INTRODUCTION

It’s a motion detection system refers to the capability of the surveillance system to detect motion and capture the events. Motion detection is usually a software - hardware based monitoring algorithm which, when it detects motions will signal the surveillance camera to begin capturing the event. Also called activity detection. An advanced motion detection surveillance system can also send the mail and sms notification as well as video clip will be uploaded to internet.

**Quick Descriptions about the itempi raspberry**

**s**

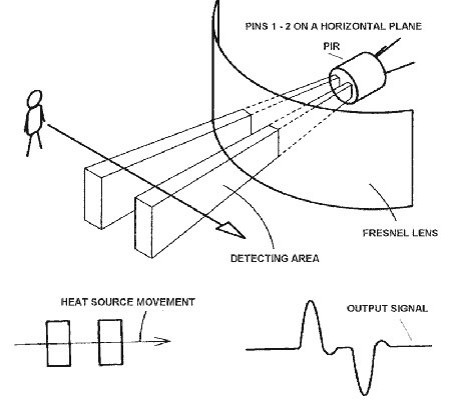
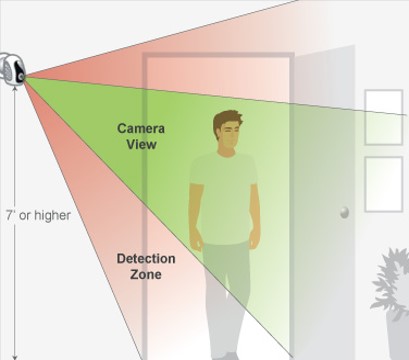
The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It’s capable of doing everything you’d expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

Passive Infrared Sensor (PIR sensor) 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.

Dropbox is a home for all your photos, docs, videos, and files. Anything you add to Dropbox will automatically show up on all your computers, phones and even the Dropbox website - so you can access your stuff from anywhere.

DESCRIPTION

Motion detection system is the first essential process in the extraction of information regarding moving objects and makes use of stabilization in functional areas, such as tracking, classification, recognition, and so on. In this project, we propose an idea and accurate approach to motion detection for the automatic video surveillance system. Our method achieves complete detection of moving objects by involving significant parts: motion detection by a sensitive PIR sensor, recording video footage by a camera module, all these are controlled by a raspberry pi. When a motion is detected by the PIR sensor then it sends a signal to the raspberry pi and then raspberry pi activates the camera and starts recording until the motion stopped. A mail notification is send to a given mail address and video footage is processed and then it will be uploaded to dropbox and then at last an sms notification is being send to the registered number.



We will use Python programming language to write the codes in Raspberry Pi. Python is a widely-used high level programming language. Its elegant syntax allows you to clearly define application behaviour using fewer lines of code than would be required in other languages like VB. It supports multiple programming paradigms including imperative, functional and object oriented styles, allowing a wide range of tasks to be performed.

**Procedure to make this project:**

For this project we need :

1. Raspberry Pi

2. Camera module

3. PIR sensor

4. Female-to-female jumper wires

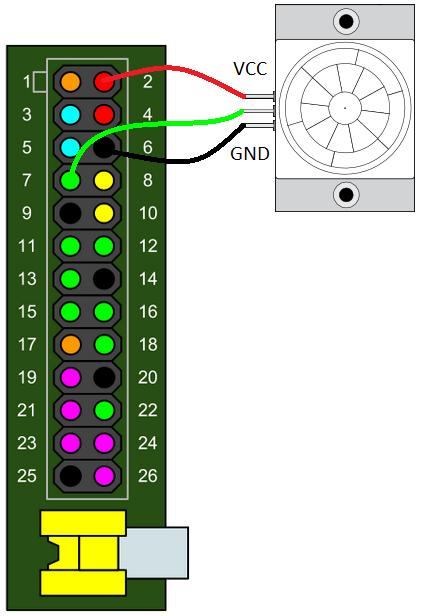
5. Power supply

6. SD card (pre-installed Raspbian)

7. Internet connectivity

**STEP 1: CONNECT THE PIR MOTION SENSOR**

Before booting your Raspberry Pi, connect the PIR module to the Raspberry Pi.



