


## Lesson 5 How to detect Human Face

In this lesson, we're going to experience the process of face detection by programming.

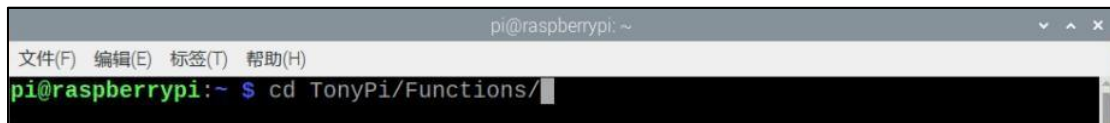
Firstly, locate the human face in screen, and then analyze its characteristics and compare it with the known human face stored in database to determine whether it is a human face.

### 1. Operation Steps

1) Turn on TonyPi Pro and connect to VNC.

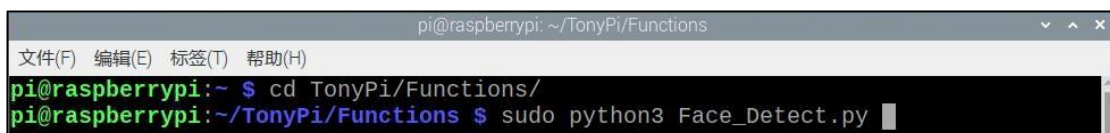
2) Press "Ctrl+Alt+T" or click  icon in the upper left corner to open LX terminal.

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



```
pi@raspberrypi: ~  
文件(F) 编辑(E) 标签(T) 帮助(H)  
pi@raspberrypi:~ $ cd TonyPi/Functions/
```

4) Enter "sudo python3 Face\_Detect.py" command, and then press "Enter" to start game.



```
pi@raspberrypi: ~/TonyPi/Functions  
文件(F) 编辑(E) 标签(T) 帮助(H)  
pi@raspberrypi:~ $ cd TonyPi/Functions/  
pi@raspberrypi:~/TonyPi/Functions $ sudo python3 Face_Detect.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.

### 2. Project Outcome

After starting game, the robot will detect whether there is a human face ahead. When the human face is detected, program will circle the recognized face in the returned image and make "Beep Beep" alarm sound.

### 3. Project Analysis

#### ◆ Image Acquisition and Processing

Before acquiring and processing the image, import the software package first:

```
import cv2  
import numpy as np
```

Image processing is the processing to number. The image is acquired after initializing and turning on the camera, and then duplicate, remap and display image.

```
img = my_camera.frame  
  
    if img is not None:  
  
        frame = img.copy()  
  
        frame = cv2.remap(frame, mapx, mapy, cv2.INTER_LINEAR) # distortion  
        correction  
  
        Frame = run(frame)  
  
        cv2.imshow('Frame', Frame)
```

#### ◆ Retrieve Face Detection Library

The face recognition library used is the Convolutional Neural Network Face Model trained by caffe, and the model is placed in the specified path, and then called by the function of OpenCV.

```
# Model Position  
  
modelFile = "/home/pi/TonyPi/models/res10_300x300_ssd_iter_140000_fp16.caffemodel"  
  
configFile = "/home/pi/TonyPi/models/deploy.prototxt"  
  
net = cv2.dnn.readNetFromCaffe(configFile, modelFile)
```

#### ◆ Face Recognition Judgement

Next, define a variable “conf\_threshold” to store the threshold value, and then calculate the currently detected human face and the similarity that compares the detected face with all user’ facial features in face recognition library to get the “confidence”. Finally, this value is compared with “conf\_threshold” to determine

the next action of the robot.

```
# Threshold

conf_threshold = 0.6

blob = cv2.dnn.blobFromImage(img_copy, 1, (150, 150), [104, 117, 123], False, False)

net.setInput(blob)

detections = net.forward() #calculate recognition

for i in range(detections.shape[2]):

    confidence = detections[0, 0, i, 2]

    if confidence > conf_threshold:
```

### ◆ Face Coordinate Conversion and Alarm

Finally, when “confidence” value is greater than the threshold, we convert the face coordinates to the coordinates before unscaling to frame the face, and set buzzer to sound once as feedback.

```
x1 = int(detections[0, 0, i, 3] * img_w)

y1 = int(detections[0, 0, i, 4] * img_h)

x2 = int(detections[0, 0, i, 5] * img_w)

y2 = int(detections[0, 0, i, 6] * img_h)

cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2, 8) #Frame the recognized face

Board.setBuzzer(0) # Close

Board.setBuzzer(1) # Open

time.sleep(0.3) # Delay

Board.setBuzzer(0) #Close
```