بسم الله الرحمن الرحيم

Faculty of Engineering & Information Technology

Interfacing Laboratory - ENCS 412

Experiments 9+10 Report

Raspberry PI

Instructor: Dr. Ahmad Afaneh

Teacher Assistant: Eng. Mohammed Modallal

Name: Hussein Dahir, **ID:** 1131138

Partner's Name: Yazeed Obaid, **ID:** 1130036

Partner's Name: Maher Saleem, **ID:** 1130258

Partner's Name: Majd Bassoumi, **ID:** 1130018

Section: 2

Date: 27 April 2017

Abstract:

In these two experiments, Raspberry PI was introduced alongside with some useful sensors which are the motion sensor, the gas sensor and the ultrasonic sensor. Many applications were applied using these components, which made the students more familiar with the usage of Raspberry PI, and understand its efficiency, especially with its small size. It can be used for many projects and ideas, with no high costs.

Contents

Theory 1
  What is Raspberry Pi? 1
  Raspberry Pi Board 1
  GPIO PINs 1
  PIR sensor 2
  Gas sensor 3
  Raspberry PI Camera 3
Procedure & Discussion 4
  Experiment 9 Tasks 4
    Running the Two Programs Together 4
  Experiment 10 Tasks 9
    Testing Camera 9
    Smart camera 9
    Time Lapse Camera 11
    Color Detection 12
    Face Detection 13
    Arduino with Ultrasonic 13
References 17

Theory [1+2]:

What is Raspberry Pi?

The Raspberry Pi is a computer, very like the computers with which you're already familiar. It uses a different kind of processor you can install several versions of the Linux operating system that look and feel very much like Windows. If you want to, you can use the Raspberry Pi to surf the internet, send an email or write a letter using a word processor. But you can also do so much more.

This little board here is low cost, it's easily accessible, it's very simple to use. When you power it up you get a nice little desktop environment, it includes all of the things that you need to do to get started to learn programming. There's lots of information on the internet that you can take away and start programming code in to make things happen.

The great thing about these boards as well is in addition to software, you can play with hardware. So these little general purpose pins here allow access to the processor and you can hang off little hardware projects that you build and you can control via the code you are writing through the software application. So, this is a great tool for kids to learn how computers work at a grassroots level.

Raspberry Pi Board:

In this experiment, we will use the most recent board which is the Raspberry Pi 2 model B. Figure 1 below shows Raspberry Pi 2 model B board components.

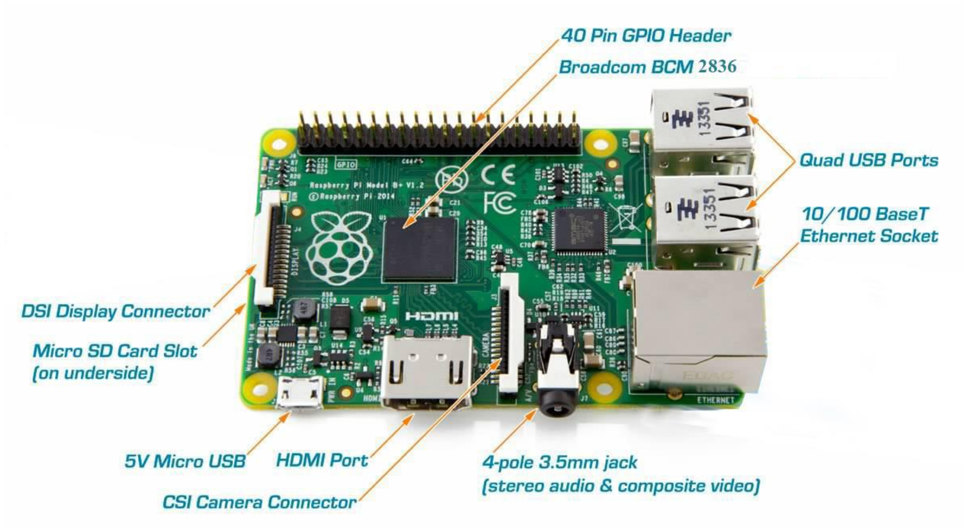

40 Pin GPIO Header
Broadcom BCM 2836
Quad USB Ports
10/100 BaseT Ethernet Socket
DSI Display Connector
Micro SD Card Slot (on underside)
5V Micro USB
HDMI Port
CSI Camera Connector
4-pole 3.5mm jack (stereo audio & composite video)

Figure 1: Raspberry Pi 2 board components.

GPIO PINs:

One powerful feature of the Raspberry Pi is the row of GPIO (general purpose input/output) pins along the edge of the board, next to the video out socket.

These pins are a physical interface between the Raspberry Pi and the outside world. At the simplest level, you can think of them as switches that you can turn on or off (input) or that the Raspberry Pi can turn on or off (output). 26 of the 40 pins are GPIO pins; the others are power or ground pins.

GPIO voltage levels are 3.3v and are not 5v tolerant. There is no over-voltage protection on the board. A voltage near 3.3 V is interpreted as a logic one while a voltage near zero volts is a logic zero.

