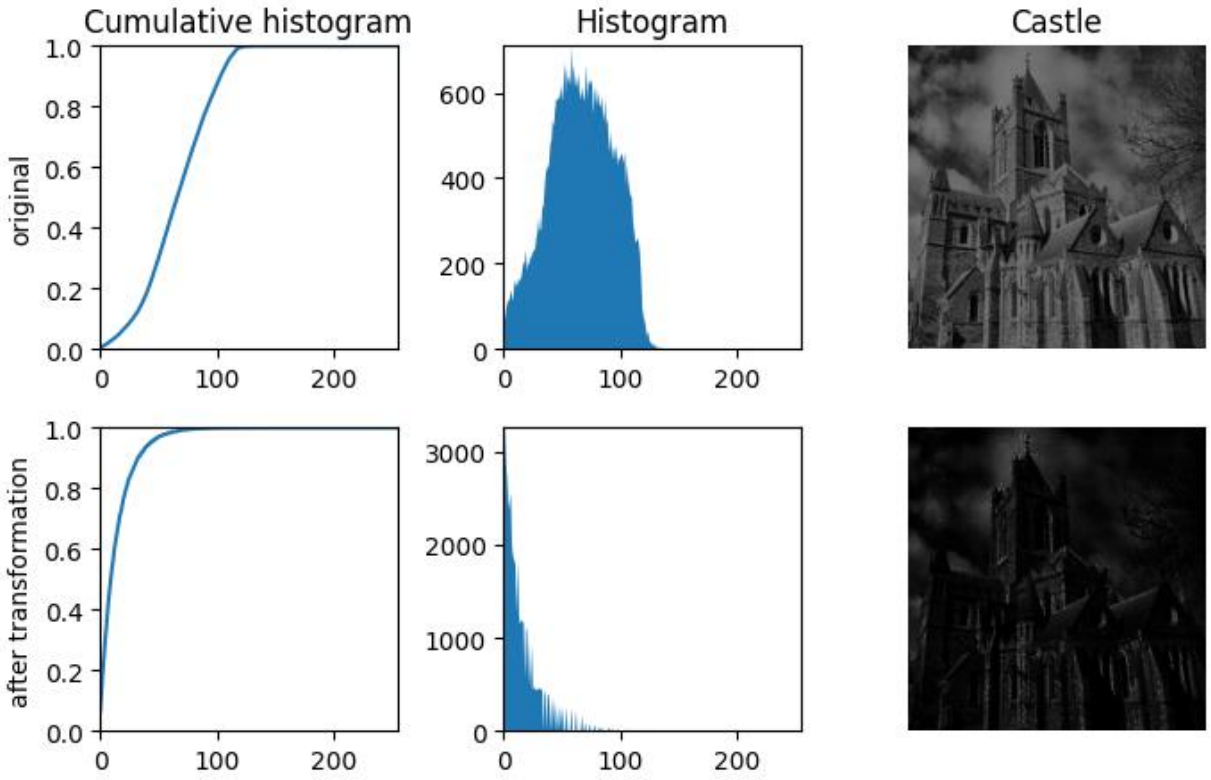

Рисунок 8. Влияние экспоненциального преобразования при $\alpha = 0.07$.

Можно заметить, что, чем меньше α, тем с более меньшей скоростью растет кумулятивная гистограмма. Кумулятивная гистограмма растет по экспоненте. Значения на обычной гистограмме сосредоточены близко к нулю, поэтому фотографии получаются в темных тонах.

Листинг 7. Функция расчета экспоненциального преобразования.

```
def exponential_transform(img: ndarray, cum_hist: ndarray, a) -> ndarray:
    i_min = np.min(img)
    new_img = ndarray(img.shape)
    for x in range(img.shape[0]):
        for y in range(img.shape[1]):
            new_img[x][y] = i_min - 1/a * np.log(1 - cum_hist[img[x][y]])
    return new_img.astype(np.uint8)
```

1.6 Преобразование по закону Рэлея

Выполним преобразование по закону Рэлея.

