A. V. LOUIS BIGGIE, WILLIAM E. BUCHANAN, PAUL L. HAZAN, and ALEXANDER KOSSIAKOFF

A MULTIMEDIA RAPID PROTOTYPING TOOL FOR THE DEVELOPMENT OF COMPUTER-ASSISTED INSTRUCTION

A rapid prototyping tool has been designed to aid in the creation of computer-assisted instruction (CAI) software for children with learning disabilities and mental retardation. The tool, which was conceived and developed under the collaborative program between APL and the JHU Division of Education in the School of Continuing Studies, has enabled a multidisciplinary team of educators and computer engineers to visualize and test all features of proposed CAI programs on-line during regular design sessions held around a conference table. Computer program development time has thereby been significantly shortened, and significant gains have been made in the quality of educational products produced.

INTRODUCTION

A team of senior APL staff, JHU Division of Education faculty, and teachers of special education have collaborated since 1983 in the development of computer-assisted instruction (CAI) for students with learning problems. Although the initial products created by the Hopkins team have met with widespread approval of educators across the country, the team's first CAI programs took a very long time to define and implement. The main purpose behind developing a rapid prototyping tool, therefore, was to compress CAI software development time. Use of the tool, however, has also fundamentally improved the quality of the resulting educational products by affording educators increased opportunity to participate both in system design and in detailed decisions that are best made before time is committed to the writing of program code.

SPECIAL CAI NEEDS FOR SPECIAL EDUCATION

Software for CAI is, for the most part, produced and distributed by small commercial publishers. Although a broad range of high-quality CAI is available for general education students in preschool through grade 12, special educators have found that very few of the many thousands of available products address the curricular needs and individual strengths and weaknesses of their students. Since CAI for special education is expensive to develop and the market is small, the promise of the computer as a patient and versatile tutor of students with disabilities remains largely unfulfilled.

The features discussed in the following sections distinguish the requirements of CAI for students with serious learning problems from CAI that has proved successful in helping students who do not have perceptual, memory, neuromuscular, or intellectual deficits. Implementation of the required special features can transform an otherwise straightforward computer program into a complex and memory-intensive one. The final product, which is often a complicated program, needs to run in the small computers available to special education, typically 64-KB Apple IIs. Programs need to be fully optimized to fit in such a small configuration, a requirement that makes design modifications very difficult.

Individualization

A special education student is often characterized by a substantial discrepancy between ability and performance. Techniques that work for most students frequently are not effective in the special education classroom. What the exceptional child, and particularly the child with severe learning problems, needs are instructional approaches that make the most of strengths, interests, and preferred learning styles while remediating and/or minimizing the need for knowledge and skills that may be deficient.

The need for individualization has led the APL-JHU team to develop an authoring system that enables teachers to specifically tailor CAI to the developmental and chronological ages of their students and to special curricular needs as well (see the boxed insert entitled "Authoring Systems"). The Hopkins system is menu-driven for simplicity of use and designed so that a teacher without computer training can design a lesson that is complete and ready for student use in about one-half hour.

Feedback and Branching

In a typical CAI program, the student is asked to respond to questions, and the program either progresses automatically to the next question or determines the next question on the basis of the student's response or pattern of responses. The instructional approach for students in general education tends to be heuristic: the student is to find out which answers are correct and

AUTHORING SYSTEMS

An authoring system usually has two essential software components: a lesson program, which provides a learning experience for one or more students, and a program that allows a teacher or curriculum specialist to design lessons. Such a system is called an authoring system because it enables an author (the teacher) to "write" CAI lessons based on an established curriculum or on a student's individual needs. The author's choices of subject matter and presentation are written into a data file. When the lesson is presented to the student, the data file is read into memory, and the lesson runs in accordance with the author's choices or specifications. It is desirable for the lesson program design to provide the teacher with quick-change ("fast-track") capabilities so that the teacher can fine-tune each lesson to meet the physical and perceptual limitations of individual students or classes. CAI lessons with limited provisions for changes before running are often referred to as "shells." In contrast, a true authoring system enables a teacher to control all important elements of a lesson—as a minimum, the subject matter, questions, distractors, rate of presentation, feedback for right and wrong answers, and some branching based on student responses.

which are wrong, determine the underlying learning principles, and make orderly progress toward mastering the material. Special education students often cannot arrive at generalizations in such a straightforward fashion. They need explanations and special help as they go along. The feedback that they receive should be frequent, consistent, and explicit. All information presented must be carefully phrased and filtered to eliminate distraction and confusion.