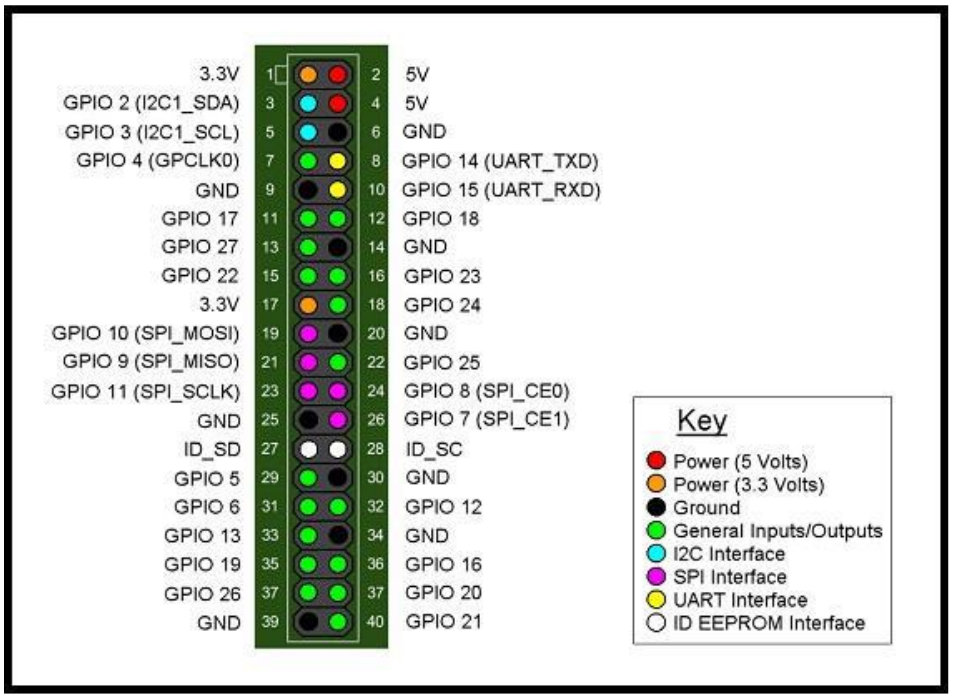


Figure 2: Raspberry Pi 2 GPIO pins

PIR sensor:

PIR sensors allow you to sense motion, almost always used to detect whether a human has moved in or out of the sensors range. They are small, inexpensive, low-power, easy to use and don't wear out. For that reason, they are commonly found in appliances and gadgets used in homes or businesses. They are often referred to as PIR, "Passive Infrared", "Pyroelectric", or "IR motion sensors". PIRs are basically made of a pyroelectric sensor, which can detect levels of infrared radiation. Everything emits some low-level radiation, and the hotter something is, the more radiation is emitted. The sensor in a motion detector is actually split in two halves.

The reason for that is that we are looking to detect motion (change) not average IR levels. The two halves are wired up so that they cancel each other out. If one half sees more or less IR radiation than the other, the output will swing high or low.

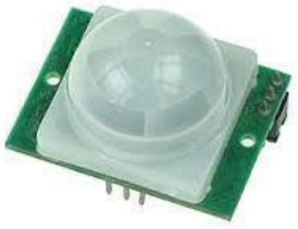


Figure 3: PIR (motion detector) sensor

Gas sensor:

This is a simple-to-use liquefied petroleum gas (LPG) sensor, suitable for sensing LPG (composed of mostly propane and butane) concentrations in the air. The MQ-6 can detect gas concentrations anywhere from 200 to 10000ppm.

This sensor has a high sensitivity and fast response time. The sensor's output is an analog resistance. The drive circuit is very simple; all you need to do is power the heater coil with 5V, add a load resistance, and connect the output to an ADC.

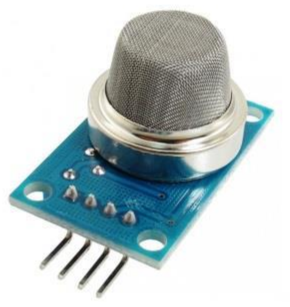


Figure 4: Gas sensor

Raspberry PI Camera:

Since 2012, the Raspberry Pi Foundation had been reporting that an official camera module was in development. In May 2013, an announcement was made by RS Components and Premier Farnell/Element 14, distribution partners of Raspberry Pi, that the camera module was available. The camera consists of a small (25mm by 20mm by 9mm) circuit board, which connects to the Raspberry Pi's Camera Serial Interface (CSI) bus connector via a flexible ribbon cable. The camera's image sensor has a native resolution of five megapixels and has a fixed focus lens. The software for the camera supports full resolution still images up to 2592x1944 and video resolutions of 1080p30, 720p60 and 640x480p60/90. The camera module is shown below:

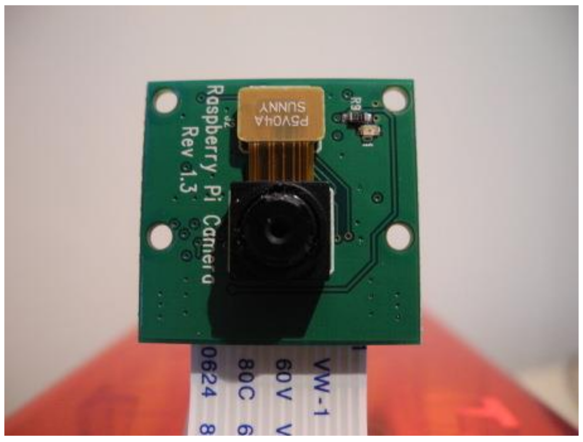


Figure 5: Raspberry PI Camera

Procedure & Discussion:

Experiment 9 Tasks:

Running the Two Programs Together:

In this part, we will run the programs in the previous two parts together, that is the motion detection with LDR and the gas detection with a LED.

