



Interfacing laboratory

Report 4

Raspberry PI

Yara Al-Shafei

1120068

Teacher Assistant: Mohammed mudlal

Instructor: Dr Wasel Ghanem

Abstract

Nowadays, the world tending to make devices size smaller and smaller. Raspberry is one of these devices which made a huge revolution in technology.

In this report will be divided into four sections, an introduction about the raspberry PI, a procure about how we programmed Raspberry PI and connected it with different components, and a conclusion about the experiments.

Table of contents

1. Introduction1

2. Procedure and Discussion3

3. Conclusion12

4. References13

1. Introduction

1.1 Raspberry PI (board, language, pins)

Raspberry PI is a board provides functionalities let it work as a computer. It contains memory and CPU as usual computers, it can be connected with a screen and a mouse to have a full computer. Also, many components connected with it like camera.

This board allows people to build many projects starting from simple projects to complicated ones.

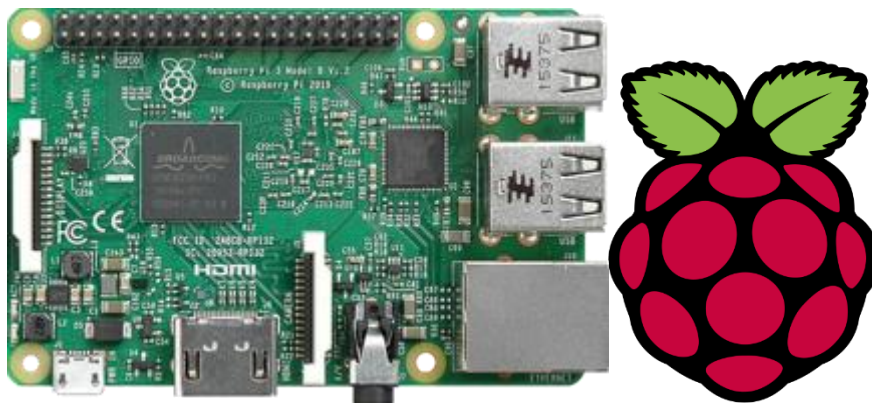


Figure 1.1.1 Raspberry board

Raspberry is programmable device, it works on Linux operating system and it is programmed using python programming language.

Raspberry PI board has general input/output pins on it is edge and these pins are giving this board a powerful feature. Beside these pins there is pins used to connect Raspberry with other components and devices such as the USB port which used to connect it with a computer.



Figure 1.1.2 Raspberry PI pins^[2]

1.2 Raspberry PI camera

Raspberry PI connected with a camera allows the user to take pictures, record videos and apply effects on these pictures. This module can work with all Raspberry models,

and it has a library to access it called Plicamera. This camera is widely used in home secure application.

1.3 PIR sensor

Motion detection sensor or PIR sensor is used to detect the motion of humans or any moving object. It contains lens to detect the inferred (IR) radiation in a specific distance. So any change in the IR level means that there is a motion. We can connect it with other components like LEDs, for example if the motion is detected then we can light a LED. This sensor is a complicated sensor if we compare it to other sensors.

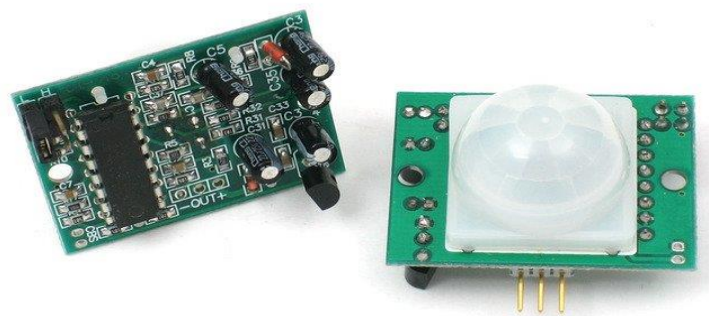


Figure 1.3.1 PIR sensor^[5]

1.4 Gas sensor

gas sensor is used to detect the gas emotions. It has four pins to connect it with a microcontroller like Arduino. It can be used in fire systems to give an alarm when there is fire.



