PyCon China 2024

For Good . For fun.

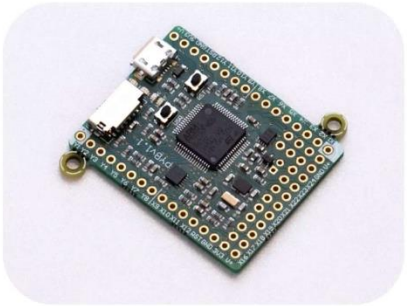
2024/11/23 中国 上海

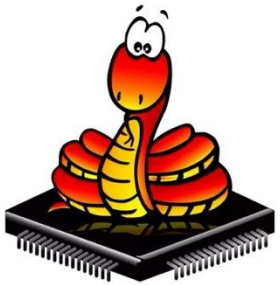
人生苦短，嵌入式AI用“Python”

Rb

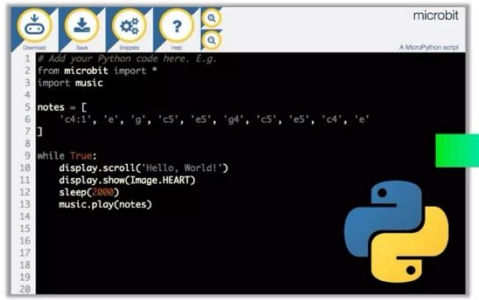
MicroPython

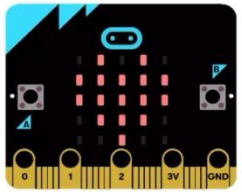
Kickstarter 2013 : PyBoard + MicroPython





BBC micro:bit : 2016





nRF51822, 16 MHz ARM Cortex-M0

MicroPython on servers and desktop

debian

FreeBSD

rpm

gentoo linux

mac OS

archlinux

https://github.com/micropython/micropython/wiki/Getting-Started

ESP8266

- Wifi 802.11 b/g/n
- 4MB Flash
- 32 KiB instruction RAM
- 32 KiB instruction cache RAM
- 80 KiB user-data RAM

US$ 5

ESP32

- Wifi 802.11 b/g/n
- Bluetooth v4.2 BR/EDR(*) and BLE(*)
- Dual Core
- 4MB Flash
- 520 KiB SRAM

US$ 8

*) BR = Basic Rate, EDR = Enhanced Data Rate, BLE = Bluetooth Low Energy

MicroPython on bare metal Raspberry Pi Zero

https://github.com/boochow/micropython-raspberrypi

Introduction to the rtthread rtos

01. 高度可定制化

- 结构化清晰，组件间耦合性低.
- 方便的定制化工具

02. 完整的生态

- 初具完整的生态：芯片，组件/软件包，开发者，网络应用

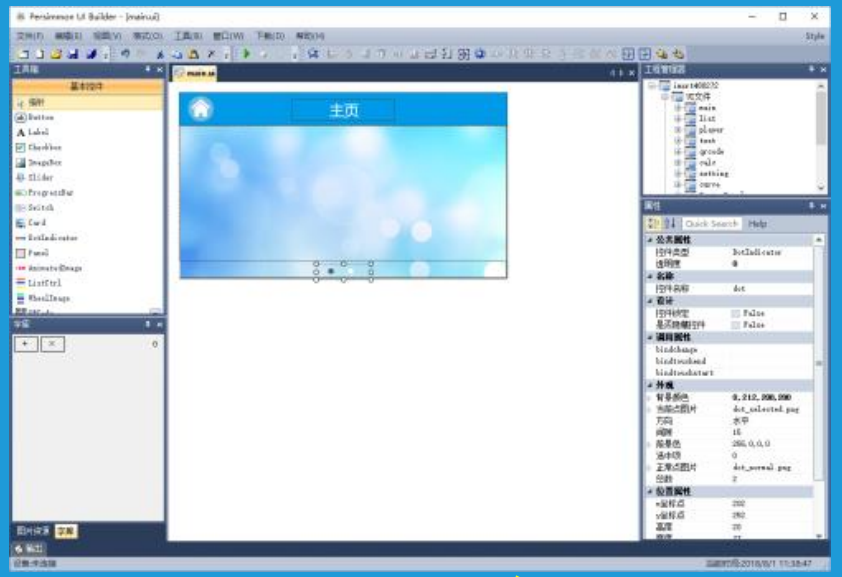
03. 一体化系统

- 应用与内核编译在一起，做为一个整体工程进行维护
- 资源占用精炼，但功能复杂后，维护代价大，容易带来安全性问题

RT-Thread Applets Support

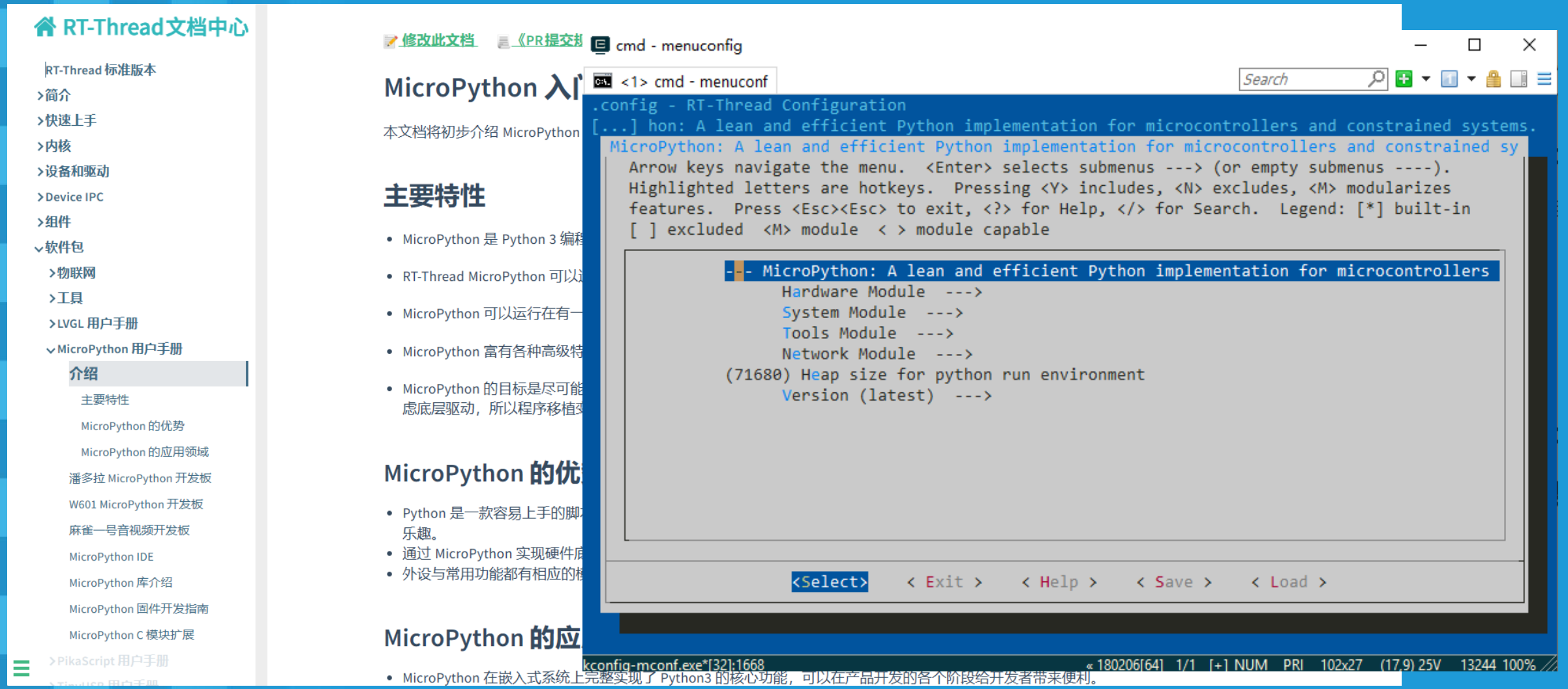
追求极致简单，最大化提升生成效率

- 突破原有固件弊端：
  - 维护多套版本
  - 程序复杂后，难以维护
- 以JavaScript、MicroPython作为应用开发语言
- 通过udb打通PC、设备的交互障碍：
  - udb over USB
  - udb over 网络
  - 叠加vscode

