

```
1  # import packages
2  import pygame
3  from pygame.locals import*
4  import os
5  import RPi.GPIO as GPIO
6  import subprocess
7  import random
8  import time
9  import cv2
10 import numpy as np
11 import cv2 as cv
12
13 # set up piTFT touchscreen
14 #os.putenv('SDL_VIDEODRIVER', 'fbcon')
15 #os.putenv('SDL_FBDEV', '/dev/fb0')
16 #os.putenv('SDL_MOUSEDRV', 'TSLIB')
17 #os.putenv('SDL_MOUSEDEV', '/dev/input/touchscreen')
18
19 pygame.init()
20 #pygame.mouse.set_visible(False)
21
22 # set up quit button
23 GPIO.setmode(GPIO.BCM)
24 GPIO.setup(17, GPIO.IN, pull_up_down=GPIO.PUD_UP)
25
26 def GPIO17_call_back(channel):
27     global code_running
28     global color_selection
29     global select
30     code_running=False
31     color_selection=False
32     select=False
33     start=False
34     sort=False
35
36 GPIO.add_event_detect(17, GPIO.FALLING, callback=GPIO17_call_back,bouncetime=300)
37
38 # motor rotation set up (first motor controller)
39 GPIO.setup(5,GPIO.OUT)
40 GPIO.setup(6,GPIO.OUT)
41 GPIO.setup(13,GPIO.OUT) # PWM A
42
43 # track motor set up (first motor controller)
44 GPIO.setup(20,GPIO.OUT)
45 GPIO.setup(21,GPIO.OUT)
```

```
46  GPIO.setup(16,GPIO.OUT) # PWM B
47
48  # led set up (second motor controller)
49  GPIO.setup(25,GPIO.OUT)
50  GPIO.setup(23,GPIO.OUT)
51  GPIO.setup(24,GPIO.OUT) # PWM A
52
53  # track motor set up (second motor controller)
54  GPIO.setup(22,GPIO.OUT)
55  GPIO.setup(4,GPIO.OUT)
56  GPIO.setup(27,GPIO.OUT) # PWM B
57
58  # color set up
59  WHITE=255,255,255
60  BLACK=0,0,0
61  RED = 230,0,0
62  GREEN = 0, 200, 0
63  GREEN_START = 35,150,10
64
65  # screen setup
66  screen = pygame.display.set_mode((320,240))
67  screen.fill(BLACK)
68  my_font=pygame.font.Font(None,25)
69  my_font2=pygame.font.Font(None,20)
70  screen.fill(BLACK)
71  my_text=""
72  text_pos= [0,0]
73
74  # import basket image
75  basket_image=pygame.image.load("laundry_basket.png")
76  basket_image=pygame.transform.scale(basket_image, (110,110))
77
78  def basket(x,y):
79      screen.blit(basket_image,(x,y))
80
81  # initalize variables
82  code_running=True
83  color_selection=True
84  select=True
85  sort=False
86  start=True
87  starttime=time.time()
88
89  # basket 1 initialization
90  choose_white1=False
```

```
91 choose_color1=False
92 choose_red1=False
93 choose_light1=False
94 choose_dark1=False
95
96 #basket 2 initialization
97 choose_white2=False
98 choose_color2=False
99 choose_red2=False
100 choose_light2=False
101 choose_dark2=False
102
103 #basket 3 initialization
104 choose_white3=False
105 choose_color3=False
106 choose_red3=False
107 choose_light3=False
108 choose_dark3=False
109
110 # clothing color initialization
111 CLOTHING_RED=False
112 CLOTHING_COLOR=False
113 CLOTHING_WHITE=False
114 CLOTHING_DARK=False
115 CLOTHING_LIGHT=False
116
117 # red:
118 light_red1=np.array([0,51,20])
119 dark_red1=np.array([15,255,255])
120 light_red2=np.array([160,51,20])
121 dark_red2=np.array([180,255,255])
122
123 # white:
124 #light_white=np.array([0,0,225])
125 #dark_white=np.array([180,30,255])
126 light_white=np.array([0,0,200])
127 dark_white=np.array([180,80,255])
128
129 # darks:
130 light_dark1=np.array([0,100,0])
131 dark_dark1=np.array([180,255,100])
132 light_dark2=np.array([110,150,0])
133 dark_dark2=np.array([160,255,100])
134
135 # lights:
```

```
136 light_light=np.array([0,30,200])
137 dark_light=np.array([180,100,225])
138
139 # colors:
140 #light_color=np.array([0,100,100])
141 #dark_color=np.array([180,225,225])
142 light_color=np.array([0,75,75])
143 dark_color=np.array([180,175,175])
144
145 # Initialize motors
146 p1=GPIO.PWM(13,50)
147 p2=GPIO.PWM(16,50)
148 p_led=GPIO.PWM(24,50)
