

Санкт-Петербургский Политехнический Университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

ОТЧЕТ
по лабораторной работе

«Операции с гистограммой изображения»
Разработка графических приложений

Работу выполнил студент

гр. 3540901/91502 _____ Дьячков В.В.

Работу принял преподаватель

_____ Абрамов Н.А.

Санкт-Петербург

2019

Содержание

1	Программа работы	3
2	Выполнение работы	3
2.1	Линейное растяжение гистограммы	3
2.2	Устойчивое линейное растяжение гистограммы	5
2.3	Эквализация гистограммы	8
2.4	Приведение гистограммы	10
3	Выводы	12

1. Программа работы

1. Реализовать линейное растяжение гистограммы.
2. Реализовать эквализацию гистограммы.
3. Реализовать приведение гистограммы к заданной.

2. Выполнение работы

2.1. Линейное растяжение гистограммы

Значение пикселя p_{in} преобразуется в p_{out} по формуле:

$$p_{out} = (p_{in} - I_{low}) \cdot \frac{O_{high} - O_{low}}{I_{high} - I_{low}} + O_{low},$$

где I – исходное изображение, а O – результирующее изображение.

При растяжении гистограммы используем $O_{high} = 255$ и $O_{low} = 0$, тогда

$$p_{out} = (p_{in} - I_{low}) \cdot \frac{255}{I_{high} - I_{low}}$$

Реализуем растяжение гистограммы в более общем виде, где O_{high} и O_{low} могут быть заданы через параметры.

```
1 def stretch(img, low=0, high=K):  
2     img_low = img.min()  
3     img_high = img.max()  
4     return ((img - img_low) / (img_high - img_low) * (high - low) + low).astype('uint8')
```

Применим растяжение гистограммы к тестовым изображениям. Дополнительно выведем гистограмму исходного и получившегося изображения.

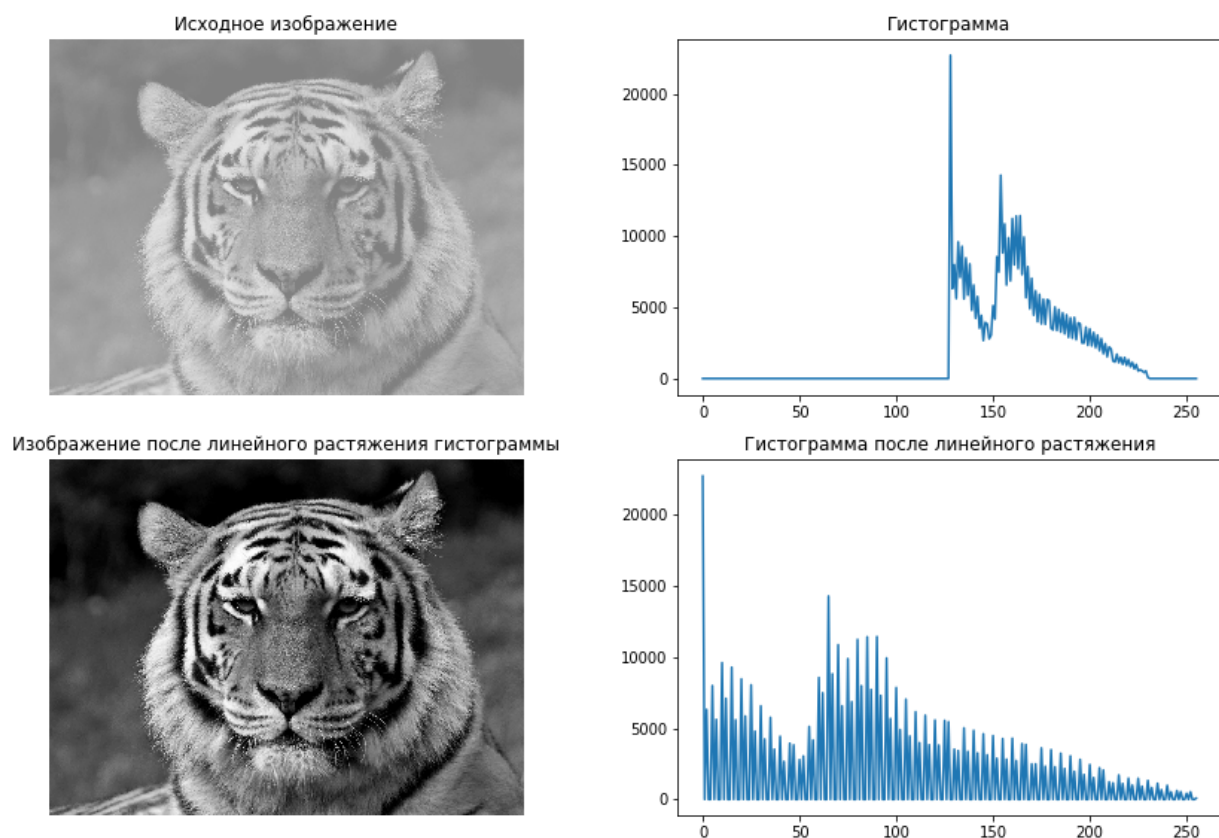


Рис. 2.1: Линейное растяжение гистограммы (1)