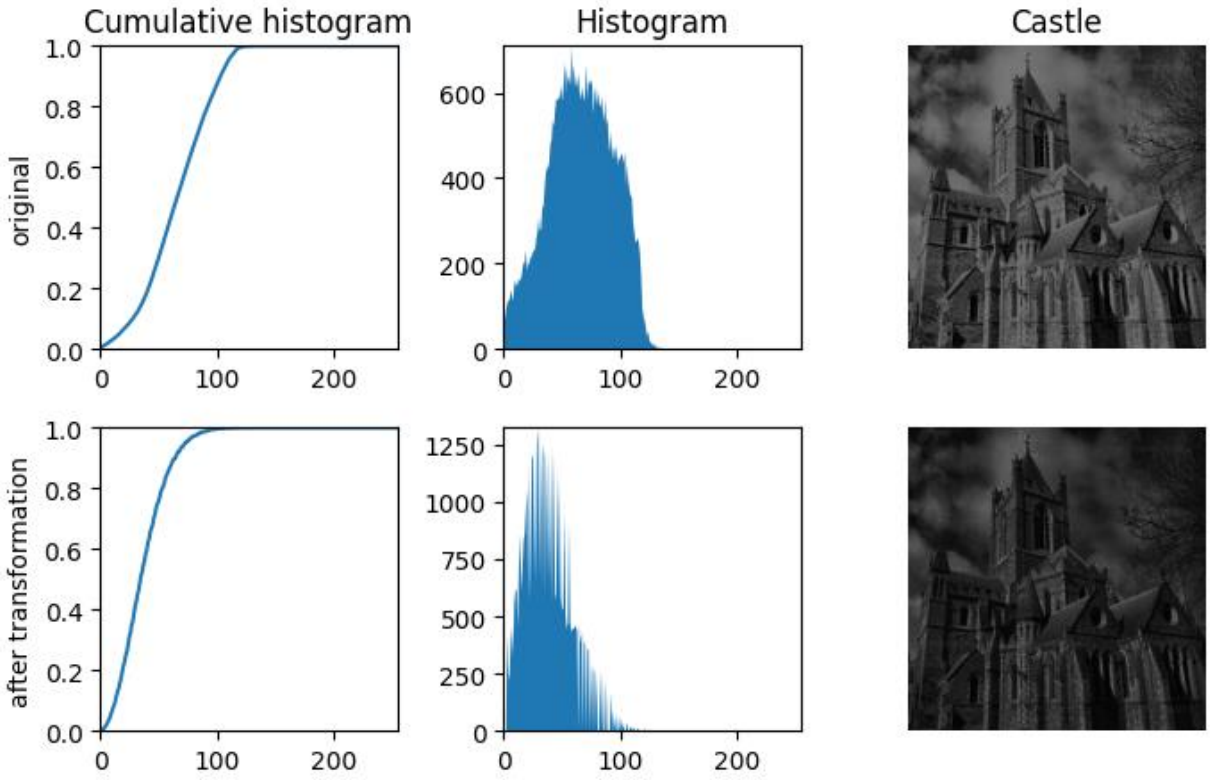

Рисунок 9. Влияние преобразования по закону Рэлея при $\alpha = 20$.

