# Canny edge detection

$$Canny\\ edge\\ detection$$

$$kai\ huang$$

$$April\ 29, 2024$$

# 1.The outline

**1.) Use a Gaussian filter to smooth out the image and filter out noise**

**2.)Calculate the gradient intensity and direction of each pixel in the image**

**3.)Non-maximal (NMS) suppression is applied to eliminate spurious responses from edge detection**

**4.)Apply double threshold detection to identify real and potential edges**

**5.)Edge detection is finally accomplished by suppressing isolated weakened edges**

# 2.Gaussian filter

## The generate background

Remove noise from the image and smooth out the entire image

## Principle analysis

Using the convolutional kernel technique and normalization processing, the closer to the core the value, the larger the value, showing a Gaussian (normal) distribution, so that the part close to the brightness of the graph is retained and the part that is far away is not visible

## The formula

for exmple:
this is a 5*5 Gaussian convolutional kernel

$$G = \frac{1}{275} * \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

（note:The first is origial image and the second is the image after been worked

# 3.Gradients and their direction

## The generate backgrond

Through mathematical gradients, we look for the difference between the inside and outside of the contours of the image (iconographic gradients seem to be somewhat different from mathematics, which refers to direction), and here refers to both direction and the size of the difference

### The formula

$$\theta = arctan(Gy/Gx)\ \ slope$$

**sobel operator**

$$G = \sqrt{Gx^2 + Gy^2}$$

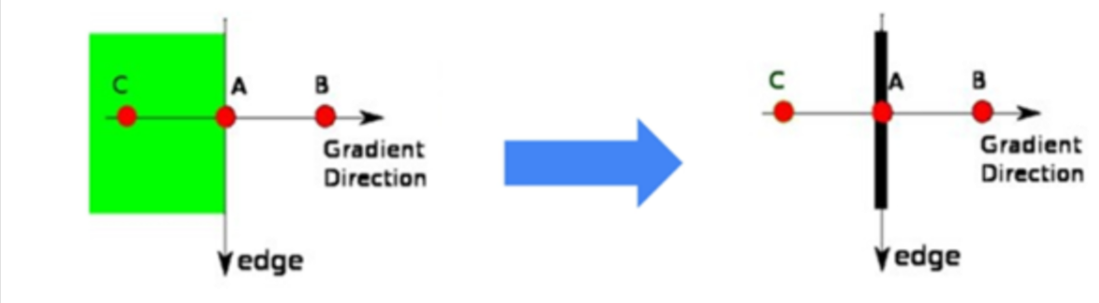
$$Sx = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad Sy = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

# 4.Non-Maximum Suppression (NMS algorithm)

## The backgrond

For each pixel, we do non-maximum suppression, and for non-edge pixels, we filter out the blurred boundaries to make the blurry boundaries clear

### The rules



#### 1.Approximate

Approximate the gradient values

#### 2.fix

Compare the gradient intensity of the pixel and its gradient in the positive and negative directions, if the pixel intensity is the largest, keep it, otherwise set it to 0 for suppression

