4. Camera display

Code path:/home/pi/Yahboom_Project/1.OpenCV Course/04 Advanced Tutorial/Camera.ipynb

Code implementation process

Since our entire tutorial runs in JupyterLab, we must understand the various components inside. Here we need to use the image display component.

Common API functions used by OpenCV:

1. cv2.VideoCapture() function:

cap = cv2.VideoCapture(0)

The parameter in VideoCapture() is 0, which means Raspberry Pi video0.

(Note: You can view the current camera through the command ls /dev/)

```
pi@yahboom4wd:~ $ ls /dev
autofs           loop7         raw             tty25  tty56      vcsa3
block            loop-control  rfkill          tty26  tty57      vcsa4
btrfs-control    mapper        rpivid-h264mem  tty27  tty58      vcsa5
bus              media0        rpivid-hevcmem  tty28  tty59      vcsa6
cachefiles       media1        rpivid-intcmem  tty29  tty6       vcsa7
char             media2        rpivid-vp9mem   tty3   tty60      vcsm
console          mem           serial0         tty30  tty61      vcsm-cma
cpu_dma_latency  mmcblk0       serial1         tty31  tty62      vcsu
cuse             mmcblk0p1     shm             tty32  tty63      vcsu1
disk             mmcblk0p2     snd             tty33  tty7       vcsu2
dma_heap         mqueue        spidev0.0       tty34  tty8       vcsu3
dri              net           spidev0.1       tty35  tty9       vcsu4
fb0              null          stderr          tty36  ttyAMA0    vcsu5
fd               port          stdin           tty37  ttyprintk  vcsu6
full             ppp           stdout          tty38  ttyS0      vcsu7
fuse             ptmx          tty             tty39  uhid       vga_arbiter
gpiochip0        pts           tty0            tty4   uinput     vhci
gpiochip1        ram0          tty1            tty40  urandom    video0
gpiomem          ram1          tty10           tty41  usb        video1
hidraw0          ram10         tty11           tty42  v4l        video10
hidraw1          ram11         tty12           tty43  vchiq      video11
hwrng            ram12         tty13           tty44  vcio       video12
i2c-1            ram13         tty14           tty45  vc-mem     video13
initctl          ram14         tty15           tty46  vcs        video14
input            ram15         tty16           tty47  vcs1       video15
kmsg             ram2          tty17           tty48  vcs2       video16
log              ram3          tty18           tty49  vcs3       watchdog
loop0            ram4          tty19           tty5   vcs4       watchdog0
loop1            ram5          tty2            tty50  vcs5       zero
loop2            ram6          tty20           tty51  vcs6
loop3            ram7          tty21           tty52  vcs7
loop4            ram8          tty22           tty53  vcsa
loop5            ram9          tty23           tty54  vcsa1
loop6            random        tty24           tty55  vcsa2
```

cap = cv2.VideoCapture(".../1.avi")

VideoCapture(".../1.avi") means that if the parameter is the video file path, the video will be opened.

2. cap.set() function

Set camera parameters. Do not modify them at will. Common configuration methods:

capture.set(CV_CAP_PROP_FRAME_WIDTH, 1920); #Width

capture.set(CV_CAP_PROP_FRAME_HEIGHT, 1080); #Height

capture.set(CV_CAP_PROP_FPS, 30); #Frame number

capture.set(CV_CAP_PROP_BRIGHTNESS, 1); #Brightness 1

capture.set(CV_CAP_PROP_CONTRAST,40); #Contrast 40

capture.set(CV_CAP_PROP_SATURATION, 50); #Saturation 50

capture.set(CV_CAP_PROP_HUE, 50); #Hue 50

capture.set(CV_CAP_PROP_EXPOSURE, 50); #Exposure 50

CV_CAP_PROP_POS_MSEC - current position of the video, get timestamp as milliseconds or video

CV_CAP_PROP_POS_FRAMES - Frame index that will be decompressed/acquired next, starting from 0

CV_CAP_PROP_POS_AVI_RATIO - relative position of the video file (0 - start of video, 1 - end of video)

CV_CAP_PROP_FRAME_WIDTH - Frame width in the video stream

CV_CAP_PROP_FRAME_HEIGHT - Frame height in the video stream

CV_CAP_PROP_FPS - frame rate

CV_CAP_PROP_FOURCC - Four characters representing the codec

CV_CAP_PROP_FRAME_COUNT - Total number of frames in the video file

The function cvGetCaptureProperty obtains the specified properties of the camera or video file.