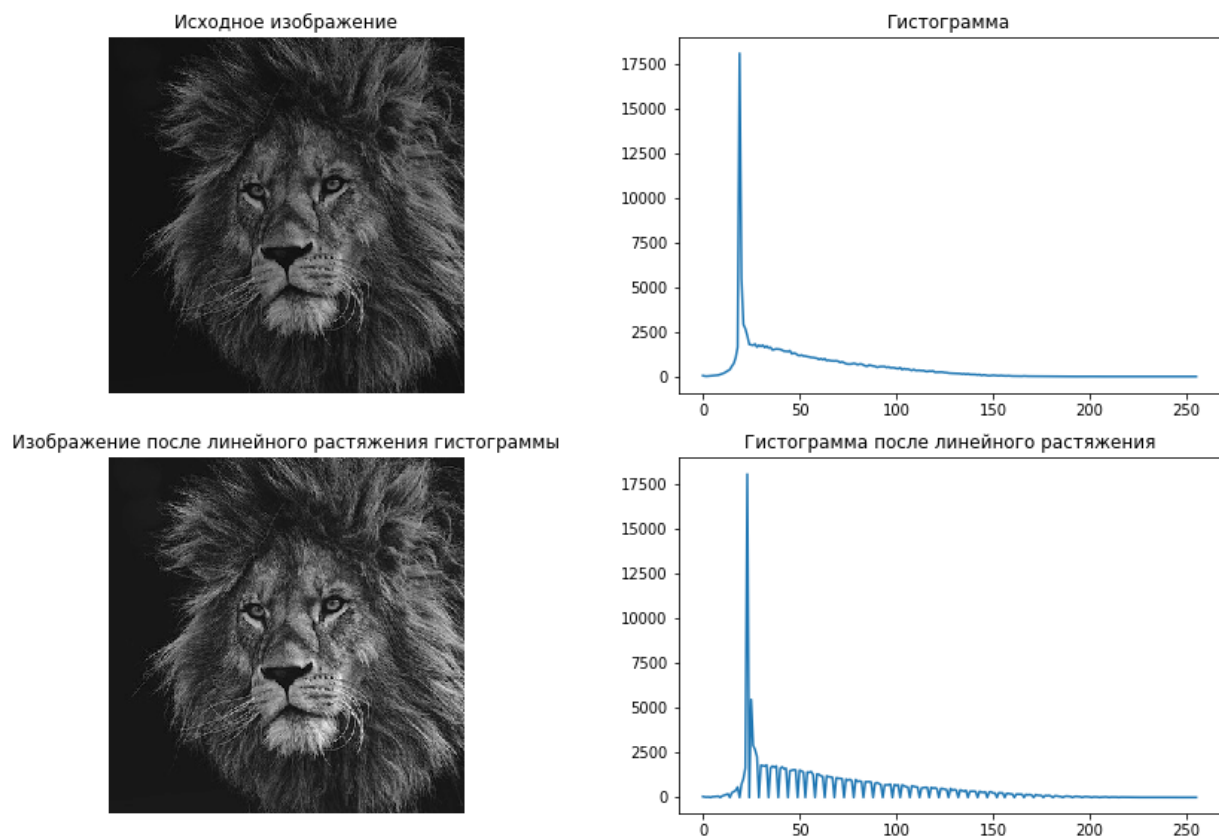


Рис. 2.2: Линейное растяжение гистограммы (2)