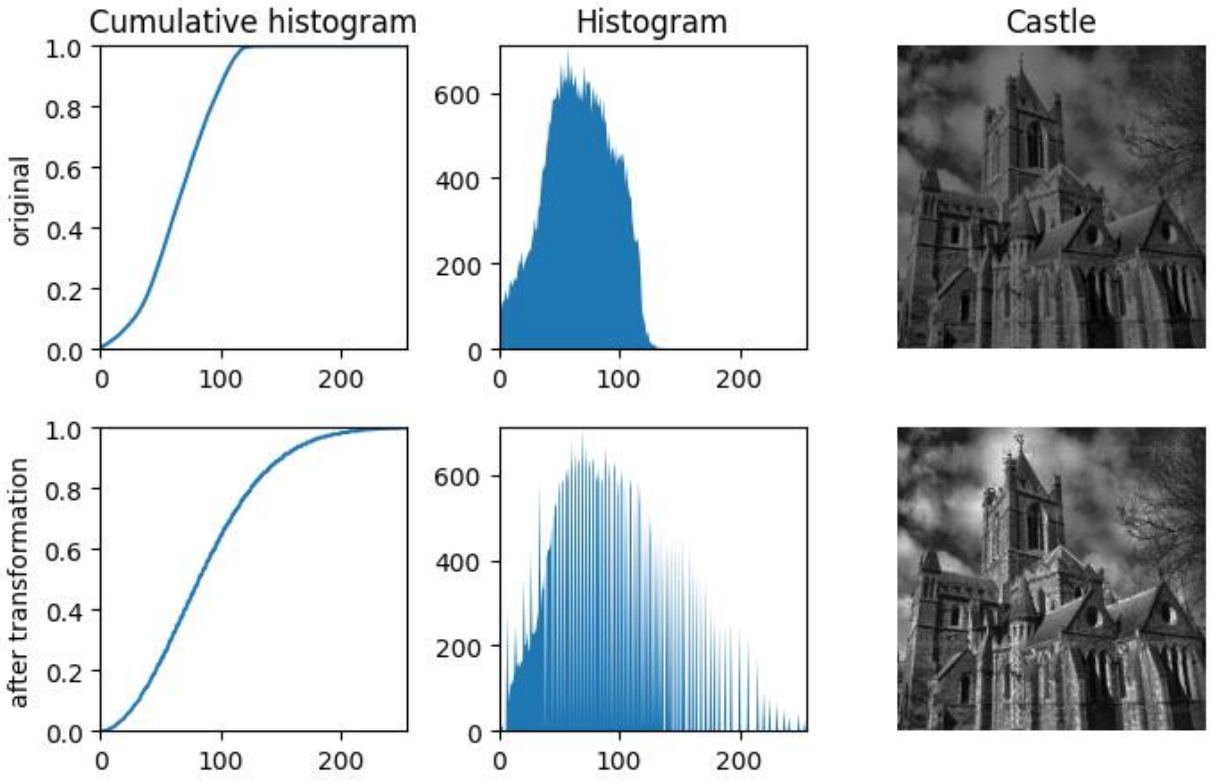

Рисунок 10. Влияние преобразования по закону Рэлея при $\alpha = 70$.

Можно заметить, что при увеличении α увеличивается контраст изображения, но только до определенного значения α, после которого, начинает теряться информация об изображении.

Листинг 8. Функция расчета преобразования по закону Рэлея.

```
def rayleigh_low_transform(img: ndarray, cum_hist, a: float = 100):
    i_min = np.min(img)
    new_img = ndarray(img.shape)
    for x in range(img.shape[0]):
        for y in range(img.shape[1]):
            new_img[x][y] = i_min + np.power(2*np.power(a, 2) * np.log(1 / (1
- cum_hist[img[x][y]])), 0.5)
    return new_img.astype(np.uint8)
```

1.7 Преобразование по закону $^2/_3$

Вычислим новое значение интенсивностей по формуле:

$$I_{new} = (P(I))^{2/3}$$

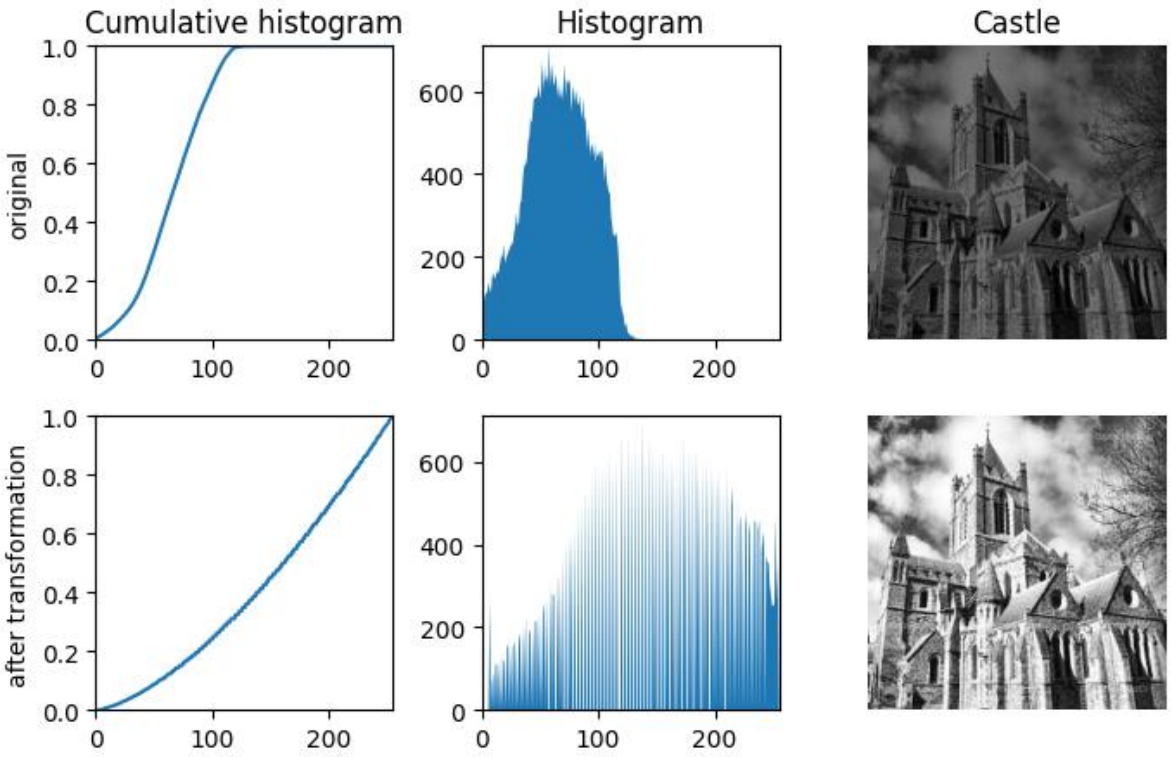

Рисунок 11. Влияние преобразования по закону $^2/_3$.

