DSP Report. Images processing. Filters

функция procImageUsingKernel(image, kernel)
Несколько модифицированная функция proc из методички ЦОС.
Функция применяет маску - kernel, к изображению image.
изменения:

image = image/255
image = img_as_float(image)
функция img_as_float является псевдонимом для img_as_float64 - конвертирует изображение приведенное к ndarray в формат с плавающей точкой двойной точности (64-bit)

```
if weighted_pixel_sum > 1.:
        weighted_pixel_sum=1.
elif weighted_pixel_sum < 0.:
        weighted_pixel_sum=0.
```

```
np.clip(weighted_pixel_sum, 0, 1)
```

данное изменение позволяет нормализовать цветные изображения(за исключением случаев, когда фильтры работают только с одноканальными изображениями, в данном случае необходимо конвертировать цветное изображение в черно-белое).

конвертация цветных изображений в черно-белое

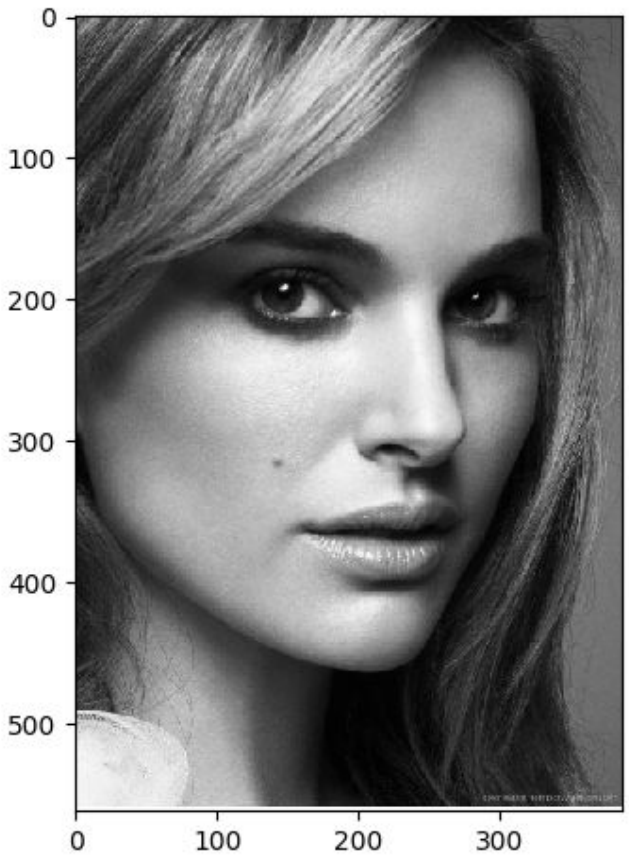
```
red = img_float[:, :, 0]
green = img_float[:, :, 1]
blue = img_float[:, :, 2]

gray = 0.2126 * red + 0.7152 * green + 0.0722 * blue
```

1. Сглаживание и подавление шумов в изображении

маска np.array([[1., 1., 1.], [1., 1., 1.], [1., 1., 1.]]) / 9.

слева исходное изображение

справа изображение после применения маски



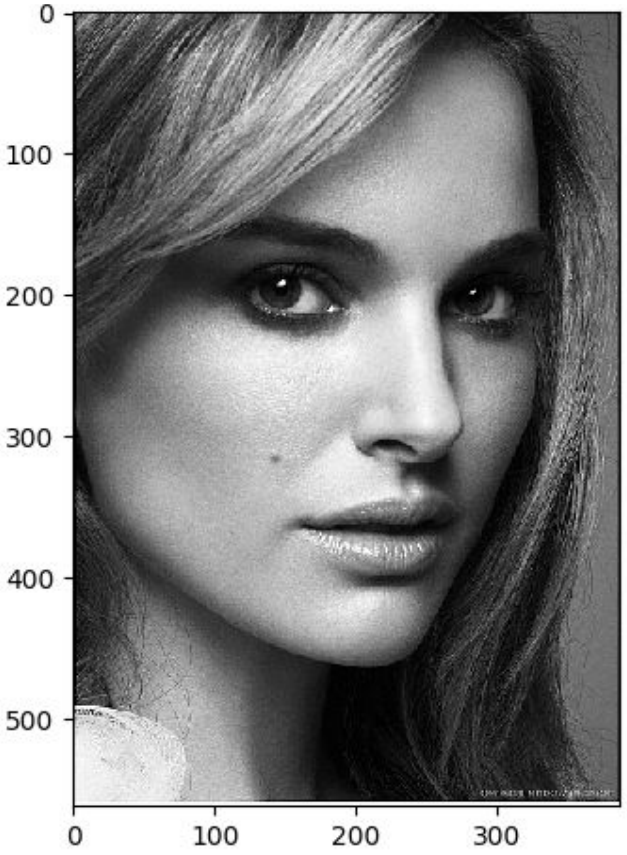
2. Подчеркивание контуров

маски

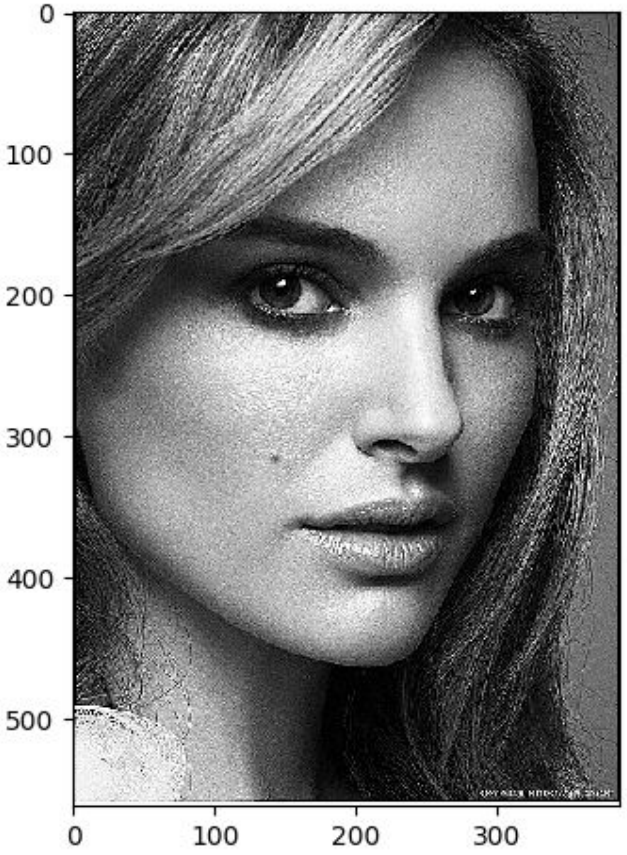
1. np.array([[-1., -1., -1.], [-1., 17., -1.], [-1., -1., -1.]]) / 9.
2. np.array([[0., -1., 0.], [-1., 5., -1.], [0., -1., 0.]])

