

Lesson 7 Tag Recognition

1. Program Logic

AprilTag, a visual positioning marker, can quickly detect the marker and calculate the position. It's mainly applied to AR, robot and camera calibration, etc.

First, detect AprilTag through positioning, image segmentation, and contour search. Then the quadrilateral detection is performed after the contour is positioned. Connect the four corner points with a straight line to form a closed loop. Encoding and decoding the detected tags. Finally, add the corresponding execution action according to the decoding tags with different IDs.

The source code of the program is located in:

/home/pi/TonyPi/Functions/ApriltagDetect.py

```

105 # Detect apriltag
106 detector = apriltag.Detector(searchpath=apriltag._get_demo_searchpath())
107 def apriltagDetect(img):
108     gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
109     detections = detector.detect(gray, return_image=False)
110
111     if len(detections) != 0:
112         for detection in detections:
113             corners = np.int(detection.corners) # Get the four corner points
114             cv2.drawContours(img, [np.array(corners, np.int)], -1, (0, 255, 255), 2)
115
116             tag_family = str(detection.tag_family, encoding='utf-8') # Get tag_family
117             tag_id = int(detection.tag_id) # Get tag_id
118
119             object_center_x, object_center_y = int(detection.center[0]), int(detection.center[1]) # Center point
120
121             object_angle = int(math.degrees(math.atan2(corners[0][1] - corners[1][1], corners[0][0] - corners[1][0]))) # calculate the rotation angle
122
123             return tag_family, tag_id

```

2. Operation Steps



Pay attention to the text format in the input of instructions.

1) Turn on robot and connect to Raspberry Pi desktop with VNC.

2) Click  or press “Ctrl+Alt+T” to enter the LX terminal.



3) Enter “cd TonyPi/Functions/” command, and then press “Enter” to come to the category of games programmings.

```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ cd TonyPi/Functions/
```

4) Enter “sudo python3 ApriltagDetect.py”, then press “Enter” to start the game.

```
pi@raspberrypi: ~/TonyPi/Functions  
File Edit Tabs Help  
pi@raspberrypi:~ $ cd TonyPi/Functions/  
pi@raspberrypi:~/TonyPi/Functions $ sudo python3 ApriltagDetect.py
```

5) If you want to exit the game programming, press “Ctrl+C” in the LX terminal interface. If the exit fails, please try it few more times.

3. Project Outcome

i Please run this game on a solid color or a white background. Dark background such as black will affect the tag recognition performance.

After starting the tag recognition, place the tag cards in front of the camera to recognize in turns. TonyPi will execute the corresponding actions when the tad is recognized.

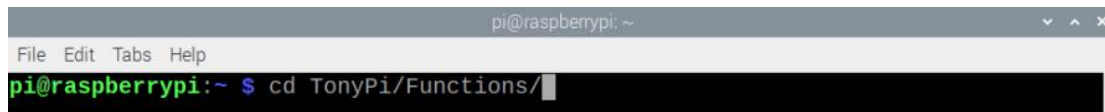
Tag ID	Action
1	Bowing
2	Mark time
3	Dancing

4. Function Extension

4.1 Modify the Action Corresponding to the Tag

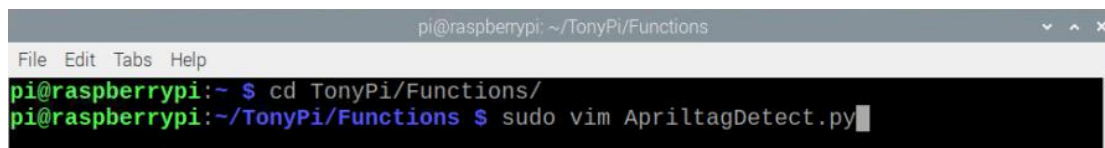
Program default setting is that TonyPi will bow when the tag ID 1 is detected. We can revise the feedback action to wave hand for example.

Step 1: Enter command “cd TonyPi/Functions/” to the directory where the game program is located.



```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ cd TonyPi/Functions/
```

Step 2: Enter command “sudo vim ApriltagDetect.py” to go into the game program through vi editor.



```
pi@raspberrypi: ~/TonyPi/Functions
File Edit Tabs Help
pi@raspberrypi:~ $ cd TonyPi/Functions/
pi@raspberrypi:~/TonyPi/Functions $ sudo vim ApriltagDetect.py
```

Step 3: Input “76” and press “shfit+g” to the line for modification.