Refer to the diagram above for pin numbers. If you look closely at the pins on your PIR module, you'll see some white text on the PCB near the base of each one. VCC is for a +5 volts input. Take one of the female to female jumpers and connect the VCC pin to pin 2 on the Pi (coloured red on the diagram); this will make the Pi provide 5 volts of power to the PIR module. Use another jumper to connect GND on the module to pin 6 on the Pi (coloured black on the diagram); this completes the circuit and allows current to flow back out of the module into ground. Now do the same for the sensor pin OUT; you can use any of the green pins on the Pi for this. I am going to use pin 7, since it's the first general purpose one.

Note: If you have a different PIR module to the one pictured then your pin layout might be different; this is why I include the labels VCC GND and OUT.

Now boot your Pi and log in.

**STEP 2: TEST THE PIR MOTION SENSOR**

We're going to use the Python programming language to write some code that will detect movement and print out some text; we can extend the program to involve the camera board later on. When movement is detected the PIR motion sensor applies power to its OUT pin, which we have connected to GPIO pin 7 on the Pi. So in our code we just need to continually check pin 7 to see if it has power or not.

If a pin has power we call it HIGH and if not we call it LOW.

The program is pretty simple. We will first set up the Raspberry Pi GPIO pins to allow us to use pin 7 as an input; it can then detect when the PIR module sends power. We need to continually check the pin for any changes, so a while True loop is used for this. This is an infinite loop so the program will run continuously unless we stop it manually with Ctrl + C.

We then use two Boolean (True or False) variables for the previous and current states of the pin, the previous state being what the current state was the preceding time around the loop. Inside the loop we compare the previous state to the current state to detect when they're different. We don't want to keep displaying a message if there has been no change.

Firstly create a blank Python file with the following command:

nano pirtest.py

Enter the code below:

#!/usr/bin/python

import RPi.GPIO as GPIO

import time

sensorPin = 7

GPIO.setmode(GPIO.BOARD)

GPIO.setup(sensorPin, GPIO.IN, pull\_up\_down=GPIO.PUD\_DOWN)

prevState = False

currState = False

while True:

time.sleep(0.1)

prevState = currState

currState = GPIO.input(sensorPin)

if currState != prevState:

newState = "HIGH" if currState else "LOW"

print "GPIO pin %s is %s" % (sensorPin, newState)

Press Ctrl + O to save and Ctrl + X to quit.

Next make the Python file executable and then you can run it:

chmod +x pirtest.py

sudo ./pirtest.py

If you get an error saying RuntimeError: No access to /dev/mem it means you forgot to use sudo. You must run programs that access the GPIO as root and sudo does this for you; to help remember you can think of it as 'super-user-do'.

If you start moving or waving the sensor pin will go HIGH. Keep on waving and it will stay HIGH, and only go back to LOW if you keep still again. If you see the sensor behave like this, then everything is working correctly. If not, something is wrong and you need to go back and troubleshoot.

GPIO pin 7 is HIGH

GPIO pin 7 is LOW

GPIO pin 7 is HIGH

Press Ctrl + C when you want to exit.



On the PIR module you should see two orange components with sockets that fit a Phillips screwdriver (see above). These are called potentiometers, and they allow you to adjust the sensitivity of the sensor and the detection time. I would suggest setting the sensitivity to max and the time to min, but the choice is yours.

**Step 3: CONNECTING THE CAMERA**

The flex cable inserts into the connector situated between the Ethernet and HDMI ports, with the silver connectors facing the HDMI port. The flex cable connector should be opened by pulling the tabs on the top of the connector upwards then towards the Ethernet port. The flex cable should be inserted firmly into the connector, with care taken not to bend the flex at too acute an angle. The top part of the connector should then be pushed towards the HDMI connector and down, while the flex cable is held in place.