To do so, first connect the following circuit:

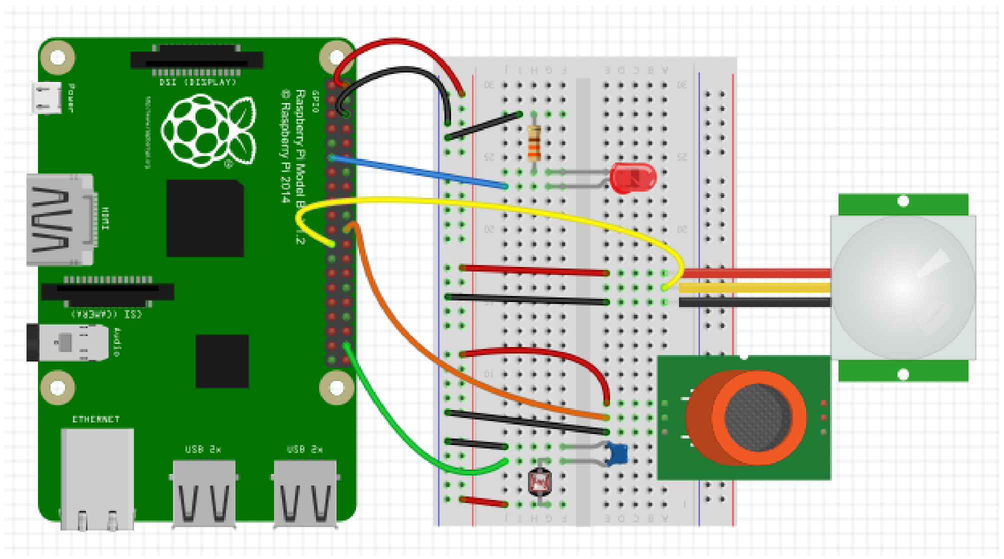


Figure 6: Circuit connection of all sensors and the LED with Raspberry PI

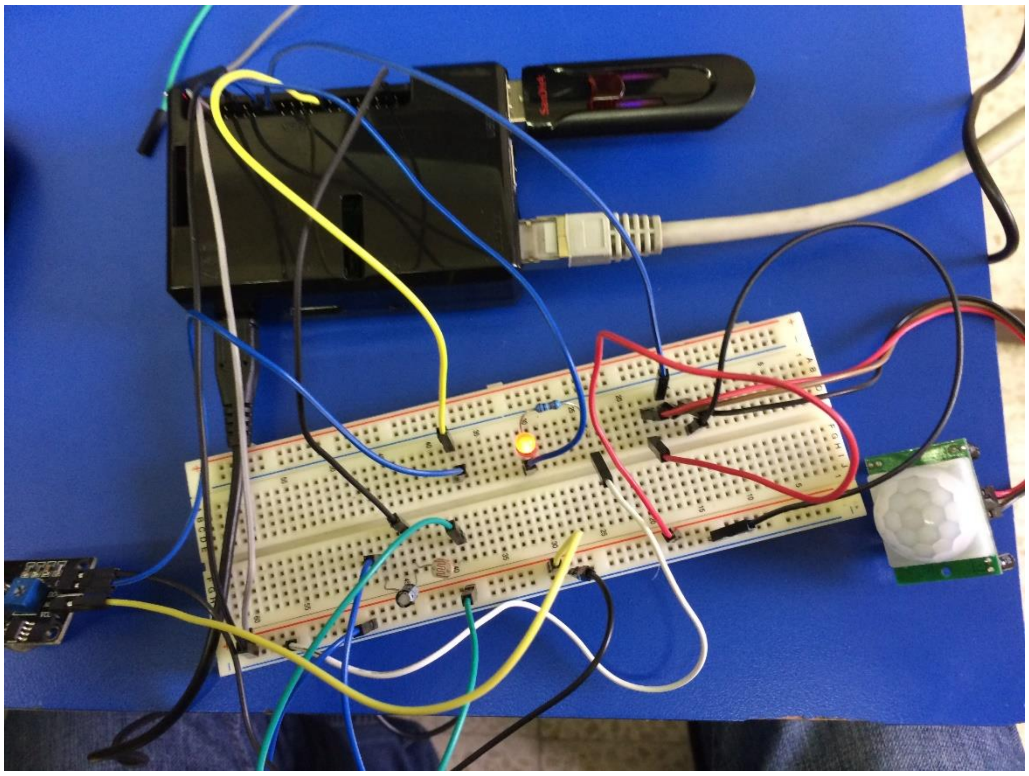


Figure 7: Circuit connection of all sensors and LED with Raspberry PI in lab

Now to run both programs together we just need to run the following two commands:

Sudo python motion.py&

Sudo python gas.py&

These two commands will run each program in the background, and so they both run simultaneously. Now, let's see how each program works.

1- PIR with LED and LDR:

Since raspberry pi can't read analog signal and because LDR only gives analog, output we need to find a way to connect the LDR to it without using ADC. The way we do this is by taking advantage of a basic electronic property of resistors and capacitors. It turns out that if you take a capacitor that is initially storing no voltage, and then connect it to power (like 3.3V) through a resistor, it will charge up to the power voltage slowly. The bigger the resistor, the slower it is.

To be clearer, we can explain this as follows:

$\tau = RC$

If R is low, τ is low, and so the capacitor charging process is fast, and when R is high, τ is high, and so the process is slow. So, we may depend on this idea to know if the LDR resistance is high or not, and so, to know if there is a light or not (high LDR resistance means there is no light, and low resistance means that there is). If the capacitor is charged, that is its voltage is read as HIGH from the Raspberry PI in less than a specific number of loops (a threshold, which equal to 20 in our experiment), then the process is fast, and so LDR resistance is low, so there is light. If more than “threshold” loops has elapsed and the read capacitor voltage still read as LOW from the Raspberry PI, then the charging process is slow, LDR resistance is high, and so there is no light.

