IMPLEMENTATION OF A METHOD TO ASSESS STUDENT PROGRESS/PERFORMANCE IN AN INTERACTIVE, INTERNET-BASED, SELF-LEARNING TOOL

Brian L. F. Daku[1] and Heidi A. Diefes-Dux [2]

Abstract *¾ An interactive, multimedia, self-learning tool has been developed to teach the basic concepts of MATLAB. This tool, referred to as M-Tutor, is available on a CDROM for individual student use. It was designed to alleviate the time limitation of learning to using MATLAB in engineering courses. But the logistics of delivering, supporting, and evaluating a CDROM-based tutorial in large lower division engineering courses can be overwhelming.*

The approach taken here was to modify M-Tutor to run in a Terminal Server environment to permit internet delivery. An obvious difficulty with an internet-based tool is assessing student progress and performance. This has been addressed by keeping track of both the material accessed and the exercise performance. The instructor can use this information to monitor student progress and identify difficulties, or it could be used as part of a for-credit evaluation procedure.

The paper reviews M-Tutor and its Windows Terminal Server implementation. Focus is on the implementation of the method used to assess student progress and performance.

Index Terms ¾ MATLAB, Internet delivered course, Student Assessment, Computer based instruction.

INTRODUCTION

MATLAB [1] is a very popular software tool that has become an integral part of many engineering curriculums. Since MATLAB is a software tool, it lends itself well to computer-based instruction that can be integrated with MATLAB problem solving sessions. This approach has been used in a MATLAB tutorial that combines computer-based instruction with interactive MATLAB exercises. This tutorial, called M-Tutor, is CDROM-based and was meant for individual student use at home [2]. In addition to M-Tutor, the student requires an installation of MATLAB to effectively use the tutorial on a computer.

M-Tutor has been used as supplemental material for lower year engineering courses. But for larger classes the logistics of delivering, supporting, and evaluating a CDROM-based tutorial can be overwhelming. Also, since MATLAB is required, the tutorial is generally used on-campus and thus for large classes the required hardware computer resources can be great. One possible solution to these problems is to deliver the tutorial by way of the internet to the student's homes.

The traditional internet delivery tool is a web browser and one approach to achieve internet delivery is to convert the M-Tutor content to a form that could be directly accessed by a web browser. The problem with this approach is that a project such as this requires a significant commitment of time and resources. For example, the development effort for the original M-Tutor tutorial was approximately three man-years. An alternate approach, and the one that was selected, was to modify M-Tutor to run on a terminal server. This is an ideal solution because: 1) the modifications to M-Tutor are relatively minor, 2) the student does not have to purchase MATLAB, and 3) student progress with M-Tutor can be easily evaluated by the instructor since all students will be running M-Tutor from a single campus location. A prototype terminal server implementation of M-Tutor was completed in the summer of 2001. This implementation has been tested both at the University of Saskatchewan and at Purdue University.

The problem of assessing student progress and performance has been addressed in the M-Tutor terminal server implementation. For example, the M-Tutor software keeps a record of all pages accessed in the tutorial, all hotwords accessed and all page exercises, summary quizzes and summary exercise quizzes that are completed. The record of page visits and hotword selections indicate what material the student has viewed. How well the student understands the material is indicated by the exercise and quiz performance. The instructor can use this information to monitor student progress and be alerted to aspects of the content with which the student is having difficulty, or it could be used as part of a for-credit evaluation procedure.

The paper consists of four sections: first an overview is given of M-Tutor, then the Windows Terminal Server implementation is described, this is followed by a description of the implementation of the progress/performance method, and finally the conclusions are presented.

OVERVIEW OF M-TUTOR

[1] Brian Daku, Department of Electrical Engineering, University of Saskatchewan, Saskatoon, Saskatchewan, Canada, S7N 5A9, daku@engr.usask.ca
[2] Heidi A. Diefes-Dux, Department of Freshman Engineering, Purdue University, 1286 Engineering Administration Building, Room 206 West Lafayette, IN 47907-1286, hdiefes@purdue.edu

This section presents a brief description of the MATLAB tutorial called M-Tutor. More detailed information is available on the web site www.m-tutor.usask.ca and in two previously published papers [3],[4].