ENABLING THE CAMERA

Open the raspi-config tool from the Terminal:

sudo raspi-config

Select Enable camera and hit Enter, then go to Finish and you'll be prompted to reboot.

**STEP 4: INSTALL THE PICAMERA PYTHON MODULE**

To control the Camera Board using Python we need to install a module called [picamera](http://picamera.readthedocs.org/). Use the following commands to do this at the Linux prompt:

sudo apt-get update

sudo apt-get install python-picamera

This will take a minute or two to complete.

**STEP 5: PROGRAM THE CAMERA TO PREVIEW ON MOVEMENT**

Now we're ready to extend our previous program to give it the ability to control the Camera Board. To start with, let's just make our program display what the camera can see when movement is detected; we can set up recording to a file later.

Make a copy of the previous program and we'll use that for this step:

cp pirtest.py pirCamera.py

Now use the following command to edit the file:

nano pirCamera.py

We first need to add the import picamera statement at the top; this allows your program to access the pre-made code which can control the Camera Board. We then declare the camera object cam, which provides all the camera control functions that we need to use. Then inside the while loop where we print the HIGH or LOW message, we can test to see if currState is HIGH / True (meaning movement is detected); we can then start or stop the camera preview accordingly.

Either modify manually or copy the code below:

#!/usr/bin/python

import RPi.GPIO as GPIO

import time

import picamera # new

sensorPin = 7

GPIO.setmode(GPIO.BOARD)

GPIO.setup(sensorPin, GPIO.IN, pull\_up\_down=GPIO.PUD\_DOWN)

prevState = False

currState = False

cam = picamera.PiCamera() # new

while True:

time.sleep(0.1)

prevState = currState

currState = GPIO.input(sensorPin)

if currState != prevState:

newState = "HIGH" if currState else "LOW"

print "GPIO pin %s is %s" % (sensorPin, newState)

if currState: # new

cam.start\_preview()

else:

cam.stop\_preview()

Press Ctrl + O to save and Ctrl + X to quit. To run the program use the following command:

sudo ./pirCamera.py

Press Ctrl + C when you want to exit.

**Step 6: Configure email with raspberry pi**

Installing and configuring SSMTP

1. Make sure your repositories are up-to-date:

sudo apt-get update

2. Install SSMTP and mail utilitites:

apt-get install ssmtp

apt-get install mailutils

3. Edit the SSMTP configuration file:

sudo nano /etc/ssmtp/ssmtp.conf

Write these lines in it:

#

# Config file for sSMTP sendmail

#

# The person who gets all mail for userids < 1000

# Make this empty to disable rewriting.

root=postmaster

# The place where the mail goes. The actual machine name is required no

# MX records are consulted. Commonly mailhosts are named mail.domain.com

mailhub=smtp.gmail.com:587

# Where will the mail seem to come from?

#rewriteDomain=

# The full hostname

hostname=raspberrypi

AuthUser=YOUR GMAIL USERNAME@gmail.com

AuthPass=YOUR GMAIL PASSWORD

UseSTARTTLS=YES

UseTLS=YES

# Are users allowed to set their own From: address?

# YES - Allow the user to specify their own From: address

# NO - Use the system generated From: address

#FromLineOverride=YES

Be sure to specify the correct GMail user name and password here, otherwise you will get authentication errors.

If the host name of your Raspberry Pi is different from “raspberrypi”, specify your actual host name here instead.

b) Optional lines:

rewriteDomain=your.domain

Specify this if you would like the outgoing emails to appear to be sent from your.domain (instead of from [gmail.com](http://gmail.com/)).

FromLineOverride=YES

Specify this if you would like SSMTP to leave the From field of the emails untouched. Otherwise it will overwrite the from field with the name of the Linux user which sends the email. The overwriting name is taken from the 5th value in the line that corresponds to the sending user, from the /etc/passwd file. If you plan to send emails from a website (for example from a WordPress plugin) and wish to have nice sender names like “John Doe”, I recommend commenting this line (which is equal to setting the value to NO), otherwise your website will only be able to send emails with less nice sender names, like xyzabc@your.domain. In other words, you probably want SSMTP to overwrite the sender field with a nice name taken from the/etc/passwd file.