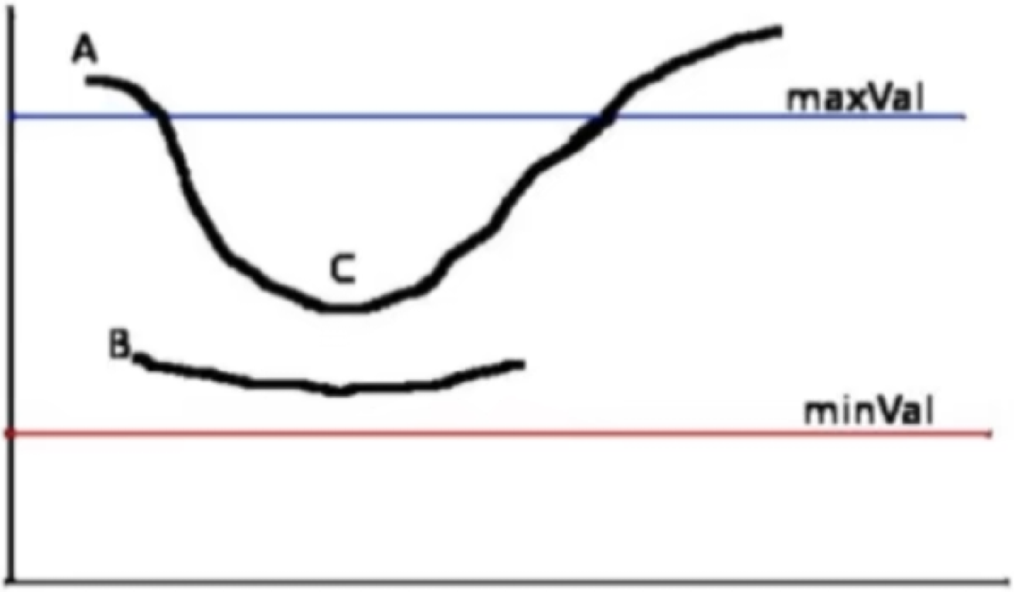
# 5.Double-threshold detection

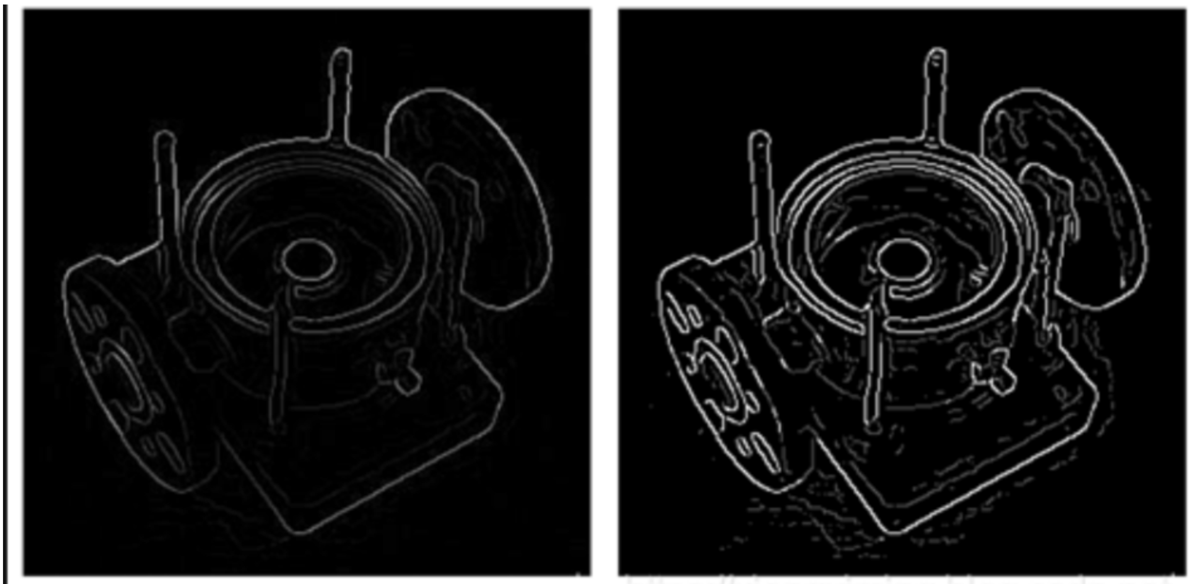
### The backgrond

After non-maximum suppression, there are still a lot of noise spots in the image

### Principle analysis

If the pixels in the image are larger than the upper bound of the threshold, it is considered to be retained (called the strong boundary), and if it is less than the lower bound of the threshold, it is considered to be a candidate (called the weak boundary) and needs to be further processed