M-Tutor was developed for students who have had little or no exposure to MATLAB. The content is divided into the following eight sections:

1. Getting Started
2. MATLAB Variables
3. Scalar Math
4. Vector Math
5. Vectors and Basic Plotting
6. Relational and Logical Math
7. Writing Basic MATLAB Programs
8. Matrix Math

Many of these sections are also divided into a number of subsections that focus on a common topic. A listing of the detailed Table of Contents can be viewed at the internet site www.m-tutor.usask.ca.

The tutorial content is delivered on 'computer pages', each of which, presents the student with a new concept or with instructions and examples for a set of MATLAB commands. Learners can select highlighted hotwords to obtain detailed definitions or additional descriptions. Hotwords are also used to request actual MATLAB results for the examples. The majority of the pages have a number of interactive exercises in which students can explore the concepts or commands presented on that page. Note that the amount of information presented on a page has intentionally been kept small so the student can easily digest the material and the exercises can focus directly on that material. An example of one of the computer pages is shown in Figure 1.

The user interface is a comprehensive navigator that has been developed to help students manage their path through the M-Tutor tutorial. While the tutorial content is organized in a traditional manner, students have the freedom to cover the sections in whichever order suits their needs. The features of this navigator include:

1. On page buttons to easily navigate through the tutorial. These buttons can be seen in the top, left-hand corner of Figure 1. From left to right these buttons are: Go back one page, Go to the main menu, Go ahead one page, Access a list of visited pages, Access the bookmarks list, Jump to the last