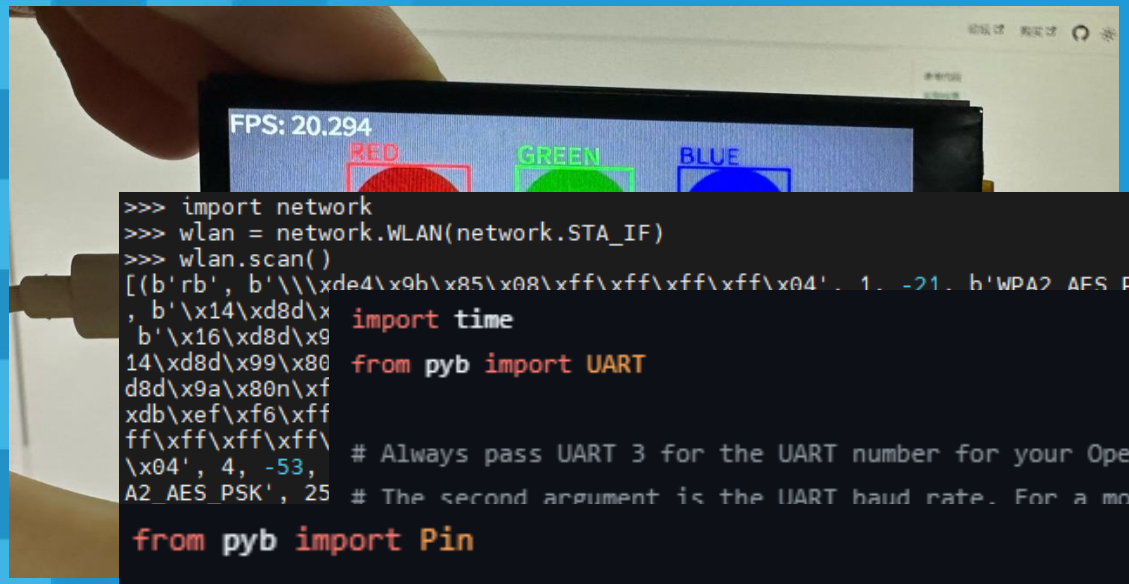


通过USB或网络下载

MicroPython for RT-Thread



MicroPython Peripherals




```
>>> import network
>>> wlan = network.WLAN(network.STA_IF)
>>> wlan.scan()
[(b'rb', b'\\\xde4\x9b\x85\x08\xff\xff\xff\xff\x04', 1, -21, b'WPA2_AES_PSK', 255), (b'realthread'
, b'\x14\xd8d\x
 b'\x16\xd8d\x9
14\xd8d\x99\x80
d8d\x9a\x80n\xf
xdb\xef\xf6\xff
ff\xff\xff\xff\
\x04', 4, -53,
A2_AES_PSK', 25
```

```
import time
from pyb import UART

# Always pass UART 3 for the UART number for your OpenMV Cam.
# The second argument is the UART baud rate. For a more advanced UART control
```

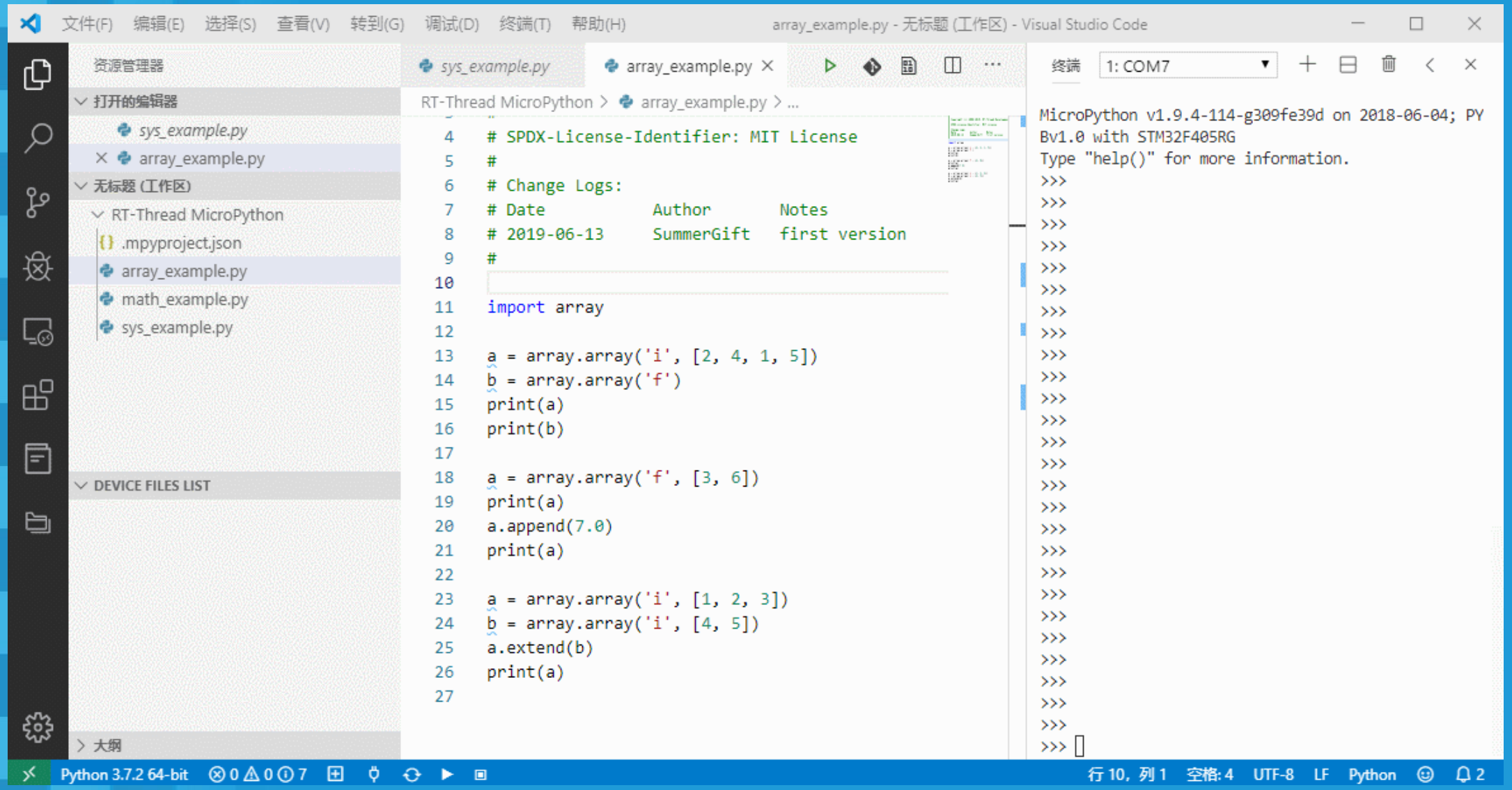
```
from pyb import Pin


# Connect a switch to pin 0 that will pull it low when the switch is closed.
# Pin 1 will then light up.
pin0 = Pin("P0", Pin.IN, Pin.PULL_UP)
pin1 = Pin("P1", Pin.OUT_PP, Pin.PULL_NONE)


while True:
    pin1.value(not pin0.value())
```

