OpenCV-python 学习笔记 OpenCV图像阈值

原始图像：

1. 简单阈值

使用`cv2.threshold()`和`cv2.adaptiveThreshold()`函数

当某个像素值高于阈值时，给这个像素赋予一个新值，否则我们赋予另外一个新值。实现这一功能的函数是`cv2.threshold()`。

这个函数的第一个参数是原图像,原图像应当是灰度图，第二个参数是用于对像素值分类的阈值。第三个参数是当像素值高于（或小于）阈值时，应当被赋予的新像素值。第四个参数为阈值方法，方法包括：

- cv2.THRESH_BINARY
- cv2.THRESH_BINARY_INV
- cv2.THRESH_TRUNC
- cv2.THRESH_TOZERO
- cv2.THRESH_TOZERO_INV

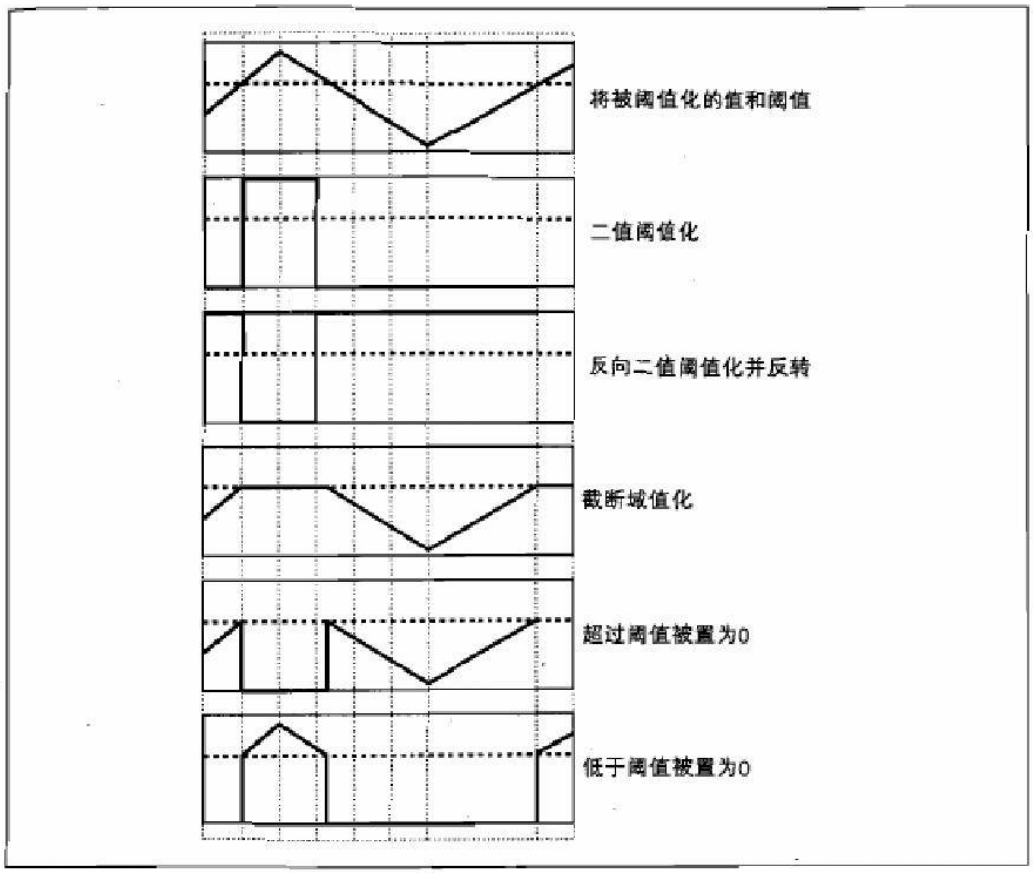


图 5-23：在 cvThreshold ()中不同阈值类型的操作结果。每个图表的水平虚线代表应用到最上方图的阈值，5 种阈值类型的操作结果列于其后 【136】

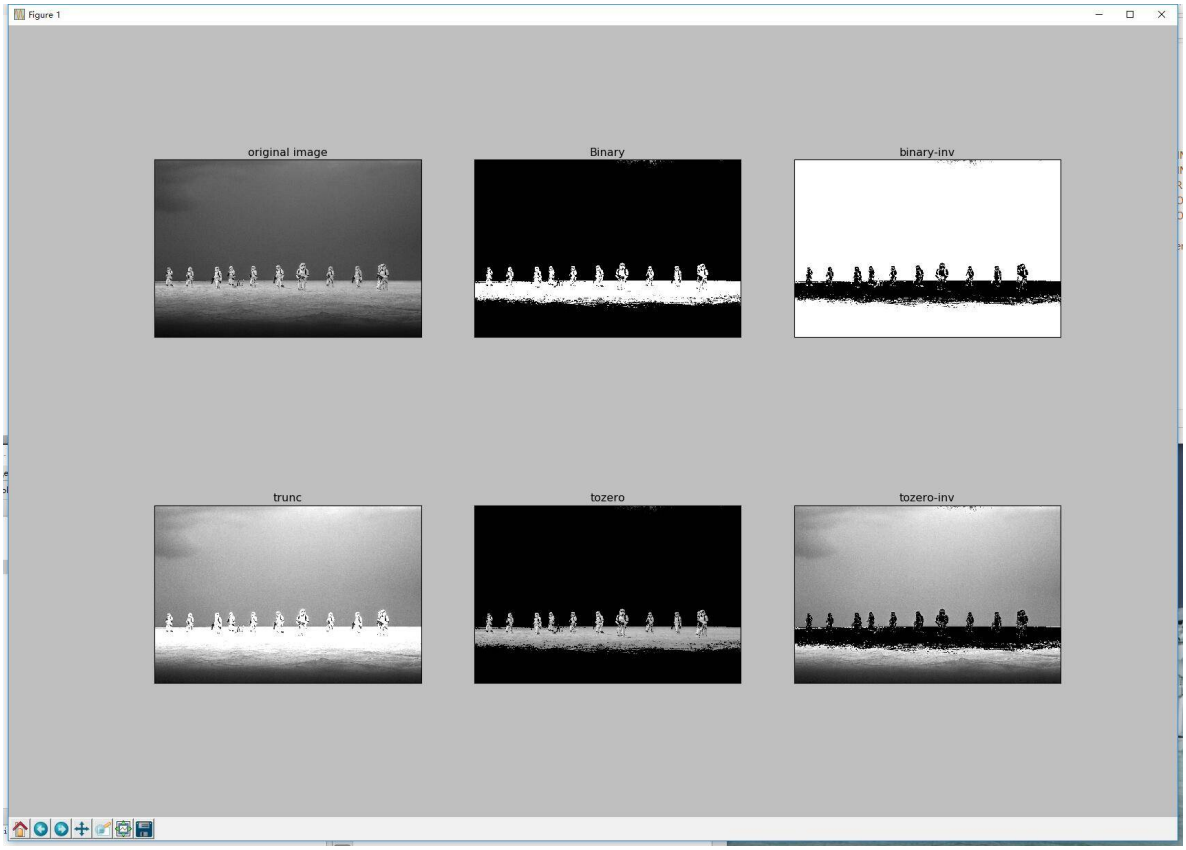
该函数有两个返回值，第一个是`retVal`，第二个是阈值化后的结果图像。

例如：

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('719100.jpg',0)
ret , thresh1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
ret , thresh2 = cv2.threshold(img,127,255,cv2.THRESH_BINARY_INV)
ret , thresh3 = cv2.threshold(img,127,255,cv2.THRESH_TRUNC)
ret , thresh4 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO)
ret , thresh5 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO_INV)

titles = ['original image','Binary','binary-inv','trunc','tozero','tozero-inv']
images = [img,thresh1,thresh2,thresh3,thresh4,thresh5]

