Open Source CV image processing and drawing text line segments

1、OpenCV image grayscale processing

1.1、Grayscale image

The process of converting color images into grayscale images is the grayscale processing of the images. The color of each pixel in a color image is determined by three components: R, G, and B, and each component can take a value of 0-255. This allows a pixel to have a color range of over 16 million (256 * 256 * 256=1677256). A grayscale image is a special type of color image with the same components of R, G, and B, and one pixel has a range of 256 variations. Therefore, in digital image processing, various formats of images are generally converted into grayscale images to reduce the computational complexity of subsequent images. The description of grayscale images, like color images, still reflects the distribution and characteristics of the overall and local chromaticity and brightness levels of the entire image.

1.2、Image grayscale processing

Grayscale processing is the process of converting a color image into a grayscale image. Color images are divided into three components: R, G, and B, each displaying various colors such as red, green, and blue. Grayscale is the process of making the R, G, and B components of colors equal. Pixels with higher grayscale values are brighter (with a maximum pixel value of 255, indicating white), while those with lower grayscale values are darker (with a minimum pixel value of 0, indicating black).

The core idea of image grayscale is that R=G=B, which is also known as grayscale value.

1. Maximum method: Make the converted values of R, G, and B equal to the largest of the three values before conversion, that is, R=G=B=max (R, G, B). The grayscale image converted by this method has high brightness.
2. Average method: The values of R, G, and B after conversion are the average values of R, G, and B before conversion. That is: R=G=B=(R+G+B)/3. This method produces a softer grayscale image.

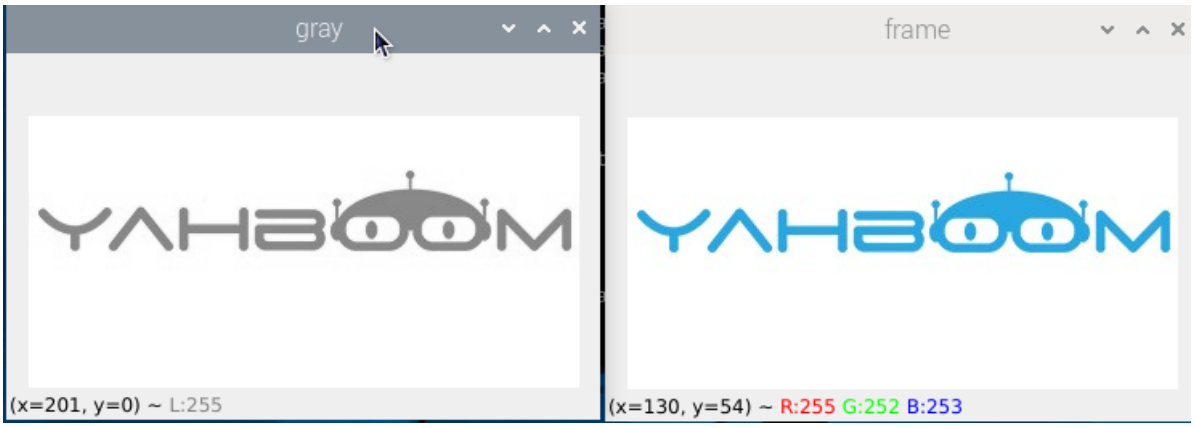
In OpenCV, use cv2. cvtColor (img1, cv2. COLOR_BGR2GRAY) to achieve grayscale processing of images

1.3、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_1.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('gray', gray)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

After running the program, the effect image,



2、OpenCV image binarization processing

2.1、The core idea of binarization

Set a threshold, with values greater than 0 (black) or 255 (white), to make the image a black and white image. The threshold can be fixed or adaptive. The adaptive threshold is generally a comparison between the average value of a pixel in a region and the pixel in the middle order, or the weighted sum of Gaussian distributions. A difference can be set or not set.

2.2、The threshold function is provided in Python OpenCV:

cv2.threshold (src, threshold, maxValue, thresholdType)

Parameter meanings:

src: Original image

threshold: Current threshold

maxVal: Maximum threshold, usually 255

thresholdType: Threshold types generally have the following values

enum ThresholdTypes { THRESH_BINARY = 0, #The grayscale value of pixels larger than the threshold is set to maxValue (such as a maximum of 255 for 8-bit grayscale values), and the grayscale value of pixels smaller than the threshold is set to 0. THRESH-BINARY-INV=1, # Pixels greater than the threshold have their grayscale value set to 0, while pixels less than the threshold have their grayscale value set to maxValue. THRESH-TRUNC=2, # Pixels above the threshold have

a grayscale value set to 0, while pixels below the threshold have a grayscale value set to maxValue. THRESH-TOZERO=3, # Pixels with grayscale values below this threshold do not undergo any changes, while those with grayscale values above this threshold all become 0. THRESH-TOZERO-INV=4 # Pixels with grayscale values greater than this threshold will not be changed, while pixels with grayscale values less than this threshold will all have grayscale values set to 0. }

return：

retval：Consistent with the threshold parameter

dst： result image

notes：Before binarizing, we need to grayscale the color image to obtain a grayscale image.