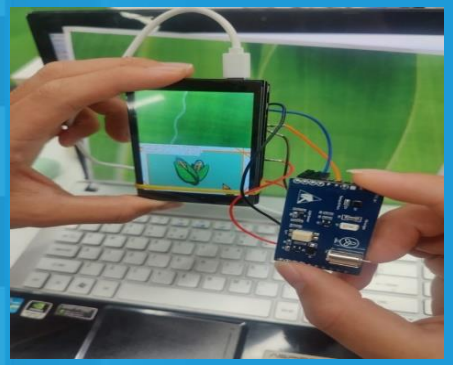
- Thread：RT-Thread 系统功能模块
- Network：网络功能配置模块
- Machine：硬件控制模块
  - PIN
  - I2C
  - SPI
  - UART
  - LCD
  - RTC
  - PWM
  - ADC
  - TIMER
  - ……

MicroPython for VSCode



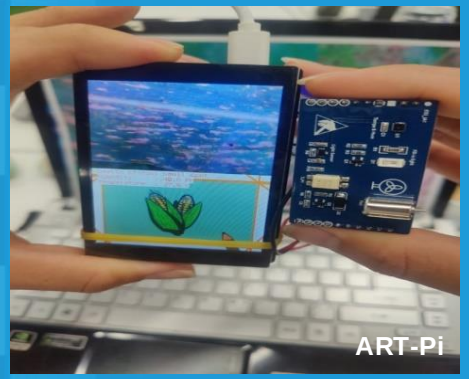
MicroPython Application

玉米叶病虫害检测



飞机图像分割

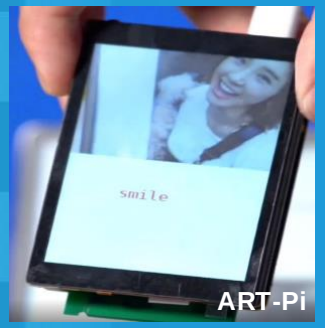
人脸特征点检测



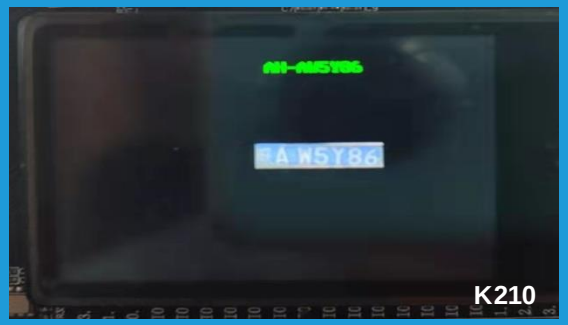
火灾检测

猜拳游戏

人脸表情分类



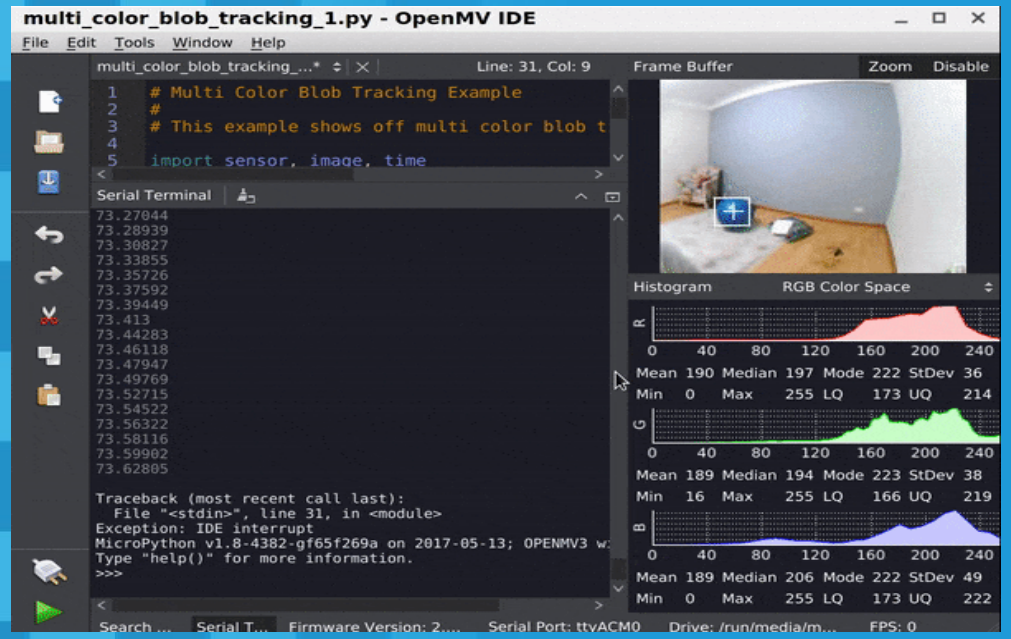
车牌识别

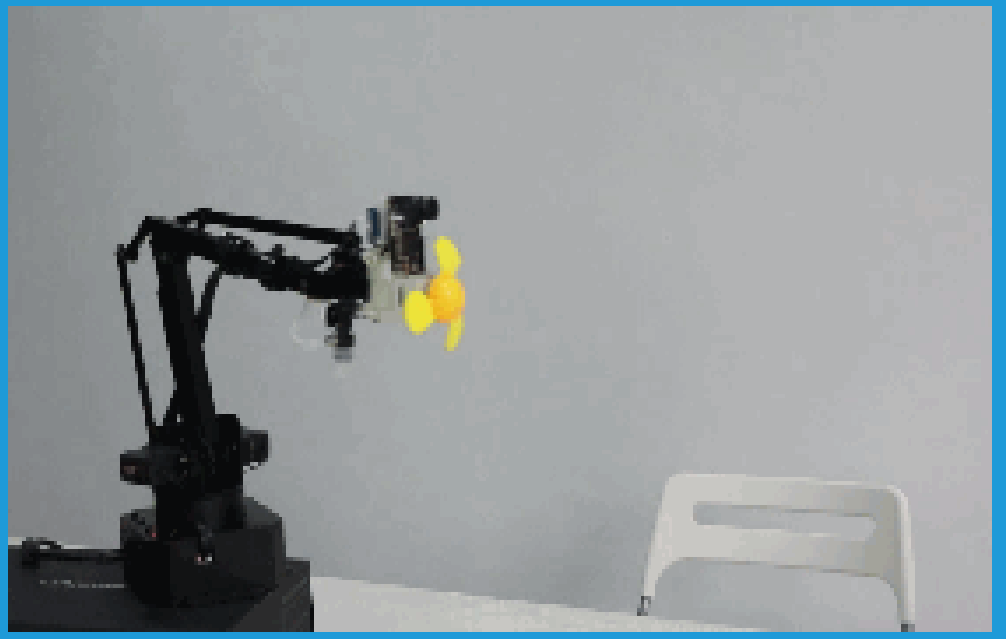


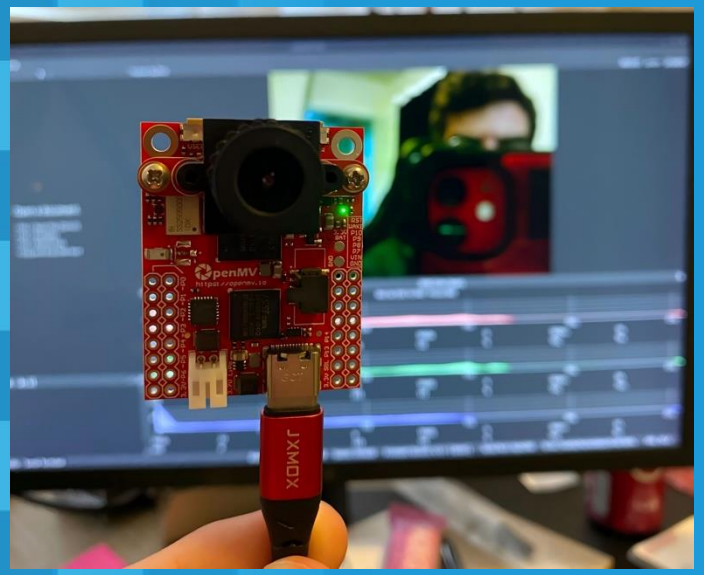
手势识别

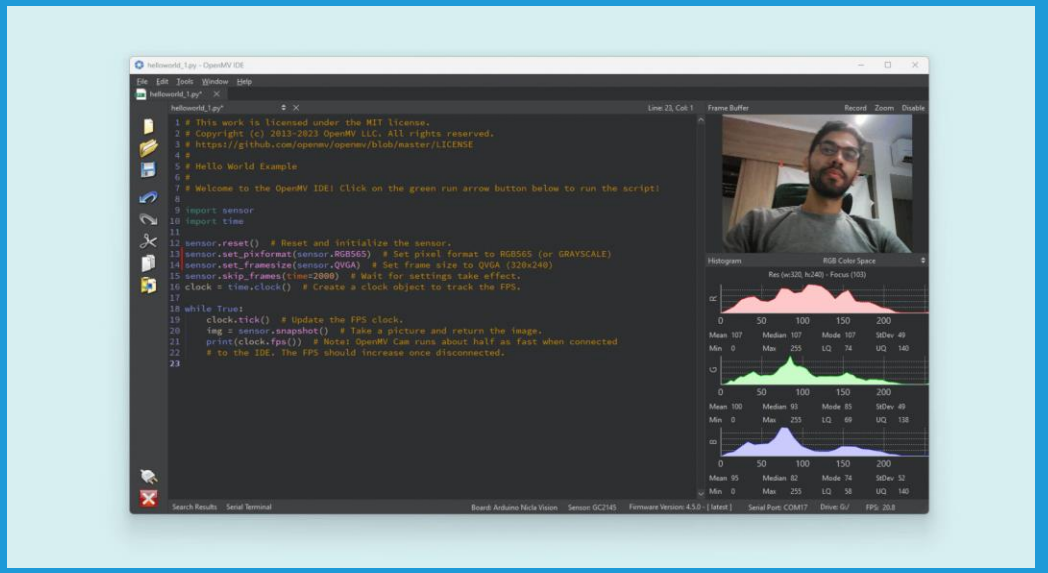
鲜花分类

OpenMV Project



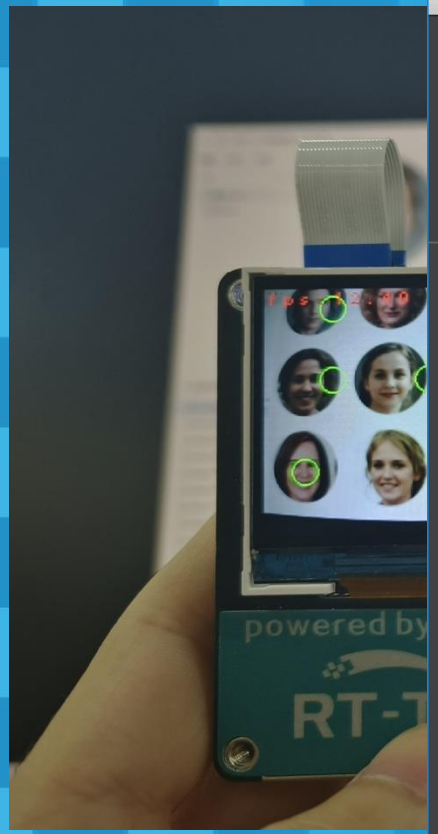






- 颜色追踪
- AptilTag
- 二维码
- 条形码
- 人脸识别
- 人眼追踪