for i in range(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')
```

```
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.show()
```

注：这里不推荐使用plt显示，与原图相差较大



2. 自适应阈值

根据图像上每一个小区域计算与之对应的阈值。因此在同一幅图像的不同区域采用不同的阈值，从而可以在亮度不同的情况下得到更好的结果。

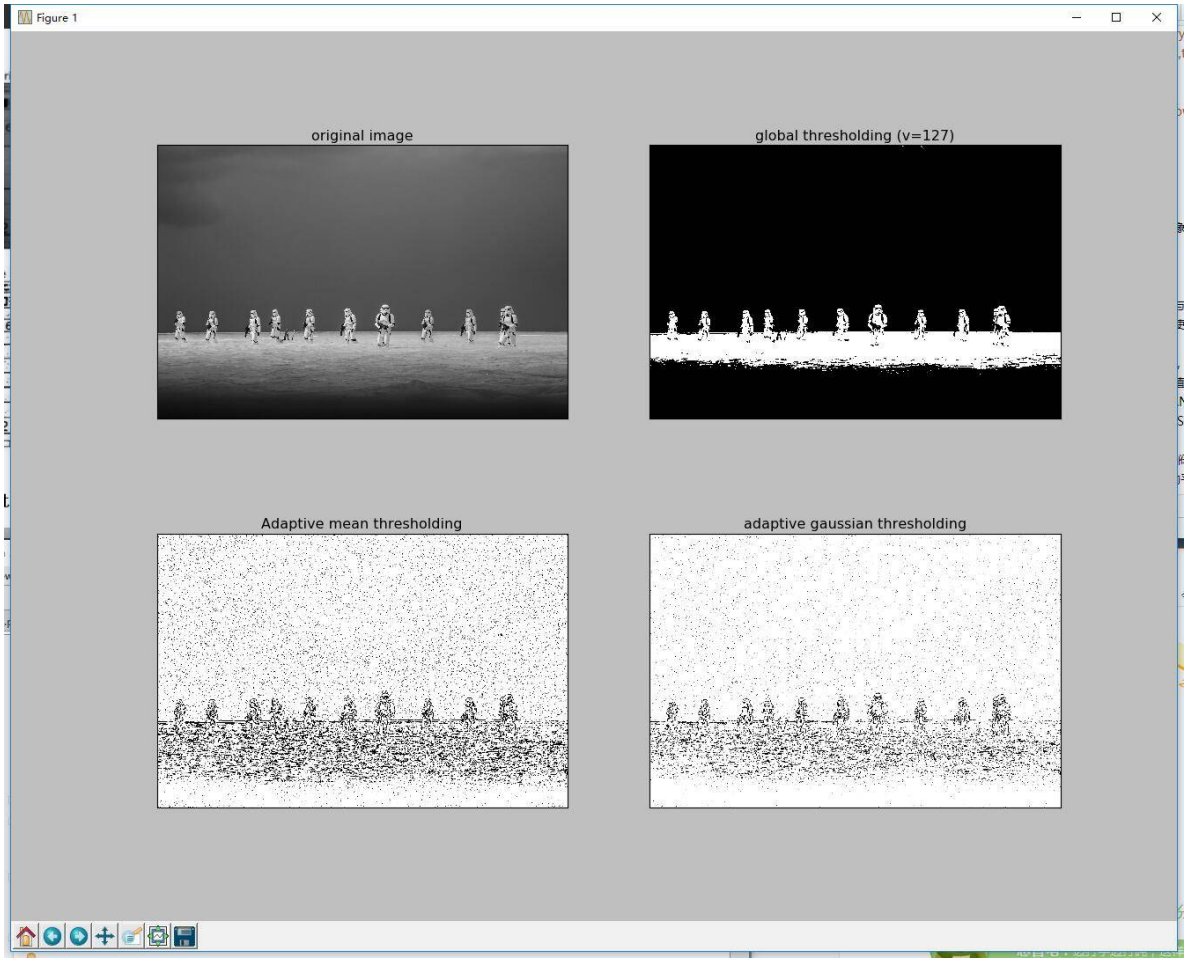
这种方法需要我们指定三个参数，返回值只有一个。
`cv2.adaptiveMethod()` 指定计算阀值的方法

`cv2.ADAPTIVE_THRESH_MEAN_C`:阀值取自相邻区域的平均值
`cv2.ADAPTIVE_THRESH_GAUSSIAN_C`:阀值取自相邻区域的加权和，权重为一个高斯窗口
`Block Size` 邻域大小（用来计算阀值的区域大小）
`C`这就是一个常数，阀值就等于的平均值或者加权平均值减去这个常数 。

例如：

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

```
4
5  img = cv2.imread('719100.jpg',0)
6  #中值滤波
7  img = cv2.medianBlur(img,5)
8
9  ret , th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
10 # 11为block size，2为C值
11 th2 =
   cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_MEAN_C ,
   cv2.THRESH_BINARY,11,2 )
12 th3 =
   cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C
   , cv2.THRESH_BINARY,11,2)
13
14 titles = ['original image' , 'global thresholding
   (v=127)','Adaptive mean thresholding',
15           'adaptive gaussian thresholding']
16 images = [img,th1,th2,th3]
17
18 for i in range(4):
19     plt.subplot(2,2,i+1),plt.imshow(images[i],'gray')
20     plt.title(titles[i])
21     plt.xticks([]),plt.yticks([])
22
23 plt.show()
```



3. Otsu's二值化

我们前面说到，`cv2.threshold`函数是有两个返回值的，前面一直用的第二个返回值，也就是阈值处理后的图像，那么第一个返回值（得到图像的阈值）将会在这里用到。
前面对于阈值的处理上，我们选择的阈值都是127，那么实际情况下，怎么去选择这个127呢？有的图像可能阈值不是127得到的效果更好。那么这里我们需要算法自己去寻找到一个阈值，而Otsu's就可以自己找到一个认为最好的阈值。并且Otsu's非常适合于图像灰度直方图具有双峰的情况，他会在双峰之间找到一个值作为阈值，对于非双峰图像，可能并不是很好用。那么经过Otsu's得到的那个阈值就是函数`cv2.threshold`的第一个参数了。因为Otsu's方法会产生一个阈值，那么函数`cv2.threshold`的的第二个参数（设置阈值）就是0了，并且在`cv2.threshold`的方法参数中还得加上语句`cv2.THRESH_OTSU`。

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread('719100.jpg',0)

ret1,th1=cv2.threshold(img,127,255,cv2.THRESH_BINARY)

