

```

1  import sensor, image, time, math, pyb
2  from pyb import UART, LED
3  import json
4  import ustruct
5
6  sensor.reset()
7  sensor.set_pixformat(sensor.RGB565) # 灰度更快
8  sensor.set_framesize(sensor.QQVGA)
9  sensor.skip_frames(time = 2000)
10 clock = time.clock()
11 LED_R = pyb.LED(1)
12 LED_G = pyb.LED(2)
13 LED_B = pyb.LED(3)
14 uart_1 = UART(1, 115200) # 定义串口1变量
15 uart_1.init(115200, bits=8, parity=None, stop=1)
16
17 uart = UART(3, 115200) # 定义串口3变量
18 uart.init(115200, bits=8, parity=None, stop=1)
19
20 #define status number(convert pixel to actual)
21 ratio = 0.569
22
23 #define all the control variations:
24 x_sum_r = 0
25 y_sum_r = 0
26 x_sum_b = 0
27 y_sum_b = 0
28 x_average_r = 0
29 y_average_r = 0
30 x_average_b = 0
31 y_average_b = 0
32 x = 0
33 y = 0
34 c = 0
35 r = 0
36 i = 0
37 judge = 0
38 R_or_B = 0
39 CTR = 0
40 color = 0
41 shape = 0
42 color_shape_type = 0
43 color_shape_type_x = [0]*6
44 color_shape_type_y = [0]*6
45 rx_buff=[]
46 state = 0
47 tx_flag = 0
48 min_degree = 1
49 max_degree = 179
50 angle_degree=bytearray(3)
51 Angle=bytearray(3)
52 max_angle=0
53 mid_angle=0
54 min_angle=0
55 sum_angle=0

```

```

56 count=0
57 status = 0
58 d = 0
59 abs_x = 0
60 abs_y = 0
61
62 #define the send_data functions
63 #def sending_data_1(data1,data2,data3):
64     #i =0
65     #global uart;
66     #data = ustruct.pack("bbbbbb",
67                           #0x2C,
68                           #0x12,
69                           #int(data1),
70                           #int(data2),
71                           #int(data3),
72                           #int(data4),
73                           #0xFF,
74                           #)
75     #uart.write(data)
76     #for i in data:
77         #print((i),end=" ")
78     #print(" ")
79
80 def sending_data(error_x,error_y,cmd,color_shape):
81     global uart
82     data = ustruct.pack("
<bbbbbb",0x41,0x43,0x00,0x04,int(error_x),int(error_y),int(cmd),int(color
_shape))
83     #<bbbbffb为数据包内的数据类型，必须以<开头，b为8位
84     #两位包头，一位类型，一位消息长度，四位数据位
85     time.sleep_ms(100)
86     uart.write(data)
87     for i in data:
88         print(" ",i,end=" ; " )
89     print("*****")
90
91
92 #定义寻找色块面积最大的函数
93 def find_max(blobs):
94     max_size=0
95     for blob in blobs:
96         if blob.pixels() > max_size:
97             max_blob=blob
98             max_size = blob.pixels()
99     return max_blob
100
101 #define calculate diviation
102 def diviation(x,y):
103     z = math.sqrt(x*x+y*y)
104     z = int(z)
105     return z
106
107 #define receive function(useless)
108 # 0x0d data 0x5b
109 def Receive_Prepare(data):
110     global state
111     global tx_flag

