# TOWARDS THE INTEGRATION OF COMPUTER-BASED INSTRUCTION WITH MATLAB®

**Nizar Al-Holou and N. Mohankrishnan**
**Department of Electrical and Computer Engineering**
**University of Detroit Mercy**
**Detroit, MI 48219**
**alholoun@udmercy.edu**

Abstract

Authorware® is a powerful presentation medium for the design of Computer-Based Instruction modules. However, one limitation is that it is not very suitable for the use of open-ended problem formulations which require the learner to make parameter choices in the solution process; such choices have to be pre-determined by the module designer and, consequently, inhibit the learning process. On the other hand, this is the strength of many computational software tools such as Matlab®. The objective of this report is to describe a first step in the establishment of such a bridge between Authorware® and Matlab® and to discuss the attendant educational benefits.

## I. Introduction

The continuing advances in computer technology such as increasingly faster CPUs, cheaper RAM memory, enhanced CD-ROM capabilities, etc., in combination with emerging standards for picture compression and the availability of full-motion digital video, have inspired efforts by educators to investigate alternative processes to educate students, in particular the non-traditional ones. One such alternative approach that is increasingly being considered at a number of educational institutions is known as Computer-Based Instruction (CBI). This instructional approach takes advantage of the computer-enabled combination of text, still and moving images, and sound.

One advantage of CBI from the learner's point-of-view is that it is self-paced and interactive, and students are able to follow an instructional pace geared to their individual convenience. But additionally, from the instructor's perspective, it has other valuable features such as a logical, changeable (on demand) presentation flow of information, and the ability to track student progress throughout the instructional program by recording test results. This powerful combination of functionality may increasingly, over time, enable it to replace some aspects of traditional classroom instruction and learning. CBI delivery will reduce the interruption of work schedules for students who are also working professionals, by eliminating the need to travel to school or training center sites. In addition, it can also save time and effort for traditional students as well.

Well before the advent of the CBI trend, publishing houses were bundling powerful computational software such as Matlab, Mathcad®, etc. [1], with some of their engineering textbook offerings. In general the inclusions were such that it enabled fairly complicated numerical examples to be covered in a manner concurrent with the exposition of the theory, and in an interactive framework. It thus provided a connectedness between theory and problem solving in the classroom that did not exist before; earlier on this bridge was generally established by using simplified examples, or canned non-interactive expositions of more complex problems. The birth and evolution of the CBI method in engineering education took place in such an environment.

One of the most widely used software tools for designing CBI modules today is Authorware Professional for Windows [2]. While Authorware has very sophisticated presentation capabilities, it lacks the computational power of Matlab. This inhibits the use of numerical examples which incorporate some degree of open-endedness and evolve through choices made by students during the solution process. It was then natural for one to wonder whether it would be possible to develop CBI modules that capitalized on the strengths of both tools by getting them to work together. In other words, would it be possible for a CBI module to be based on the powerful presentation features of Authorware while, at the same time, having the computational muscle of Matlab at its disposal.

In this work an initial attempt to integrate CBI modules created using Authorware with Matlab is described. An example is presented of an interactive problem that takes advantage of this link and the resultant instructional advantages are discussed. Section II describes the module in which the experiment was carried out. The process of establishing the interface is described in Section III. Section IV contains a discussion of the benefits derived as well as a preview of where to go from here.

## II. The Electroscience Module and RLC Circuit Transient Response

This work was carried out under the aegis of the NSF-sponsored Greenfield Coalition which is attempting to develop a new curriculum model for manufacturing engineering education. Electroscience [3] is the acronym for the knowledge area within this curriculum that is derived from the integration of topics from the traditional areas of Electrical Engineering (for non-majors) and the Electricity and Magnetism sections of Physics. One topic contained within the Electroscience module deals with techniques of circuit analysis. The examples we present illustrating the Authorware-Matlab nexus are taken specifically from the transient analysis of RLC circuits module.

An RLC circuit is characterized by the fact that its transient response can be one of three types called underdamped, critically damped, and overdamped; these terms relate to the graphical appearance of the different responses and are determined by the choice of component values. The relationship between the component choices and the nature of the response is vital to understanding. In particular, understanding is enhanced if the use of pre-determined combinations of component values and their associated solution curves are replaced by interactive selections made by the learner. The Authorware-Matlab link that we have developed enables this to be done.