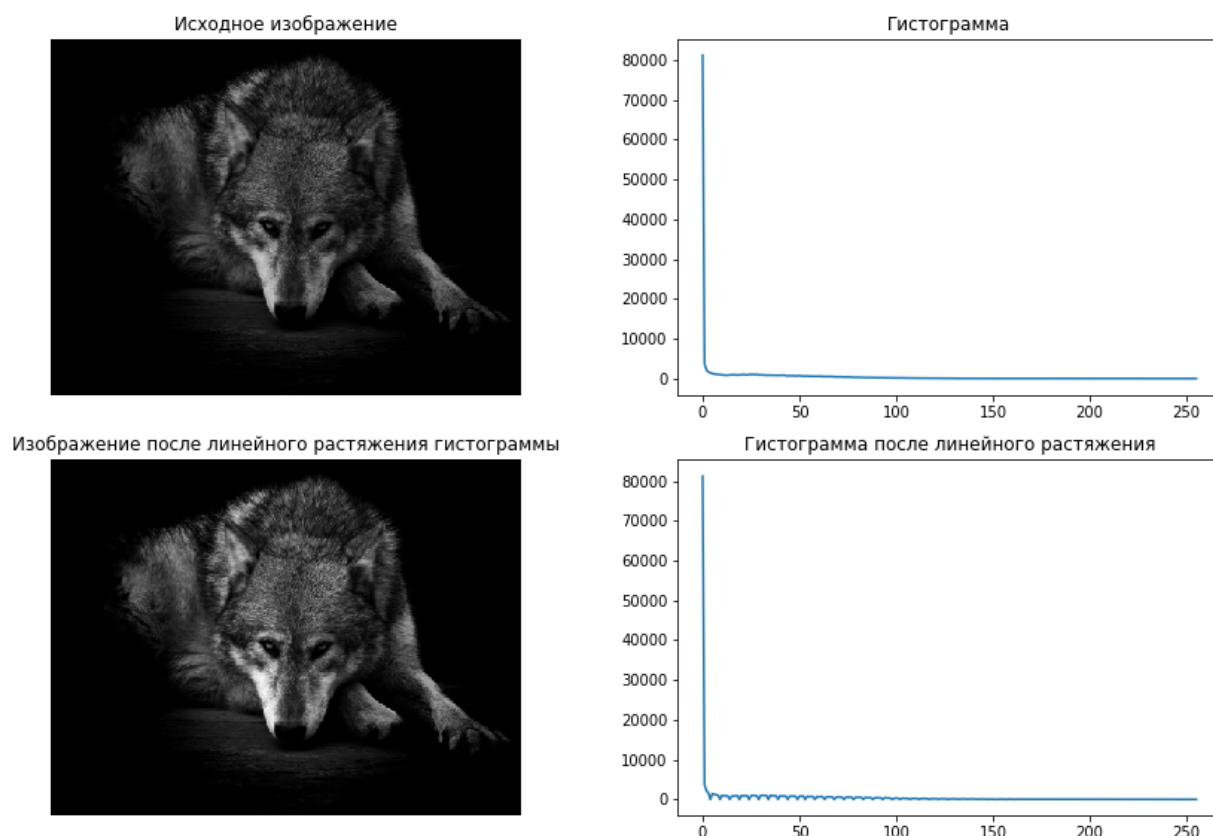


Рис. 2.3: Линейное растяжение гистограммы (3)

2.2. Устойчивое линейное растяжение гистограммы

Попробуем применить устойчивое линейное растяжение гистограммы: будем отбрасывать $M\%$ (например, 5%) самых темных и самых светлых пикселей при подсчете минимума и максимума гистограммы исходного изображения. Это позволяет более устойчиво применить растягивание гистограммы, когда слишком светлых или слишком темных пикселей небольшое количество.

Реализуем функцию, для определения минимального и максимального значения пикселя в исходном изображении, которое должно быть сохранено. Затем создадим функцию для применения устойчивого линейного растяжения гистограммы, которая будет принимать эти значения на вход.

```

1 def find_min_and_max(img, hist, drop=0.05):
2     k = int(drop * img.size)
3     x_min, x_max = 0, 0
4
5     cnt = 0
6     for i in np.arange(K + 1):
7         cnt += hist[i]
8         if cnt >= k:
9             x_min = i
10            break
11
12     cnt = 0
13     for i in np.arange(K - 1, -1, -1):

```

```

14     cnt += hist[i]
15     if cnt >= k:
16         x_max = i
17         break
18
19     return x_min, x_max
20
21 def stable_stretch(img, img_low, img_high, low=0, high=255):
22     img = img.astype('float')
23     corrected = (img - img_low) * (high - low) / (img_high - img_low)
24     res = np.clip(corrected, low, high)
25     return res.astype('uint8')

```

Применим алгоритм к изображениям. Дополнительно на гистограмме исходного изображения выделим вычисленные границы значений пикселей исходного изображения.

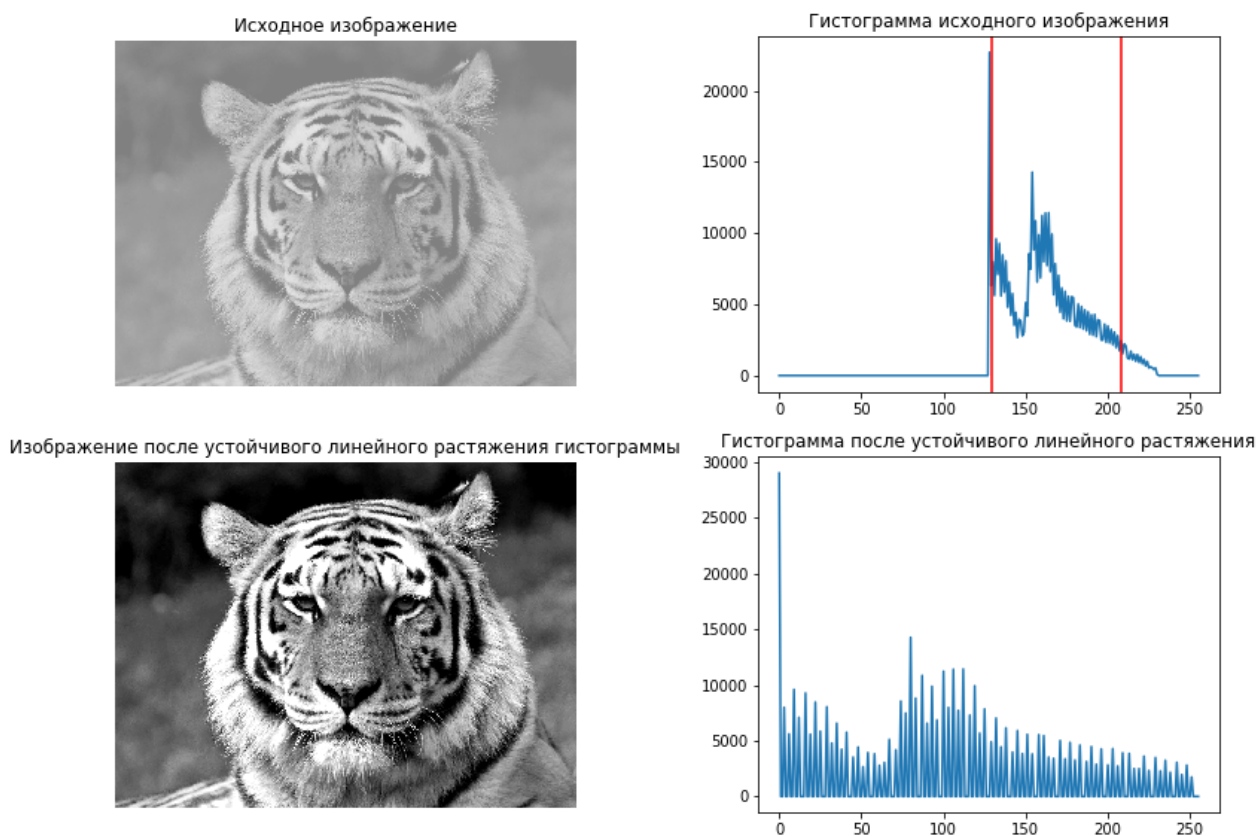


Рис. 2.4: Устойчивое линейное растяжение гистограммы (1)