One recent CAI program, developed by the Hopkins team as part of a package to help students with learning disabilities improve their reading skills, illustrates the special feedback and branching requirements. The skill being taught is the recognition of analogies, and one of the modes of teaching the skill requires the student to identify an analogy by its class. For example, one relationship between a wing and an airplane is that the wing is a part of the whole airplane, that is, a part–whole relationship. In a typical CAI lesson program, it would be adequate to follow an incorrect response with an indication of the correct analogy type and then move on to the next question. This would be the usual CAI "drill-and-practice" approach. A program for students with learning problems, however, demands a more sophisticated computer response to a student's incorrect answer. In the Hopkins CAI analogy lesson, the student can be told that "a wing is part of an airplane, just as a wheel is part of a car." This response helps the student in three ways: it states the relationship explicitly, it uses a second analogy to help the student generalize the concept, and it uses simple vocabulary in the second analogy to make sure that the student is focusing on the concept. The lesson program, therefore, is sensitive to the type of error being made; it determines whether the error is due to a misunderstanding of the concept, a lack of vocabulary, or confusion over the process. An explanation such as providing a second analogy would not help a student who had simply pressed the wrong key. CAI for disabled learners must anticipate all likely sources of error.

Record Keeping

Most question-and-answer (drill-and-practice) educational software programs provide the student with a score at the end of a lesson. In some cases the score is recorded on disk for future reference by the teacher. Teachers in special education are very much concerned with diagnosing their students' problems and are legally required to document information on student progress toward agreed-upon, short- and long-range educational goals. Accordingly, detailed records leading to informed analysis of a student's preferred learning styles, aptitudes, and need for remedial aids must be kept. Programming the computer to record and analyze the student's learning behavior in a way that can be immediately helpful to the teacher in offering special help and in authoring subsequent lessons adds further complexity (and interest) to the CAI designer's task.

Student and Teacher Interfaces

Special education CAI lessons must provide easy access for students with neuromuscular and sensory impairments and must be absolutely "bulletproof" to improper input procedures and rough keyboard treatment. They also must be adaptable to students with physical handicaps through the provision of special switches, touch screens, and other pointers. Instructions to students advising them on how to make selections and the selection procedure itself must also be carefully designed and tested. Similarly, the process by which teachers author individualized lessons for their students must be intuitive, menu-driven, and consistent throughout. The Hopkins team has taken great care to make its products easy to learn and use both by the students and by their teachers, the authors of student lessons. Considerable design effort is expended on the precise wording, size, and arrangement of instructions on the screen and on designing each screen so that information of a specific type (feedback, prompts) always appears at the same place, at the same time, and only when needed. "Help" features are available to further clarify the process, and, for students, voice and graphics are employed when needed to clarify concepts, instructions, and vocabulary.

Multimedia Features

Teaching students with severe learning disabilities to read is best accomplished through a highly individualized, labor-intensive process involving the rigorous application of multisensory information. In learning words and understanding sentences, students are presented with phonemes, words, phrases, and sentences that they can see, hear, and literally feel as they write. The method, which is known as VAK or VAKT (visual, auditory, kinesthetic, and tactile), is best offered in a one-on-one teacher–student setting or in a group in which a highly skilled practitioner works with two or three students in a special resource room or a reading clinic. Without this concentrated attention, students with severe reading problems are unable to achieve at their grade level and are prone to fall seriously behind in all areas of their schoolwork. The ability of a computer to present materials to students consistently at the proper pace and with a sensitive, built-in, yet adaptive understanding of the problems the student is facing can be a powerful aid to learning when multimedia graphics, speech, and live television action are properly integrated in the CAI.

PROBLEMS OF THE MULTIDISCIPLINARY TEAM IN THE DESIGN ENVIRONMENT

A central problem of conventional approaches to software design is the difficulty of visualizing the final product. A step-by-step textual description of a computer program is subject to a different interpretation by each member of a design team. Surprises often occur when the application, or domain, experts (in the Hopkins case the teachers and special education faculty) are not experienced in software design. They may find the coded results to be a reasonable approximation of what they had in mind, yet not satisfactory in certain important details—often the details they did not visualize during the design session or did not indicate precisely in their paper screen designs. The applications expert may attribute the discrepancy to misinterpretation or lack of sophistication on the part of the programmer, whereas the programmer may blame the ambiguity or the inadequacy of the specification. In any case the first product is rarely satisfactory, a great deal of programming time is wasted, and frustrations can mount from session to session. Partial products are plentiful, completed products are few, and satisfactory ones are fewer still.

