树莓派颜色识别

彩色模型

数字图像处理中常用的采用模型是 RGB（红，绿，蓝）模型和 HSV（色调，饱和度，亮度），RGB 广泛应用于彩色监视器和彩色视频摄像机，我们平时的图片一般都是 RGB 模型。而 HSV 模型更符合人描述和解释颜色的方式，HSV 的彩色描述对人来说是自然且非常直观的。

HSV 模型

HSV 模型中颜色的参数分别是：色调（H：hue），饱和度（S：saturation），亮度（V：value）。由 A. R. Smith 在 1978 年创建的一种颜色空间, 也称六角锥体模型(Hexcone Model)。

- 色调（H：hue）：用角度度量，取值范围为 0°～360°，从红色开始按逆时针方向计算，红色为 0°，绿色为 120°,蓝色为 240°。它们的补色是：黄色为 60°，青色为 180°,品红为 300°；
- 饱和度（S：saturation）：取值范围为 0.0～1.0，值越大，颜色越饱和。
- 亮度（V：value）：取值范围为 0(黑色)～255(白色)。

定义视频对象

视频对象用于捕获摄像头视频流。

```
import cv2import numpy as np



cap = cv2.VideoCapture(0)
```

设置 HSV 红色阈值

```
redLower = np.array([170, 100, 100])
redUpper = np.array([179, 255, 255])
```

获取视频帧并转成 HSV 格式

利用 cvtColor()将 BGR 格式转成 HSV 格式，参数为 cv2.COLOR_BGR2HSV。

```
# get a frame and show

ret, frame = cap.read()

cv2.imshow('Capture', frame)# change to hsv model

hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
```

获取 mask

利用 inRange()函数和 HSV 模型中红色范围的上下界获取 mask

```
# get mask

mask = cv2.inRange(hsv, redLower, redUpper)
```

腐蚀操作

```
mask = cv2.erode(mask, None, iterations=2)
```

膨胀操作

mask = cv2.dilate(mask, None, iterations=2)

在膨胀时，图像中的物体会想周围“扩张”；腐蚀时，图像中的物体会“收缩”。比较这两幅图像，由于其变化的区域只发生在边缘。所以这时将两幅图像相减，得到的就是图像中物体的边缘。

识别轮廓

```
cnts = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)[-2]
```

第一个参数是寻找轮廓的图像；

第二个参数表示轮廓的检索模式，有四种（本文介绍的都是新的 cv2 接口）：

cv2.RETR_EXTERNAL 表示只检测外轮廓

cv2.RETR_LIST 检测的轮廓不建立等级关系

cv2.RETR_CCOMP 建立两个等级的轮廓，上面的一层为外边界，里面的一层为内孔的边界信息。如果内孔内还有一个连通物体，这个物体的边界也在顶层。

cv2.RETR_TREE 建立一个等级树结构的轮廓。

第三个参数 method 为轮廓的近似办法

cv2.CHAIN_APPROX_NONE 存储所有的轮廓点，相邻的两个点的像素位置差不超过 1，即 max（abs（x1-x2），abs（y2-y1））==1

cv2.CHAIN_APPROX_SIMPLE 压缩水平方向，垂直方向，对角线方向的元素，只保留该方向的终点坐标，例如一个矩形轮廓只需 4 个点来保存轮廓信息

cv2.CHAIN_APPROX_TC89_L1，CV_CHAIN_APPROX_TC89_KCOS 使用 teh-Chinl chain 近似算法

轮廓的外接圆

cv2.minEnclosingCircle

利用迭代算法，对给定的二维点集寻找计算可包围点集的最小圆形，其定义如下

```
void cv::minEnclosingCircle ( InputArray points, Point2f & center, float & radius )
```

points:输入的二维点集，数据类型为 vector<>或 Mat 类型
center:绘制圆的圆心坐标
radius:圆的半径

矩

图像的矩可以帮助我们计算图像的质心，面积等。
函数 cv2.moments()会将计算得到的矩以一个字典的形式返回。

根据这些矩的值，我们可以计算出对象的重心：

$$C_x = \frac{M_{10}}{M_{00}}, \quad C_y = \frac{M_{01}}{M_{00}},$$

图像上绘制文字

cv2.putText(I,'there 0 error(s):',(50,150),cv2.FONT_HERSHEY_COMPLEX,6,(0,0,255),25)

各参数依次是：照片/添加的文字/左上角坐标/字体/字体大小/颜色/字体粗细

显示图片

cv2.imshow('Frame', frame)