- 圆形识别
- 矩形识别
- 数字识别
- 线性回归-巡线
- 模板匹配
- 特征点追踪
- 光流
- 边缘检测
- ……

OpenMV



helloworld 1.py - OpenMV IDE

File Edit Tools Window Help

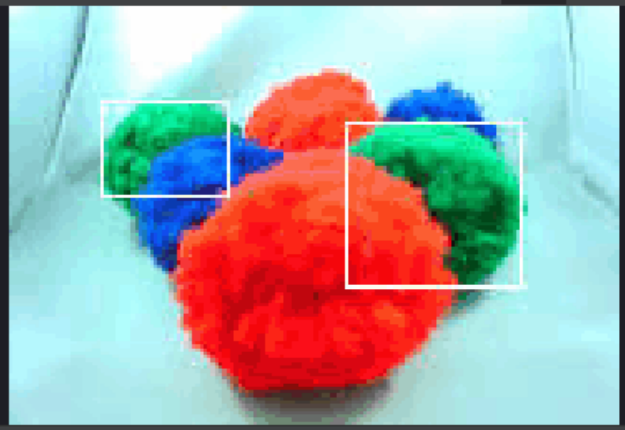
helloworld 1.py* Line: 18, Col: 34

```
1  import sensor, image, time
2
3  # Setup Camera
4  sensor.reset()
5  sensor.set_pixformat(sensor.RGB565)
6  sensor.set_framesize(sensor.QQVGA)
7  sensor.skip_frames(10)
8  threshold = (10, 90, -80, -30, 20, 50)
9  clock = time.clock()
10
11 # Find blobs
12 while(True):
13     clock.tick()
14     img = sensor.snapshot()
15     for b in img.find_blobs([threshold]):
16         img.draw_rectangle(b[0:4])
17         print("====\nBlob %s" % str(b))
18     print("FPS %d" % clock.fps())
19
```