149 p_rot=GPIO.PWM(27,50)
150 p1.start(0)
151 p2.start(0)
152 p_rot.start(0)
153 p_led.start(0)
154
155 # video capture source camera setup
156 #cap = cv2.VideoCapture(0)
157
158 # led direction
159 GPIO.output(23,GPIO.LOW)
160 GPIO.output(25,GPIO.HIGH)
161
162 # motor direction
163
164 def basket1():
165     # motor rotate
166     print("rotate")
167     GPIO.output(4,GPIO.LOW)
168     GPIO.output(22,GPIO.HIGH)
169     p_rot.ChangeDutyCycle(80)
170     time.sleep(0.25)
171     p_rot.ChangeDutyCycle(0)
172
173     # motor forward
174     print("forward")
175     GPIO.output(6,GPIO.HIGH)
176     GPIO.output(5,GPIO.LOW)
177     GPIO.output(20,GPIO.HIGH)
178     GPIO.output(21,GPIO.LOW)
179     p1.ChangeDutyCycle(100)
180     p2.ChangeDutyCycle(100)
```

```
181     time.sleep(0.7)
182
183     # motor pause
184     p1.ChangeDutyCycle(0)
185     p2.ChangeDutyCycle(0)
186     time.sleep(0.2)
187
188     # motor back
189     print("back")
190     GPIO.output(5, GPIO.HIGH)
191     GPIO.output(6, GPIO.LOW)
192     GPIO.output(21, GPIO.HIGH)
193     GPIO.output(20, GPIO.LOW)
194     p1.ChangeDutyCycle(85)
195     p2.ChangeDutyCycle(85)
196     time.sleep(1.1)
197     p1.ChangeDutyCycle(0)
198     p2.ChangeDutyCycle(0)
199     time.sleep(0.2)
200
201     # motor rotate
202     print("rotate back")
203     GPIO.output(22, GPIO.LOW)
204     GPIO.output(4, GPIO.HIGH)
205     p_rot.ChangeDutyCycle(80)
206     time.sleep(0.25)
207     p_rot.ChangeDutyCycle(0)
208
209 def basket2():
210     # motor forward
211     print("forward")
212     GPIO.output(6, GPIO.HIGH)
213     GPIO.output(5, GPIO.LOW)
214     GPIO.output(20, GPIO.HIGH)
215     GPIO.output(21, GPIO.LOW)
216     p1.ChangeDutyCycle(100)
217     p2.ChangeDutyCycle(100)
218     time.sleep(0.75)
219
220     # motor pause
221     p1.ChangeDutyCycle(0)
222     p2.ChangeDutyCycle(0)
223     time.sleep(0.2)
224
225     # motor back
```

```
226     print("back")
227     GPIO.output(5, GPIO.HIGH)
228     GPIO.output(6, GPIO.LOW)
229     GPIO.output(21, GPIO.HIGH)
230     GPIO.output(20, GPIO.LOW)
231     p1.ChangeDutyCycle(85)
232     p2.ChangeDutyCycle(85)
233     time.sleep(1.1)
234     p1.ChangeDutyCycle(0)
235     p2.ChangeDutyCycle(0)
236     time.sleep(0.2)
237
238 def basket3():
239
240     # motor rotate
241     print("rotate")
242     GPIO.output(22, GPIO.LOW)
243     GPIO.output(4, GPIO.HIGH)
244     p_rot.ChangeDutyCycle(80)
245     time.sleep(0.25)
246     p_rot.ChangeDutyCycle(0)
247
248     # motor forward
249     print("forward")
250     GPIO.output(6, GPIO.HIGH)
251     GPIO.output(5, GPIO.LOW)
252     GPIO.output(20, GPIO.HIGH)
253     GPIO.output(21, GPIO.LOW)
254     p1.ChangeDutyCycle(100)
255     p2.ChangeDutyCycle(100)
256     time.sleep(0.75)
257
258     # motor pause
259     p1.ChangeDutyCycle(0)
260     p2.ChangeDutyCycle(0)
261     time.sleep(0.2)
262
263     # motor back
264     print("back")
265     GPIO.output(5, GPIO.HIGH)
266     GPIO.output(6, GPIO.LOW)
267     GPIO.output(21, GPIO.HIGH)
268     GPIO.output(20, GPIO.LOW)
269     p1.ChangeDutyCycle(85)
270     p2.ChangeDutyCycle(85)
```

```
271     time.sleep(1.1)
272     p1.ChangeDutyCycle(0)
273     p2.ChangeDutyCycle(0)
274     time.sleep(0.2)
275
276     # motor rotate
277     print("rotate back")
278     GPIO.output(4, GPIO.LOW)
279     GPIO.output(22, GPIO.HIGH)
280     p_rot.ChangeDutyCycle(80)
281     time.sleep(0.25)
282     p_rot.ChangeDutyCycle(0)
283
284 def okbutton():
285     pygame.draw.circle(screen, GREEN_START, (260, 200), 25)
286     my_buttons={'OK': (260, 200)}
287     for my_text, text_pos, in my_buttons.items():
288         text_surface=my_font.render(my_text, True, WHITE)