Можно заметить, что после преобразования изображение стало сильно контрастным с преобладанием светлых тонов.

Листинг 9. Функция преобразования по закону $^2/_3$.

```
def two_thirds_low_transform(img: ndarray, cum_hist: ndarray):
    new_img = ndarray(img.shape)
    for x in range(img.shape[0]):
        for y in range(img.shape[1]):
            print(cum_hist[img[x][y]])
            new_img[x][y] = 255*np.power((cum_hist[img[x][y]]), 2/3)
    return new_img.astype(np.uint8)
```

1.8 Преобразование по гиперболическому закону

Выполним преобразование по гиперболическому преобразованию:

α часто выбирают равным I_{min}.

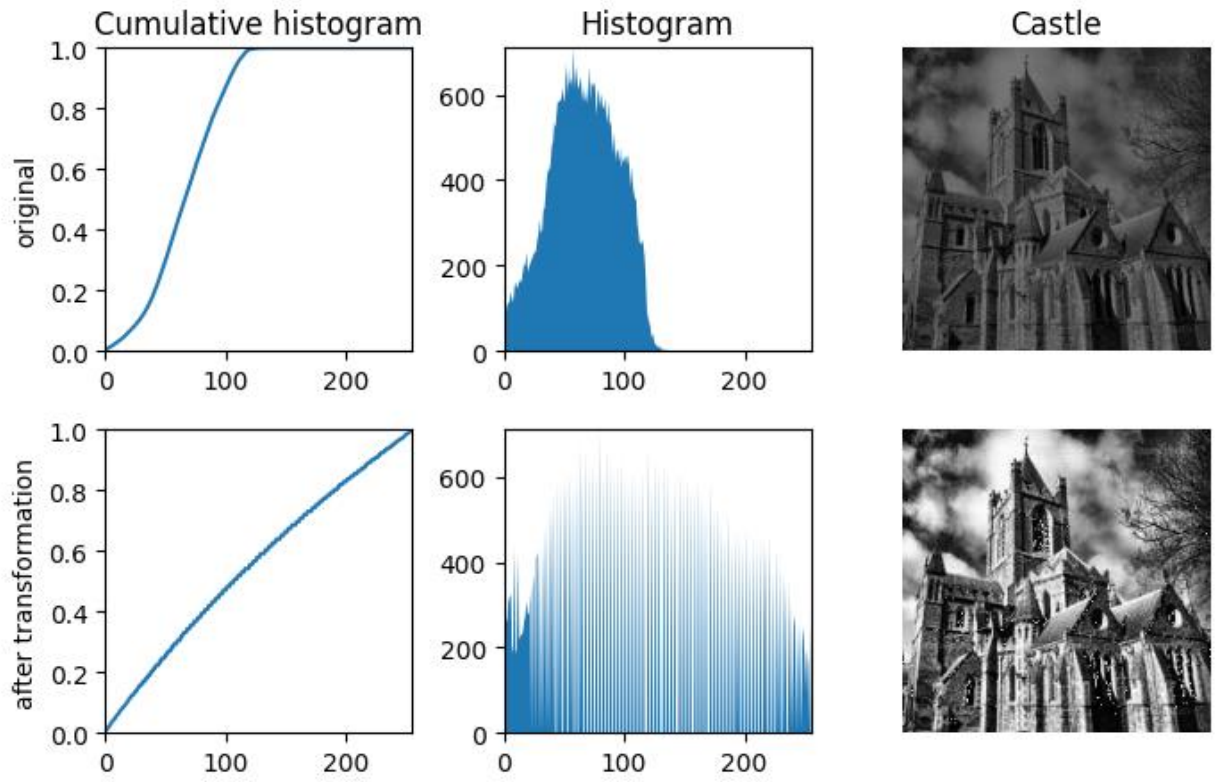

Рисунок 12. Влияние преобразования по гиперболическому закону.

Можно заметить, что после преобразования изображение стало сильно контрастным.

