

Автоконтраст черно-белого изображения

Прочитайте изображение из файла `img.png`. Примените к нему линейное выравнивание яркости: примените к каждому пикселю функцию

$$f(x) = (x - x_{\min}) \cdot \frac{255}{x_{\max} - x_{\min}}$$

После вычисления функции значения изображения окажутся вещественными. Чтобы привести их к целым числам, используйте метод `img.astype('uint8')`, который возвращает изображение в целых числах. Результат сохраните в файл `out_img.png`.

В примере входа и выхода указаны ссылки на файлы. Скачав эти файлы, можно протестировать свою программу. Для сравнения вашего ответа с верным используйте функцию `numpy.array_equal`.

Sample Input:

<https://stepik.org/media/attachments/lesson/58402/tiger-low-contrast.png>

Sample Output:

<https://stepik.org/media/attachments/lesson/58402/tiger-high-contrast.png>

✓ Верно.

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете [помочь другим](#) учащимся в комментариях.

```
1 from skimage.io import imread, imshow, imsave
2 from skimage import img_as_float, img_as_ubyte
3 import numpy as np
4
5 image = imread('img.png')
6 xmax = max(image.ravel())
7 xmin = min(image.ravel())
8 k = 255/(xmax-xmin)
9 imgN = ((image-xmin)*k).astype('uint8')
10 imsave('out_img.png', imgN)
```

Следующий шаг

Решить снова

Подсчет минимума и максимума устойчивого автоконтраста

Прочитайте изображение из файла `img.png`. Подсчитайте минимум и максимум яркости для стабильного автоконтраста этого изображения. Необходимо отбросить 5% самых светлых и 5% самых темных пикселей. Для получения числа отбрасываемых пикселей используйте формулу

$$k = \text{round}(\#pix \cdot 0.05)$$

Два посчитанных числа (минимум и максимум) выведите на стандартный вывод через пробел.

Попробуйте подсчитать минимум и максимум для стабильного автоконтраста двумя способами, указанными в видео.

В примере входа указана ссылка на файлы. Скачав этот файл, можно протестировать свою программу.

Sample Input:

<https://stepik.org/media/attachments/lesson/58402/tiger-low-contrast.png>

Sample Output:

129 208

✓ Всё получилось!

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете [помочь другим](#) учащимся в комментариях.

```
1 from skimage.io import imread, imshow, imsave
2 from skimage import img_as_float, img_as_ubyte
3 import numpy as np
4
5 img = imread('img.png')
6 pix = img.shape[0]*img.shape[1]
7 k=round(pix*0.05)
8 v = img.ravel()
9 v.sort()
10 xmin = v[k]
11 xmax = v[pix-k]
12 print(xmin, xmax)
```

Следующий шаг

Решить снова

Устойчивый автоконтраст черно-белого изображения

Прочитайте изображение из файла `img.png`. Примените к нему линейное выравнивание яркости: примените к каждому пикселю функцию

$$f(x) = (x - x_{min}) \cdot \frac{255}{x_{max} - x_{min}}$$

Для вычисления максимума и минимума отбрасывайте по 5% самых светлых и самых темных пикселей (как в предыдущем задании). Перед вычислениями приведите изображение в вещественные числа (`img.astype('float')`), иначе может возникнуть переполнение (т.к. значения некоторых пикселей мы игнорируем при подсчете минимума и максимума). После растяжения яркости обрежьте значения изображения от 0 до 255 с помощью функции `numpy.clip`.

После вычисления функции значения изображения окажутся вещественными. Чтобы привести их к целым числам, используйте метод `img.astype('uint8')`, который возвращает изображение в целых числах. Результат сохраните в файл `out_img.png`.

В примере входа и выхода указаны ссылки на файлы. Скачав эти файлы, можно протестировать свою программу. Для сравнения вашего ответа с верным используйте функцию `numpy.array_equal`.