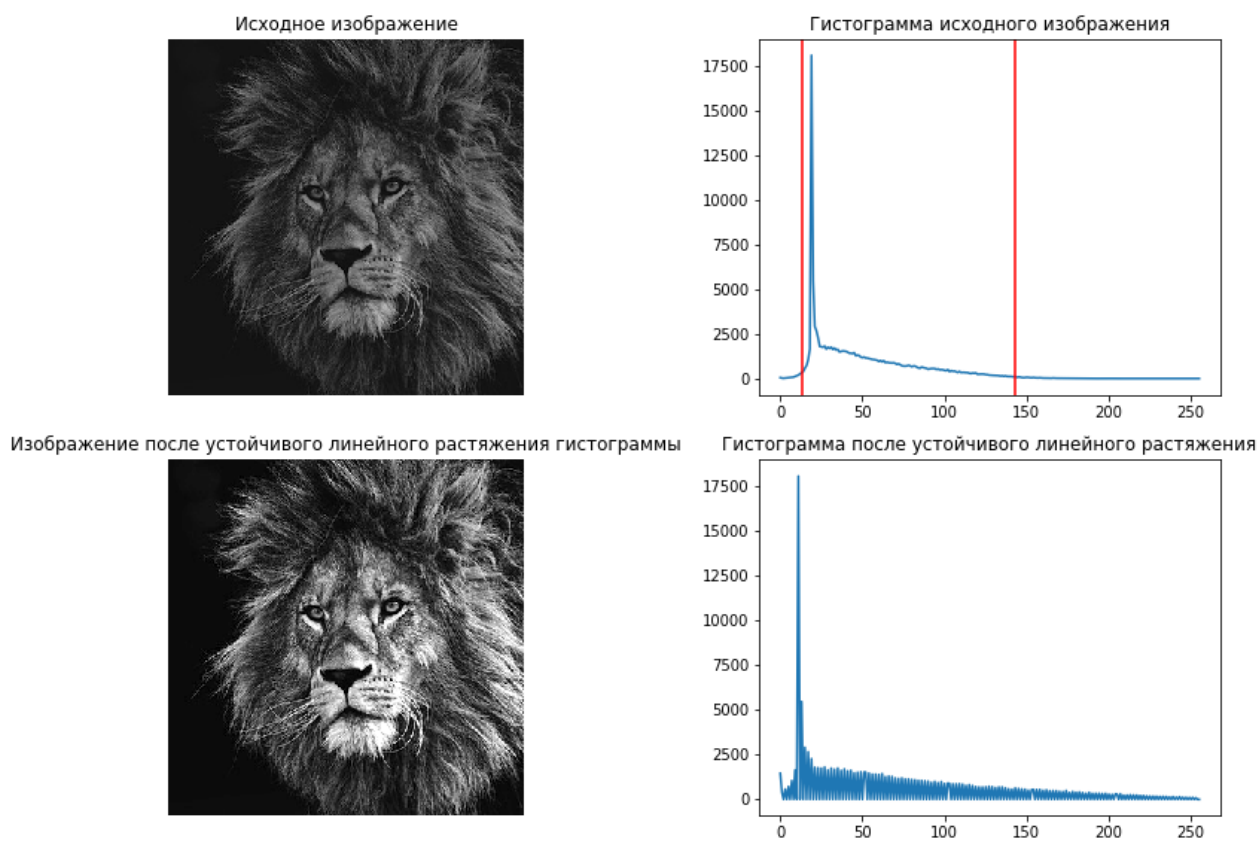


Рис. 2.5: Устойчивое линейное растяжение гистограммы (2)

