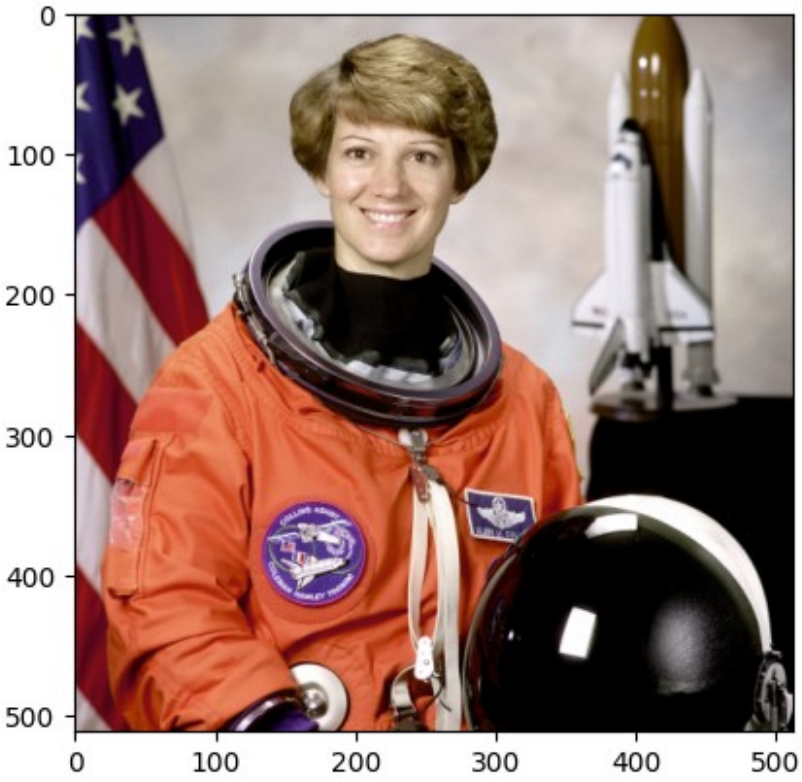
Name: Nisha Kini | Roll No: C050

Aim: To determine and highlight edges using DoG

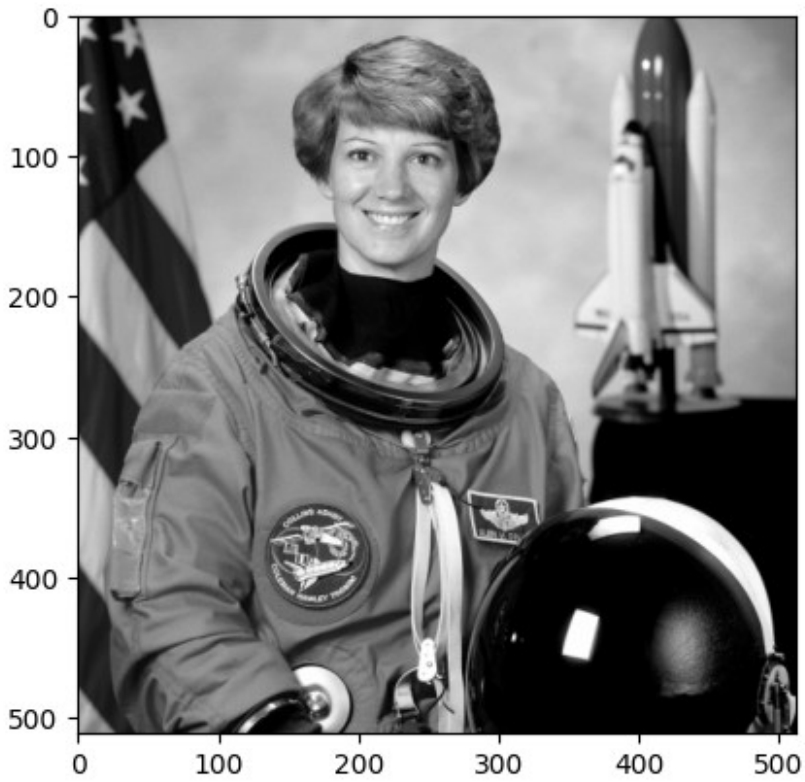
```
import cv2
from skimage import data
from skimage.color import rgb2gray
import numpy as np

import matplotlib.pyplot as plt

img = data.astronaut()
plt.imshow(img, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x782310dd8c50>
```