```

```

112     if state==0:
113         if data == 0x0d:#帧头
114             state = 1
115         else:
116             state = 0
117             rx_buff.clear()
118     elif state==1:
119         rx_buff.append(data)
120         state = 2
121     elif state == 2:
122         if data == 0x5b:
123             tx_flag = int(rx_buff[0])
124             state = 3    #同样进入else语句内
125         else:
126             state = 0
127             rx_buff.clear()
128
129     #define diviation function
130     def diviation(x,y):
131         global d
132         d = math.sqrt(x*x+y*y)
133         return d
134     #define select mode:
135     def status_judge(d):
136         global status
137         if(d>36):
138             status = 2
139         else:
140             status = 3
141
142     #define "gather all data to detected the type" function
143
144
145     #define all the thresholds:
146     blue_threshold = (14, 42, 6, 72, -50, -15)
147     red_threshold = (0, 68, 12, 70, 19, 123)
148
149
150     while(True):
151
152         LED_R.on()
153         LED_B.on()
154
155         #参数重置区
156         color = 0
157         shape = 0
158         #color_shape_type = 0
159         #i = 0
160         tx_flag = 0
161         color_shape_type_x = [0]*6
162         color_shape_type_y = [0]*6
163         #status = 0
164
165         #摄像头初始化区
166         img = sensor.snapshot()
167         img.lens_corr(1.8)
168
169

```

```

170     #寻找蓝色红色色块
171     blobs_blue = img.find_blobs([blue_threshold],roi=(30,10,100,100))
172     blobs_red = img.find_blobs([red_threshold],roi=
(30,10,100,100),x_stride=15, y_stride=15)
173
174
175     ##检测三角形（蓝色和红色）
176     for blob in img.find_blobs([red_threshold],roi=
(30,10,100,100),x_stride=15, y_stride=15):
177         #img.draw_rectangle(blob.rect(),color = (0,0,0))
178         #print("blob.density is ",blob.density())
179         if 0.5<blob.density()<0.6:
180             #print("detected red triangle:",blob.cx(),blob.cy())
181             color_shape_type = 1     ****001红色三角形***
182             #print("红色三角形  x,y=",blob.cx(),blob.cy())
183             color_shape_type_x[0] = blob.cx()
184             color_shape_type_y[0] = blob.cy()
185             color_shape_type = 1
186         for blob in img.find_blobs([blue_threshold],roi=
(30,10,100,100),x_stride=15, y_stride=15):
187             #img.draw_rectangle(blob.rect(),color = (0,0,0))
188             #print("blob.density is ",blob.density())
189             if 0.5<blob.density()<0.6:
190                 color_shape_type = 2     ****002蓝色三角形***
191                 #print("蓝色三角形  x,y=",blob.cx(),blob.cy())
192                 color_shape_type_x[1] = blob.cx()
193                 color_shape_type_y[1] = blob.cy()
194                 color_shape_type = 2
195                 #print("detected red triangle:",blob.cx(),blob.cy())
196
197
198     #以下是检测色块，返回坐标值，同时也识别形状。
199     if blobs_blue:
200         for r in img.find_rects(threshold = 23000):
201             #img.draw_rectangle(r.rect(), color = (0,0, 0),roi=
(30,10,100,100))
202             #print("蓝色方形的坐标 x,y = ",r.rect()[0],r.rect()[1])
203             color_shape_type_x[5] = r.rect()[0]
204             color_shape_type_y[5] = r.rect()[1]
205             color_shape_type = 6
206         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)):
207             #img.draw_circle(c.x(), c.y(), c.r(), color = (0, 0, 0),roi=
(30,10,100,100))
208             #print("蓝色圆形的坐标 x,y = ",c.x(), c.y())
209             color_shape_type_x[3] = c.x()
210             color_shape_type_y[3] = c.y()
211             color_shape_type = 4
212     if blobs_red:
213         for r in img.find_rects(threshold = 23000):
214             #img.draw_rectangle(r.rect(), color = (0,0, 0),roi=
(30,10,100,100))
215             #print("红色方形的坐标 x,y = ",r.rect()[0],r.rect()[1])
216             color_shape_type_x[4] = r.rect()[0]
217             color_shape_type_y[4] = r.rect()[1]
218             color_shape_type = 5
219         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)):

