

# VISUALIZATION OF SIGNAL PROCESSING CONCEPTS

*Janna Shaffer, Jonathan Hamaker, Joseph Picone*

Institute for Signal and Information Processing  
Department of Electrical and Computer Engineering  
Mississippi State University, Mississippi State, Mississippi 39762  
{shaffer, hamaker, picone}@isip.msstate.edu

**Abstract** - One of the key difficulties in a Signals and Systems course is the visualization of mathematically complex concepts presented. Thus, there is a need for graphical tools which enhance the students' comprehension of these difficult concepts by allowing interactive learning. In this paper we present a software package to assist in the explanation and visualization of signal processing concepts for an educational environment. We provide a set of Java-based tools for understanding the concepts of convolution, spectral analysis, and pole/zero system response. The wide availability and platform-independence provided by Java make this tool highly portable and easily accessible to a broader audience of students than comparable systems based on Matlab or other commercial software. The software described in this paper is available in the public domain at our website: <http://isip.msstate.edu/>.

## 1. INTRODUCTION

The goal for any engineering course is to convey to student engineers the knowledge and skills which will make them valuable to industry. The job market calls for engineers with a broad understanding of the mathematics and theory underlying engineering concepts, but also, engineers who are able to apply those concepts with ingenuity and expediency [1]. The second of these skills must be learned by repeated application of the concepts over a wide variety of test conditions. In doing so, the student builds an intuition for the problems involved. For many years, the primary method of teaching engineering concepts has come in the form of classroom interaction and homework exercises. These methods do an adequate job of conveying mathematics but often leave the student lacking the intuition necessary to apply the concepts.

The main stumbling block in the building of intuition about a concept is the student's inability to visualize the items being learned. Many concepts have complex visual representations which can not be sufficiently explored through the standard classroom experience. One example is the relationship between the poles and zeros of a system and the three-dimensional surface representing the system's response as shown in Figure 1. This phenomenon

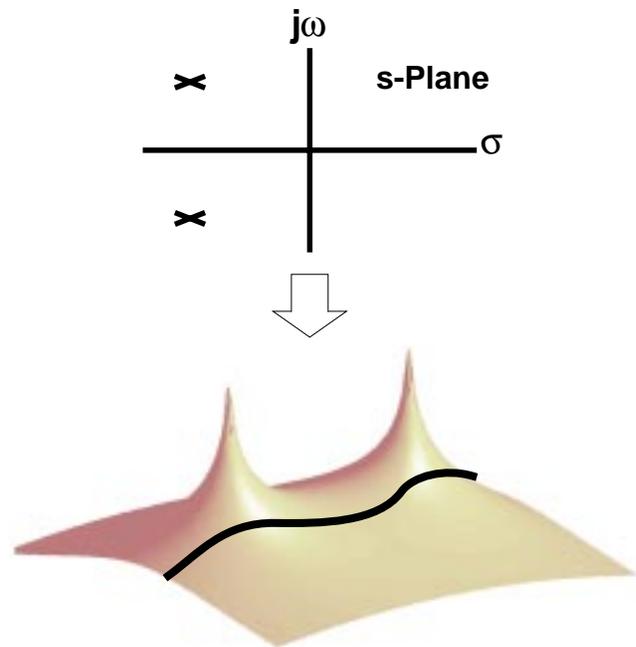


Figure 1. Three-dimensional frequency response of a linear system.

is usually sparsely covered in textbook problems if not ignored altogether. Yet, it is an important concept in the understanding of linear systems. To build intuition on these concepts would require the working of an impractical number of mathematically overwhelming exercises.

For these reasons, a need arises for a set of tools which provide students the opportunity to visually and interactively explore the classroom concepts without use of pen or paper. There has been increasing demand within the educational community for such tools which give the student full control of the learning process. With these, the student would no longer be constrained to the textbook examples, but would rather be immersed in the learning process. In doing so, the student would gain both insight and intuition as well as an increased retention.

