Int. J. Communications, Network and System Sciences, **2011, 4, 104-110**
doi:10.4236/ijcns.2011.42013 Published Online February 2011 (http://www.SciRP.org/journal/ijcns)

An Intelligent Data Filtering Scheme for Real Time Monitoring of Physiological Traits*

Israfil Biswas[1], Brendan Walker[2]
[1]*Electronics Research Group* (*ERG*), *University of Aberdeen*, *King's College*, *Aberdeen*, *UK*
[2]*Aerial and University of Nottingham*, *London*, *England*
E-mail: *israfil@erg.abdn.ac.uk*, *info@aerial.fm*
Received December 29, 2010; *revised February* 7, 2011; *accepted February* 10, 2011

Abstract

This paper illustrates a real time monitoring system to examine psychological behaviors over wireless networks and recommends a generic filter library for the system. Our goal is to develop "PsychoFIZZ"—a Python source project that communicates with multiple Digital Sample Units (DSUs) by filtering raw data and interpreting the data. The new filtering scheme should be able to work in coordination with the current monitoring system and our experiment results show that the library enhances the performance of the monitoring system.

Keywords: Sensors, Wireless Networks, Bluetooth

1. Introduction

Real time monitoring of psychological behaviour like heart rate, sweat response are becoming an important research of today's medical science and entertainment world. Personal monitoring of psychological traits would benefit individuals by providing continuous feedback about their stress levels and by helping their medical doctors to independently evaluate stress exposure between visits [1].

Monitoring of psychological events is also used for entertainment purpose like in "Thrill Laboratory" [2]. These laboratories are devoted to the practical quest of creating, producing and examining new forms of thrilling experience to entertain people. Each Laboratory experience may be different depending on the contributors, audience or venue but all with the same aim to create and examine new forms of exciting experience.

This paper describes "PsychoFIZZ"—a real time monitoring system of psychological traits from multiple subjects, where the data will be used in a variety of third party creative applications e.g. live theatre, adaptive theme park rides or in a reality TV project. PsychoFIZZ is based on gathering and analyzing medical data from multiple devices distributed across Bluetooth or WiFi networks.

Our goal of this project is to develop PsychoFIZZ for a new reality TV project based on wireless networks of sensors [3]. Wireless intelligent sensors have made possible a new generation of non-persistent, not noticeable personal medical monitors or thrill experience for entertainment applicable during normal activity.

Individual monitors will be integrated into a distributed wireless system for synchronized monitoring of a group of subjects. We use measures of heart-rate variability, galvanic skin response to quantify stress level prior to and during experiments as well as to predict stress resistance. This task requires reliable, high-precision instrumentation and synchronized measurements from a group of individuals over prolonged periods.

In this paper we propose a generic data filtering scheme by clipping the noise signals that captured by the DSU to enhance the performance of PsychoFIZZ. Then depending on the received data (*i.e.*, strength of the signals), the filtering scheme will be used for monitoring. Our results indicate that the new filter library can be deployed to extend PsychoFIZZ.

The remainder of the paper is structured as follows: section 2 provides a description about monitoring over wireless network and well known research in this area. Section 3 describes the hardware units and sensors have been used in this project. Section 4 illustrates the PsychoFIZZ monitoring system. Section 5 shows experiment results with the new filter library and the last section concludes with future works for this project.

*Work funded in the framework of the Horizon Digital Economy Hub by the University of Nottingham, UK.

2. Real Time Monitoring

Conventionally, monitoring was used to store information only in the world of medical science *i.e*. Electroencephalography (EEG), Electrocardiography (ECG) monitoring. However, the recent enhancements of wireless and mobile technology have laid a foundation for the new generation of wireless networks using sensors [4]. Hence, the same technology has made the monitoring more efficient that can provide real-time feedback to the feasible, either as a warning of impending medical emergency or as a thrilling experience during entertainment. Real time monitoring not only can significantly decrease the number of hospitalizations and nursing visits but also explores links between pleasure, arousal, performance, and emotional experience.

