A SOFTWARE LEARNING TOOL FOR VECTOR AND COMPLEX-NUMBER FUNDAMENTALS: TECHNICAL DEVELOPMENT

Howard A. Smolleck, Nadipuram R. Prasad, Barbara Powell, Bhargava Jayanti, Shakir Manshad, Sashadry Divakarla, Paul Arellanes

Klipsch School of Electrical and Computer Engineering
New Mexico State University

Abstract

Working under a National Science Foundation Advanced Technological Education (NSF/ATE) grant, the authors have been involved for two years in the development of graphical, interactive software for use by students in Engineering and Technology programs. The project is primarily oriented toward two-year, community-college institutions and has involved the building of a small consortium of such schools.

The first module in the series to be completed and implemented, entitled ***Vector Vision***, is now in use in the author's institution and in several two-year partner schools. ***Vector Vision*** provides interactive instructional and problem-solving resources for introducing and reviewing fundamental vector and complex-number concepts for lower-division Engineering and Technology students.

The philosophy and objectives underlying the series of software programs (already discussed in previous works) are briefly reviewed, with specific attention to the ***Vector Vision*** module. The primary goal of the present paper, however, is to describe in detail how such a graphical, animated, interactive educational package with audiovisual aids and versatile on-screen commands is developed.

The specific languages, tools, and techniques employed in program development are discussed, and some sample coding is presented. Specific relevant techniques associated with the Visual C++ development and production are enumerated. Program use is briefly demonstrated, and some screen views illustrating program use are included. The development and inclusion of animated HELP sequences, tutorials, and other aids is described.

Introduction

For several years, the senior author and his colleagues at New Mexico State University (NMSU) and elsewhere have been involved in the development of simple but comprehensive means for graphically illustrating certain concepts fundamental to Engineering and Technology courses. These basic

concepts, which underlie much of Electrical Technology and other branches of the field to a lesser degree, include vector concepts, steady-state ac network analysis, energy-conversion, and electromagnetic systems. The present project seeks to address the need for such materials through the development, implementation, evaluation, and dissemination of interactive, animated, graphical software aids for classroom and out-of-class use.

The authors have thus far completed two major modules for instruction in mathematical and circuits topics. The first module in the series to be completed and implemented, entitled ***Vector Vision***, is the subject of this paper.

Vector Vision provides interactive instructional and problem-solving resources for introducing and reviewing fundamental vector and complex-number concepts for lower-division Engineering and Technology students. It incorporates animated illustrative sequences, "talking-head" explanatory video files, audio messages, and graphical and analytical problem-solving capabilities. The software is in use in several departments of the authors' institution. It is also beginning to be incorporated into classes in a number of two-year associate-degree-granting schools as well as in four-year programs in mathematics, physics, engineering, and technology.

The specific languages, tools, and techniques employed in program development are discussed, and some sample coding is presented. Specific relevant techniques associated with the Visual C++ development and production are enumerated. Program use is briefly demonstrated, and some screen views illustrating program use are included. The development and inclusion of animated HELP sequences, tutorials, and other aids is described.

Background and prior development

Prior to beginning the development of this recent set of instructional aids, the authors acquired considerable experience at New Mexico State University in the development of very applied demonstration software for industry training programs and for academic use in electric power systems education[1-8]. These have included demonstration and analysis packages, incorporating interactive animated color graphics, for investigation of rotating ac machine behavior under a variety of unbalanced (as well as balanced) operating conditions, software for illustrating steady-state synchronous-machine behavior, Fourier analysis and the Fast Fourier Transform, and power-systems relaying (see [1,7] for a summary of these, and [2,3] for discussions of the first-developed program in the series).

More recently, they have concentrated on the production of more sophisticated, user-friendly, microcomputer-based instructional aids which **enhance student performance and participation** in early engineering and technology courses, and **provide problem-solving resources** that students can use as they progress through succeeding courses. The complete project seeks to address the following areas, through the creation of several software modules:

1) basic vector analysis
2) steady-state ac circuit analysis fundamentals
3) vector analysis of steady-state ac circuits
4) steady-state electromagnetic device behavior (including coupled-coil operation)
5) introductory power-systems and utility-related concepts

As mentioned earlier, the present paper is concerned with the first module of the series. A brief summary of the purpose and capability of this module are as follows.

