Repurpose Old Smartphones for Home Automation by Turning Sensors into Signals

Mpho C. Mphego
Test & Verification Engineer
Correlator-Beamformer
SKA South Africa
Pinelands, Cape Town, South Africa
mmphego@ska.ac.za

S.P. Daniel Chowdhury
Electrical Engineering Department
Tshwane University of Technology, TUT
Pretoria, South Africa
Spchowdhury2010@gmail.com

***Abstract*— This paper proposes an approach to build a low-cost offline home automation or appliance automation by means of re-purposing old and unused smartphones by means of exploiting low-level sensors such as accelerometer, microphone, GPS and temperature. It takes information about the surrounding environment through the low-level sensors from a smartphone's sensor(s) and uploads it directly to the Raspberry Pi for processing before making relevant changes to the home environment for instance, switching light on/off by means of shaking your smartphone which utilizes the low-level accelerometer sensor. Experimental results demonstrated that the system can accurately control a home environment and having been offline the user doesn't risk uploading their personal information to the internet which offers privacy and at this day of age - cybersecurity is a priority when you work home automation systems as most of them depend on being online as well as uploading important information to the internet which some people might end up using against the home-owner.**

***Index Terms*—Android, Home Automation, IoT, Raspberry Pi, Security**

I. Introduction

As technology advances a lot of automation implementation in various fields has been introduced to maintain security, time and cost. In this process, our homes lags-behind, even though a lot of advanced equipment's are introduced each year, the use of these equipment is limited in the context of our homes. The most critical obstruction in home automation is the availability of these technologies and the cost implications involved as well as the maintenance.

Hence the idea to design and build a low-cost home automation system based on a smartphone and does not allow any sophisticated installations and home infrastructure excessive modifications. The major concern in this case is affordability, usability and security - which leads us to the design of a low-cost home automation system which offers seamless control by means of utilizing an old smartphone device.

II. Problem Statement

The main problem statement for the current system being developed is the ability to make life a lot easier by means of automating household equipment. A good example would be the automatic washing machine which helped in transforming washing which is the most-hardest and dull domestic duties in a household to being one of the least burdensome work (in my humble opinion). Humans have gotten to the point where they prefer simplified and non-sophisticated systems to simplify their livelihood. This led to the development of a home automation system by utilizing smartphone sensors and control which offers variety of solutions such as lighting, indoor weather, energy usage and gas monitoring as well as security. Most current systems in the market do not conveniently integrate the house systems in one product. Instead, the buyer must purchase various expensive devices and integrate them to form a system, which often requires significant technical knowledge as well as being connected to the internet to fully integrate with each other. Furthermore, most commercially available products integrate their products using a power line technology instead of wireless network. [1] This leads to decreased network security and increased hassle for installation. [2]

III. Importance and Benefits of the Project

There are various commercially available products in the home automation industry that perform functions like the home automation system via the smartphone. However, most current systems in the market do not conveniently integrate the house systems in one product.

- ***Why the need of another system of this type?***

The proposed solution adds various unique features not currently found in this market that make the proposed system unique:

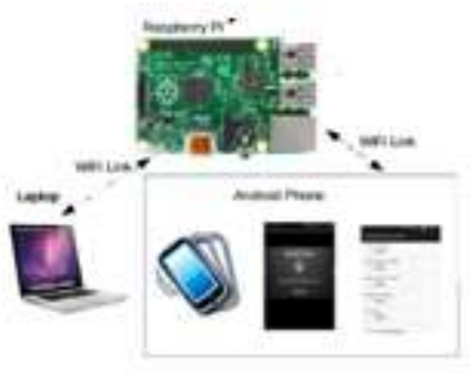
- The system allows historical data gathering of the system so that they can be viewed by the user to help detect any significant changes by means of logging.

This can help in the diagnostics of problems by being able to see past data. Many top end building automation systems provide this feature, but the lower end home automation systems do not.