Оригинал изображения

применение маски 1

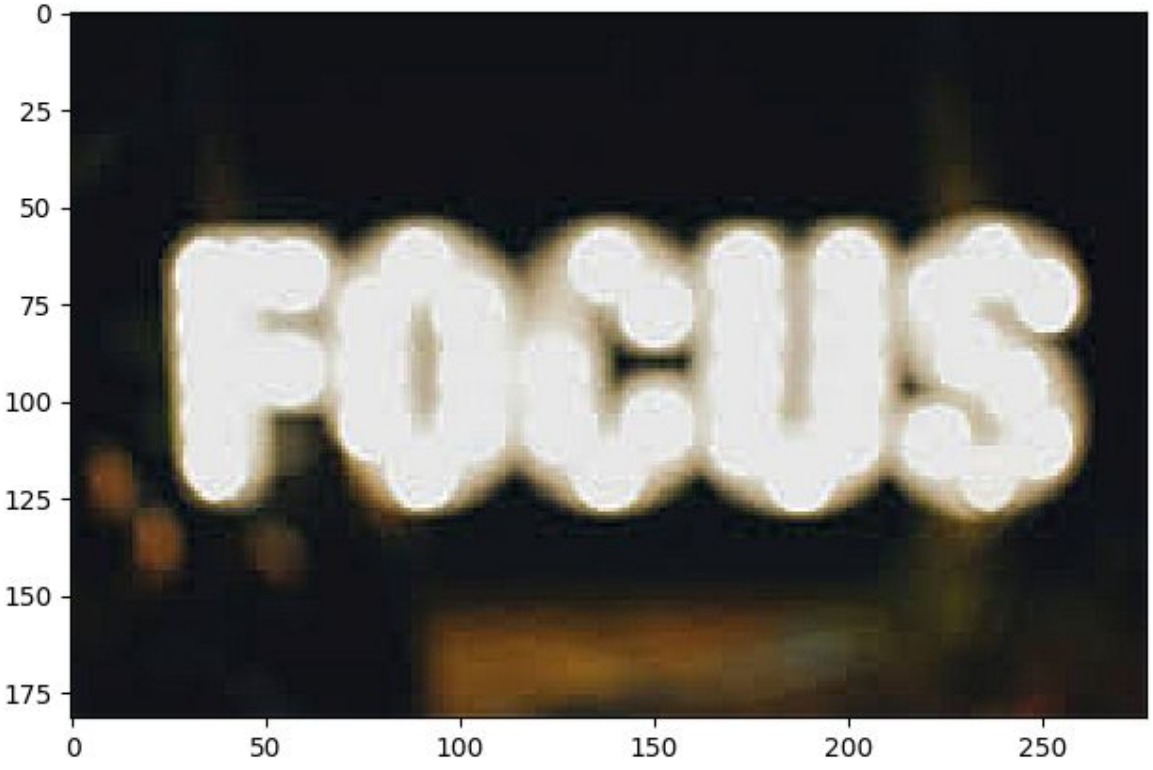


применение маски 2

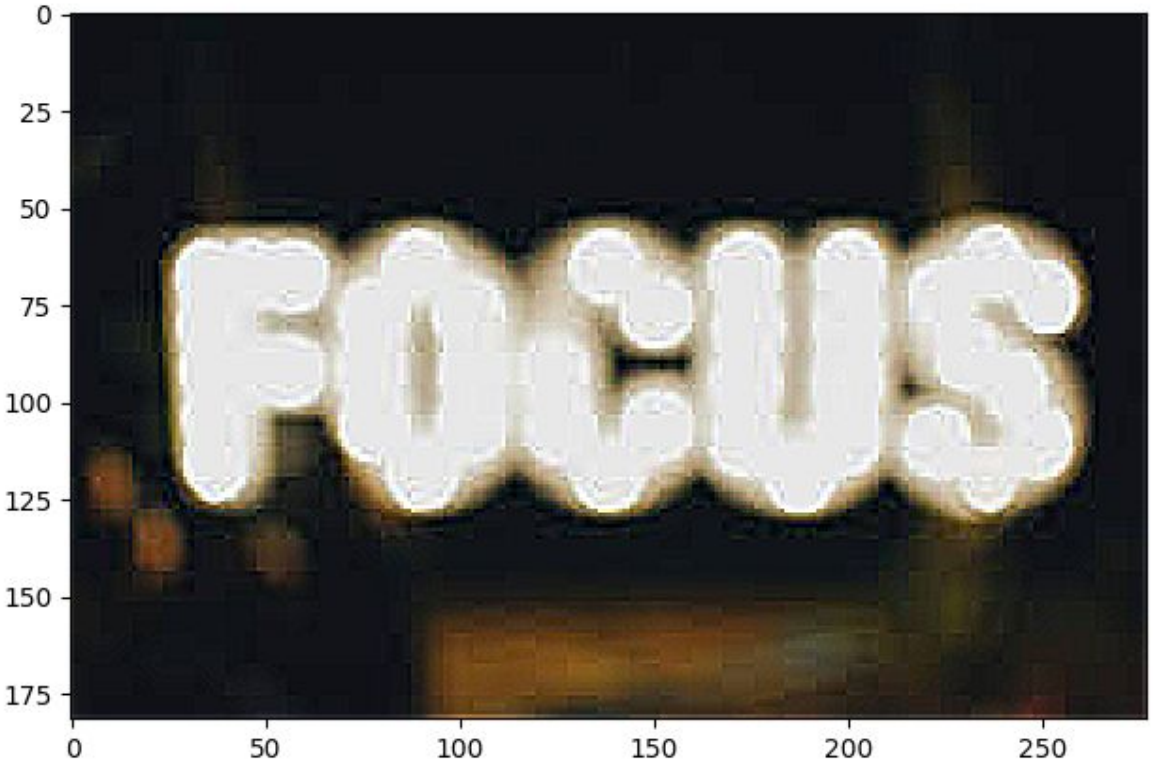


оригинал цветного изображения

применение маски 1



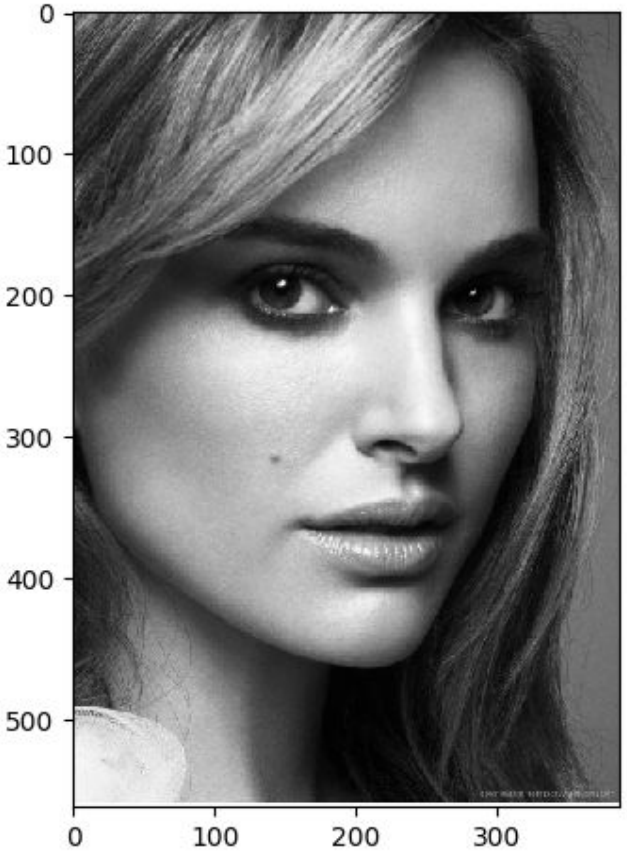
применение маски 2



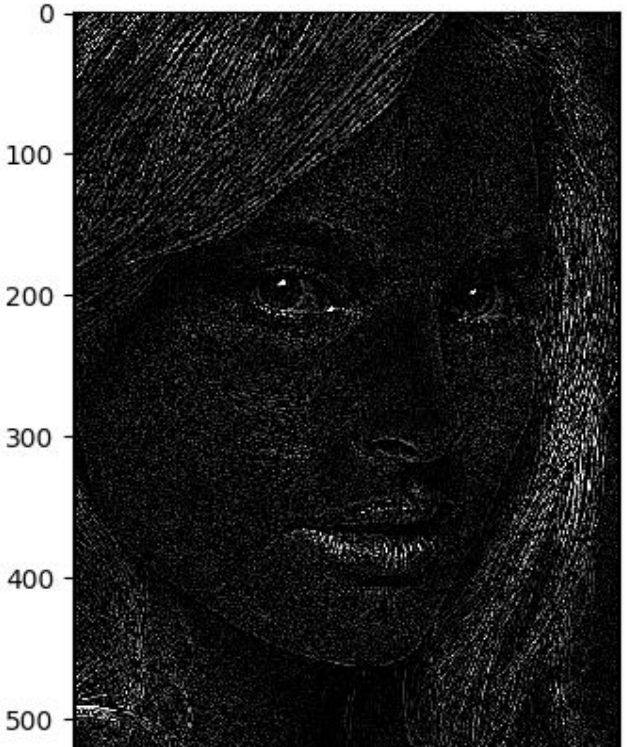
Подчеркивание контуров
Данные маски могут потребовать предварительной конвертации цветного изображения в одноканальное, при наличии высокого значения яркости (Y) изображения.

