Personal Health Monitoring with Bluetooth Sensors and Android Smartphones

AUTHORS **Tareq Alhamwi & Erdal Oruklu**

Introduction

The objective of this research project is to utilize a multi-modal biosensor platform incorporating multiple MEMS sensors for continuous vital signs monitoring, and integrate it with a smartphone unit for cellular network connectivity and communication.

Using Bluetooth interface, sensor data is transferred to smartphone devices equipped with Android operating system. This enables real time data analysis with smartphone devices. Smartphones are used for displaying incoming sensor data and performing signal processing. With the use of internet enabled smartphones, patients and doctors can be more in touch, and doctors can monitor/keep track of their patients' health status. The miniature size of the sensors allows them to be practical and convenient to be carried around in everyday activities. Since most patients already carry personal smartphones, health monitoring technology can be deemed affordable, requiring only Bluetooth enabled sensors and an application software.

Although several patient health monitoring applications exist, these solutions typically have a separate device to process the data and send it wirelessly. The goal of this research is to replace this extra hardware device with the patient's personal phone to do the processing and the sharing of the data. The proposed solution makes it easier for the patients to carry the device around, since it is comprised of only the sensors and an application software running on the host smartphone platform that we all carry in our daily lives.

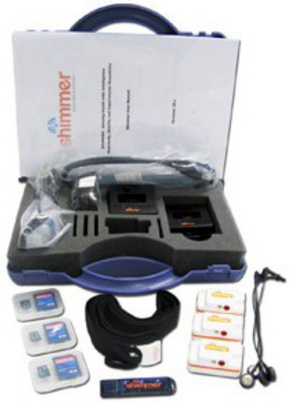
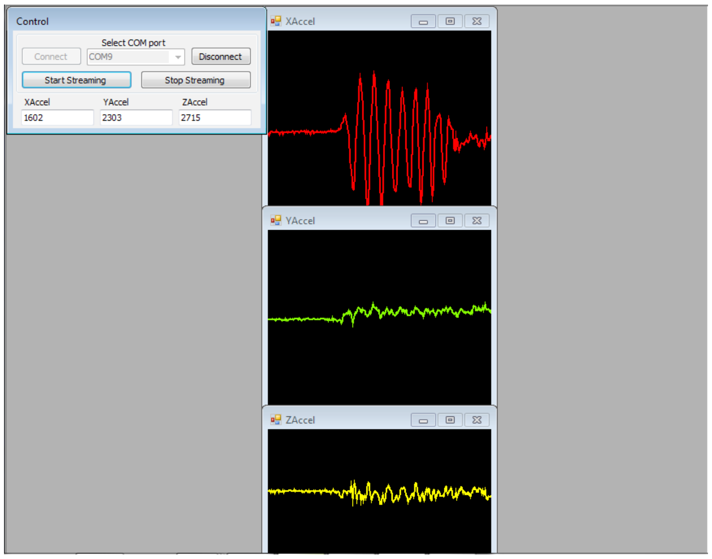


Figure 1. *Shimmer Bluetooth Medical Sensors [1]*

For the initial prototype, we will be using Shimmer starter kit [1]. As shown in figure 1, the kit comes with a flash drive that has all the programs and the manuals, programming dock for the sensors, a Bluetooth hub, and multiple sensors (ECG, EMG, Accelerometer, and Strain gauge), as well as a mini USB cable, and four electrodes.

Methods

The first task was getting familiar with the sensor kit. In order to connect the Shimmer sensors to the desktop computer and acquire sensor data, 'ShimmerConnect' software was used. 'ShimmerConnect' plots the data from the sensor. The software can connect to more than one sensor simultaneously, and saves the data in a CSV database file to allow processing on the data, or use of different plotting software. Figure 2 shows the screenshot with X-axis (top), Y-axis (middle), and Z-axis (bottom) of an accelerometer. In the screenshot, the sensor is being moved along the X-axis only, which explains the signals shown in the graphs.



COM9

XAccel	YAccel	ZAccel
1602	2303	2715

Figure 2. *Accelerometer sensor data acquired with Bluetooth connection*

In order to activate the Bluetooth connectivity on the sensor, a specific software package needs to be executed. Using the dock and the cable, the sensor is connected to the computer and initially 'bsl430' software is run. This software enables multiple packages to be downloaded to the sensor. In particular, choosing 'Boilerplate_shimmer2r' will enable the Bluetooth connectivity of the sensor.