What stimulated the Hopkins team to work on a rapid prototyping tool was the conclusion, after years of hammering out designs the hard way, that static paper specifications, no matter how carefully annotated and documented, could not satisfactorily convey the dynamics of the final product and, furthermore, represented a formidable barrier to expeditious design. They realized that significant progress in the design of CAI lessons and authoring systems could be achieved only by creating a special software tool that could display an approvable program module on a computer screen.

PROTOTYPING TOOL REQUIREMENTS

The prototyping tool was seen to have several basic requirements and some additional capabilities that represented state-of-the-art advances, at least in CAI software design. First, the use of the prototype should require no coding. Taking advantage of newly developed interactive software environments, team members should be able to access buttons, pull-down menus, and self-explanatory dialog boxes to assemble a dynamic model of CAI lesson software. The model should be operational and interactive so that designers can see the display as well as test programs to evaluate the format, logic, and flow of the lesson sequences and the ease of use of the student interface. The tool should be versatile enough to accommodate many different kinds of changes and additions on-line during a design session. The team's experience had shown that features needed to be added, tested, discarded, and substituted at will and at a pace fast enough to keep up in real time with the ideas generated during design sessions by a creative, uninhibited group. In short, the construction set should be incremental, flexible, versatile, and totally interactive.

A MULTIMEDIA RAPID PROTOTYPING TOOL FOR DEVELOPMENT OF CAI

The new class of high-level microcomputer tools combining "hypermedia" programming, fast and easy-to-use interfaces, high-resolution screens, and an array of new peripherals arrived at precisely the right time to help the Hopkins design team in its mission of creating a powerful new system for authoring CAI. On the basis of the special requirements discussed in the preceding section, one can appreciate a special education teacher being given the means to create a family of instructional software that has the multisensory ability to tutor students in the skills and process of effective reading. Such a system and the lessons it makes possible can come into being only after months of experimentation and "give and take" by members of a multidisciplinary design and development team. The ensuing discussion describes how those months have been reduced to days and, in some cases, hours following the development of a Multimedia Rapid Prototyping Tool (MRPT) based on HyperCard (see the boxed insert entitled "Hypercard").

HYPERCARD

HyperCard is a system software package developed by Apple to run on the Macintosh. Apple describes it as an "information environment" that can be used to store words, numbers, charts, pictures, sounds, and any other kind of information that can be stored by a computer. HyperCard is based on the Hypertext concept, which enables users to easily link one piece of information with another. In HyperCard, since information is not limited to textual data, this feature might be used to link a picture of an object with a textual explanation.

A HyperCard file typically consists of screens on which information is displayed. Each screen of information is called a card, and a collection of cards is called a stack. The user interacts with a stack by using a mouse to move a pointer. By moving to a particular area of the screen and clicking the switch on the mouse, the user can activate software routines. Screen areas selected in this way to engage embedded software routines are called buttons.

HyperCard comes with its own programming language, Hypertalk. Hypertalk can be used to write high-level, almost conversational code associated with HyperCard objects (buttons, cards, and stacks). Additionally, software developers may use a general-purpose programming language such as Pascal or C in developing routines to be used in HyperCard.

MRPT PROGRAM DESCRIPTION AND OPERATION

The MRPT program, developed at APL and put into use in 1988, runs on a Macintosh computer in the HyperCard environment (see the boxed inserts entitled "HyperCard" and "MRPT Hardware Considerations"). The code is written in Hypertalk, the high-level programming language that comes with HyperCard. Additional routines have been appended to the HyperCard shell to support peripherals (Fig. 1).

HyperCard enables nonprogrammers to create a wide range of application programs. Routines, functions, and graphics are invoked by an intuitive system of pull-down menus, icons, and interaction with the mouse. HyperCard also has all the features of a conventional database. A central feature used by the MRPT is HyperCard's ability to store routines in the form of "buttons," which, if accessed with the mouse, execute a software routine without the user ever having to see or even be aware of the code that is associated with it. Graphics, voice, music, and access to videodiscs are easily incorporated into HyperCard-based designs.

