1. opencv basic course

1.1. Image reading and presentation

1.1.1 Image read in

img = cv2.imread('yahboom.jpg', 0) The first parameter is the path of the image, the second parameter is how to read this image.

cv2.IMREAD_UNCHANGED: keep the original format unchanged, -1;

cv2.IMREAD_GRAYSCALE: read the picture in grayscale mode, can be represented by 0;

cv2.IMREAD_COLOR:, read in a color picture, can be expressed as 1;

cv2.IMREAD_UNCHANGED: read in a picture and include its alpha channel, can be represented by 2.

1.1.2 Image show

cv.imshow('frame', frame): open a window named frame and display frame frame data (image/video data)

Parameter meanings:

The first parameter indicates the name of the window to create the open window;

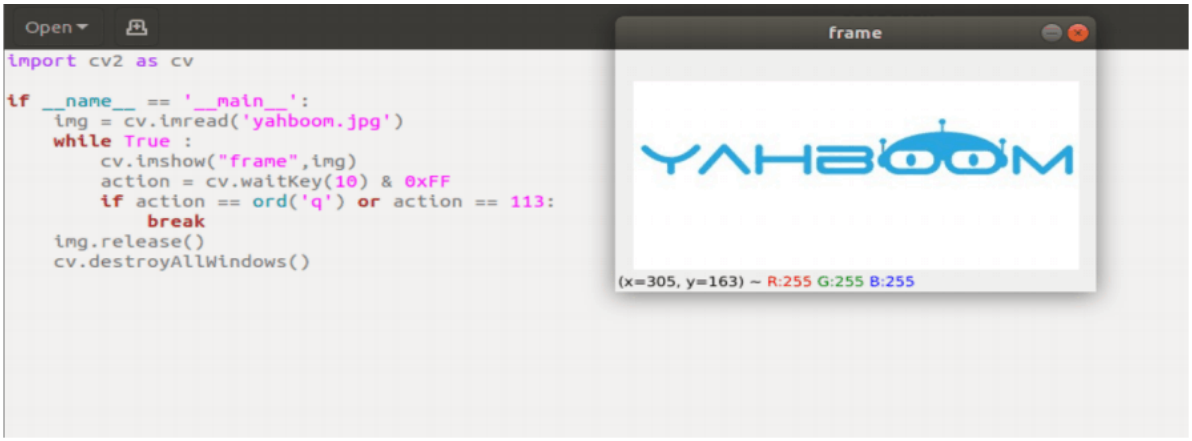
The second parameter indicates the image to be displayed.

1.1.3 Code and actual effect demonstration

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 1_1.py
```

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



1.2 opencv image write

1.2.1 Function method: cv2.imwrite('new_img_name', img)

Parameter meaning:

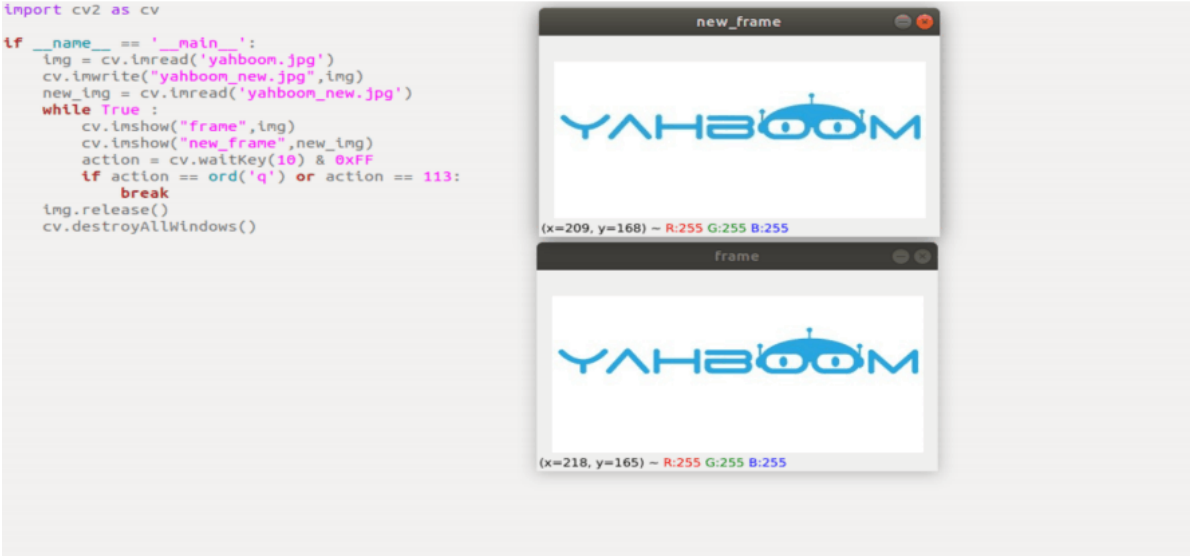
The first parameter is the saved file name, the second parameter is the saved image.

1.2.2 Code and actual effect display

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 1_2.py
```

```
import cv2 as cv
if __name__ == '__main__':
    img = cv.imread('yahboom.jpg')
    cv.imwrite("yahboom_new.jpg",img) #新建文件yahboom_new.jpg，并且把yahboom.jpg写进去
    # Create a new file yahboom_new.jpg and write yahboom.jpg to it.
    new_img = cv.imread('yahboom_new.jpg') #读取新写入的图片 #Read newly written images
    while True :
        cv.imshow("frame",img)
        cv.imshow("new_frame",new_img)
    action = cv.waitKey(10) & 0xFF
    if action == ord('q') or action == 113:
        break
    img.release()
    cv.destroyAllWindows()
```



1.3 opencv camera reading and displaying video

1.3.1 camera reading

capture=cv.VideoCapture(0)

Parameter meaning:

VideoCapture() in the parameter is 0, said to open the laptop's built-in camera, the parameter is the video file path to open the video, such as cap =

cv2.VideoCapture("... /test.avi")

1.3.2 Display the webcam video

ret,img = frame.read()

Return value meaning:

ret: ret is a bool value that determines whether the correct frame is read back or not