Mobility within the monitoring area became possible with the introduction of wireless local area network (WLAN) connectivity between sensors and storage server or PC. However, sensors in this type of system are still wired using a wireless Body Area Network (BAN). A wearable monitoring device using a personal area network (PAN) or BAN can be integrated into a user's clothing [5,6] although this is unsuitable for lengthy, continuous monitoring. On a larger scale, wireless network connectivity allows monitoring outside the venue using satellite and cell phone links.

The rapid development of hi-tech in low power microprocessors/microcontrollers, application-specific integrated circuits (ASICs), battery capacity, and wireless technology made possible the increased intelligence of monitors. Therefore, wireless connectivity of sensors has emerged on of the vital research trend and requires extremely low power consumption at the expense of lower communication range, and bandwidth. In addition, reliable transfer of data from the sensors is one of the most important tasks that the researchers are facing.

Already various research and commercial projects in the field of real-time monitoring have been done. A typical research project was *Warfighter Physiological Status Monitoring (WPSM)* that led by the U. S. Army Research Institute of Environmental Medicine (USARIEM) and the U.S. Army Medical Research and Materiel Command (USAMMRC) [7]. This model consists of sensors for heart rate, metabolic energy cost of walking, core and skin temperatures, GPS location, and activity/inactivity. The goal of the project is not only to provide medics with valuable information about wounded soldiers but also the final system is expected to be able to predict the critical aspects of performance under extreme conditions. To continuously monitor heartbeat rate using a photopplethysmograph (PPG) signal and send data wirelessly to a host computer, scientists at the d'Arbeloff Laboratory for Information Systems and Technology at MIT have developed a "ring sensor" [8]. Moreover, researchers at Kansas State University developed a wearable, Bluetooth-enabled portable health monitoring system [9].

Other related research includes recording of nerve and muscle signals from animals during their normal activity by developing a mini microcomputer from the University of Washington, Caltech, and CaseWestern Reserve University [10]. A device has been developed by the researchers at the Biomechanics-Laboratory of the Orthopaedic Hospital of the Free University of Berlin that measures hip joint forces and temperatures for the hip joint prosthesis [11].

3. Hardware Units and Sensors

This section describes the hardware units and sensors that have been used for this project. We have used Vilistus [12] Digital Sampling Unit (DSU) for this project. Our experiments start with Vilistus-8 DSU which has 8 channels for psycho-physiological monitoring and the feedback platforms utilizes Bluetooth 1.1 class 2 wireless communication. A screen shot of a development version of 8-channel Vilistus DSU is given in **Figure 1**. The DSU exploit an extended OpenEEG P3 data transfer protocol providing a maximum sample rate of 256 samples per second (SPS). The details about EEG protocol data for- mat is given in the next section.

The DSU provides a generalized amplification and data transfer bus for the various sensors that are provided with the unit. These include EEG, ECG, SC/GSR and skin tem- perature. All channels are available for all sensors and the Vilistus software provides the ability to alter both sensor types and data transfer rates dynamically. **Figure 2** represents a screen shot of Galvanic Skin Response (GSR) sensor.

The sensors can be used in any position within the sensor socket array and it is recommended to always use the same sensors in the same sockets. We have used the

Figure 1. Vilistus digital sampling unit (DSU).

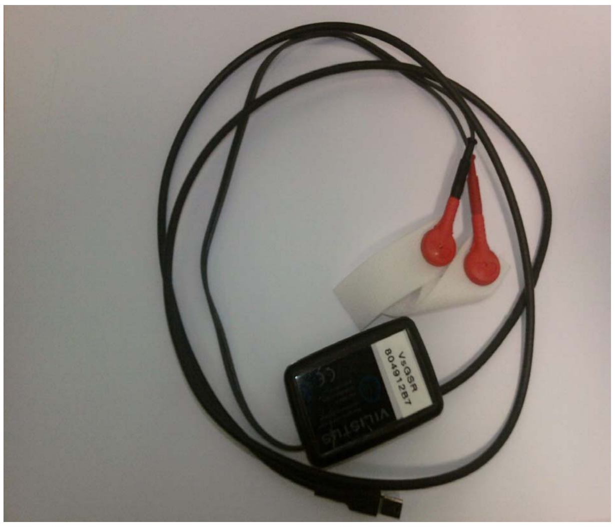

Figure 2. Vilistus GSR sensor.