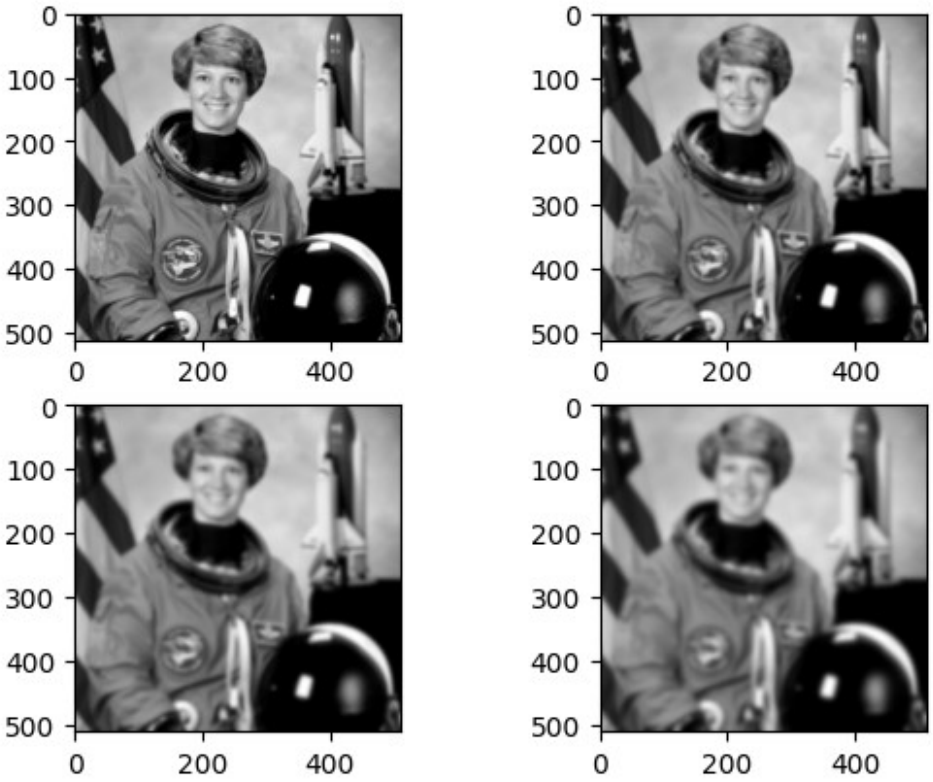

```
img_g= rgb2gray(img)
plt.imshow(img_g, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x782310b19850>
```



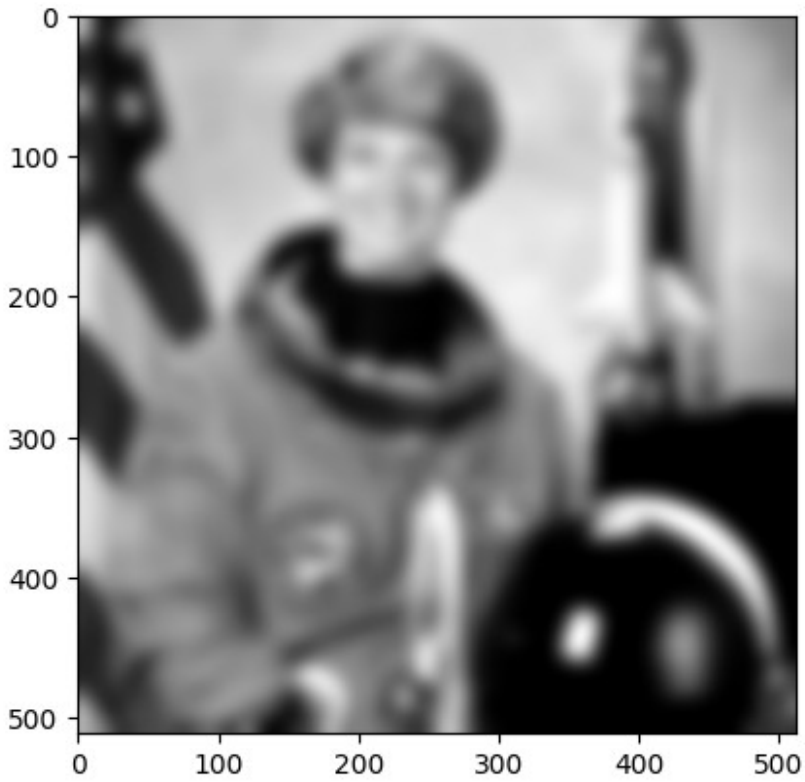

```
img_blur_1= cv2.GaussianBlur(img_g, (31,31), 1)
plt.subplot(2,2,1)
plt.imshow(img_blur_1, cmap='gray')
img_blur_2= cv2.GaussianBlur(img_g, (31,31), 2)
plt.subplot(2,2,2)
plt.imshow(img_blur_2, cmap='gray')
img_blur_3= cv2.GaussianBlur(img_g,(31,31), 3)
plt.subplot(2,2,3)
plt.imshow(img_blur_3, cmap='gray')
img_blur_4= cv2.GaussianBlur(img_g, (31,31), 4)
plt.subplot(2,2,4)
plt.imshow(img_blur_4, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7823109fc810>
```