1. np.array([[-1., -1., -1.], [-1., 8., -1.], [-1., -1., -1.]])
2. np.array([[0., -1., 0.], [-1., 4., -1.], [0., -1., 0.]])

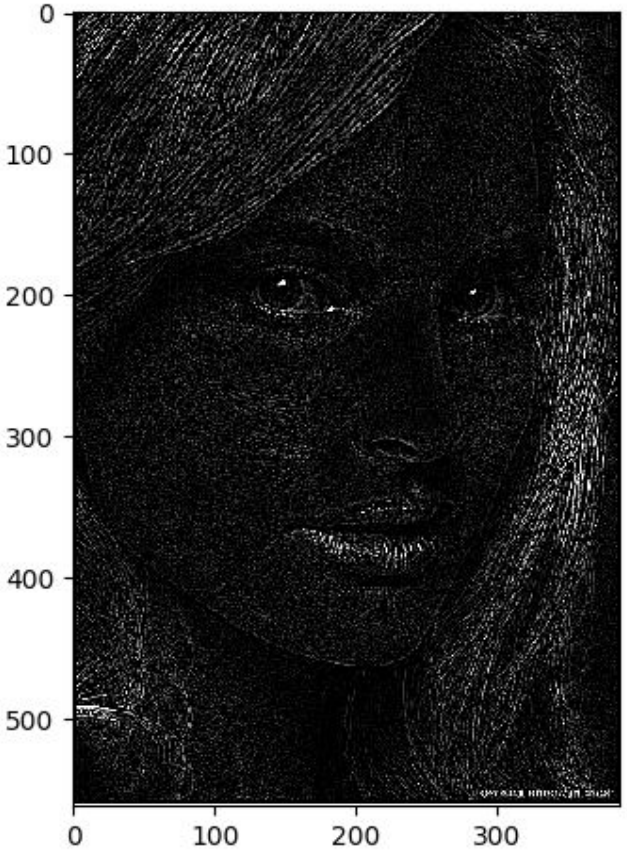
оригинал черно-белого, изображения, но данное изображение содержит 3 канала



маска 1 (с применением конвертации в одноканальное изображение)

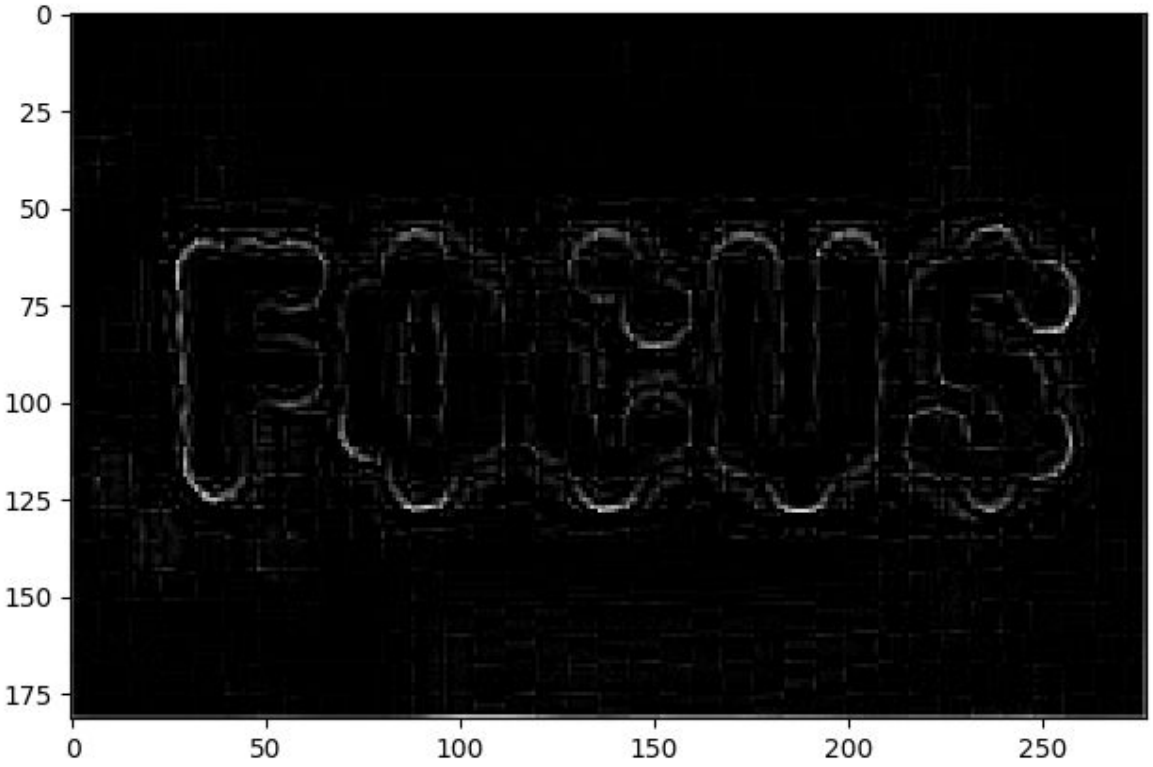


маска 2 (с применением конвертации в одноканальное изображение)

