

# k230 fall detection

---

## k230 fall detection

k230 and PICO2 communication

1. Experimental Prerequisites
2. Experimental wiring
3. Main code explanation
4. Experimental Phenomenon

## k230 and PICO2 communication

---

### 1. Experimental Prerequisites

This tutorial uses the PICO2 development board, and the corresponding routine path is [14.export\PICO-K230\10\_pico\_k230\_falldown\_detect.py].

K230 needs to run the [14.export\CanmvIDE-K230\10.falldown\_detect.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

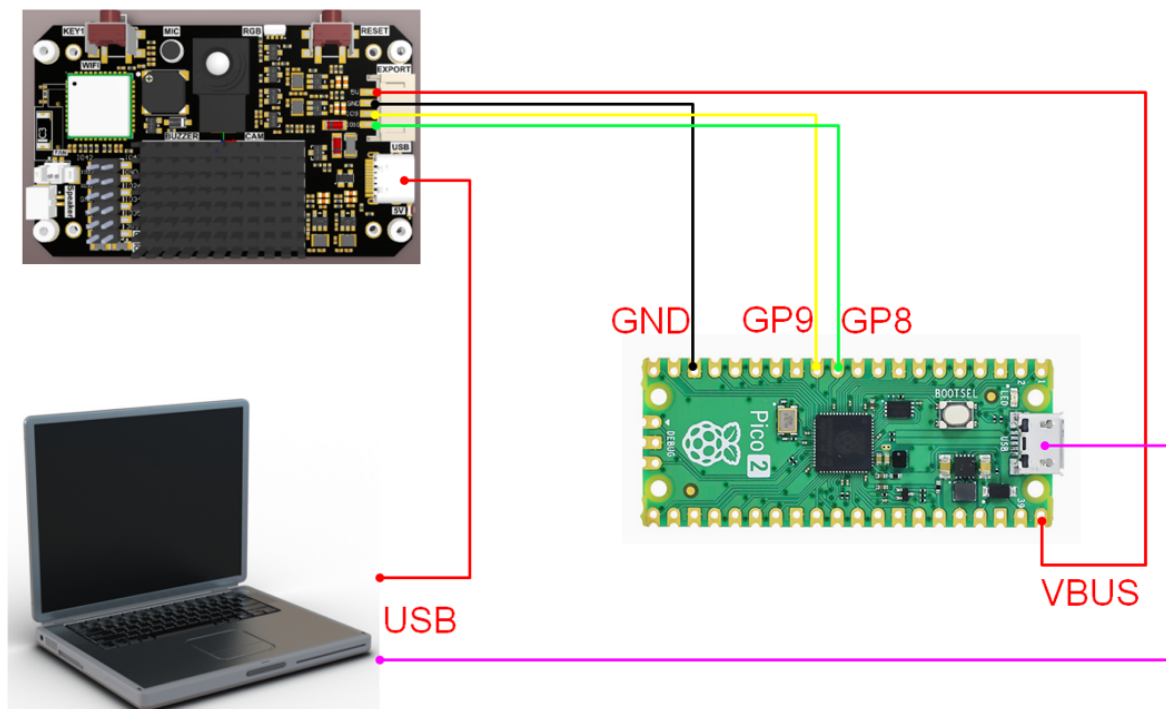
K230 visual module (including TF card with image burned in)

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 = 10

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)
            x = int(data_list[2])
            y = int(data_list[3])
            w = int(data_list[4])
            h = int(data_list[5])
            msg = data_list[6]
            score = int(data_list[7])/100.0
            return x, y, w, h, msg, 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, -1, -1, -1, "", -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()
        x, y, w, h, msg, score = parse_data(data.rstrip(b'\n'))
        print("falldown:x:%d, y:%d, w:%d, h:%d, %s, %.2f" % (x, y, w, h,
msg, score))
    else:
        last_data = last_data + cur_data

```

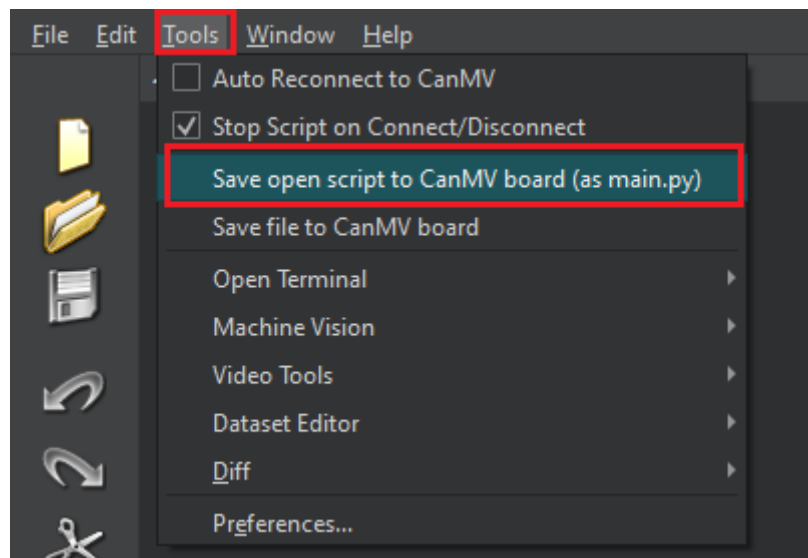
The above program is for parsing K230 data. Only when it complies with specific protocols can the corresponding data be parsed.

in

- x: is the horizontal coordinate of the upper left corner of the recognized box
- y: is the vertical coordinate of the upper left corner of the recognized box
- w: is the width of the recognized frame
- h: is the length of the recognized frame
- state: is the state of recognition. If a fall is detected, it is "Fall". If no fall is detected, it is "NoFall".
- score: is the score of the fall state

## 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.



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 a human body, the terminal will parse and print out the information transmitted by the K230.

in

- x: is the horizontal coordinate of the upper left corner of the recognized box
- y: is the vertical coordinate of the upper left corner of the recognized box
- w: is the width of the recognized frame
- h: is the length of the recognized frame
- state: is the state of recognition. If a fall is detected, it is "Fall". If no fall is detected, it is "NoFall".
- score: is the score of the fall state

As shown in the figure below

```
falldown:x:196, y:72, w:197, h:268, state:'Fall', score:0.79
falldown:x:200, y:73, w:194, h:250, state:'Fall', score:0.80
falldown:x:205, y:69, w:184, h:240, state:'Fall', score:0.86
falldown:x:196, y:61, w:194, h:250, state:'Fall', score:0.87
falldown:x:202, y:64, w:183, h:240, state:'Fall', score:0.87
```

```
[2025-04-30 11:54:59.032]# RECV ASCII>
```

```
falldown:x:202, y:41, w:183, h:268, state:'Fall', score:0.86
```

```
[2025-04-30 11:54:59.079]# RECV ASCII>
```

```
falldown:x:202, y:32, w:190, h:269, state:'Fall', score:0.84
```

```
[2025-04-30 11:54:59.142]# RECV ASCII>
```

```
falldown:x:187, y:0, w:225, h:456, state:'NoFall', score:0.36
```