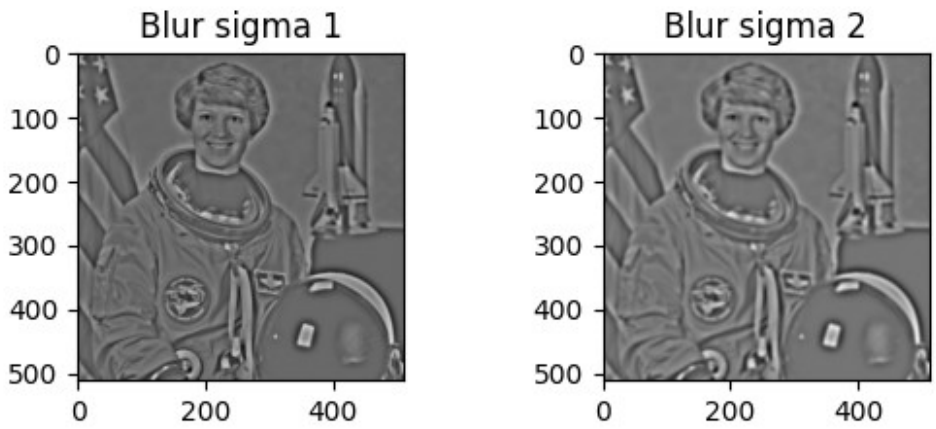

```
img_blur_hg= cv2.GaussianBlur(img_g,(31,31), 7)
plt.imshow(img_blur_hg, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7823108ebfd0>
```



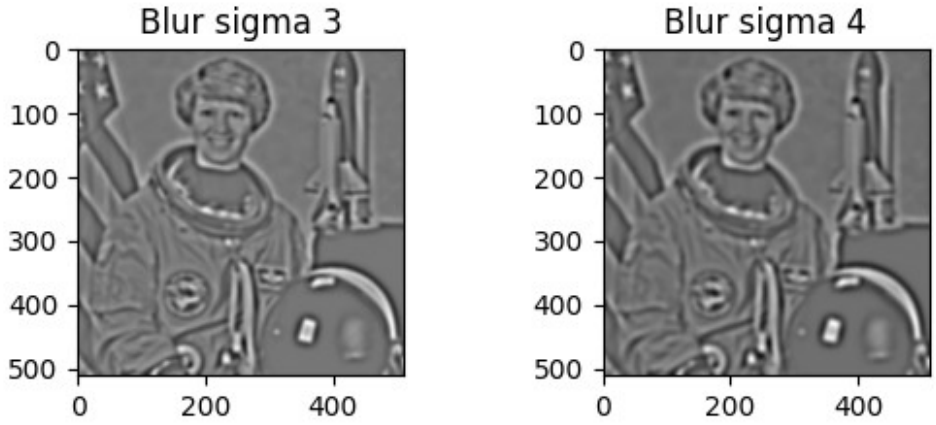

```
DoG_1hg= img_blur_1-img_blur_hg
DoG_2hg= img_blur_2-img_blur_hg
DoG_3hg= img_blur_3-img_blur_hg
DoG_4hg= img_blur_4-img_blur_hg

plt.subplot(2,2,1)
plt.imshow(DoG_1hg, cmap='gray')
plt.title('Blur sigma 1')
plt.subplot(2,2,2)
plt.imshow(DoG_2hg, cmap='gray')
plt.title('Blur sigma 2')
```

```
Text(0.5, 1.0, 'Blur sigma 2')
```



