Signal Quality and Data Visualizer for Emotiv EPOC

Author:
Uku Loskit
ukuloskit@gmail.com

Supervisor:
Ilya Kuzovkin

May 27, 2014
1 Introduction

Brain-computer interfaces (BCIs) are communication links between the human brain and computer systems which enable their users to control computers via their brain. BCI have been developed and used since the 1970s mainly for clinical and experimental purposes. Today, the prices of some BCI devices have become affordable for ordinary consumers, making way for developments for BCI technologies outside the scientific and clinical realms. In this report, we will focus on BCI techniques with the main focus on non-clinical applications, such as for use in simplistic scientific experiments and non-scientific pursuits.

There are several methods for measuring brain signals which differ greatly in their accuracy, resilience to noise, invasiveness and price. In the broadest sense, BCIs could be divided into two main categories: invasive and non-invasive. invasive techniques refer to methods which rely on performing operations on test subjects (typically humans, rats or monkeys) for gathering brain data, while non-invasive techniques do not. On the one hand invasive techniques are usually more accurate and provide a high resolution data. However, on the other hand they are limited in their applications, even in science, as invasive techniques, such as placing electrons in to the brains of human patients, is prohibited and also limited for other species.

Perhaps, the two most important non-invasive neuroimaging techniques for BCI are functional magnetic resonance imaging (fMRI), and electric encephalography (EEG). fMRI works by measuring brain activity by changes in associated blood flow. fMRI a very high spatial resolution and is resilient to noise, but the fMRI equipment is both expensive and bulky, making it impractical for most non-scientific and non-clinical applications. EEG neuroimaging technique works by recording the electrical activity along the scalp. It has a very high temporal resolution, but it is privy to noise, as there can be much interference between the scalp and the brain due to the cranium, intermediate tissues etc.

Whilst both techniques have their disadvantages and advantages, it seems that for non-clinical and non-scientific purposes EEG has several advantages over fMRI: it is cheaper and the devices are smaller, even wireless. In this report, we will look at a one (of many) consumer-grade BCI devices called Emotiv EPOC.
2 Emotiv EPOC

Emotiv Systems is an Australian company that produces consumer-grade BCIs which employ electroencephalography (EEG). Emotiv Systems markets two EEG consumer products: Emotiv two Emotiv EPOC neuroheadset and EEG neuroheadset. Emotiv EPOC is the cheaper version of the product, which is touted to be useful in consumer products such as augmenting computer games, controlling peripheral devices such as electric wheelchairs etc. The headset includes 14 electrodes plus two reference electrodes. The electrodes are connected to felt pads which are moistened in saline solution to improve electrical conductivity between the scalp and the electrodes.

3 Emokit

Since Emotiv is a proprietary commercial product which is bundled with a closed source application that comes with a licencing fee, its applications are somewhat limited due to mere licencing cost. Thankfully, a group of enthusiasts have created an open source project called Emokit that enables making use of the Emotiv EPOC headset without the proprietary software. The Emokit library was developed by Cody Brocious and Kyle Machulis. Emokit is a driver which exposes the raw data sent from the headset to the USB dongle.

The use of the raw Emokit driver is not without its problems, though. A major part of creating applications and using the headset for experiments are the diagnostics – how well are the electrodes placed on the subject’s head and how good is the signal quality. Emotiv SDK Lite is a free version of the Emotiv proprietary software, which is used for demonstrating the capabilities of the Emotiv EPOC. Unfortunately, it does not include a way to measure the signal quality, as the diagnostic program included with it does not work. Without the diagnostic information, it is very difficult to tell whether we are really observing some interesting phenomenon or are just seeing random noise. The Emokit library provides raw information about the signal, the values received range from 0-8000 $\mu$V for each channel (electrode).
4 Project

As stated previously, the main goal of this project to provide a graphical interface to Emokit that would display the information from the electrodes in real-time. This could prove useful for sanity checks and setting up scientific projects, as well as any other software development or integration for the Emotiv EPOC headset.