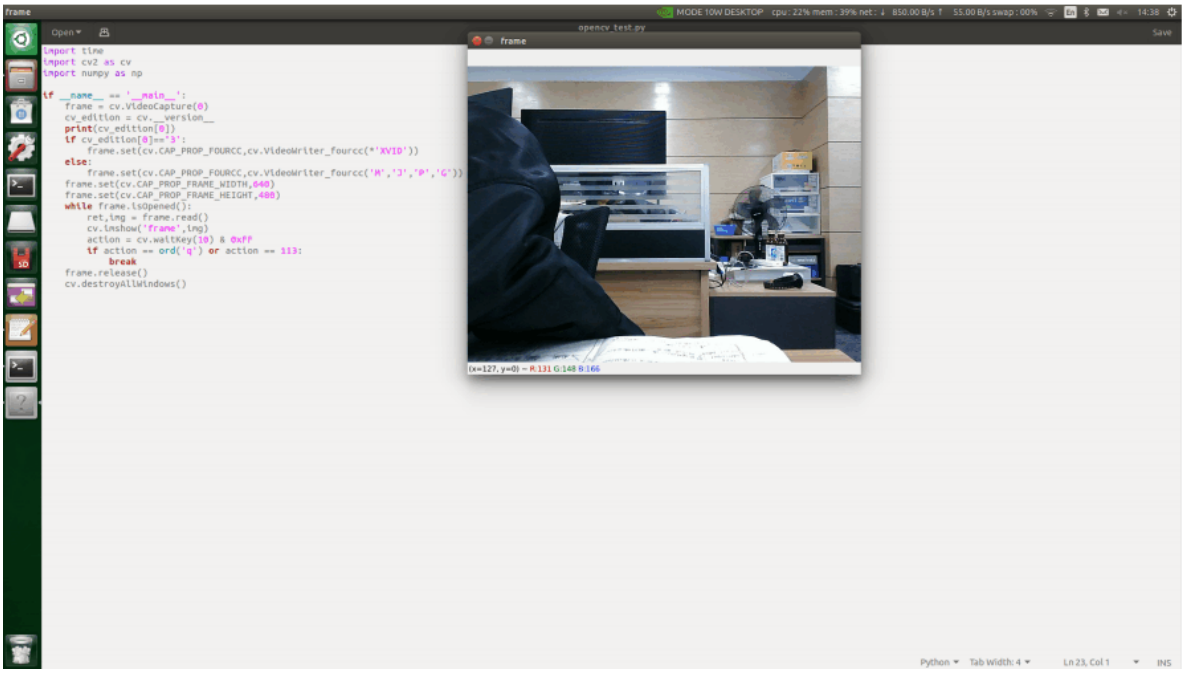
img: image data of each frame

1.3.3 Code and actual effect display

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 1_3.py
```

```
import cv2 as cv
if __name__ == '__main__':
    frame = cv.VideoCapture(0)
    while frame.isOpened():
        ret,img = frame.read()
        cv.imshow('frame',img)
        action = cv.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
        break
    frame.release()
    cv.destroyAllWindows()
```



1.4. openc pixel operations

1.4.1 Pixel manipulation, we can change a new pixel color for any position.

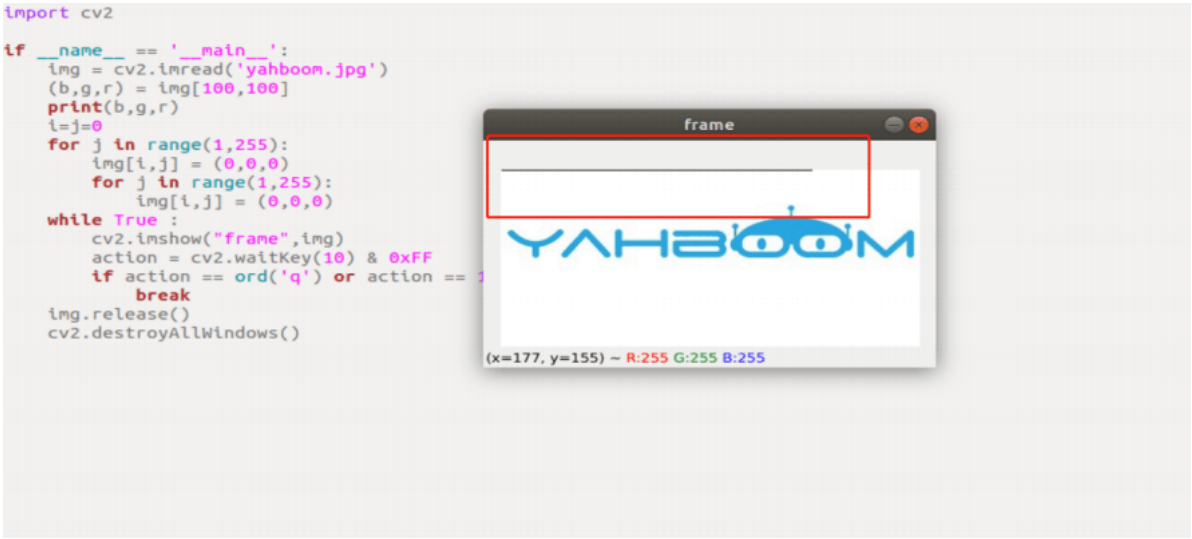
First, we have to read the image and then modify the value of bgr to assign a region to be black.

1.4.2 Code and real effect demonstration*

Run the program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 1_4.py
```

```
import cv2
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    (b,g,r) = img[100,100]
    print(b,g,r)
    i=j=0
    for j in range(1,255):
    img[i,j] = (0,0,0)
    for j in range(1,255):
    img[i,j] = (0,0,0)
    while True :
        cv2.imshow("frame",img)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



The red box part is the modified pigment value.

2.1 opencv image scaling

2.1.1 cv2.resize(InputArray src,OutputArray dst, Size, fx, fy, interpolation)

Parameter meaning:

InputArray src: input image

OutputArray ds: the output image.

Size: size of the output image

fx,fy: scaling factor along x-axis and y-axis.

interpolation: insertion mode, can choose INTER_NEAREST (nearest neighbor interpolation), INTER_LINEAR (bilinear interpolation (default setting)), INTER_LINEAR (bilinear interpolation (default setting)), INTER_INTER_INTERPOLATION (default setting)

(default setting)), INTER_AREA (resampling using pixel-area relations), INTER_CUBIC (bicubic interpolation of 4x4 pixel neighborhoods), INTER_LANCY (bicubic interpolation of 4x4 pixel neighborhoods), and

(bicubic interpolation for 4x4 pixel neighborhoods), INTER_LANCZOS4 (Lanczos interpolation for 8x8 pixel neighborhoods).

Note that the output size format is (width, height):

- The output size format is (width, height)
- The default interpolation method is: bilinear interpolation.

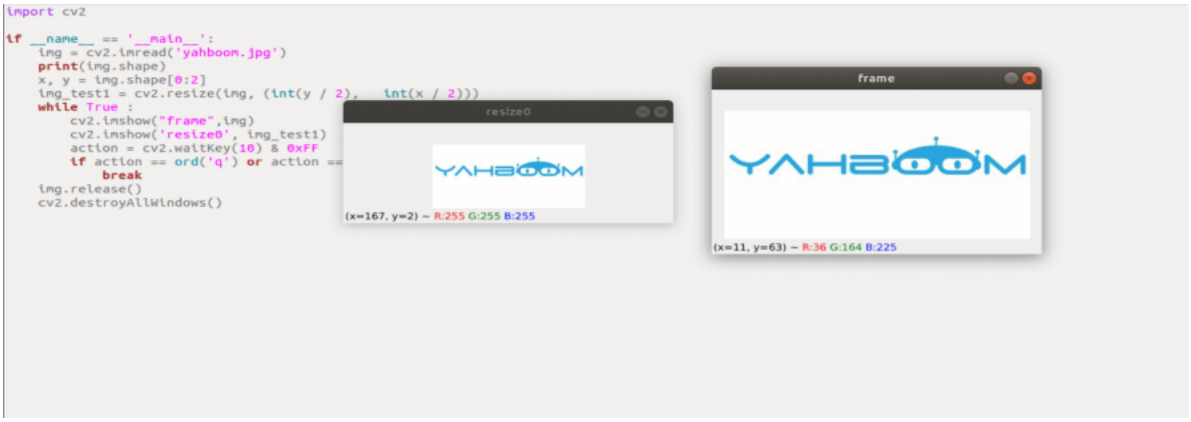
2.1.2 Code and actual effect demonstration

Run the program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 2_1.py
