4.2 OpenCV

4.2.1 About OpenCV

The name of OpenCV is Open Source Computer Vision Library, an open source computer vision library. As shown in the above figure, we see the OpenCV logo, which can be seen by three small rings of distinct R, G, and B primary colors. That is, it is a set of API libraries for open source code for computer vision.

This means that:

(1) Whether it is scientific research or commercial application, it can be used for development;

(2) The source code of all API functions is public, you can see the program steps of its internal implementation;

(3) You can modify the source code of OpenCV and compile and generate the specific API functions you need.

Part of the image processing on Jetbot also uses some functions of OpenCV's library. If you are interested in the development of Opencv computer vision library and want to know more about it, here are a few websites for your reference study:

OpenCV official homepage: https://www.opencv.org

OpenCV Chinese Forum: http://www.opencv.org.cn

OpenCV CSDN Forum: https://bbs.csdn.net/forums/OpenCV

4.2.2 Image reading and saving

img = cv2.imread('yahboom.jpg', 0)： The first argument is the path to the image, and the second argument is how to read the image.

cv2.IMREAD_COLOR： Read the image in grayscale mode, which can be represented by 0；

cv2.IMREAD_GRAYSCALE,1： read in a color picture, which can be represented by 1；

cv2.IMREAD_UNCHANGED： Read in an image and include its alpha channel, which can be represented by 2. 2, the image is saved；

cv2.imwrite('car.jpg', img)： The first parameter is the name of saved file and the second parameter is the saved image.

4.2.3 Video stream read and save

Learn to read, display and save videos. The functions are cv2.VideoCapture() and cv2.VideoWrite();

Get the video stream from the camera:

Open the camera: cap = cv2.VideoCapture(0)

The parameter 0 indicates the default camera of the device. When the device has multiple cameras, the parameter selection can be changed.

Read the video stream of the camera: ret, frame = cap.read()

No parameters, but need to be placed in an infinite loop to continuously read the video

Release camera resources: cap.release()

No parameters, be sure to turn off the camera and release resources before the program closes.

Read the video file: cap = cv2.VideoCapture('filename')

In the Jetbot robot car camera driver camera.py, this function is used to obtain the video stream data of the camera.

```
    super(Camera, self).__init__(*args, **kwargs)
    try:
        self.cap = cv2.VideoCapture(self._gst_str(), cv2.CAP_GSTREAMER)
        re, image = self.cap.read()
```

Save video:

Create a VideoWriter object and specify the output file name

Specify the use of the XVID encoder:

fourcc = cv2.VideoWriter_fourcc(*'XVID')

Specify the output file:

out = cv2.VideoWriter('output.avi', fourcc, 20.0, (640,480))

The above function parameters in turn represent the output video name, encoder, frame rate, frame width and height.

4.2.4 Draw lines, rectangles, circles, and write text

Learn to draw different geometry using OpenCV, the related functions are cv2.line(), cv2.circle(), cv2.rectangle(), cv2.putText(), etc.

Draw a line:

```
cv2.line(img, startPoint, endPoint, color, thickness)
```

About parameter:

img	The target image object that needs to be drawn
startPoint	Starting position pixel coordinates
endPoint	Ending position pixel coordinates
color	Draw the color of line used
thickness	Draw the width of line used

Draw a circle:

```
cv2.circle(img, centerPoint, radius, color, thickness)
```

About parameter:

img	The target image object that needs to be drawn
centerPoint	Pixel coordinates of the center of the circle
radius	The radius of the drawn circle
color	Draw the color of line used
thickness	Draw the width of line used

Draw a rectangle:

```
cv2.rectangle(img, point1, point2, color, thickness)
```

Point1 is the upper left vertex and point2 is the other vertex on point1 diagonal

About parameter:

img	Target image object to draw
point1	Top left vertex position pixel coordinates
point2	Bottom right vertex position pixel coordinates
color	Draw the color of line used
thickness	Draw the width of line used

Write text:

```
cv2.putText( img, text, point, font, size, color, thickness )
```

Text is the text to be written, point is the lower left coordinate of the first character, font is the font type, and size is the font size.

About parameter:

img	Target image object to draw
text	Drawn text
point	Top left vertex position pixel coordinates
font	Drawn text format
size	The size of the text drawn
color	Draw the color of line used
thickness	Draw the width of line used

4.2.5 OpenCV image scaling (cv2.resize)

This function is particularly important when using the multi-AI model framework. Because if our application needs to use two different frameworks or models trained by different training methods, but we set the capture from the beginning of the camera. The image resolution is fixed, and the resolution of their model operations is inconsistent.

At this time, we can use the image scaling function of OpenCV to scale the image to the target resolution for AI operation.

Function prototype:

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

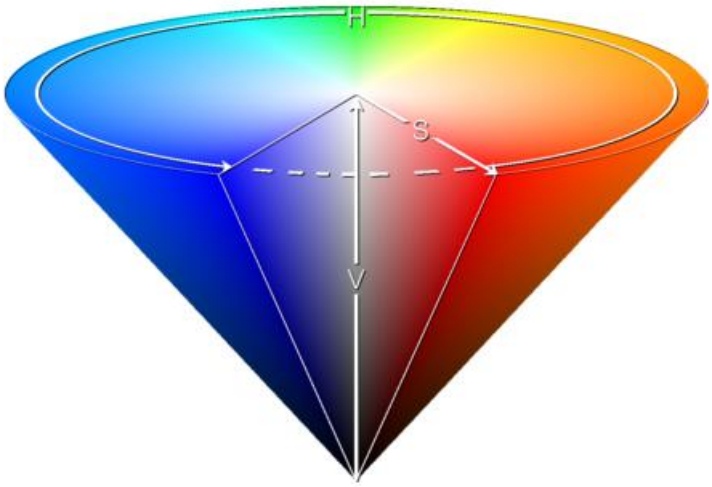
About parameter:

InputArray src	Input picture
OutputArray dst	Output picture
Size	Size of picture
fx, fy	Zoom factor along the x-axis, y-axis
interpolation	Insertion method

4.2.6 HSV color space and color space conversion (RGB-HSV)

Introduction to HSV color space:

The RGB color space, that is, the three primary color spaces, any one of which can be a mixture of the three colors. However, the effective processing of the image in the color space is generally performed in the HSV space. HSV (Hue, Saturation, Brightness Value) is a color space created according to the intuitive characteristics of the color, also called the hexagonal cone model. As shown below.



Conversion of three color spaces (gray BGR HSV):

There are more than 150 methods of color space conversion in OpenCV, but there are only two of them that we often use, namely BGR->Gray and BGR->HSV. Note that Gray and HSV cannot be converted to each other. Color space conversion: cv2.cvtColor(input_image, flag)

BGR->Gray: flag is cv2.COLOR_BGR2GRAY

BGR->HSV: flag is cv2.COLOR_BGR2HSV

The value range of the HSV color space in OpenCV:

H [0, 179] S [0, 255] V [0, 255]

Range of commonly used colors, as shown below.

	black	gray	white	red		orange	yellow	green	cyan	blue	purple
hmin	0	0	0	0	156	11	26	35	78	100	125
hmax	180	180	180	10	180	25	34	77	99	124	155
smin	0	0	0	43		43	43	43	43	43	43
smax	255	43	30	255		255	255	255	255	255	255
vmin	0	46	221	46		46	46	46	46	46	46
vmax	46	220	255	255		255	255	255	255	255	255