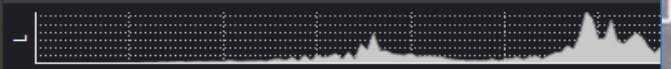
Serial Terminal

```
====
Blob (88, 33, 46, 48, 839, 116, 50, 0.9424605, 1, 1)
FPS 15
====
Blob (24, 27, 34, 28, 411, 37, 37, 2.640746, 1, 1)
====
Blob (88, 33, 46, 47, 833, 115, 50, 0.898667, 1, 1)
FPS 15
====
Blob (24, 27, 34, 28, 413, 37, 37, 2.638215, 1, 1)
====
Blob (88, 33, 47, 48, 842, 116, 50, 0.9409514, 1, 1)
FPS 15
====
Blob (24, 27, 34, 28, 408, 37, 37, 2.625929, 1, 1)
====
Blob (88, 33, 47, 48, 839, 115, 50, 0.9321314, 1, 1)
FPS 15
```

Frame Buffer Zoom Disable

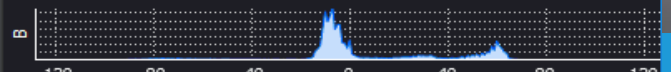


Histogram LAB Color Space



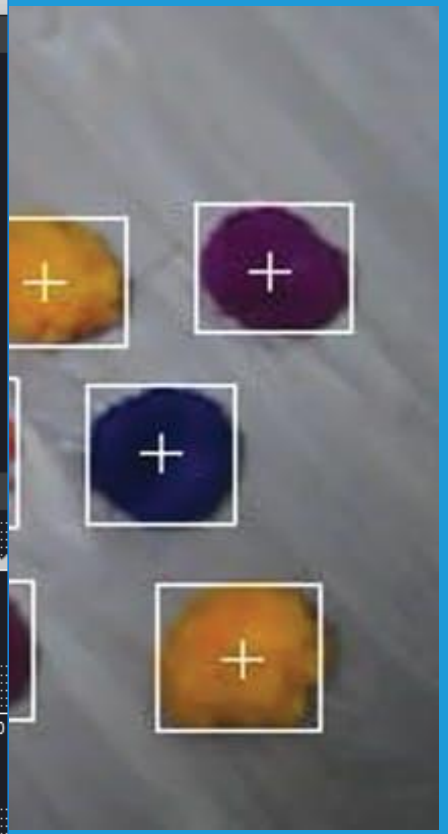
Mean	76	Median	86	Mode	88	StDev	20
Min	11	Max	100	LQ	56	UQ	92

Mean	2	Median	-17	Mode	-18	StDev	40
Min	-75	Max	81	LQ	-21	UQ	32

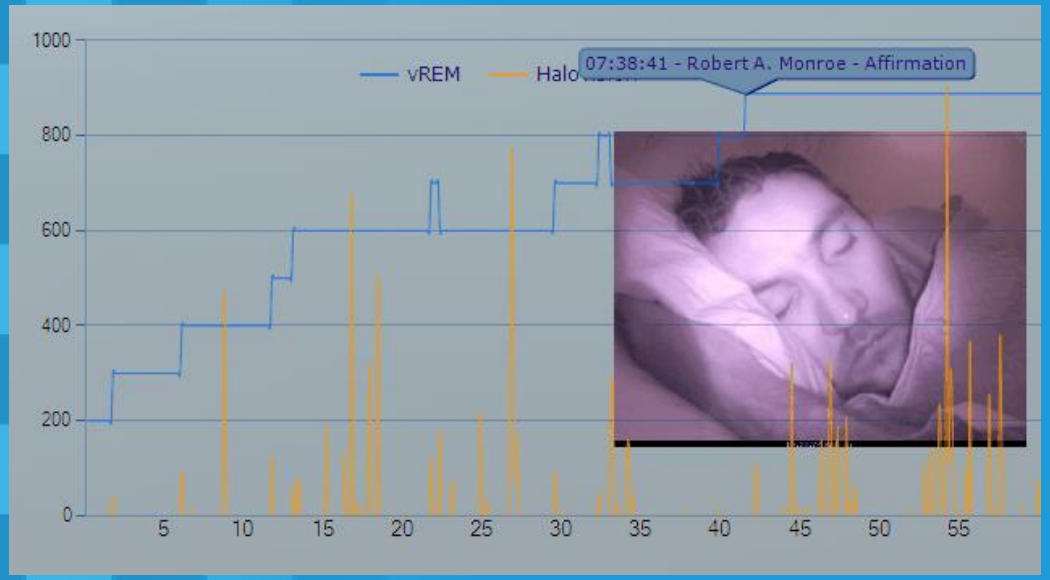


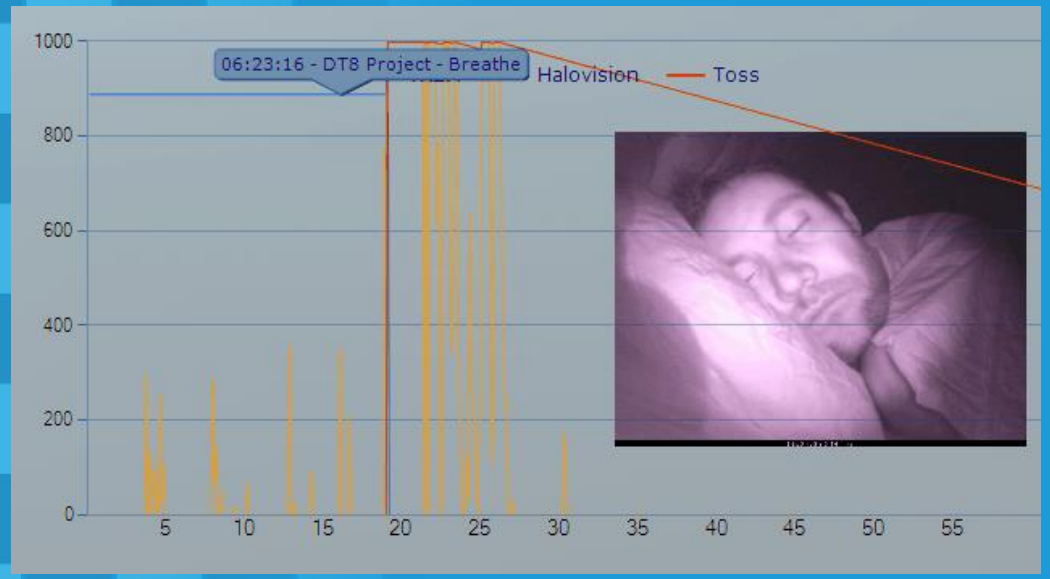
Mean	7	Median	-5	Mode	-7	StDev	31
Min	-103	Max	70	LQ	-9	UQ	29