**Step 7: Configure raspberry pi with Dropbox.**

Download and uncompress the Python SDK. To install the dropbox module and any dependencies, run the setup script (you may need sudo).

python setup.py install

Alternatively, you can use pip to automatically download and install the module.

pip install dropbox

[**How to use Dropbox with Raspberry Pi**](http://raspi.tv/2013/how-to-use-dropbox-with-raspberry-pi)

The steps include:

Set up Dropbox account if you don’t already have one.

Download and set up Dropbox Uploader on the Raspberry Pi (this is done from the command line).

Create a Dropbox app and get access credentials.

These are the settings I used on the Dropbox developer site when creating a new app for the camera:

Better to set file type to all file types.

Setting up other/different Dropbox folders

First, you must have a Dropbox account with enough storage space for the content.  This may require the [purchase](https://www.dropbox.com/help/15/en) of addtional space, as Dropbox currently limits free space to 2GB.  Here is current [instructions](https://www.dropbox.com/gs) provided by Dropbox.

You must then create an "App" on the Dropbox server:

<https://www.dropbox.com/developers/apps/create>

Select "Dropbox API app"

Select  "files and datastores"

Select "Yes" for the "Can your app be limited to its own, private folder?" question

Enter an appname that is unique

Select the "Create App" button

The Dropbox Server will then present you a web page filled with a variety of information.  You will need the contents of the fields labelled:

App key

App secret

One more step to test :

Test the Dropbox program on the Raspberry Pi by uploading a file (anything will do).

**Step 8: Configure raspberry pi with free sms service.**

Get Gmail Notification Sms Free

With the help of some third-party services or using sms gateway in email forward option one can get ‘gmail sms alerts’ but of course these services are not free and there might be some privacy related issues. However you will get instant answers to these queries regarding a just-received email.

Subject of the email.

Name and email address of the sender.

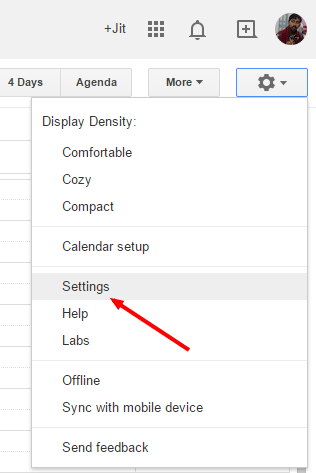
Exact time with date for each new individual unread email.

Getting started with ‘Free Gmail mobile alert system’, you just need to cover few simple steps as mention below.

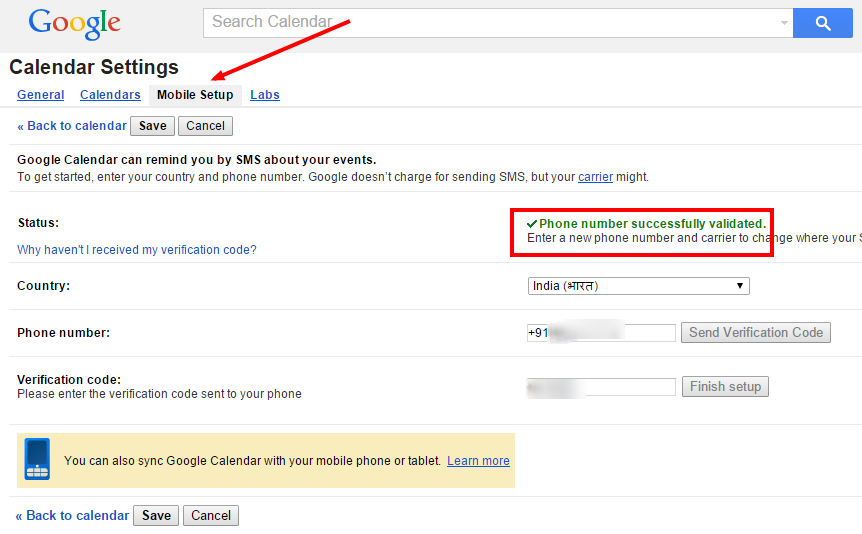
Step I: Make Free Sms Reminder with Google Calendar Set-up

Sign in to [Google Calender](https://www.google.com/calendar/).

