7 camera display experiment

7.1 the experimental goals

This lesson is mainly for learning microPython drive the camera to the display function.

The present experiments the reference code path is: CanMV\03-Hardware\camera.py

7.2 experimental procedure

Module factory firmware has been integrated camera sensor control module, if the download other firmware, please burn back to the factory firmware and then perform the experiment.

1. Import sensor and lcd.

```
import sensor, lcd
import time
```

2. Initialize the LCD display and camera sensor.

```
lcd.init()
sensor.reset()
```

3. Camera settings pixel format is RGB565, and the frame size in pixels as QVGA（320*240）， began to grab the image, ignore the frame image.

```
sensor.set_pixformat(sensor.RGB565)
sensor.set_framesize(sensor.QVGA)
sensor.run(1)
sensor.skip_frames(10)
```

Wherein, `sensor.set_pixformat(format)`: Used to set the camera's output format.

Parameter `format` Represents the frame format, with optional frame format with `GRAYSCALE`, `RGB565`, `YUV422`; the

Returns a value of True indicates success, False indicates failure.

`sensor.set_framesize(framesize)`: Used to set the camera output frame size, k210 maximum support VGA format, is greater than the VGA will not be able to get the image.

Parameter `framesize` Indicates the frame size, optional parameters are `VGA` (640 * 480), the `QVGA` (320 * 240), the `QQVGA` (160 * 120), the screen resolution is 320*240, the recommended settings for QVGA format.

`sensor.run(enable)`: Image Capture function control, set successful return True, return False indicates failure.

Parameter `enable`: 1 means that the start grabbing image, 0 means stop grabbing images.

`sensor.skip_frames(n, [, time])`: Indicates that skips the specified frame or skip a specified time within the image, allowing the camera to image the change camera settings after stabilized.

Parameter `n`: Skip the n-th frame image;

`time`: Skip the specified time, in milliseconds.

If n and time are not specified, the method skips 300 MS of the frame; if both are specified, the method will skip n number of frames, but will in time milliseconds after the return.

4. Create a clock clock object, used to calculate the camera's frame rate.

```
clock = time.clock()
```

5. Create a while loop, get the camera of the current screen, in the upper left corner of the Add current frame rate, and display on the LCD screen.

```
while(True):
 clock.tick()
 img = sensor.snapshot()
 fps = clock.fps()
 img.draw_string(0, 0, "%2.1 ffps" %(fps), color=(0, 0, 255), scale=2.0)
 lcd.display(img)
 print(fps)
```