```

```
import cv2
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    print(img.shape)
```

```
    x, y = img.shape[0:2]
    img_test1 = cv2.resize(img, (int(y / 2), int(x / 2)))
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('resize0', img_test1)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



2.2 opencv image clipping

2.2.1 Image clipping

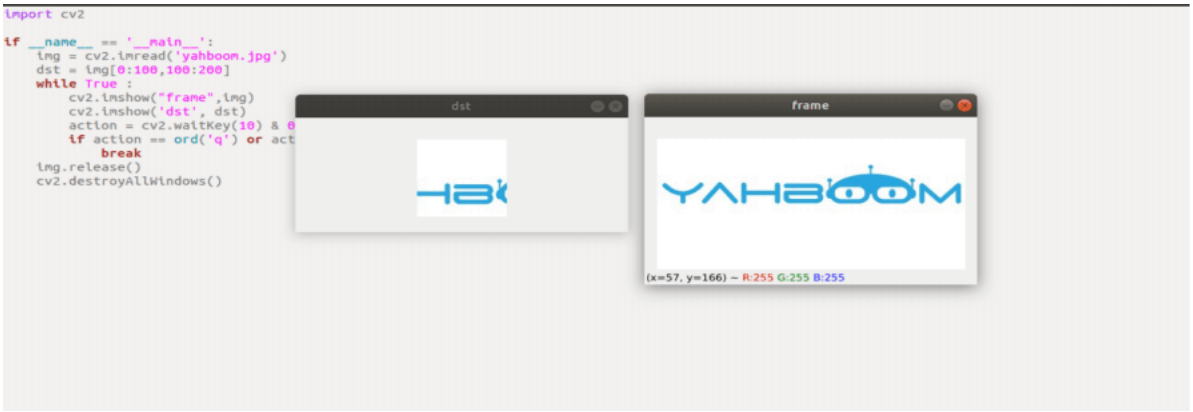
First read the image, and then get the pixel point region in the array. The following code to select the shape of the region X: 300-500 Y: 500-700, pay attention to the image size is 800 * 800, so select the region do not exceed this resolution.

2.2.2 Code and actual effect display

Run the program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 2_2.py
```

```
import cv2
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    dst = img[0:100,100:200]
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('dst', dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
        img.release()
        cv2.destroyAllWindows()
```



2.3 opencv image panning

2.3.1 cv2.warpAffine(src, M, dsize[,dst[, flags[, borderMode[, borderValue]]]])

Parameter meanings:

- src: input image
- M: transform matrix
- dsize: size of the output image
- flags: combination of interpolation methods (int type!)
- borderMode: border pixel mode (int type!)
- borderValue: (emphasis added!) border fill value; by default, it is 0

In the above parameters: M as the affine transformation matrix, generally reflecting the relationship between translation or rotation, is a 2×3 transformation matrix of type InputArray.

In the daily affine transformation, only set the first three parameters, such as cv2.warpAffine(img,M,(rows,cols)) can realize the

basic affine transformations.

2.3.2 How to get the conversion matrix M? The following example illustrates.

Through the conversion matrix M to realize the original image src to the target image dst:

dst(x, y) = src(M11x + M12y+M13, M21x+M22y+M23)

Shifting the original image src 200 and 100 pixels to the right and 100 pixels downward, the correspondence is:

dst(x, y) = src(x+200, y+100)

Complete the above expression, i.e:

dst(x, y) = src(1-x + 0-y + 200, 0-x + 1-y + 100)

Based on the above expression, the values of the individual elements of the corresponding transformation matrix M can be determined as:

M11=1

M12=0

M13=200

M21=0

M22=1

M23=100

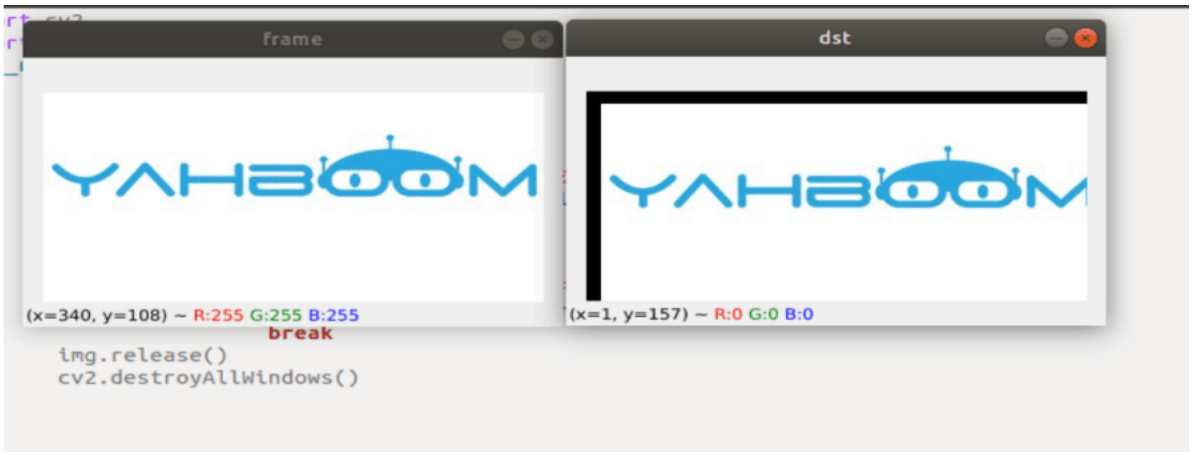
Substituting the above values into the transformation matrix M, we get:

M = [ ]

2.3.3 Code and practical effect demonstration

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 2_3.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    matShift = np.float32([[1,0,10],[0,1,10]])# 2*3
    dst = cv2.warpAffine(img, matShift, (width,height))
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('dst', dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



2.4 opencv image mirroring

2.4.1 Principle of image mirroring

There are two types of image mirroring transformations: horizontal mirroring and vertical mirroring. Horizontal mirroring takes the vertical center line of the image as the axis, and swaps the pixels of the image, that is, the left half of the image and the right half of the image. Vertical mirroring, on the other hand, uses the horizontal centerline of the image as an axis to swap the top half of the image with the bottom half.

Principle of transformation:

Let the width of the image be width and the length be height.(x,y) are the coordinates of the transformed image and (x0,y0) are the coordinates of the original image.

- Horizontal Mirror Transform

  Forward mapping: x=width-x0-1,y=y0

  Backward mapping: x0=width-x-1,y0=y

- Vertical Mirror Transform

  Upward mapping: x=x0,y=height-y0-1

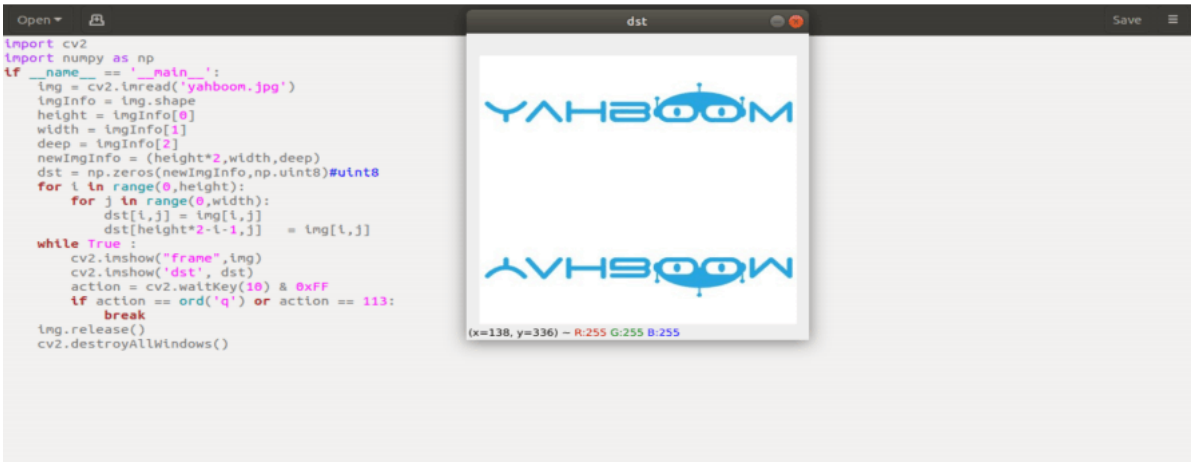
  Downward mapping: x0=x, y0=height-y-1

To summarize: In the horizontal mirror transform, the entire image is traversed, and then each pixel is processed according to the mapping relationship. In fact, the horizontal mirroring transform is the image coordinates of the columns to the right, the right side of the columns to the left, it is possible to do the transformation in columns. The same is true for the vertical mirroring transform, which can be done on a column-by-column basis.

2.4.2 See how Python implements the Vertical Transform as an example.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 2_4.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    deep = imgInfo[2]
    newImgInfo = (height*2,width,deep)
    dst = np.zeros(newImgInfo,np.uint8)#uint8
    for i in range(0,height):
        for j in range(0,width):
            dst[i,j] = img[i,j]
            dst[height*2-i-1,j] = img[i,j]
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('dst', dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



3.1 opencv image grayscale processing

3.1.1 Image Grayscaling

The process of converting a color image into a grayscale image is the grayscaling of the image. The color of each pixel in a color image is determined by the three components R, G, B, and each component can take the value 0-255, so that a pixel point can have more than 16 million (256 * 256 * 256 = 1677256) of the color of the range of changes. The grayscale image is a special color image with the same R, G, and B components, in which a pixel point can have a range of 256 variations. Therefore, in digital image processing, various formats are generally converted to grayscale images in order to make the calculation of the subsequent image less. The description of a grayscale image still reflects the distribution and characteristics of the overall and local chromaticity and highlight levels of the whole image as in the case of a color image.

3.1.2 Image Grayscale Processing

Grayscale processing is the process of converting a color image into a grayscale image. Color image is divided into R, G, B three components, respectively, showing a variety of colors such as red, green, blue, etc. Grayscaling is the process of making the R, G, B components of color equal. Pixels with large grayscale values are brighter (pixels with a maximum value of 255 are white) and vice versa are darker (pixels with a minimum value of 0 are black). The core idea of image grayscaling is R = G = B , this value is also called the gray value.

1) Maximum method: so that the converted R, G, B is equal to the largest of the three values before the conversion, that is: R = G = B = max (R, G, B). This method of conversion of gray-scale map brightness is very high.

