Int. J. Appl. Engng Ed. Vol. 7, No. 5, pp. 358–367, 1991
Printed in Great Britain.

0742–0269/91 $3.00+0.00
© 1991 TEMPUS Publications.

A CAI Approach to Teach Human Factors in Engineering*

A. HESS†
104 *Marston, Iowa State University, Ames, IA* 50010, *U.S.A.*

Interactive computer-aided instruction (CAI) is offering a new dimension of learning to students by integrating text, graphics, sound, animation, and video to an intelligent network of information. CAI is able to attract more people than traditional learning media and can improve the individual learning success substantially. This paper presents a methodology for the development of CAI. Additionally, a CAI about noise in the working environment is described. The topic is part of an undergraduate human factors course taught in the Industrial Engineering curriculum at Iowa State University. The program is authored in SuperCard 1.0 and stored on CD-ROM.

INTRODUCTION

IN THE last decade, computer-aided instruction has changed significantly. Ten years ago, instructional programs were primarily text oriented and frequently only emulations of books. Users had almost no navigational control over the program and interaction was fairly low. Additionally, most systems depended on large and expensive mainframe computers. Since CPU time was shared by many users, a considerable time delay in executing user commands was not uncommon, which made the programs rather unattractive.

Today, computer technology has made some remarkable improvments. It is now possible to run sophisticated CAI on inexpensive microcomputers. These innovative learning systems realize the concept of hypermedia, which was first envisioned by Vannevar Bush in 1945 [1]. It describes the combination of different types of data such as text, sound, animation, graphics, and video to an intelligent network of information. The information is retrieved non-sequentially by the user in an interactive dialog with the system. Although hypermedia features make instructional programs fairly large, it is possible to store them in a convenient way on CD-ROM. New authoring languages such as HyperCard make the creation of hypermedia applications less difficult, thus decreasing development time and costs.

The impact of CAI is significant. In 1986, Kulik and Kulik [2] performed a meta-analysis of findings from 101 controlled evaluations. The results showed that CAI is able to raise examination scores substantially. In addition, it has a positive effect on the student's motivation and attitude towards the instruction. Instructional time is reduced by one-third compared with conventionally taught lessons. Similar results were found in a major survey of relevant literature by Niemiec and Walberg in 1987 [3]. However, as Hativa [4] reports in an investigation about student's attitudes towards specific features of CAI, no CAI with a fixed set of characteristics can accommodate effectively the learning needs of students. To be effective, a CAI has to be flexible and adaptive to the individual learning requirements.

Since advanced computer technology makes the generation of CAI less difficult, the developer's focus is changing from writing the program code to the instruction itself. Issues such as user-interfaces, program structures, screen layouts, the use of sound, the level of interactivity, etc., are now demanding most of the development time and effort. The methodology for the design of CAI presented here is an attempt to give developers some guidelines in how to address these issues. The methodology was developed together with the CAI 'Noise in the Working Environment' which is described later in this paper. The simultaneous approach made it possible to constantly upgrade the methodology by incorporating issues which appeared during the CAI development. In addition, the effect and validity of the methodology could be seen instantaneously.

METHODOLOGY FOR THE DESIGN OF CAI

Unfortunately, a large number of CAIs are developed by starting with the design of the first card without having an exact idea about the goal, the scope, and the structure of the instructional program. Such an intuitive approach often results in programs which are difficult to navigate and

* Paper accepted 11 December 1990.

† Now with the Hamburg Educational Partnership, Berliner Tor 21. 2000 Hamburg 1. Germany.

which have confusing and cluttered screen layouts. To avoid this, a detailed methodology for the design process was developed. Such a methodology ensures a systematic and effective approach, which is very helpful in keeping the program structure consistent and the outlook homogeneous. Additionally, the number of iterations in the design process can be decreased which leads to a reduction in development time. A significant help in developing the methodology was Apple's Stack Design Guidelines [5]. Other valuable sources are Gagne, Wagner, and Rojas [6], Alessi and Trollip [7], Soulier [8], and Briggs and Wagner [9]. A detailed introduction to the theoretical foundations of instructional design can be found in Gagne and Briggs [10]. The steps of the methodology are as follows:

(1) *Select the right subject*. Not all subjects are equally suitable for CAI. Select a topic which utilizes the power of interactive hypermedia instruction, a topic which can be presented using color graphics, video, animation, simulation, and audio. In this project, noise was chosen as a subject because so many concepts of noise can be explained effectively by simulating them using the sound capabilities of the authoring system.