- Most of the automation systems on the market today have the ability to send alarms via e-mail or by pre-recorded voice messages. The problem with these systems is that if the system fails (loss of power, lighting strike, internet service goes down) the remote interface does not indicate a problem until the operator tries to access the system. In many cases the time when you need it most, like loss of power, is the time the system is unable to warn the owner of the problem. The solution to this problem is to have the application on the smart-phone test the status of the system periodically. If the system fails to respond in a present length of time the smart phone can notify the owner that there may be a potential problem. This handshake between the systems allows for a more reliable warning system than the currently available 'call out' systems.
- By choosing a wireless communication medium, we could save wiring and installation cost as we could reuse existing infrastructure with minimal modification. By developing an application that enables users to control devices from the computer instead of a dedicated console, we could save the cost

IV. Practical Design

This section provides an overview of the system and steps in the developments process. The system interfaces with external aspects. Each of the external aspects is described in the text below. Figure below shows the interactions among the various interfaces of the system as per the designer.



The system can be configured to work remotely however it is never advisable unless the system is configured on a different network subnet on a dedicated router or dedicated VLAN, for the system to be isolated from the local network.

This kind of system presents many advantages compared to others on the same market.

- Uninterrupted Power Supply in the case of power outages as the smartphone will always be connected to the power supply – which means it will always be on depending, unless power to the house is disconnected or power supply breaks. That will lead to utilizing the battery API such that when the power is lost – the system should notify the home owner via the SMS API.
- HVAC control using IR and temperature and humidity monitoring (outdoors and indoors) – By utilizing the temperature API and IR API the system can be utilized to control and monitor.
- Energy usage monitoring by means of using a pulse counter mounted onto the household electricity meter.
- Bluetooth or Wi-Fi-based passive Bluetooth or Wi-Fi presence detector, which can notify the home owner of anyone that has connected to the network – Who's home and who's not.
- Smart Alarm API running on the Raspberry Pi, which wakes the home-owner up and then reads out the current weather and the day's forecast, as well as the current news while it opens the blinds. It switches the coffee maker on such that as soon as the home-owner is ready to leave the house, he/she can enjoy a freshly brewed cup of coffee.
- Smart closet API running on the Raspberry Pi, which notifies home-owner to carry a jacket or umbrella before they leave the house depending on the weather.
- Smart doorbell API running on the Raspberry Pi, which sends an SMS/E-mail and Push notification as well as takes a picture of the visitor and sends to home-owner.
- TV proximity sensors (connected to a Raspberry Pi), to avoid kid's straining their eyes by standing close to the TV.
- Voice recognition and gesture control, by means using low-level smartphone sensors, which can be used to switch on/off appliances in the house by means of shaking the device at a certain speed or sending voice commands.

V. Design and Development

The paper discusses the concept of utilizing an old Android smartphones sensor to control and monitor a household environment, this concept can be scaled and applied to industry.

- ***Android Mobile Control***

The Android mobile application was written in Java programming language under Android Studio. Android Studio is the official integrated development environment (IDE) for the Android platform. [3] The mobile application was designed with simplicity in-mind, it uses the Android System WebView which accesses Google's chrome with minimal features, and the application on start-up automatically connects to the Raspberry Pi Apache webserver considering that the mobile phone is connected to the same network as the Raspberry Pi. [4]

Android's WebView, as described by Google, is a "system component powered by Chrome that allows Android apps to display web content." In other words, WebView allows 3rd party apps to show content in an in-app browser or in an app screen that pulls from the web. It's important, and has only recently (with Lollipop) been decoupled as a stand-alone system component that can be updated as Google sees fit. And that's important, because it allows Google to push security fixes and other enhancements without the need to push an entire system update. [5]

Figure 1 Android Control UI below shows how the mobile control interface currently looks like - with its simplistic controls. It features only the lighting controls. More development shall follow, for instance adding an indoor temperature monitoring gauge.

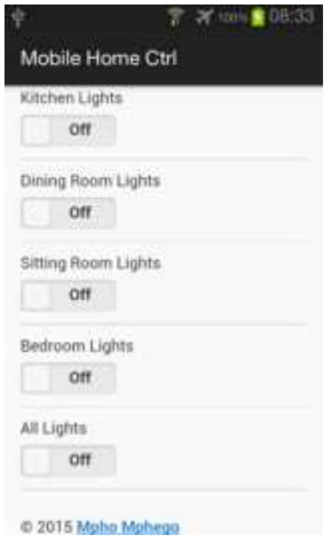


Figure 1 Android Control UI

- ***Android Gesture Control***

The gesture control consists of two (2) features, namely Voice Recognition and Mobile Shake control. Figure below shows a Mobile Gesture Control dialog presented upon application launch, which is written in Python programming language, that is running on QPython application installed on the Android smartphone. [6]