The following code was used for this:

```
import time
import RPi.GPIO as GPIO
import os
DEBUG = 1
THRESHOLD = 20                                  # threshold to decide if light on or off
                                                # if reading > THRESHOLD, then light off,
since LDR will be
                                                # large, and so it will take time for
capacitor to charge
                                                # that is we use the reading as some kind of
timer
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11,GPIO.OUT)
GPIO.setwarnings(False)

def RCtime (RCpin) :
      reading = 0
      GPIO.setup(RCpin,GPIO.OUT)
      GPIO.output(RCpin, GPIO.LOW)
      time.sleep(0.1)

      GPIO.setup(RCpin,GPIO.IN)
      while(GPIO.input(RCpin) == GPIO.LOW):
            reading +=1
      return reading

def motion (mpin) :
      GPIO.setup(mpin,GPIO.IN)
      motionstate = False
      motionstate = GPIO.input(mpin)
      return motionstate

while True:
      if(motion(13) == True and RCtime(38) > THRESHOLD):
                 print "Motion Detected, Light off"
            GPIO.output(11,True)
            time.sleep(2)
      elif (motion(13) == True and RCtime(38) < THRESHOLD):
                 print "Motion Detected, Light on"
            GPIO.output(11,False)
      elif (motion(13) == False and RCtime(38) > THRESHOLD):
                 print "No Motion Detected, Light off"
            GPIO.output(11,False)
      time.sleep(0.1)
```

2- Gas Sensor:

The second program will be to run a gas sensor, the object of this program is turn on a fan (or buzzer) if the gas sensor detect gas.

Its code is very simple, which was as follows:

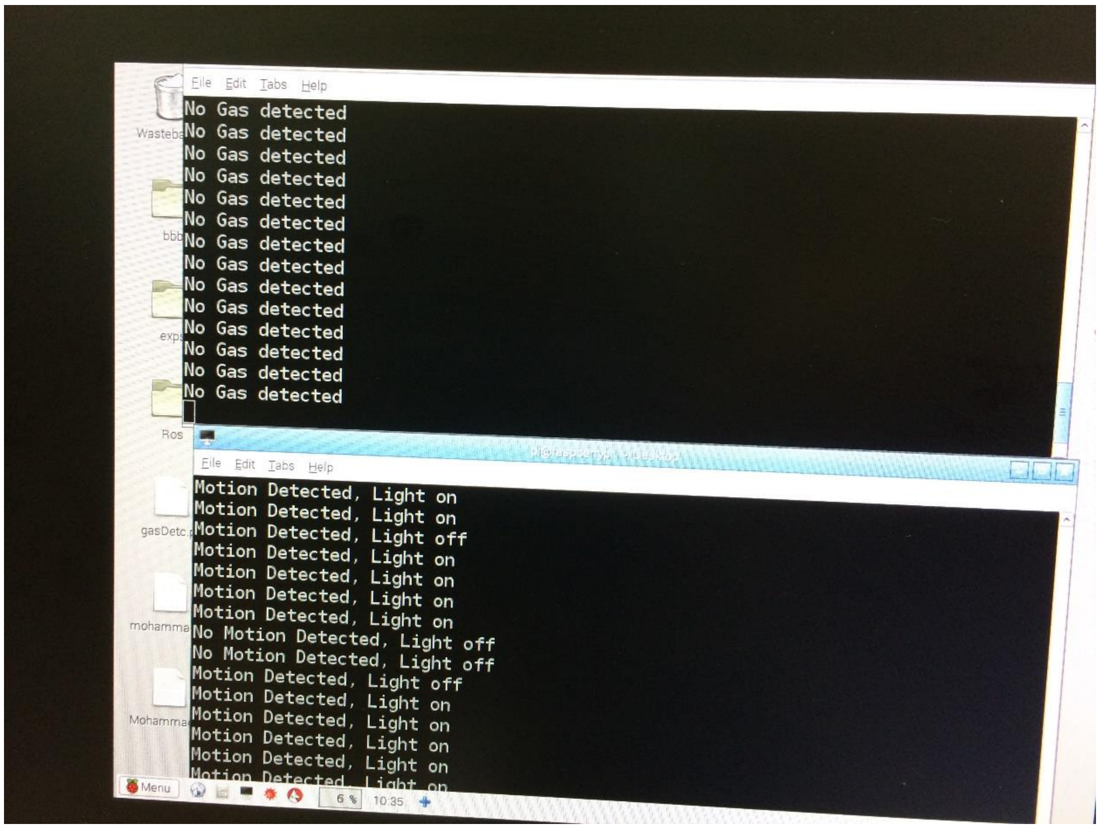
```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(35, GPIO.OUT)

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

while True:
        if GasDet(23) == False:
                print "Gas detected"
                #GPIO.output(35, True)
        else:
                print "No Gas detected"
                #GPIO.output(35, False)

      time.sleep(1)
```

The output from these two programs, when running them together was as shown in the following figure:



```
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
No Gas detected
```

```
Motion Detected, Light on
Motion Detected, Light on
Motion Detected, Light off
Motion Detected, Light on
Motion Detected, Light on
Motion Detected, Light on
Motion Detected, Light on
No Motion Detected, Light off
No Motion Detected, Light off
Motion Detected, Light off
Motion Detected, Light on
Motion Detected, Light on
Motion Detected, Light on
Motion Detected, Light on
Motion Detected, Light on
```

Figure 8: Output of the two programs together

Experiment 10 Tasks:

Testing Camera:

Connect the camera to the Raspberry PI and use this command:

fswebcam -d /dev/video0 -r 640x480 pic.jpeg

picture will be saved to /home/pi.