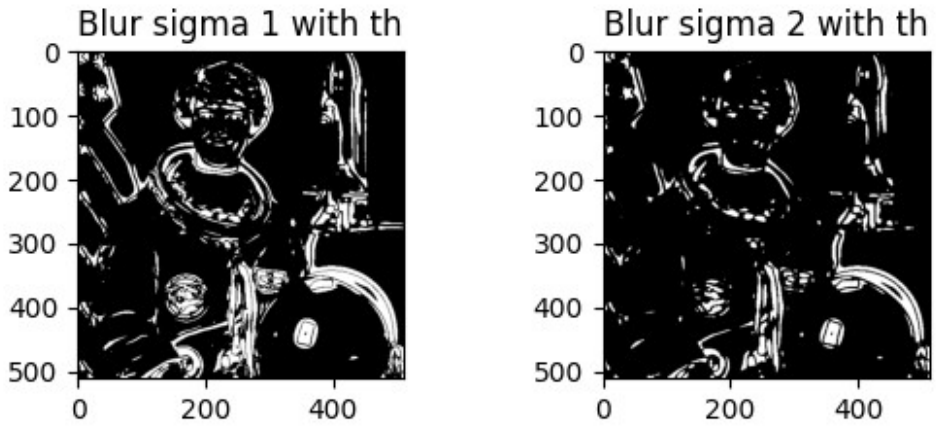

```
plt.subplot(2,2,3)
plt.imshow(DoG_3hg, cmap='gray')
plt.title('Blur sigma 3')
plt.subplot(2,2,4)
plt.imshow(DoG_4hg, cmap='gray')
plt.title('Blur sigma 4')
```

```
Text(0.5, 1.0, 'Blur sigma 4')
```

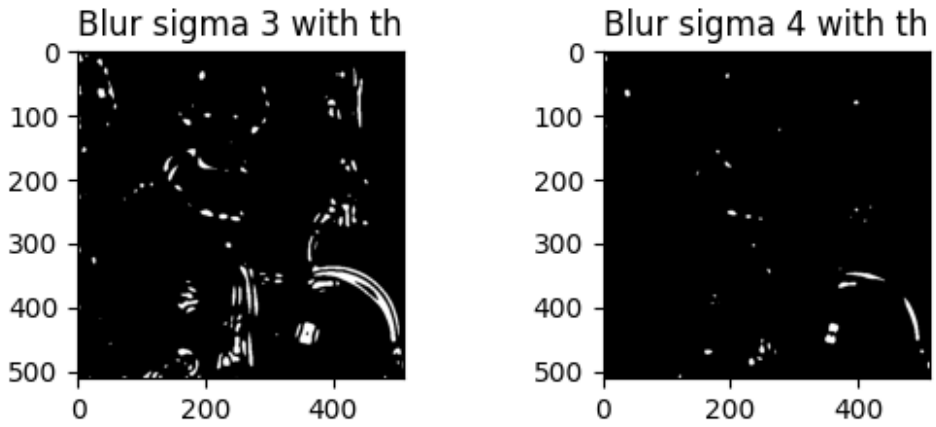



```
mx= np.max(DoG_1hg)
th= mx*0.2
DoG_1hg[np.abs(DoG_1hg)>th]=255
DoG_2hg[np.abs(DoG_2hg)>th]=255
DoG_3hg[np.abs(DoG_3hg)>th]=255
DoG_4hg[np.abs(DoG_4hg)>th]=255

plt.subplot(2,2,1)
plt.imshow(DoG_1hg, cmap='gray')
plt.title('Blur sigma 1 with th')
plt.subplot(2,2,2)
```

```
plt.imshow(DoG_2hg, cmap='gray')
plt.title('Blur sigma 2 with th')

Text(0.5, 1.0, 'Blur sigma 2 with th')
```




```
plt.subplot(2,2,3)
plt.imshow(DoG_3hg, cmap='gray')
plt.title('Blur sigma 3 with th')
plt.subplot(2,2,4)
plt.imshow(DoG_4hg, cmap='gray')
plt.title('Blur sigma 4 with th')

Text(0.5, 1.0, 'Blur sigma 4 with th')
```



Pre-Conclusion: Difference of Gaussian is applied to identify the edges of the given image for the following pairs of standard deviation, DoG shows different no of edge pixels.