The purpose of Vector Vision is to acquaint students with vector concepts through presentation of historical and technical information, graphical demonstrations, and mathematical exercises. The initial targeted level is the freshman or sophomore technology student, with usefulness in other (and later) classes as well. The program is animated and fully interactive, with audio and movie-quality video.

Usual Windows 95 ™ (or later) commands and procedures, such as minimizing, maximizing, or otherwise manipulating a window, may be invoked throughout the program. A HELP file is provided at each stage where it might be needed. The software is designed so that students having a rudimentary knowledge of Windows 95 ™ may be able to navigate quickly and successfully through the entire package with little additional instruction.

Calculation and demonstration capabilities include explanation of unit vectors, multiplication by a scalar, vector addition and subtraction, vector projection, the dot and cross product, etc. Most of these are graphical and some are animated [10].

Since a major use of this program is as a precursor to steady-state alternating-current instruction (as embodied in ***AC Insights Plus***, a second module in this series), a section of the program treats vectors as **complex numbers**, or **phasors**. The wording in this part of the program was chosen to emphasize the fact that these values are actually quantities represented by complex numbers in the two-dimensional complex plane.

This complex-number section is of particular interest to *Electrical* Technology students and educators. Using it, students can explore the mathematical manipulations of complex numbers, including resolution, addition, subtraction, scalar and complex-number multiplication and division, and the essence of the complex plane. Click-and-drag options characterize this part of the program. For instance, vector addition can be simulated by clicking the left mouse button in the complex plane at the point of the first vector, clicking the right mouse button at the point of the second vector, and then viewing the sum of the two, with appropriate constructional lines shown dashed. Of course, numerical entries may be input as well, and the results scaled to match the student's desire or need. Users learn the need for appropriate scaling, and gain practice in implementing it.

Vector Vision: specific design objectives and characteristics

As the Vector Vision software was being designed and in the early stages of its development, several objectives were established and have been satisfied**.** These include

- Windows-based appearance and operation
- An eye-catching, informative introduction
- Animated audiovisual presentation format
- Interactive tutorial and problem-solving capabilities
- Full mouse-driven interactive capability
- Delivery on, and easy installation from, CD-ROM media

Programming language

Based upon the experience of the authors as described above, the original plan was to develop an interactive, graphical program in the C language. That language had been chosen for the earlier, successful software [2-7] because of its extensive interactive and graphics-display capabilities. The first module of the present series was actually begun in C. However, examination of recent multi-media usage, and recently-acquired equipment to perform this kind of development, led to the use of the Visual C++ ™ (rather than C++) language. Visual C++ has advantages over C in its ability to create programs which function in a very windows-like environment, including display of quality still-life graphics, animated graphical sequences with audio capability, and interactive features such as "buttons", scrollbars, etc.

Other relevant development software used

In completing the development of the Vector Vision module, software including Adobe Premiere ™ and Director 6.0 ™ were utilized to synchronize (1) animations created in Lightwave 3D ™, (2) audio and camera-generated video files, and (3) other graphic backgrounds and stills. Audiovisual (.AVI) files are created therefrom, and are accessed through the Visual C++ windows program.

Other state-of-the-art capabilities were used as well. These include the Microsoft application Camcorder ™ which provides an animated display of mouse movements and other screen command sequences and is useful in generating HELP screens. Audio comments were added to these HELP and introductory screens to enhance program effectiveness.

Although it was obtained too late to provide much use in Vector Vision development, the developers have begun to use MacroForm 2.0 ™ to produce novel three-dimensionalized pictures to capture student interest and attention in some of the succeeding software products.

The use of some of these packages is described in detail in the sections below.