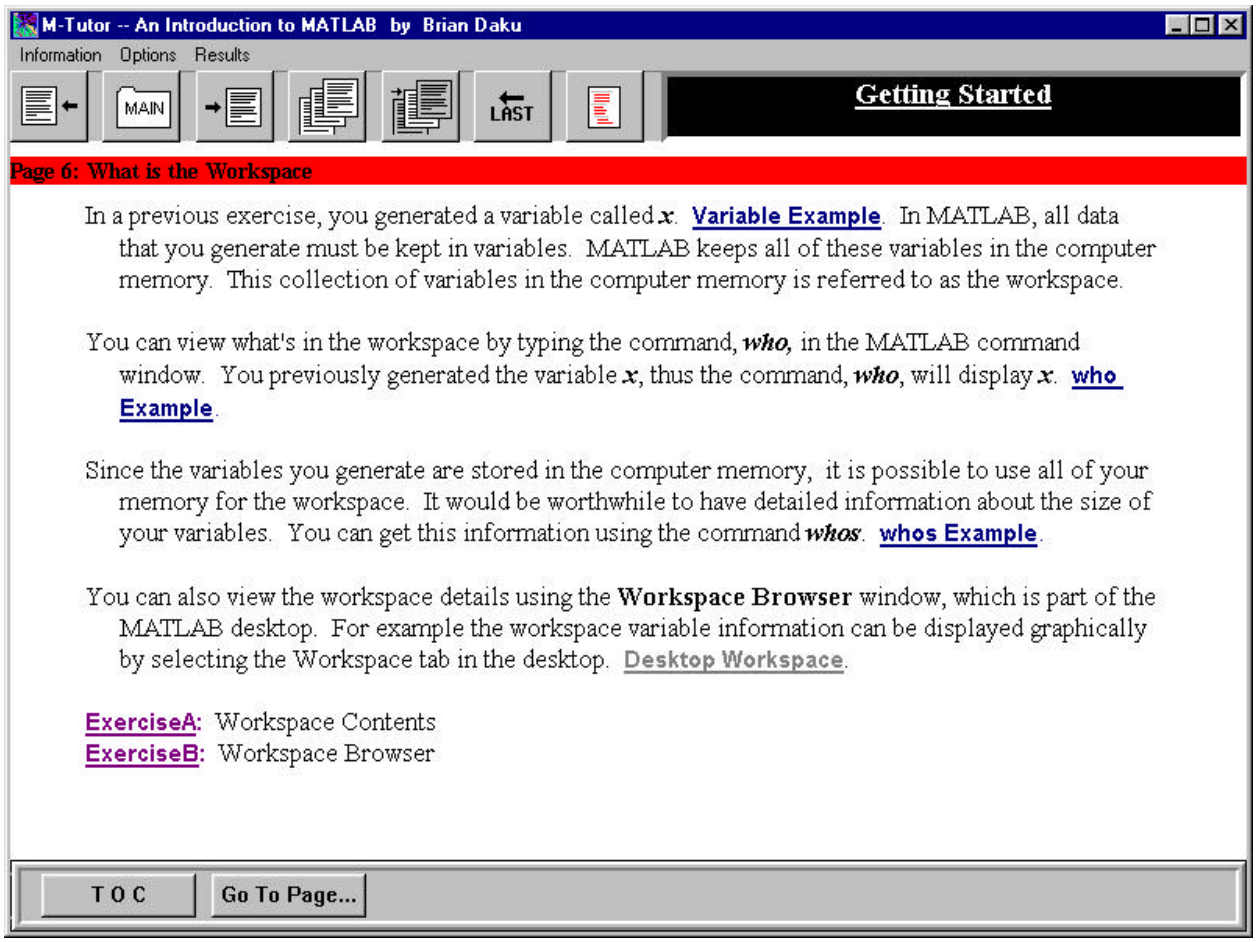

FIGURE 1
EXAMPLE PAGE FROM THE MATLAB TUTORIAL

visited page and Jump to the section contents page.
2. Information, Options, and Results menu items that access useful information about the tutorial, change tutorial options, and present exercise and quiz evaluation results, respectively. These menus are located at the top, left-hand corner of Figure 1.
3. A Help window that describes how to use the navigator. This help facility is accessible from the Information menu.

The navigator is also responsible for maintaining the record of the student's progress through the tutorial. A database file, keyed to the student's name, keeps an audit trail of the student's progress. The database is also used to color code material within the tutorial so the student can easily determine which material has been completed.

Easy access to the various pages in the tutorial is provided by several methods. One of these is the Main Menu, which consists of a hierarchical set of color-coded submenus that represent the sections and subsections in the tutorial. This submenu approach reduces the amount of information that is presented to the student, which is beneficial for first time users of the tutorial. The Main Menu is accessed using the on-page navigation button, MAIN, shown in Figure 1.

The submenu approach can become tedious once the user becomes familiar with the organization of the tutorial, thus an alternate method to access the pages is also available. This method can be accessed by selecting the TOC button (Table of Contents) located at the bottom left-hand corner of Figure 1 or through the Information menu. Selecting this button displays a window that lists all of the sections, subsections and pages in the tutorial. A scroll bar can be used to move up and down the list. Selecting any of the hotwords in this window will take the student to the corresponding page in the tutorial. Finally, specific pages can be accessed directly by page number with the Go to Page. . . button, located in the lower left-hand corner of Figure 1.

A unique feature of M-Tutor is the comprehensive set of reinforcement exercise problems. In the initial Getting Started section of the tutorial, the student uses the MATLAB desktop to perform some simple exercises. This exposes the student to the actual MATLAB interface. In all of the subsequent sections, the student uses a specially designed exercise window to perform the exercises, an example of which is shown in Figure 2.

The exercise problem to be solved is displayed in the top left-hand corner of the window. The user can select the Hints button, which progressively displays a list of hints to aid the student in solving the problem. Students enter their solutions to the problem (MATLAB language expressions) in the center subwindow, which is labeled Enter MATLAB Commands Here. This window has the full functionality of the MATLAB command window, including the use of the arrow keys to access to commands previously executed.. The student enters a MATLAB command and the result is displayed in the MATLAB Response subwindow. This response is the exact response that would be seen if the command were executed in MATLAB's own command window. This is done by evaluating the commands using the MATLAB engine.

