```
# taking a picture to test the camera

import cv2
import numpy as np
import RPi.GPIO as GPIO
import time
import os

# initalize variables
code_running=True

# quit button
GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def GPIO17_call_back(channel):
    global code_running
    code_running=False

# led set up (second motor controller)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(24,GPIO.OUT) # PWM A
p_led=GPIO.PWM(24,50)
p_led.start(0)

# red:
light_red1=np.array([0,51,20])
dark_red1=np.array([15,255,255])
light_red2=np.array([160,51,20])
dark_red2=np.array([180,255,255])

# white:
light_white=np.array([0,0,225])
dark_white=np.array([180,30,255])

# darks:
light_dark1=np.array([0,100,0])
dark_dark1=np.array([180,255,100])
light_dark2=np.array([110,150,0])
dark_dark2=np.array([160,255,100])

# lights:
light_light=np.array([0,30,175])
dark_light=np.array([180,100,225])
```

```
46
47  # colors:
48  light_color=np.array([0,100,100])
49  dark_color=np.array([180,225,225])
50
51  # led direction
52  GPIO.output(23,GPIO.LOW)
53  GPIO.output(25,GPIO.HIGH)
54
55  cap = cv2.VideoCapture(0) #video capture source camera
56
57
58  CLOTHING_RED=False
59  CLOTHING_COLOR=False
60  CLOTHING_WHITE=False
61  CLOTHING_DARK=False
62  CLOTHING_LIGHT=False
63
64  # turn light on
65  p_led.start(100)
66  time.sleep(0.5)
67  print("light on")
68
69  ret,frame = cap.read()
70  print("picture taken")
71
72  # turn off light
73  time.sleep(0.5)
74  p_led.ChangeDutyCycle(0)
75
76  # convert RBG to HSV
77  hsv=cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)
78
79  # find colors
80  mask_red1=cv2.inRange(hsv,light_red1,dark_red1)
81  mask_red2=cv2.inRange(hsv,light_red2,dark_red2)
82  mask_white=cv2.inRange(hsv,light_white,dark_white)
83  mask_dark1=cv2.inRange(hsv,light_dark1,dark_dark1)
84  mask_dark2=cv2.inRange(hsv,light_dark2,dark_dark2)
85  mask_light=cv2.inRange(hsv,light_light,dark_light)
86  mask_color=cv2.inRange(hsv,light_color,dark_color)
87
88  # output
89  red1=cv2.bitwise_and(frame,frame,mask=mask_red1)
90  red2=cv2.bitwise_and(frame,frame,mask=mask_red2)
```

```
 91  white=cv2.bitwise_and(frame,frame,mask=mask_white)
 92  dark1=cv2.bitwise_and(frame,frame,mask=mask_dark1)
 93  dark2=cv2.bitwise_and(frame,frame,mask=mask_dark2)
 94  light=cv2.bitwise_and(frame,frame,mask=mask_light)
 95  color=cv2.bitwise_and(frame,frame,mask=mask_color)
 96
 97  # Find area
 98  area_red1 = 0
 99  area_red2 = 0
100  area_white = 0
101  area_dark1 = 0
102  area_dark2 = 0
103  area_light = 0
104  area_color = 0
105  area_red = 0
106  area_dark = 0
107
108  # Remove noise
109  kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,3))
110  opening_red1 = cv2.morphologyEx(mask_red1, cv2.MORPH_OPEN, kernel, iterations=1)
111  opening_red2 = cv2.morphologyEx(mask_red2, cv2.MORPH_OPEN, kernel, iterations=1)
112  opening_white = cv2.morphologyEx(mask_white, cv2.MORPH_OPEN, kernel, iterations=1)
113  opening_dark1 = cv2.morphologyEx(mask_dark1, cv2.MORPH_OPEN, kernel, iterations=1)
114  opening_dark2 = cv2.morphologyEx(mask_dark2, cv2.MORPH_OPEN, kernel, iterations=1)
115  opening_light = cv2.morphologyEx(mask_light, cv2.MORPH_OPEN, kernel, iterations=1)
116  opening_color = cv2.morphologyEx(mask_color, cv2.MORPH_OPEN, kernel, iterations=1)
117
118
119  # Find contours
120  original = frame.copy()
121  cnts_red1 = cv2.findContours(opening_red1, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