default sampling rate of 256 sps for both GSR and EKG sensors. We have used EEG Ground Reference cable that came with the Vilistus DSU. The purpose of the ground reference is to remove common artifact from the active electrodes. The GSR sensor measures the arousal state of the client in terms of conductance, rather than resistance. Therefore, a relative change in skin conductance is measured by the sensor. The skin conductance reading generally declines during relaxation.

4. PsychoFIZZ

PsychoFIZZ is a system that receives flow of data from sensors and transfers the data into an understandable video output. **Figure 3** illustrates the dynamic real time monitoring system of "PsychoFIZZ". The Electroencephalography (EEG) data that exploited by the DSU is in ModularEEG packet format and under of the OpenEEG project [13]. P3 in the version number means this firmware is for packet version 3.

P3—Firmware Protocol is the ModularEEG Packet Format, which is the successor of the P2 protocol [14]. It provides a packet transmission that could achieve higher sampling rates with the same baudrate. One packet can have zero, two, four or six channels (or more).

The default is a 6-channel packet, shown below.

0ppppppx packet header
0xxxxxxx
0aaaaaaa channel 0 LSB
0bbbbbbb channel 1 LSB
0aaa-bbb channel 0 and 1 MSB
0ccccccc channel 2 LSB
0ddddddd channel 3 LSB
0ccc-ddd channel 2 and 3 MSB
0eeeeeee channel 4 LSB
0fffffff channel 5 LSB
1eee-fff channel 4 and 5 MSB

Key: 1 and 0 = sync bits. Note that the "1" sync bit is in the last byte of the packet, regardless of how many channels are in the packet.

p = 6-bit packet counter
x = auxillary channel byte
a-f = 10-bit samples from ADC channels 0 – 5
- = unused, must be zero

There are 8 auxillary channels that are transmitted in sequence. The 3 least significant bits of the packet counter determine what channel is transmitted in the current packet.

Aux Channel Allocations:

0: Zero-terminated ID-string (ASCII, currently "mEEGv1.0")

4: Port D status bits (Aux channels 1-3 and 4-7 are currently free)

When the system gathers and analyses medical data (e.g. heart rate, sweat response) from multiple sensors, the data are distributed over wireless networks through Bluetooth or WiFi. The DSU works through a COM port that has to be assigned manually and more than one DSU can be configured depending upon requirements. The assignment can be done after completing the pairing process with Bluetooth or WiFi connection.

PsychoFIZZ is an initial invite-only workshop is held to discuss the development of a flexible, low cost, open source platform for physiological monitoring. PsychoFIZZ is based on software commissioned for Self Examination at the Mayhem Horror Film Festival [15].

The project works in python environment and designed to work with Vilistus hardware that connected to a communication port. As motioned before, it parses extended P3 data packets that the system produces. The current system is combined of different threads and the purpose of each thread is described below:

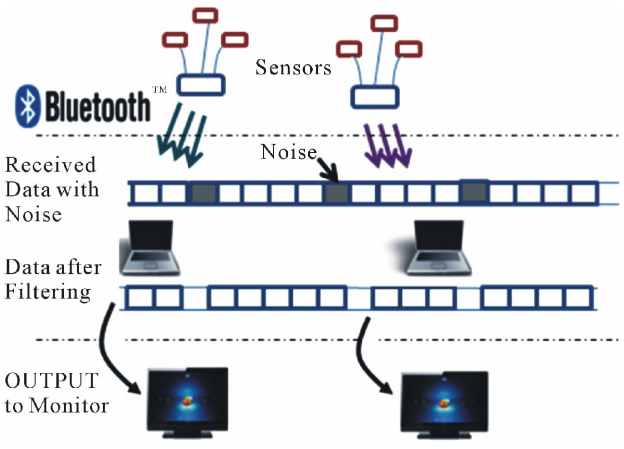

Figure 3. Dynamic illustration of PsychoFIZZ.