2.3、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_2.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    ret,thresh1=cv2.threshold(gray,180,255,cv2.THRESH_BINARY_INV)
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('gray', gray)
        cv2.imshow("binary",thresh1)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

After running the program, the effect image，

3、OpenCV Image Edge Detection

3.1、The purpose of image edge detection

Significantly reduce the data size of the image while retaining its original image attributes. At present, there are multiple algorithms available for edge detection. Although the Canny algorithm is ancient, it can be said to be a standard algorithm for edge detection and is still widely used in research.

3.2、Canny edge detection algorithm

Among the currently commonly used edge detection methods, the Canny edge detection algorithm is one of the methods with strict definitions that can provide good and reliable detection. Due to its advantages of meeting the three standards of edge detection and simple implementation process, it has become one of the most popular algorithms for edge detection.

The Canny edge detection algorithm can be divided into the following 5 steps：

（1）. Using Gaussian filters to smooth images and filter out noise

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

（3）. Apply Non Maximum Suppression to eliminate spurious responses caused by edge detection

（4）. Applying Double Threshold Detection to Determine True and Potential Edges

（5）. Edge detection is ultimately achieved by suppressing isolated weak edges

3.3、How do we implement it in OpenCV? It's very simple, divided into three steps

（1）. Grayscale image：gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

（2）. Gaussian filtering (denoising) image：GaussianBlur（src，ksize，sigmaX [，dst [，sigmaY [，borderType]]]）-> dst

Parameter meanings：

src：The input image is usually a grayscale image

ksize：Gaussian kernel size

sigmaX： Gaussian kernel standard deviation in the X direction

sigmaY： Gaussian kernel standard deviation in the Y direction

dst：Processed image

（3）. Canny method for processing images：edges=cv2.Canny(image, threshold1, threshold2[, apertureSize[, L2gradient]])

Parameter meanings：

edges：Calculated edge image

image： Calculated edge image，Generally, the image obtained after Gaussian processing

threshold1： The first threshold during the processing

threshold2： Second threshold during processing

apertureSize：Sobel The aperture size of the operator

L2gradient：The default value for identifying the gradient magnitude of an image is False. If true, use a more accurate L2 norm for calculation (i.e. the sum of squared derivatives in both directions and then the root), otherwise use the L1 norm (directly adding the absolute values of derivatives in both directions).

3.4、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_3.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    imgG = cv2.GaussianBlur(gray,(3,3),0)
    dst = cv2.Canny(imgG,50,50)
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('gray', gray)
        cv2.imshow("canny",dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

After running the program, the effect image,

4、OpenCV line segment drawing

4.1、When using OpenCV to process images, we sometimes need to draw line segments, rectangles, etc. on the image. Used in OpenCV

cv2.line（dst，pt1，pt2，color，thickness=None，lineType=None，shift=None）Draw line segments using functions.

Parameter meanings：

dst：Output Image。

pt1，pt2：Required parameter. The coordinate points of a line segment represent the starting and ending points, respectively

color：Required parameter. Used to set the color of line segments

thickness：Optional parameters. Used to set the width of line segments

lineType：Optional parameters. Used to set the type of line segment, optional 8 (8 adjacency connectors - default), 4 (4 adjacency connectors), and cv2. LINE.AA for anti aliasing

4.2、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_4.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    line = cv2.line(img, (50,20), (20,100), (255,0,255), 10)
    while True :
        cv2.imshow("line",line)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

After running the program, the effect image，

5、OpenCV drawing rectangles

5.1、In OpenCV, drawing rectangles is done using

cv2.rectangle（img，pt1，pt2，color，thickness=None，lineType=None，shift=None）

Parameter meanings：

img：Canvas or carrier images

pt1，pt2：Required parameter. The vertices of a rectangle represent the vertices and diagonal vertices respectively, namely the upper left corner and lower right corner of the rectangle (these two vertices can determine a unique rectangle), which can be understood as diagonals.