Sample Input:

<https://stepik.org/media/attachments/lesson/58402/tiger-low-contrast.png>

Активация Windows

Чтобы активировать Windows, перейдите в раздел "Параметры".

Sample Output:

<https://stepik.org/media/attachments/lesson/58402/tiger-stable-contrast.png>

✓ Прекрасный ответ.

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

```
1 from skimage.io import imread, imshow, imsave
2 from skimage import img_as_float, img_as_ubyte
3 from numpy import clip
4
5 img = imread('img.png')
6 img = img.astype('float')
7 pix = img.shape[0]*img.shape[1]
8 k=round(pix*0.05)
9 v = img.copy()
10 v = v.ravel()
11 v.sort()
12 xmin = v[k]
13 xmax = v[pix-k]
14
15 k = 255/(xmax-xmin)
16 imgN = (img-xmin)*k
17 imgN = (clip(imgN, 0, 255)).astype('uint8')
18 imsave('out_img.png', imgN)
```

Активация Windows

Чтобы активировать Windows, перейдите в раздел "Параметры".

Следующий шаг

Решить снова

Устойчивый цветной автоконтраст

Прочитайте цветное изображение из файла `img.png`. Примените к нему устойчивый автоконтраст. Для этого:

1. Переведите изображение в вещественные числа от 0 до 1.
2. Переведите изображение в пространство YUV по формулам:

$$\begin{aligned}Y &= 0.2126 \cdot R + 0.7152 \cdot G + 0.0722 \cdot B \\U &= -0.0999 \cdot R - 0.3360 \cdot G + 0.4360 \cdot B \\V &= 0.6150 \cdot R - 0.5586 \cdot G - 0.0563 \cdot B\end{aligned}$$

3. Найдите максимум и минимум для устойчивого автоконтраста с отбрасыванием 5% самых светлых и 5% самых темных пикселей.
4. Примените линейное растяжение канала Y по формуле

$$f(x) = (x - x_{min}) \cdot \frac{255}{x_{max} - x_{min}}$$

5. Обрежьте значения канала Y от 0 до 1.
6. Переведите изображение в пространство RGB по формулам:

$$\begin{aligned}R &= Y + 1.2803 \cdot V \\G &= Y - 0.2148 \cdot U - 0.3805 \cdot V \\B &= Y + 2.1279 \cdot U\end{aligned}$$

7. Обрежьте значения изображения от 0 до 1.
8. Переведите изображение в целые числа от 0 до 255.

Результат сохраните в файл `out_img.png`.

В примере входа и выхода указаны ссылки на файлы. Скачав эти файлы, можно протестировать свою программу. Для сравнения вашего ответа с верным используйте функцию `numpy.array_equal`.

Sample Input:

<https://stepik.org/media/attachments/lesson/60609/tiger-color.png>

Активация Windows

Чтобы активировать Windows, перейдите в раздел "Параметры".

✓ Абсолютно точно.

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете [помочь другим](#) учащимся в комментариях.

```
1 from skimage.io import imread, imshow, imsave
2 from skimage import img_as_float, img_as_ubyte
3 from numpy import clip
4 from numpy import dstack
5
6 img = imread('img.png')
7 imgN = img_as_float(img)
8 Y = 0.2126*imgN[:, :, 0]+0.7152*imgN[:, :, 1]+0.0722*imgN[:, :, 2]
9 U = -0.0999*imgN[:, :, 0]-0.3360*imgN[:, :, 1]+0.4360*imgN[:, :, 2]
10 V = 0.6150*imgN[:, :, 0]-0.5586*imgN[:, :, 1]-0.0563*imgN[:, :, 2]
11
12 pix = img.shape[0]*img.shape[1]
13 n,m = img.shape[0],img.shape[1]
14 Vk=Y.copy()
15 Vk=Vk.ravel()
16 Vk.sort()
17 d=len(Vk)
18 minVk=min(Vk)
19 xmin = Vk[int(d*0.05)]
20 xmax = Vk[int(d*0.95)]
21 k=1/(xmax-xmin)
22 Y=clip((Y-xmin)*k,0,1)
23 R = clip(Y+1.2803*V,0,1)
24 G = clip(Y-0.2148*U-0.3805*V,0,1)
25 B = clip(Y+2.1279*U,0,1)
26 img_f = dstack((R,G,B))
27 imsave('out_img.png',img_as_ubyte(img_f))
```

Следующий шаг

Решить снова

Активация Windows
Чтобы активировать Window
раздел "Параметры".