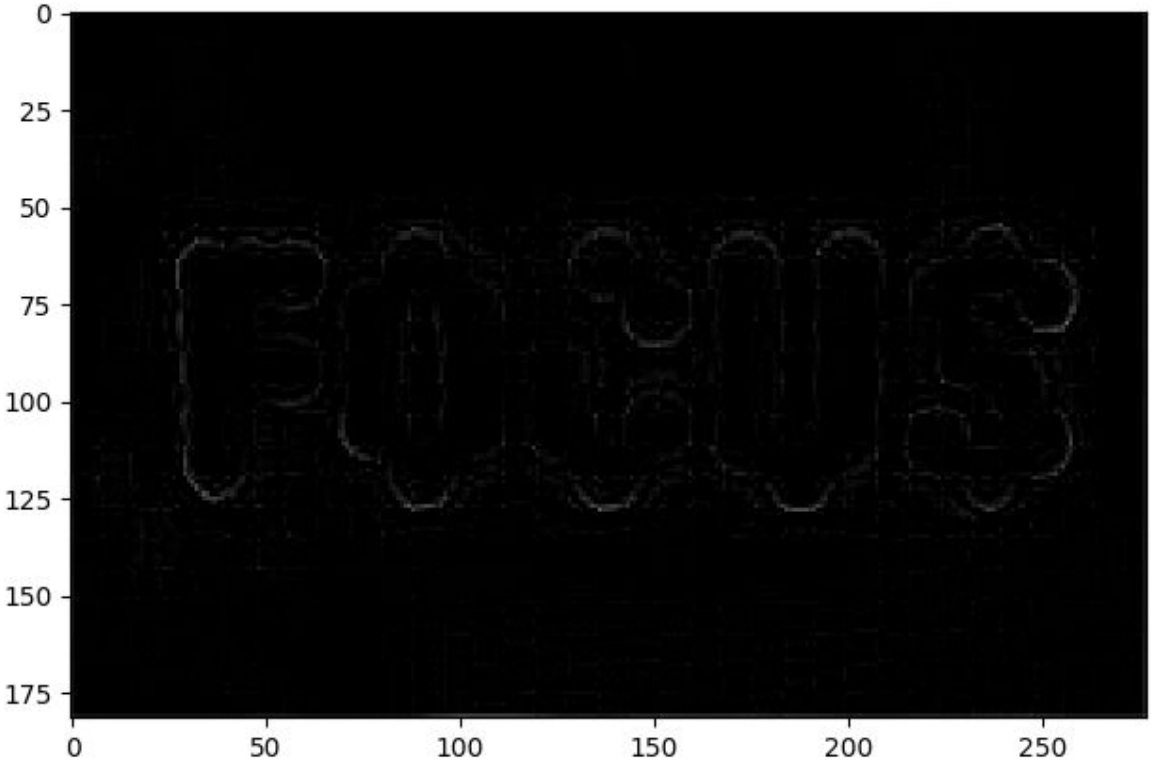


оригинал цветного изображения

маска 1 (без применения конвертации в одноканальное изображение)

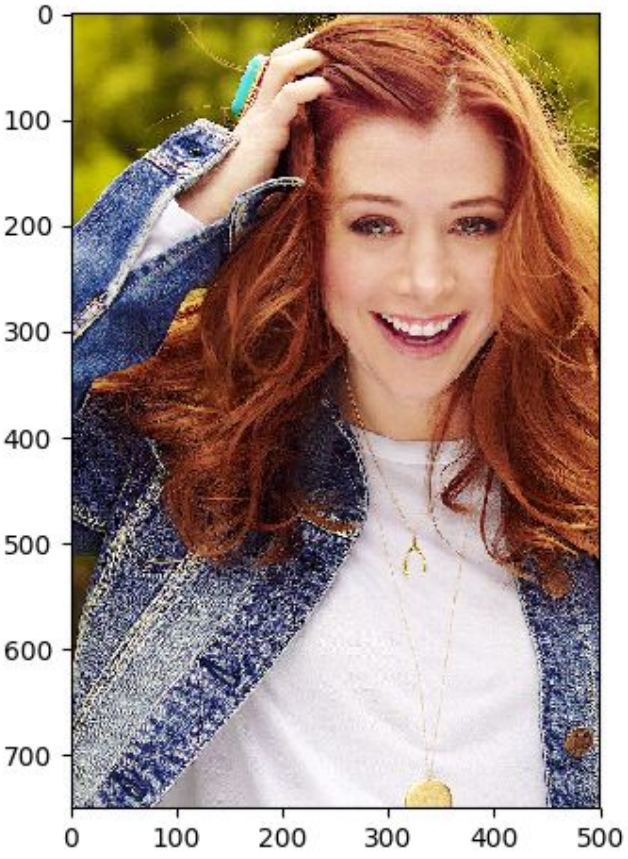


маска 2 (без применения конвертации в одноканальное изображение)

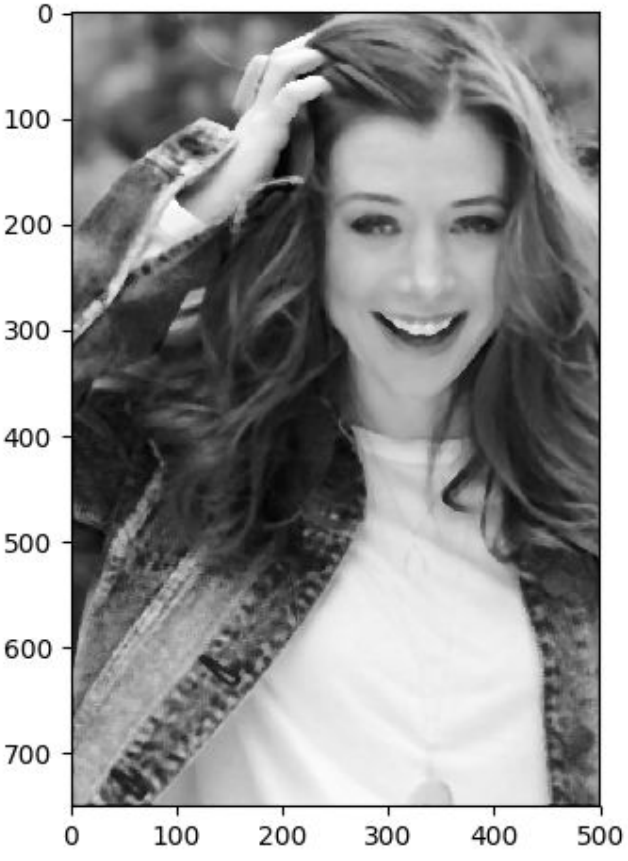


Медианный фильтр
Применение функции median с приведением цветного изображения к одноканальному

