Color recognition

Color recognition
1. Introduction
2. Code analysis
1.3 Steps

1. Introduction

This course mainly uses the camera of raspberry pie to obtain the image of the camera, and analyzes the image through the opencv library to separate and process the part of the image that conforms to the HSV value of the color you choose from the background, and then display it.HSV color correspondence table.

	黑	灰	白	红		橙	黄	绿	青	蓝	紫
hmin	0	0	0	0	156	11	26	35	78	100	125
hmax	180	180	180	10	180	25	34	77	99	124	155
smin	0	0	0	43		43	43	43	43	43	43
smax	255	43	30	255		255	255	255	255	255	255
vmin	0	46	221	46		46	46	46	46	46	46
vmax	46	220	255	255		255	255	255	255	255	255

2. Code analysis

Create two new camera display controls.

```
# 摄像头显示控件  Camera display widgets
DISPLAY_WIDTH = 640
DISPLAY_HEIGHT = 480
origin_widget = widgets.Image(format='jpeg', width=DISPLAY_WIDTH, height=DISPLAY_HEIGHT)
mask_widget = widgets.Image(format='jpeg',width=DISPLAY_WIDTH, height=DISPLAY_HEIGHT)
result_widget = widgets.Image(format='jpeg',width=DISPLAY_WIDTH, height=DISPLAY_HEIGHT)
```

Create a new button to turn off the camera.

```
# 关闭摄像头 Close_Camera
button_Close_Camera = widgets.Button(
    value=False,
    description=Name_widgets['Close_Camera'][g_ENABLE_CHINESE],
    button_style='danger', # 'success', 'info', 'warning', 'danger' or ''
    tooltip='Description',
    icon='uncheck' )


# 按键按下事件处理   Key press event processing
def on_button_close_camera(b):
    if b.description == Name_widgets['Close_Camera'][g_ENABLE_CHINESE]:
        # 停止线程，释放摄像头  Stop the thread and release the camera
        b.icon = 'uncheck'
        stop_thread(thread1)
        image.release()

# 关联按键事件回调 Button event callbacks
button_Close_Camera.on_click(on_button_close_camera)
```

Set the HSV value of the color to be recognized.

```
# 默认选择红色的，想识别其他请注释下面红色区间代码，放开后面其他区间代码段
# The default is red. If you want to identify others, please comment the
# 红色区间  Red range
color_lower = np.array([0, 43, 46])
color_upper = np.array([10, 255, 255])

# #绿色区间 Green range
# color_lower = np.array([35, 43, 46])
# color_upper = np.array([77, 255, 255])

# #蓝色区间 Blue range
# color_lower=np.array([100, 43, 46])
# color_upper = np.array([124, 255, 255])

# #黄色区间 Yello range
# color_lower = np.array([26, 43, 46])
# color_upper = np.array([34, 255, 255])
```

```
# 颜色识别功能任务  Color recognition task
def Color_Recongnize_Task():
    while(True):
        ret, frame = image.read()
        origin_widget.value = bgr8_to_jpeg(frame)

        # 改为HSV模型 change to hsv model
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

        # get mask 利用inRange()函数和HSV模型中所选颜色范围的上下界获取mask,
        # mask中原视频中的所选颜色部分会被弄成白色，其他部分黑色。
        # Get Mask uses the inRange() function and the upper and lower bounds of
        # The selected color part of the mask in the original video will be made
        mask = cv2.inRange(hsv, color_lower, color_upper)
        mask_widget.value = bgr8_to_jpeg(mask)

        # detect blue 将mask于原视频帧进行按位与操作,
        # 则会把mask中的白色用真实的图像替换
        # Detect Blue will bitwise and operate the mask in the original video fra
        # then replace the white in the mask with the real image
        res = cv2.bitwise_and(frame, frame, mask=mask)
        result_widget.value = bgr8_to_jpeg(res)
```

Start a daemon thread, run the camera identification task, and display the camera image.

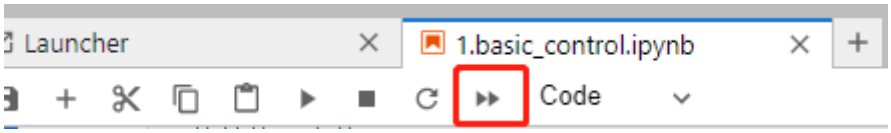
```
# 启动摄像头显示任务  Start the camera display task
thread1 = threading.Thread(target=Color_Recongnize_Task)
thread1.setDaemon(True)
thread1.start()

# 创建一个横向的盒子容器，以便将图像小部件相邻放置
# create a horizontal box container to place the image widget next to eachother
image_container_mask = widgets.HBox([origin_widget, mask_widget])
image_container_result = widgets.HBox([origin_widget, result_widget])
box_display = widgets.VBox([image_container_mask, image_container_result, button_Close_Camera])
display(box_display)
```

1.3 Steps

Open the jupyterLab client and find following code path

```
DOGZILLA/Samples/3_AI_Visual/1.color_recognition.ipynb
```

Click the following icon to run all cells, and then pull to the bottom to see the generated controls.



The left side shows the original image, and the right side shows the result after separation.

The upper part shows the selected color as white, the others are black, the lower part shows the selected color as the original color, and the others are black.

Due to the problems of camera acquisition and image analysis, there may be some differences in the recognized colors. It is better to identify the colors with a large difference from the background color.

Finally, click close_camera button turns off the camera.