From top-right corner of your browser Click on Gear button and select Settings.

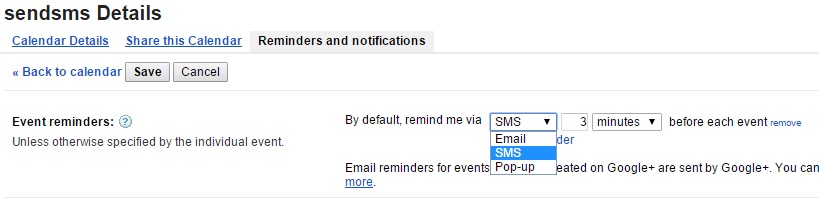


Go to Mobile Setup tab and verify your mobile number there.

[](http://www.ampercent.com/wp/wp-content/uploads/Google-Calendar-phone-verification.png)

You will automatically be redirected to Reminders and Notifications page.

Set Sms option as your default event reminder and set the time to 3 minutes.



Now click on save button to complete Google calendar configuration for phone notification.

Step II: Configure Gmail and Create Filter

Sign in to your Gmail account and from the icon select settings.

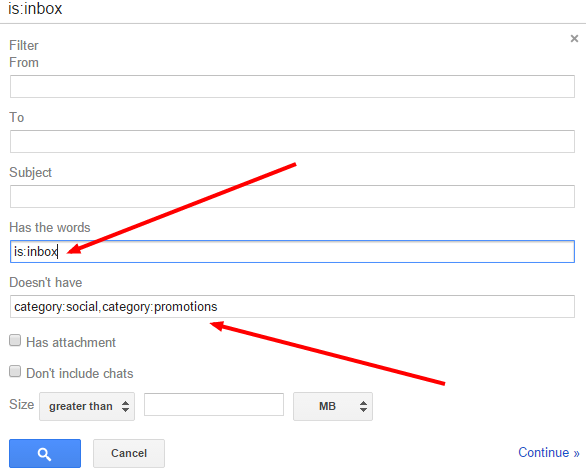
Go to labels tag, create a new label with the name sendsms.

Now create a new filter from Filters tab and to get sms alerts for all new emails from Gmail inbox.

Enter “is:inbox” in the ‘Has the words’ field.

To bypass email alerts for social and promotional emails enter “category:social,category:promotions” in “Doesn’t have” field.

See the picture below:



Select “Create Picture with this Search” option.

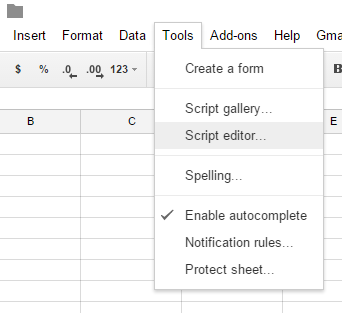
Tick the box “Apply the label” > then select “sendsms” > click “Create filter”.

Step III: Configure Google Spreadsheet and add some actionscript

Sign in to [Google spreadsheet](https://docs.google.com/spreadsheets/).

Create a new spreadsheet.

Navigate to Tools and choose Script Editor.