2) Average method: the value of R, G, B after conversion is the average value of R, G, B before conversion. That is: R=G=B=(R+G+B)/3. This method produces a softer grayscale image.

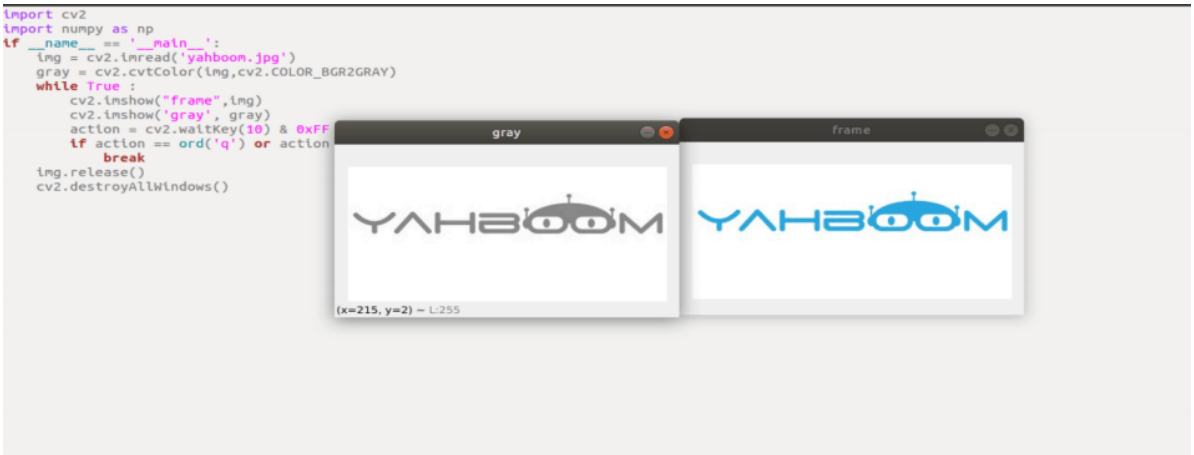
In OpenCV, cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY) is used to implement grayscaling of the image

3.1.3 Code and actual effect demonstration

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```



3.2 opencv image binarization processing

3.2.1 The core idea of binarization

Set a threshold, greater than the threshold for 0 (black) or 255 (white), so that the image is called a black and white map. The threshold can be fixed or adaptive threshold. Adaptive thresholding is generally a comparison of a pixel at a point with the mean of the pixels in the region where that point is the median or a weighted sum of Gaussian distributions, where a difference may or may not be set.

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

Parameter meaning:

- src: original image
- threshold: current threshold
- maxVal: the maximum threshold value, usually 255.
- thresholdType: the type of threshold, generally has the following values.
    - THRESH_BINARY = 0, # the gray value of pixels larger than the threshold is set to maxValue (e.g., 8-bit gray value max 255), and the gray value of pixels with gray value smaller than the threshold is set to 0.
    - THRESH_BINARY_INV = 1, # The gray value of pixel points larger than the threshold is set to 0, and those smaller than that threshold are set to maxValue.
    - THRESH_TRUNC = 2, #The gray value of pixel points larger than the threshold is set to 0, while those smaller than the threshold are set to maxValue.
    - THRESH_TOZERO = 3, #Pixel points with a grayscale value less than the threshold are not changed in any way, while those greater than the threshold have their grayscale

values all changed to 0.
    - THRESH_TOZERO_INV = 4 # Pixel points whose grayscale value is greater than this threshold are not changed in any way, and pixel points whose grayscale value is less than this threshold have all their grayscale values changed to 0.
- Return Value:
    - retval: consistent with parameter thresh 3
    - dst: result image

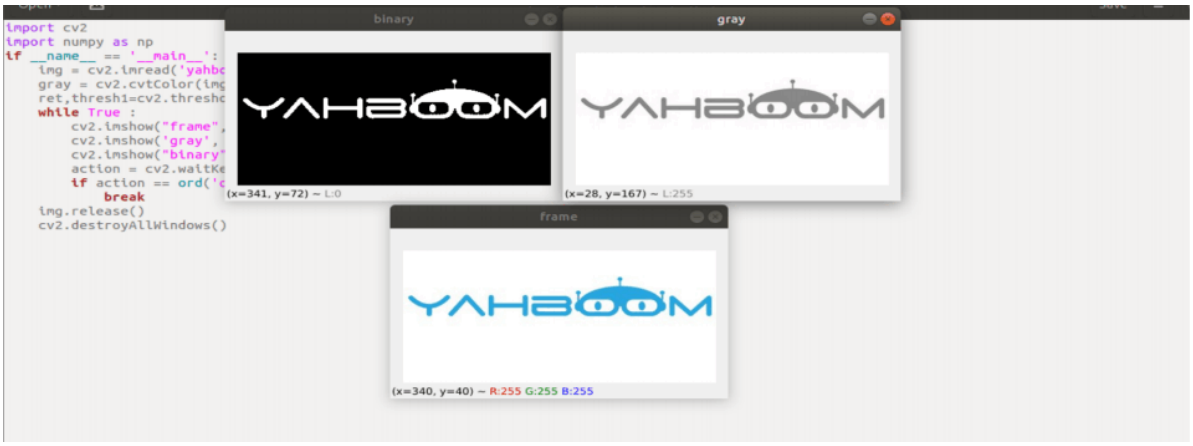
Note: Before binarization, we need to grayscale the color image to get a grayscale image.

3.2.3 Code and actual effect demonstration

Run the program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```



3.3 opencv image edge detection

3.3.1 Purpose of Image Edge Detection

Significantly reduce the data size of the image while preserving the original image properties. There are a variety of algorithms for edge detection, and although the Canny algorithm is old, it can be said that it is a standard algorithm for edge detection and is still widely used in research.

3.3.2 Canny Edge Detection Algorithm

Among the currently used edge detection methods, Canny edge detection algorithm is one of the methods with a strict definition that can provide good and reliable detection. It has become one of the most popular algorithms for edge detection due to its advantages of satisfying the three criteria for edge detection and simplicity of implementation process.