(2) *Determine the scope of the project*. It is essential to decide at the beginning of the design process which material the instructional program should cover, including the level of detail. This involves consideration of the amount of funds and time available for the project. Instructional programs are never really finished; without a clear limit, development will continue forever. For larger projects a CPM chart is helpful to meet deadlines.

(3) *Design the user-interface*. The interface should be effective, consistent, and intuitive. Use audio and visual elements together with metaphors from the real world such as icons to accomplish these goals [5]. Color coding can be used as an additional channel for transferring information. Give the user maximum control over the instructions; this means that the student can chose his own speed during the session (**self pacing**). It also means that the student can decide where to go next in the program (**self-sequencing**). These possibilities make the instruction adaptable to the individual learning requirements of the user. Provide immediate feedback that confirms the operation carried out. In the CAI 'Noise in the Working Environment' for instance, a click on a button is acknowledged immediately by playing a click sound. Keep the user informed about an operation by using dialogs, alerts, or special indicators such as the watch cursor. Use navigation aids such as 'You are here' and progress indicators [11]. Menus are useful when the program branches to several locations. The ultimate reference about user-interfaces for Macintosh applications is Apple's Humans Interface Guidelines [5], which describes in detail how to use color, text, menus, windows, etc. In case it is intended to create several instructional programs, it is useful to develop only one interface for all of them; this not only reduces development time, it also ensures a consistent learning environment for the students.

(4) *Determine the program structure*. Several structures such as tree or network are possible [11]. Which one is best depends on the actual application; for example, the CAI about noise has a jump-linear structure (see Fig. 3). In any case, the main goal should be to create a program structure which makes navigation easy and efficient; users shouldn't get confused or lost. Draw the program structure out; include all the sections covered by the program and the links between the sections.

(5) *Design the layouts of all cards*. This involves integration of text, graphics, sound, and simulations. All these elements have to work together harmoniously, otherwise users get confused. Avoid crowded and complicated displays, all operations should be intuitive and self-explanatory. It is advantageous to design all card layouts by drawing them on paper as this allows faster and easier changes to be made. The more details are included in the layout the easier coding will be. Don't start coding until all cards have been laid out, otherwise the program is likely to become inconsistent. In addition, it is not very effective to change the code every time you make a change in the program. Don't emulate books; reduce the amount of text as much as possible. Nothing is less motivating and tiresome than reading long paragraphs of text from the screen. High quality color graphics can explain simple facts much better than lengthy written descriptions. For more complex concepts, animations, and simulations should be used. Another way of making reading from the screen easier is to speak and record the displayed text, digitize it, and play it back whenever the corresponding section appears on the screen. In addition, sound can be used to give the user additional information about a subject.

(6) *Create the program*. After all cards have been laid out in such a detail, coding should become fairly easy. A library of standard program modules for test sections, tutorials, etc., are very helpful when it is planned to develop more than one CAI program. This also applies to a library of icons, buttons, and clip art. Even a library of sound tracks can be considered.

(7) *Test the program*. To locate confusing and problematic parts in the program, student testing is essential. Get as much feedback from the test persons as possible. Watch them when they work through the program and see what they do. If several users make a mistake at the same point a redesign of that portion should be considered. Ask the test persons for their comments and recommendations; in this project, a questionnaire was used to get feedback from the students. Implementing students' ideas and comments into the instruction can improve the impact of CAI significantly. If a lot of changes have been made, a second testing will be in order.

(8) *Publish the CAI program*. Before going public, make certain all copyright questions have

been resolved. This concerns not only scanned pictures and figures, but also recorded music used in the instruction as well as external commands or animation drivers which might fall under copyright protection. Check the final version of the program for viruses as you do not want to be responsible for propagating a virus infection.

DESCRIPTION OF THE DEVELOPMENT SYSTEM

As stated earlier, the CAI 'Noise in the Working Environment' was developed together with the design methodology. Development took place on a Macintosh IIcx with 4 Mbyte of RAM and an 80 Mbyte internal hard disk. To store the program during the development process an additional 300 Mbyte hard disk was used. A tape back-up unit ensured data security against hard disk failures. Tape was also used to deliver the program to the disk mastering company. Other hardware available for the project was a CD-ROM drive, a MacRecorder to digitize sound, a black-and-white scanner, headphones, a LaserWriter II, and an ImageWriter II. The new SuperCard 1.0 was selected as authoring system. This program is not only HyperCard compatible which implies all of HyperCard's external commands and functions can be used, it also features additional capabilities such as color and multiple windows. Other supplementary software used in the development process was MacroMind's Director 2.0 for the creation of animations, SoundEdit 2.0 for editing digitized sound, and Think C 4.0 for the generation of external commands and functions.