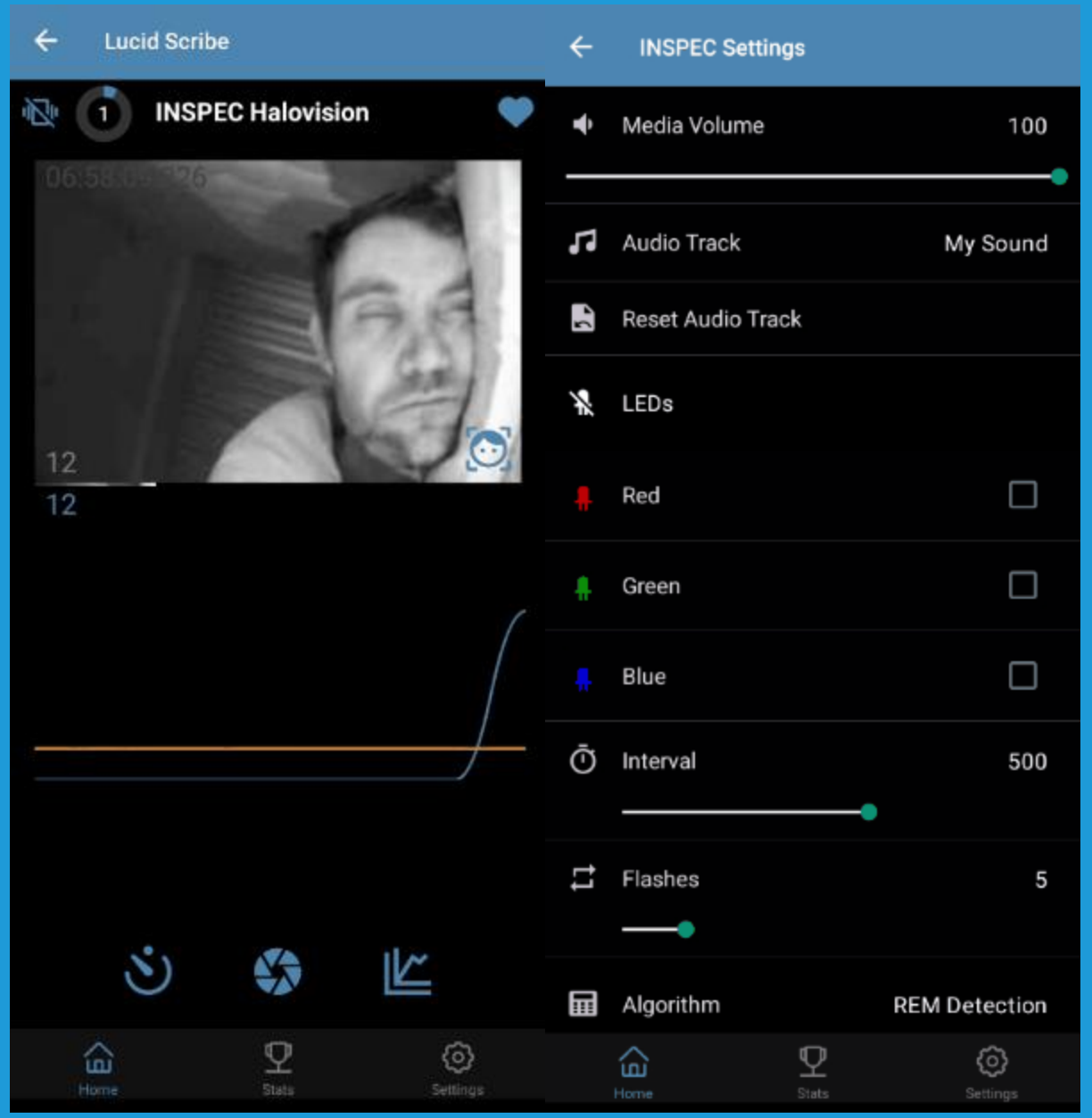
Search Results Serial Terminal Firmware Version: 2.0.0 - [latest] Serial Port: COM5 Drive: I:/ FPS: 15.2