289         rect=text_surface.get_rect(center=text_pos)
290         screen.blit(text_surface, rect)
291
292 def whitebutton(white_input):
293     if white_input==True:
294         pygame.draw.rect(screen, GREEN, pygame.Rect(20, 80, 80, 30))
295     else:
296         pygame.draw.rect(screen, RED, pygame.Rect(20, 80, 80, 30))
297
298     for my_text, text_pos, in my_buttons.items():
299         text_surface=my_font.render("Whites", True, WHITE)
300         rect=text_surface.get_rect(center=(60, 95))
301         screen.blit(text_surface, rect)
302     pygame.display.flip()
303
304 def colorbutton(color_input):
305     if color_input==True:
306         pygame.draw.rect(screen, GREEN, pygame.Rect(120, 80, 80, 30))
307     else:
308         pygame.draw.rect(screen, RED, pygame.Rect(120, 80, 80, 30))
309
310     for my_text, text_pos, in my_buttons.items():
311         text_surface=my_font.render("Colors", True, WHITE)
312         rect=text_surface.get_rect(center=(160, 95))
313         screen.blit(text_surface, rect)
314     pygame.display.flip()
315
```

```
316 def redbutton(red_input):
317     if red_input==True:
318         pygame.draw.rect(screen, GREEN, pygame.Rect(220, 80, 80, 30))
319     else:
320         pygame.draw.rect(screen, RED, pygame.Rect(220, 80, 80, 30))
321
322     for my_text, text_pos, in my_buttons.items():
323         text_surface=my_font.render("Reds", True, WHITE)
324         rect=text_surface.get_rect(center=(260, 95))
325         screen.blit(text_surface, rect)
326     pygame.display.flip()
327
328 def lightbutton(light_input):
329     if light_input==True:
330         pygame.draw.rect(screen, GREEN, pygame.Rect(70, 130, 80, 30))
331     else:
332         pygame.draw.rect(screen, RED, pygame.Rect(70, 130, 80, 30))
333
334     for my_text, text_pos, in my_buttons.items():
335         text_surface=my_font.render("Lights", True, WHITE)
336         rect=text_surface.get_rect(center=(110, 145))
337         screen.blit(text_surface, rect)
338     pygame.display.flip()
339
340 def darkbutton(dark_input):
341     if dark_input==True:
342         pygame.draw.rect(screen, GREEN, pygame.Rect(170, 130, 80, 30))
343     else:
344         pygame.draw.rect(screen, RED, pygame.Rect(170, 130, 80, 30))
345
346     for my_text, text_pos, in my_buttons.items():
347         text_surface=my_font.render("Darks", True, WHITE)
348         rect=text_surface.get_rect(center=(210, 145))
349         screen.blit(text_surface, rect)
350     pygame.display.flip()
351
352 # main code
353 while code_running:
354
355     time.sleep(0.1)
356     color_selection = True
357     start=True
358
359     screen.fill(BLACK)
360
```



```

361     # initalize screen
362     text_surface=my_font.render("Select basket below:",True,WHITE)
363     rect=text_surface.get_rect(center=(160,30))
364     screen.blit(text_surface,rect)
365
366     basket(13,47)
367     basket(105,47)
368     basket(195,47)
369
370     pygame.draw.circle(screen, GREEN_START, (260,200), 25)
371     my_buttons={'start':(260,200)}
372     for my_text,text_pos, in my_buttons.items():
373         text_surface=my_font.render(my_text,True,WHITE)
374         rect=text_surface.get_rect(center=text_pos)
375         screen.blit(text_surface,rect)
376
377
378     my_buttons={'Basket 1':(70,155), 'Basket 2':(160,155), 'Basket 3': (250,155)}
379     for my_text,text_pos, in my_buttons.items():
380         text_surface=my_font2.render(my_text,True,WHITE)
381         rect=text_surface.get_rect(center=text_pos)
382         screen.blit(text_surface,rect)
383
384     pygame.display.flip()
385
386     ##### select basket #####
387
388     for event in pygame.event.get():
389         if(event.type is MOUSEBUTTONDOWN):