Vector Vision: some programming challenges and solutions

Real-time audio and video files

When ***Vector Vision*** is accessed, the title screen appears as an eye-catching, animated movement of words, letters, graphics, and background to the accompaniment of appropriate music (Figure 1 is a still shot illustrating a group of "vectors" which shatter and break through a "window", followed by moving letters which ultimately spell out the program name). Credits for project support and personnel then appear, including logos of the supporting entities, in an animated manner such as in a commercial film production. A brief narrated introduction by the developer, with voice and video, personalizes the program.

Following these, the user is provided a screen containing several icons from which to choose. The user may access a history review, tutorials, problem-solving screens, or illustrative graphics. The history review, for example, is audio-visual, narrated by a "talking head" and includes appropriate and engaging background views. Some of the tutorials are chalkboard-style, showing a lecturer in front of an actual blackboard (Figure 2). As another illustration of an audiovisual

presentation, Figure 3 shows a still shot of a slowly-rotating helix which is used in one of the tutorials to illustrate and emphasize the operation of the cross product. Each of these demonstrations is accessed by clicking on an icon or button (A discussion of how this capability is programmed is presented in a later subsection).

To produce this kind of animated presentation, animated video (movie) files were developed in .AVI format from actual videocamera shots and through computer-generated animation. The process of creating a computer animated presentation is three-fold. First, live footage is shot of the actors performing their dialogue. During the recording process, the footage is saved as a Microsoft Windows audio-visual (.AVI) file. Once saved, the footage is edited using Adobe Premiere™ and Sound Forge™ to "clean up" any visual problems or audio noise. When this first edit is completed, the file is saved as another .AVI file.

The second stage in creating the presentation is actually the design and creation of the computer animation. Before any computer work can begin, graphical sketches of the desired concepts are created to make sure that what is desired is understandable to the intended audience. Once this has been accomplished, the next phase is to actually start creating the animations. The animations are created and rendered using Newtek's Lightwave 5.0 ™. The process of creating, editing, and rendering a computer animation can be very time consuming, sometimes taking weeks to complete the more complicated animations. Rendering an animation means combining all the components of a scene file, which can include many animated objects, light movements, and special effects, together to create an .AVI file. Each time a change is made to the finished animation, it must be re-rendered.

The third stage in creating the presentation is putting together the live footage with the computer animation. This process is done in Adobe Premiere. For the process the two components must be carefully spliced together so that the finished product is as fluid as possible. Sometimes during the process, the computer animation will not exactly fit in with the live footage. In such cases, the animation has to be edited and re-rendered again. Once this has been accomplished, sound effects and an appropriate soundtrack are laid down to compliment the product when necessary. The end result is a seamless merging of real-world video footage with computer animation.

Mouse-driven capability

One reason for the choice of the Visual C++ development software was to enable full mouse-driven capability of the completed Vector Vision software. In Vector Vision, the mouse is used not only to select menu items throughout the program (including the accessing of audio and movie files), but also for such technical purposes as to enter and stretch vectors graphically.

Mouse movement is easily built into a program developed using Visual C++. In the program, the member function

OnLButtonDblClk()

is added to the program with the help of **Class Wizard**, a tool in Visual C++. When the user presses the left button of the mouse twice, Microsoft Windows sends the message WM_LBUTTONDBLCLK

to that window, and the program performs the necessary action in response to the message.

Figure 1. Vector Vision introductory screen (still shot of animated motion).