- The Serial Data Source reads from the given com port number a P3 data packet with a given number of channels as a string
- The Data Getter reads the data from the source and passes on a tuple of the channels' data
- The Data Collector is a thread safe data collector with averaging the data. This is also keeps the history of a channel
- The Data Filter will apply filtering while pass through it. The current set of filters in C++ was created using the very fine on-line filter code generating engine at the University of York [16]. An example code for the filtering can be found in Appendix A.

The output then can be used in live theatre, adaptive theme park rides like events. **Figure 4** shows the different working threads in the PsychoFIZZ.

5. Experiment Results

This section describes experiment results, when the data will be filtered after receiving from the Data Collector thread. The generic filter library is written in C++ and designed to work in two parts, 1) Noise filtering and 2) Data filtering. **Figure 5** shows the flow chart of the data filtering process. Firstly, when the data arrives for filtering, it checks for noise/unwanted signals and the noise filtering phase is activated. The noise filter clips the top and bottom of the signal to catch some types of noise. It very high or low signals received by the DSU.

The Data Filtering phase activated as soon as it passes the noise filtering test. Then it selects what type filtering is can be used depending on the sensor type. For instance, Bessel Lowpass filter can be used for GSR type sensor. The filters are just designed considering the mostly used filtering techniques from the past decades, but can be chosen depending on the requirements. The example filtering scheme can be used for any data that received from an EMG or GSR type sensors, and can be filtered for monitoring purpose.

We have used Bluetooth enabled DSU, an open wireless technology standard for exchanging data over short distances. Using short wavelength radio transmissions from fixed and mobile devices, Bluetooth creates personal area networks (PANs) with high levels of security nowadays. It can connect several devices, overcoming problems of synchronization and our project requires synchronization with the PC during retrieving data from a DSU.

This filtering scheme is an interactive filter design package, for designing digital filters by the bilinear transform or matched *z*-transform method [16]. For lowpass filter in our experiment, the number of poles is mentioned by the filter order, but can be changed depending on the order of the filter. Whereas, we considered the number of poles are twice the order in the bandpass filters [16].

All the graphs presented in this paper represents *x* axis as time, which is in samples (*i.e.* 250 represents 1 second) and *y* axis as filter response (linear, normalized). We have used a sample rate of 250 samples per second. Our design also considers the corner frequency for Butterworth andBessel filter design. For lowpass designs, the corner frequency is the frequency at which the magnitude of the response is –3 dB.

For Chebyshev lowpass designs, the corner frequency is the highest frequency at which the magnitude of the response is equal to the specified ripple. For highpass, bandpass and bandstop, the above definition is modified in an obvious way. For lowpass filter one frequency is used. For bandpass filters, two corner frequencies are used.

We have started our experiment by filtering some received data from the DSU after the Data collector thread. First we analysed data by passing through a ButterworthLowpass filter. Butterworth Lowpass because, this type of signal processing filter designed to have as flat a frequency response as possible in the pass band so that it is also termed a maximally flat magnitude filter [17], which can be useful for monitoring. **Figure 6** shows data filtering using Butterworth Lowpass filter.

In electronics and signal processing, a Bessel filter is a type of linear filter with a maximally flat group delay (maximally linear phase response). Bessel filters are often used in audio crossover systems. Analog Bessel filters are characterized by almost constant group delay across the entire passband, thus preserving the wave shape of filtered signals in the passband [18]. **Figure 7** shows filtering using Bessel lowpass filter. We can see the longer delay step response from Bessel lowpass filter compare to the Butterworth lowpass filter.

