6. Face detection

File path:/home/pi/Yahboom_Project/1.OpenCV course/04 advanced tutorial/face_detaction.ipynb

The test code is as follows:

```python
#bgr8 to jpeg format

import enum

import cv2



def bgr8_to_jpeg(value, quality=75):

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

```python
#Camera display component

import cv2

import ipywidgets.widgets as widgets

import threading

import time

importsys



image_widget = widgets.Image(format='jpeg', width=320, height=240)

display(image_widget)
```

```python
image = cv2.VideoCapture(0)

image.set(3,320)

image.set(4,240)

ret, frame = image.read()

image_widget.value = bgr8_to_jpeg(frame)
```

```python
# Thread function operation library

import inspect

import ctypes
```

```python
def _async_raise(tid, exctype):

    """"raises the exception, performs cleanup if needed"""

    tid = ctypes.c_long(tid)

    if not inspect.isclass(exctype):

        exctype = type(exctype)

    res = ctypes.pythonapi.PyThreadState_SetAsyncExc(tid,
ctypes.py_object(exctype))

    if res == 0:

        raise ValueError("invalid thread id")

    elif res != 1:

        # """if it returns a number greater than one, you're in trouble,

        # and you should call it again with exc=NULL to revert the effect"""

        ctypes.pythonapi.PyThreadState_SetAsyncExc(tid, None)


def stop_thread(thread):

    _async_raise(thread.ident, SystemExit)
```

```python
# body_haar = cv2.CascadeClassifier("haarcascade_upperbody.xml")

# face_haar = cv2.CascadeClassifier("haarcascade_profileface.xml")

face_haar = cv2.CascadeClassifier("haarcascade_fullbody.xml")

#eye_haar = cv2.CascadeClassifier("haarcascade_eye.xml")

eye_haar = cv2.CascadeClassifier("haarcascade_eye_tree_eyeglasses.xml")

def Camera_display():

    while 1:

        ret, frame = image.read()

        # Convert the image to black and white

        gray_img = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

# # Detect all pedestrians in the image

# bodies = body_haar.detectMultiScale(gray_img, 1.3, 5)

# for body_x,body_y,body_w,body_h in bodies:
```

```
        # cv2.rectangle(frame, (body_x, body_y), (body_x+body_w, body_y+body_h),
        (0,255,0), 2)


            faces = face_haar.detectMultiScale(gray_img, 1.1, 3)

            for face_x,face_y,face_w,face_h in faces:

                cv2.rectangle(frame, (face_x, face_y), (face_x+face_w,
        face_y+face_h), (0,255,0), 2)


            eyes = eye_haar.detectMultiScale(gray_img, 1.1, 3)

            for eye_x,eye_y,eye_w,eye_h in eyes:

                cv2.rectangle(frame, (eye_x,eye_y), (eye_x+eye_w, eye_y+eye_h),
        (255,0,0), 2)


        # eyes = eye_haar.detectMultiScale(gray_img, 1.3, 5)

        # for eye_x,eye_y,eye_w,eye_h in eyes:

        # cv2.rectangle(frame, (eye_x,eye_y), (eye_x+eye_w, eye_y+eye_h), (255,0,0), 2)


            image_widget.value = bgr8_to_jpeg(frame)

            time.sleep(0.010)
```

```
#Start thread

thread1 = threading.Thread(target=Camera_display)

thread1.setDaemon(True)

thread1.start()
```

```
#End the process and release the camera. Execute when needed.

stop_thread(thread1)

image.release()
```

After running, we can see the picture displayed by the camera below the display camera component and frame the person's face and eyes. Note that light may affect the experimental results.
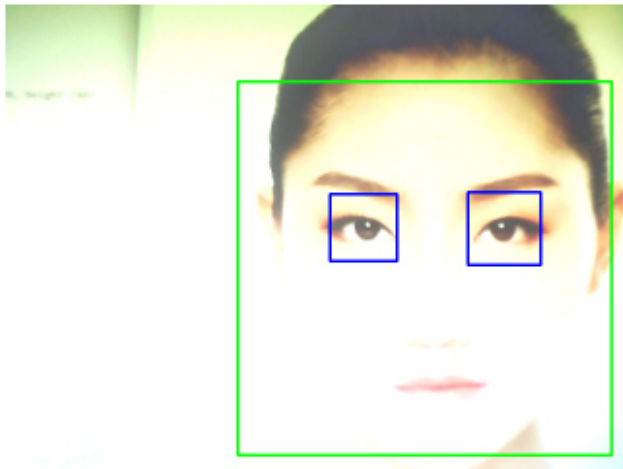
```
[1]: #bgr8转jpeg格式
     import enum
     import cv2

     def bgr8_to_jpeg(value, quality=75):
         return bytes(cv2.imencode('.jpg', value)[1])
```

```
[2]: #摄像头显示组件
     import cv2
     import ipywidgets.widgets as widgets
     import threading
     import time
     import sys

     image_widget = widgets.Image(format='jpeg', width=320, height=240)
     display(image_widget)
```



```
[3]: image = cv2.VideoCapture(0)
     image.set(3,320)
     image.set(4,240)
     image.set(cv2.CAP_PROP_BRIGHTNESS, 60) #设置亮度 -64 - 64  0.0
     image.set(cv2.CAP_PROP_CONTRAST, 50)   #设置对比度 -64 - 64  2.0
     image.set(cv2.CAP_PROP_EXPOSURE, 156)  #设置曝光值 1.0 - 5000  156.0
     ret, frame = image.read()
     image_widget.value = bgr8_to_jpeg(frame)
```

```
[4]: # 线程功能操作库
     import inspect
```

(ipykernel) | Idle