Преобразование серого мира

Прочитайте изображение из файла `img.png`. Примените к нему преобразование серого мира. Для этого:

1. Сконвертируйте изображение в вещественные числа.
2. Подсчитайте коэффициенты r_w , g_w , b_w как описано в видео.
3. Поделите каналы изображения на коэффициенты.
4. Обрежьте значения пикселей, чтобы они не выходили из допустимого диапазона $([0; 255]$ или $[0;1])$.

Результат сохраните в файл `out_img.png`.

В примере входа и выхода указаны ссылки на файлы. Скачав эти файлы, можно протестировать свою программу. Для сравнения вашего ответа с верным используйте функцию `numpy.array_equal`.

Sample Input:

<https://stepik.org/media/attachments/lesson/60610/railroad.png>

Sample Output:

<https://stepik.org/media/attachments/lesson/60610/railroad-gray-world.png>

✓ Отличное решение!

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

```
1 from skimage.io import imread, imshow, imsave
2 from skimage import img_as_float, img_as_ubyte
3 from numpy import clip
4 from numpy import dstack
5 from numpy import average
6 img = img_as_float(imread('img.png'))
7 R = img[:, :, 0]
8 G = img[:, :, 1]
9 B = img[:, :, 2]
10 Rs = average(R)
11 Gs = average(G)
12 Bs = average(B)
13 avg = (Rs+Gs+Bs)/3
14 rw = Rs/avg
15 gw = Gs/avg
16 bw = Bs/avg
17 R = R/rw
18 G = G/gw
19 B = B/bw
20 imsave('out_img.png', img_as_ubyte(dstack((clip(R, 0, 1), clip(G, 0, 1), clip(B, 0, 1)))))
```

Активация Windows

Чтобы активировать Windows, перейдите в раздел "Параметры".

Следующий шаг

Решить снова

Выравнивание гистограммы

[Слайды](#) видео.

Прочитайте изображение из файла `img.png`. Примените к нему выравнивание гистограммы по алгоритму, описанному в слайдах и видео. Работать достаточно в целых числах, помещающихся в байт (т.е. изображение конвертировать не нужно). Результат сохраните в файл `out_img.png`.

В примере входа и выхода указаны ссылки на файлы. Скачав эти файлы, можно протестировать свою программу. Для сравнения вашего ответа с верным используйте функцию `numpy.array_equal`.

Sample Input:

<https://stepik.org/media/attachments/lesson/60611/landscape.png>

Sample Output:

<https://stepik.org/media/attachments/lesson/60611/landscape-histeq.png>

✓ Всё получилось!

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете [помочь другим](#) учащимся в комментариях.

```
1 from skimage.io import imread, imshow, imsave
2 from skimage import img_as_float, img_as_ubyte
3 import numpy as np
4 img = imread('img.png')
5 sdf = np.histogram(img, bins=255, range=(0, 255))
6 cdf = np.zeros(255)
7 cdf_k = np.zeros(255)
8 s = 0
9 k = 255/(img.size-1)
10 for i in range(255):
11     s+=int(sdf[0][i])
12     cdf[i]=s
13 cdf_min = cdf[np.where(cdf>0)[0][0]]
14 for i in range(img.shape[0]):
15     for j in range(img.shape[1]):
16         img[i][j] = round((cdf[img[i][j]]-cdf_min)*k)
17 imsave('out_img.png', img)
```

Следующий шаг

Решить снова

Активация Windows

Чтобы активировать Windows, перейдите в раздел "Параметры".