1. Standard dev: 1,7
2. Standard dev: 2,7
3. Standard dev: 3,7
4. Standard dev: 4,7

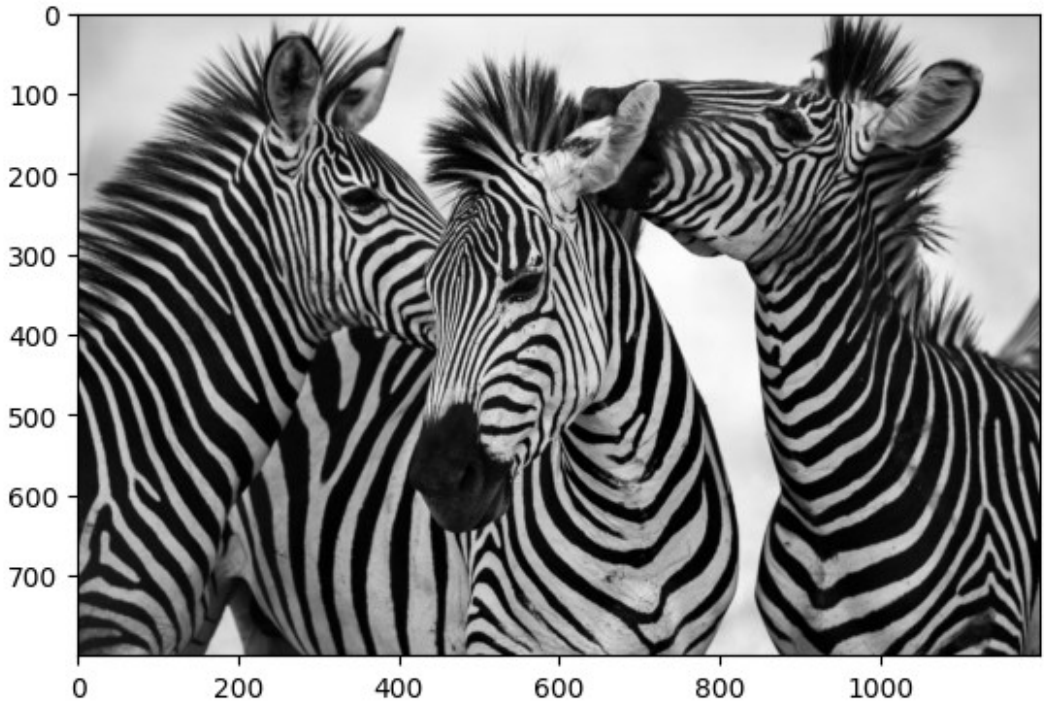
For 1st pair that is for std 1-7, the small edge pixels are noticeable in its corresponding DoG. For last pair, large pixels are retained, since std of 4 blurred image does not have small pixels. Hence, small edge pixels can be retained by using low sigma value(1 ftgi) & only large edge pixel can be identified using large value(4 ftgi)

Post-conclusion: If size is reduced to 15*15 however if size is reduced to 9*9 the DoG doesnt cross the th, which is 20% of mx value.

Self used image

```
imag = cv2.imread('/content/zebra.jpg')
img_g= rgb2gray(imag)
plt.imshow(img_g, cmap='gray')
```