Canny edge detection algorithm can be categorized into following 5 steps:

- Using Gaussian filter to smooth the image and filter out the noise
- Calculate the gradient strength and direction for each pixel in the image.
- Apply Non-Maximum Suppression to remove spurious response from edge detection.
- Apply Double-Threshold detection to identify real and potential edges.
- Finalize edge detection by suppressing isolated weak edges.

3.3.3 opencv Implementation Steps

- Image grayscale: gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
- Gaussian filtering (noise reduction processing) image: GaussianBlur (src, ksize, sigmaX [, dst [, sigmaY [, borderType]]]) -> dst
    - src: the input image, usually after grayscale
    - ksize : Gaussian kernel size
    - sigmaX : standard deviation of the Gaussian kernel in X direction
    - sigmaY : the standard deviation of the Gaussian kernel in Y direction
    - dst: the processed image
- Canny method to get the image: edges=cv2.Canny(image, threshold1, threshold2[, apertureSize[,L2gradient]])
    - edges: computed edge images
    - image: the calculated edge image, generally the image obtained after Gaussian processing
    - threshold1 : the first threshold in the process
    - threshold2: the second threshold in the process.
    - apertureSize: the size of the aperture of the Sobel operator.
    - L2gradient: the flag for calculating the gradient magnitude of the image, the default value is False, if it is True, the more accurate L2 norm will be used (i.e., the square of the derivatives of the two directions are squared and re-squared), otherwise, the L1 norm will be used (the absolute value of the derivatives of the two directions are directly added together).

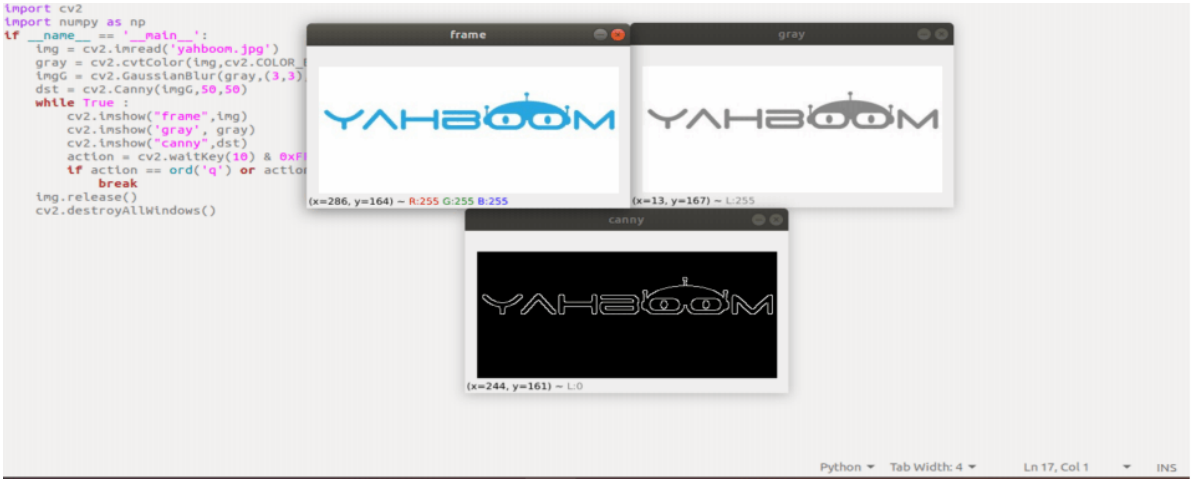
3.3.4 Demonstration of code and practical effects

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```



3.4 opencv line drawing

3.4.1 cv2.line (dst, pt1, pt2, color, thickness=None, lineType=None, shift=None)

Parameter meaning:

- dst: output image
- pt1, pt2: mandatory parameters. The coordinate points of the line segment, indicating the start point and the end point respectively
- color: mandatory parameter. Used to set the color of the line segment.
- thickness: optional parameter. Used to set the width of the line.
- lineType: optional parameter. Used to set the type of the line, can choose 8 (8 neighboring lines - default), 4 (4 neighboring lines) and cv2.LINE_AA for anti-aliasing.

3.4.2 Code and actual effect display

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```

3.5 opencv draw rectangle

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

Parameter meaning:

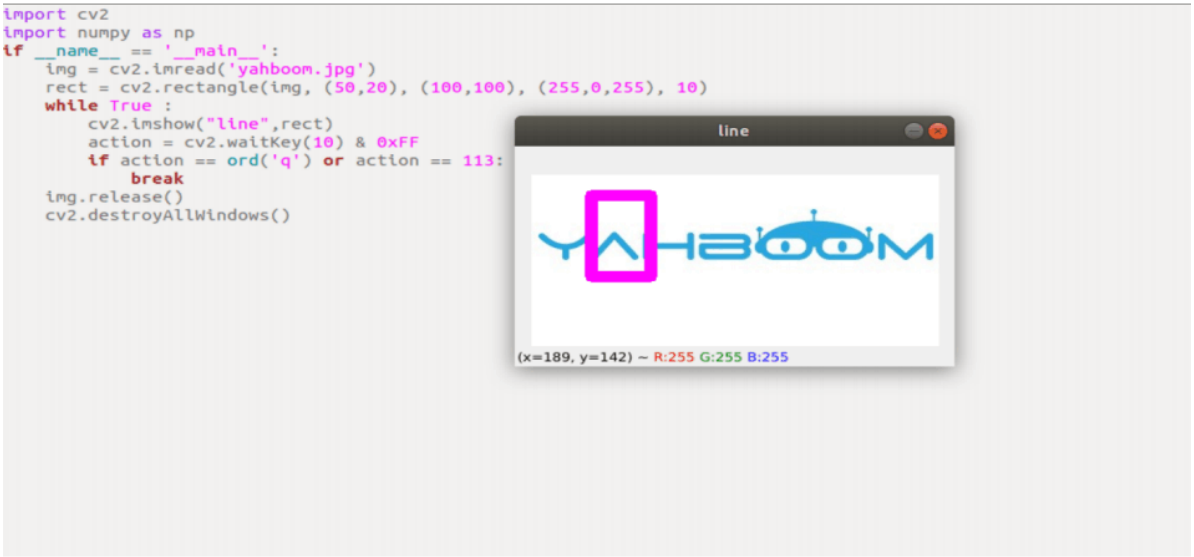
- img: canvas or carrier image
- pt1, pt2: mandatory parameters. The vertices of the rectangle, which represent the vertex and diagonal vertices, i.e., the upper left and lower right corners of the rectangle (these two vertices can determine a unique rectangle), which can be interpreted as diagonal lines.
- color: mandatory. Use to set the color of the rectangle.
- thickness: optional parameter. Used to set the width of the sides of the rectangle, when the value is negative, it means that the rectangle is filled.
- lineType: optional parameter. Used to set the type of the line, can be 8 (8 neighbor connecting lines - default), 4 (4 neighbor connecting lines) cv2.LINE_AA for anti-aliasing

