```python
import sensor, image,time,math,pyb
from pyb import UART,LED
import json
import ustruct

sensor.reset()
sensor.set_pixformat(sensor.RGB565) # 灰度更快
sensor.set_framesize(sensor.QQVGA)
sensor.skip_frames(time = 2000)
clock = time.clock()
LED_R = pyb.LED(1)
LED_G = pyb.LED(2)
LED_B = pyb.LED(3)
uart_1 = UART(1,115200)   #定义串口1变量
uart_1.init(115200, bits=8, parity=None, stop=1)

uart = UART(3,115200)   #定义串口3变量
uart.init(115200, bits=8, parity=None, stop=1)

#define status number(convert pixel to actual)
ratio = 0.569

#define all the control variations:
x_sum_r = 0
y_sum_r = 0
x_sum_b = 0
y_sum_b = 0
x_average_r = 0
y_average_r = 0
x_average_b = 0
y_average_b = 0
x = 0
y = 0
c = 0
r = 0
i = 0
judge = 0
R_or_B = 0
CTR = 0
color = 0
shape = 0
color_shape_type = 0
color_shape_type_x = [0]*6
color_shape_type_y = [0]*6
rx_buff=[]
state = 0
tx_flag = 0
min_degree = 1
max_degree = 179
angle_degree=bytearray(3)
Angle=bytearray(3)
max_angle=0
mid_angle=0
min_angle=0
sum_angle=0
```

```
count=0
status = 0
d = 0
abs_x = 0
abs_y = 0

#define the send_data functions
#def sending_data_1(data1,data2,data3):
    #i =0
    #global uart;
    #data = ustruct.pack("bbbbbb",
                    #0x2C,
                    #0x12,
                    #int(data1),
                    #int(data2),
                    #int(data3),
                    #int(data4),
                    #0xFF,
                    #)
    #uart.write(data)
    #for i in data:
        #print((i),end=" ")
    #print("  ")

def sending_data(error_x,error_y,cmd,color_shape):
    global uart
    data = ustruct.pack("
<bbbbbbbb",0x41,0x43,0x00,0x04,int(error_x),int(error_y),int(cmd),int(color
_shape))
    #<bbbbffb为数据包内的数据类型，必须以<开头，b为8位
    #两位包头，一位类型，一位消息长度，四位数据位
    time.sleep_ms(100)
    uart.write(data)
    for i in data:
        print(" ",i,end= " ; " )
    print("*********************************************")


#定义寻找色块面积最大的函数
def find_max(blobs):
    max_size=0
    for blob in blobs:
        if blob.pixels() > max_size:
            max_blob=blob
            max_size = blob.pixels()
    return max_blob

#define calculate diviation
def diviation(x,y):
    z = math.sqrt(x*x+y*y)
    z = int(z)
    return z

#define receive function(useless)
# 0x0d data 0x5b
def Receive_Prepare(data):
    global state
    global tx_flag
```

```
    if state==0:
        if data == 0x0d:#帧头
            state = 1
        else:
            state = 0
            rx_buff.clear()
    elif state==1:
        rx_buff.append(data)
        state = 2
    elif state == 2:
        if data == 0x5b:
            tx_flag = int(rx_buff[0])
            state = 3   #同样进入else语句内
    else:
        state = 0
        rx_buff.clear()

#define diviation function
def diviation(x,y):
    global d
    d = math.sqrt(x*x+y*y)
    return d
#define select mode:
def status_judge(d):
    global status
    if(d>36):
        status = 2
    else:
        status = 3

#define "gather all data to detected the type" function


#define all the thresholds:
blue_threshold = (14, 42, 6, 72, -50, -15)
red_threshold = (0, 68, 12, 70, 19, 123)


while(True):

    LED_R.on()
    LED_B.on()

    #参数重置区
    color = 0
    shape = 0
    #color_shape_type = 0
    #i = 0
    tx_flag = 0
    color_shape_type_x = [0]*6
    color_shape_type_y = [0]*6
    #status = 0

    #摄像头初始化区
    img = sensor.snapshot()
    img.lens_corr(1.8)


```

```
    #寻找蓝色红色色块
    blobs_blue = img.find_blobs([blue_threshold],roi=(30,10,100,100))
    blobs_red = img.find_blobs([red_threshold],roi=(30,10,100,100),x_stride=15, y_stride=15)


    ##检测三角形（蓝色和红色）
    for blob in img.find_blobs([red_threshold],roi=(30,10,100,100),x_stride=15, y_stride=15):
        #img.draw_rectangle(blob.rect(),color = (0,0,0))
        #print("blob.density is ",blob.density())
        if 0.5<blob.density()<0.6:
            #print("detected red triangle:",blob.cx(),blob.cy())
            color_shape_type = 1    #***001红色三角形***
            #print("红色三角形  x,y=",blob.cx(),blob.cy())
            color_shape_type_x[0] = blob.cx()
            color_shape_type_y[0] = blob.cy()
            color_shape_type = 1
    for blob in img.find_blobs([blue_threshold],roi=(30,10,100,100),x_stride=15, y_stride=15):
        #img.draw_rectangle(blob.rect(),color = (0,0,0))
        #print("blob.density is ",blob.density())
        if 0.5<blob.density()<0.6:
            color_shape_type = 2    #***002蓝色三角形***
            #print("蓝色三角形  x,y=",blob.cx(),blob.cy())
            color_shape_type_x[1] = blob.cx()
            color_shape_type_y[1] = blob.cy()
            color_shape_type = 2
            #print("detected red triangle:",blob.cx(),blob.cy())


    #以下是检测色块，返回坐标值，同时也识别形状.