The design of the MRPT involved the merging of the features native to HyperCard with routines that were identified as necessary for designing CAI for special education. The code associated with the program functions had to be transparent to the user and easily accessible. Therefore, a menu-driven system based on a simulated control panel was designed.

To allow the designer to focus on the design of a system application rather than on the details of its implementation, software routines have been incorporated to merge several steps into one or two mouse clicks. This feature makes implementation of software functions considerably faster than in the raw HyperCard environment. The MRPT also provides access to all the routines built into HyperCard. For example, a user can access a bit-mapped painting program, which includes all the tools usually used by computer graphic artists.

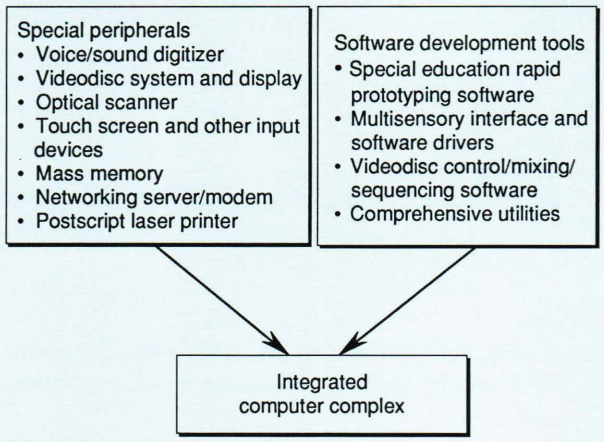
The MRPT enables the Hopkins CAI design team to create realistic models of student lessons and authoring protocols without writing code. Educators and computer system designers are thus able to focus on their respective fields of expertise. Routines can be invoked in the model instantly so that a wide range of ideas and potential system configurations can be explored and evaluated. Misunderstandings among members of the design team are minimized because the system permits the visualization of each software module and of its role in the final product. Design alternatives are presented in a realistic environment. Because the model is operational, the tests to verify usability of the system begin in the design session rather than after the system has been programmed and placed in the field. Such real-time rapid prototyping is more likely to yield first-pass success and thereby avoid the time-consuming process of redesigning and reprogramming software.

The tool facilitates the design and scrutiny of the layout and organization of text, graphics, and multimedia material; the use of the keyboard and other input

MRPT HARDWARE CONSIDERATIONS

The MRPT runs on a Macintosh computer with a minimum of 2 MB of random-access memory (RAM) and a hard disk drive. An optical scanner is used to read pictures into student lesson programs, and Macrecorder, an inexpensive speech digitizer, provides a means of incorporating speech. During meetings of the multidisciplinary development team, a liquid-crystal display projection device is used with a high-quality overhead projector to display the Macintosh screen. The programmer spends the meeting at the keyboard continuously orchestrating and implementing design ideas as they arise.

After a substantial part of a design (e.g., a lesson or authoring module) is approved by the team, the prototype MRPT-based modules are coded on an Apple IIgs, which is part of the development laboratory. The resulting authoring system programs used by teachers are designed to run on Apple IIe/IIgs computers with 128 KB of RAM, two disk drives, and a printer. Student lessons generated by authoring programs are configured to run also on all Apple II machines, including the older Apple II Plus computers with only 64 KB of RAM and one disk drive.


Special peripherals
• Voice/sound digitizer
• Videodisc system and display
• Optical scanner
• Touch screen and other input devices
• Mass memory
• Networking server/modem
• Postscript laser printer
Software development tools
• Special education rapid prototyping software
• Multisensory interface and software drivers
• Videodisc control/mixing/sequencing software
• Comprehensive utilities
Integrated computer complex

Environment for multimedia rapid prototyping.

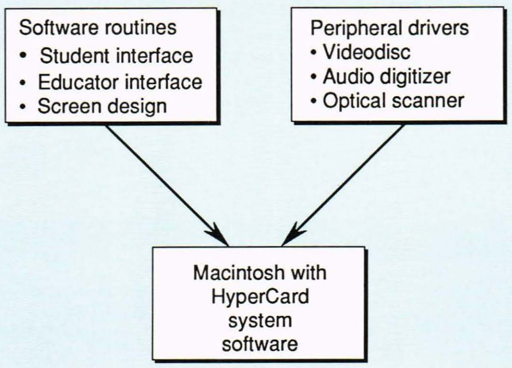


Figure 1. Multimedia rapid prototyping software.

devices; branching control; feedback; and record keeping. It allows several candidate display models to be conceived, created, displayed, and assessed at a single design session. The result is a more effective design implemented in a shorter time.