OpenMV AI

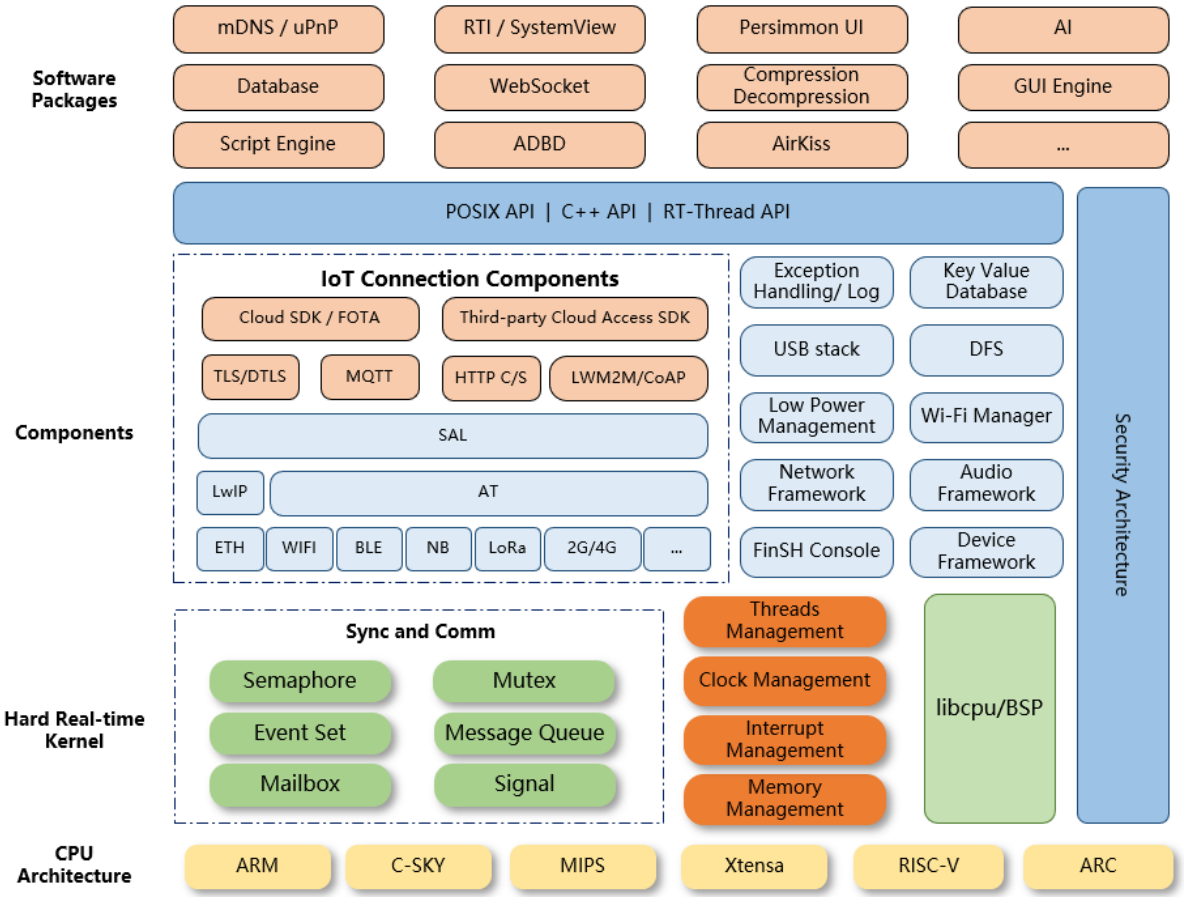


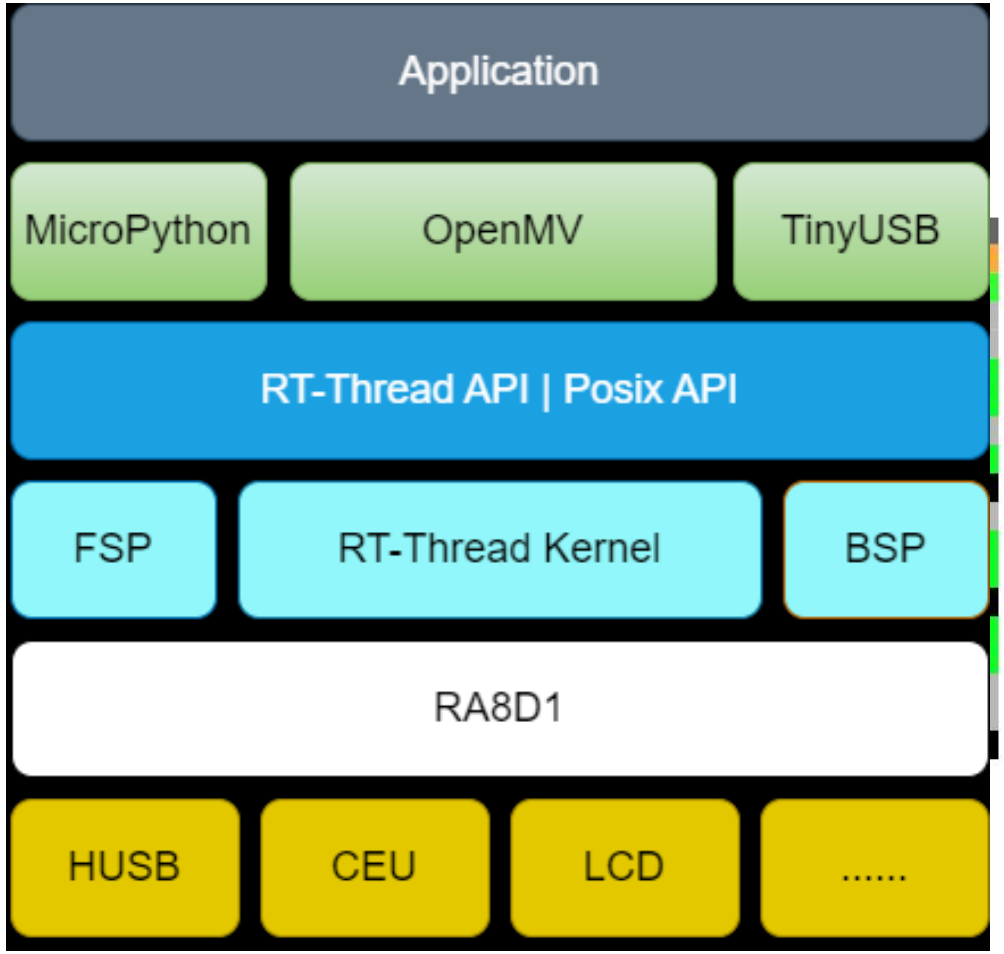




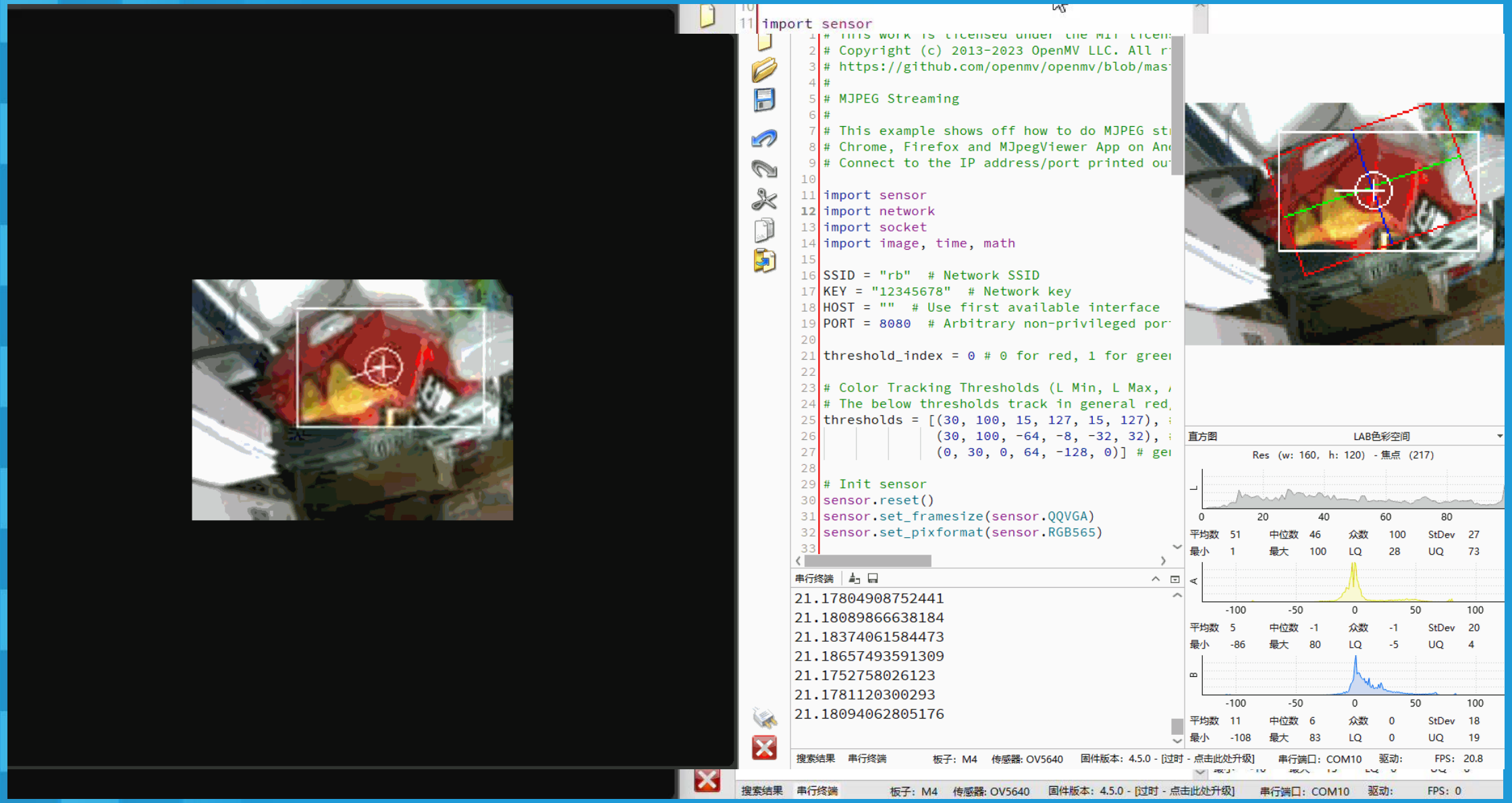
from https://openmv.io/blogs/news

OpenMV Hardware





Video Streaming

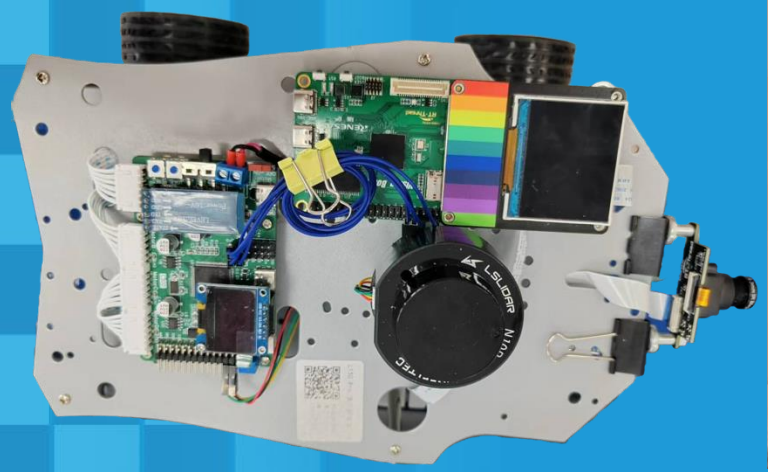


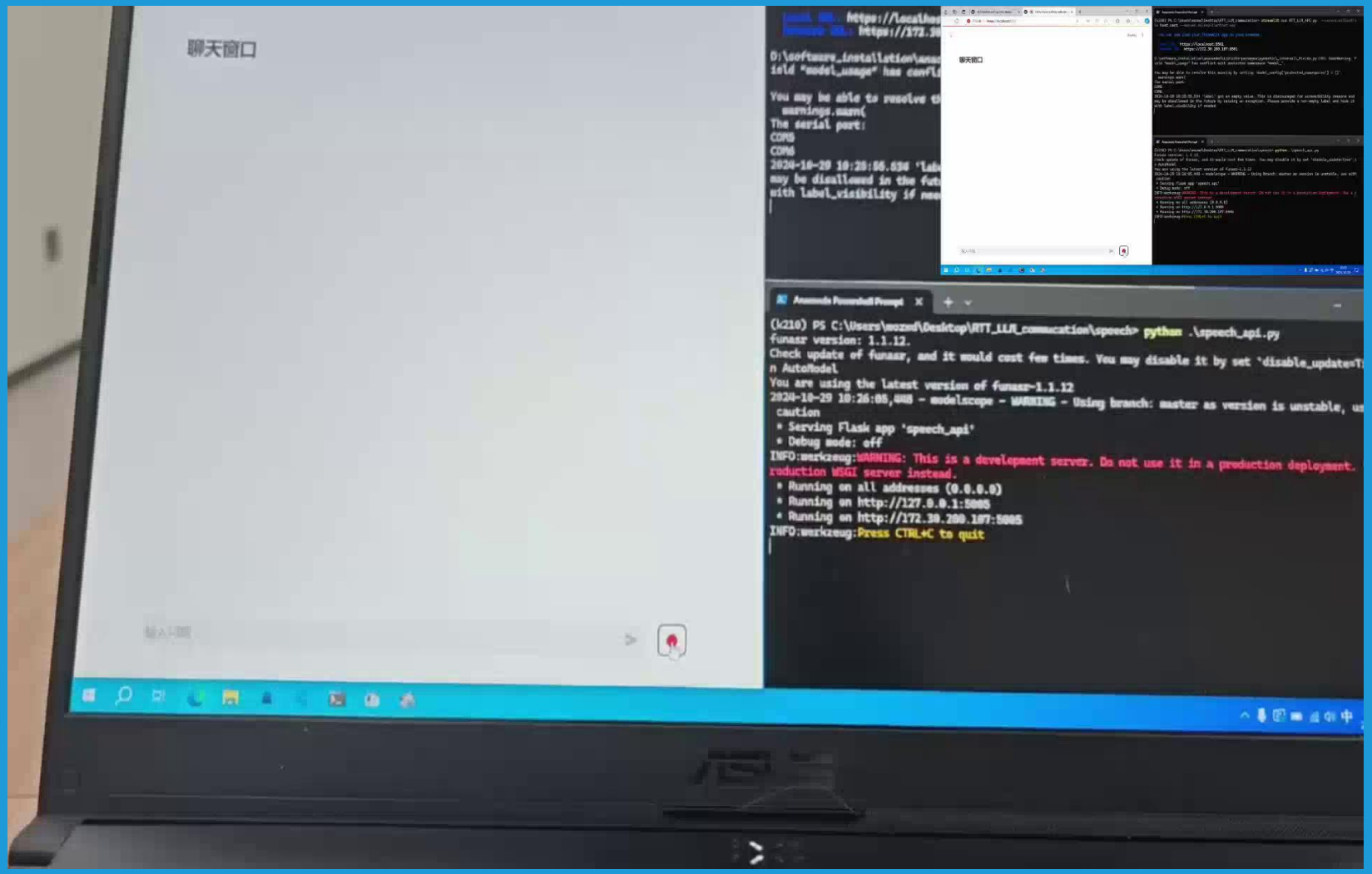
AI-CAR

颜色追踪

颜色巡线

MicroPython AI-CAR





APP流程

```
import sensor, time, image

# Reset sensor
sensor.reset()

# Sensor settings
sensor.set_contrast(3)
sensor.set_gainceiling(16)
# HQVGA and GRAYSCALE are the best for f
sensor.set_framesize(sensor.HQVGA)
sensor.set_pixformat(sensor.GRAYSCALE)

# Load Haar Cascade
# By default this will use all stages, l
face_cascade = image.HaarCascade("fronta
print(face_cascade)

# FPS clock
clock = time.clock()

while (True):
    clock.tick()

    # Capture snapshot
    img = sensor.snapshot()

    # Find objects.
    # Note: Lower scale factor scales-do
    # Higher threshold results in a high
    objects = img.find_features(face_cas

    # Draw objects
    for r in objects:
        img.draw_rectangle(r)

    # Print FPS.
    # Note: Actual FPS is higher, stream
    print(clock.fps())
```

```
sensor.reset()                      # Reset and initialize the sensor.
sensor.set_pixformat(sensor.RGB565) # Set pixel format to RGB565 (or GRAYSCALE)
sensor.set_framesize(sensor.QVGA)   # Set frame size to QVGA (320x240)
sensor.set_windowing((240, 240))    # Set 240x240 window.
sensor.skip_frames(time=2000)       # Let the camera adjust.
```

1

```
net = tf.load('<object_detection_network>', load_to_fb=True)
labels = []
```

2

```
try: # Load labels if they exist
    labels = [line.rstrip('\n') for line in open("labels.txt")]