DESCRIPTION OF THE USER–INTERFACE

A considerable amount of time was spent on the development of a consistent graphical user-interface, based on cognitive psychology as it is discussed for instance by Allen [12] and Barker [13]. The importance of a good interface cannot be stressed enough; if it is poorly designed, the effectiveness of the entire instruction will suffer [14]. The interface developed for this project is self-explanatory; users don't have to read manuals and remember commands to perform tasks. All operations are performed by using the mouse; however, keyboard shortcuts are available for some of the tasks.

The interface gives the user maximum control over the instructional session. Every student can choose his own speed to work through the program (self-pacing). The student can also decide where to go next in the program structure (self-sequencing). These options make the instruction adaptable to the individual learning requirements of the user. A special window offers complete control over the sound features. The user can adjust the volume and shut the recorded voice on or off. Finally, access to all reference information is available at any time.

The interface provides immediate feedback to all user actions. A click on a button for instance is confirmed by playing a click sound. Other actions cause the appearance of dialog boxes or the playback of recorded audio messages.

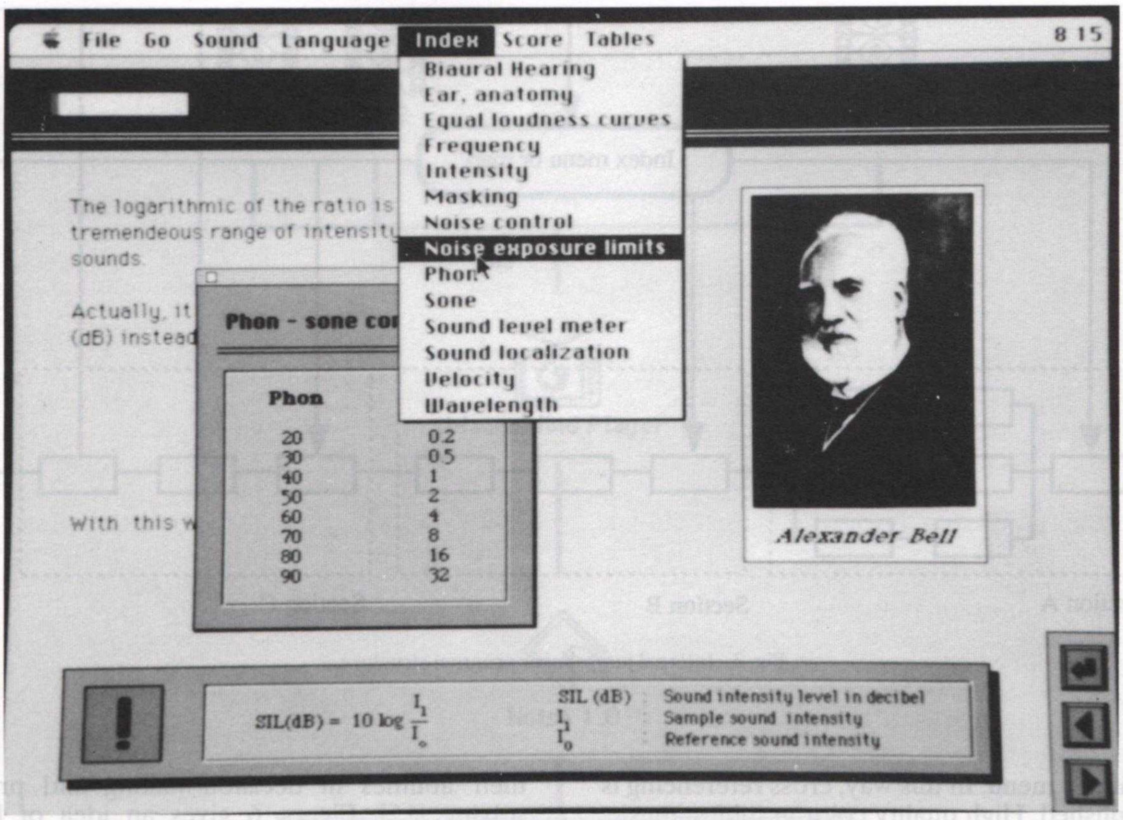
Special emphasis was put on a well-structured and consistent screen layout. Multiple windows and menus avoid crowded and confusing displays. Icons give the user information about options such as using a simulation or getting additional help. Important information is either highlighted in the text or summarized in a special attention box displayed in the lower part of the screen (see Fig. 1).