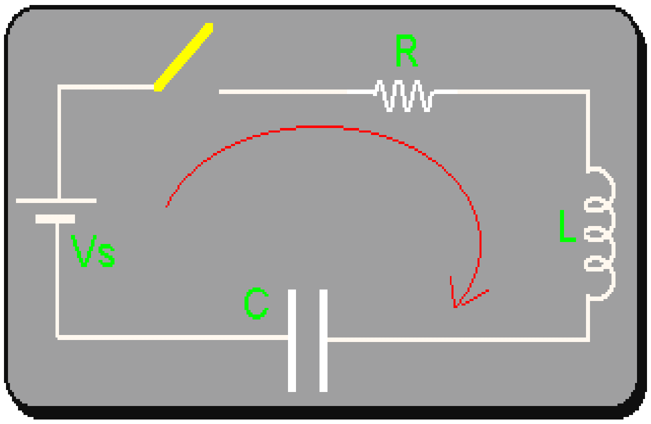


**Figure 1**: Series RLC Circuit

An RLC circuit is shown in Figure 1. The detailed theory of the natural and step responses of the RLC circuit can be found in [4]. The following is a brief summary of the key elements of the theory. A series RLC circuit is characterized by a second order differential equation of the form:

$$\frac{d^2v}{dt^2} + \frac{R}{L}\frac{dv}{dt} + \frac{1}{LC} = constant \tag{1}$$

with a corresponding characteristic equation given by:

$$s^2 + \frac{R}{L}s + \frac{1}{LC} = 0 \tag{2}$$

The nature of the response is determined by the nature of the roots of the above equation which are:

$$s_{1,2} = -\frac{R}{2L} \pm \sqrt{\left(\frac{R}{2L}\right)^2 - \frac{1}{LC}} \tag{3}$$

It is evident from the above that the roots can be complex, real-equal, or real-unequal, based on the component values. These correspond to underdamped, critically damped, and overdamped responses respectively. As mentioned above, the names are derived from the graphical form of the responses, typical examples of which are provided in Figure 2. Implementation of the Authorware-Matlab interface and the enhanced

