31st Annual International Conference of the IEEE EMBS
Minneapolis, Minnesota, USA, September 2-6, 2009

Software for Biomedical Engineering Signal Processing Laboratory Experiments

Willis J. Tompkins, *Life Fellow, IEEE,* and J. Adam Wilson, *Student Member, IEEE*

***Abstract*—In the early 1990's we developed a special computer program called UW DigiScope to provide a mechanism for anyone interested in biomedical digital signal processing to study the field without requiring any other instrument except a personal computer. There are many digital filtering and pattern recognition algorithms used in processing biomedical signals. In general, students have very limited opportunity to have hands-on access to the mechanisms of digital signal processing. In a typical course, the filters are designed non-interactively, which does not provide the student with significant understanding of the design constraints of such filters nor their actual performance characteristics. UW DigiScope 3.0 is the first major update since version 2.0 was released in 1994. This paper provides details on how the new version based on MATLAB™ works with signals, including the filter design tool that is the programming interface between UW DigiScope and processing algorithms.**

I. Introduction

UW DigiScope is special software that provides a mechanism for anyone interested in biomedical signal processing to study the field without requiring any other instrument except a personal computer. It is a program that gives the user a range of basic functions typical of a processing digital oscilloscope that was originally written to support the concepts in a biomedical engineering signal processing textbook [1]. These include features such as data storage, adjustment of sensitivity controls, and display of waveforms. More important, this program also provides a digital signal processing package with a set of libraries and a pre-sampled signal database. It includes graphical tools for designing FIR, IIR, and integer filters. In the filter design tool, magnitude, phase, and pole-zero plots as well as a graph of the unit impulse response assist the student in visualizing the design. Libraries for ECG processing include modules that facilitate QRS detection, signal compression, and waveform generation.

We wrote the previous versions of DigiScope for the outdated MS DOS environment. They are increasingly incompatible with newer computer systems such as Windows XP and Windows Vista and were not available on other platforms such as Mac OS X or Linux without a DOS emulator. Version 3.0 features a complete re-write of the program using the MATLAB™ programming language, providing cross-platform support for Windows, Macintosh, and Linux systems, and complete code transparency, allowing students to observe the inner-workings of the DigiScope system. Furthermore, DigiScope 3.0 was designed to easily allow students to add additional signal processing algorithms, or modify existing ones, that can be used within DigiScope.

DigiScope provides an environment in which the student can do lab experiments, design digital filters of different types, and visualize the results of frequency and time-domain processing of sampled signals. Included is a library of sampled signals and a signal generation tool, as well as the capability for the student to save processed signal files and filter designs.

There are many digital filtering and pattern recognition algorithms used in processing biomedical signals. In a typical biomedical engineering course, students have limited real-world signal processing design opportunity. They can run digital signal processing algorithms and can analyze the operation of these algorithms. However, it is not easy to implement new algorithms. Thus, a student has limited opportunity to design and program a processing algorithm.

II. UW DigiScope 3.0 Example

Figure 1 shows the main display screen of UW DigiScope 3.0. The screen shows an ECG read from a file displayed in the top channel and the results of processing the ECG with three different actions in the three channels below the original signal channel. The upper left window shows the action chain, the sequence of actions shown in the channel on the right.

In this case, the first action, SIGNAL, corresponds to the signal in the top channel on the right. This particular pre-digitized ECG signal was loaded from a file. Alternately, signals like sine and square waves can be produced using an included signal generator or created from templates such as single ECG wave complexes that can be repeated to produce a synthesized ECG.

The next action, Add Noise, provides a tool to add 50-Hz, 60-Hz, or random noise of specified amplitude to the signal of the top channel. The resulting signal appears in the channel beneath the signal. The third action, Power, generates the power spectrum of the signal plus noise that appears in the third channel clearly showing a peak at 60 Hz. The final action, Filter Designer, is the result of a filter design shown in Fig. 2. The filter design tool provides for designing filters of different types including FIR and IIR filters. In this case, the coefficients of an FIR were entered to produce the notch filter shown by the equation at the bottom

Manuscript received April 23, 2009. W. J. Tompkins is with the Department of Biomedical Engineering, University of Wisconsin-Madison, 1550 Engineering Drive, Madison, WI 53706 USA phone: 608-263-1581; fax: 608-265-9239; e-mail: tompkins@engr.wisc.edu.