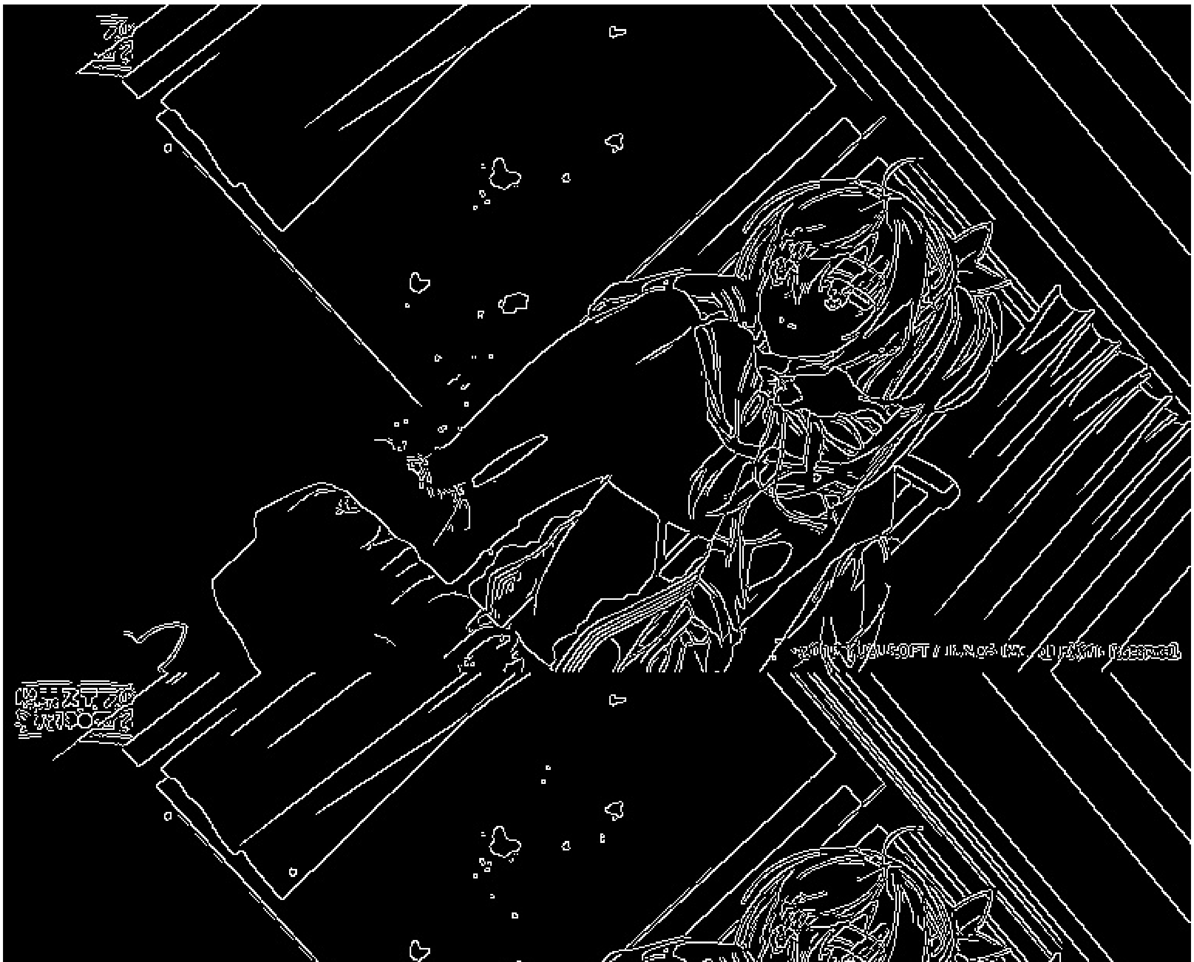


(a) Edges after non-maximum sup- (b) Double thresholding

## Comparison of high and low thresholds

(note:In the second image,the first one is hign thresholds and the second is low thresholds

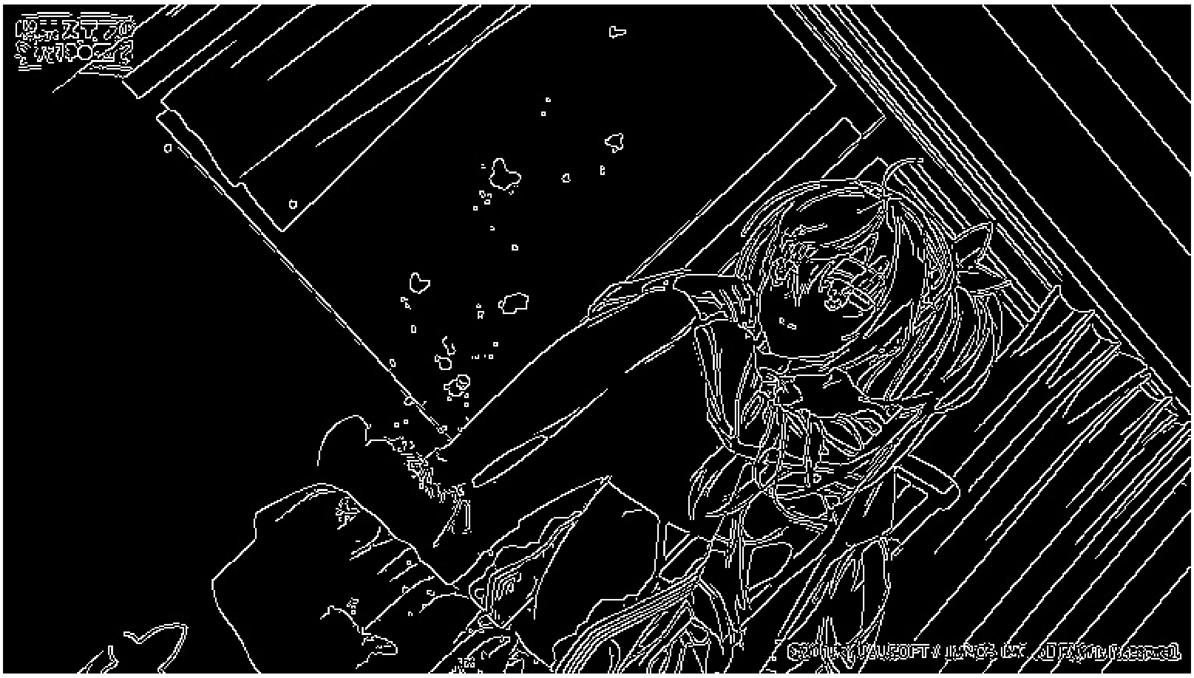
喫茶ステラと死神の蝶
©2019 YUZUSOFT / JUNOS INC. All Rights Reserved.



# 6.Weakened edge

Boundaries are tracked via the canny operator

©2019 YUZUSOFT / JUNOS INC. All Rights Reserved.



```
#show function
def cv_show(img,x):
    cv.imshow('name',img)
    cv.imwrite(f"D:\project.test\canny/Canny_test00{x}.jpg",img)
    cv.waitKey(0)
    cv.destroyAllWindows()
#The image is read and grayscaled
img=cv.imread("D:\project.test\canny/cafestella.jpg",0)
#resize my image
img=cv.resize(img,(0,0),fx=0.4,fy=0.4)
#Gaussian filtering
img=cv.GaussianBlur(img,(5,5),0)
cv_show(img,0)
#canny operator
```

```
v1=cv2.Canny(img,80,150)
v2=cv2.Canny(img,50,100)
#show my image
res=np.vstack((v1,v2))
cv_show(res,1)
```