except:
    pass

colors = [ # Add more colors if you are detecting more than 7 types of classes at once.
    (255,   0,   0),
    (  0, 255,   0),
    (255, 255,   0),
    (  0,   0, 255),
    (255,   0, 255),
    (  0, 255, 255),
    (255, 255, 255),
]

clock = time.clock()
while(True):
    clock.tick()

    img = sensor.snapshot()

    # detect() segments an object using the provided segmentation model. This produces mutliple
    # grayscale images per object class that we are trying to detect. detect() then runs
    # find_blobs() internally on the segmented images to find all blob locations and then returns
    # the bound boxes of all blobs found per object class. So, detect() returns a list of lists of
    # classification objects and the respective confidence level.

    for i, detection_list in enumerate(net.detect(img, thresholds=[(128, 255)])):
        if (i < len(labels)):
            print("********** %s **********" % labels[i])
        for d in detection_list:
            print(d)
            img.draw_rectangle(d.rect(), color=colors[i])
```

3

```
le is faster

 get their rotation angle in degrees.
n angle.

 and `y2()` methods to get their end-points
as one 4 value tuple for `draw_line()`.

 for 2.8mm lens...

the image are found. Only lines with
than `threshold` are detected...

the image contributes a magnitude value
 is the magintude for that line. Then
are added togheter. Note that `threshold`
efore merging. To see the magnitude of
`rho_margin` to 0...

 merging similar lines. If two lines
s than the margins then they are merged.

 theta_margin = 25, rho_margin = 25):
heta() <= max_degree):
255, 0, 0))
```

Thanks for watching.