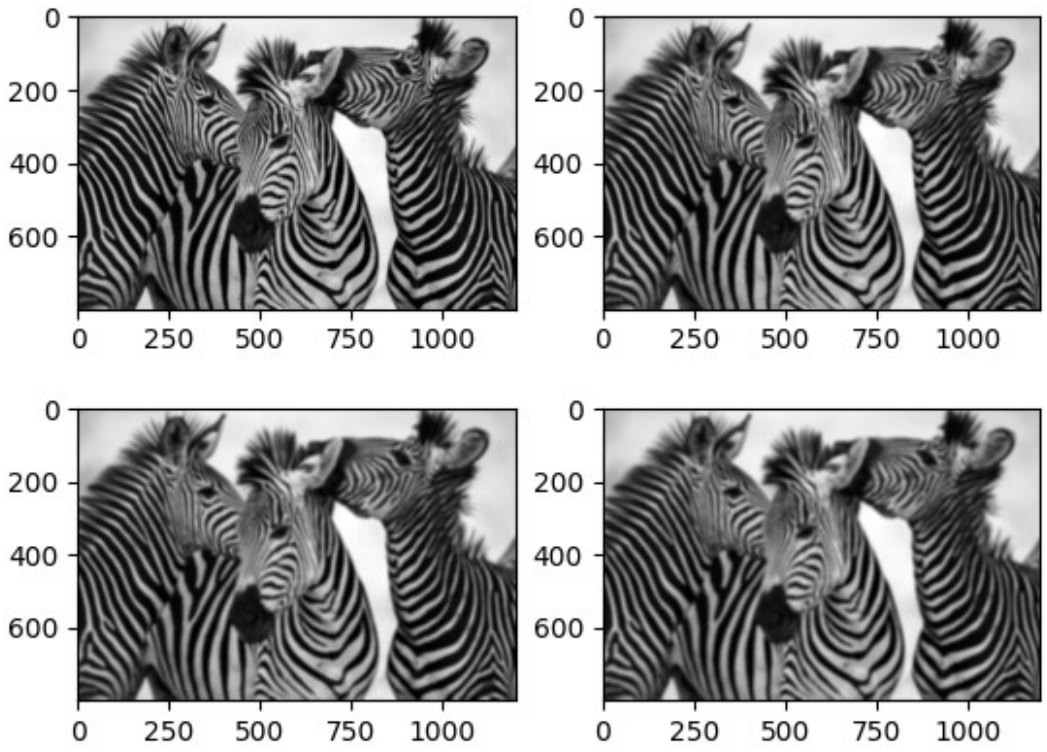
```
<matplotlib.image.AxesImage at 0x78231122c290>
```




```
img_blur_1= cv2.GaussianBlur(img_g, (15,15), 2)
plt.subplot(2,2,1)
plt.imshow(img_blur_1, cmap='gray')
img_blur_2= cv2.GaussianBlur(img_g, (15,15), 3)
plt.subplot(2,2,2)
plt.imshow(img_blur_2, cmap='gray')
img_blur_3= cv2.GaussianBlur(img_g, (15,15), 4)
plt.subplot(2,2,3)
plt.imshow(img_blur_3, cmap='gray')
img_blur_4= cv2.GaussianBlur(img_g, (15,15), 5)
```

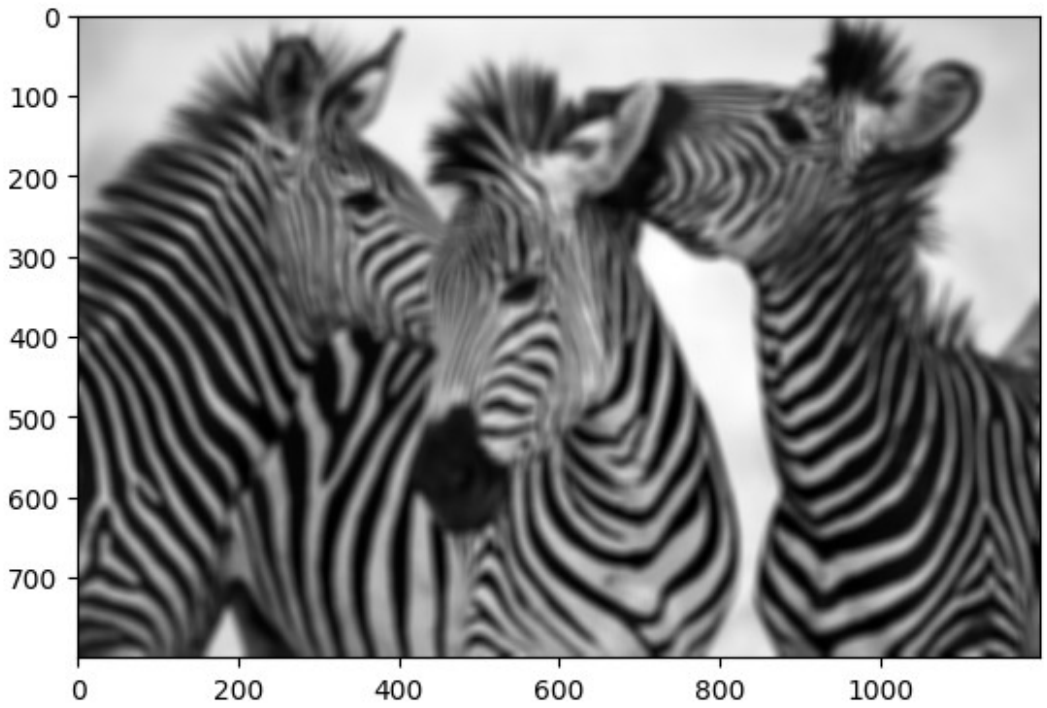
```
plt.subplot(2,2,4)
plt.imshow(img_blur_4, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x782311101110>
```