390             pos=pygame.mouse.get_pos()
391             x,y=pos
392         elif(event.type is MOUSEBUTTONUP):
393
394             time.sleep(0.1)
395             while color_selection and start:
396
397                 time.sleep(0.1)
398
399                 ##### basket 1 #####
400                 if (y> 60 and y<180 and x<105):
401
402                     time.sleep(0.1)
403                     select=True
404
405                     screen.fill(BLACK)

```

```
406         okbutton()
407
408         text_surface=my_font.render("Basket 1:",True,WHITE)
409         rect=text_surface.get_rect(center=(160,40))
410         screen.blit(text_surface,rect)
411
412         pygame.display.flip()
413
414         # color selection
415         while select:
416
417             # white button
418             whitebutton(choose_white1)
419
420             # color buttton
421             colorbutton(choose_color1)
422
423             # red buttton
424             redbutton(choose_red1)
425
426             # light button
427             lightbutton(choose_light1)
428
429             # dark button
430             darkbutton(choose_dark1)
431
432         for event in pygame.event.get():
433             if(event.type is MOUSEBUTTONDOWN):
434                 pos=pygame.mouse.get_pos()
435                 m,n=pos
436                 elif(event.type is MOUSEBUTTONUP):
437
438                     if (m>20 and m<=100 and n>80 and n <=110):
439                         choose_white1 = not choose_white1
440
441                     elif (m>120 and m<=200 and n>80 and
442                             n<=110):
443                         choose_color1 = not choose_color1
444
445                     elif (m>220 and m<=300 and n>80 and
446                             n<=110):
447                         choose_red1 = not choose_red1
448
449                     elif (m>70 and m<=150 and n>130 and n<=160):
450                         choose_light1 = not
```

```
        choose_light1
```

```
    elif (m>170 and m<=250 and n>130 and n<=160):
```

```
        choose_dark1 = not choose_dark1
```

```
    # okay
```

```
    elif (m>200 and n>17):
```

```
        select = False
```

```
        color_selection=False
```

```
##### basket 2 #####
```

```
elif (y>60 and y<180 and x>=105 and x<215):
```

```
    time.sleep(0.1)
```

```
    select=True
```

```
    screen.fill(BLACK)
```

```
    okbutton()
```

```
    text_surface=my_font.render("Basket 2:", True, WHITE)
```

```
    rect=text_surface.get_rect(center=(160,40))
```

```
    screen.blit(text_surface, rect)
```

```
    pygame.display.flip()
```

```
    # color selection
```

```
    while select:
```

```
        # white button
```

```
        whitebutton(choose_white2)
```

```
        # color buttton
```

```
        colorbutton(choose_color2)
```

```
        # red buttton
```

```
        redbutton(choose_red2)
```

```
        # light button
```

```
        lightbutton(choose_light2)
```

```
        # dark button
```

```
        darkbutton(choose_dark2)
```

```
    for event in pygame.event.get():
```

```

493         if(event.type is MOUSEBUTTONDOWN):
494             pos=pygame.mouse.get_pos()
495             m,n=pos
496         elif(event.type is MOUSEBUTTONUP):
497
498             if (m>20 and m<=100 and n>80 and n <=110):
499                 choose_white2 = not choose_white2
500
501             elif (m>120 and m<=200 and n>80 and
502                  n<=110):
503                 choose_color2 = not choose_color2
504
505             elif (m>220 and m<=300 and n>80 and
506                  n<=110):
507                 choose_red2 = not choose_red2
508