The MRPT software display, illustrated in Figure 2, contains two major components: a control panel to manipulate the screen design and simulation, and a screen on which the simulated program is run. The control panel contains several buttons that activate a library of software routines, each of which performs a specific function involved in designing authoring or lesson screens. A function is turned on by using a mouse to move a pointer to the appropriate button.

Functions

Figure 3 illustrates the use of control buttons to add a new menu function to a screen, namely, an additional command function. This feature uses the control button in the lower section of the control panel called "New function," which creates new commands in the prototype (Fig. 3A). When selected, this button displays a pop-up menu that offers the user a choice of functions (Fig. 3B). The selected function is highlighted, and the feature is placed in the program in the form of a command key. The newly created key appears in the screen area, and the software associated with it is automatically and transparently transported (Fig. 3C). It can now be accessed, activated, and tested with the mouse. Its position, appearance, and parameters can be altered by using point-and-click dialog boxes.

A wide range of functions is included in the MRPT software, and new ones will continue to be added to respond to existing and anticipated needs.

Cards

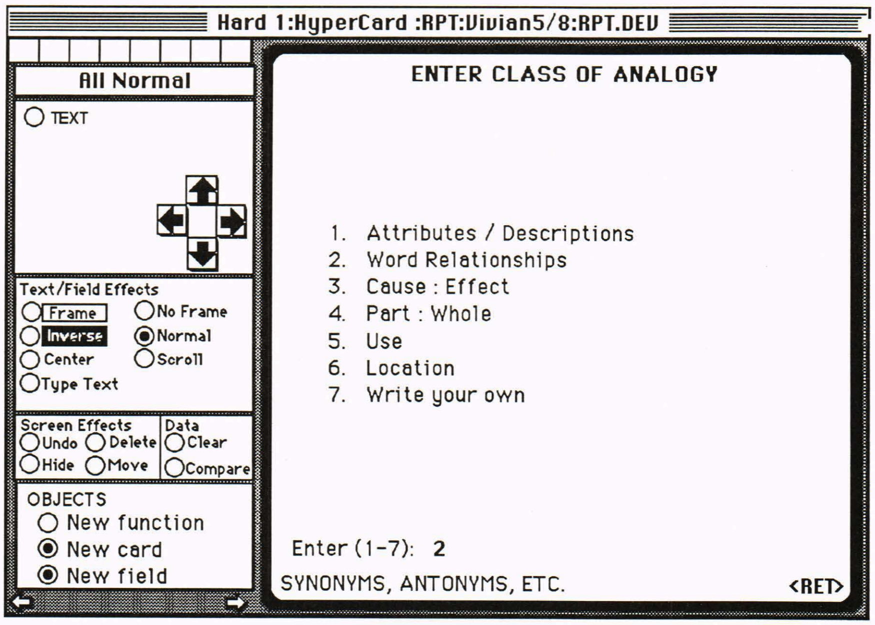
In the HyperCard environment, each screen of information is organized in a relocatable module called a "card." This feature enables program modules to be easily referenced and recalled. Control commands can quickly define links between one card and another. This ability allows designers to simulate a working program easily and quickly by creating new cards and linking them. For example, if the design team suggests that a feature be included to create a certain effect, a card is created to show the appearance of the screen before the feature is activated; a second card shows the screen after the feature is activated. Finally, a screen command in the form of a button links the first card to the second. The feature is now operational in the prototype and can be activated with the button, but no programming has been required to produce the effect.

Cards can also be used to create animation. For example, graphics can be placed on a series of cards, which are then displayed in rapid sequence. This feature is also used to demonstrate the movement of textual material from one part of the screen to another.

Cards are frequently employed so that designers can easily view the general flow of a program, and provision is made so that the sequence can be altered. The sequence of data presentation is frequently a critical element in educational software, and this easy means of evaluating alternative sequences has had very positive effects on determining the most pedagogically sound version.

