k230 self-learning object recognition

k230 and PICO2 communication

1. Experimental Prerequisites

This tutorial uses the PICO2 development board, and the corresponding routine path is [14.export\PICO-K230\16_pico_k230_self_learning.py].

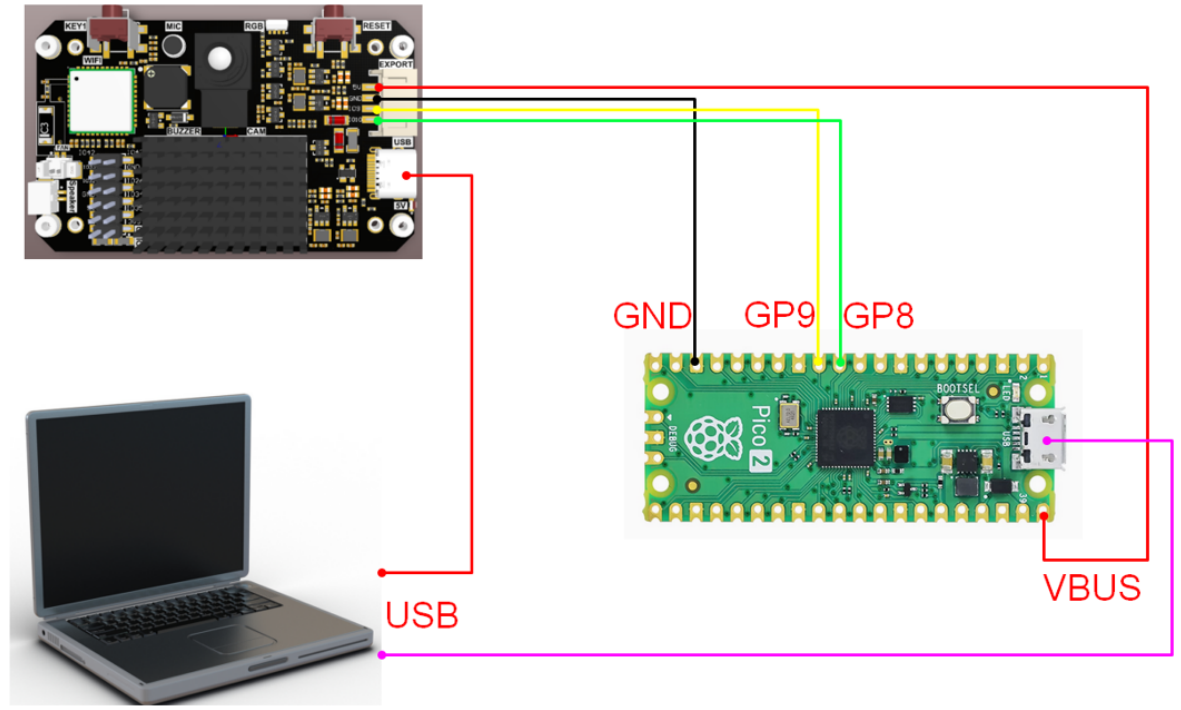
K230 needs to run the [14.export\CanmvIDE-K230\16.self_learning.py] program to start the experiment. It is recommended to download it as an offline program.

Things you need:

Windows computer
PICO2 development board
microUSB cable
K230 visual module (including TF card with image burned in)
type-C cable
connection cable

2. Experimental wiring

k230 vision module	PICO2 Development Board
5V	VCC
GND	GND
TXD(IO9)	RXD(GP9)
RXD(IO10)	TXD(GP8)



3. Main code explanation

```
from machine import UART, Pin

FUNC_ID = 16

uart1 = UART(1, baudrate=115200, tx=Pin(8), rx=Pin(9), bits=8, parity=None,
stop=0)

print("hello yahboom")


def parse_data(data):
    if data[0] == ord('$') and data[len(data)-1] == ord('#'):
        data_list = data[1:len(data)-1].decode('utf-8').split(',')
        data_len = int(data_list[0])
        data_id = int(data_list[1])
        if data_len == len(data) and data_id == FUNC_ID:
            # print(data_list)
            category = data_list[2]
            score = int(data_list[3])/100.0

            return category, score
        elif (data_len != len(data)):
            print("data len error:", data_len, len(data))
        elif(data_id != FUNC_ID):
            print("func id error:", data_id, FUNC_ID)
    else:
        print("pto error", data)
    return "", -1

last_data = bytearray()
while True:
    if uart1.any() > 0:
        cur_data = uart1.readline()
        # print("rx:", cur_data)
```

```
        if ord('\n') in cur_data:
            # data = bytearray(last_data + cur_data.decode('utf-8'), 'utf-8')
            data = last_data + cur_data
            last_data = bytearray()
            category, score = parse_data(data.rstrip(b'\n'))
            print("category:%s, score:%.2f" % (category, score))
        else:
            last_data = last_data + cur_data
```

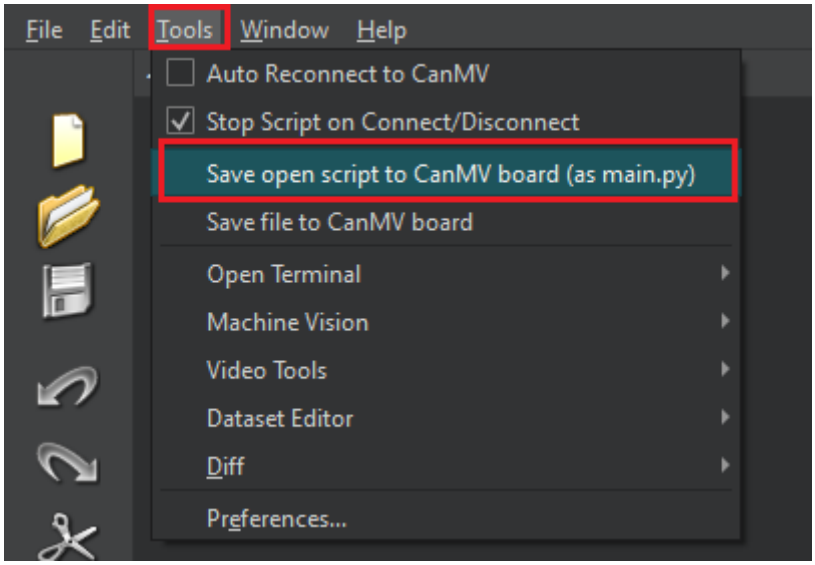
The above program is for parsing K230 data. Only when it complies with specific protocols can the corresponding data be parsed.

in

- category: is the recognized name
- score: is the score

4. Experimental Phenomenon

1. After connecting the cables, the k230 visual module runs offline. After K230 is connected to Canmv IDE, open the corresponding program, click [Save open script to CanMV board (as main.py)] on the toolbar, and then restart K230.



When K230 is turned on, a purple box will appear on the screen. Please align the purple box with the objects to be learned. There are two objects in total. Follow the on-screen prompts to learn the two objects.

After both objects have been learned, if the corresponding object appears in the purple box, the object name and score will be displayed.

2. Open the Thonny editor, connect the PICO2 mainboard, open the program file and run it.

   Note: The PICO2 mainboard needs to have the microPython firmware downloaded in advance.

3. When the K230 camera image recognizes an object, the terminal will parse and print out the information transmitted by the K230.

in

- category: is the recognized name

- score: is the score

  As shown in the figure below