```

```
ret2,th2=cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
#(5,5)为高斯核的大小，0为标准差
blur= cv2.GaussianBlur(img,(5,5),0)

#阀值一定要设为0
ret3,th3=cv2.threshold(blur,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

images=[img,0,th1,
         img,0,th2,
         img,0,th3]
titles =['original noisy image','histogram','global thresholding(v=127)',
          'original noisy image','histogram',"otsu's thresholding",
          'gaussian giltered image','histogram',"otus's thresholding"]
#这里使用了pyplot中画直方图的方法，plt.hist要注意的是他的参数是一维数组
#所以这里使用了（numpy）ravel方法，将多维数组转换成一维，也可以使用flatten方法
for i in range(3):
    plt.subplot(3,3,i*3+1),plt.imshow(images[i*3],'gray')
    plt.title(titles[i*3]),plt.xticks([]),plt.yticks([])
    plt.subplot(3,3,i*3+2),plt.hist(images[i*3].ravel(),256)
    plt.title(titles[i*3+1]),plt.xticks([]),plt.yticks([])
    plt.subplot(3,3,i*3+3),plt.imshow(images[i*3+2],'gray')
    plt.title(titles[i*3+2]),plt.xticks([]),plt.yticks([])

plt.show()
```

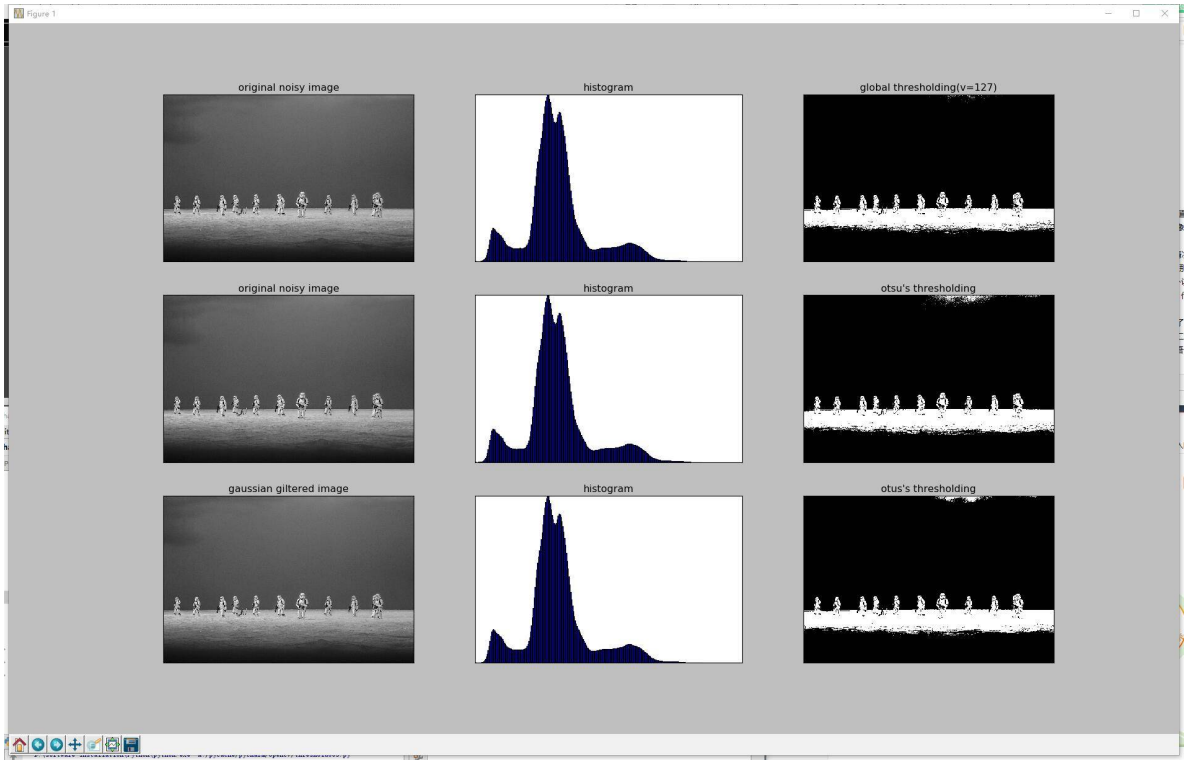


Figure 1
original noisy image
histogram
global thresholding(v=127)
original noisy image
histogram
otsu's thresholding
gaussian giltered image
histogram
otus's thresholding