```

```

220         #img.draw_circle(c.x(), c.y(), c.r(), color = (0, 0, 0),roi=
(30,10,100,100))
221         #print("红色圆形的坐标 x,y = ",c.x(), c.y())
222         color_shape_type_x[2] = c.x()
223         color_shape_type_y[2] = c.y()
224         color_shape_type = 3
225
226
227     #单片机传输数据给openmv，标记需要识别的点
228     c=uart_1.readchar()
229     i = c
230     i = 5
231
232     #判断需要的图形坐标,传输数据。
233     if(i==1):
234         x_abs = math.fabs(80-color_shape_type_x[0])
235         y_abs = math.fabs(60-color_shape_type_y[0])
236
237         d = diviation(x_abs,y_abs)
238         d = d*ratio
239
240         if(d>40.23):  #(当距离大于40.23cm，是100*100识别区域的边界实际值)
241             status = 1
242         elif(d>10):
243             status = 2
244         elif(0<d<=10):
245             status = 3
246         else:
247             status = 0
248
249         sending_data(color_shape_type_x[0],color_shape_type_y[0],status,1)
250
251         status = 0
252     elif(i==2):
253         x_abs = math.fabs(80-color_shape_type_x[1])
254         y_abs = math.fabs(60-color_shape_type_y[1])
255
256         d = diviation(x_abs,y_abs)
257         d = d*ratio
258
259         if(d>40.23):  #(当距离大于40.23cm)
260             status = 1
261         elif(d>10):
262             status = 2
263         elif(0<d<=10):
264             status = 3
265         else:
266             status = 0
267
268         sending_data(color_shape_type_x[1]-80,color_shape_type_y[1]-60,status,2)
269
270         status = 0
271     elif(i==3):
272         x_abs = math.fabs(80-color_shape_type_x[2])
273         y_abs = math.fabs(60-color_shape_type_y[2])
274
275         d = diviation(x_abs,y_abs)

```

```

276         d = d*ratio
277
278         if(d>40.23):  #(当距离大于40.23cm)
279             status = 1
280         elif(d>10):
281             status = 2
282         elif(0<d<=10):
283             status = 3
284         else:
285             status = 0
286
287     sending_data(color_shape_type_x[2]-80,color_shape_type_y[2]-60,status,3)
288     status = 0
289     elif(i==4):
290         x_abs = math.fabs(80-color_shape_type_x[3])
291         y_abs = math.fabs(60-color_shape_type_y[3])
292
293         d = diviation(x_abs,y_abs)
294         d = d*ratio
295
296         if(d>40.23):  #(当距离大于40.23cm)
297             status = 1
298         elif(d>10):
299             status = 2
300         elif(0<d<=10):
301             status = 3
302         else:
303             status = 0
304
305     sending_data(color_shape_type_x[3]-80,color_shape_type_y[3]-60,status,4)
306     status = 0
307     elif(i==5):#红色方形
308         x_abs = math.fabs(80-color_shape_type_x[4])
309         y_abs = math.fabs(60-color_shape_type_y[4])
310
311         d = diviation(x_abs,y_abs)
312         d = d*ratio
313
314         if(d>40.23):  #(当距离大于40.23cm)
315             status = 1
316         elif(d>10):
317             status = 2
318         elif(0<d<=10):
319             status = 3
320         else:
321             status = 0
322
323     sending_data(color_shape_type_x[4]-80,color_shape_type_y[4]-60,status,5)
324     status = 0
325     elif(i==6):#蓝色方形
326         x_abs = math.fabs(80-color_shape_type_x[4])
327         y_abs = math.fabs(60-color_shape_type_y[4])
328
329         d = diviation(x_abs,y_abs)
330         d = d*ratio
331
332         if(d>40.23):  #(当距离大于40.23cm)
333             status = 1

```

```
331         elif(d>10):
332             status = 2
333         elif(0<d<=10           ):
334             status = 3
335         else:
336             status = 0
337             sending_data(color_shape_type_x[5]-80,color_shape_type_y[5]-60,6)
338             status = 0
339     elif(i==7):
340         #sending_data(color_shape_type_x[color_shape_type-
341 1],color_shape_type_y[color_shape_type-1],0,color_shape_type)  #学习位
342         sending_data(0,0,0,color_shape_type)  #学习位
343     else:
344         sending_data(0,0,0,0)
```