Navigation through the program is accomplished by using the navigation buttons in the lower right corner of the screen. These buttons enable the student to go forward or backward in the program or to return to a higher level within the program hierarchy. Several navigation aids prevent that users are getting lost in the system. The most important one is the map which gives an overview about the structure of the program (see Fig. 2). The section, the student is working at momentarily, is indicated by flashing the corresponding box in the map. Clicking at one of the boxes transfers the student directly to the section represented by the box. The title of the section is displayed in the upper part of each card (see Fig. 1). Another way of jumping to a different part of the program is using the index menu. This menu lists all important subjects covered by the program in alphabetical order. Selecting one of the items transfers the user to the section where this subject is introduced. A progress indicator in the upper left corner of the screen informs the user about how much more material he has to expect (see Fig. 1).

Consistency with Apple's Desktop Interface was accomplished by following Apple's Human Interface Guidelines [5]. This makes it easy for users familiar with Macintosh applications to operate the program. Figure 1 shows several elements of the user interface. It is intended to use this interface in future CAI as well.

PROGRAM DESCRIPTION

The program consists of about 200 screen displays (cards), retains a size of 50 Mbyte, and is stored on CD-ROM. It is divided into different sections, each presenting a certain aspect of noise in the working environment. Tests and quizzes between these sections provide the user with immediate feedback about his learning progress. A built-in tutorial offers a quick and efficient introduction. As Fig. 3 shows, the program has a jump-linear structure. The linear structure within each section encourages the user to follow the provided path from card to card; this ensures that the information is received in a logical sequence. In addition, it is possible to jump from each card to the beginning of each section by using either the map



The logarithmic of the ratio is
tremendeous range of intensit
sounds.

Actually, it
(dB) instead

With this w

Phon - sone co

Phon	
20	0.2
30	0.5
40	1
50	2
60	4
70	8
80	16
90	32

$SIL(dB) = 10 \log \frac{I_1}{I_o}$

SIL (dB) : Sound intensity level in decibel
I_1 : Sample sound intensity
I_0 : Reference sound intensity

Fig. 1. Typical screen display showing some elements of the user interface.

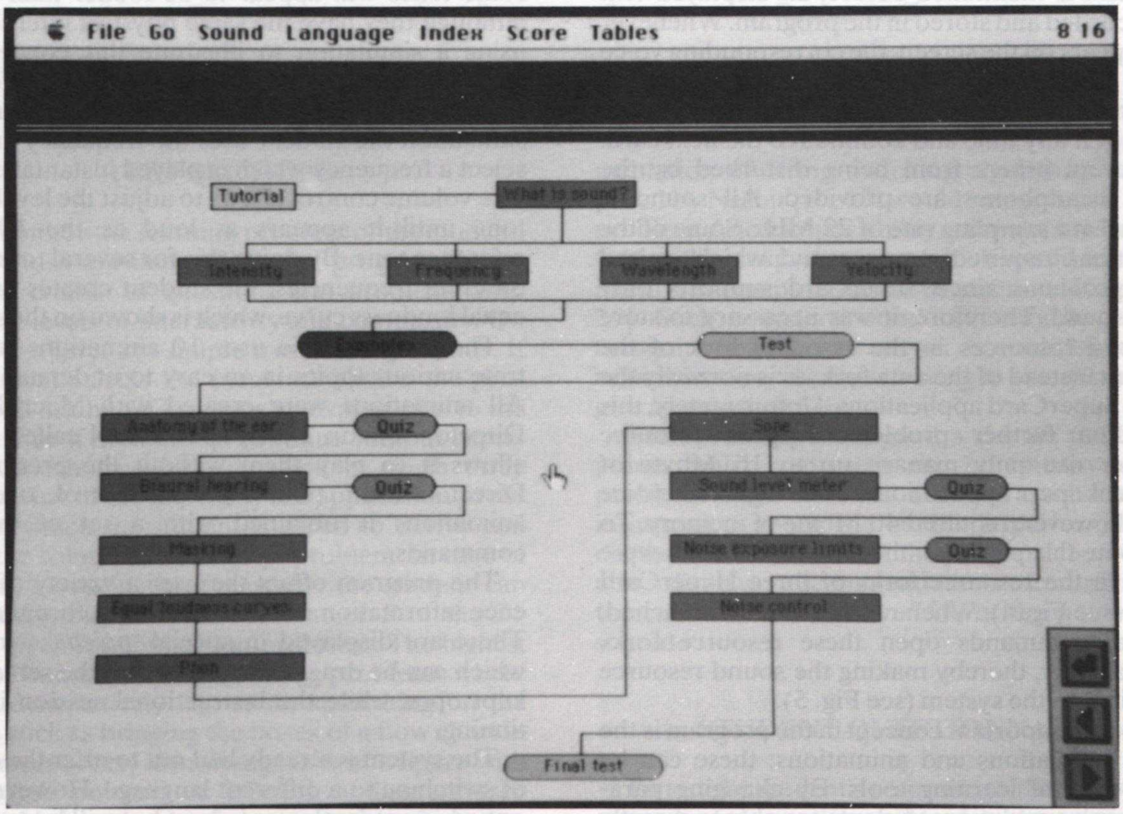


Fig. 2. The map symbolizing the program structure.