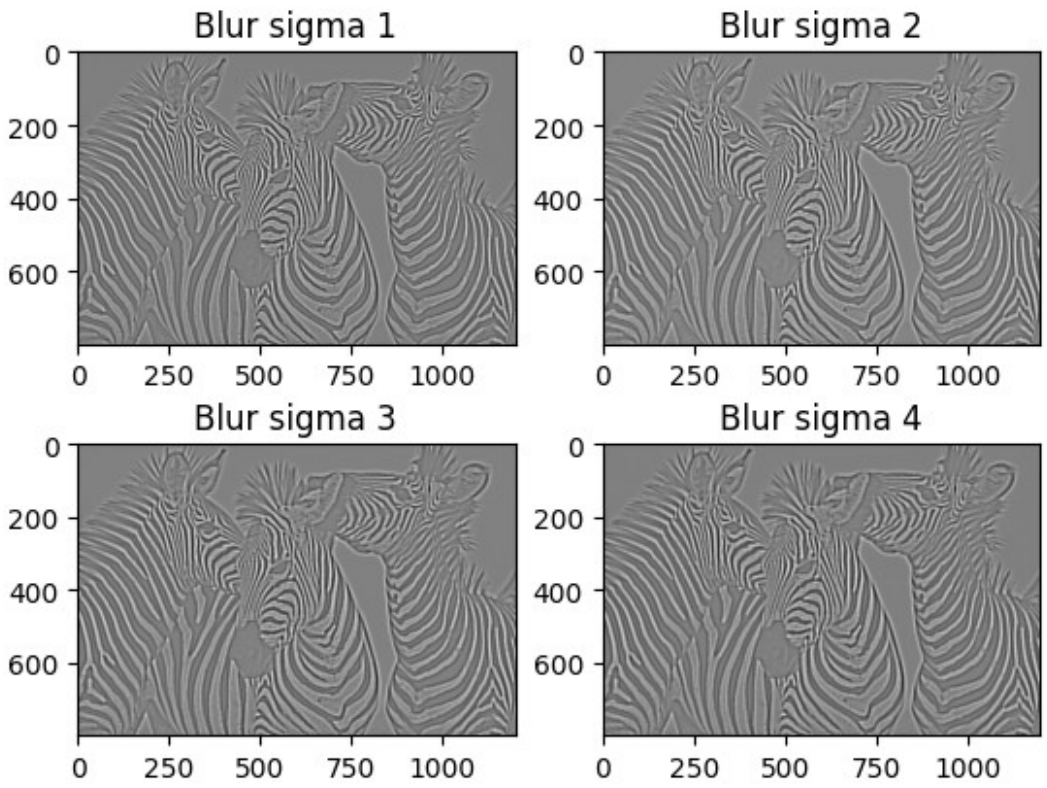



```
img_blur_hg= cv2.GaussianBlur(img_g, (15,15), 10)
plt.imshow(img_blur_hg, cmap='gray')
```