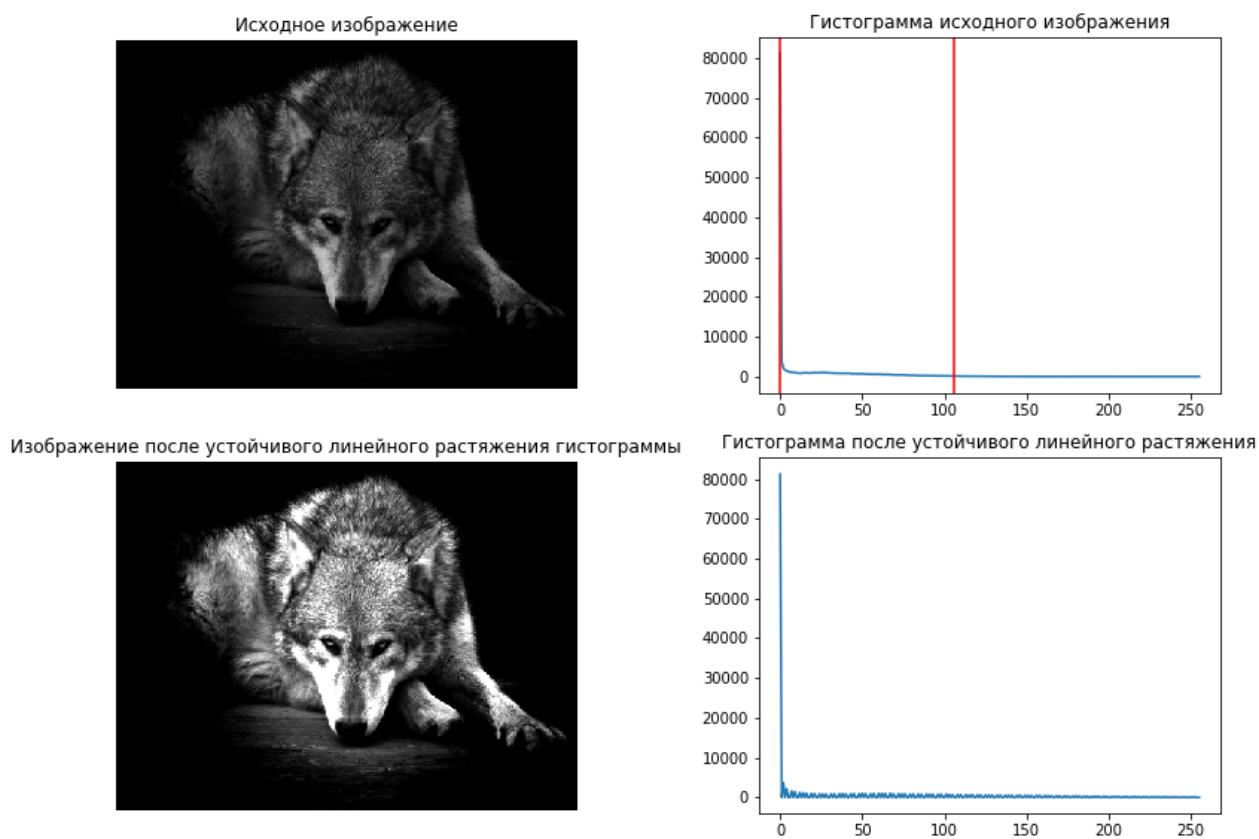


Рис. 2.6: Устойчивое линейное растяжения гистограммы (3)

2.3. Эквиализация гистограммы

Применим другой метод повышения контрастности изображения – эквиализацию гистограммы. Определим функцию распределения $cdf(n) = h(0) + h(1) + \dots + cdf(n)$. Другими словами, функция распределения является кумулятивной гистограммой. Наша задача сводится к тому, чтобы функция распределения имела вид, близкий к линейному, тогда пиксели изображения будут более равномерно использовать весь диапазон значений. Формула для преобразования пикселя входного изображения p_{in} :

$$p_{out} = round \left(\frac{cdf(p_{in}) - cdf_{min}}{N} \cdot 255 \right),$$

где N – общее число пикселей в изображении.

Реализуем функцию для нахождения кумулятивной суммы по рассчитанной гистограмме изображения. Затем применим ее для эквиализации гистограммы.

```
1 def my_cumsum(hist):
2     cdf = hist.copy()
3     for i in np.arange(1, hist.size):
4         cdf[i] = cdf[i - 1] + hist[i]
5     return cdf
6
7 np.all(my_cumsum(my_hist(tiger)) == np.cumsum(my_hist(tiger)))
8 ## True
9
10 def equalize(x, cdf, cdf_min):
11     N = cdf[-1]
12     return np.round((cdf[x] - cdf_min) / N * 255).astype('uint8')
```

Применим созданную функцию для эквиализации гистограмм тестовых изображений.