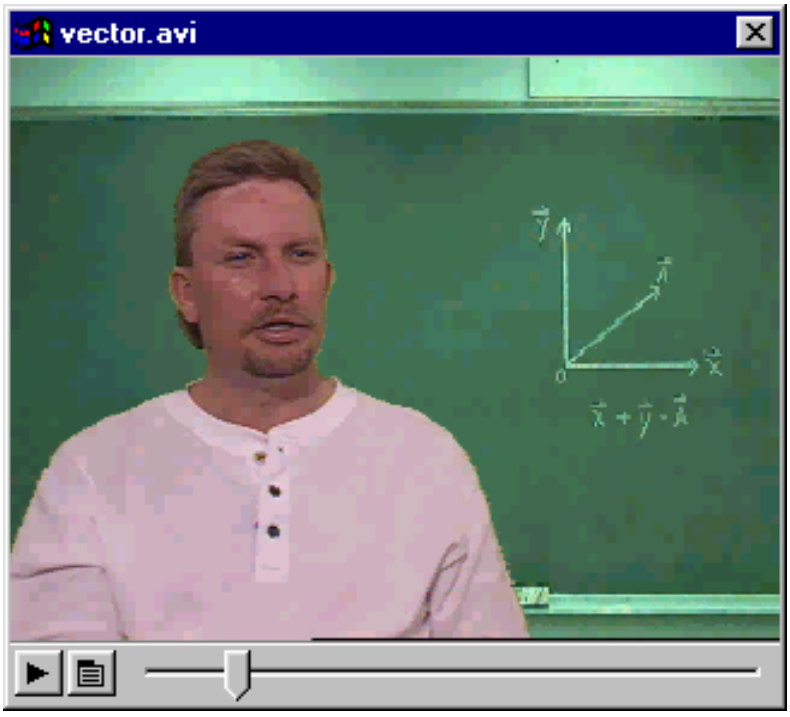

Figure 2. Vector review

.

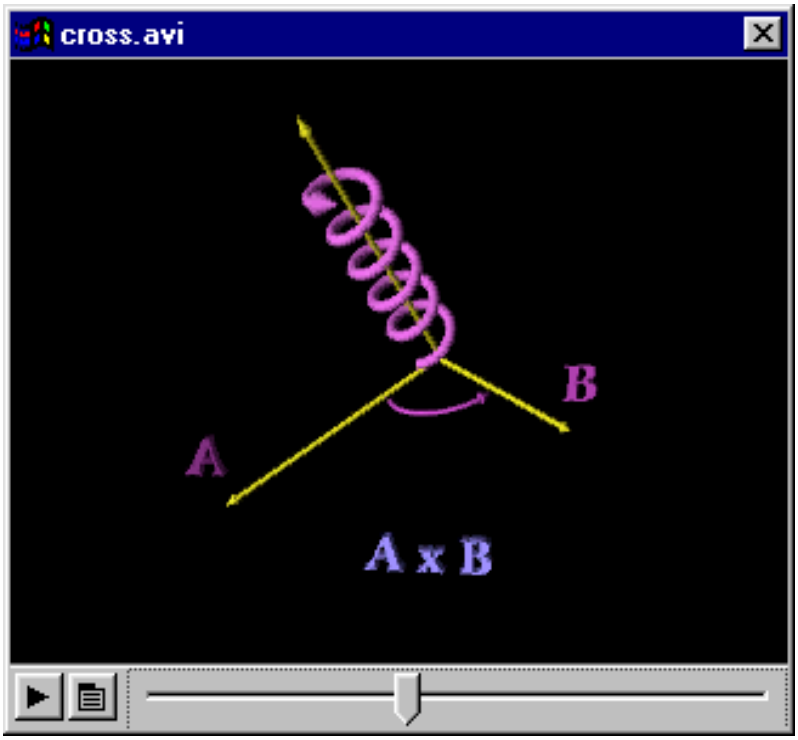

Figure 3. Illustration of cross product.

Importing and display of still-life graphics

Although Vector Vision does not make extensive use of still-life views, other modules in the series do. These are imported simply as bitmap (.BMP) files and are stored into C++ using the **Insert Resources** command under the **Resources** option. When bringing such files in, the bitmap editor of C++ can be used to edit and view these bitmaps. Such figures are sent to the screen by commands such as

```
DisplayBitmap(pDC,IDB_BMP59B,0,curpos,550,350);
```

which locates the position and determines the size of the displayed bitmap figure.

Dialog boxes and interactive calculations

One of the most important design criteria was that the instructional modules in this series provide ample opportunity for students to enter their own numerical values and see the results of calculations. Although this is probably less important in Vector Vision than in other modules being developed, there are several points within the program where this capability is required.

One example concerns calculation of the dot product of two vectors. A dialog box illustrating this is shown in Figure 4 .