122  cnts_red1 = cnts_red1[0] if len(cnts_red1) == 2 else cnts_red1[1]
123  cnts_red2 = cv2.findContours(opening_red2, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
124  cnts_red2 = cnts_red2[0] if len(cnts_red2) == 2 else cnts_red2[1]
125  cnts_white = cv2.findContours(opening_white, cv2.RETR_EXTERNAL,                    ↲
     cv2.CHAIN_APPROX_SIMPLE)
126  cnts_white = cnts_white[0] if len(cnts_white) == 2 else cnts_white[1]
127  cnts_dark1 = cv2.findContours(opening_dark1, cv2.RETR_EXTERNAL,                    ↲
     cv2.CHAIN_APPROX_SIMPLE)
128  cnts_dark1 = cnts_dark1[0] if len(cnts_dark1) == 2 else cnts_dark1[1]
129  cnts_dark2 = cv2.findContours(opening_dark2, cv2.RETR_EXTERNAL,                    ↲
     cv2.CHAIN_APPROX_SIMPLE)
130  cnts_dark2= cnts_dark2[0] if len(cnts_dark2) == 2 else cnts_dark2[1]
131  cnts_light = cv2.findContours(opening_light, cv2.RETR_EXTERNAL,                    ↲
     cv2.CHAIN_APPROX_SIMPLE)
```

```
132  cnts_light = cnts_light[0] if len(cnts_light) == 2 else cnts_light[1]
133  cnts_color = cv2.findContours(opening_color, cv2.RETR_EXTERNAL,
     cv2.CHAIN_APPROX_SIMPLE)
134  cnts_color = cnts_color[0] if len(cnts_color) == 2 else cnts_color[1]
135
136
137  for c in cnts_red1:
138      area_red1 += cv2.contourArea(c)
139      cv2.drawContours(original,[c], 0, (0,0,0), 2)
140  for c in cnts_red2:
141      area_red2 += cv2.contourArea(c)
142      cv2.drawContours(original,[c], 0, (0,0,0), 2)
143  for c in cnts_white:
144      area_white += cv2.contourArea(c)
145      cv2.drawContours(original,[c], 0, (0,0,0), 2)
146  for c in cnts_dark1:
147      area_dark1 += cv2.contourArea(c)
148      cv2.drawContours(original,[c], 0, (0,0,0), 2)
149  for c in cnts_dark2:
150      area_dark2 += cv2.contourArea(c)
151      cv2.drawContours(original,[c], 0, (0,0,0), 2)
152  for c in cnts_light:
153      area_light += cv2.contourArea(c)
154      cv2.drawContours(original,[c], 0, (0,0,0), 2)
155  for c in cnts_color:
156      area_color += cv2.contourArea(c)
157      cv2.drawContours(original,[c], 0, (0,0,0), 2)
158
159
160  # combine red and darks
161  area_red=area_red1+area_red2
162  area_dark=area_dark1+area_dark2
163
164  # Determine dominant color
165  if area_red>area_white and area_red>area_dark and area_red>area_light and
     area_red>area_color:
166      print('ning smells like cherries')
167      CLOTHING_RED = True
168  elif area_white>area_red and area_white>area_dark and area_white>area_light and
     area_white>area_color:
169      print('ning smells like a cloud')
170      CLOTHING_WHITE = True
171  elif area_dark>area_red and area_dark>area_white and area_dark>area_light and
     area_dark>area_color:
172      print('ning smells like dark chocolate')
```

```
173         CLOTHING_DARK = True
174     elif area_light>area_dark and area_light>area_red and area_light>area_white and
        area_light>area_color:
175         print('ning smells like an LED')
176         CLOTHING_LIGHT = True
177     else:
178         print('ning smells like a lucky charms')
179         CLOTHING_COLOR = True
180
181
182     cv2.imshow('img1',frame)
183     cv2.imshow('white',white)
184     cv2.imshow('light',light)
185     cv2.imshow('color',color)
186     cv2.imshow('red1',red1)
187     cv2.imshow('red2',red2)
188     cv2.imshow('dark1',dark1)
189     cv2.imshow('dark2',dark2)
190
191     cv2.waitKey(0)
192     time.sleep(0.2)
193     cap.release()
194     cv2.destroyAllWindows()
195
196
197
198
199
```