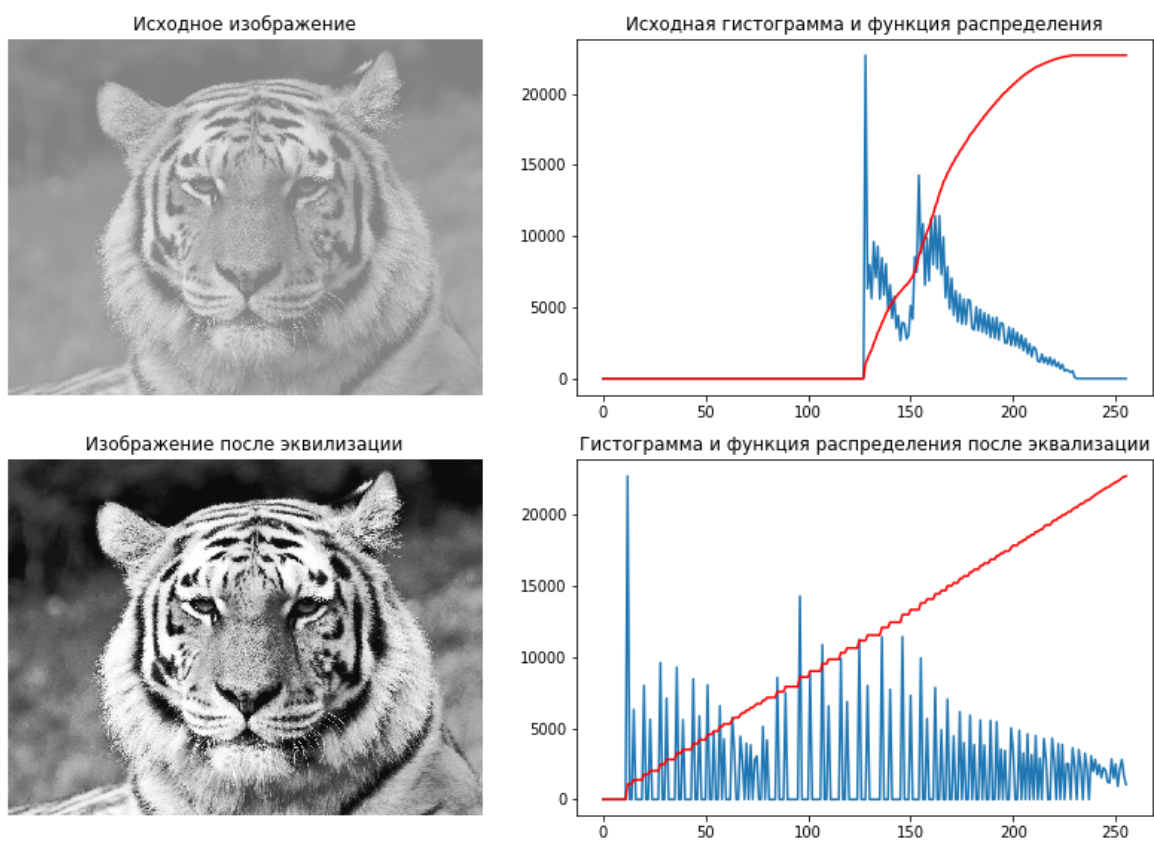


Рис. 2.7: Эквализация гистограммы (1)

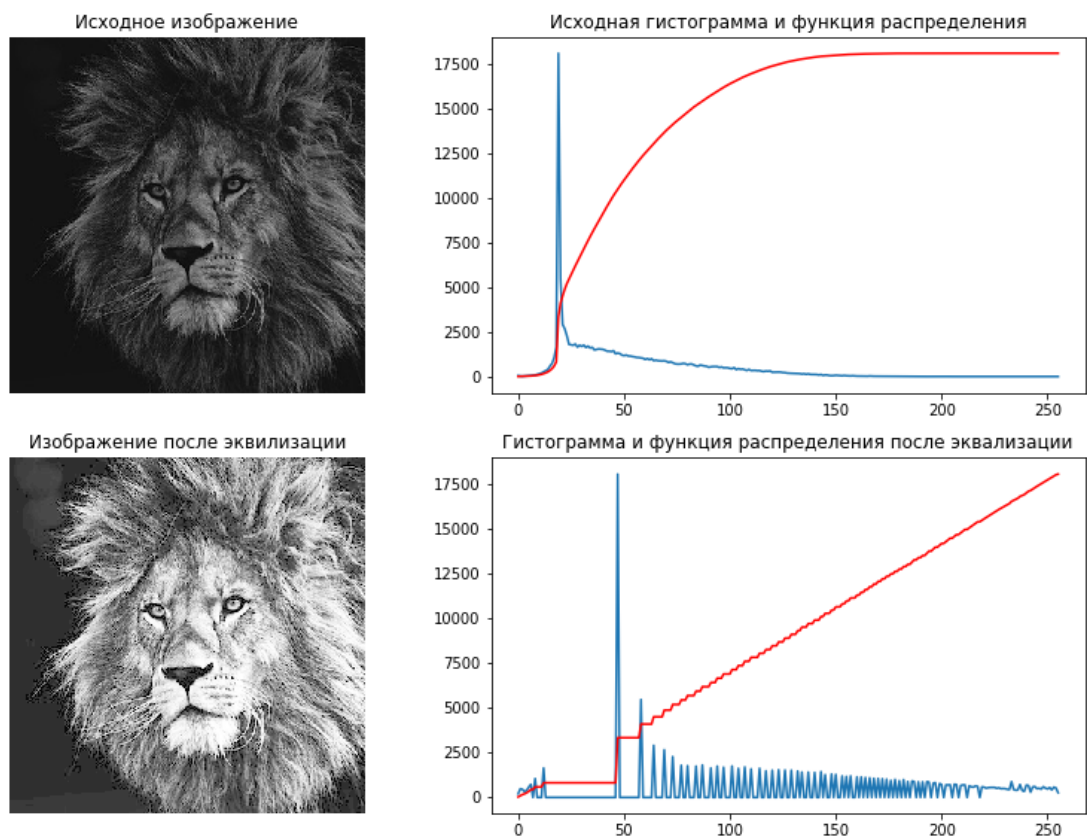


Рис. 2.8: Эквализация гистограммы (2)