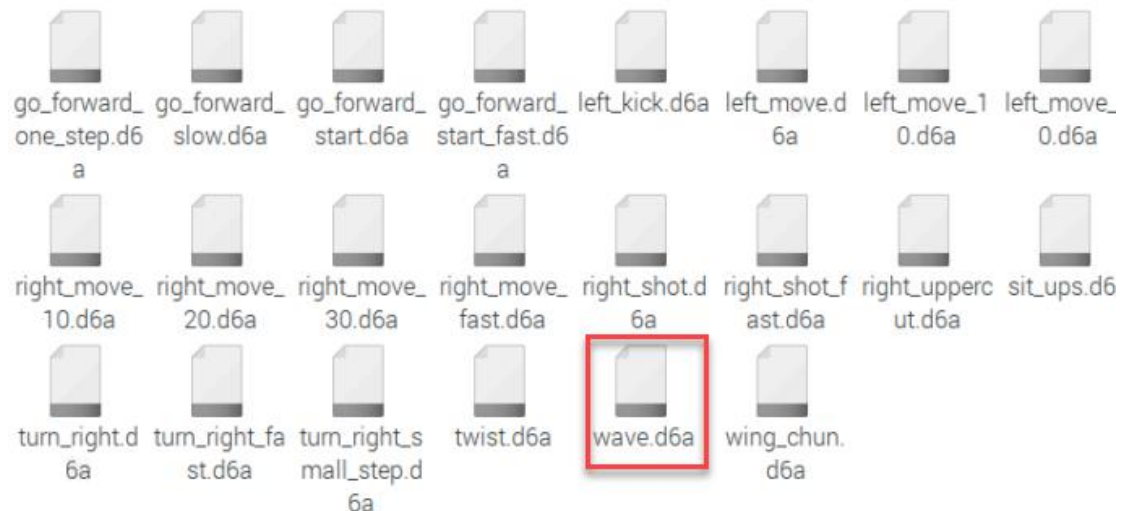
```

76         time.sleep(0.5)
77         if tag_id == 1: #标签ID为1时
78             AGC.runActionGroup('bow') #鞠躬
79             tag_id = None
80             time.sleep(1)
81             action_finish = True
82         elif tag_id == 2:
83             AGC.runActionGroup('stepping') #原地踏步
84             tag_id = None
85             time.sleep(1)
86             action_finish = True
87         elif tag_id == 3:
88             AGC.runActionGroup('twist') #扭腰
89             tag_id = None

```

Step 4: Wave.d6a is in the folder “/home/pi/TonyPi/ActionGroups” the “Wave” action group.

turn_right.d6a	Right swipe
twist.d6a	Dancing
wave.d6a	Wave



Step 5: Press “i” to enter the editing mode, then modify the ('bow') in AGC.runActionGroup('bow') to AGC.runActionGroup('wave').

Step 6: Press “Esc” to enter last line command mode. Input “:wq” to save the file and exit the editor.

```

77         if tag_id == 1: #标签ID为1时
78             AGC.runActionGroup('wave') #鞠躬
79             tag_id = None
80             time.sleep(1)
81             action_finish = True
82         elif tag_id == 2:
83             AGC.runActionGroup('stepping') #原地踏步
84             tag_id = None
85             time.sleep(1)
86             action_finish = True
87         elif tag_id == 3:
88             AGC.runActionGroup('twist') #扭腰
89             tag_id = None
90             time.sleep(1)
91             action_finish = True
92         else:
93             action_finish = True
94             time.sleep(0.01)
:wq

```

4.2 Modify or Add the Tag Recognition

The tag data is located in the "ApirlTag Tag Collection" folder under the directory of this section. (The directory needs to be unzipped first)

- ① You don't need to download materials online, please go to the directory of this section to find "ApirlTag Tag Collection" for the provided tags. (200 tags in total)
- ② There is no absolute size requirement for the tag size if you want to print your tags. It is not recommended to be too large or too small for the performance of recognition. (The tag will be circled when recognized.)
- ③ The background next to the tag will be better to keep in white. The dark background may affect the recognition.

In the following sample, we will add the ID4 as new tag. When the tag is recognized, TonyPi will run the "Cheering" action group.

- 1) Take the reference of "4.1 Modify the Action Corresponding to the Tag", enter the catalog and open the program file.

```

72         return
73     if __isRunning:
74         if tag_id is not None:
75             action_finish = False
76             time.sleep(0.5)
77             if tag_id == 1:#标签ID为1时
78                 AGC.runActionGroup('wave')#鞠躬
79                 tag_id = None
80                 time.sleep(1)
81                 action_finish = True
82             elif tag_id == 2:
83                 AGC.runActionGroup('stepping')#原地踏步
84                 tag_id = None
85                 time.sleep(1)
86                 action_finish = True
87             elif tag_id == 3:
88                 AGC.runActionGroup('twist')#扭腰
89                 tag_id = None
90                 time.sleep(1)
91                 action_finish = True
92             else:
93                 action_finish = True
94                 time.sleep(0.01)
:wq