Оригинал изображения



Применение фильтра



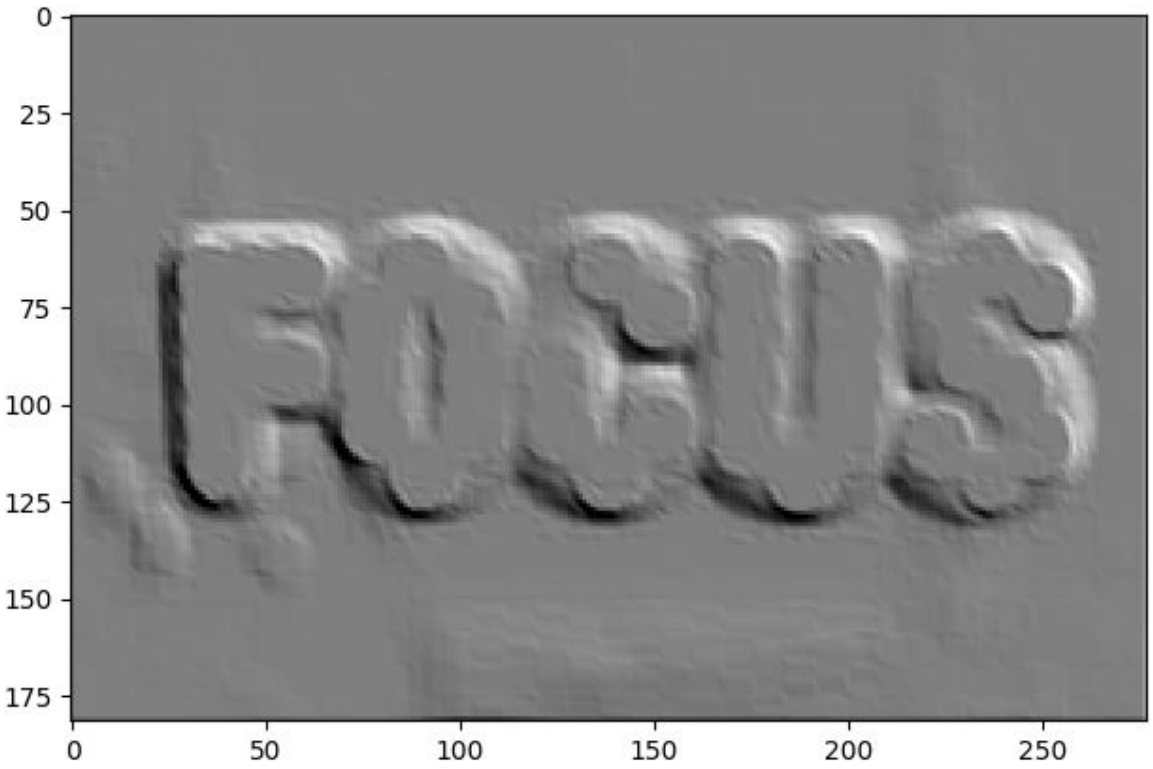
Тиснение
маски

1. np.array([[0, -1, 0], [1, 0, -1], [0, 1, 0]])
2. np.array([[0, 1, 0], [-1, 0, 1], [0, -1, 0]])

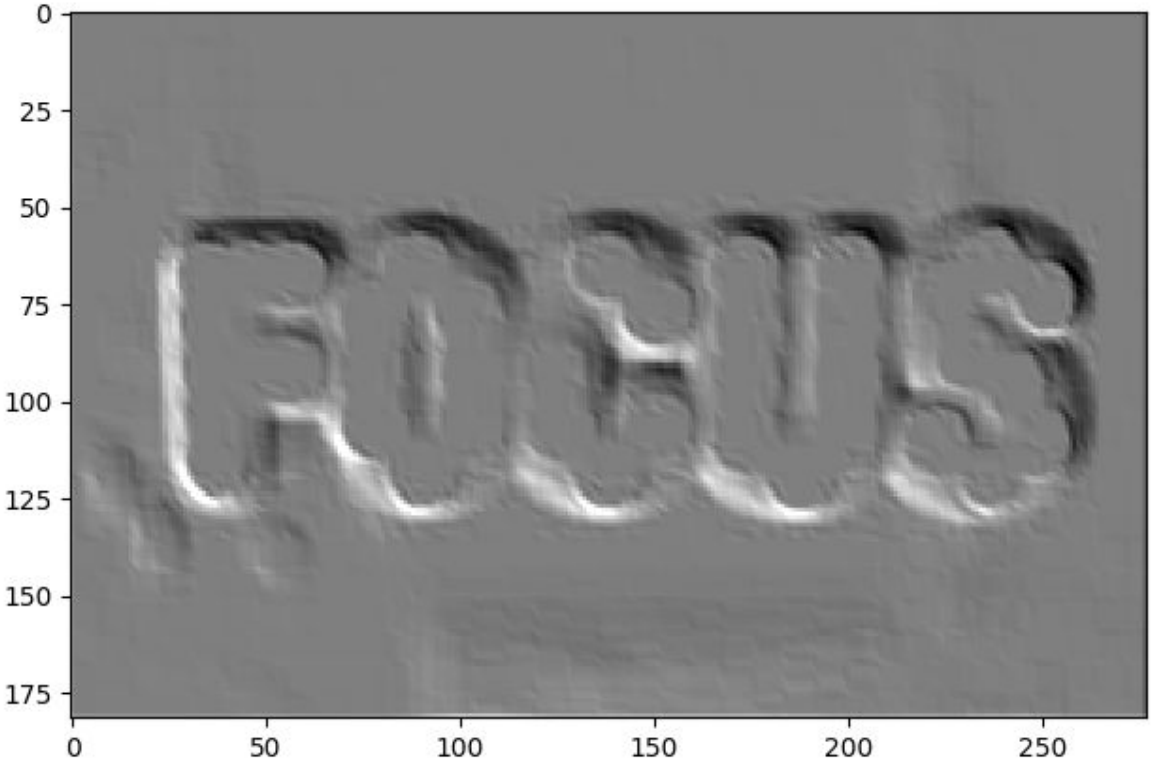
Расположение в матрице отрицательных и положительных величин 1 и -1 влияет на направление тиснения
Применение масок с конвертацией изображения в одноканальное.