J. Adam Wilson is with the Department of Biomedical Engineering, University of Wisconsin-Madison, 1550 Engineering Drive, Madison, WI 53706 USA e-mail: jawilson2@wisc.edu.

978-1-4244-3296-7/09/$25.00 ©2009 IEEE

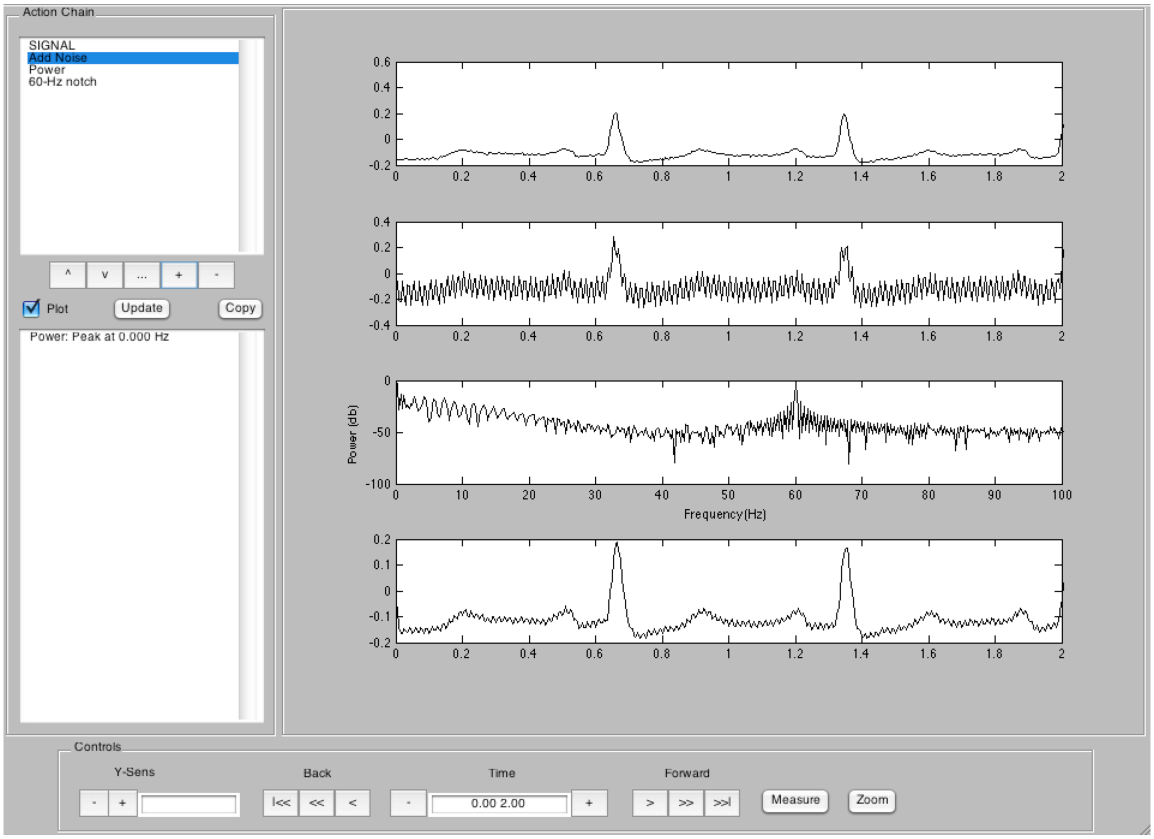

Fig. 1. UW DigiScope's main display screen. In this case, the ECG in the top channel was read from a file, then the second channel shows 60-Hz noise added to original ECG, the third channel show shows the power spectrum of the second channel with a peak at 60 Hz, and the bottom channel is the signal after processing with a 60-Hz filter (see Fig. 2 for the 60-Hz filter design).

of the right panel. Graphical panels show the magnitude response, phase response, impulse response, and pole-zero plot.

III. Extending UW DigiScope 3.0

UW DigiScope 3.0 uses a signal processing plug-in framework that allows new algorithms and visualization methods to be easily introduced into DigiScope. In order to create a new algorithm, the student can edit a plug-in template file using the MATLAB™ programming language, and save it in an appropriate DigiScope folder. When DigiScope is started, the new algorithm will be available for use. In this manner, students can easily test and debug new computational methods and instantly see the results visually. Furthermore, this framework allows DigiScope to be extended beyond ECG analysis to other biological signals, such as the electromyogram (EMG) or the electroencephalogram (EEG), simply by creating the appropriate plug-ins.