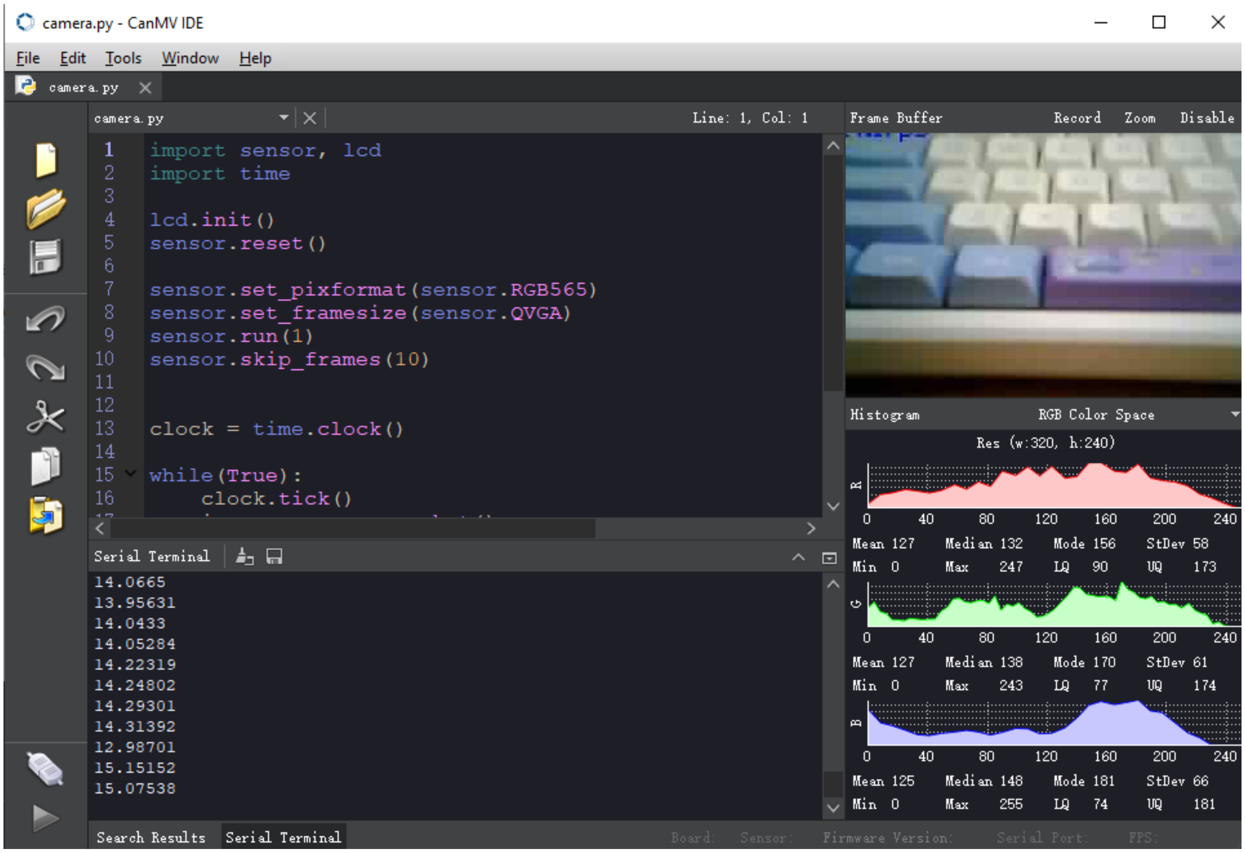
Wherein, `img = sensor.snapshot()`: Indicates to use the camera to take a picture.

`clock.tick()` And `fps = clock.fps()`: Indicates calculate the current frame rate fps.

7.3 experimental results

Connect the K210 module to the computer through the microUSB data cable, CanMV IDE click the connect button, after the connection is completed click the Run button to run the routine code. You can also download the code as main.py and run it in the K210 module.

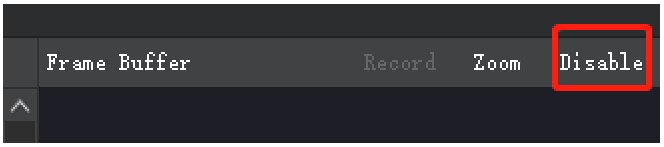
You can see the display screen real-time display the current Camera Picture, left corner of the screen to print the current camera frame rate, while the IDE to the right of the camera preview window is also synchronizes the display screen.



```
import sensor, lcd
import time

lcd.init()
sensor.reset()

sensor.set_pixformat(sensor.RGB565)
sensor.set_framesize(sensor.QVGA)
sensor.run(1)
sensor.skip_frames(10)


clock = time.clock()

while(True):
    clock.tick()
```

```
14.0665
13.95631
14.0433
14.05284
14.22319
14.24802
14.29301
14.31392
12.98701
15.15152
15.07538
```

Due to the K210 chip itself is not with the USB communication module, but the use of USB to serial chip, the transmission of the camera when the screen takes a certain amount of resources, resulting in a frame rate drop, and the easy loss of data, caused by the serial port is disconnected, so it is recommended to disable the IDE of the camera preview function, or directly download the main.py program file to the chip internal use.



7.4 the experiments are summarized

Use CanMV IDE, with the factory firmware write a good MicroPython syntax, configure the camera's simple operation, you can use the system clock signal to calculate the camera in real-time frame rate, but due to the USB-to-serial port problems, you need to disable the IDE to the image preview, to improve the frame rate, to avoid the IDE no disconnect problem.