оригинал

маска 1



маска 2

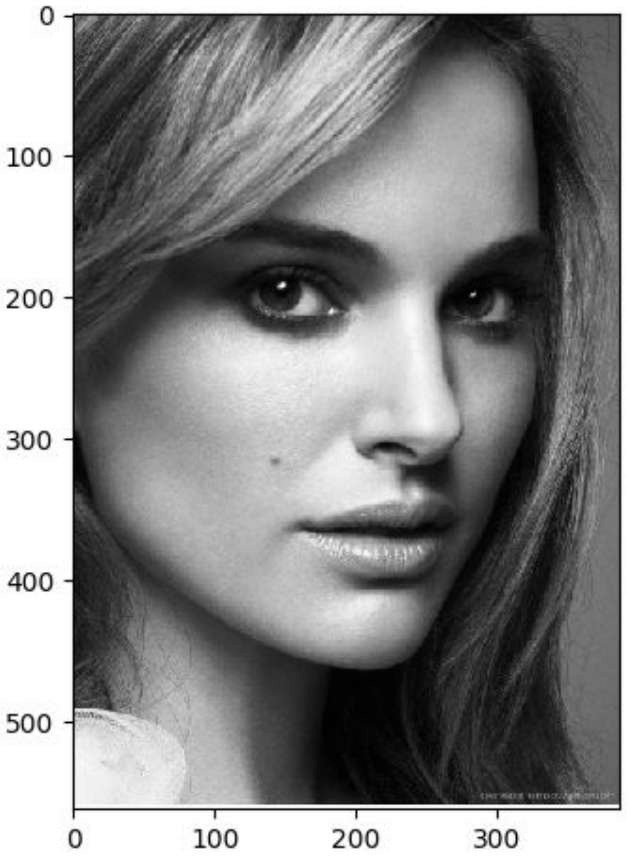


Акварельный фильтр

Акварельный фильтр это применение медианного фильтра к изображению, после чего на изображение накладывается маска подчеркивания контуров

np.array([[-1., -1., -1.], [-1., 17., -1.], [-1., -1., -1.]]) / 9.

оригинал



применение фильтрации

Фильтр размытия по Гауссу

Фильтр с применением к цветному изображению. Фильтр размытия по Гауссу не дает артефактов в отличии от применения маски

np.array([[1., 1., 1.], [1., 1., 1.], [1., 1., 1.]]) / 9.

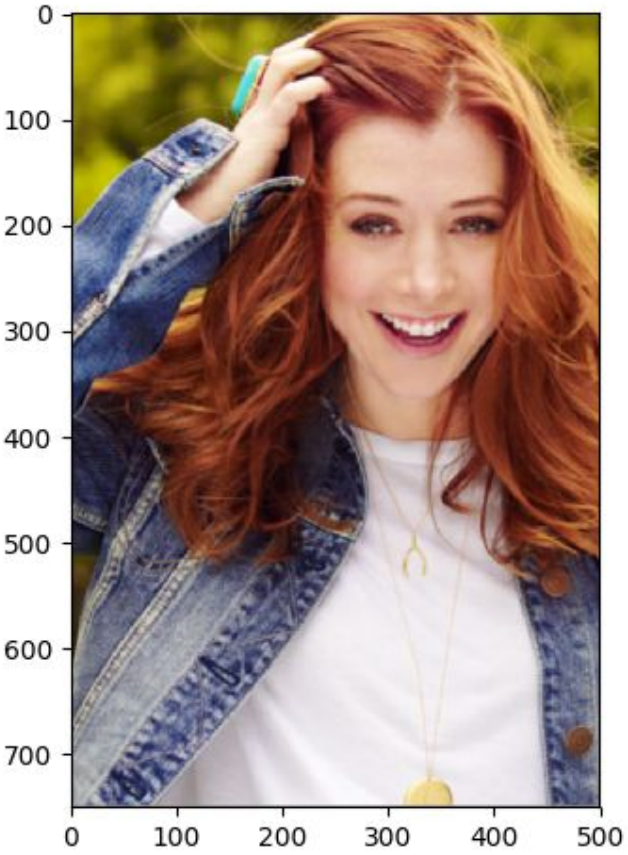
если применить размытие к фотографии с множеством деталей, к примеру крупный план газонной травы, в некоторых случаях могут быть заметны артефакты на изображении.

используется маска

np.array([[1, 4, 6, 4, 1], [4, 16, 24, 16, 4], [6, 24, 36, 24, 6], [4, 16, 24, 16, 4], [1, 4, 6, 4, 1]]) / 256.

оригинал

размытие по Гауссу



Фильтр выделения контуров размером 3 x 9 и 9 x 3
маска

1. 3 x 9

   [[-1. -1. -1.]

   [-1. -1. -1.]

   [-1. -1. -1.]

   [-1. -1. -1.]

   [-1. 26. -1.]

   [-1. -1. -1.]

   [-1. -1. -1.]

   [-1. -1. -1.]

   [-1. -1. -1.]]