```

2) Copy the 87-91 lines of code in the elif branch. Move the mouse cursor to the 86 line of elif, enter "5yy" (copy 5 lines) on the keyboard, you can see that the prompt "5 lines yanked" will appear below, which means the copy is successful.

```

87         elif tag_id == 3:
88             AGC.runActionGroup('twist')#扭腰
89             tag_id = None
90             time.sleep(1)
91             action_finish = True
92         else:
93             action_finish = True
94             time.sleep(0.01)
5 lines yanked
87,17 43%

```

3) Then paste these 5 lines of code, and move the mouse cursor to the position shown in the figure below:

```
87         elif tag_id == 3:
88             AGC.runActionGroup('twist')#扭腰
89             tag_id = None
90             time.sleep(1)
91             action_finish = True
92         else:
93             action_finish = True
94             time.sleep(0.01)
95     else:
96         time.sleep(0.01)
97 else:
98     time.sleep(0.01)
99
100 # 运行子线程
101 th = threading.Thread(target=move)
102 th.setDaemon(True)
103 th.start()
```

4) Enter "p" on the keyboard to paste the previously copied 5 lines of code to below:

```
87         elif tag_id == 3:
88             AGC.runActionGroup('twist')#扭腰
89             tag_id = None
90             time.sleep(1)
91             action_finish = True
92         elif tag_id == 3:
93             AGC.runActionGroup('twist')#扭腰
94             tag_id = None
95             time.sleep(1)
96             action_finish = True
```

5) Modify the copy code. Enter "i" to the editing mode and revise "tag_id" to "4", and the action in the "AGC.runActionGroup" to "chest".

The built-in action groups can be found in "/home/pi/TonyPi/ActionGroups".

```

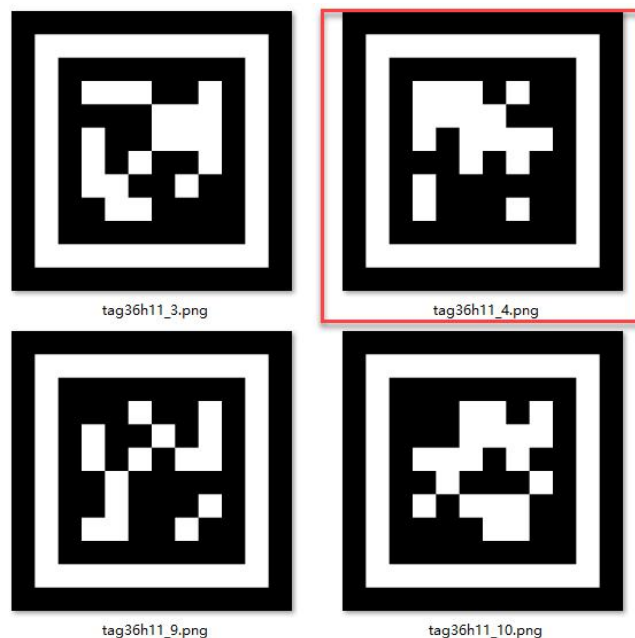
76     time.sleep(0.5)
77     if tag_id == 1: #标签 ID为1时
78         AGC.runActionGroup('bow') #鞠躬
79         tag_id = None
80         time.sleep(1)
81         action_finish = True
82     elif tag_id == 2:
83         AGC.runActionGroup('stepping') #原地踏步
84         tag_id = None
85         time.sleep(1)
86         action_finish = True
87     elif tag_id == 3:
88         AGC.runActionGroup('twist') #扭腰
89         tag_id = None
90         time.sleep(1)
91         action_finish = True
92     elif tag_id == 4:
93         AGC.runActionGroup('chest') #扭腰
94         tag_id = None
95         time.sleep(1)
96         action_finish = True
97     else:
98         action_finish = True
-- 插入 --