The Shimmer kit uses an operating system called 'TinyOS', an open-source operating system designed for wireless devices [2], written in 'nesC' language (a dialect of C) [3]. Although nesC is derived from C, software development proved to be difficult due to lack of sufficient support. Therefore, all the development was done on the smartphone hardware running Android operating system. Working on the phone side was much easier, due vast support for Android app development.

The main obstacle of the software development phase has been establishing the Bluetooth connection between the phone and the sensor. Shimmer sensors are using the RFCOMM Protocol, which is very widespread and supported. In addition, it is a publicly available API. In Figure 3, two connection codes are shown, the first is RFCOMM, and the other is Insecure RFCOMM connection code.

The difference is that the first one is secure, meaning it prompts the user when establishing a communications channel, while the other (Insecure) establishes the connection without prompting the user to any security procedure, which makes them a threat to the "man in the middle".

```
if (secure) {
    tmp = mAdapter.listenUsingRfcommWithServiceRecord(NAME_SECURE,
        MY_UUID_SECURE);
} else {
    tmp = mAdapter.listenUsingInsecureRfcommWithServiceRecord(
            NAME_INSECURE, MY_UUID_INSECURE);
}
```

Figure 3. *RFCOMM connection example*

Both connection types take the same parameters (String Name, UUID), the string is just a name of the service, and can be set to any valid string. UUID which stands for (Universally Unique Identifier), is a 32 hexadecimal digits, grouped in five groups, this number is unique for every application with Bluetooth service. Both the host and the client device have to use the same UUID code, otherwise the connection won't be possible.

On the other device, the following lines of code should be executed:

```
if (secure) {
    tmp = device.createRfcommSocketToServiceRecord(
            MY_UUID_SECURE);
} else {
    tmp = device.createInsecureRfcommSocketToServiceRecord(
            MY_UUID_INSECURE);
}
```

Figure 4. *RFCOMM connection example*

This command initiates the connection between the two devices, using the given UUID. The device that is providing the server socket is the server, while the device that is initiating the connection is the client.

The second part of the software application development involves the use of ‘ShimmerConnect’ functions by Shimmer that feature the ability to store the data in a file. This application acquires the data, preprocesses it and displays it for the user on the smartphone screen. Furthermore, application can be written for streaming it to a pre-defined IP address (i.e. care-provider or doctor).

Results

Some preliminary results have been obtained with respect to both parts of the software developed. The first screenshot in Figure 5 is from the Bluetooth connection application. It has a simple layout, and it shows patient’s device’s Bluetooth name and MAC address as well as a list of the devices paired to the phone.

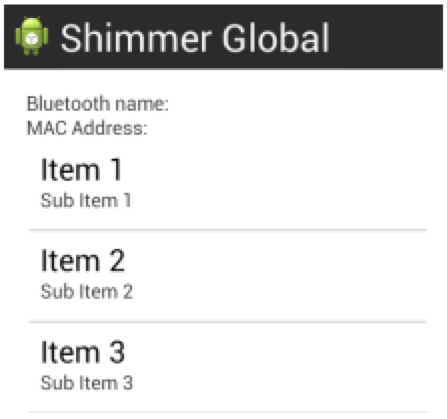
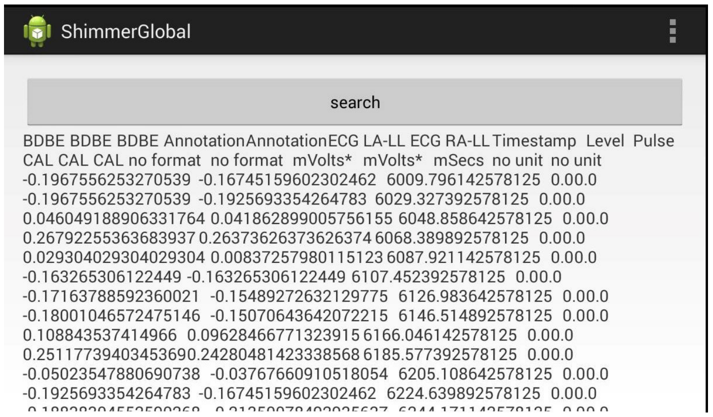


Figure 5. *Bluetooth connection screen*

Listing contains only the paired devices since Android OS requires the devices to be paired before exchanging data; i.e. data cannot be sent or received unless the devices are paired.