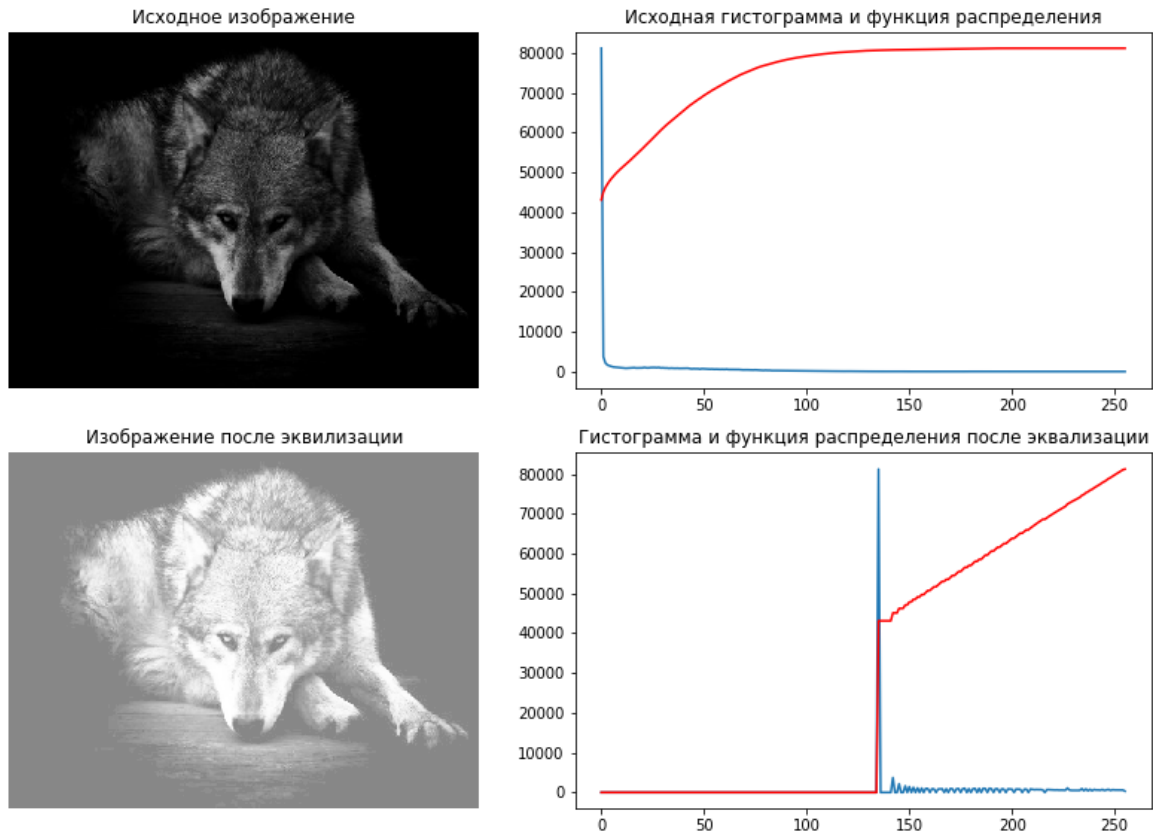


Рис. 2.9: Эквализация гистограммы (3)

2.4. Приведение гистограммы

В том случае, если у нас есть референсное изображение, мы можем использовать его гистограмму для преобразования входного изображения. Для этого создадим отображение каждого значения входного изображения в выходное (всего 256 возможных входных и выходных значений), после чего отобразим значение каждого пикселя входного изображения в выходное.

Рассчитаем гистограммы и функции распределений входного (cdf_1) и референсного (cdf_2) изображений, после чего найдем такие значения пикселей p_1 и p_2 , что:

$$cdf_1(p_1) = cdf_2(p_2),$$

тогда значение пикселя p_1 отображается в значение p_2 .

Реализуем функцию приведения гистограммы.

```

1 def match_histogram(img, ref):
2     res = img.copy()
3
4     img_hist = my_hist(img)
5     img_cdf = my_cumsum(img_hist)
6     img_cdf_norm = img_cdf / img.size
7
8     ref_hist = my_hist(ref)
9     ref_cdf = my_cumsum(ref_hist)
10    ref_cdf_norm = ref_cdf / ref.size
11

```

```

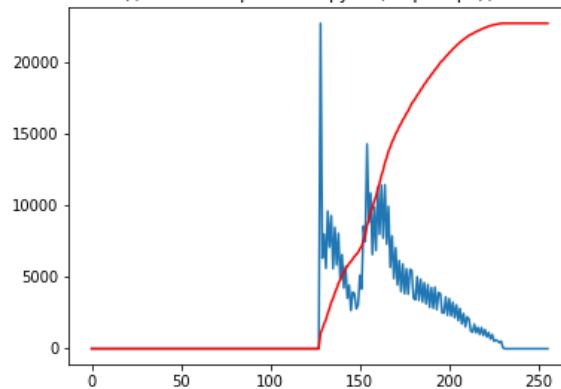
12 mapping = np.zeros(K + 1)
13 for i in np.arange(K + 1):
14     j = K
15     while True:
16         mapping[i] = j
17         j = j - 1
18         if j < 0 or img_cdf_norm[i] > ref_cdf_norm[j]:
19             break
20
21 for i in np.arange(res.shape[0]):
22     for j in np.arange(res.shape[1]):
23         a = res.item(i,j)
24         b = mapping[a]
25         res.itemset((i,j), b)
26
27 return res

```

Исходное изображение



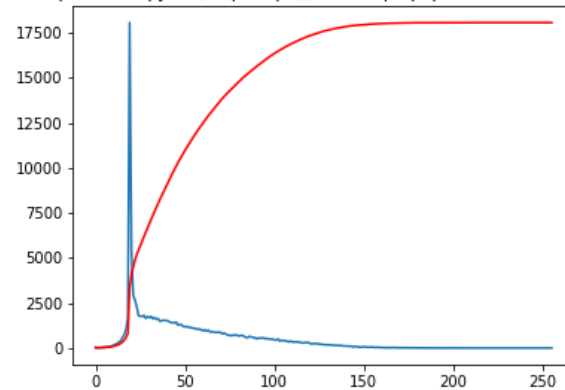
Исходная гистограмма и функция распределения