The code is:

function sendsms()

{

try

{

var label = GmailApp.getUserLabelByName('sendsms');

var threads = label.getThreads();

var now = new Date().getTime();

var cal=getCalendar();

for(i in threads)

{

cal.createEvent(threads[i].getFirstMessageSubject()+": "+threads[i].getMessages()[0].getFrom(),new Date(now+60000),

new Date(now+60000)).removeAllReminders().addSmsReminder(0);

}

label.removeFromThreads(threads);

}

catch(err)

{

Logger.log("Error Occured"+ err.toString());

}

}

function getCalendar()

{

try

{

var cal=CalendarApp.getOwnedCalendarsByName("sendsms")[0];

if (!cal)

{

cal=CalendarApp.createCalendar("sendsms");

return cal;

}

else

{

return cal;

}

}

catch(e)

{

Logger.log("Error Occured"+ e.toString());

}

}

function onOpen()

{

var ss = SpreadsheetApp.getActiveSpreadsheet();

var menuEntries = [ {name: "Authorize", functionName: "authorize"},

{name: "Stop SMS alerts", functionName: "stopsmsalerts"}

];

ss.addMenu("Gmail SMS Alerts", menuEntries);

}

function authorize(){}

function stopsmsalerts()

{

var allTriggers = ScriptApp.getScriptTriggers();

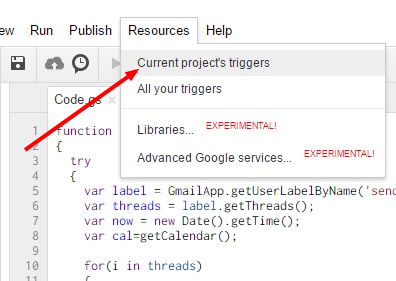
for(var i=0; i < allTriggers.length; i++)

ScriptApp.deleteTrigger(allTriggers[i]);

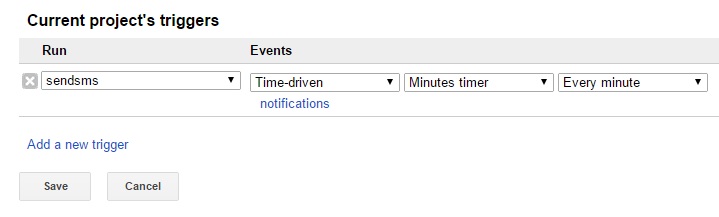
Browser.msgBox("Success", "You will not be getting SMS alerts anymore.",Browser.Buttons.OK);

}

Select Resources > Current project’s triggers.



Click on “Add a new trigger” and configure the trigger as per the below image.

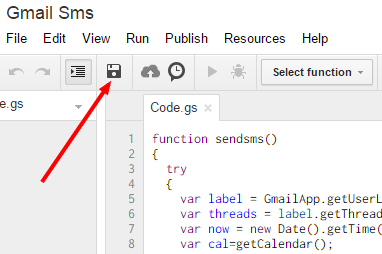


At the Run field select ‘sendsms‘.

At the Events field select Time Driven >> Minutes Timer >> Every Minute. Then click Save button.

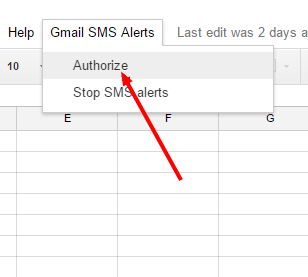
To authorize the script, click on continue and Save respectively.

Now click on Save icon from the top menu and close the Window.



Now back to Google spreadsheet and you will find ‘Gmail Sms Alerts’ tab there.

Select ‘Authorize’.



Click on Ok.

Finally when you will be asked for Gmail integration permission, Accept it.

That’s it. Now you’ll instantly get an sms alert as soon as any email will come to your Gmail Inbox.

Edit a test file named message.txt( it contains the mail body part)

Sudo nano /home/pi/message.txt

A recent motion has been DETECTED .....

Check the video footage for details.....