Once the search button is clicked, the application first checks whether Bluetooth is turned on or not. If it’s on, it goes on to the next step. Otherwise, it asks the user to turn it on in order to continue. After making sure that Bluetooth is on, a list of the paired devices is displayed. Next, the smartphone device is connected to the sensor once the user selects it from the list. Here, the user has the option to display or stream their medical data to their doctor/care-provider.

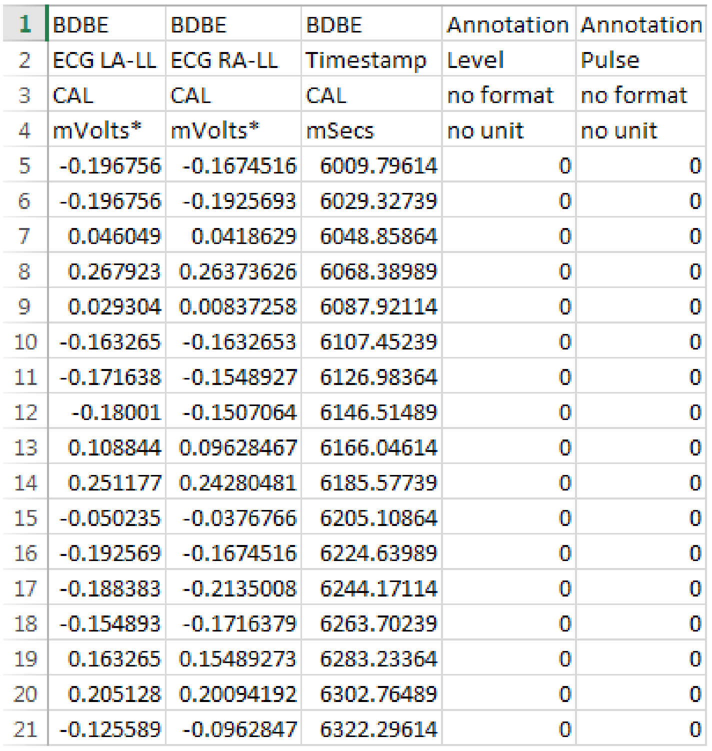
The second application we developed shows the data in its raw form. So far, the application acquires the data and displays it. The screenshot in Figure 6 shows the raw data.



```
BDBE BDBE BDBE AnnotationAnnotationECG LA-LL ECG RA-LL Timestamp  Level  Pulse
CAL CAL CAL no format  no format  mVolts*  mVolts*  mSecs  no unit  no unit
-0.1967556253270539  -0.16745159602302462  6009.796142578125  0.00.0
-0.1967556253270539  -0.1925693354264783  6029.327392578125  0.00.0
0.046049188906331764 0.041862899005756155 6048.858642578125  0.00.0
0.26792255363683937 0.26373626373626374 6068.389892578125  0.00.0
0.029304029304029304 0.00837257980115123 6087.921142578125  0.00.0
-0.163265306122449 -0.163265306122449 6107.452392578125  0.00.0
-0.17163788592360021  -0.15489272632129775  6126.983642578125  0.00.0
-0.18001046572475146  -0.15070643642072215  6146.514892578125  0.00.0
0.108843537414966  0.09628466771323915 6166.046142578125  0.00.0
0.25117739403453690.24280481423338568 6185.577392578125  0.00.0
-0.05023547880690738  -0.03767660910518054  6205.108642578125  0.00.0
-0.1925693354264783  -0.16745159602302462  6224.639892578125  0.00.0
```

Figure 6. *Raw ECG sensor data displayed on smartphone*

The following screenshot in Figure 7 is from an Excel file. The first row shows the data from the LA-LL and RA-LL electrodes, (LA stands for Left Arm. LL for Left Leg. RA for Right Arm). The second row says CAL which stands for calibrated, this is a feature in the application to calibrate the results for plotting. The results in the third column are simply the time, as time changes the results in both the first column and the second column change.