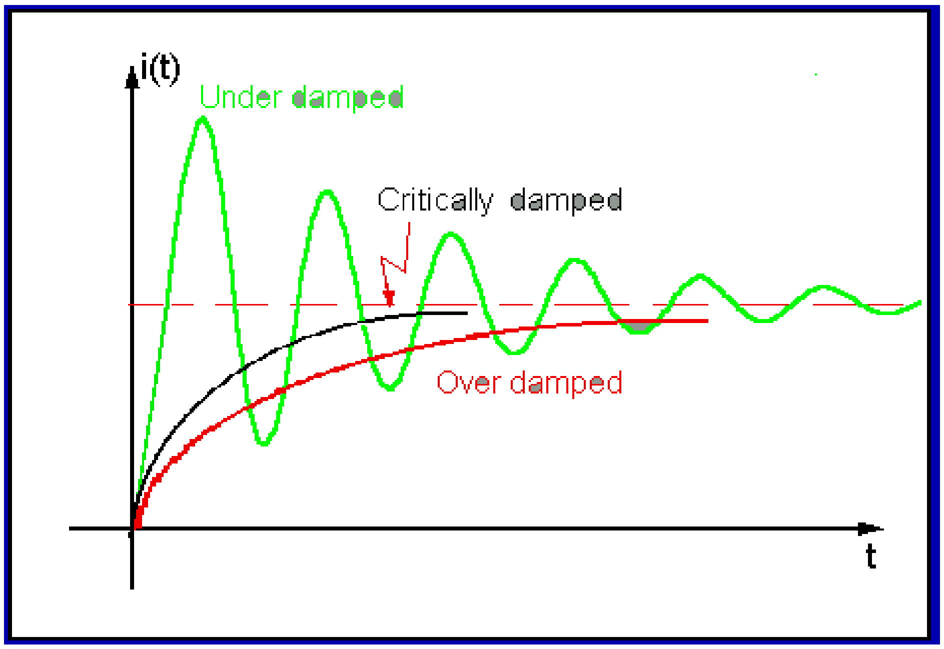


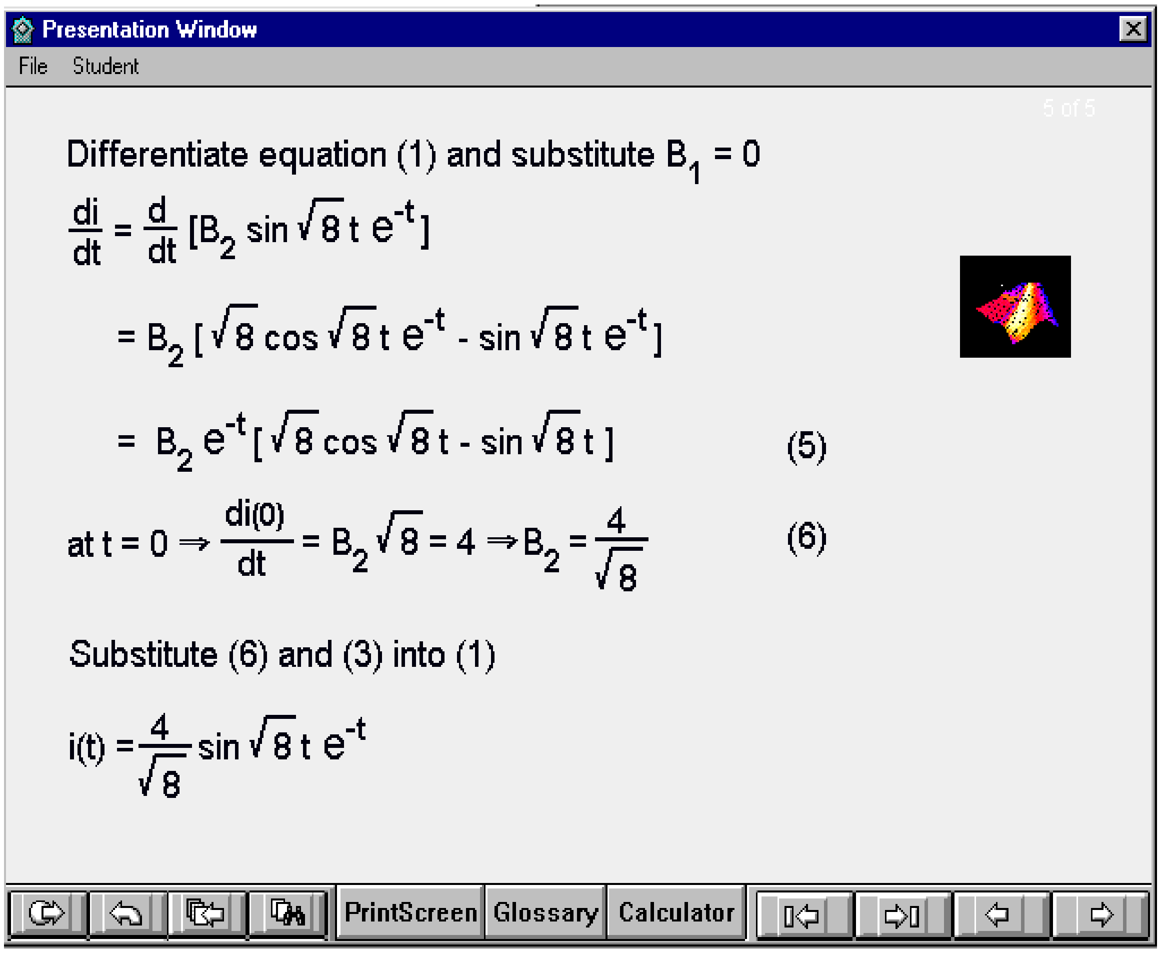
**Figure 2**: Typical forms of RLC transient response

instructional possibilities it affords in the context of the RLC problem is discussed in the next section.

## III. The Authorware-Matlab Interface

Authorware has a programming feature that enables it to run other application software from within its environment, while it continues to run in the background. This is achieved through a built-in function called “JumpOutReturn”. When a student clicks on a Matlab icon provided by the developer within an Authorware screen, control is temporarily transferred to Matlab through this pre-programmed function. A Matlab command window opens up enabling execution of M-files that have been pre-written to support instruction. Upon completion of the exercise, use of the exit command in Matlab returns control to Authorware.

Once within Matlab the developer has two options to enable the learner to make his own component choices in a circuit. The first option is to provide simple instructions to the student to enter these choices in the required Matlab format before running the prescribed M-file. The second option is somewhat more elegant and takes advantage of the graphical user interface (GUI) capabilities of Matlab. GUI enables the use of such graphical objects as lists, buttons, sliders, etc., on the screen. These objects can be used to accept inputs and provide outputs to a user in a very natural manner. In other words they serve as pipelines to a Matlab program and their inclusion in an application simplifies the process of user interaction, and makes the application more attractive and easier to relate to.