## 2. IMPLEMENTATION

We have addressed these needs by developing a Java-based

[2] set of tools that incorporates interactive interfaces to such signal processing concepts. Each tool gives the user full control of the learning process by providing hands-on manipulation of the concepts explored. We have written these tools in the platform-independent Java language, allowing access by the widest audience possible.

### 2.1. Traditional solutions

Commercial packages such as Matlab, Visual Basic and Visual C++ or public domain systems like Tcl/Tk have often been used in the development of such tools. However, each of these lacks at least one component which is vital when preparing software for consumption by students. Specifically they are sparse in either portability, accessibility or both.

Matlab is highly portable, yet it requires a commercial product to run the programs. Furthermore, Matlab requires that copies of the interface software be kept on the user's local machine. This is a problem because a number of students do not have the resources to buy a copy of Matlab. Of course, the university could supply computers with Matlab installed, but this would limit the accessibility to students as they would be forced to work on those computers rather than from home.

Portability is the issue for Visual Basic and Visual C++. Both have nice facilities for building graphical interfaces, yet they are available on a limited number of platforms. Also, the students would be required to compile the software on their local computer in order to use the tools. An additional consideration is the support of software written in these languages, since the development of cross-platform utilities would essentially require different copies of the software for each platform.

Tcl/Tk is another consideration when developing user interfaces. Accessibility is again the problem with this as it requires a copy of the software scripts to be on each user's computer. Speed is also an issue with Tcl/Tk since it is an interpretive rather than compiled language. It has proven to be too slow for a number of applications we have developed.

### 2.2. The Java solution

Java provides the accessibility and portability lacking in the approaches examined earlier. It is a platform-independent language [2] developed at Sun Microsystems to take full advantage of the internet by allowing a single compiled program to be run on many different machines.

Java takes care of the portability problem by virtue of its

platform-independence. Compiled Java programs are not executables, but rather byte-compiled code which can be held remotely and executed locally. This is accomplished through use of the Java "Virtual Machine". This is the portion of the Java system which interprets the bytecode and executes it in the local machines natural language.

Java addresses the accessibility issue in two ways. First and foremost, Java is free. This is an important consideration when designing software aimed toward students as they will be more inclined to use a tool which is freely available than one which is costly. Secondly, Java is incorporated into the most popular web-browsers. There are Java-capable web-browsers available for every major platform. This is also important since students will be more likely to use a tool when it is viewed as "surfing the web".

There are some difficulties with the portability of the Java graphics that plague most all interface building environments (including Matlab and Tcl/Tk). The problem is its dependence on the low-level widget implementations. For instance, the Java button widget uses the button defined in the graphics package provided with the operating system. This causes problems when developing the interface because the look of the GUI on one platform may be somewhat different on another. Thus, it is necessary that the developer test the design on the platforms of interest.

## 3. A SIGNALS AND SYSTEMS PACKAGE

The ultimate goal of this project is to provide a comprehensive set of tools for use in undergraduate Signals and Systems and DSP courses. To date, we have created utilities which cover three of the most difficult topics to understand visually: graphical convolution, spectral analysis, and pole/zero system response. Each tool provides the user with a graphical interface for direct manipulation of concepts involved. Furthermore, each was designed such that the interface did not interfere with the learning process. It was our desire that a novice user could begin efficiently using the tool in a short amount of time. In other words, the user was given as much control as possible without being overwhelmed by options.

### 3.1. Graphical Convolution

Graphical convolution is the method by which the convolution integral can be evaluated visually. However, this complicated process is often difficult for students to understand. For this reason, we have developed a utility [3] which performs graphical convolution automatically. The interface allows the student to specify the two functions to be convolved by either drawing in the signal or choosing from a list of common signals. The program then

incrementally displays the construction of the graphical convolution process as well as the output. In this manner, the student is able to both check his or her handwritten work or can work with more complex signals which would be impractical for a classroom assignment.