3.5.2 Code and effect display

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```



3.6 opencv drawing circles

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

Parameter meaning:

- img: draw or carrier image cloth
- center: the center of the circle, format: (50,50)
- radius: radius
- thickness: thickness of the line. Default is 1. If -1 then it is a filled solid center
- lineType: line type. Default is 8, the connection type. The following table describes

parameter	Instructions
cv2.FILLED	Fill in
cv2.LINE_4	4 Connection Types
cv2.LINE_8	8 Connection Types
cv2.LINE_AA	Anti-aliasing, the three uncles will make the lines smoother

3.6.2. Code and actual effect display

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```

3.7 opencv drawing ellipse

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

Parameter meaning:

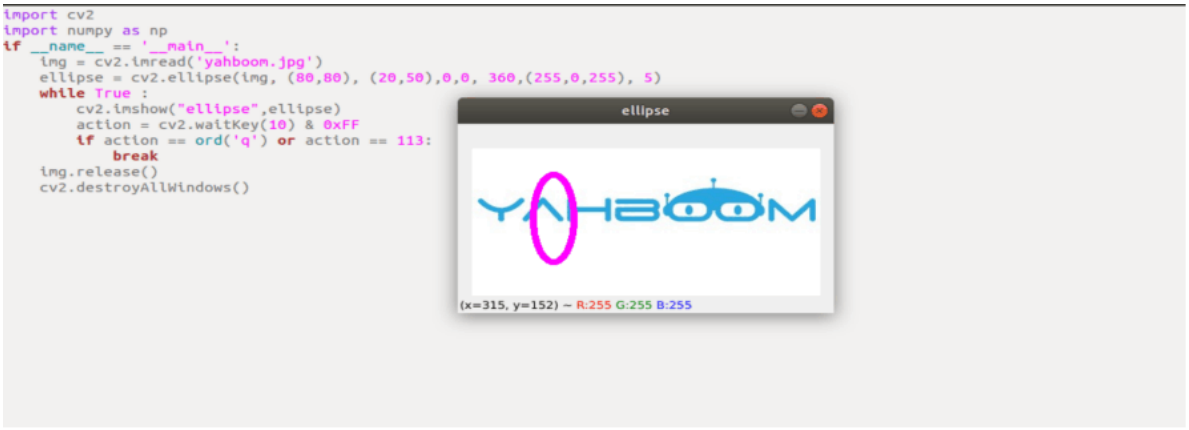
- center: the center of the ellipse, (x, y)
- axes: short and long radii, (x, y)
- StartAngle: the angle of the start angle of the arc
- endAngle: the angle of the end angle of the arc.
- img, color, thickness, lineType can refer to the description of circle

3.7.2 Code and actual effect display

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```



3.8 opencv drawing polygons

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

Parameter meanings:

- pts: vertices of the polygon
- isClosed: whether it is closed. (True/False)
- Other parameters refer to circle drawing parameters

3.8.2 Code and actual effect demonstration

Run the program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```

3.9, opencv draw text

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

Parameter meaning:

- img: input image
- str: the text drawn
- origin: the coordinates of the upper left corner (integer), can be interpreted as the text from where to start
- font: font
- size: font size
- color: font color
- thickness: font thickness
- where the font is optional.

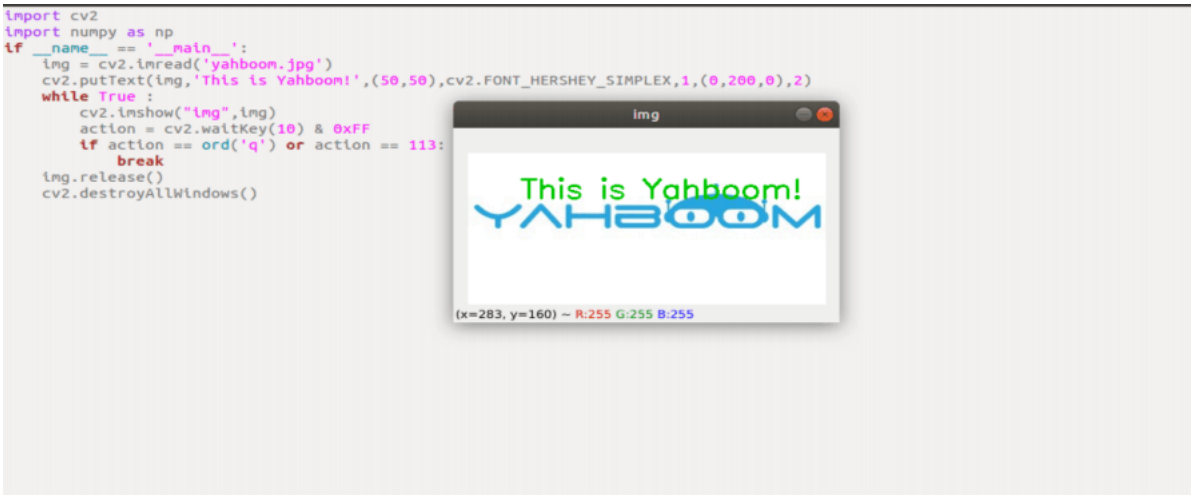
FONT_HERSHEY_SIMPLEX Python: cv.FONT_HERSHEY_SIMPLEX	正常大小sans-serif字体
FONT_HERSHEY_PLAIN Python: cv.FONT_HERSHEY_PLAIN	小尺寸sans-serif字体
FONT_HERSHEY_DUPLEX Python: cv.FONT_HERSHEY_DUPLEX	正常大小的sans-serif字体（比FONT_HERSHEY_SIMPLEX更复杂）
FONT_HERSHEY_COMPLEX Python: cv.FONT_HERSHEY_COMPLEX	正常大小的衬线字体
FONT_HERSHEY_TRIPLEX 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