This dialog box is created with the help of the Dialog Editor of Visual C++. Using the **Class Wizard**, also a tool of Visual C++ , a dialog box is created and this class is derived from the class CDialog. A sample code is given on the following page:

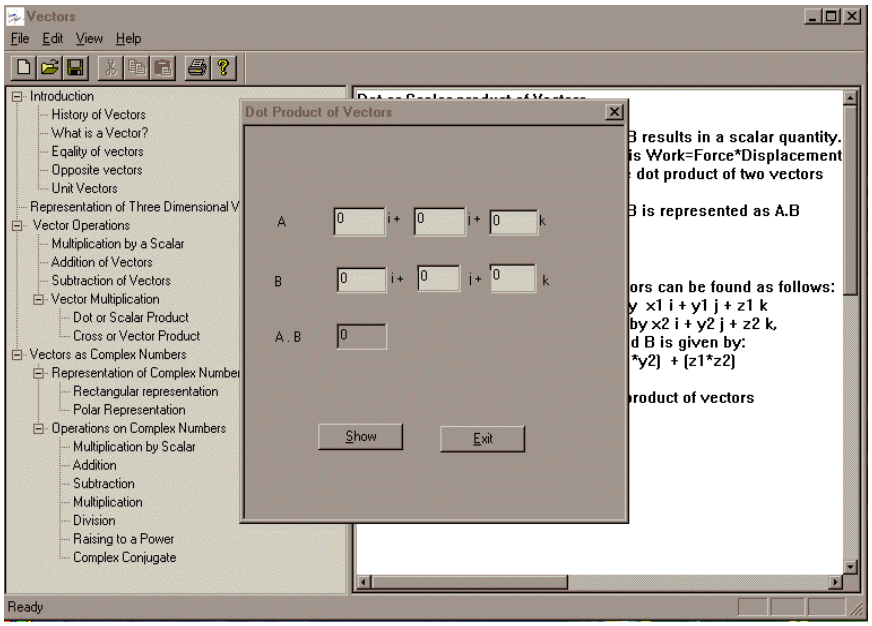

Figure 4. Example of dialog box.

```
CDotproDlg::CDotproDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CDotproDlg::IDD, pParent)
{
        //{{AFX_DATA_INIT(CDotproDlg)
        m_editabi= 0.0;
        m_editai = 0.0;
        m_editaj = 0.0;
        m_editak = 0.0;
        m_editbi = 0.0;
        m_editbj = 0.0;
        m_editbk = 0.0;
        //}}AFX_DATA_INIT
}
```

The Class Wizard adds the associated code and header file to the Developer Studio Project. It is used to add the necessary data members, member functions, validation functions, message handlers for the dialog box's buttons, etc. to the dialog class.

Animated HELP sequences

An example of an animated HELP sequence is that which is available in the complex-number graphical display section. In this part of Vector Vision, the user can enter vectors on the screen by right- or left-clicking on a point on the grid to identify the head of a vector.

The Microsoft application Camcorder ™ was used to make an animated "movie" illustrating the mouse movements and other screen command sequences required in this part of the program. When this HELP option is accessed, a large arrow moves around the screen to illustrate the options as a background voice describes them. An illustration of screen appearance appears in Figure 5.

Compiling, linking, and construction of *setup* file

Separate parts of the program, including the main C++ code and other procedures, are each individually compiled into respective .EXE files. These are then brought together into a setup file. The setup file is created using Exemplar Setup Toolkit, which is provided with Visual C++. While creating the setup file, the Exemplar Setup Toolkit asks the programmer to enter the path of the "makefile" of the application to be installed. (In the process it also asks for name of the application and options for database systems like ODBC, ActiveX, DAO which are not required here.)

The final application is written to a CD-ROM disk via a CD-creator program and is then ready to use.