QPython is a script engine that runs Python on android devices. It lets your android device run Python scripts and projects. It contains the Python interpreter, console, editor, and the SL4A Library for Android

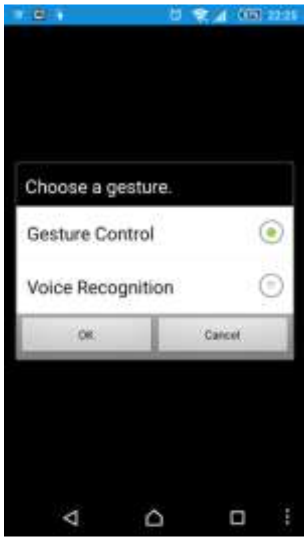


Figure 2 Mobile Gesture Control

1) Voice Recognition [Front End.]

Voice/Speech recognition is the ability of a machine or program to identify words and phrases in spoken language and convert them to a machine-readable format (binary 1 and/or 0). Basic speech recognition software has a limited vocabulary of words and phrases and may only identify these if they are spoken very clearly. [7]

By using the Google's Android API's by utilizing the smartphones microphone we can have access to the low-level controls of an Android smartphone and have control of low-level Offline Google Android application. [8]

How does the mobile voice recognition work?

In this section, we will breakdown how we exploit Google's API Speech-to-Text and how we send the text(string) over UDP to the server, in this case a Raspberry Pi listening in on an open port and be able to switch different appliances in and around the house using a smartphone. [9]

When the application is launches, a dialog box is displayed offering the user two (2) options - Voice recognition and mobile shake control. When voice recognition option is selected – the application starts broadcasting its UDP connection to all ports open on the entire subnet network, if the Raspberry Pi accepts the UDP connection then the smartphone's application requests the user to input voice commands, Figure 3 Voice recognition speak request below shows the pop up request, If any words are recognized they're then sent over to the server for further processing, else if the words are not clearly recognized it will request user to input voice commands once more this is done in a loop until the command '*Exit'* is received, which will exit the Python script.

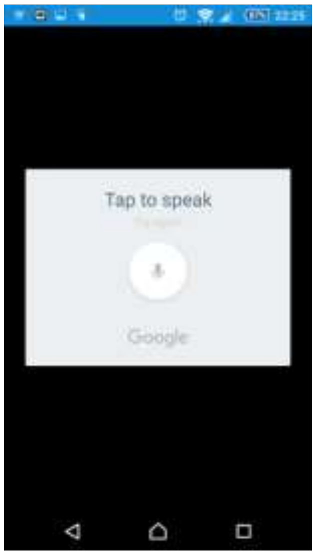


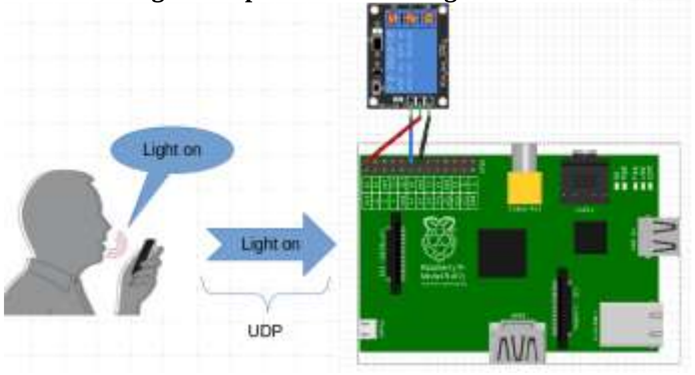
Figure 3 Voice recognition speak request

2) Voice Recognition [back end]

The Raspberry Pi runs a script that constantly listen for UDP packets on a certain specific port, the script is executed upon system reboot with an exception of restarting the program in-case of any runtime failures. Upon receiving the UDP packets sent from the Android smartphone as a list of strings – Speech to text via Google's API, the script evaluates if whether the string is of text or numbers (accelerometer data), if the raw data received via UDP are a list of strings their compared to the strings that are already stored on the running script - These strings / commands are programmable via the configuration file, this phenomenon is knows as keyword spotting. [10]

The program uses keyword spotting– as it listens to specific words or text if the word is invalid this gets logged into a file for analysis on a later stage. Figure 4 Speech-to-Text diagram illustrates how this feature of the proposed system is implemented. When a certain known pattern of word is recognized such as 'Bedroom light on', the bedroom light should ultimately be switched on via the relay it is connected on as illustrated. When the Python script is executed a string of words are recognized and depending on the word matched, this will execute a relay control command.