Figure 1.4.1 Gas sensor

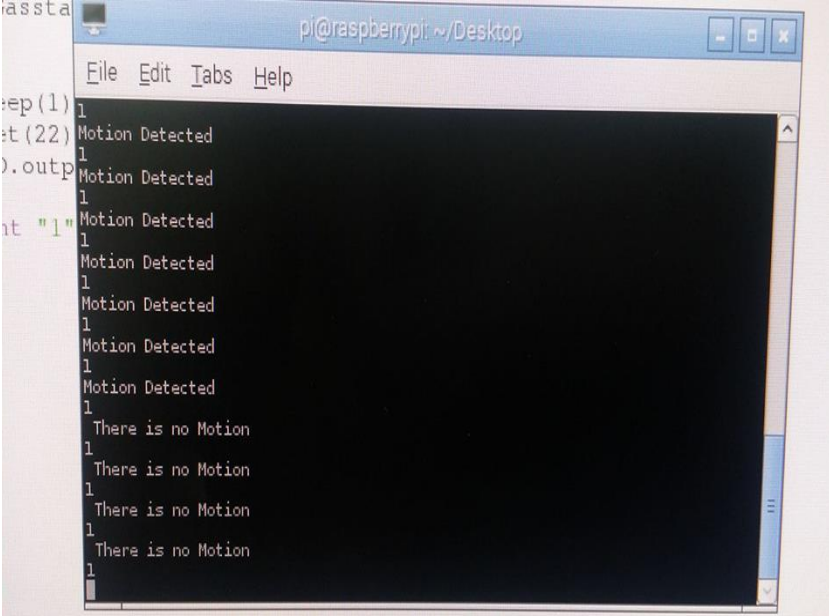
2. Procedure and Discussion

2.1 PIR Motion Detector

We started working with Raspberry PI by connecting it with PIR sensor. In this part we will use this sensor to detect the motion, based on this motion we will light a LED connected with the Raspberry PI too.

We connected the PIR with pin #23 in the Raspberry PI and printed on the screen if there is motion or not, just to know if the sensor working or not.

The following screen shot shows the output printed:



```
pi@raspberrypi: ~/Desktop
File Edit Tabs Help
1
Motion Detected
1
Motion Detected
1
Motion Detected
1
Motion Detected
1
Motion Detected
1
Motion Detected
1
Motion Detected
1
Motion Detected
1
There is no Motion
1
There is no Motion
1
There is no Motion
1
There is no Motion
1
1
```

Figure 2.1.1 output on a screen

After that, we connected the LED with them. The code we uploaded on the Raspberry PI was given in the experiment, here is the code:

```

1 import time
2 import RPi.GPIO as GPIO #import GPIO library
3 GPIO.setmode(GPIO.BOARD) #configuration GPIO pins
4 GPIO.setup(23, GPIO.IN) #define pin #23 as input pin
5 GPIO.setup(11, GPIO.out) #define pin #11 as output pin
6
7 while True:
8     if (GPIO.input(23) == True): #true means high
9         print " Motion Detected "
10        GPIO.output(11,1) #turn on led 1 is high
11    else:
12        print " There is No Motion "
13        GPIO.output(11,0) #turn of led 0 is low
14        time.sleep(1) #delay for one second
15

```

The code shows that the PIR sensor is connected with pin #23 and the LED is connected with the pin #11. It reads from pin #23 if there is motion or not and based on this it decides to light the LED or not.

2.2 PIR with led and LDR:

In this part we added LDR to the previous part connection as in the next picture:

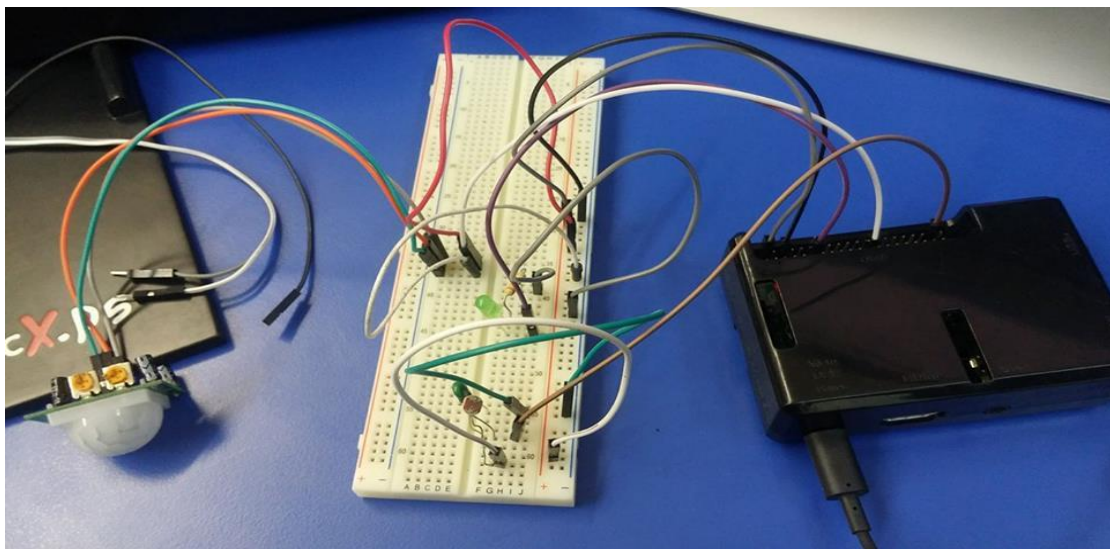


Figure 2.2.1 The connection

Here, we take the advantage of that the capacitor is charging and discharging and the resistance to let them work like ADC. Since the Raspberry PI cannot read analog values like this we can make it read.

Also this part code was given in the experiment. The following screen shot is for the code:


```

1  import time
2  import RPi.GPIO as GPIO
3  import os
4  DEBUG = 1
5
6  GPIO.setmode(GPIO.BOARD)
7  GPIO.setup(11, GPIO.OUT)
8
9  def Rctime (RCpin):
10     reading = 0
11     GPIO.setup(RCpin, GPIO.OUT)
12     GPIO.output(RCpin, GPIO.LOW)
13     time.sleep(0.1)
14
15     GPIO.setup(RCpin, GPIO.IN)
16     # This takes about 1 millisecond per loop cycle
17     while (GPIO.input(RCpin) == GPIO.LOW):
18         reading += 1
19     return reading
20
21  def motion (mpin) :
22     GPIO.setup(mpin, GPIO.IN)
23     motionstate = False
24     motionstate = GPIO.input (mpin)
25     return motionstate
26
27
28  while True:
29     if (motion(23) == True and Rctime(38)>20):
30         # print " Motion Detected " +"Light Off "
31         GPIO.output(11,True)
32         time.sleep(2)
33     elif motion(23) == True and Rctime(38)<20:
34         # print " Motion Detected "+"Light On"
35         GPIO.output(11,False)
36
37     elif motion(23) == False and Rctime(38)>20:
38         # print " There is No Motion "+"Light Off"
39         GPIO.output(11,False)
40
41     time.sleep(0.1)
42

```

The Rctime function is used to calculate the time for charging. We check it's value in which range, beside checking if there is motion or not. Based on the readings we decide to turn the LED on or not.

2.3 Gas Sensor

In this part we will connect the Raspberry PI with different sensor which is the Gas sensor. We connected it with pin #22. The connection was as in the following picture:

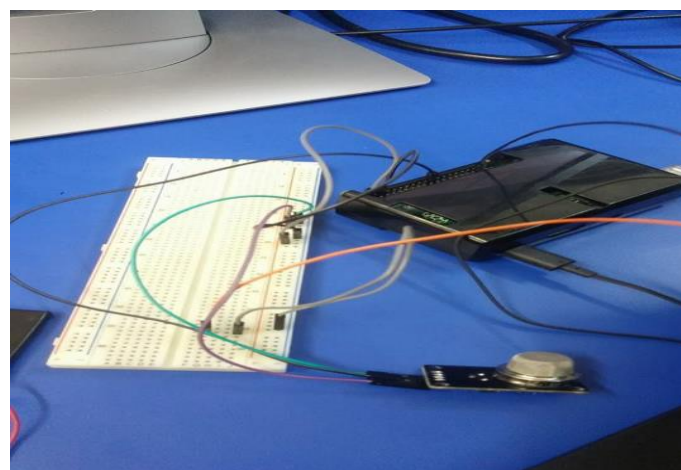
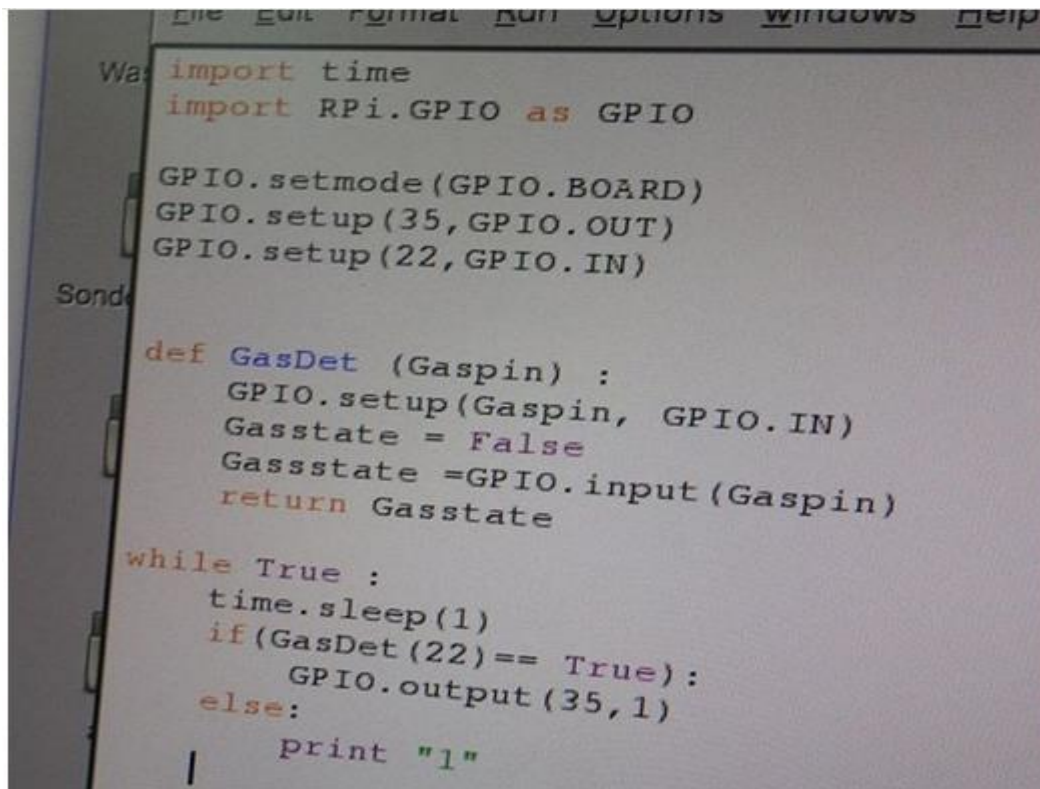


Figure 2.3.1 The connection

The code was given but we inserted the pin number we connected the gas sensor with. The next screen shot is for the code:


```
1 import time
2 import RPi.GPIO as GPIO
3
4
5 GPIO.setmode(GPIO.BOARD)
6 GPIO.setup( , GPIO.OUT)
7
8 def GasDet (Gaspin) :
9     GPIO.setup(Gaspin, GPIO.IN)
10    Gasstate = False
11    Gasstate = GPIO.input(Gaspin)
12    return Gasstate
13
14 while True:
15
16     time.sleep(1)
17
```

After that, we connected the Gas sensor with a buzzer. The buzzer is connected on pin #35, if the gas is detected the buzzer will give a sound. The following is the code for doing this:



```
File Edit Format Run Options Windows Help
Wa: import time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
GPIO.setup(35, GPIO.OUT)
GPIO.setup(22, GPIO.IN)

Sonde
def GasDet (Gaspin) :
    GPIO.setup(Gaspin, GPIO.IN)
    Gasstate = False
    Gasstate =GPIO.input(Gaspin)
    return Gasstate

while True :
    time.sleep(1)
    if(GasDet(22)== True):
        GPIO.output(35, 1)
    else:
        print "1"
```