Differentiate equation (1) and substitute $B_1 = 0$

$$\frac{di}{dt} = \frac{d}{dt}[B_2 \sin\sqrt{8}\,t\, e^{-t}]$$

$$= B_2 [\sqrt{8}\cos\sqrt{8}\,t\, e^{-t} - \sin\sqrt{8}\,t\, e^{-t}]$$

$$= B_2 e^{-t}[\sqrt{8}\cos\sqrt{8}\,t - \sin\sqrt{8}\,t] \qquad (5)$$

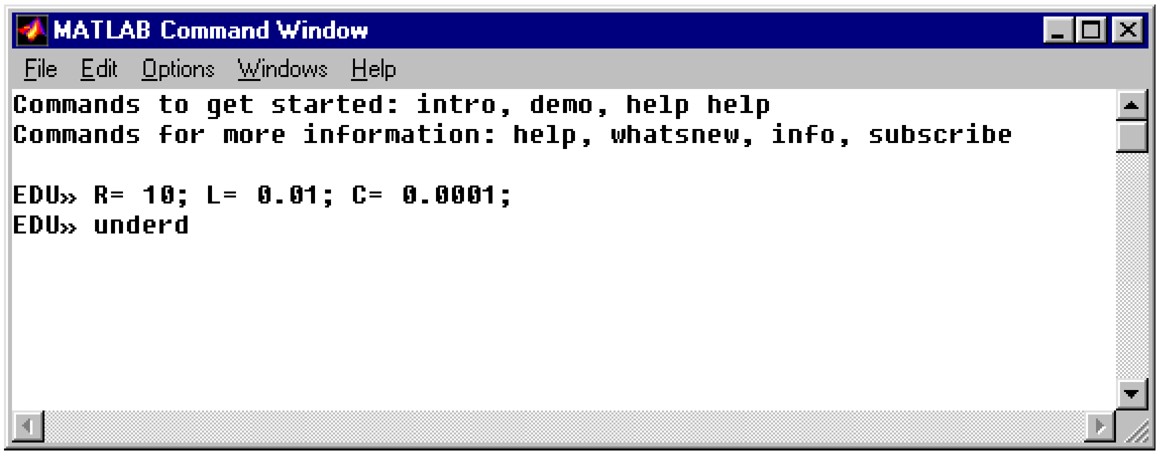
$$\text{at } t = 0 \Rightarrow \frac{di(0)}{dt} = B_2\sqrt{8} = 4 \Rightarrow B_2 = \frac{4}{\sqrt{8}} \qquad (6)$$

Substitute (6) and (3) into (1)

$$i(t) = \frac{4}{\sqrt{8}}\sin\sqrt{8}\,t\, e^{-t}$$

**Figure 3**: Matlab launch window

A walk-through from the learner’s perspective of the RLC transient response problem is provided through the successive module screens shown in s 3-6. Figure 3 shows the parent Authorware screen from which Matlab can be launched by clicking on its icon. When this is done control is temporarily transferred to the Matlab command window shown in Figure 4 where component values can be redefined before an M-file is run. Figures 5 and 6 show two responses that are underdamped to different extents as a result of the component values chosen. The next section contains a discussion of the benefits of this work.



```
Commands to get started: intro, demo, help help
Commands for more information: help, whatsnew, info, subscribe

EDU» R= 10; L= 0.01; C= 0.0001;
EDU» underd
```