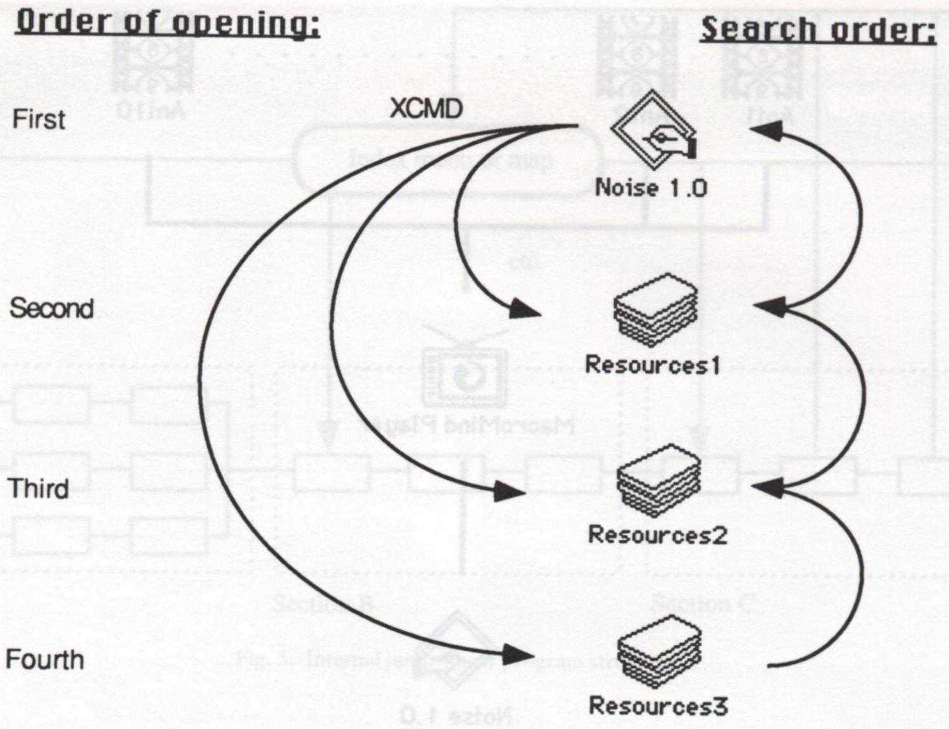
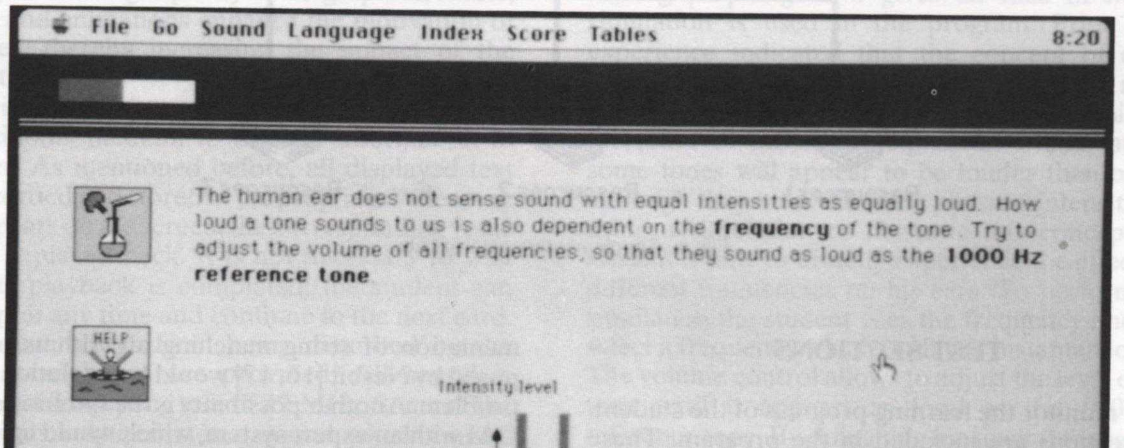


Fig. 5. Handling of sound resources.