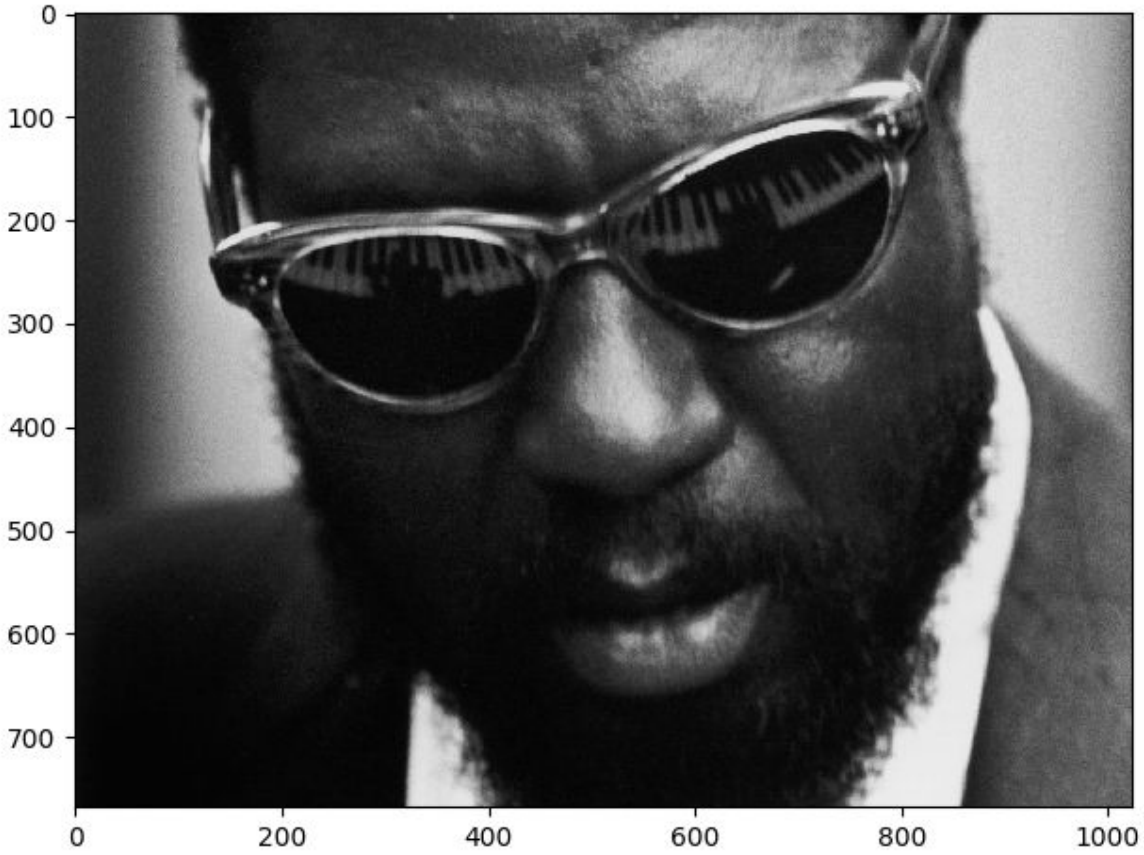
2. 9 x 3

   [[-1. -1. -1. -1. -1. -1. -1. -1. -1.]

   [-1. -1. -1. -1. 27. -1. -1. -1. -1.]

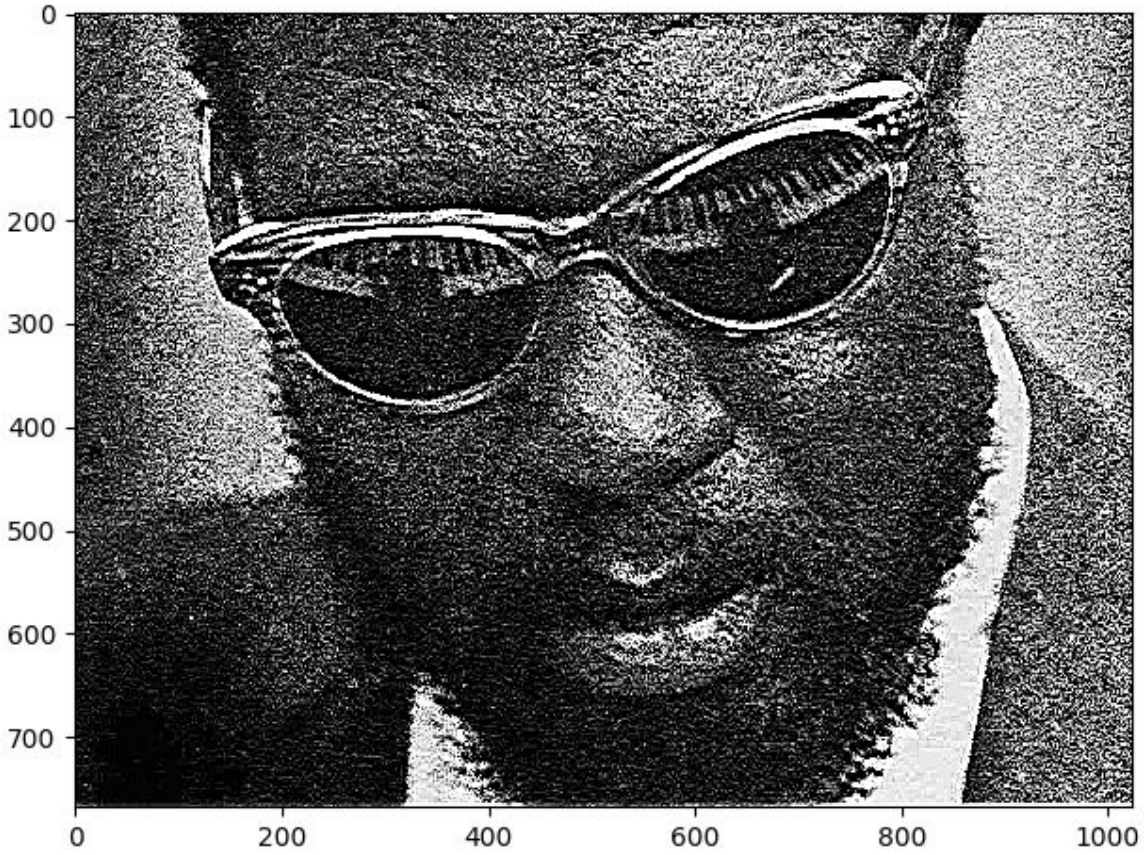
   [-1. -1. -1. -1. -1. -1. -1. -1. -1.]]

оригинал



маска 3 x 9

маска 9 x 3



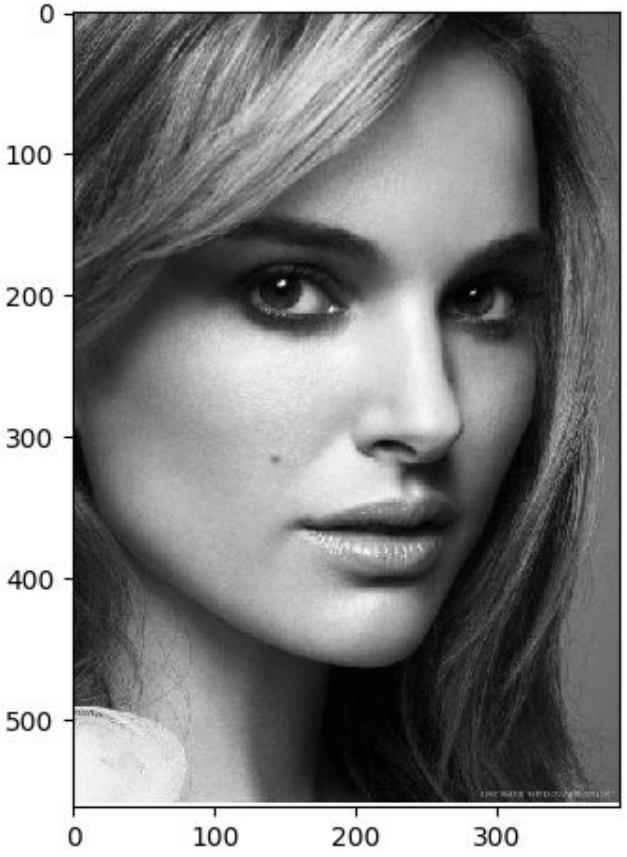
фильтр более глубокого тиснения 3 x 9 (фильтр тиснения произвольного размера)