Figure 4 Speech-to-Text diagram



3) Mobile Shake Control [front end]

By exploiting the Google's Android API's, we can have access to the low-level controls of an Android ran smartphone and retrieve sensor data. In this paragraph, we will explain how the mobile shake control works. The Android mobile platforms support various sensor. Such as:

- Motion sensors: By measuring acceleration forces and rotational forces along three axes (in most devices) [e.g.: accelerometers, gravity sensors, gyroscopes and rotational vector sensors]
- Position sensors: By measuring the physical position of a device [e.g.: orientation sensors and magnetometers]
- Environmental sensors: By measuring environmental parameters such as temperature, pressure, illumination and humidity (depend on device) [e.g.: Barometers, photometers, and thermometers.]

The project mainly focuses on Motion sensors i.e. accelerometer sensor. By reading raw data from Android API and using Python Script Layer for Android thereafter transmit the data over UDP protocol to the Raspberry Pi for processing.

The Scripting Layer for Android (abridged as SL4A, and previously named Android Scripting Environment or ASE) is a library that allows the creation and running of scripts written in various scripting languages directly on Android devices. SL4A is designed for developers and (as of March 2016) is still alpha quality software. [11]

When the application is launched a dialogue, box is displayed offering the user two (2) options [see Figure 2 Mobile Gesture Control] - Voice recognition and mobile shake control, upon selecting mobile shake control the logic on the flow chart is used.

When and if the mobile shake control option is selected, the application starts broadcasting its UDP connection to all ports open on the network, If the Raspberry Pi accepts the UDP connection then the Python script starts sending packets. The packets are a list of strings of raw accelerometer (X, Y and Z coordinates see Figure 5 3-Axis Accelerometer) data read from the Android device via the Android API at a sampling rate of 100ms. This activity is done every 500ms until the user decides to exit the application. There are no calculations done at this point.

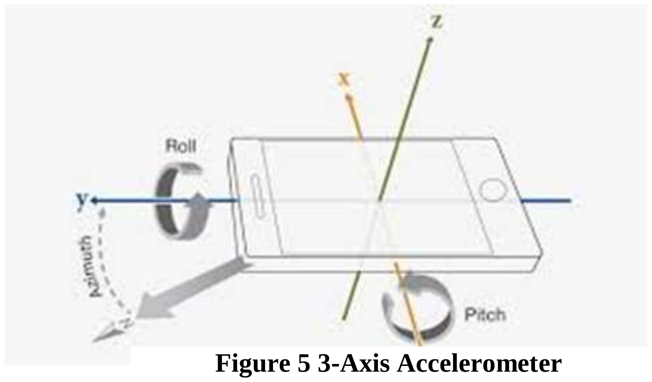


Figure 5 3-Axis Accelerometer

What is an accelerometer?
An accelerometer is a device that measures proper acceleration ("g-force"). Proper acceleration is not the same as coordinate acceleration (rate of change of velocity). For example, an accelerometer at rest on the surface of the Earth will measure an acceleration straight upwards. By contrast, accelerometers in free fall (falling toward the center of the Earth at a rate of) will measure zero. [12]

How to measure acceleration?
The accelerometer in the mobile device provides the XYZ coordinate raw values, which is used to measure the position and the acceleration of the device. The XYZ coordinate represents direction and position of the device at which acceleration occurred. The rotation direction and position are measured using gyroscope sensors that are found in the Android devices and exposed via the Android/Google API. The mobile device rest in the Earth includes the acceleration due to gravity and the acceleration value. The accelerometer values provided by the device normally includes the gravity as well. Accelerometer along with the linear acceleration and gyroscope will provide results close to accuracy. Linear acceleration does not include the gravity. [12]