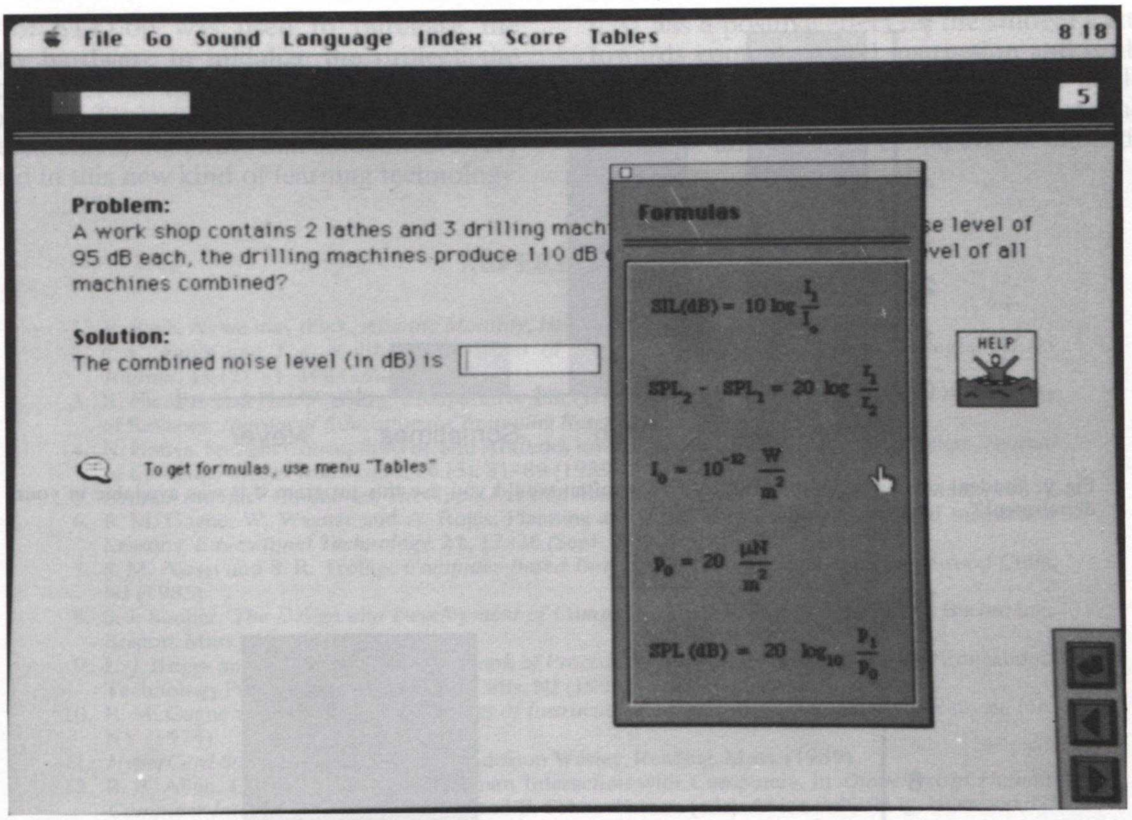


The human ear does not sense sound with equal intensities as equally loud. How loud a tone sounds to us is also dependent on the **frequency** of the tone . Try to adjust the volume of all frequencies, so that they sound as loud as the **1000 Hz reference tone**

If you have problems conducting the experiment, use the help button!

Fig. 6. Screen experiment explaining equal loudness curves.

human factors class were asked to work with a beta version of the instructional program. As Fig. 8 shows, most of them had little or no experience with Macintosh computers. The students were monitored while they were working with the program to find out what answers they tried and where they had navigation or other problems. After finishing the test, they were asked to fill out a questionnaire about their feelings towards CAI in general and the program ‘Noise in the Working



Problem:
A work shop contains 2 lathes and 3 drilling mach se level of 95 dB each, the drilling machines produce 110 dB vel of all machines combined?

Solution:
The combined noise level (in dB) is □

To get formulas, use menu "Tables"

Formulas

$$SIL(dB) = 10 \log \frac{I_1}{I_o}$$

$$SPL_2 - SPL_1 = 20 \log \frac{I_1}{I_2}$$

$$I_o = 10^{-12} \frac{W}{m^2}$$

$$p_o = 20 \frac{\mu N}{m^2}$$

$$SPL\,(dB) = 20 \log_{10} \frac{p_1}{p_o}$$

Fig. 7. Typical problem in a test section.