We have examined data using Butterworth bandpass

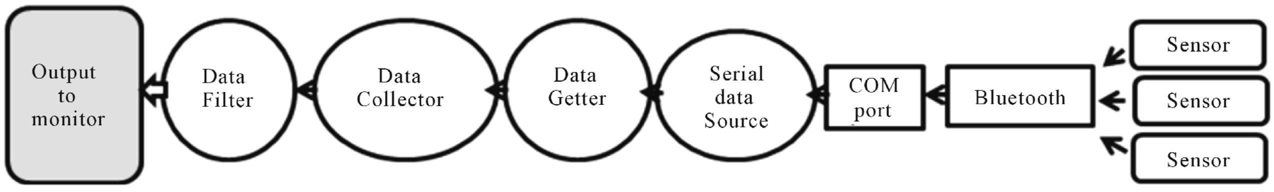

Figure 4. Illustration of the PsychoFIZZ system.

filtering (shown in **Figure 8**). Bandpass filter is used to re- move the noisy ECG or EKG signals [19]. The signals may be distorted at the output in Butterworth. Hence, we have tested data using Bessel filters (shown in **Figure 9**). In this circumstance, as the Bessel filters preserves the wave shape of filtered signals in the passband, it shows more steady response to the output.

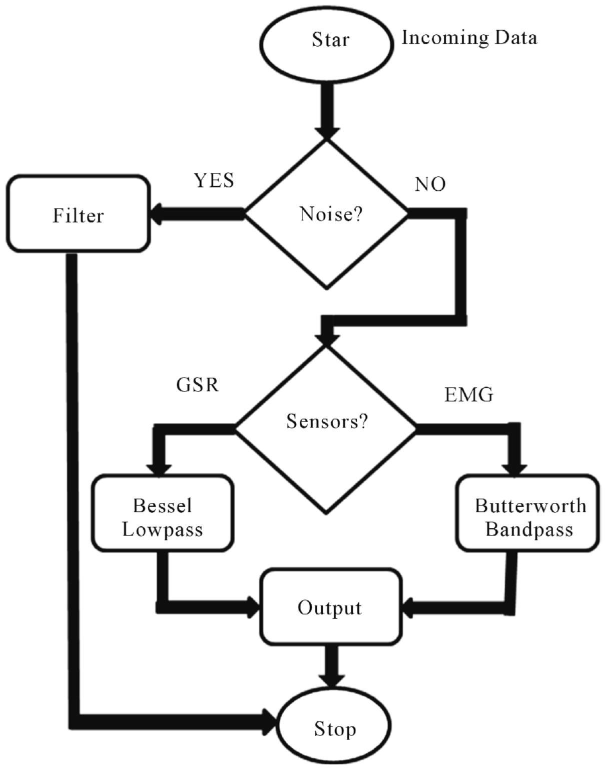

Figure 5. Illiustration of the PsychoFIZZ data filtering.

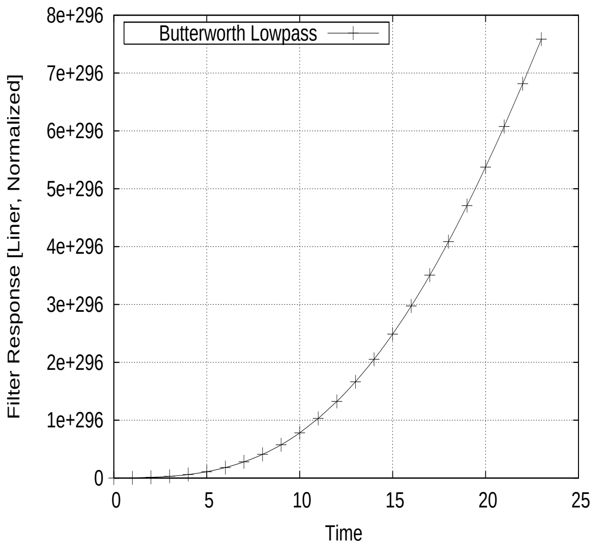

Figure 6. Data filtering using butterworth lowpass filter.