The exercise window uses an ActiveX interface to the MATLAB engine to execute the MATLAB commands entered by the student. The student selects the Evaluate button after entering the MATLAB commands and is then informed of whether the proposed solution is correct or incorrect. The evaluation method uses the MATLAB engine to compare the workspace contents and variables of a correct solution with those of the student's proposed solution. The student can access an example of a correct solution from the Hints menu, after using the Evaluate button.

The main advantage of the exercise window is the direct interface to MATLAB, which allows the student to use MATLAB and get instant feedback from within the tutorial. The exercise window is also used to guide the student through the problems using hints. The student is informed if the proposed solution is correct or not and then the student is given access to the actual solution to the problem.

Evaluation of students' mastery of the content occurs at the end of each subsection of the tutorial. This is done using two forms of quizzes: a Summary Quiz and an Exercise Quiz. Each Summary Quiz consists of a number of short questions, the majority of which are multiple choice. These questions review the information presented in the subsection.

The Exercise Quiz consists of problems that use the exercise window. There is one exercise for each page in the subsection. The exercises in the Quiz do not provide hints, but they do provide the answer after students have attempted the solution. The results of the Summary Quiz and the Exercise Quiz are tabulated and displayed using the Results menu item. Keeping a record of the quiz results, which the student can access, provides motivation for the student to seriously attempt the questions.

M-TUTOR WINDOWS TERMINAL SERVER IMPLEMENTATION

Microsoft produces a multi-user operating system product called the Windows 2000 Advanced Server [5]. This product can be used to deliver the Windows desktop, plus the latest Windows applications, to virtually any remote desktop computer including those that cannot run Windows. There are three components that make up the Windows terminal server environment. These

components are the server, the communication link and the client. The server is basically a large computer, with