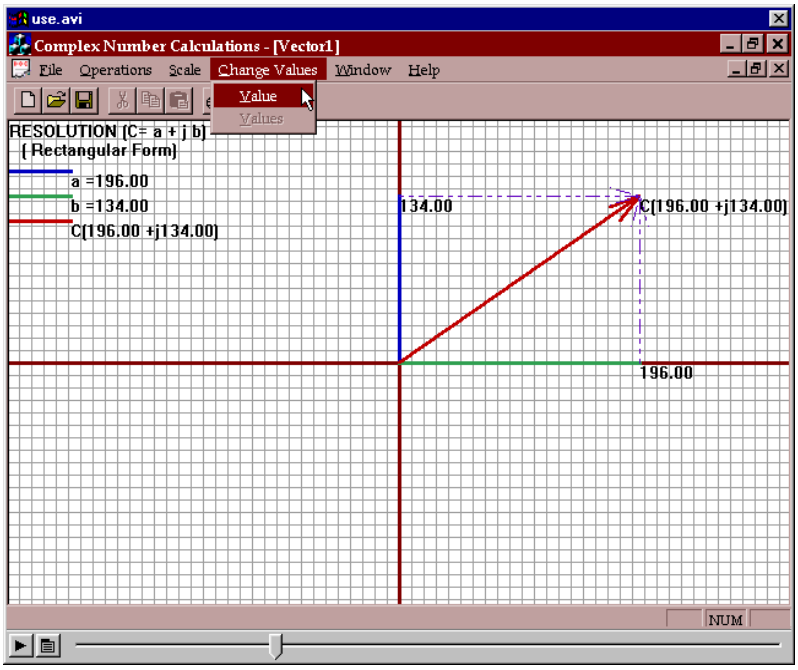

Figure 5. Illustration of HELP screen.

System requirements

Vector Vision requires at least a Windows 95 environment with a CD-ROM drive, and a minimum of 4 MB of hard-disk space for partial installation, or about 300 MB for full installation (see below). It also needs at least 16 MB of RAM; preferably 32 MB, and a sound card and set of speakers to allow the use of the audio features.

Distribution media and installation

Since today's freshman and sophomore students typically have ready access to Windows 95 ™ or a later version, a CD-ROM drive, and appropriate high-resolution monitors, and since they are nearly all familiar with windows-type operations, and because of the significant storage requirements of the audiovisual (AVI) files, the modules of this series are provided on CD-ROM.

Vector Vision is installed using the standard **setup** command. The setup file provided with Vector Vision allows two installation options: either ***full* or *compact installation***. A number of very large .AVI and .WAV files are not copied to the user's hard drive if ***compact installation*** is chosen, in which case the CD-ROM disk must be left in its drive during program use in order to access these audiovisual features.

Once the software packages in this series have been beta-tested, it is anticipated that they will be available on CD-ROM for a nominal fee from the lead author.

Anticipated learning enhancement

Vector Vision is intended to be used as a concept demonstrator in the classroom, as an assigned aid in learning concepts and solving problems, and as a self paced learning tool. Although not all instructors use the software in all of these modes, feedback received from faculty and student users at community-college partner schools suggests that the program is meeting all of these objectives.

The third objective is perhaps the most intriguing. **Vector Vision** is being used in several partner schools having a large percentage of first-generation or otherwise disadvantaged college students. Observations suggest that many of these students feel intimidated when presented with concepts in class that they do not understand but perceive that their classmates *do* understand, to the point of refraining from asking questions in class. Such students seem to adapt readily to the computer, however (and probably have considerable experience in using the computer for entertainment as well as intellectual pursuits). It will be very interesting to verify our contention that such students, when allowed out-of-class time alone with an entertaining and challenging program such as this one, will remedy some of their deficiencies on their own through program use.

Conclusions

Guidelines and procedures relating to the construction of a robust interactive audiovisual instructional module with tutorial problem-solving capability are shown. Very specific design objectives and their solutions are indicated. The feasibility of developing such a module for Windows operation, using the Visual C++ language and other programs as development tools transparent to the final user, is demonstrated.

Acknowledgements

The authors are grateful for the support of this project, provided in part by the National Science Foundation, the Westinghouse Foundation, and the Klipsch School of Electrical and Computer Engineering at New Mexico State University. They are also grateful for the help of the Project Evaluator (Dr. Keith Mc Neil) and the faculty at the participating two-year institutions.

Bibliography

1. H. A. Smolleck, "The effective use of interactive computational demonstrators in electric power engineering education". Proc. of the ASEE/IEEE Frontiers in Educ. Conf., Nashville, TN, Nov. 11-15, 1992, pp. 480-488.