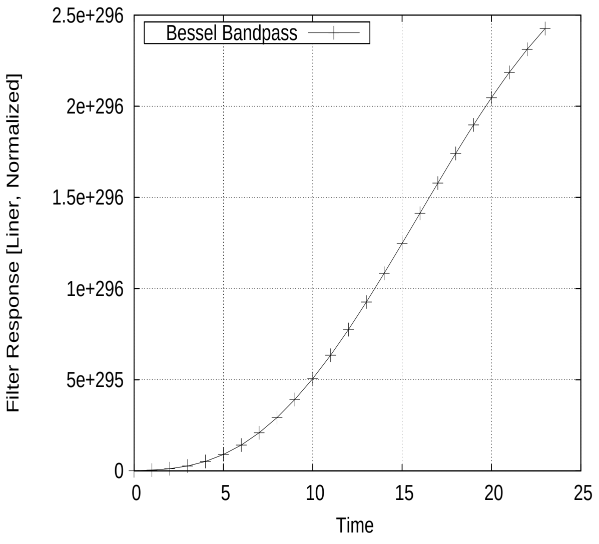

Figure 7. Data filtering using bessel lowpass filter.

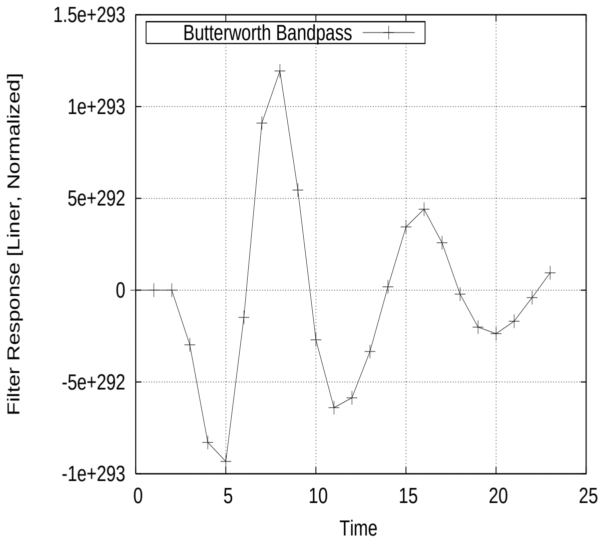

Figure 8. Data filtering using butterworth bandpass filter.

6. Conclusions

This paper introduces a new filter library to the existing real time monitoring system PsychoFIZZ with some experiments. The library is designed in C++, which is able to work over existing python project and any can be used with any C/C++ source project. In future the filter library can be used for other developments like GPS filtering with customized filtering schemes (*i.e*. Convolution Matrix filter [20]).

Bluetooth devices like DSU are still struggling to overcome problems of synchronization and our project requires synchronization with the PC during retrieving data from a DSU (as mentioned earlier). Therefore, in near

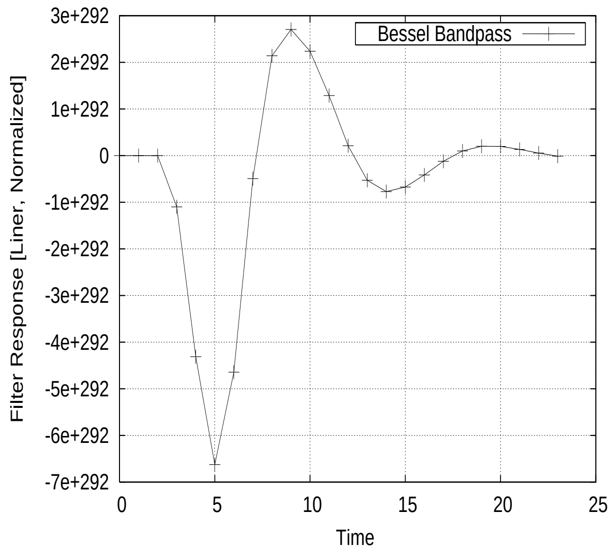

Figure 9. Data filtering using bessel bandpass filter.

future we are also looking forward to replace Bluetooth DSU with WiFi to do a set of tests.