2.4 Running the two programs together

In the last part of the first experiment, we connected the PIR with Gas sensor. The connection was as in the next picture:

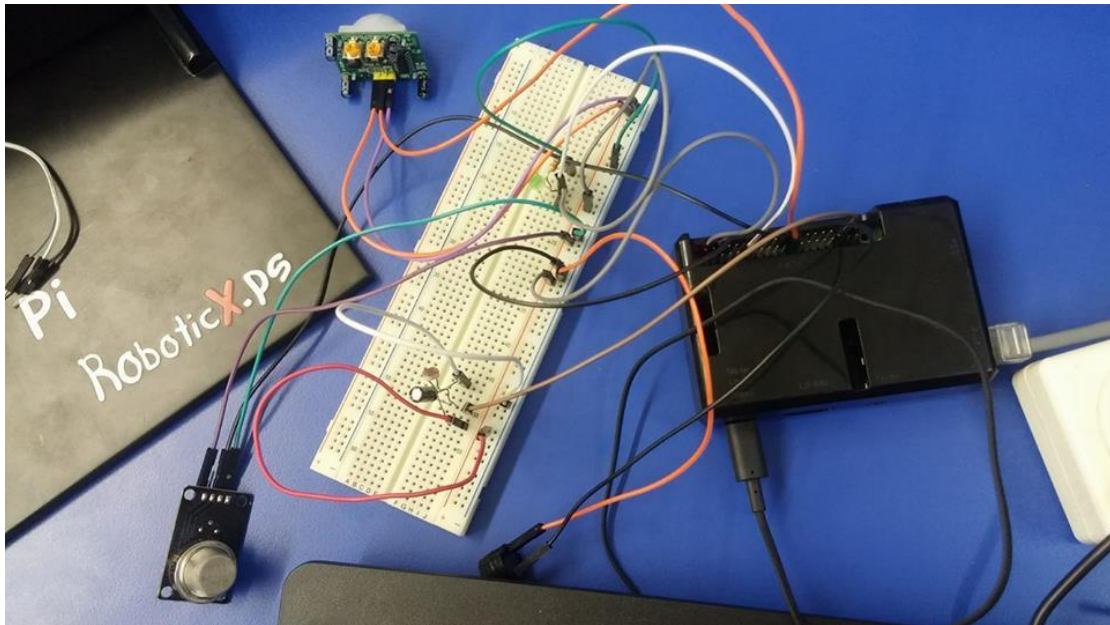


Figure 2.4.1 The connection

We ran the two codes by writing commands given in the experiment.

```
Sudo python motion.py&
```

```
Sudo python gas.py&
```

2.5 Smart camera

First, we connected the camera with Raspberry PI and connected it to the computer. Then we connected the PIR sensor with the Raspberry PI as in the picture:

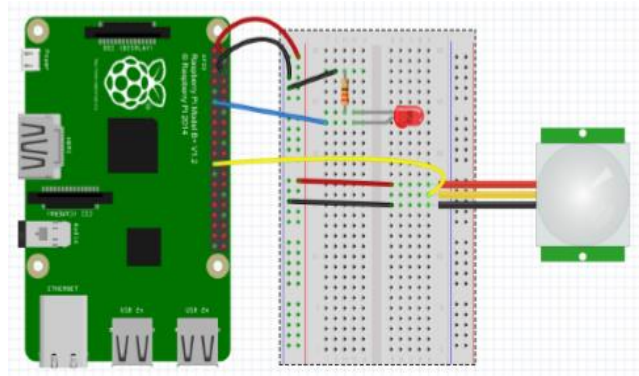


Figure 2.5 connection

After that, we wrote the following code:

```

1 import time
2 import RPi.GPIO as GPIO
3 import os
4 GPIO.setmode(GPIO.BOARD)
5 GPIO.setup(23,GPIO.IN)
6
7
8 while True:
9     if(GPIO.input == True):
10        os.system("fswebcam -d /dev/video0 -r 640x480 pic.jpeg")
11    else:
12
13        time.sleep(1)

```

This code will let the camera take pictures when the motion is detected.