Листинг 10. Функция преобразования по гиперболическому закону.

```
def hyperbolic_transform(img: ndarray, cum_hist: ndarray, a=None):
    if a is None:
        a = np.min(img)
    if a == 0 or a == 1:
        a = 2
    new_img = ndarray(img.shape)
    for x in range(img.shape[0]):
        for y in range(img.shape[1]):
            print(cum_hist[img[x][y]])
            new_img[x][y] = 255*np.power(a, cum_hist[img[x][y]])
    return new_img.astype(np.uint8)
```

1.9 Преобразование с помощью встроенных в OpenCV функций

Воспользуемся функцией *equalizeHist()* из OpenCV

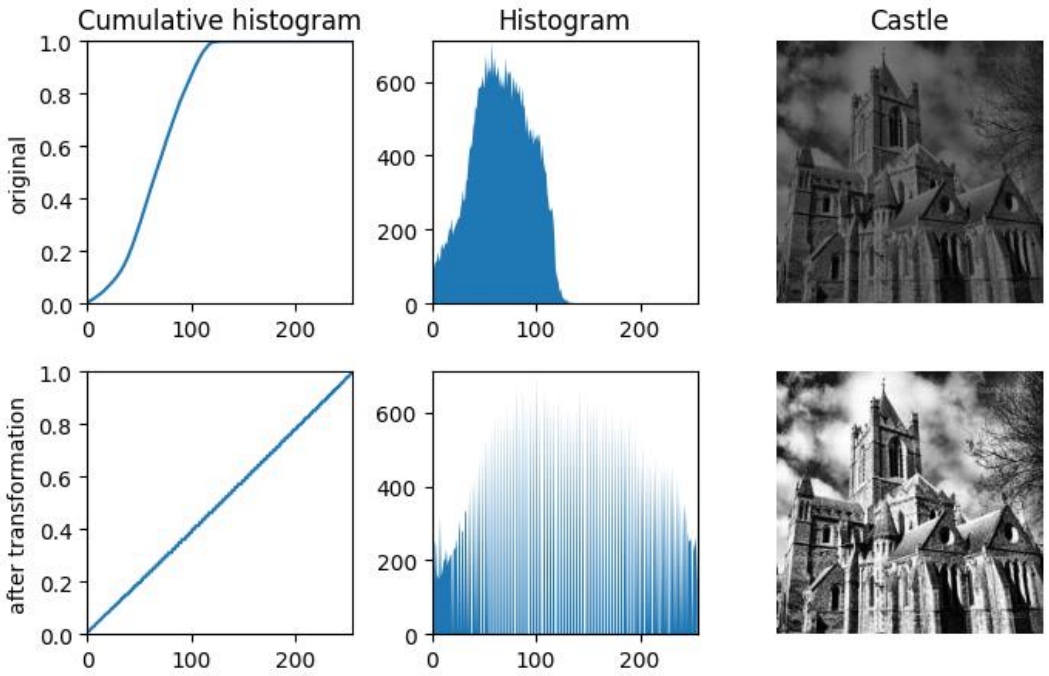

Рисунок 13. Влияние преобразования с помощью функции equalizeHist().

Преобразованные изображение контрастное. При этом кумулятивная гистограмма линейная. Реализация функции аналогична алгоритму равномерного преобразования, за исключением того, что $I_{min} = 0$, а $I_{max} = 255$, чтобы максимально увеличить контраст изображения.

Воспользуемся функцией *createCLAHE()* из OpenCV с параметрами *clipLimit = 6* и *tileGridSize = (10, 11),* и преобразуем изображение.

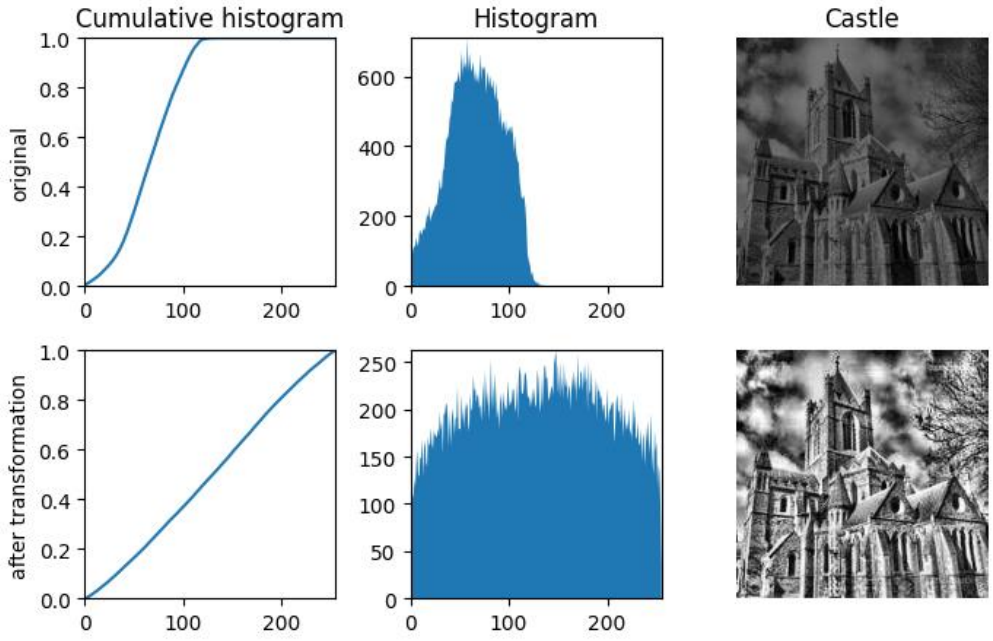

Рисунок 14. Влияние алгоритма CLAHE.