7. Acknowledgements

The authors wish to thank Duncan Rowland, Paul Harter and Stephen Clark for their valuable suggestions and all time support. This work was supported by a grant from Horizon Digital Economy Hub at the University of Nottingham, UK.

8. References

[1] P. Heidenreich, C. Ruggerio and B. Massie, "Effect of a Home Monitoring System on Hospitalization and Resource Use for Patients with Heart Failure," *American Heart Journal*, Vol. 138, No. 4, 1999, pp. 633-640. doi:10.1016/S0002-8703(99)70176-6

[2] Thril Labratory, 2010. http://www.thrilllaboratory.com/

[3] DARPA Sensor Information Technology, January 2003. http://dtsn.darpa.mil/ixo/sensit.asp

[4] D. P. Agrawal and Q. Zeng, "Introduction to Wireless and Mobile Systems," Brooks/Cole Publishing Company, Pacific Grove, 2003.

[5] S. Park and S. Jayaraman, "Enhancing the Quality of Life through Wearable Technology," *IEEE Engineering in Medicine and Biology Magazine*, Vol. 22, No. 3, 2003, pp. 41-48. doi:10.1109/MEMB.2003.1213625

[6] M. Gorlick, "Electric Suspenders: A Fabric Power Bus and Data Network for Wearable Digital Devices," *Digest of Papers*, *The* 3*rd International Symposium on Wearable Computers*, San Francisco, 18-19 October 1999, pp. 114-121.

[7] R. W. Hoyt, J. Reifman, T. S. Coster and M. J. Buller, "Combat Medical Informatics: Present and Future," In: I. S. Kohane, Ed., *Biomedical Informatics*: *One Discipline*, AMIA, Bethesda, 2002.

[8] H. H. Asada, P. Shaltis, A. Reisner, S. Rhee and R. C. Hutchinson, "Mobile Monitoring with Wearable Photo-Plethysmographic Biosensors," *IEEE Engineering in Medicine and Biology Magazine*, Vol. 22, No. 3, 2003, pp. 28-40. doi:10.1109/MEMB.2003.1213624

[9] G. E. Barnes and S. Warren, "A Wearable, Bluetooth-Enabled System for Home Health Care," *Proceedings of* 2*nd Joint EMBS*/*BMES Conference*, Houston, 23-26 October 2002, pp. 1879-1880.

[10] C. Diorio and J. Mavoori, "Computer Electronics Meet Animal Brains," *IEEE Computer*, Vol. 36, No. 1, 2003, pp. 69-75.

[11] F. Graichen, G. Bergmann and A. Rohlmann, "Implantable Telemetry System for Measurement of Hip Joint Force and Temperature," *Proceedings of the* 15*th International Symposium on Biotelemetry*, Juneau, 9-14 May 1999, pp. 661-669.

[12] Vilistus, 2010. http://www.vilistus.com/

[13] The ModularEEG, 2010. http://openeeg.sourceforge.net/doc/modeeg/modeeg.html

[14] Christoph Veigl, BrainBay—Developer Manual, Version 1.0, 2006-08-31. http://shifz.org/brainbay/manuals/brainbay_developer_manual.pdf

[15] A Day in the Park: PsychoFIZZ, 2010. http://ditphorizon.blogspot.com/2010/03psychofizz.html

[16] Butterworth/Bessel/Chebyshev Filters, 2010. http://www-users.cs.york.ac.uk/~fisher/mkfilter/trad.html