509             elif (m>70 and m<=150 and n>130 and n<=160):
510                 choose_light2 = not
511                 choose_light2
512
513             elif (m>170 and m<=250 and n>130 and n<=160):
514                 choose_dark2 = not choose_dark2
515
516             # okay
517             elif (m>200 and n>170):
518                 select = False
519                 color_selection=False
520
521             ##### basket 3 #####
522             elif (y>60 and y<=180 and x>=215):
523
524                 time.sleep(0.1)
525                 select=True
526
527                 screen.fill(BLACK)
528                 okbutton()
529
530                 text_surface=my_font.render("Basket 3:", True, WHITE)
531                 rect=text_surface.get_rect(center=(160,40))
532                 screen.blit(text_surface, rect)
533
534                 pygame.display.flip()
535
536             # color selection

```

```

535         while select:
536
537             # white button
538             whitebutton(choose_white3)
539
540             # color buttton
541             colorbutton(choose_color3)
542
543             # red buttton
544             redbutton(choose_red3)
545
546             # light button
547             lightbutton(choose_light3)
548
549             # dark button
550             darkbutton(choose_dark3)
551
552         for event in pygame.event.get():
553             if(event.type is MOUSEBUTTONDOWN):
554                 pos=pygame.mouse.get_pos()
555                 m,n=pos
556             elif(event.type is MOUSEBUTTONUP):
557
558                 if (m>20 and m<=100 and n>80 and n <=110):
559                     choose_white3 = not choose_white3
560
561                 elif (m>120 and m<=200 and n>80 and
562                     n<=110):
563                     choose_color3 = not choose_color3
564
565                 elif (m>220 and m<=300 and n>80 and
566                     n<=110):
567                     choose_red3 = not choose_red3
568
569                 elif (m>70 and m<=150 and n>130 and n<=160):
570                     choose_light3 = not
571                     choose_light3
572
573                 elif (m>170 and m<=250 and n>130 and n<=160):
574                     choose_dark3 = not choose_dark3
575
576             # okay
577             elif (m>200 and n>170):
578                 select = False
579                 color_selection=False

```

```
577
578 ##### start #####
579     elif (y>180 and x>215):
580
581         time.sleep(0.1)
582
583         cap = cv2.VideoCapture(0) #video capture source camera
584
585         sort = True
586
587         CLOTHING_RED=False
588         CLOTHING_COLOR=False
589         CLOTHING_WHITE=False
590         CLOTHING_DARK=False
591         CLOTHING_LIGHT=False
592
593         # turn light on
594         p_led.start(100)
595         time.sleep(0.5)
596         print("light on")
597
598         ret,frame = cap.read()
599         print("picture taken")
600
601         # turn off light
602         time.sleep(0.5)
603         p_led.ChangeDutyCycle(0)
604
605         # convert RBG to HSV
606         hsv=cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)
607
608         # find colors
609         mask_red1=cv2.inRange(hsv, light_red1, dark_red1)
610         mask_red2=cv2.inRange(hsv, light_red2, dark_red2)
611         mask_white=cv2.inRange(hsv, light_white, dark_white)
612         mask_dark1=cv2.inRange(hsv, light_dark1, dark_dark1)
613         mask_dark2=cv2.inRange(hsv, light_dark2, dark_dark2)
614         mask_light=cv2.inRange(hsv, light_light, dark_light)
615         mask_color=cv2.inRange(hsv, light_color, dark_color)
616
617         # output
618         red1=cv2.bitwise_and(frame,frame,mask=mask_red1)