**Figure 4**: Matlab command window

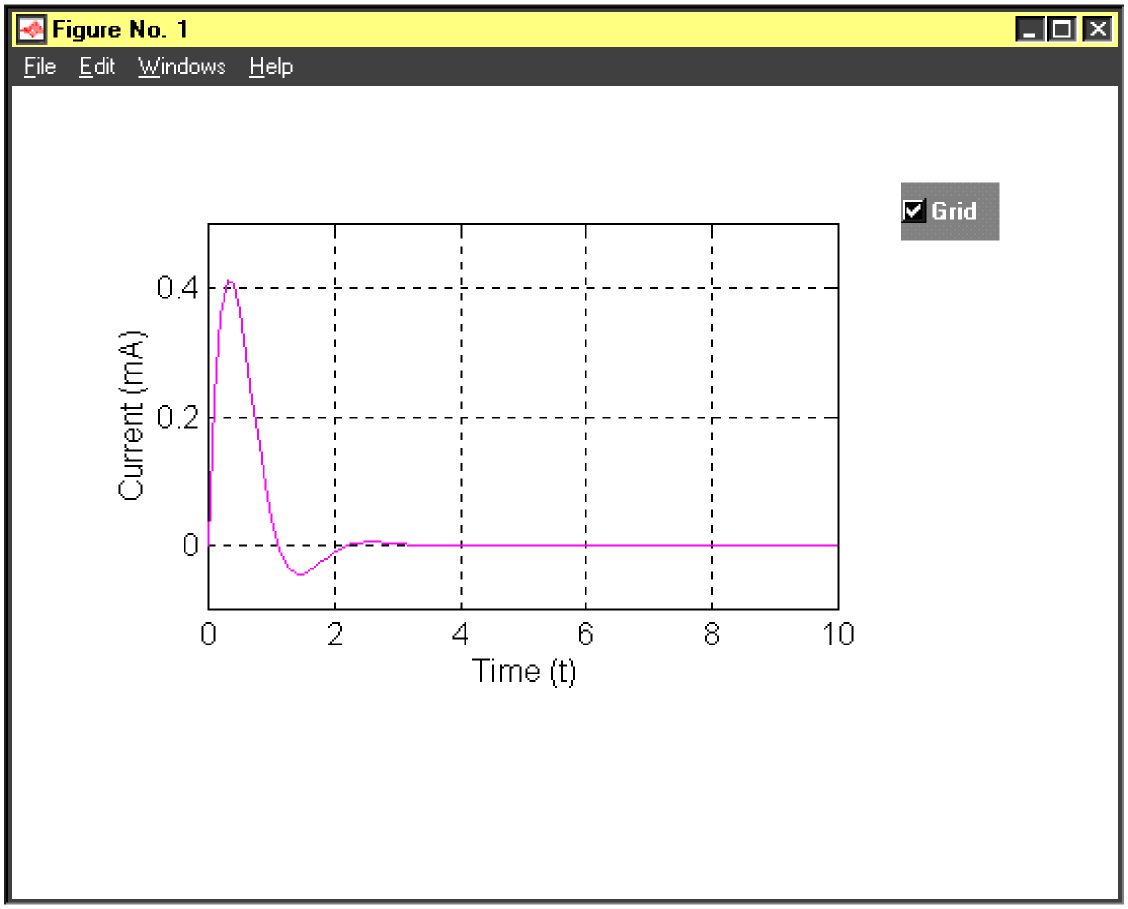


**Figure 5**: Mildly underdamped response

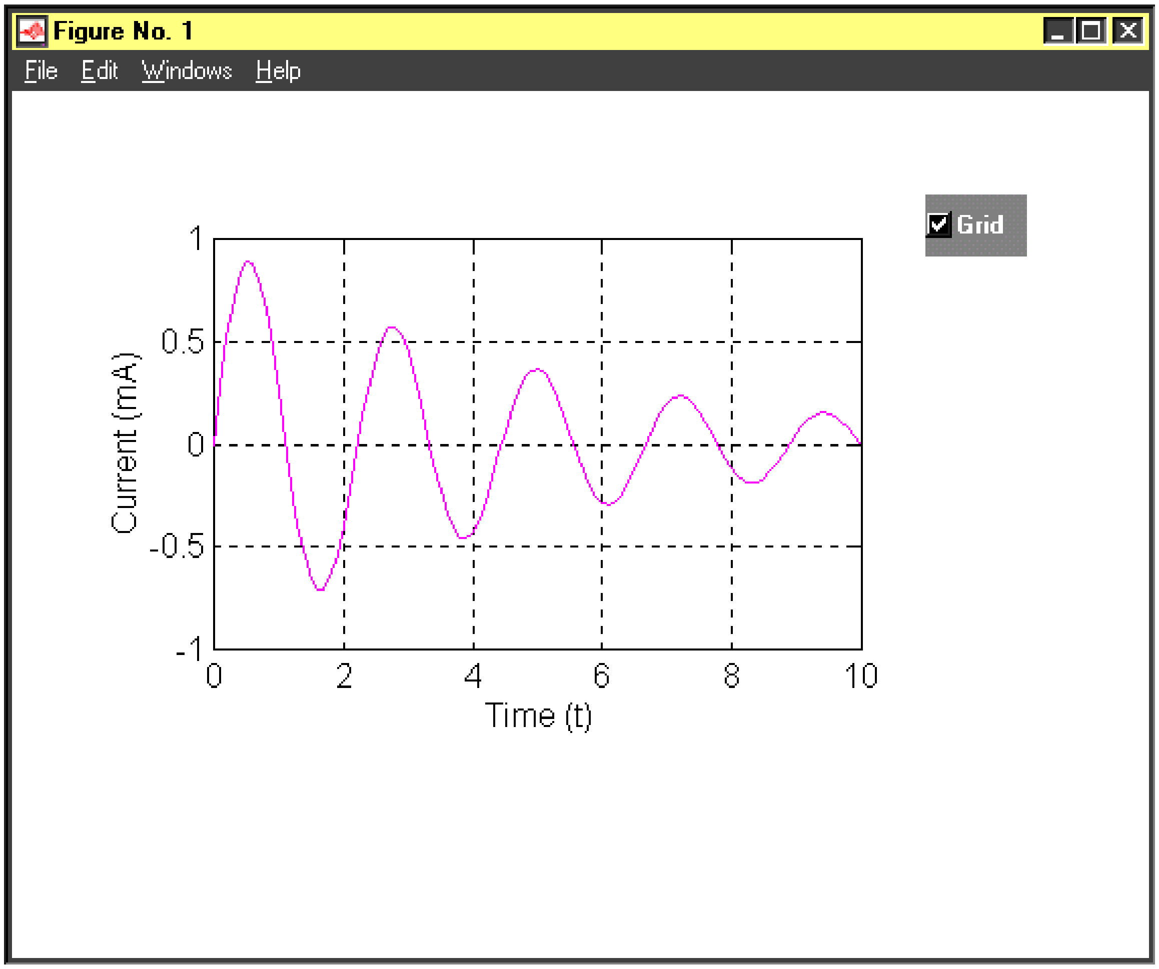


**Figure 6**: Severely underdamped response

## IV. Discussion

This is just an initial attempt in an ongoing process to mate the presentation capabilities of Authorware with the computational power of Matlab. In its present form the interface enables the learner to invoke Matlab at certain pre-selected points in the module and execute some pre-written programs. The ideal goal of such an endeavor would be to make the Authorware to Matlab interface completely seamless so that the learner can chose to bring up the latter at will if he/she chooses to use it to enhance understanding; the use of the word *seamless* is intended to include such capabilities as being able to make component changes in a circuit diagram on an Authorware screen and have the new analysis results instantly computed and available through Matlab. Additionally, of course, there would be other pre-determined points in the module chosen by the developer, where the learner would be specifically directed to invoke Matlab, either to get a point across or to lay the grounds for an open-ended exercise.

Authorware is a powerful tool in the development of CBI modules. However, one weakness it has is the inability to carry out high-level computation. As a result of this the typical mathematical problem used to demonstrate a principle or to test a student is of the prescriptive type where the solution paths and numerical results are pre-determined and programmed in advance. This weakness can be addressed by linking it to a computational package such as Matlab. If in the middle of a presentation a student finds his understanding of a mathematical concept being discussed incomplete - if he is not "getting it" - he will have