### 3.2. Spectral Analysis

Spectral analysis is one of the most important ideas in the Electrical Engineering curriculum, having application in numerous fields. Yet, it is not an intuitive process and is rarely understood by examining mathematical formulae. Particularly difficult are the concepts of phase response and the effects of windowing on the spectrum.

We created a tool which addressed these problems by performing the spectral analysis (phase and magnitude response) of a user-defined signal automatically [4]. Also included is the ability to apply a window function to the input signal and view the windowed spectrum. Some key features of the tool include: interactive drawing of the input signal and the ability to use speech data files as input. Additionally, the spectrum analysis tool allows the user to select portions of the signal to be windowed so that the effects of assumed periodicity can be examined.

This tool presents a quick and effective way of demonstrating certain properties of the Fourier Transform such as time/frequency duality and the time delay equivalency to phase shift. This is invaluable since these properties are difficult to comprehend using only formulas. These components, together, provide the backbone for an increased understanding of signal spectral analysis.

### 3.3. Pole/Zero System Response

Poles and zeros are fundamental to any linear system. They are building blocks that can be used to describe the frequency domain and time domain responses. Knowledge of the poles and zeros of a system completely characterizes that system. Commonly, the phase and magnitude responses of linear systems are shown in a two-dimensional plot versus frequency. However, the pole/zero response is not a two-dimensional response but rather a three-dimensional surface produced by the interaction of the poles and zeros. A sample of this surface is shown in Figure 1. The two-dimensional response is a slice of the three-dimensional response along the imaginary frequency line as shown by the dark line in Figure 1.

There are a number of methods available for determining the two-dimensional response (bode plots, for instance) of a system given its characteristic poles and zeros. However, there are no methods which visually relates a pole or zero's

location in the s-plane and z-plane to the shape of the three-dimensional surface.

The pole/zero system response tool [5] provides three important functions. First, is the ability to manipulate the poles and zeros directly on the pole/zero map. Second, is the interactive analysis of the magnitude response, phase response and impulse response of the system described by the set of poles and zeros. Lastly, and most importantly, is the ability to visually explore the three-dimensional response of a system based on the poles and zeros of the system. This is presented as a surface plot. An example of the interface for this tool which is representative of the other tools is shown in Figure 2.

## 4. CONCLUSIONS AND FUTURE WORK

In this work we have presented a software package developed to aid in the visualization and comprehension of Signals and Systems and DSP concepts. These tools take a different approach in that they use Java in the place of the traditional commercial packages. We also provide a set of hands-on tools for exploring the important concepts of convolution, spectral analysis, and system response.

Though this work represents a good beginning, there are many more tools yet to be developed. In the near future we plan to write utilities which explore such concepts as Nyquist criterion, linear discriminant analysis, correlation, and spectrogram analysis. Note that these utilities are freely available at our web site: <http://isip.msstate.edu/>

## 5. REFERENCES

- [1] T. N. Trick, "Educating Electrical and Computer Engineers for the Global Renaissance," *Engineering Education*, vol. 83, pp. 57-62, 1994.
- [2] J. Gosling and H. McGilton, "The Java Language Environment: A White Paper," available at <http://java.sun.com/>, Sun Microsystems, 1996.
- [3] E. Wheeler, "Graphical Convolution in Java," available at [http://isip.msstate.edu/publications/1996/ee\\_4012/](http://isip.msstate.edu/publications/1996/ee_4012/), 1996.
- [4] J. Shaffer, "Spectrum Analysis Using Java," available at [http://isip.msstate.edu/publications/1997/ee\\_4012/](http://isip.msstate.edu/publications/1997/ee_4012/), 1997.
- [5] J. Hamaker, "Tutorial for the Java-based System Response Tool," available at: [http://isip.msstate.edu/software/java\\_system\\_response/tutorial/tutorial.html](http://isip.msstate.edu/software/java_system_response/tutorial/tutorial.html), 1997.