Можно заметить, что изображение стало контрастным и при этом распределение интенсивностей почти равномерно.

Листинг 11. Использование встроенных функций.

```
# CLAHE
algorithm_CLAHE = cv.createCLAHE(10, (10, 11))
image_2 = algorithm_CLAHE.apply(image)
# equalize Hist
image_3 = cv.equalizeHist(imagE)
```

1.10 Lookup-Table

Воспользуемся методом соляризации. Рассчитаем lookup-table по формуле:

$$y = 4x(255 - x)$$

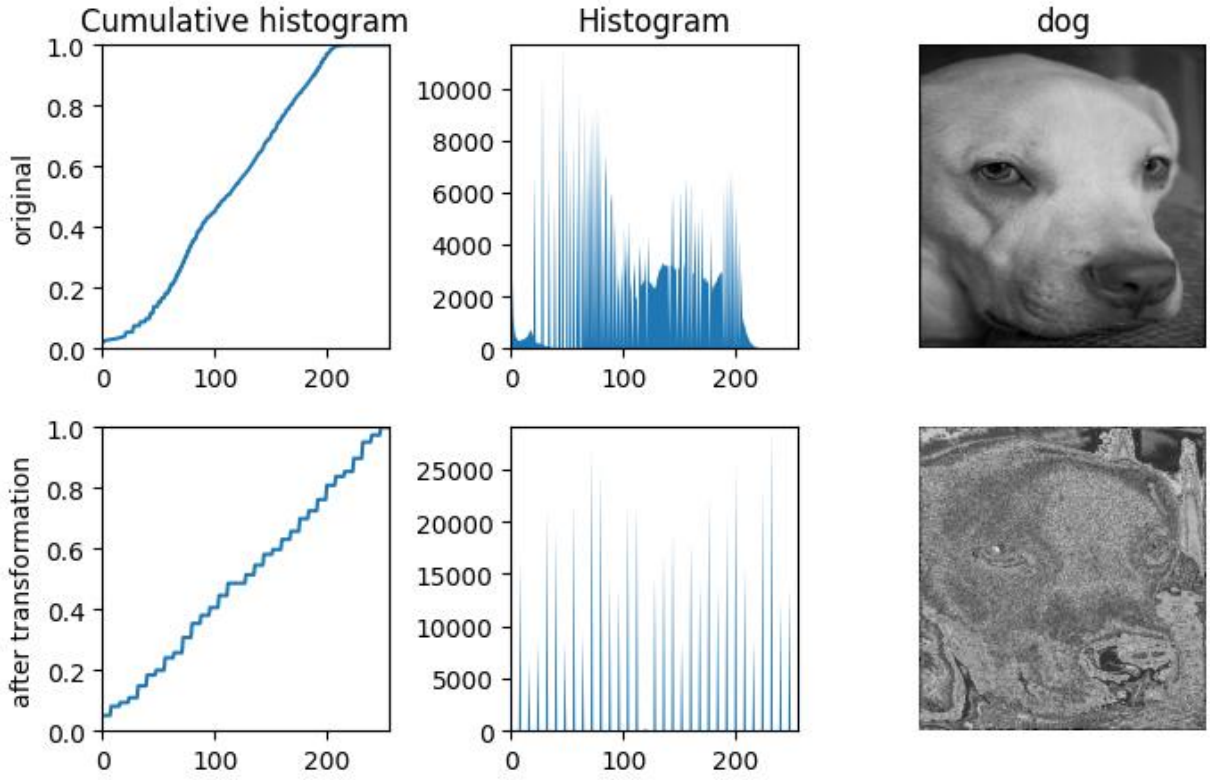

Рисунок 15. Влияние соляризации.

С помощью LUT можно задавать произвольное преобразование интенсивностей.