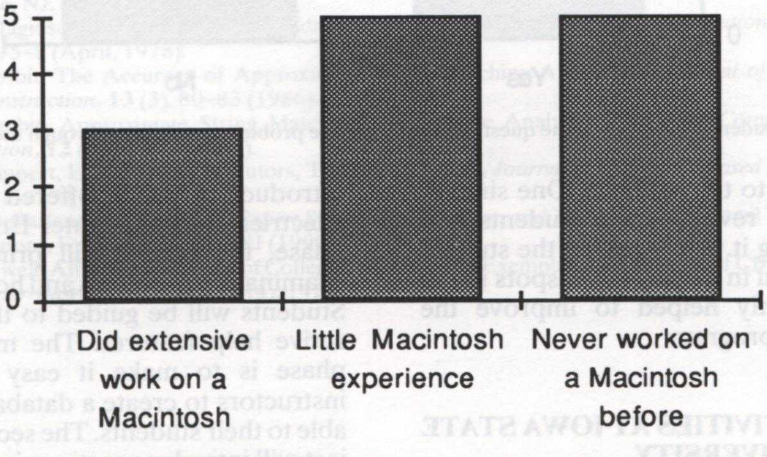


Fig. 8. Macintosh experience of volunteer students participating in testing.

Environment' in particular. The outcome of the questionnaire matches with similar studies [2, 3, 20]: CAI produces positive changes in student attitudes towards computers and instruction. The general reaction towards CAI was overwhelmingly positive, almost all test candidates said they would use the program, if available, 'very often' or 'often' (see Fig. 9). Most students found self-pacing and self-sequencing extremely helpful. Only two students stated problems in navigating through the progam (see Fig. 10). Other positive comments concerned sound, simulations, and the ability to get instant help while solving problems. Several students said they were motivated by the color graphics and the sound features. Some of the negative remarks included repetitive computer responses such as 'Yes, this is correct!', some of the sound features, and the difficulty of conducting some simulations. The comments were used to improve the final version of the program before it was stored on CD-ROM. Several cards were added to the tutorial to explain certain features of the program in more detail. The section with sample problems was expanded to offer the students more guidance in how to apply the achieved knowledge. Some parts in the program structure were found confusing by the students; these were improved by changing the program structure or by giving

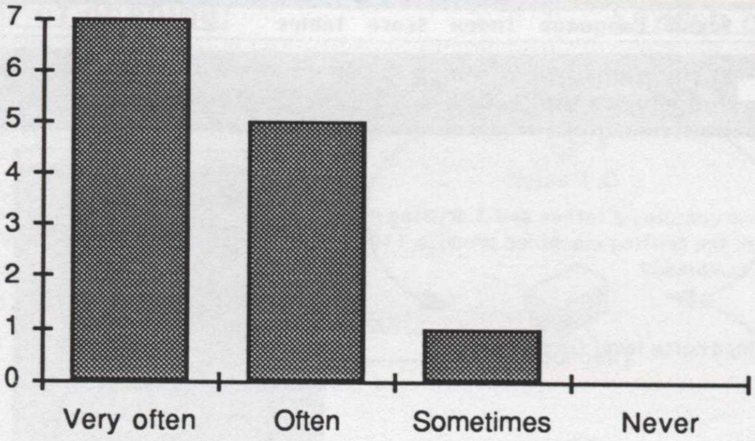


Fig. 9. Student responses to the question: 'How often would you use this program if it was available in your department?'

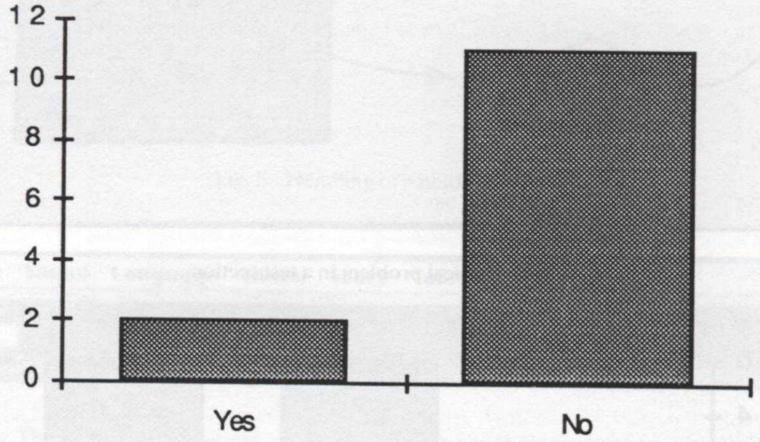


Fig. 10. Students responses to the question: 'Did you have problems navigating through the program?'

additional directions to the students. One simulation was completely revised since students had difficulties conducting it. All together, the student testing was very useful in finding weak spots in the program and certainly helped to improve the overall quality of the program.

FURTHER CAI ACTIVITIES AT IOWA STATE UNIVERSITY

Although the CAI 'Noise in the Working Environment' has been developed and published successfully, research in the area will continue. It is planned to develop a second CAI program using the new SuperCard 1.5. The topic of this instruction has not been determined yet but another subject from the *human factors* class is likely. Unlike its predecessor it will feature full motion video using interactive video. To avoid the need for a separate video monitor, a color video card has been purchased. It is felt that this new media will improve the quality of the instruction substantially.