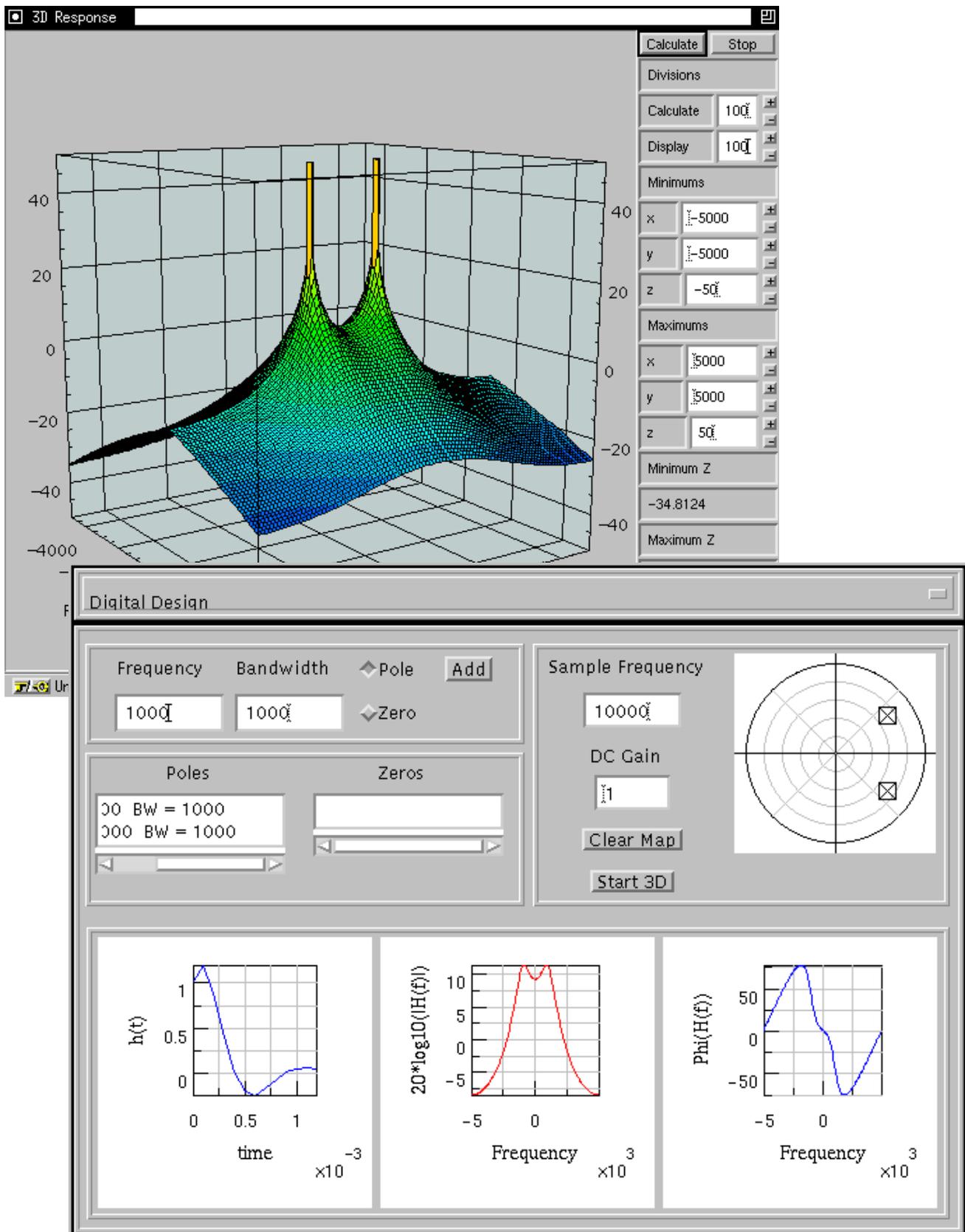


Figure 2. Pole/Zero system response applet showing a complex conjugate pole pair.