REGARDS,

PI (SAKE NAWAB ARZOO)

At last we have to write last python script **(MAIN SCRIPT)**

Sudo nano ./test1.py

**Code:**

#!/usr/bin/python

import RPi.GPIO as GPIO

import time

import picamera

import datetime

import subprocess

import dropbox

import os

# Get your app key and secret from the Dropbox developer website

app\_key = 'write the key got from the dropbox'

app\_secret = 'write the secret key'

def getFileName():

return datetime.datetime.now().strftime("%Y-%m-%d\_%H.%M.%S.h264")

def dropboxAuth():

accessTokenFileOverwrite = open("accessToken.txt", "w+")

flow = dropbox.client.DropboxOAuth2FlowNoRedirect(app\_key, app\_secret)

authorize\_url = flow.start()

# Have the user sign in and authorize this token

authorize\_url = flow.start()

print '1. Go to: ' + authorize\_url

print '2. Click "Allow" (you might have to log in first)'

print '3. Copy the authorization code.'

code = raw\_input("Enter the authorization code here: ").strip()

try:

# This will fail if the user enters an invalid authorization code

access\_token, user\_id = flow.finish(code)

accessTokenFileOverwrite.write(access\_token)

except:

print "failed authorization, restart"

accessTokenFileOverwrite.close()

os.remove("accessToken.txt")

accessTokenFileOverwrite.close()

def dropboxUpload(fileToUpload):

if not os.path.isfile("accessToken.txt"):

dropboxAuth()

#get access token from file

accessTokenFileRead = open("accessToken.txt", "r")

access\_token = accessTokenFileRead.read()

accessTokenFileRead.close()

# make client

client = dropbox.client.DropboxClient(access\_token)

#upload file

fileToUploadObject = open(fileToUpload, "rb")

response = client.put\_file(fileToUpload, fileToUploadObject)

fileToUploadObject.close()

sensorPin = 7

GPIO.setmode(GPIO.BOARD)

GPIO.setup(sensorPin, GPIO.IN, pull\_up\_down=GPIO.PUD\_DOWN)

prevState = False

currState = False

cam = picamera.PiCamera()

while True:

time.sleep(0.1)

prevState = currState

currState = GPIO.input(sensorPin)

if currState != prevState:

newState = "HIGH" if currState else "LOW"

print "GPIO pin %s is %s" % (sensorPin, newState)

if currState:

print "A MOTION HAS BEEN DETECTED !!!"

fileName = getFileName()

print "Starting Recording..."

cam.start\_preview()

cam.start\_recording(fileName)

print (fileName)

else:

print "MOTION STOPPED"

cam.stop\_preview()

cam.stop\_recording()

print "Stopped Recording"

print "Sending Mail Notification..."

subprocess.call("mail -s 'Motion Detected' desire.email.address@xyz.com < /home/pi/message.txt", shell=True)

print "Complete"

print "Uploading the footage to Dropbox..."

dropboxUpload(fileName)

print "Complete"

print "Sending SMS Notification..."

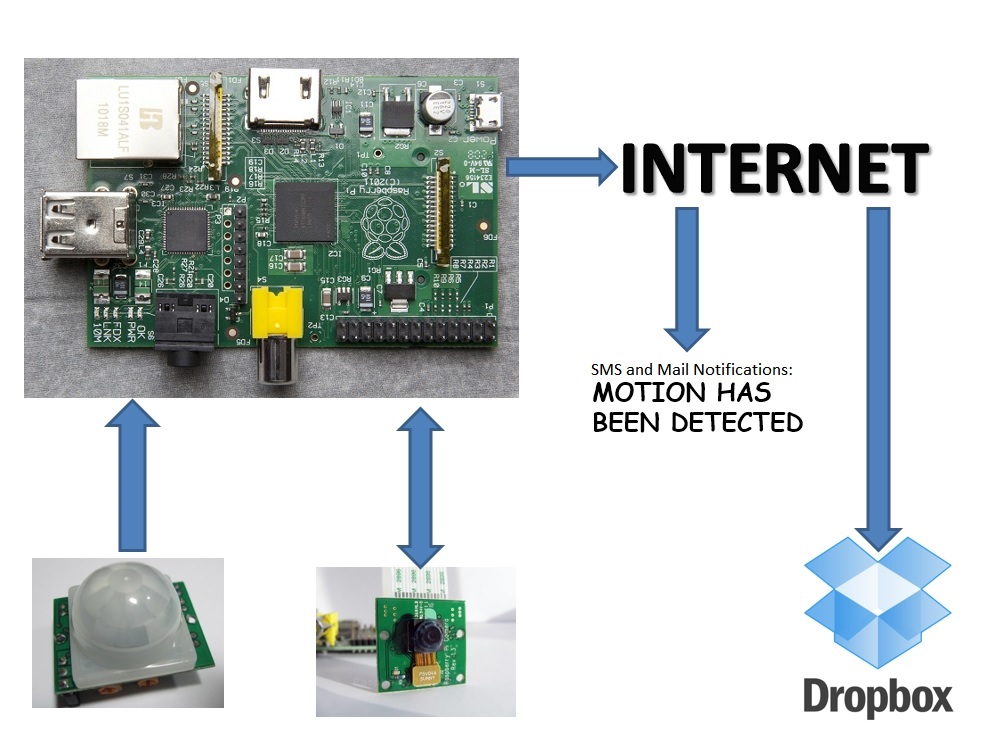
subprocess.call("mail -s 'Motion Detected' own.email.address@gmail.com < /home/pi/message.txt", shell=True)