2.6 TimeLapse Camera- Making timelapse

Here, we will use the timelapse technique, this technique is used in photography to show an object in fast mode by taking many frames for a while of time but showing them in a short time. In this part we used the camera connected with the Raspberry PI to make this mode.

We started by uploading the code given in the experiment to Raspberry PI. The code will count the number of frames the camera takes until they reach 30 frame, it takes a frame each 30 seconds. The code is:

```

1 import os
2 import time
3 import sys
4
5 FRAMES = 30 #is the number of the wanted pictures in 5 minutes which is 5*6=30
6
7
8 frameCount = 0
9 while frameCount < FRAMES:
10    imageNumber = str(frameCount).zfill(4)
11    #str: is for converting numbers to a string and zfill is to make it 4 digits.
12
13    os.system("fswebcam -d /dev/video0 -r 640x480 /home/pi/New/image%s.jpg"%(imageNumber))
14    #os.system to execute the command between branches.
15
16    frameCount += 1
17    time.sleep(10) #Takes roughly 10 seconds to take a picture
18 sys.exit()

```

Before making the timelapse of the captured frames, we created a text file to save the frames in it. We made a video using the open source program MEncoder using the captured frames. Now, we start to do the timelapse from the video using the following command:

```

mencoder -nosound -ovc lavc -lavcopts vcodec=mpeg4:aspect=16/9:vbitrate=8000000 -vf
scale=640:480 -o timelapse.avi -mf type=jpeg:fps=24 mf://@list.txt

```

Finally, we ran the video using this command:

Omxplayer videoName.avi

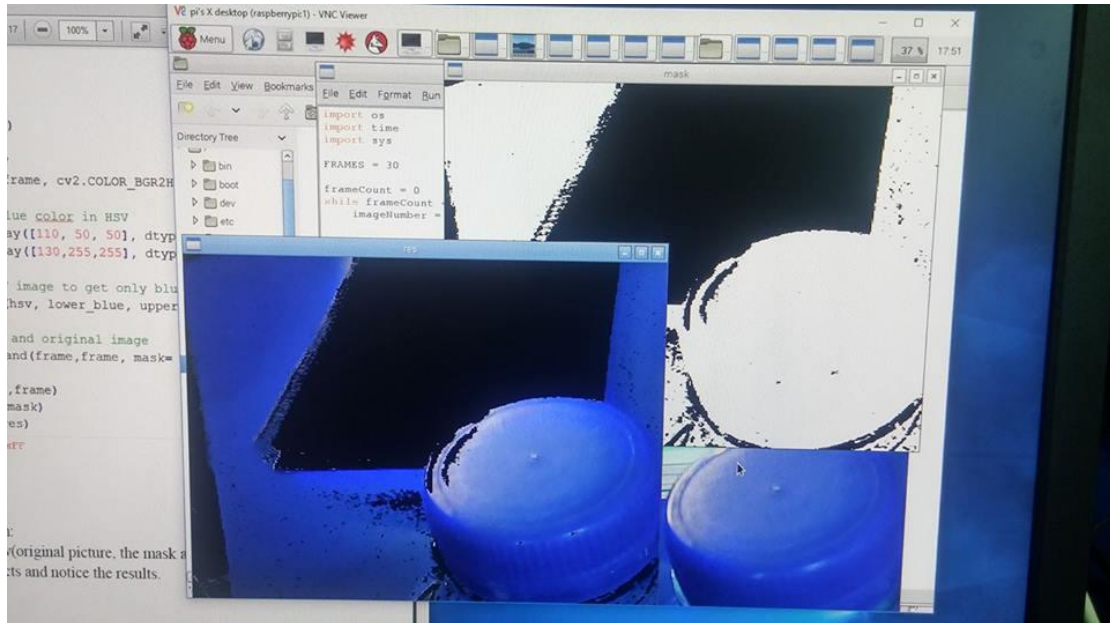
2.7 Color Detection

In this part we will detect the blue color using the OpenCV library. The blue color has a range from (50, 50, 110) to(130, 255, 255).