Smart camera:

This task is about making the web camera take picture only if a motion is sensed from the PIR sensor. When motion detected a LED will light on and a message indicating that motion was detected will be displayed. If no motion detected the camera won't take any picture, the LED will stay off and a message indicating that there is no motion detected will be displayed.

To do so, the following circuit should be connected:

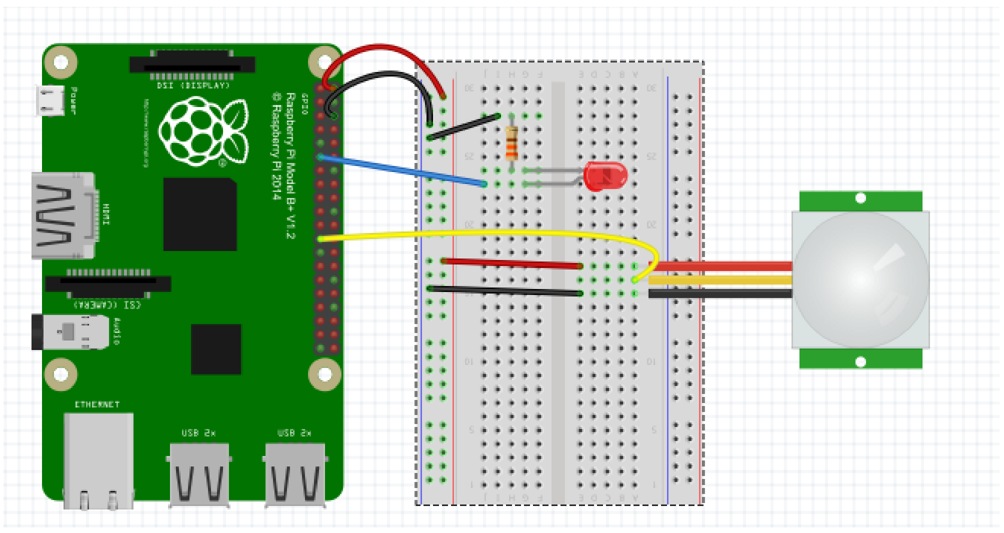


Figure 9: Circuit showing Raspberry PI connection with PIR and LED

The following python code was used:

```
import time
import RPi.GPIO as GPIO
import os

GPIO.setmode(GPIO.BOARD)
GPIO.setup(11,GPIO.OUT)      # pin 11 (LED) is output
GPIO.setup(23,GPIO.IN)       # pin 23 (PIR) is input


while True:

     if (GPIO.input(23) == True):
          print " Motion Detected "
          GPIO.output(11,1)
          os.system("fswebcam -d /dev/video0 -r 640x480 pic.jpeg")
     else:
          print " There is No Motion Detected"
          GPIO.output(11,0)

     time.sleep(1)
```