Figure 1: Emotiv TestBench: on the left, there are the electrodes, on the right, the 14 electrodes and realtime info of the signal

In addition to measuring the quality of the EEG and visualizing the signal, some more improvements could be made in terms of usability in comparison to the official Emotiv SDK. One of the most obvious shortcomings of the
official SDK is that the electrode signal quality indicator does not include the names of the electrodes, thus one cannot easily correct the location on the head without knowing which name corresponds to which electrode. One of the goals of the project is to be as platform-agnostic as possible. For this purpose, the Python programming language with the Tk toolkit (prepackaged with Python installations) will be used to ensure availability on the three major platforms, Windows, Mac and GNU/Linux. The Tk graphical framework allows us to easily embed interactive plots from the matplotlib library to visualize the output of different electrodes. The final application will be packaged with PyInstaller into a single binary (for each platform) in order to alleviate the troublesome dependency installation process. The graphical user interface produced as part of the project should be fairly similar to Emotiv TestBench, indicating the quality of the signal and the plot signal for each of the 14 electrodes in real time.

5 Results

The Emotiv Visualizer program was created in Python, powered by numpy, matplotlib and Tkinter. The graphical user-interface of the application can be seen below:
The plots on the left represent 14 electrodes in real-time. The image on the right updates the quality of signal in real time. The electrode colors indicate the quality of the signal at any given point in time, so the user of the headset may fit the electrodes on his head to improve the signal quality and get a more accurate reading. The electrode colors range from black (no signal) to red, yellow and finally green (perfect signal).

Despite my initial goal of creating a platform independent solution, I was only able to create a Linux-specific version. This was due to the way reading
from the Emotiv device is handled in the emokit library. The library does not expose a proper way to detect whether the USB dongle (nor the headset, for that matter) is connected or not, but fails ungracefully if the device is not present. This, however, cannot be properly handled by the library consumer. A possible workaround is to run the headset setup procedure in a separate process, and check the after a small timeout whether the started process failed with an error return code. Unfortunately, this approach was not adopted, as I hoped to make the appropriate changes in library, but failed to do so, mainly due to lack of time.

5.1 Installing the application

There are two options for installation. First of all, by just using the binary; this document is accompanied by a single GNU/Linux 64-bit executable visualizer which can be run without any prior installation. An alternative is to install

5.2 Running the application

The application can be run in two different modes: using real data and using mock data. The mock data mode does not require the Emotiv headset, but generates the corresponding data packets randomly. It is quite helpful for developing and debugging the application. To run the application in mock data mode supply the command line arguments visualizer –mockdata. Invoking the visualizer command without arguments attempts to read the data from the headset or waits for the headset to become available.

5.3 Distributing and building the application

5.4 Future work

5.4.1 Portability

The current visualizer software is distributed as 64-bit GNU/Linux binary. Due to limitations of Pyinstaller (no support for 32-bit cross-compilation), the executable only runs on 64-bit GNU/Linux operating systems. This limitation also requires the build toolchain to include a 32-bit operating system (more specifically the 32-bit version of the Python runtime).
5.4.2 FFT support

In terms future work, the visualizer program could also support the real-time plotting of the Fast Fourier Transform (FFT), as the official Emotiv SDK Configuration Manager does. This could be quite easily facilitated by pylab, which provides the FFT algorithm implementation and by complementing the current user interface with separate tabs for plain EEG signal and FFT.

5.4.3 User interface improvements

Improvements could be made in several respects. Firstly, the electrodes that are painted on the underlying image could be smoothened to make them look better. Secondly, the user should have

5.4.4 Reduce binary size

One of the negative aspects of using Pyinstaller is that the produced binaries are quite big. The binary size is highly inflated by including the whole Python runtime along with huge libraries such as numpy and matplotlib of which only a fraction is actually used. Even after employing the universal binary packer upx the size of the binary is remains over 50 megabytes. In comparison to Emotiv’s Windows installer, which is 31.6MB, but provides more functionality. Conversely, adding more functionality would not affect the binary size much, as the necessary libraries that take up most of the space, are there. Furthermore, downloading all of the dependencies separately would amount to much larger download size and would a much greater hassle.

5.4.5 Privilege escalation

On GNU/Linux (and probably on other operating systems as well), reading raw data from the USB (hidraw) device requires “administrator” privileges. Currently there is no user interface for privilege escalation and the application will fail ungracefully, if it is started with non-administrative privileges. A future improvement would be to either provide a user interface for escalating to the administrator privileges, or providing an automated way to grant the user permissions for accessing the Emotiv device. On GNU/Linux this could be achieved by installing the appropriate udev rules.
6 Conclusion

As a result of this project, a free and open source tool for visualizing and measuring the signal quality was created. While the project was somewhat successful, the final result still lacks very much in terms of portability, functionality and user-friendliness to provide a viable free alternative to the Emotiv SDK tool. This report laid a foundation for further development to improve the current solutions and its limitations and deficiencies.