1	BDBE	BDBE	BDBE	Annotation	Annotation
2	ECG LA-LL	ECG RA-LL	Timestamp	Level	Pulse
3	CAL	CAL	CAL	no format	no format
4	mVolts*	mVolts*	mSecs	no unit	no unit
5	-0.196756	-0.1674516	6009.79614	0	0
6	-0.196756	-0.1925693	6029.32739	0	0
7	0.046049	0.0418629	6048.85864	0	0
8	0.267923	0.26373626	6068.38989	0	0
9	0.029304	0.00837258	6087.92114	0	0
10	-0.163265	-0.1632653	6107.45239	0	0
11	-0.171638	-0.1548927	6126.98364	0	0
12	-0.18001	-0.1507064	6146.51489	0	0
13	0.108844	0.09628467	6166.04614	0	0
14	0.251177	0.24280481	6185.57739	0	0
15	-0.050235	-0.0376766	6205.10864	0	0
16	-0.192569	-0.1674516	6224.63989	0	0
17	-0.188383	-0.2135008	6244.17114	0	0
18	-0.154893	-0.1716379	6263.70239	0	0
19	0.163265	0.15489273	6283.23364	0	0
20	0.205128	0.20094192	6302.76489	0	0
21	-0.125589	-0.0962847	6322.29614	0	0

Figure 7. *Raw ECG data displayed on Excel*

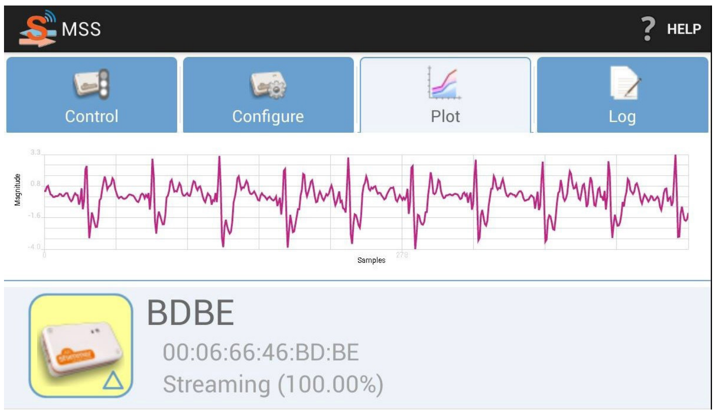


Figure 8. *ECG plot acquired with a smartphone*

Results as shown from the table, vary between positive and negative, which can be simply explained by looking at the graph in Figure 8.

The graph shows normal heart beats, and it's going between negative and positive values, depending on the magnitude value of the data from the sensor. After streaming the data to the doctor's computer, data can be plotted again. For demonstration purposes, we used MATLAB.

Conclusion

While working on this research, several difficulties were encountered; a big challenge was to program the sensor directly. The fact that the sensor uses 'nesC' language makes it harder to understand the right Bluetooth protocol for the sensor for establishing the connection. Overall, the sensor data acquired from the medical sensor have been verified to be valid and our custom Android application is capable of displaying the raw signals on the smartphone screen. For the next phase of the research, we will be emphasizing smart signal processing of the medical data with the smartphones prior to streaming to care provider. Smartphone based personal health monitoring applications provide the patients with the utmost convenience to constantly monitor their health as well as sharing it with their doctors instantly.

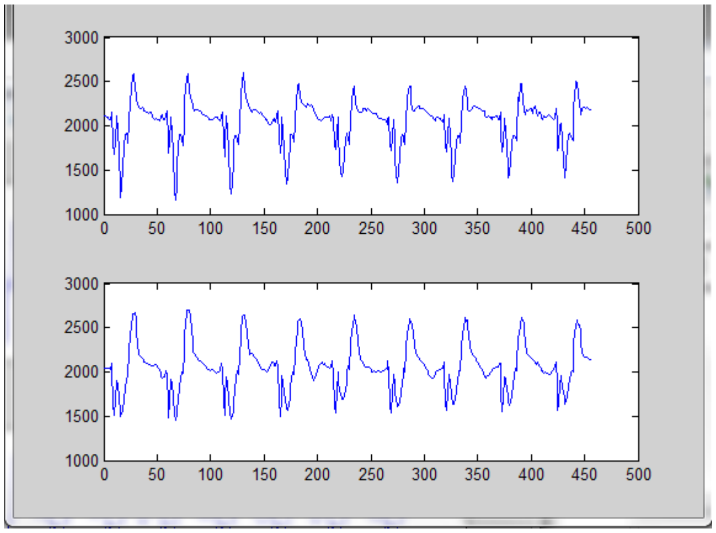


Figure 9. *Matlab plot for ECG data transmitted with a smartphone.*

Acknowledgements

We would like to acknowledge IIT Armour College of Engineering (ACE) Undergraduate Research Program which provided the funding that made this research possible.

References

1: http://www.shimmersensing.com/support/wireless-sensor-networks-download/

2: http://nescc.sourceforge.net/

3: http://tinyos.stanford.edu/tinyos-wiki/index.php/TinyOS_Overview