Референсное изображение



Гистограмма и функция распределения референсного изображения



Изображение после приведения гистограммы



Гистограмма и функция распределения после приведения гистограммы

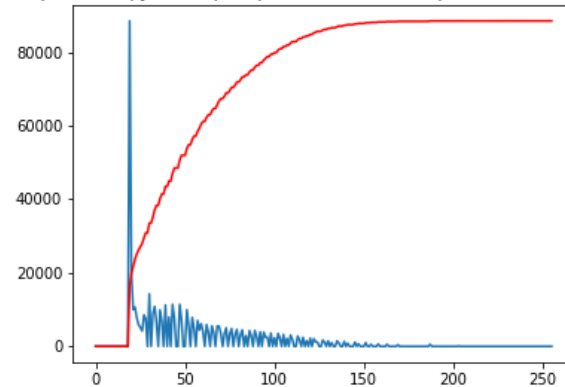


Рис. 2.10: Приведение гистограммы (1)

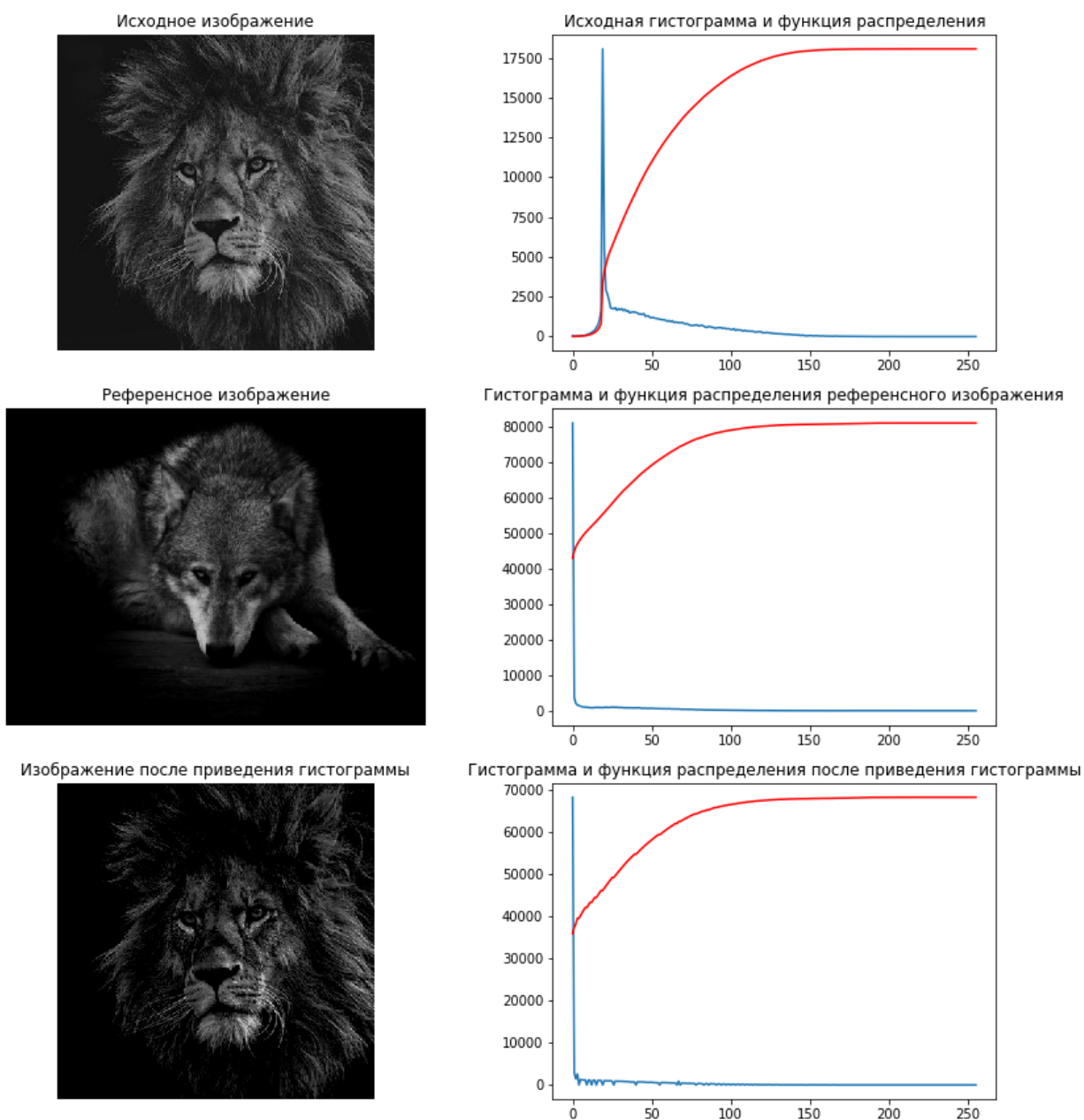


Рис. 2.11: Приведение гистограммы (2)

3. Выводы

В данной работе были реализованы различные операции над гистограммой изображения:

- линейное растяжение и устойчивое линейное растяжение гистограммы;
- эквализация гистограммы;
- приведение гистограммы изображения к заданному виду.