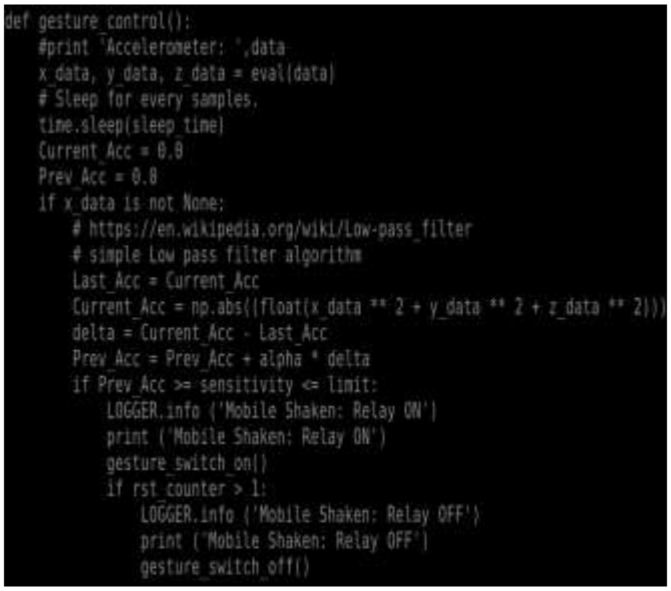
4) *Mobile gesture [back end]*

The Raspberry Pi runs a script that constantly listen for UDP packets on a certain open-port, this script is executed upon system boot with an exception of restarting the program in-case of any runtime failures.
Upon receiving the UDP packets sent from the Android smartphone as a list of strings with 3-raw accelerator sensor, the raw sensor data from the x, y, and z axes are analyzed and used to check the presence of movements. Here, we stabilize the sensor data values by filtering each value from the x, y, and z axes through a software based Low-pass filter. A low-pass filter is a filter that passes signals with a frequency lower than a certain cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency. The amount of attenuation for each frequency depends on the filter design. [13]

This discrete-time implementation of a simple RC low-pass filter is the exponentially weighted moving average and the following algorithm simulates the effect of a low-pass filter on a series of digital samples. [13]

$$y_i = \alpha x_i + (1-\alpha) y_{i-1} \qquad \text{where} \qquad \alpha \triangleq \frac{\Delta_T}{RC + \Delta_T}$$

The resulting values are compared with the pre-selected Threshold value to detect the presence of movements on each axis. The highest value among the detected values is the directional value for the gesture. Then, we decide if the smartphone was shaken by data values on each axis. We need to set up interval timings for measurements and need to determine the threshold value to detect movements as well as shake information.



```
def gesture_control():
    #print 'Accelerometer: ',data
    x_data, y_data, z_data = eval(data)
    # Sleep for every samples.
    time.sleep(sleep_time)
    Current_Acc = 0.0
    Prev_Acc = 0.0
    if x_data is not None:
        # https://en.wikipedia.org/wiki/Low-pass_filter
        # simple Low pass filter algorithm
        Last_Acc = Current_Acc
        Current_Acc = np.abs((float(x_data ** 2 + y_data ** 2 + z_data ** 2)))
        delta = Current_Acc - Last_Acc
        Prev_Acc = Prev_Acc + alpha * delta
        if Prev_Acc >= sensitivity <= limit:
            LOGGER.info ('Mobile Shaken: Relay ON')
            print ('Mobile Shaken: Relay ON')
            gesture_switch_on()
            if rst_counter > 1:
                LOGGER.info ('Mobile Shaken: Relay OFF')
                print ('Mobile Shaken: Relay OFF')
                gesture_switch_off()
```