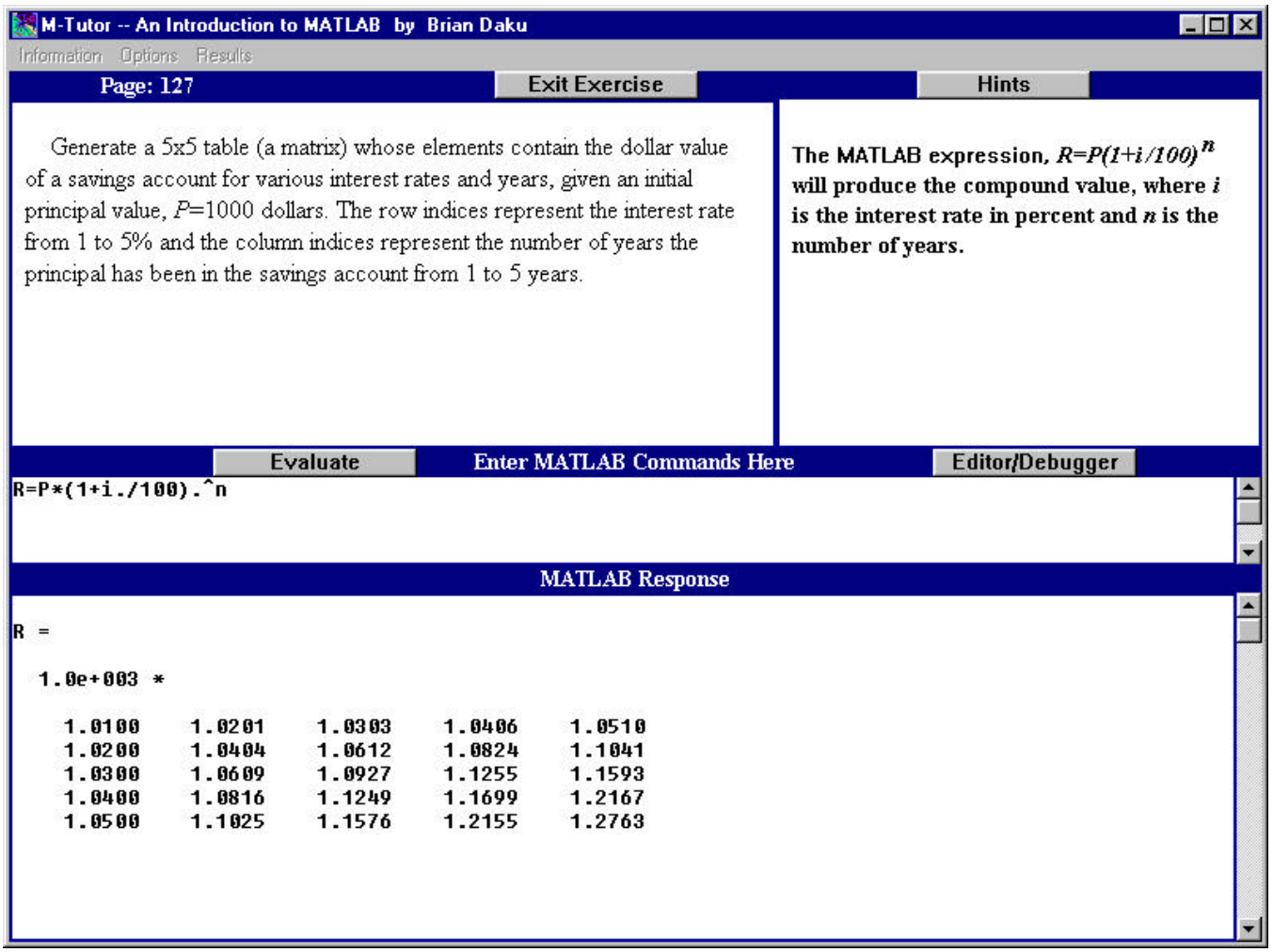

FIGURE 2
EXERCISE WINDOW FROM THE MATLAB TUTORIAL

the appropriate software, that provides virtually all of the computing resources for the server environment. The communication link is the connection between the server and the client. This link uses a Remote Desktop Protocol that relies on a TCP/IP interface. This protocol efficiently moves graphical information across a network. The final component is the client, which is the student's personal computer connected to the internet through a telephone, DSL modem or cable modem.

The student can only access a Windows Terminal Server by first installing the terminal service's client software on their computer. Client software is available for a wide variety of personal computer operating systems, including Microsoft Windows, Apple Macintosh, and Unix. Once the student installs the client software, the computer can be used to log onto a server, over the internet, with a username and password. The student can then run Windows-based applications on the server computer. The execution of the application and all data storage occur on the server computer. Only the keyboard, mouse and display information are transmitted over the internet to the students computer. Multiple students can log onto the server computer, but each student sees only their individual session and each session is managed transparently by the server operating system, independent of other users sessions.

Porting the M-Tutor software to a Windows Terminal Server computer was relatively straightforward, requiring only the following modifications to M-Tutor:

1. All writeable files had to be located on a student accessible hard drive mounted on the server computer. These files are generally used to record student progress within the tutorial. These files are located in a central location on the server, thus, an added benefit is that they can be

easily accessed by the instructor for evaluation purposes.

2. The audio, which is an added feature in the CDROM version, was disabled to reduce the bandwidth requirements to accommodate network connections using low-speed modems.
3. AVI movies were converted to Viewlet [6] based movies that have much lower bandwidth requirements. This was also done to accommodate low-speed modem connections.

The terminal server implementation of M-Tutor was initially tested by 60 second-year electrical engineering students at the University of Saskatchewan. This was done in a four day intensive MATLAB/Simulink camp given prior to the start of classes in September, 2001. This was an on-campus test using University desktop computers to connect to the server. No major problems were identified in these tests. A second test involving 300 freshman engineers was done in January 2002 at Purdue University. Here M-Tutor is part of a WebCT delivery that can be accessed remotely from the campus. The installation and testing of the tutorial was very successful.