Листинг 12. Функция создания LUT для соляризации.

```
def create_sabattier_lut():
    lut = np.arange(256, dtype=np.uint8)
    lut = 4*lut*(255-lut)
    lut = np.where(lut > 0, lut, 0)
    lut = np.clip(lut, 0, 255)
    return lut
```

Листинг 13. Применение соляризации.

```
image_2 = cv.LUT(image, create_sabattier_lut())
```

2. Проекции

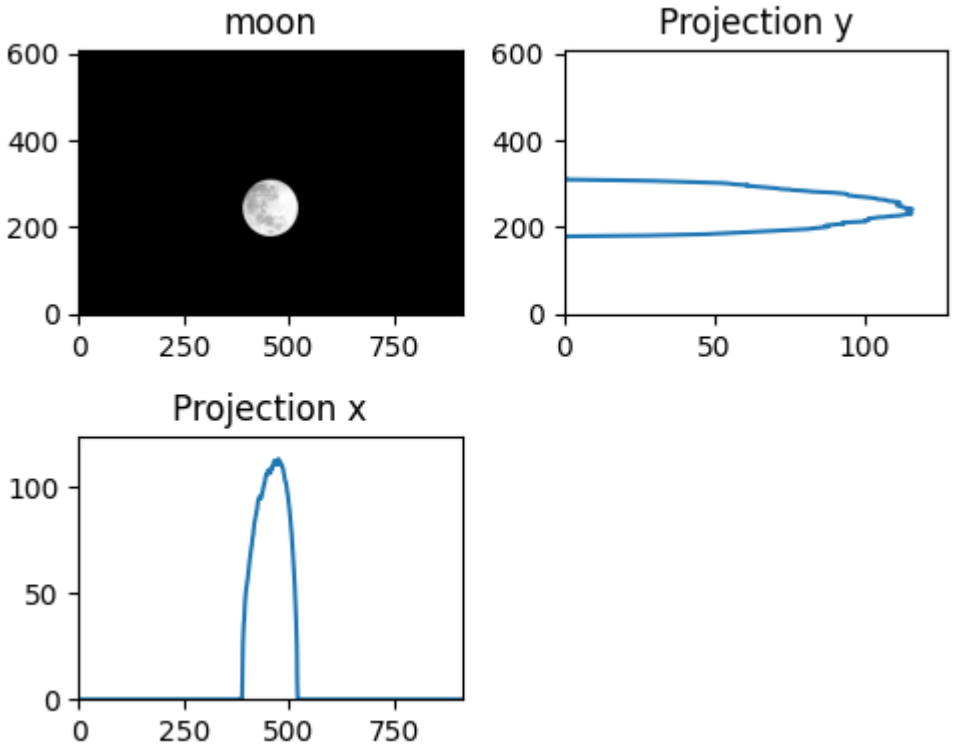

Рисунок 16. Проекции изображения.

По проекциям можно определить границы объекта на монотонном фоне. К примеру, на данном изображении, границы по оси x будут примерно 385 и 525 а по оси y и 180 и 315.

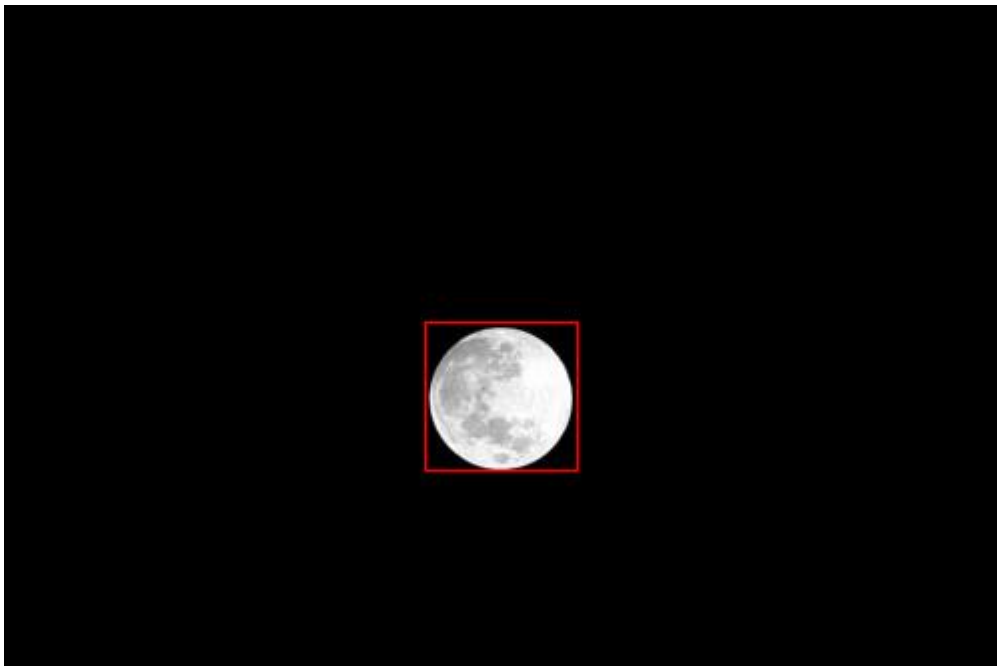

Рисунок 17. Выделение границ.