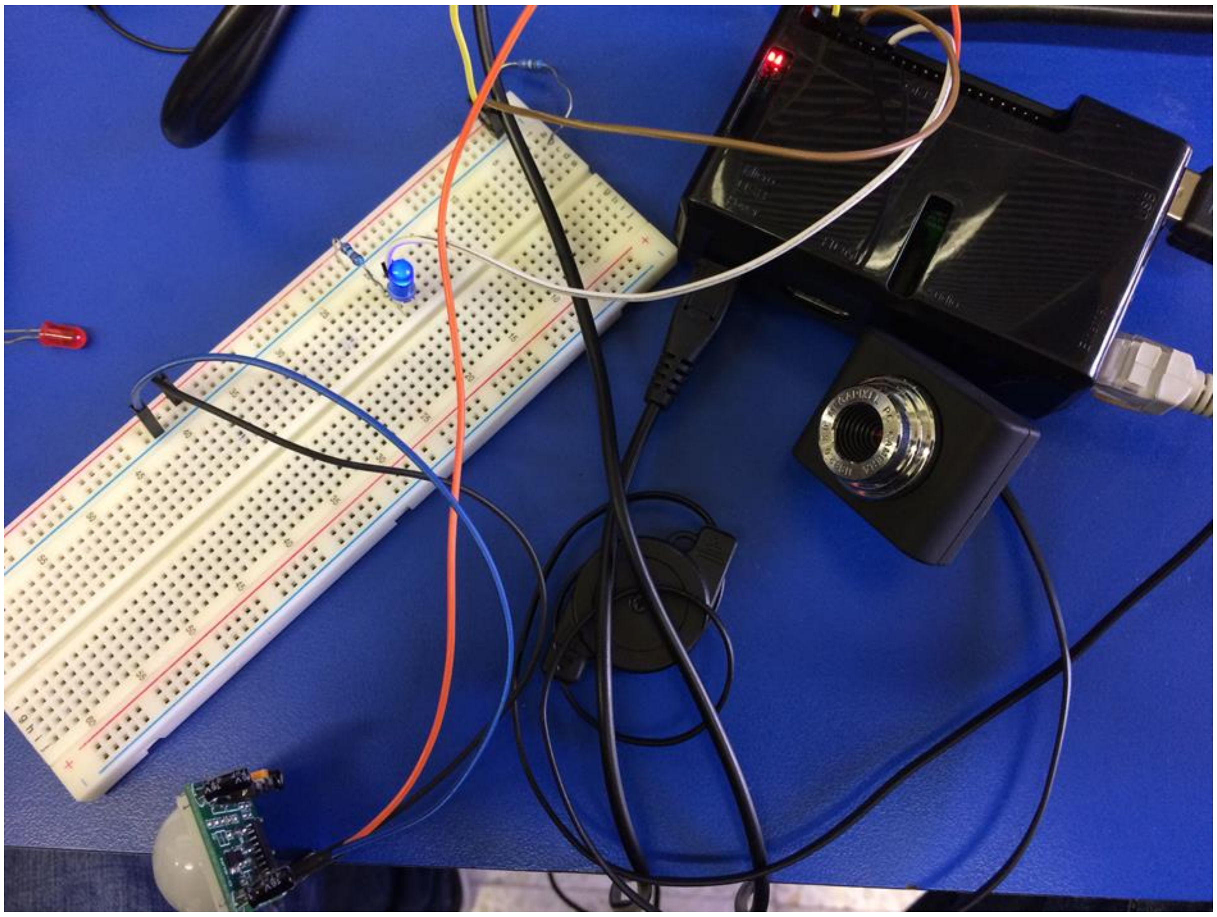


Figure 10: Picture showing the connections of the PIR with camera circuit in the lab

Time Lapse Camera:

Time-lapse photography is a technique whereby the frequency at which film frames are captured (the frame rate) is much lower than that used to view the sequence. That is, if you took a 30 minutes video, using time-lapse, you may show them in 2 or 3 minutes for example, depending on show speed.

To do so, first we need to write a program to take some pictures for the time-lapse. The following code will do the job:

```
import os
import time
import sys

FRAMES = 30 #is the number of the wanted pictures in 5 minutes which is 5*6=30
frameCount = 0
while frameCount < FRAMES:

# str is for converting numbers to a string and is to make it 4 digits.
imageNumber = str(frameCount).zfill(4)

# os.system to execute the command between branches.
os.system("fswebcam -d /dev/video0 -r 640x480 pic.jpg" %(imageNumber))

frameCount += 1
time.sleep(10) #Takes roughly 10 seconds to take a picture
sys.exit()
```

The raspberry pi will start to take pictures every 10 second for 5 minutes, and the picture will be saved in the folder that you have created. Now we need to make a video using the capture images using a tool that is called “MEncoder” which is installed on your raspberry pi. MEncoder is an open source program that can build video in different formats and sounds.

To make time-lapse using MEncoder, we just use this 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

a video with file name “timelapse.avi” will be created, to run it we may use this command:

Omxplayer timelapse.avi

Autostart Python Programs After Login:

To add a script that you want to run after the boot, you need to edit the file of /etc/profile, you may use this command to do so:

Sudo nano /etc/profile

Then add this command to the file and save it, so that you want it to get executed after boot:

Sudo python /home/pi/ timelapse /pythonprogrogram.py

Color Detection:

OpenCV Library will be used to detect colors using the webcam, the color that will be detected is blue, starting from (50, 50, 110) to (130, 255, 255).

The following code was used to detect blue color. Its main idea is simple, it captures a picture (or frame), convert it from RGB to HSV. It then defines the range for blue color (the min value is (50, 50, 110) and the max value is (130, 255, 255)), and threshold the image using these two thresholds to get a mask that indicates the location of blue pixels in the 2D array (the image). Finally, it makes a bitwise AND operation between the original image and this mask, so that it takes the pixels with blue color from the original image and display it.

```
import cv2
import numpy as np

cap = cv2.VideoCapture(0)

while (1):

    _, frame = cap.read()

    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    lower_blue = np.array([110, 50, 50], dtype = np.uint8)
    upper_blue = np.array([130, 255, 255], dtype = np.uint8)

    mask = cv2.inRange(hsv, lower_blue, upper_blue)

    res = cv2.bitwise_and(frame, frame ,mask = mask)

    cv2.imshow('frame', frame)
    cv2.imshow('mask', mask)
    cv2.imshow('res', res)

    k = cv2.waitKey(5) & 0xFF

    if k == 27:
        break

cv2.destroyAllWindows()
```

Face Detection:

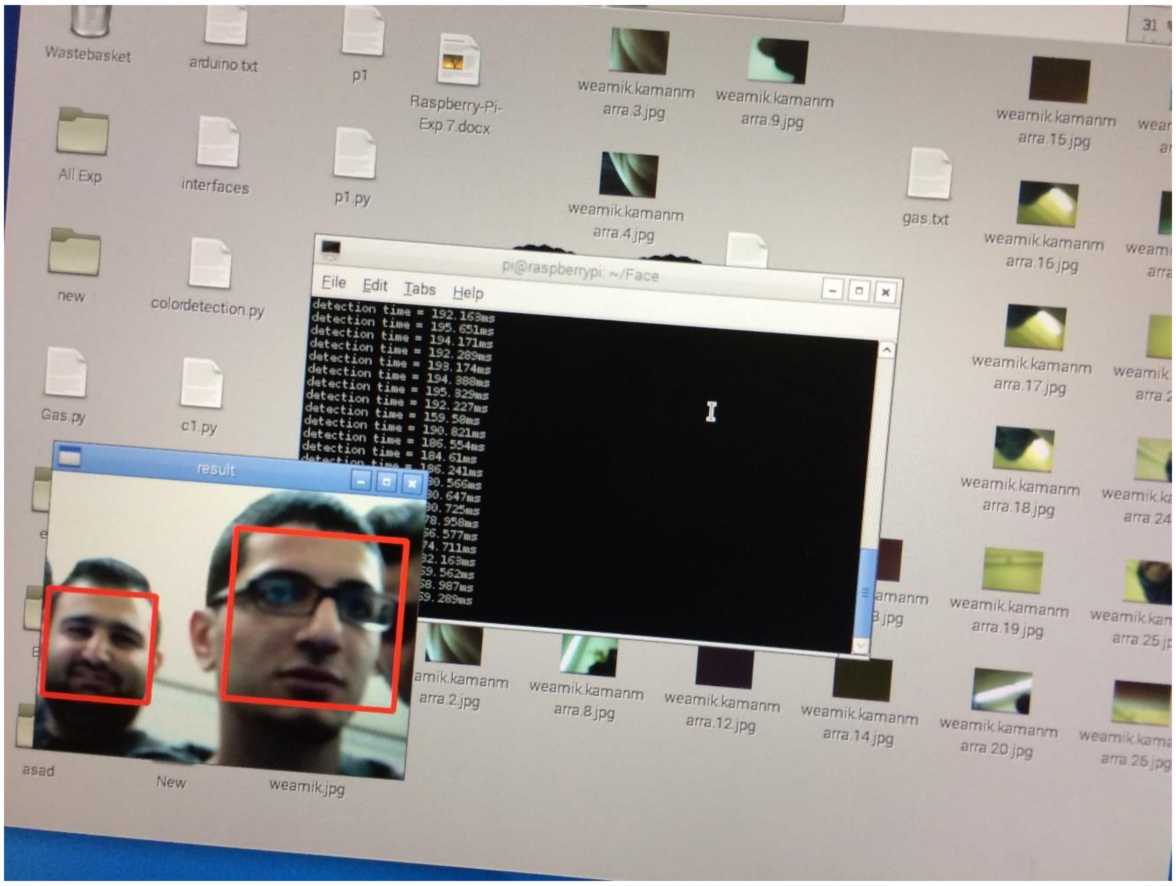
Face detection is one of the exciting application that can be done by raspberry pi and OpenCV. To run the face detection program, enter the following folder, using this command:

cd /home/pi/Face

then run the following command:

python facedetect.py –cascade=face.xml 0

The Interface will be shown to enable you to detect faces, just like the following figures:



```
detection time = 192.163ms
detection time = 195.651ms
detection time = 194.171ms
detection time = 192.289ms
detection time = 193.174ms
detection time = 194.388ms
detection time = 195.829ms
detection time = 192.227ms
detection time = 159.58ms
detection time = 190.821ms
detection time = 186.554ms
detection time = 184.61ms
detection time = 186.241ms
                 30.566ms
                 30.647ms
                 30.725ms
                 78.958ms
                 56.577ms
                 74.711ms
                 32.163ms
                 59.562ms
                 58.987ms
                 59.289ms
```

Figure 11: Face Detection

Arduino with Ultrasonic:

The Ultrasonic sensor is used to measure the distance between it and the first object facing it. We can use it on Arduino, and use a Raspberry PI to configure the Arduino.

To do so, first we need to connect the circuit in the following figure:

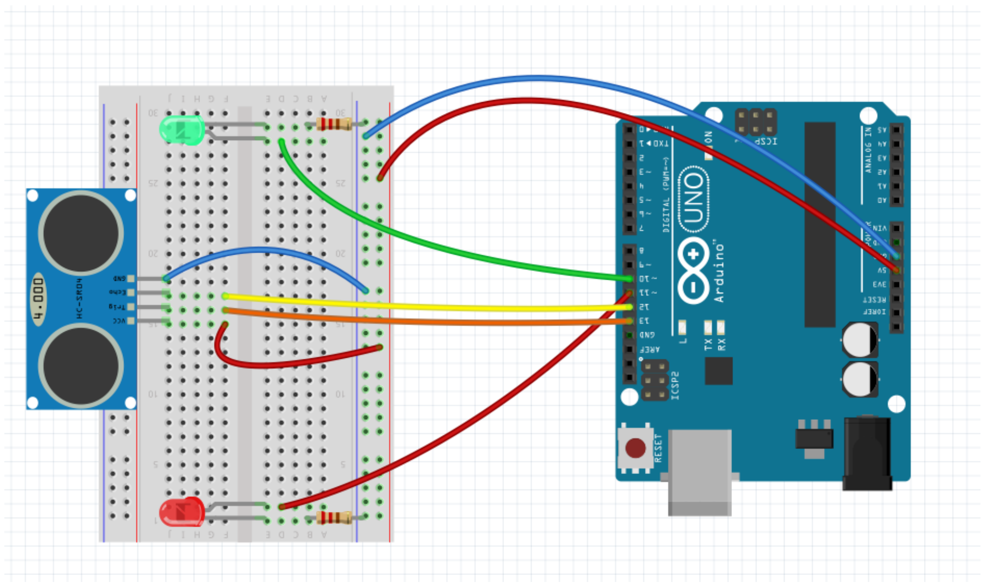


Figure 12: Circuit showing connection of ultrasonic sensor to the Arduino

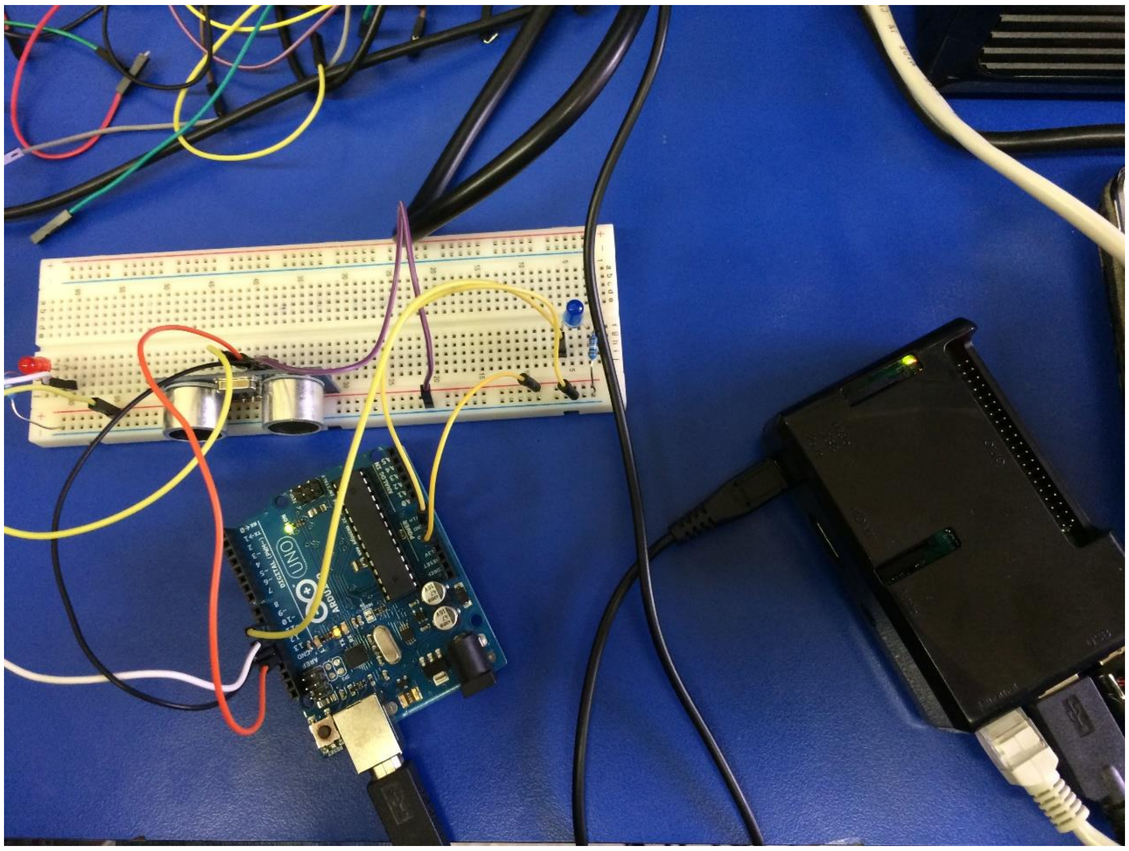


Figure 13: Picture showing the connections of the ultrasonic circuit in the lab

The following code was used and uploaded to the Arduino:

```
/*
HC-SR04 Ping distance sensor]
VCC to arduino 5v GND to arduino GND
Echo to Arduino pin 13 Trig to Arduino pin 12
Red POS to Arduino pin 11
Green POS to Arduino pin 10
220 ohm resistor to both LED NEG and GRD power rail
*/

#define trigPin 13
#define echoPin 12
#define led 11
#define led2 10

void setup() {
      Serial.begin (9600);
      pinMode(trigPin, OUTPUT);
      pinMode(echoPin, INPUT);
      pinMode(led, OUTPUT);
      pinMode(led2, OUTPUT);
}

void loop() {
      long duration, distance;
      digitalWrite(trigPin, LOW);
      delayMicroseconds(2);
      digitalWrite(trigPin, HIGH);
      delayMicroseconds(10);
      digitalWrite(trigPin, LOW);
      duration = pulseIn(echoPin, HIGH);
      distance = (duration/2) / 29.1;

      if (distance >= 200 || distance <= 0){
            Serial.println("Out of range");
            digitalWrite(led,HIGH);
            digitalWrite(led2,LOW);
      }
      else {
            Serial.print(distance);
            Serial.println(" cm");
            digitalWrite(led,LOW);
            digitalWrite(led2, HIGH);
      }

      delay (500);
}
```

This code will is to the ultrasonic and take readings (in an infinite loop), if the read distance was larger than 200 cm or less than 0 cm, then a message saying “Out of range” will be displayed on the serial monitor, the red light will be on and the other one will be off. Else, if the distance was between 0 and 200 cm, then the read distance will be displayed on the serial monitor, the red light will be off, and the other one on.

Conclusion:

From this experiment, students have introduced to Raspberry PI, and understood how powerful it is, with its small size specifically, which takes no space, consumes a very small amount of power, but gives a huge amount of benefits. It was introduced alongside with some sensors that has used to make so nice ideas, like lighting the lights if a motion has detected and only if there is no light (i.e. it’s night). Another nice idea was using the gas sensor to detect gas and turn on a fan to remove it (or turn on a buzzer as an alarm). The Raspberry PI camera was also used with many programs, like face detection, smart camera (which takes a picture when motion is detected), color detection, and time-lapse which all were like small prototypes for large and good projects that can be really implemented. Moreover, ultrasonic sensor was used with Arduino that was configured by the Raspberry PI, which shows another good benefit for the Raspberry PI, which is the ability to communicate and control other micro-controllers or interfacing components.

To conclude, these two experiments introduced us to many software and hardware resources that can be very useful for technical projects and studies.

References:

[1]: Interfacing Lab Manual - Experiment 9 (Accessed: 27 April 2017).

[2]: Interfacing Lab Manual - Experiment 10 (Accessed: 27 April 2017).