    if blobs_blue:
        for r in img.find_rects(threshold = 23000):
            #img.draw_rectangle(r.rect(), color = (0,0, 0),roi=(30,10,100,100))
            #print("蓝色方形的坐标 x,y = ",r.rect()[0],r.rect()[1])
            color_shape_type_x[5] = r.rect()[0]
            color_shape_type_y[5] = r.rect()[1]
            color_shape_type = 6
        for c in img.find_circles(threshold = 4300, x_margin =2, y_margin =2, r_margin = 10,r_min = 8,r_max = 100, r_step = 2,roi=(30,10,100,100)):
            #img.draw_circle(c.x(), c.y(), c.r(), color = (0, 0, 0),roi=(30,10,100,100))
            #print("蓝色圆形的坐标 x,y = ",c.x(), c.y())
            color_shape_type_x[3] = c.x()
            color_shape_type_y[3] = c.y()
            color_shape_type = 4
    if blobs_red:
        for r in img.find_rects(threshold = 23000):
            #img.draw_rectangle(r.rect(), color = (0,0, 0),roi=(30,10,100,100))
            #print("红色方形的坐标 x,y = ",r.rect()[0],r.rect()[1])
            color_shape_type_x[4] = r.rect()[0]
            color_shape_type_y[4] = r.rect()[1]
            color_shape_type = 5
        for c in img.find_circles(threshold = 4300, x_margin =2, y_margin =2, r_margin = 10,r_min = 5,r_max = 100, r_step = 2,roi=(30,10,100,100)):
```

```
            #img.draw_circle(c.x(), c.y(), c.r(), color = (0, 0, 0),roi=(30,10,100,100))
            #print("红色圆形的坐标 x,y = ",c.x(), c.y())
            color_shape_type_x[2] = c.x()
            color_shape_type_y[2] = c.y()
            color_shape_type = 3


    #单片机传输数据给openmv，标记需要识别的点
    c=uart_1.readchar()
    i = c
    i = 5

    #判断需要的图形坐标,传输数据。
    if(i==1):
        x_abs = math.fabs(80-color_shape_type_x[0])
        y_abs = math.fabs(60-color_shape_type_y[0])

        d = diviation(x_abs,y_abs)
        d = d*ratio

        if(d>40.23):  #(当距离大于40.23cm，是100*100识别区域的边界实际值)
            status = 1
        elif(d>10):
            status = 2
        elif(0<d<=10):
            status = 3
        else:
            status = 0

        sending_data(color_shape_type_x[0],color_shape_type_y[0],status,1)

        status = 0
    elif(i==2):
        x_abs = math.fabs(80-color_shape_type_x[1])
        y_abs = math.fabs(60-color_shape_type_y[1])

        d = diviation(x_abs,y_abs)
        d = d*ratio

        if(d>40.23):  #(当距离大于40.23cm)
            status = 1
        elif(d>10):
            status = 2
        elif(0<d<=10):
            status = 3
        else:
            status = 0


 sending_data(color_shape_type_x[1]-80,color_shape_type_y[1]-60,status,2)

        status = 0
    elif(i==3):
        x_abs = math.fabs(80-color_shape_type_x[2])
        y_abs = math.fabs(60-color_shape_type_y[2])

        d = diviation(x_abs,y_abs)
```

```
        d = d*ratio

        if(d>40.23):  #(当距离大于40.23cm)
            status = 1
        elif(d>10):
            status = 2
        elif(0<d<=10):
            status = 3
        else:
            status = 0

sending_data(color_shape_type_x[2]-80,color_shape_type_y[2]-60,status,3)
        status = 0
    elif(i==4):
        x_abs = math.fabs(80-color_shape_type_x[3])
        y_abs = math.fabs(60-color_shape_type_y[3])

        d = diviation(x_abs,y_abs)
        d = d*ratio

        if(d>40.23):  #(当距离大于40.23cm)
            status = 1
        elif(d>10):
            status = 2
        elif(0<d<=10):
            status = 3
        else:
            status = 0

sending_data(color_shape_type_x[3]-80,color_shape_type_y[3]-60,status,4)
        status = 0
    elif(i==5):#红色方形
        x_abs = math.fabs(80-color_shape_type_x[4])
        y_abs = math.fabs(60-color_shape_type_y[4])

        d = diviation(x_abs,y_abs)
        d = d*ratio

        if(d>40.23):  #(当距离大于40.23cm)
            status = 1
        elif(d>10):
            status = 2
        elif(0<d<=10):
            status = 3
        else:
            status = 0

sending_data(color_shape_type_x[4]-80,color_shape_type_y[4]-60,status,5)
        status = 0
    elif(i==6):#蓝色方形
        x_abs = math.fabs(80-color_shape_type_x[4])
        y_abs = math.fabs(60-color_shape_type_y[4])

        d = diviation(x_abs,y_abs)
        d = d*ratio

        if(d>40.23):  #(当距离大于40.23cm)
            status = 1
```

```
        elif(d>10):
            status = 2
        elif(0<d<=10              ):
            status = 3
        else:
            status = 0
        sending_data(color_shape_type_x[5]-80,color_shape_type_y[5]-60,6)
        status = 0
    elif(i==7):
        #sending_data(color_shape_type_x[color_shape_type-
1],color_shape_type_y[color_shape_type-1],0,color_shape_type)  #学习位
        sending_data(0,0,0,color_shape_type)  #学习位
    else:
        sending_data(0,0,0,0)

```