color：Required parameter. Used to set the color of the rectangle

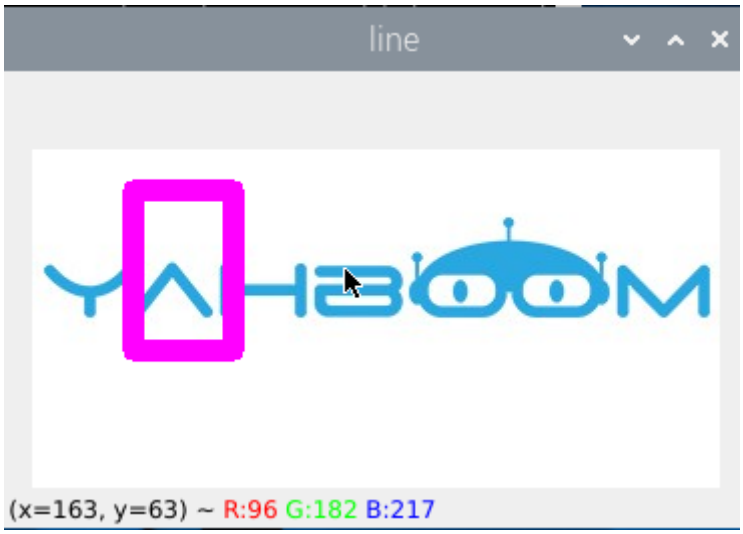
thickness：Optional parameters. Used to set the width of the rectangular edge. When the value is negative, it indicates filling the rectangle

lineType：Optional parameters. Used to set the type of line segment, optional 8 (8 adjacency connectors - default), 4 (4 adjacency connectors), and cv2. LINE.AA for anti aliasing

5.2、Code and Effect Display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_5.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    rect = cv2.rectangle(img, (50,20), (100,100), (255,0,255), 10)
    while True :
        cv2.imshow("line",rect)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



6、OpenCV drawing circles

6.1、In OpenCV, drawing a circle is done using

cv2.circle(img, center, radius, color[,thickness[,lineType]])

Parameter meanings：

img:Painting or carrier image cloth

center：For center coordinates, format： (50,50)

radius：radius

thickness：The thickness of the lines. Default is 1. If -1, it is filled with solid

lineType：Line type. The default is 8, connection type. The following table explains

parameter	explain
cv2.FILLED	filling
cv2.LINE_4	4 Connection types
cv2.LINE_8	8 Connection types
cv2.LINE_AA	Anti aliasing, this parameter will make the lines smoother

6.2、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_6.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    circle = cv2.circle(img, (80,80), 50, (255,0,255), 10)
    while True :
        cv2.imshow("circle",circle)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

7、OpenCV drawing ellipses

7.1、In OpenCV, drawing ellipses is done using

cv2.ellipse(img, center, axes, angle, StartAngle, endAngle, color[,thickness[,lineType]

Parameter meanings：

center：The center point of an ellipse， （x，y）

axes：Refers to short radius and long radius， （x，y）

StartAngle：The angle of the starting angle of the arc

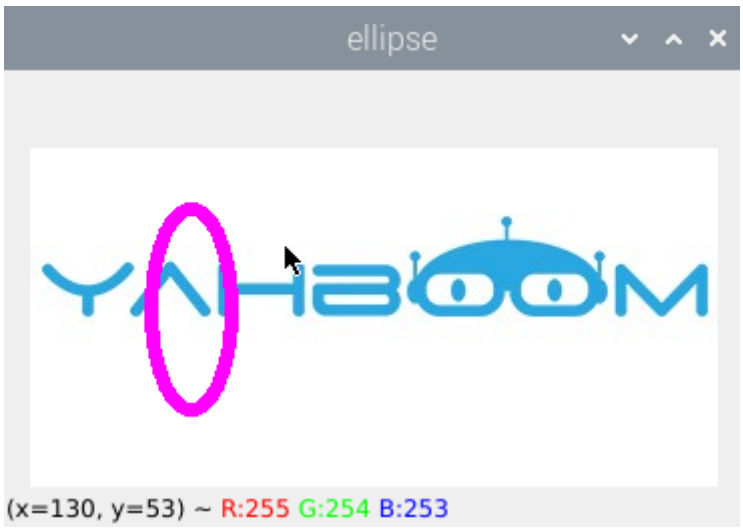
endAngle：The angle of the ending angle of the arc

img、color、thickness、lineType You can refer to the explanation of the circle

7.2、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_7.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    ellipse = cv2.ellipse(img, (80,80), (20,50),0,0, 360,(255,0,255), 5)
    while True :
        cv2.imshow("ellipse",ellipse)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



8、OpenCV drawing polygons

8.1、In OpenCV, polygons are drawn using

cv2.polylines(img,[pts],isClosed, color[,thickness[,lineType]])