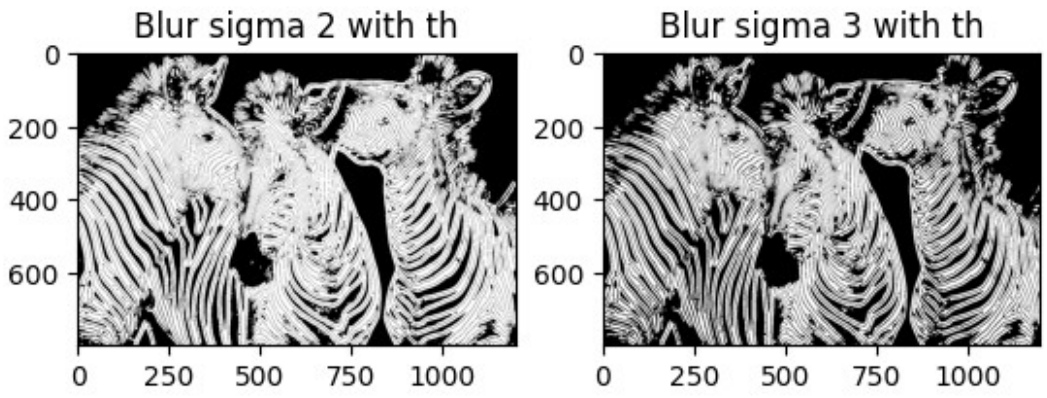
```
<matplotlib.image.AxesImage at 0x782310fffb90>
```



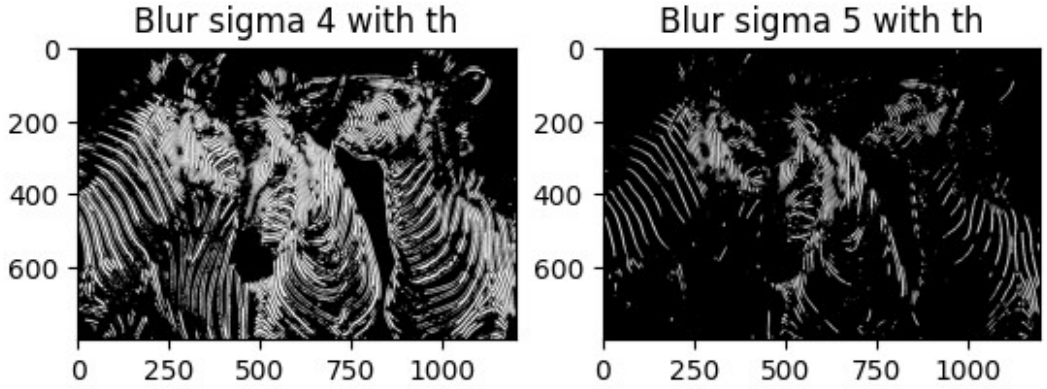

```
DoG_1hg= img_blur_1-img_blur_hg
DoG_2hg= img_blur_2-img_blur_hg
DoG_3hg= img_blur_3-img_blur_hg
DoG_4hg= img_blur_4-img_blur_hg

plt.subplot(2,2,1)
plt.imshow(DoG_1hg, cmap='gray')
plt.title('Blur sigma 1')
plt.subplot(2,2,2)
plt.imshow(DoG_2hg, cmap='gray')
plt.title('Blur sigma 2')
plt.subplot(2,2,3)
plt.imshow(DoG_3hg, cmap='gray')
plt.title('Blur sigma 3')
plt.subplot(2,2,4)
plt.imshow(DoG_4hg, cmap='gray')
plt.title('Blur sigma 4')

Text(0.5, 1.0, 'Blur sigma 4')
```




```
mx= np.max(DoG_1hg)
th= mx*0.01
DoG_1hg[np.abs(DoG_1hg)>th]=255
DoG_2hg[np.abs(DoG_2hg)>th]=255
DoG_3hg[np.abs(DoG_3hg)>th]=255
DoG_4hg[np.abs(DoG_4hg)>th]=255

plt.subplot(2,2,1)
plt.imshow(DoG_1hg, cmap='gray')
plt.title('Blur sigma 2 with th')
plt.subplot(2,2,2)
plt.imshow(DoG_2hg, cmap='gray')
plt.title('Blur sigma 3 with th')

Text(0.5, 1.0, 'Blur sigma 3 with th')
```




```
plt.subplot(2,2,3)
plt.imshow(DoG_3hg, cmap='gray')
plt.title('Blur sigma 4 with th')
plt.subplot(2,2,4)
plt.imshow(DoG_4hg, cmap='gray')
plt.title('Blur sigma 5 with th')

Text(0.5, 1.0, 'Blur sigma 5 with th')
```



If th is reduced from 20% to 10%, the edges appear thicker and additional small edge pixels also are retained, if th is increased to 30% small edge pixels disappear even for low values of stds,