[17] G. Bianchi and R. Sorrentino, "Electronic Filter Simulation & Design," McGraw-Hill Professional, Boston, 2007, pp. 17-20. http://books.google.com/books?id=5S3LCIxnYCcC&pg=PT32&dq=Butterworth-approximation+maximally-flat&lr=&as_brr=3&ei=SiyWSt_yH5jGM5TdidcH#v=onepage&q=Butterworth-approximation%20maximally-flat&f=false

[18] G. Bianchi and R. Sorrentino, "Electronic Filter Simulation & Design," McGraw-Hill Professional, Boston, 2007, pp. 31-43. http://books.google.com/books?id=5S3LCIxnYCcC&pg=PT53&dq=Bessel+filter+polynomial&lr=&as_brr=3&ei=gyeWSvTbIpmwNPyaqNcH#v=onepage&q=Bessel%20filter%20polynomial&f=false

[19] ECG Measurement System, 2010. http://www.cisl.columbia.edu/kinget_group/studentprojects/ECG%20Report/E6001%20ECG%20final%20report.htm

[20] Convolution Matrix, 2010. http://docs.gimp.org/en/plug-in-convmatrix.html

Appendix A: Example Code for Data Filtering

```
double CFilter_GEN::genericc(double input)
{
        double out = input;
//clips the top and bottom of the signal to catch some //types of noise
    if(input > 980.0 || input < 50)out = last;
    last = out;
    return out;
// For lower signal do bandpass filtering
    if (input < = 700.0)
    {
    double result = bandpass(abs(input - 512.0));
    if(result < 0.0)result = 0.0;
    return result;
    }
    else
    {
        double filtered = chasingThreshold (lowpass (input));
        return filtered;
    }
}
double CFilter_GEN::bandpass(double input)
{
    xv[0] = xv[1]; xv[1] = xv[2]; xv[2] = xv[3]; xv[3] = xv[4]; xv[4] = xv[5]; xv[5] = xv[6]; xv[6] = xv[7]; xv[7] = xv[8];
    xv[8] = input/2.783712976;
    yv[0] = yv[1]; yv[1] = yv[2]; yv[2] = yv[3]; yv[3] = yv[4]; yv[4] = yv[5]; yv[5] = yv[6]; yv[6] = yv[7]; yv[7] = yv[8];
    yv[8] = (xv[0] + xv[8]) – 4* (xv[2] + xv[6]) + 6* xv[4]
            + (–0.1291302835* yv[0]) + (0.2373236445 * yv[1])
            + (0.5188044618* yv[2]) + (–1.0922924416 * yv[3])
            + (–0.9388583822* yv[4]) + (1.6650807823 * yv[5])
            + (0.5116173874* yv[6]) + (–1.8476766755* yv[7]);
    return yv[8];
}
double CFilter_GEN::chasingThreshold(double input)
{
int i;
double sum1 = 0;
double sum2 = 0;
for(i = 0; i < 20; i++)
{
  av[i] = av[i + 1];
  sum1 + = av[i];
}
for(i = 20; i < 39; i++)
{
  av[i] = av[i + 1];
  sum2 + = av[i];
}
sum2 + = input;
av[39] = input;
double result = sum2 – sum1;
if(result < 0.0) return 0.0;
return result;
}
//For higher signals, do lowpass filtering
double CFilter_GEN::lowpass(double input)
{
xv[0] = xv[1]; xv[1] = xv[2]; xv[2] = xv[3]; xv[3] = xv[4]; xv[4] = xv[5];
    xv[5] = input/1.648922999e + 06;
    yv[0] = yv[1]; yv[1] = yv[2]; yv[2] = yv[3]; yv[3] = yv[4]; yv[4] = yv[5];
    yv[5] = (xv[0] + xv[5]) + 5* (xv[1] + xv[4]) + 10* (xv[2] + xv[3])
            + (0.6344519043* yv[0]) + (–3.4650204658 * yv[1])
            + (7.5796043901* yv[2]) + (–8.3013446171 * yv[3])
            + (4.5522893820* yv[4]);
    return yv[5];
}
```