The following are detailed parameters:

#define CV_CAP_PROP_POS_MSEC 0 //Current position in milliseconds

#define CV_CAP_PROP_POS_FRAMES 1 //Calculate the current position in frames

#define CV_CAP_PROP_POS_AVI_RATIO 2 //The relative position of the video, from 0 to 1. The first three parameters should be related to video playback and reading related dynamic information.

#define CV_CAP_PROP_FRAME_WIDTH 3 //Frame width

#define CV_CAP_PROP_FRAME_HEIGHT 4 //Frame height

#define CV_CAP_PROP_FPS 5 //Frame rate

#define CV_CAP_PROP_FOURCC 6 //4 character encoding method

#define CV_CAP_PROP_FRAME_COUNT 7 //Video frame number

#define CV_CAP_PROP_FORMAT 8 //Video format

#define CV_CAP_PROP_MODE 9 //Backend specific value indicating the current capture mode.

#define CV_CAP_PROP_BRIGHTNESS 10 //Brightness

#define CV_CAP_PROP_CONTRAST 11 //Contrast

#define CV_CAP_PROP_SATURATION 12 //Saturation

#define CV_CAP_PROP_HUE 13 //Hue

#define CV_CAP_PROP_GAIN 14 //Gain

#define CV_CAP_PROP_EXPOSURE 15 //Exposure

#define CV_CAP_PROP_CONVERT_RGB 16 //Boolean flag whether the image should be converted to RGB.

#define CV_CAP_PROP_WHITE_BALANCE 17 //White balance

#define CV_CAP_PROP_RECTIFICATION 18 //Stereo camera correction flag (note: only supports DC1394 v2. x end cur-rently)

3, cap.isOpened() function:

Return true to indicate success, false to indicate unsuccessful

4, ret,frame = cap.read() function:

cap.read() reads the video frame by frame. ret and frame are the two return values of the cap.read() method. where ret is a Boolean value. If the read frame is correct, it returns True. If the file is not read to the end, its return value is False.

Frame is the image of each frame, which is a three-dimensional matrix.

5. cv2.waitKey() function:

The parameter is 1, which means switching to the next image with a delay of 1ms. If the parameter is too large, such as cv2.waitKey(1000), it will cause lag due to too long delay.

The parameter is 0. For example, cv2.waitKey(0) only displays the current frame image, which is equivalent to pausing the video.

6. cap.release() and destroyAllWindows() functions:

cap.release() releases the video and calls destroyAllWindows() to close all image windows.

1. Import library:

import ipywidgets.widgets as widgets

2. Set up the Image component:

image_widget = widgets.Image(format='jpeg', width=600, height=500)

l format: display format.

l width: width.

l height: height.

3. Display the Image component:

display(image_widget)

4. Turn on the camera and read the image:

image = cv2.VideoCapture(0) #Open the camera

ret, frame = image.read() #Read camera data

5. Assign values to components

#Convert the image to jpeg and assign it to the video display component

image_widget.value = bgr8_to_jpeg(frame)

Code content:

```
import cv2

import ipywidgets.widgets as widgets

import threading

import time

#Set camera display component

image_widget = widgets.Image(format='jpeg', width=500, height=400)

display(image_widget) #Display camera component
```

```
#bgr8 to jpeg format

import enum

import cv2

def bgr8_to_jpeg(value, quality=75):

     return bytes(cv2.imencode('.jpg', value)[1])
```

```
image = cv2.VideoCapture(0) #Open the camera



# width=1280

# height=960

# cap.set(cv2.CAP_PROP_FRAME_WIDTH,width)#Set image width

# cap.set(cv2.CAP_PROP_FRAME_HEIGHT,height)#Set the image height



image.set(3,600)

image.set(4,500)

image.set(5, 30) #Set frame rate
```

```
image.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter.fourcc('M', 'J', 'P', 'G'))

image.set(cv2.CAP_PROP_BRIGHTNESS, 40) #Set brightness -64 - 64 0.0

image.set(cv2.CAP_PROP_CONTRAST, 50) #Set contrast -64 - 64 2.0

image.set(cv2.CAP_PROP_EXPOSURE, 156) #Set exposure value 1.0 - 5000 156.0


ret, frame = image.read() #Read camera data

image_widget.value = bgr8_to_jpeg(frame)
```

```
try:
     while 1:

          ret, frame = image.read()

          image_widget.value = bgr8_to_jpeg(frame)

          time.sleep(0.010)
except KeyboardInterrupt:
     image.release() #Capture ctrl +c to release the camera
```

If we want to end the program, we can press the icon on jupyterlab in the last code box to release the camera