```

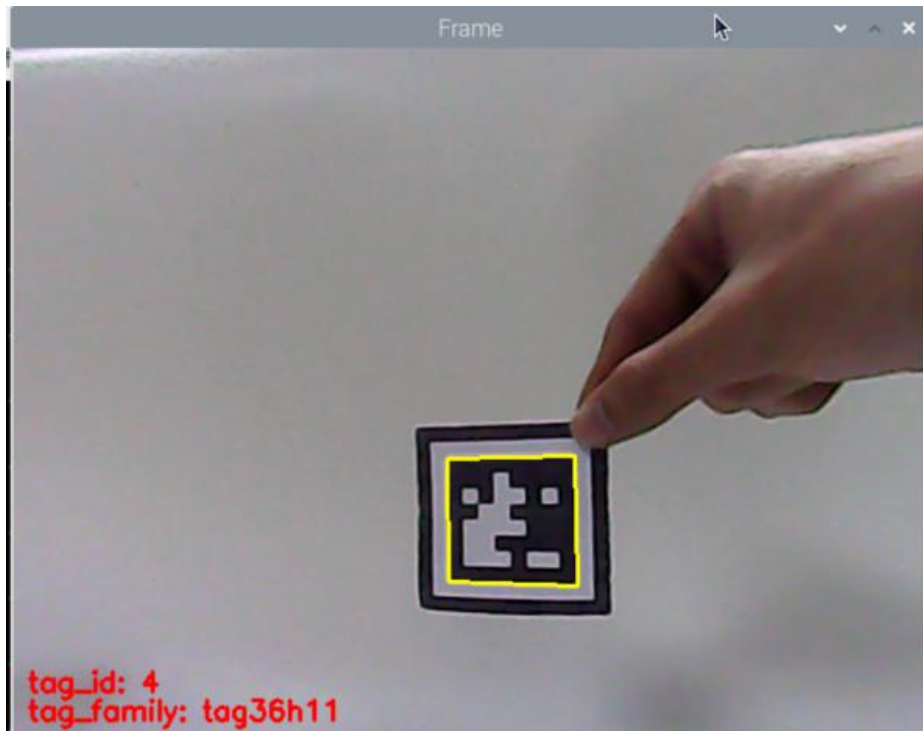
93, 46 44%

6) The modification is completed now. Press “Esc” to enter last line command mode. Input “:wq” to save the file and exit the editor.

7) Take the ID4 tag in folder “ApirlTag Tag Collection” and print it directly.



8) Check the project outcome according to the commands in previous learning.



5. Program Parameter Instruction

5.1 Tag Detection Parameter

The parameters involved in detection are as follow:

- 1) After getting the information of the four corner points from tag, `cv2.drawContours()` function is used to outline the tag, as the figure shown below:

```
110 if len(detections) != 0:
111     for detection in detections:
112         corners = np.rint(detection.corners) # 获取四个角点
113         cv2.drawContours(img, [np.array(corners, np.int)], -1, (0, 255, 255), 2)
114
```

The first parameter “img” is the input image.

The second parameter “[np.array(corners, np.int)]” is the contour itself and it is list in Python.

The third parameter “-1” is the index of the contour. The value “-1” represents all contours in the drawn contour list.

The fourth parameter “(0, 255, 255)” is the contour color and its order is B,G,R. Here it is yellow.

The fifth parameter “2” is the width of contour.

2) After the detection is completed, get the tag ID, as the figure shown below:

```
115 tag_family = str(detection.tag_family, encoding='utf-8') # 获取tag_family
116 tag_id = int(detection.tag_id) # 获取tag_id
```

“tag_id” represents the Tag ID detected.

5.2 Tag Recognition Parameter

The control parameters mainly involved in the process of tag recognition are as follow:

After detecting, cv2.putText() function is used to add text in the returned screen, as the figure shown below:

```
if tag_id is not None:
    cv2.putText(img, "tag_id: " + str(tag_id), (10, img.shape[0] - 30), cv2.FONT_HERSHEY_SIMPLEX, 0.65, [0, 255, 255], 2)
    cv2.putText(img, "tag_family: " + tag_family, (10, img.shape[0] - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.65, [0, 255, 255], 2)
```

Take code “cv2.putText(img, “tag_id: ” + str(tag_id), (10, img.shape[0] - 30), cv2.FONT_HERSHEY_SIMPLEX, 0.65, [0, 255, 255], 2)” as example:

The first parameter “img” is the input image.

The second parameter ““tag_id: ” + str(tag_id)” is the added text.

The third parameter “(10, img.shape[0] - 30)” is the coordinate of the upper left corner of the text.

The fourth parameter “cv2.FONT_HERSHEY_SIMPLEX” is the font of the added content.

The fifth parameter “0.65” is the font size.

The sixth parameter “(0, 255, 255)” is the font color and its order is B,G,R.

Here it is yellow.

The seventh parameter “2” is the font thickness.

5.3 Execute Action Parameter

According to the detected Tag ID, runActionGroup function is used to call the corresponding action group file to control the movement of robot.

```
76 if tag_id == 1:#标签ID为1时
77     AGC.runActionGroup('wave')#鞠躬
78     tag_id = None
79     time.sleep(1)
80     action_finish = True
81 elif tag_id == 2:
82     AGC.runActionGroup('stepping')#原地踏步
83     tag_id = None
84     time.sleep(1)
85     action_finish = True
86 elif tag_id == 3:
87     AGC.runActionGroup('twist')#扭腰
88     tag_id = None
89     time.sleep(1)
90     action_finish = True
```