619         red2=cv2.bitwise_and(frame,frame,mask=mask_red2)
620         white=cv2.bitwise_and(frame,frame,mask=mask_white)
621         dark1=cv2.bitwise_and(frame,frame,mask=mask_dark1)
```

```

622         dark2=cv2.bitwise_and(frame,frame,mask=mask_dark2)
623         light=cv2.bitwise_and(frame,frame,mask=mask_light)
624         color=cv2.bitwise_and(frame,frame,mask=mask_color)
625
626         # Find area
627         area_red1 = 0
628         area_red2 = 0
629         area_white = 0
630         area_dark1 = 0
631         area_dark2 = 0
632         area_light = 0
633         area_color = 0
634         area_red = 0
635         area_dark = 0
636
637         # Remove noise
638         kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,3))
639         opening_red1 = cv2.morphologyEx(mask_red1, cv2.MORPH_OPEN,          ↗
        kernel, iterations=1)
640         opening_red2 = cv2.morphologyEx(mask_red2, cv2.MORPH_OPEN,          ↗
        kernel, iterations=1)
641         opening_white = cv2.morphologyEx(mask_white, cv2.MORPH_OPEN,        ↗
        kernel, iterations=1)
642         opening_dark1 = cv2.morphologyEx(mask_dark1, cv2.MORPH_OPEN,        ↗
        kernel, iterations=1)
643         opening_dark2 = cv2.morphologyEx(mask_dark2, cv2.MORPH_OPEN,        ↗
        kernel, iterations=1)
644         opening_light = cv2.morphologyEx(mask_light, cv2.MORPH_OPEN,        ↗
        kernel, iterations=1)
645         opening_color = cv2.morphologyEx(mask_color, cv2.MORPH_OPEN,        ↗
        kernel, iterations=1)
646
647         # Find contours
648         original = frame.copy()
649         cnts_red1 = cv2.findContours(opening_red1, cv2.RETR_EXTERNAL,        ↗
        cv2.CHAIN_APPROX_SIMPLE)
650         cnts_red1 = cnts_red1[0] if len(cnts_red1) == 2 else cnts_red1[1]
651         cnts_red2 = cv2.findContours(opening_red2, cv2.RETR_EXTERNAL,        ↗
        cv2.CHAIN_APPROX_SIMPLE)
652         cnts_red2 = cnts_red2[0] if len(cnts_red2) == 2 else cnts_red2[1]
653         cnts_white = cv2.findContours(opening_white, cv2.RETR_EXTERNAL,      ↗
        cv2.CHAIN_APPROX_SIMPLE)
654         cnts_white = cnts_white[0] if len(cnts_white) == 2 else            ↗
        cnts_white[1]
655         cnts_dark1 = cv2.findContours(opening_dark1, cv2.RETR_EXTERNAL,      ↗

```

```

cv2.CHAIN_APPROX_SIMPLE)
656 cnts_dark1 = cnts_dark1[0] if len(cnts_dark1) == 2 else
cnts_dark1[1]
657 cnts_dark2 = cv2.findContours(opening_dark2, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
658 cnts_dark2= cnts_dark2[0] if len(cnts_dark2) == 2 else
cnts_dark2[1]
659 cnts_light = cv2.findContours(opening_light, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
660 cnts_light = cnts_light[0] if len(cnts_light) == 2 else
cnts_light[1]
661 cnts_color = cv2.findContours(opening_color, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
662 cnts_color = cnts_color[0] if len(cnts_color) == 2 else
cnts_color[1]
663
664 for c in cnts_red1:
665     area_red1 += cv2.contourArea(c)
666     cv2.drawContours(original,[c], 0, (0,0,0), 2)
667 for c in cnts_red2:
668     area_red2 += cv2.contourArea(c)
669     cv2.drawContours(original,[c], 0, (0,0,0), 2)
670 for c in cnts_white:
671     area_white += cv2.contourArea(c)
672     cv2.drawContours(original,[c], 0, (0,0,0), 2)
673 for c in cnts_dark1:
674     area_dark1 += cv2.contourArea(c)
675     cv2.drawContours(original,[c], 0, (0,0,0),
2)
676 for c in cnts_dark2:
677     area_dark2 += cv2.contourArea(c)
678     cv2.drawContours(original,[c], 0, (0,0,0), 2)
679 for c in cnts_light:
680     area_light += cv2.contourArea(c)
681     cv2.drawContours(original,[c], 0, (0,0,0), 2)
682 for c in cnts_color:
683     area_color += cv2.contourArea(c)
684     cv2.drawContours(original,[c], 0, (0,0,0), 2)
685
686 # combine red and darks
687 area_red=area_red1+area_red2
688 area_dark=area_dark1+area_dark2
689
690 # Determine dominant color
691 if area_red>area_white and area_red>area_dark and

```



```

        area_red>area_light and area_red>area_color:
692         print('ning smells like cherries')
693         CLOTHING_RED = True
694     elif area_white>area_red and area_white>area_dark and
area_white>area_light and area_white>area_color:
695         print('ning smells like a cloud')
696         CLOTHING_WHITE = True
697     elif area_dark>area_red and area_dark>area_white and
area_dark>area_light and area_dark>area_color:
698         print('ning smells like dark chocolate')
699         CLOTHING_DARK = True
700     elif area_light>area_dark and area_light>area_red and
area_light>area_white and area_light>area_color:
701         print('ning smells like an LED')
702         CLOTHING_LIGHT = True
703     else:
704         print('ning smells like a lucky charms')
705         CLOTHING_COLOR = True
706
707
708     cv2.imshow('img1', frame)
709
710     #cv2.waitKey(0)
711     time.sleep(1)
712     cap.release()
713     cv2.destroyAllWindows()
714
715     ##### analyze color detection / code motors #####
716     while sort:
717
718         ##### red #####
719         if CLOTHING_RED==True and choose_red1==True:
720             print("basket 1: red")
721
722             basket1()
723             sort=False
724             start=False
725
726         elif CLOTHING_RED==True and choose_red2==True:
727
728             basket2()
729             sort=False
730             start=False
731
732         elif CLOTHING_RED==True and choose_red3==True:

```

```
733
734         basket3()
735         sort=False
736         start=False
737
738         ##### color #####
739         if CLOTHING_COLOR==True and choose_color1==True:
740             print("basket 1: color")
741
742         basket1()
743         sort=False
744         start=False
745
746         elif CLOTHING_COLOR==True and choose_color2==True:
747
748         basket2()
749         sort=False
750         start=False
751
752         elif CLOTHING_COLOR==True and choose_color3==True:
753
754         basket3()
755         sort=False
756         start=False
757
758         ##### WHITE #####
759         if CLOTHING_WHITE==True and choose_white1==True:
760             print("basket 1: white")
761
762         basket1()
763         sort=False
764         start=False
765
766         elif CLOTHING_WHITE==True and choose_white2==True:
767
768         basket2()
769         sort=False
770         start=False
771
772         elif CLOTHING_WHITE==True and choose_white3==True:
773
774         basket3()
775         sort=False
776         start=False
777
```

```
778 ##### dark #####
779 if CLOTHING_DARK==True and choose_dark1==True:
780     print("basket 1: dark")
781
782     basket1()
783     sort=False
784     start=False
785
786 elif CLOTHING_DARK==True and choose_dark2==True:
787
788     basket2()
789     sort=False
790     start=False
791
792 elif CLOTHING_DARK==True and choose_dark3==True:
793
794     basket3()
795     sort=False
796     start=False
797
798 ##### lights #####
799 if CLOTHING_LIGHT==True and choose_light1==True:
800
801     print("basket 1: light")
802     basket1()
803     sort=False
804     start=False
805
806 elif CLOTHING_LIGHT==True and choose_light2==True:
807
808     basket2()
809     sort=False
810     start=False
811
812 elif CLOTHING_LIGHT==True and choose_light3==True:
813
814     basket3()
815     sort=False
816     start=False
817
818
819 GPIO.cleanup()
820
821
```