Parameter meanings：

pts：The vertices of a polygon

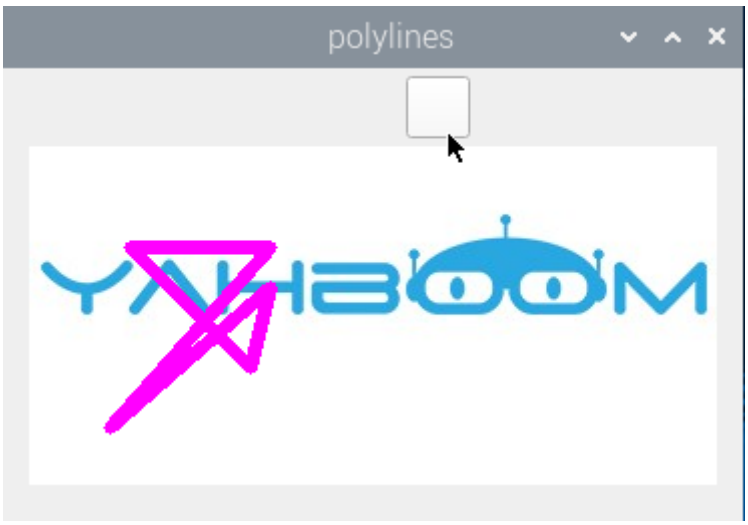
isClosed：Is it closed。 (True/False)

Other parameters refer to the drawing parameters of the circle

8.2、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_8.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    points = np.array([[120,50], [40,140], [120,70], [110,110], [50,50]],
np.int32)
    polylines = cv2.polylines(img, [points],True,(255,0,255), 5)
    while True :
        cv2.imshow("polylines",polylines)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



9、OpenCV drawing text

9.1、In OpenCV, drawing text is done using

cv2.putText(img, str, origin, font, size,color,thickness)

Parameter meanings：

img：input image

str：Text drawn

origin：The upper left corner coordinate (integer) can be understood as where the text starts

font：font

size：font size

color：font color

thickness：font thickness

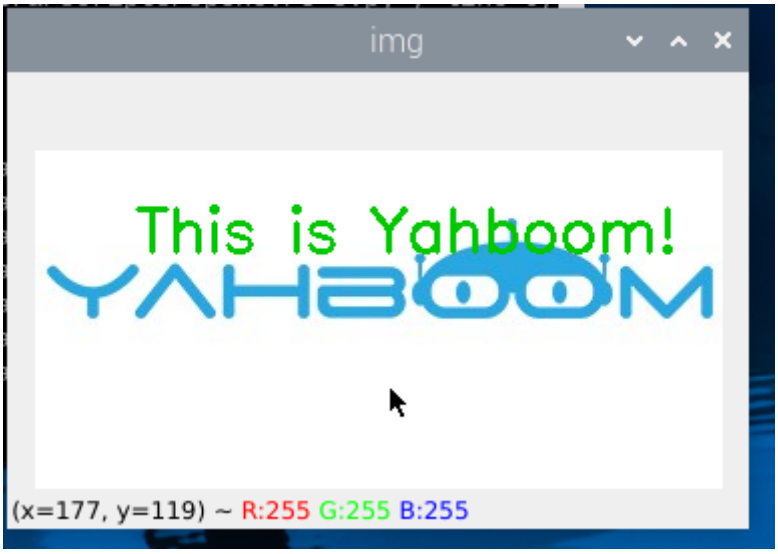
Among them, the font is optional

FONT_HERSHEY_SIMPLEX Python：cv.FONT_HERSHEY_SIMPLEX	正常大小sans-serif字体
FONT_HERSHEY_PLAIN Python：cv.FONT_HERSHEY_PLAIN	小尺寸sans-serif字体
FONT_HERSHEY_DUPLEX Python：cv.FONT_HERSHEY_DUPLEX	正常大小的sans-serif字体（比FONT_HERSHEY_SIMPLEX更复杂）
FONT_HERSHEY_COMPLEX Python：cv.FONT_HERSHEY_COMPLEX	正常大小的衬线字体
FONT_HERSHEY_TRIPLEX Python：cv.FONT_HERSHEY_TRIPLEX	正常大小的serif字体（比FONT_HERSHEY_COMPLEX更复杂）
FONT_HERSHEY_COMPLEX_SMALL Python：cv.FONT_HERSHEY_COMPLEX_SMALL	较小版本的FONT_HERSHEY_COMPLEX
FONT_HERSHEY_SCRIPT_SIMPLEX Python：cv.FONT_HERSHEY_SCRIPT_SIMPLEX	手写风格的字体
FONT_HERSHEY_SCRIPT_COMPLEX Python：cv.FONT_HERSHEY_SCRIPT_COMPLEX	更复杂的FONT_HERSHEY_SCRIPT_SIMPLEX变体
FONT_ITALIC Python：cv.FONT_ITALIC	标志为斜体字体 YahBoom

9.2、Code and actual effect display

```
cd ~/yahboomcar_ws/src/yahboom_esp32ai_car/scripts/opencv/
python3 3_9.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    cv2.putText(img,'This is Yahboom!',(50,50),cv2.FONT_HERSHEY_SIMPLEX,1,
(0,200,0),2)
    while True :
        cv2.imshow("img",img)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```


img
This is Yahboom!
YAHBOOM
(x=177, y=119) ~ R:255 G:255 B:255