Figure 6 Gesture control function

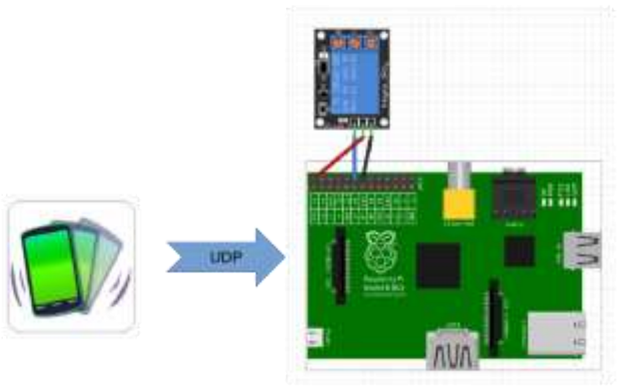


Figure 7 Mobile shake control

Figure 6 Gesture control function, a Python function, that retrieves the 3-axis accelerometer raw data passes it through a low pass filter. Effectively, this “smoothes/filters” the data by taking out the jittery, high-frequency noise and then compares the magnitude value to the sensitivity and limit value set upon start-up. If the value received detects a mobile switch then a relay is triggered on and if another mobile shake is detected again the relay is triggered off. A basic schematic and android communication is show above Figure 7 Mobile shake control See demo: https://www.youtube.com/watch?v=QxTiL1fr8xY

VI. References

[1] A. A. Atayero , A. A. Alatishe and Y. A. Ivanov, "Power Line Communication Technologies: Modeling and Simulation of PRIME Physical Layer," *IAENG,* pp. 931-936, 2012.

[2] E. Mainardi and M. Bonfè, "Powerline Communication in Home-Buildings," pp. 54-70, 2008.

[3] "Android Studio -Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Android_Studio [Accessed February 2017].

[4] "Meet Android Studio | Android Studio - Android Developers," [Online] Available: https://developer.android.com/studio/intro/index.html .

[5] "WebView | Android Developers," [Online] Available: https://developer.android.com/reference/android/webkit/WebView.html . [Accessed February 2017].

[6] "QPython - Python on Android," [Online]. Available: http://qpython.com/.

[7] R. Patel, "Voice Recognition," [Online]. Available: www.contactcenterarchitects.com/speech-recognition-vs-voice-recognition/.

[8] "Speech Recognizer class," [Online]. Available: https://developer.android.com/reference/android/speech/SpeechRecognizer.html .

[9] "User Datagram Protocol," [Online]. Available: https://en.wikipedia.org/wiki/User_Datagram_Protocol.

[10] "Keyword spotting | Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Keyword_spotting.

[11] D. Kohler, "Scripting layer for Android," [Online]. Available: https://en.wikipedia.org/wiki/Scripting_Layer_for_Android .

[12] "Accelerometer | Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Accelerometer .

[13] "Simple infinite impulse response filter | Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Low-pass_filter#Simple_infinite_impulse_response_filter.

VII. Authors

Mpho Mphego (BTech) is currently employed by the Square Kilometer Array, 3rd Floor, The Park, Park Road, Pineland, Cape Town, South Africa.

Department: Digital Back End under Correlator-Beamformer as a Software Test & Verification Engineer, Phone: +27761431543; email: mpho112@gmail.com or mmphego@ska.ac.za. He has been in the Digital Electronics industry for over 7 years and has a keen interest in Telecommunication, Control and Automation and Signal Processing.

Professor SP Daniel Chowdhury (M’02-SM’11, PhD(Eng), CEng, FIET, FIE, FIETE, SMIEEE, SMSAIEE) is presently with the Electrical Engineering Department, Tshwane University of Technology, Pretoria West, Staatsartillerie Road, Building 6-411A, Private Bag X680, Pretoria-0001, Phone:- +27 (0) 123825149; Cell:+27 (0) 713519332; Fax:-+27 (0) 123825688. (e-mail: spchowdhury2010@gmail.com). He has been in the profession of Electrical engineering for about three decades. He has graduated 10 Doctoral, 35 Masters, 48 Graduate students from Jadavpur University, Calcutta, University of Cape Town, Cape Town and Tshwane University of Technology, Pretoria with more than 10 current PG students. He has published more than 320 research papers in accredited international peer reviewed journals and conferences. He has co-authored the pioneering research book “Microgrids and Active Distribution Networks” published by the IET(UK) in 2009 in the renewable energy series and it has become the founding stone of global research in Renewable energy and grid integration as evidenced by its more than 540 Citations with its Chinese edition since 2015. He has successfully conducted and completed research projects in the area of energy research worth more than 2.5M Rupees in India, more than 11M ZAR in South Africa. He is presently directing his research and project works to alleviate energy poverty in Africa through smart microgrids.