PROGRESS/PERFORMANCE ASSESSMENT METHOD

In an academic environment it is important to assess student progress and performance for evaluation purposes. This can be difficult with a self-learning tool such as M-Tutor. The task of assessment can be significantly aided if suitable usage information is collected. The usage information is also worthwhile in terms of the broader goal of assessing the effectiveness of tutorial-based learning environments.

Collecting usage information for M-Tutor was relatively straightforward to implement, since the CDROM version of the tutorial was designed to collect some basic user information in a spreadsheet. The original implementation was modified for the increased requirements of the terminal server implementation. This section describes the type of usage data collected and how the collection process was implemented.

Usage data is collected for the two major tutorial components: content and exercises. In both these categories data is collected on all materials accessed and the rate at which the materials are used. In addition, data collected for the exercise component includes how help information was accessed and the exercise results.

The organization and type of data collected for the content component is shown in Figure 3. Every time a page is accessed the page number and name are entered as a new record in a spreadsheet. While the user is on that page, all hotwords accessed are entered in the same spreadsheet record. Timing data is collected for all page accesses and it is entered on the associated spreadsheet record. The timing data includes: the date the page was entered (In Date), the page departure date (Out Date), the time of page entry (In Time), the page departure time (Out Time), the elapsed time spent on a page (Elapsed Page Time), and the elapsed time since the start of the present M-Tutor session (Elapsed Session Time). The spreadsheet page records are recorded in chronological order.

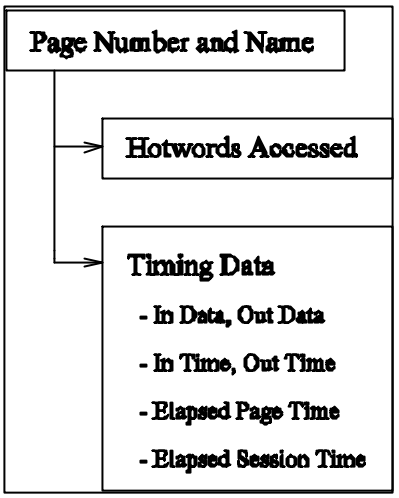

FIGURE 3
CONTENT DATA COLLECTED FOR EACH PAGE ACCESSED

The organization and type of data collected for the exercise component is shown in Figure 4. Every time an exercise is accessed the exercise name is entered as a new record in the spreadsheet. While the user is working on that exercise, timing data is collected and entered in the same spreadsheet record. The exercise timing data includes the date the exercise was started (In Date), the date it was finished (Out Date), the time the exercise was started (In Time), the finish time (Out Time), and the elapsed time spent on the exercise (Elapsed Exercise Time).

Accesses to the help data used by the student are also recorded in the spreadsheet. All hints accessed, relative to when the exercise Evaluate button is pressed (using the terms Before and After), are recorded. An entry is also added to the spreadsheet record if the answer is accessed.

Finally, the exercise result is entered in the spreadsheet record as either correct or incorrect. The solution generated by the student is evaluated using a self-checking MATLAB interface [7]. This interface uses MATLAB to compare the student solution with the correct solution.

A spreadsheet of the content and exercise data is generated for each registered student. The M-Tutor software generates the spreadsheet, and it automatically uses the student's computer login name as the spreadsheet file name. The spreadsheet files are stored on the server

computer and thus can be accessed directly by the instructor. The instructor can use this information to monitor student progress and be alerted to aspects of the content with which the student is having difficulty, or it could be used as part of a for-credit evaluation procedure.