3.9.2 Code and actual effect display

Running program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/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()
```



4.1 opencv image restoration

Image restoration is a class of algorithms in computer vision whose goal is to fill a region within an image or video. The region is identified using a binary mask, and the filling is usually done based on information about the boundaries of the region to be filled. The most common application of image restoration is to recover old scanned photographs. It is also used to remove small unwanted objects from images.

4.1.1, dst = cv2.inpaint(src, inpaintMask, inpaintRadius, flags)

Parameter meaning:

- src: the source image, that is, the image to be repaired
- inpaintMask: binary mask indicating the pixels to be repaired.
- dst: the result image
- inpaintRadius: the radius of the repair.
- flags : repair algorithms, mainly INPAINT_NS (Navier-Stokes based method) or INPAINT_TELEA (Fastmarching based method).

The Navier-Stokes based repair should be slower and tend to produce more ambiguous results than the fast marching method approach. In practice, we did not find this to be the case.INPAINT_NS produced better results in our tests and was slightly faster than INPAINT_TELEA.

4.1.2 Code and Practical Results Showing

(1), first of all, we first add breakage to the intact picture according to its intactness, which can be understood as modifying the pixel value of its specific part.

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 4_1_1.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    for i in range(50,100):
        img[i,50] = (0,0,0)
        img[i,50+1] = (0,0,0)
        img[i,50-1] = (0,0,0)
    for i in range(100,150):
        img[150,i] = (0,0,0)
        img[150,i+1] = (0,0,0)
        img[150-1,i] = (0,0,0)
    cv2.imwrite("damaged.jpg",img)
    dam_img = cv2.imread('damaged.jpg')
    while True :
        cv2.imshow("dam_img",dam_img)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

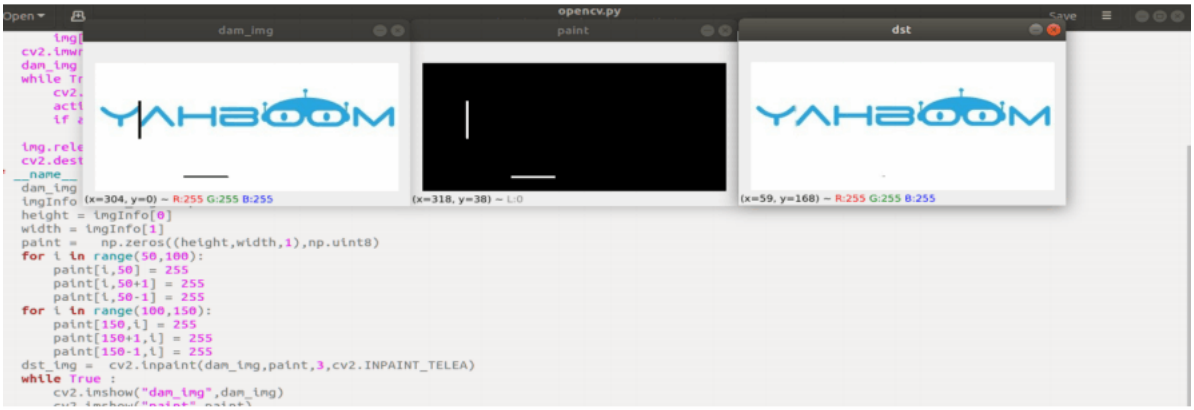
Running it generates an image which is seen as a broken image of the original image, the

(2), repair the photo just created, first read, then create a mask, and finally use the function to repair it

Run the program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 4_1_2.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    dam_img = cv2.imread('damaged.jpg')
    imgInfo = dam_img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    paint = np.zeros((height,width,1),np.uint8)
    for i in range(50,100):
        paint[i,50] = 255
        paint[i,50+1] = 255
        paint[i,50-1] = 255
    for i in range(100,150):
        paint[150,i] = 255
        paint[150+1,i] = 255
        paint[150-1,i] = 255
    dst_img = cv2.inpaint(dam_img,paint,3,cv2.INPAINT_TELEA)
    while True :
        cv2.imshow("dam_img",dam_img)
        cv2.imshow("paint",paint)
        cv2.imshow("dst",dst_img)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



As shown in the figure, the left side is before the repair, the center is the mask image, and the right side is the original image after the repair.

4.2. opencv image brightness enhancement

Implementation process: synchronized amplification of the three channel values of each pixel point, while maintaining the channel value between 0-255, in fact, it is traversing each pixel point, adding or subtracting values to them, and then determine whether the three channel rgb is in the range of 0-255, greater than or less than the value of 255 or 0.

4.2.1 Code and actual effect display

Run the program.

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 4_2.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    dst = np.zeros((height,width,3),np.uint8)
    for i in range(0,height):
        for j in range(0,width):
            (b,g,r) = img[i,j]
            bb = int(b) + 100
            gg = int(g) + 100
            rr = int(r) + 100
            if bb > 255:
                bb = 255
            if gg > 255:
                gg = 255
            if rr > 255:
                rr = 255
            dst[i,j] = (bb,gg,rr)
    while True :
        cv2.imshow("dst",dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

The picture on the left is the original picture, and the picture on the back is the picture after increasing the brightness.

4.3. opencv image skinning and whitening

OpenCV to achieve the function of the picture skin whitening, the realization of the principle and "1.20 OpenCV picture brightness enhancement" principle is basically the same, except that here we do not need to do the processing of the r-value, only according to the formula, p = p(x)*1.4 + y, which p(x) that the b-channel or g-channel, y represents the need to increase or decrease the value, the same, after adding the value of the picture, the picture is the same. or g channel, and y

denotes the value that needs to be added or subtracted. Again, after adding the value, we need to make a judgment about the value.

4.3.1 Code and actual effect display

Run the program

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/
python3 4_3.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    dst = np.zeros((height,width,3),np.uint8)
    for i in range(0,height):
        for j in range(0,width):
            (b,g,r) = img[i,j]
            bb = int(b*1.4) + 5
            gg = int(g*1.4) + 5
            if bb > 255:
                bb = 255
            if gg > 255:
                gg = 255
            dst[i,j] = (bb,gg,r)
    while True :
        cv2.imshow("origin",img)
        cv2.imshow("dst",dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```