a means of addressing this problem. The availability of the Authorware-Matlab interface would permit the inclusion of sophisticated problems of the type found in many engineering textbooks that come bundled with computational software. Such a link is essential for CBI to realize its full potential and be a *complete* presentation medium.

## References

[1] Alfred Riddle, "Mathematical Power Tools", IEEE Spectrum, November 1994, pp. 35-47.

[2] Nizar Al-Holou and Timothy Savage, "Selecting an Authoring Program for Undergraduate Engineering Computer-Based Instruction, Proceedings of the 1995 Frontiers in Education Conference, Atlanta, Georgia, November 1995.

[3] Nizar Al-Holou, "Development of Electroscience Curriculum for Greenfield Coalition", Proceedings of the 1997 Frontiers in Education Conference, Pittsburgh, Pennsylvania, November 1997.

[4] James Nilsson and Susan Reidel, "Electric Circuits", 5th edition, pp. 305-354.

## About the Authors

NIZAR AL-HOLOU is an Associate Professor of Electrical Engineering. His area of expertise is in Digital Systems, Microprocessors, Computer-Based Instruction, and Computer Architecture. He has published in the areas of Parallel Processing and Computer-Based Instruction. He joined the faculty at UDM in 1992.

N. MOHANKRISHNAN is an Associate Professor in the Department of Electrical and Computer Engineering at the University of Detroit Mercy. He is a member of IEEE and ASEE. His research interests are in the areas of Engineering Education and Digital Signal Processing Applications

## Acknowledgment

This project is funded by the NSF through the Greenfield Coalition (Grant #EEC-9221542) whose objective is to develop a new paradigm for manufacturing engineering education. The authors would also like to acknowledge Mr. Mohanad Murad for helping with the programming of the Authorware-Matlab interface.