We started by writing the code given. In the code, the frame will be checked if it contains a color its range within the blue range.

```
1  import cv2
2  import numpy as np
3
4  cap = cv2.VideoCapture(0)
5
6  while(1):
7
8      # Take each frame
9      _, frame = cap.read()
10
11     # Convert BGR to HSV
12     hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
13
14     # define range of blue color in HSV
15     lower_blue = np.array([110, 50, 50], dtype=np.uint8)
16     upper_blue = np.array([130,255,255], dtype=np.uint8)
17
18     # Threshold the HSV image to get only blue colors
19     mask = cv2.inRange(hsv, lower_blue, upper_blue)
20
21     # Bitwise-AND mask and original image
22     res = cv2.bitwise_and(frame,frame, mask= mask)
23
24     cv2.imshow('frame',frame)
25     cv2.imshow('mask',mask)
26     cv2.imshow('res',res)
27
28     k = cv2.waitKey(5) & 0xFF
29     if k == 27:
30         break
```

When running the code, we got three pictures as follow:



2.6.1 output

One of this picture is the original picture, the second is the masked from the code and the last one is the result which the blue color is detected.

2.8 Face detection

In this part we ran the face detection program which is already defined in Raspberry and OpenCV. We used the given commands in the experiment to run it:

```
cd /home/pi/face
```

```
python facedetect.py -cascade=face.xml 0
```

A window opened when we ran these commands, if a face come in front of the camera it can detected it. The next picture is for the assistance when the camera detect his face:

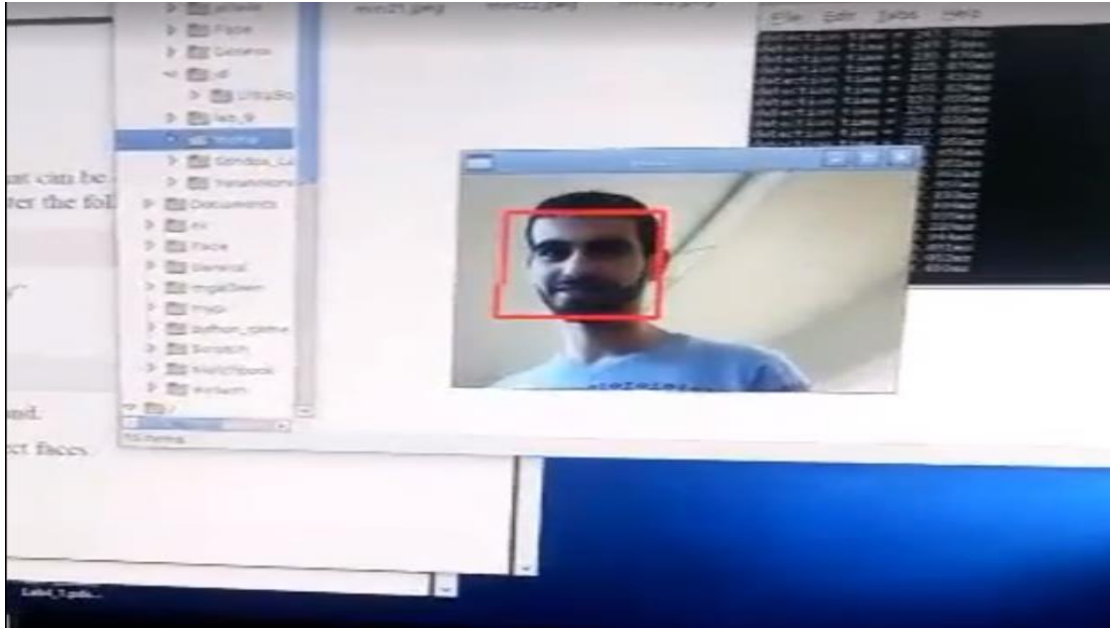


Figure 2.8.1 face detection

2.9 Arduino with UltraSonic

We worked with Arduino in the previous experiments. Now we will work with both Raspberry PI and Arduino. An ultrasonic sensor was connected with Arduino as follow:

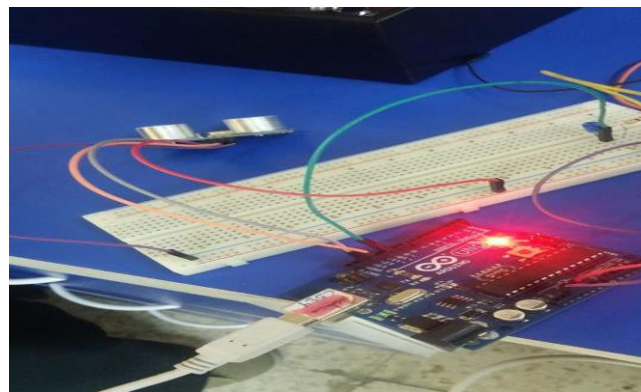
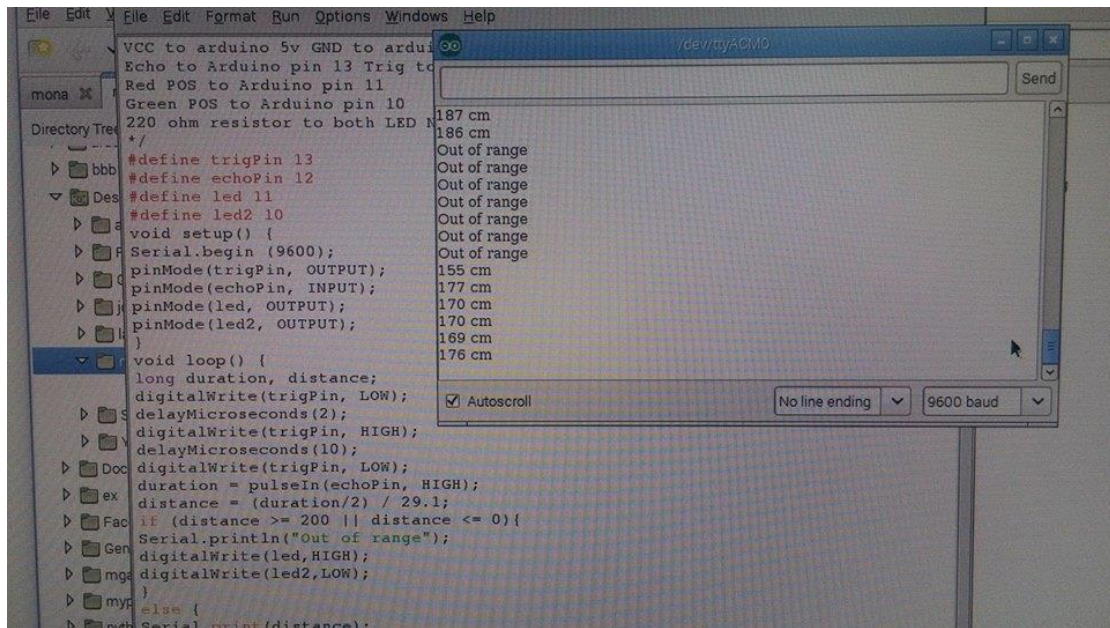


Figure 2.9.1 connection

Then we wrote the code given in the experiment and run it. We open the serial and we got this output:



3. Conclusion

Being familiar with devices such as Raspberry PI is something very important because these devices make building projects something very easy and can be done in any place. Because of their low prices and they are open source devices.

The experiments were satisfying since they allow us to program the Raspberry PI and connected with different components.

4. References

- [1] "Raspberry Pi." Wikipedia. Accessed December 15, 2016. https://en.wikipedia.org/wiki/Raspberry_Pi.
- [2] "GPIO: Raspberry Pi Models A and B." GPIO: Raspberry Pi Models A and B - Raspberry Pi Documentation. Accessed December 15, 2016. <https://www.raspberrypi.org/documentation/usage/gpio/>.
- [3] @Raspberry_Pi. "Camera Module - Raspberry Pi." Raspberry Pi. Accessed December 15, 2016. <https://www.raspberrypi.org/products/camera-module/>.
- [4] "Getting Started with Picamera." Getting Started with Picamera | Raspberry Pi Learning Resources. Accessed December 15, 2016. <https://www.raspberrypi.org/learning/getting-started-with-picamera/>.
- [5] "PIR Motion Sensor." Overview | PIR Motion Sensor | Adafruit Learning System. Accessed December 15, 2016. <https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview>.