IV. Conclusion

We hope that the use of UW DigiScope 3.0 on signal processing workstations will greatly enhance the student's hands-on design experience. The software supports a set of lab experiments. The electrocardiogram (ECG) is the model biomedical signal used for demonstrating digital signal processing techniques. By supporting a set of ECG laboratories, DigiScope helps students to learn about digital filter design and signal processing algorithms.

Acknowledgment

The following UW-Madison graduate students contributed to the initial design of UW DigiScope 3.0: Bernhard Buchli, Daniel Chang, Philip Garcia, Jason Greenwood, Mitchell Hayenga, Neil Hockert, Chris Jenkins, Robert Kohn, Al Mashal, Andrew Nere, Evan Rogers, Justin Rosenthal, Sean Sengele, and Steven Skroch.

Software Availability

UW DigiScope 3.0 is available to be downloaded for free from Sourceforge (http://digiscope.sourceforge.net/).

References

[1] W. J. Tompkins (ed.) *Biomedical Digital Signal Processing,* Prentice-Hall, 1993.

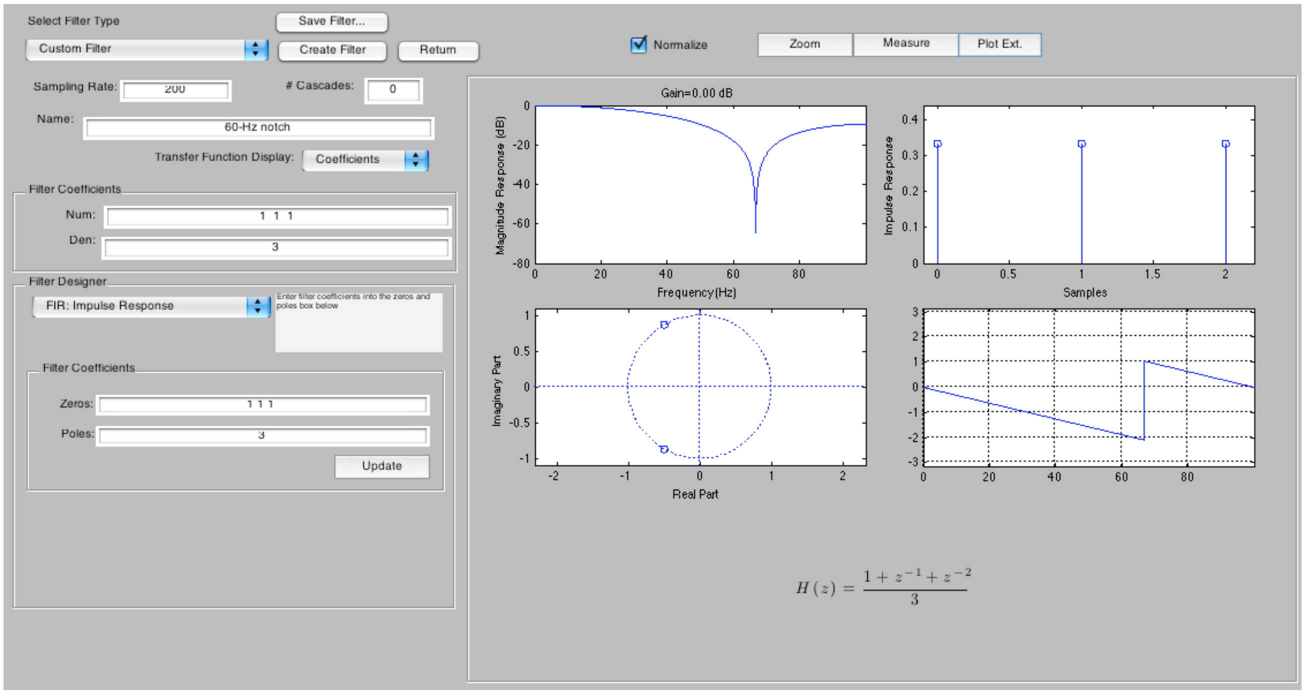

Fig. 2. UW DigiScope filter design screen image showing the design of the 60-Hz FIR filter that produced the signal in the bottom channel of Fig. 1.