print "Complete"

print "SMS Notification generally takes upto 1-2 minutes"

print "But Sometimes due to heavy newtork traffic it can take upto 10 mins"

**How the project works?**



**USES OF THIS PROJECT**

* Could be used for monitoring your private items.
* Could be used for home security.
* Could be used in parking areas, garages etc for security purpose.
* Could be used for Bank Lockers.
* Could be used for automatic enroll of persons visited a place for security.
* Could be used for wilflife photography.
* Could be used for protecting a item from being stolen or no one can disturb it .
* Could be used in banks , malls etc for automatic locking and could be used for many other purposes..

**The Advantages of Advance Motion Detection system**

Surveillance for security purposes is not the only area that best utilizes the advantages of a motion detector built into a camera. The motion detector is technology's answer to patience. It allows coverage of an area for longer periods of time than a traditional continuous filming camera does, making this device your best bet for sit-and-wait photo or filming opportunities.

**Saves Time**

If you need to find an intruder on your security camera tape, the motion detector will save you from hours of viewing uneventful tape. The camera is only turned on when it detects motion in the area. This allows you to view just the action without having to weed through an entire tape consisting of hours of filming an empty area, which is what a traditional continuous-filming security camera would offer.

**Saves Energy**

Without a motion detector, your security camera will tape continuously, which wastes energy when there is nothing to film. The motion detector turns the camera on when it is needed and allows it to stay off when nothing is happening.

**Element of Surprise**

The lights need to be on so you can see what images appear on the security camera, but with a motion detector camera, the lights can be set to go on when the camera detects the motion. This offers the element of surprise to the person of interest appearing in the security film. A thief will scope out the area for cameras and avoid them, but if they enter a room with the lights off they will not see the camera until it is too late: The motion detector has turned on the lights and started filming before the intruder realizes he has been caught in the act.

**Taping Wildlife**

Motion detector cameras are used for purposes other than to catch intruders at home and at your business. The cameras are also used out in the wild when trying to capture pictures of animals that can be hard to find. The camera with a motion detector allows you to leave the camera on for long periods of time, unlike a camera that continuously films. A camera without a motion detector would need to have the tape replaced every hour or so depending on how long the tape can run for. With a motion detector camera, you do not need to constantly change the tape because it is only filming when there is something to film. Without this feature, you would not be able to leave a camera out in the wilderness for days at a time to catch pictures of wild animals because the tape would need constant replacing.

CONCLUSION

A Raspberry Pi can be used in different field and in different way for different works . It is just a single board computer which can perform almost like a computer .

In this project we have made a Advance Motion Detection System with the help of Raspberry Pi. The detection results produced by our proposed method were both qualitatively and quantitatively analysed through visual inspection and for accuracy, along with comparisons to the results produced by other state-of-the-art methods. The analyses show that our method has a substantially higher degree of efficacy, outperforming other methods by an metric accuracy rate of up to 50 – 80 %.