Another project at Iowa State University uses AUTHORWARE to develop a CAI about logic design. This instructional program will be used in an introductory course offered in the Department of Electrical and Computer Engineering. In the first phase, the system will primarily consist of old examination problems and homework assignments. Students will be guided to the solutions by interactive help features. The main objective of this phase is to make it easy and convenient for instructors to create a database of problems available to their students. The second phase of this project will introduce sections in which basic concepts of logic design are presented. These sections will feature graphics, simulations, and sound.

Finally, Iowa State University is also a member of a coalition of universities and companies funded by the National Science Foundation to develop NEEDS, the National Engineering Education Delivery System.

CONCLUSION

This project has shown that computer technology today allows it to create sophisticated CAI in a relatively short time; the program 'Noise in the Working Environment' was developed during one year in about 800 man-hours. The equipment budget of $14,000 was used to purchase the necessary hardware to initialize the project; the development of the next programs will be much less expensive. The questionnaire results, as well as other research, demonstrate that students are very interested in this new kind of learning technology. CAI has a positive effect on the student's attitude towards computers and instruction and is able to improve scores. A new generation of intelligent CAI together with the upcoming DVI technology will show an even greater impact on education in the future.

REFERENCES

1. V. Bush, As we may think, *Atlantic Monthly*, 101–108 (July 1945).
2. C. C. Kulik and J. A. Kulik, Effectiveness of Computer-Based Education in Colleges, *AEDS Journal*, **19** (2), 81–108 (1986).
3. R. Niemiec and H. J. Walberg, Comparative Effects of Computer-Assisted Instruction: A Synthesis of Reviews, *Journal of Educational Computing Research*, **3** (1), 19–37 (1987).
4. N. Hativa, Student Conceptions of, and Attitudes toward, specific Features of a CAI System, *Journal of Computer-Based Instruction*, **16** (3), 81–89 (1989).
5. *Human Interface Guidelines: The Apple Desktop Interface*. Addison Wesley, Reading, Mass. (1987).
6. R. M. Gagne, W. Wagner and A. Rojas, Planning and Authoring Computer-Assisted Instruction Lessons, *Educational Technology*, **21**, 17–26 (Sept., 1981).
7. S. M. Alessi and S. R. Trollip, *Computer-Based Instruction*. Prentice-Hall, Inc., Englewood Cliffs, NJ (1985).
8. S. J. Soulier, *The Design and Development of Computer Based Instruction*. Allyn and Bacon, Inc., Boston, Mass. (1988).
9. L. J. Briggs and W. W. Wagner *Handbook of Procedures for the Design of Instruction*. Educational Technology Publications, Englewood Cliffs, NJ (1981).
10. R. M. Gagne and L. J. Briggs, *Principles of Instructional Design*. Holt, Rinehart and Winston, Inc., NY (1974).
11. *HyperCard Stack Design Guidelines*, Addison Wesley, Reading, Mass. (1989).
12. R. B. Allen, Cognitive Factors in Human Interaction with Computers, in *Directions in Human-Computer Interface Design*, A. Badre and B. Schneiderman (eds). Ablex Publishing, Norwood, NJ (1982).
13. P. Barker, *Basic Principles of Human-Computer Interface Design*, Hutchinson, London (1989).
14. J. R. Miller, The Role of Human-Computer Interactions in Intelligent Tutoring Systems, in *Intelligent Tutoring Systems* M. C. Polson and J. J. Richardson (eds). Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 143–189 (1988).
15. E. M. Magidson, Issue Overview: Trends in Computer-Assisted Instruction. *Educational Technology*, **18**, 5–8 (April, 1978).
16. J. C. Nesbit, The Accuracy of Approximate String Matching Algorithms. *Journal of Computer-Based Instruction*, **13** (3), 80–83 (1986).
17. J. C. Nesbit, Approximate String Matching in Response Analysis. *Journal of Computer-Based Instruction*, **12** (3), 71–75 (1985).
18. R. C. Lippert, Expert Systems: Tutors, Tools, and Tutees, *Journal of Computer-Based Instruction*, **16** (1), 11–19 (1989).
19. T. D. McFarland and R. Parker. *Expert Systems in Education and Training*, Educational Technology Publications, Englewood Cliffs, NJ (1990).
20. J. V. Powell, Affective Response of College Students to an Exemplary Application of CBI, *Journal of Computer-Based Instruction*, **14** (4), 142–145 (1987).