маски

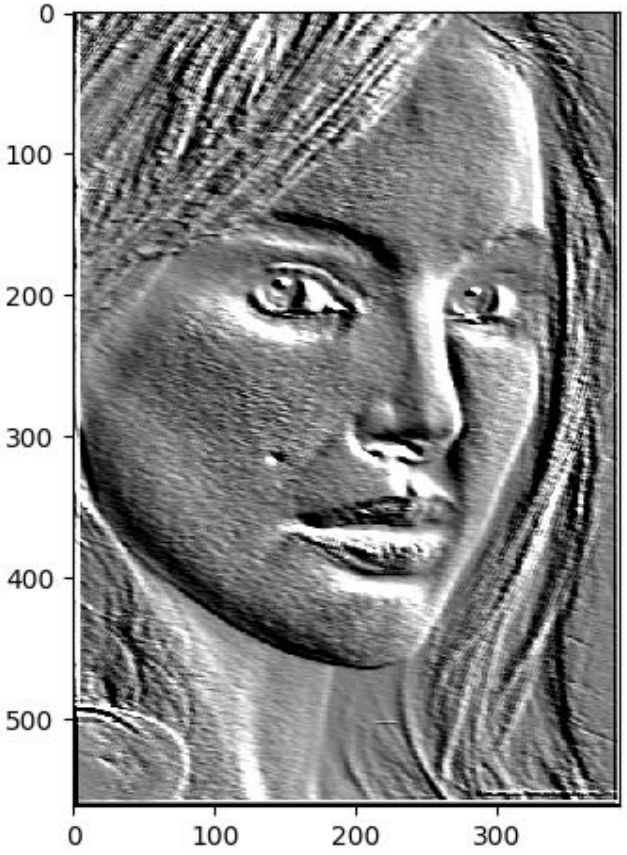
1. np.array([[0, -1, 0], [0, -1, 0], [1, -1, -1], [1, -1, -1], [1, 0, -1], [1, 1, -1], [1, 1, -1], [0, 1, 0], [0, 1, 0]])
2. инвертированная маска 1

   np.array([[0, -1, 0], [0, -1, 0], [1, -1, -1], [1, -1, -1], [1, 0, -1], [1, 1, -1], [1, 1, -1], [0, 1, 0], [0, 1, 0]]) * -1

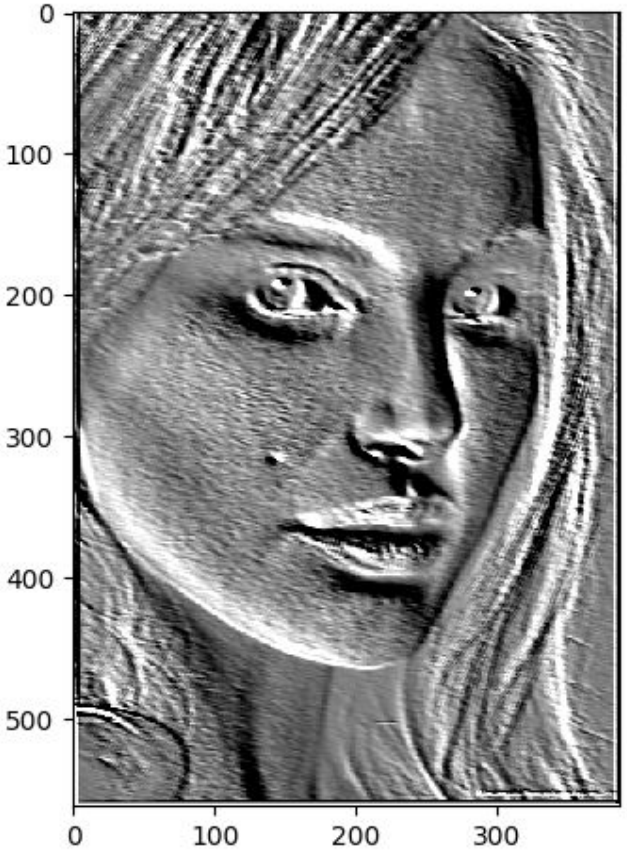
оригинал



маска 1



маска 2



Пример матрицы свертки, чтобы не изменять изображение.

Данная матрица никак не изменяет изображение.

```
np.array(
[
[0, 0, 0],
[0, 1, 0],
[0, 0, 0]
])
```

Обработка с применением матриц четного размера на данный момент возможна только с помощью библиотеки Open Source Computer Vision Library

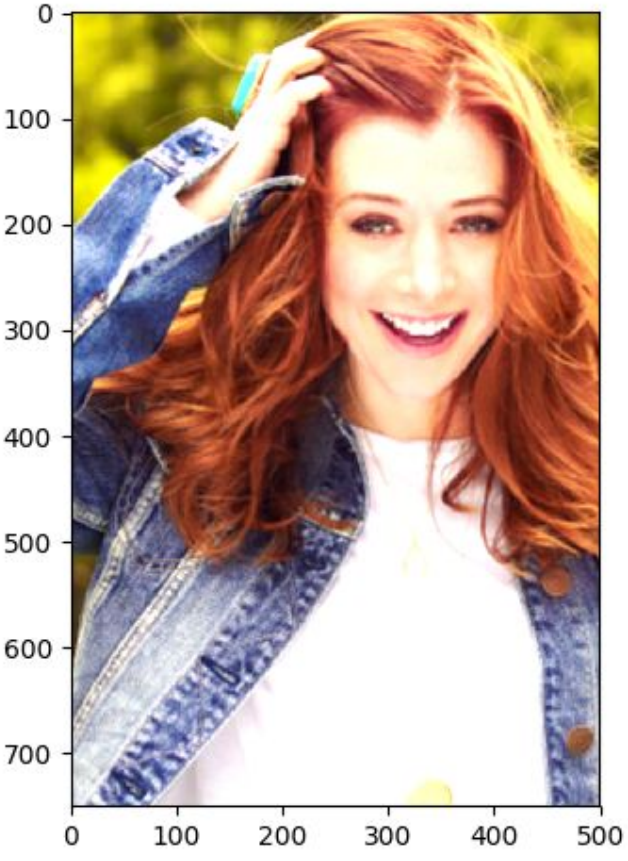
```
cv2.filter2D(img, -1, matrix)
```

пример применения маски размытия для цветного изображения с повышением контрастности

```
np.array([
[1., 1., 1., 1.],
[1., 1., 1., 1.],
[1., 1., 1., 1.],
[1., 1., 1., 1.]]) / 12
```

оригинал

маска



Исходный код https://github.com/mufasadev/practicalDSP

Авельцов Д. ЕПИМ - 1 - 18