Obviously, for large classes, the logistics of individually reviewing each of the progress/performance spreadsheets can be overwhelming. This problem has been addressed by compiling the individual student spreadsheets into a single Microsoft Access database. This is implemented by running a custom software script to perform the compilation. The Access database can be used to locate individual student usage information and it is also a powerful tool for collecting statistics on class progress and performance in using M-Tutor to learn MATLAB. This data can also be used in conjunction with other class data such as homework and exam grades as well as student background data.

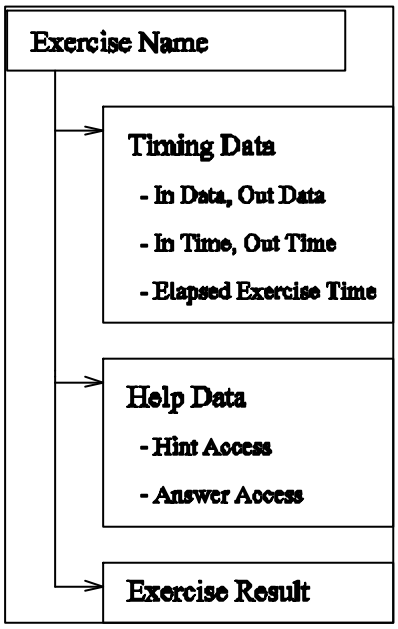

FIGURE 4
EXERCISE DATA COLLECTED FOR EACH EXERCISE ATTEMPED

CONCLUSIONS

The internet has become a very useful medium for delivering course material to students. Internet delivery can be difficult to use for existing standalone software applications such as interactive tutorials that are meant to be installed on the student's computers. One approach to delivering these types of applications, from a centralized location, is to use a terminal server. The advantages of delivering application software from a centralized location include:

1. All students have access to the same version of the software.
2. There are no problems with software installation on student computers.
3. It is possible for the instructor to monitor student progress within the application software, since the software is accessed from a centralized location.

This paper describes the use of a terminal server environment for a standalone application called M-Tutor - An Introduction to MATLAB, which is an interactive computer-based tutorial for learning MATLAB. An overview of the M-Tutor tutorial is presented. This is followed by a section that describes the implementation of M-Tutor on a windows terminal server. The last section identified perhaps one of the most unique and useful features of the terminal server version of M-Tutor, the centralized monitoring and performance evaluation facilities. The progress/performance information collected using these facilities can be used to monitor student progress and identify difficulties, or it could be used as part of a for-credit evaluation procedure.

The terminal server version of M-Tutor has been successfully tested on terminal server installations at the University of Saskatchewan and at Purdue University. In the fall 2002 term, this implementation will be the students primary means of learning MATLAB in Freshman Engineering at Purdue.

ACKNOWLEDGEMENT

This work could not have proceeded without the assistance of computer services staff at both the University of Saskatchewan and Purdue University. At the University of Saskatchewan, individual thanks go to Lorene Turner and Trevor Zintel for their invaluable terminal server support and to Jonathan Moore for writing the software to compile the Access database. At Purdue University, thanks go to Ed Evans and Brad Myers for their invaluable support in integrating the terminal server version of M-Tutor into the Purdue computer system.

REFERENCES

[1] MATLAB Web Site, www.mathworks.com .

[2] Brian Daku, M-Tutor, An Introduction to MATLAB, 2nd Edition, John Wiley & Sons, New York, 2003.

[3] B.L.F. Daku and K.D. Jeffrey, Development of an Interactive CDROM-Based Tutorial for Teaching MATLAB. *IEEE Transactions on Education* **44**(2) (2001).

[4] B.L.F. Daku and K.D. Jeffrey. An Interactive Computer-Based Tutorial for MATLAB. In *Frontiers in Education 2000,* pp. F2D-2 to F2D-7 (2000).

[5] Microsoft Windows 2000 Web Site, www.microsoft.com/windows2000

[6] Qarbon Viewlet Web Site, www.qarbon.com

[7] B.L.F. Daku, A Self-Checking Interface for MATLAB-Based Interactive Exercises, International Journal of Engineering Education **17**(6) (2001).