Листинг 14. Функции расчета проекций.

```
def projection_y(img):
    return np.sum(img, 0) / 255


def projection_x(img):
    return np.sum(img, 1) / 255
```

Листинг 15. Функции для построения проекций изображения.

```
def plot_projection_y(projection, len_y, subplot, title=None, label_y=None):
    padding = 1.1
    subplot.set_title(title)
    subplot.set_xlim([0, int(np.max(projection))*padding])
    subplot.set_ylim([0, len_y])
    subplot.plot(list(projection), list(range(len_y)))
    subplot.set_ylabel(label_y)


def plot_projection_x(projection, len_x, subplot, title=None, label_y=None):
    padding = 1.1
    subplot.set_title(title)
    subplot.set_xlim([0, len_x])
    subplot.set_ylim([0, int(np.max(projection)*padding)])
    subplot.plot((range(len_x)), projection)
    subplot.set_ylabel(label_y)
```

Листинг 16. Функция для совместного построения проекций и изображения.

```
def plot_squared_border_image(img: ndarray, point: tuple, side_a, side_b:
int):
    plt.imshow(img, cmap='gray', origin='lower')
    plt.gca().add_patch(Rectangle(point, side_a, side_b, linewidth=1,
edgecolor='r', fill=False))
    plt.axis('off')
    plt.savefig('moon_with_borders.png', bbox_inches='tight', pad_inches=0)
```

3. Профили.

Считаем штрих код и построим его профиль.

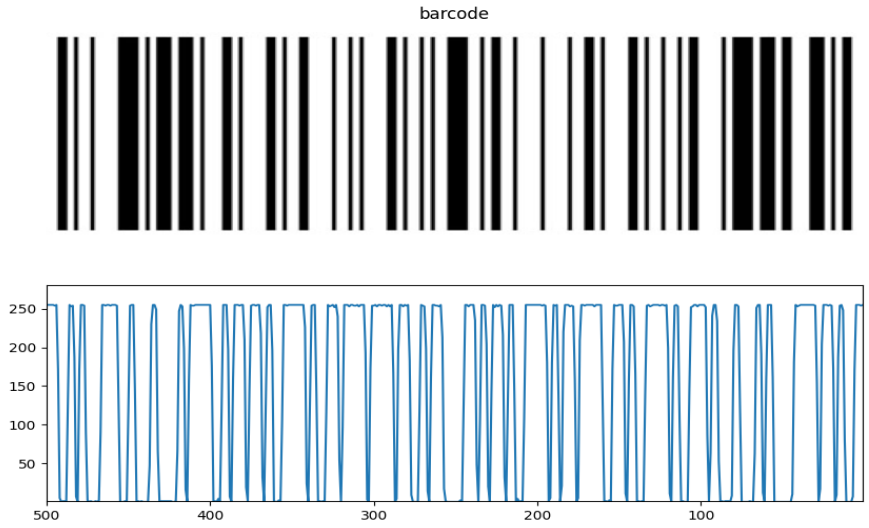

Рисунок 18. Штрих код и его профиль.

Листинг 17. Функция расчета профиля.

```
def profile_x(img):
    return img[img.shape[1] // 2]
```

Листинг 18. Функция построения профиля.

```
def plot_profile_x(prf: ndarray, subplot, title=''):
    shift = 1.1
    subplot.set_title(title)
    subplot.set_xlim(len(prf))
    subplot.set_ylim(shift*max(prf))
    subplot.plot(list(range(len(prf))), list(prf))
    subplot.invert_yaxis()
```

Листинг 19. Функция совместного построения изображения и его профиля.

```
Def plot_profile_x_image(img, profile):
    fig, (i, pr) = plt.subplots(2, 1, figsize=(10, 7))
    plot_image(img, i, 'barcode')
    plot_profile_x(profile, pr)
    plt.show()
```

Выводы

В данной лабораторной работе были рассмотрены цифровые изображения, их описание и методы, повышающие контраст изображений.

Изменение гистограммы изображения может существенно увеличить контраст изображения, не теряя при этом данные исходного изображения. Проще всего пользоваться встроенными в программный пакет функциями, которые зачастую могут дать нужный результат.

Профили и проекции – приемлемый способ понизить размерность изображения, что на практике помогает ускорить процессы. Основываясь на них, можно делать много выводов, к примеру, с помощью проекций можно определить границы объекта, если он изображен на монотонном фоне.