2. H. A. Smolleck and D. S. Dwyer, "A comprehensive interactive microcomputer capability for demonstrating ac machine operation", IEEE/PES Paper 90 SM 400-2 PWRS; IEEE Trans. PAS, Vol. 6., No. 3 (August 1991), pp. 1305-1314.

3. H. A. Smolleck and D. S. Dwyer, "Demonstration of ac machine behavior through interactive color graphics", IEEE Trans. CAP, Vol. 3, No. 4 (October 1990), pp. 49-53.

4. H. A. Smolleck, "A simple revolving-field demonstrator for the Personal Computer", IEEE Trans. Educ., Vol. 31, No. 2 (May 1988), pp. 119- 123.

5. L. M. Rust, H. A. Smolleck, and D. S. Dwyer, "Some applications and observations on the use of a new harmonic analysis and synthesis demonstrator", Proc. of the 1993 ASEE Gulf-Southwest Section Meeting, Austin, TX, April 1-2, 1993.

6. H. A. Smolleck and Hong Chen, "A software demonstrator for steady-state synchronous machine behavior", Proc. of the ASEE/IEEE Frontiers in Education Conf., Washington, DC, Nov. 7-9, 1993.

7. H. A. Smolleck, H. Chen, S. Badruzzaman, R. Bravo, and D. Pardave, "A developmental package of interactive software for illustrating power system protection principles in educational and industry training programs", IEEE Paper 94 SM 382-2 PWRS, accepted for publication in IEEE Trans. PAS.

8. H. A. Smolleck, "A new look at the effects of unbalanced voltages upon synchronous and induction machines", Electric Power Systems Research, Vol. 25, No. 3 (1992), pp.199-206.

9. H. A. Smolleck, "Development of microcomputer-based instructional aids for introductory technology courses: a new approach and initial work" (invited paper), presented at the NSF/ATE Principal Investigators' Conference, Washington, D.C., November 21-23, 1997.

10. H. A. Smolleck, "Development of Microcomputer-based instructional aids for introductory technology courses: initial accomplishments." Proc. of the ASEE/IEEE Frontiers in Education Conf., Tempe, AZ, Nov. 4-7, 1998 .

HOWARD A. SMOLLECK received his BS, MS, and Ph.D. from the University of Texas, Arlington. From 1974-79 he was on the faculty of Old Dominion University, Norfolk, VA, and since 1979 has been with the Department (now the Klipsch School) of Electrical and Computer Engineering at New Mexico State University, Las Cruces, where he currently holds the rank of Professor. Dr. Smolleck's primary area is electric power systems. He is a member of Tau Beta Pi, Eta Kappa Nu, Alpha Chi, and is a Registered Professional Engineer.

NADIPURAM R. PRASAD is presently an Associate Professor in the Klipsch School of Electrical and Computer Engineering. His research interests are in the area of Soft Computing which includes Fuzzy Logic, Neural Networks, Genetic Algorithms, and Evolutionary Computing. He has published numerous papers on the application of Soft Computing techniques for industrial applications and is the co-editor of a book entitled "Fuzzy Modeling and Control: Selected Works of M. Sugeno".

BARBARA POWELL taught engineering technology at New Mexico State University for 13 years where she created a VHDL FPGA digital design lab and served as principal and co-investigator for several projects sponsored by the National Science Foundation, Department of Energy and several industrial university partners. She is now teaching in the Technology Department at Mesa Community College in Phoenix, Az.

BHARGAVA RAM JAYANTI and SASHADRY DIVAKARLA are graduate students in the College of Engineering and the Computer Science Department at New Mexico State University. SHAKIR MANSHAD is an Electrical Engineering student and in addition holds a doctorate in mathematics.

PAUL T. ARELLANES received a BS degree in Computer Science with a minor in Math from New Mexico State University in May 1998, He was a Ronald E. McNair Post-Baccalaureate Scholar, and a member of ACM. He has recently taken a position as Software Engineer & Technology Transfer Instructor with IBM in Austin, Texas.