Fields

Typically, a student lesson screen consists of several areas or fields devoted to different uses: an instruction field, a student question field, a student answer field, and a feedback field. A new field can be created by clicking on the "New field" button. The field can be placed anywhere on the screen, and pop-up menus and dialog boxes allow the program designer to specify its type. As far as the program designer is concerned, the MRPT makes the manipulation of the field type as easy as mak-



ENTER CLASS OF ANALOGY

1. Attributes / Descriptions
2. Word Relationships
3. Cause : Effect
4. Part : Whole
5. Use
6. Location
7. Write your own

Enter (1-7): 2
SYNONYMS, ANTONYMS, ETC. <RET>

Figure 2. Sample MRPT screen. The control panel (left) controls the rapid prototype program. The buttons (labeled circles) activate a library of software routines, each of which performs a specific function involved in designing authoring or lesson screens. A function is turned on by using a mouse to move a pointer to the appropriate button. The prototype program is run in the screen area (right).

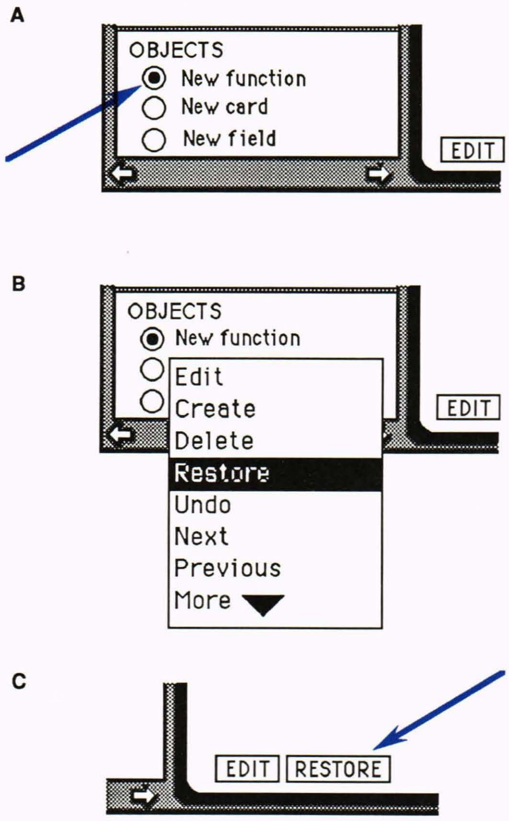


Figure 3. Creating a "Restore" function with the APL-JHU MRPT. A. The user activates the "New function" option by pointing and clicking. B. A pop-up menu appears to allow the user to choose the new function. C. The system model now has a fully operational "Restore" function. The user can now use the mouse to position it.

ing selections from menus. The approach is identical to the selection of the function of a button (see Fig. 3). Changing the field type automatically alters the code associated with it so that it plays a different role in the program. For example, an instruction field is usually noninteractive because its purpose is to provide information to the student. Thus, the code embedded in the instruction field prevents the student from typing information in that area of the screen. When a student response is needed, an answer field can be created so that the student can input data.

Program designers may also specify whether or not a field is visible. Making a field temporarily invisible permits the prototype program to be designed so that information is progressively disclosed. The order and length of time that a field is either visible or invisible can critically alter the style of teaching. Also, under certain circumstances, the visibility of a field can be placed under student control, thereby providing the ability to create "Help" screens that the student can access at will. Other fields may be permanently invisible to the student and serve only as data sources. Provision is made so that these fields can be fed from other data sources, such as word processors.

Additionally, the appearance of a field can be quickly altered by the MRPT. This feature allows the character size, font, justification, and margins within the field to be altered. A whole field can be changed from uppercase to lowercase, for example, simply by clicking the mouse twice.

Multimedia Interface

Integration of computer and video technology holds great promise for the creation of more interesting, meaningful, and effective educational materials than in the past. Programs can also be made to switch from medium to medium adaptively. For example, text can be used when it is the most suitable medium, but software can switch the program to voice, graphics, and video as needed.

Routines embedded in the MRPT allow program designers to access multimedia peripherals and incorporate their use into the system model. Media branching also creates exciting possibilities for the educator concerned with the assessment of learning disabilities. For example, a traditional reading test often does not clearly indicate whether a student's problem is linked to a reading disability or to difficulty with the subject matter being presented. A multimedia program can help solve this problem by reporting and comparing a student's performance on the same or similar materials presented in a variety of media.

Videodisc Players

A videodisc player can play back film and video material just as a videocassette recorder can, but with many extra features that have interesting implications for education. Videodisc players can be controlled by computers on a frame-by-frame basis. An educational program can be designed in which students might see a film clip and then answer questions about what they saw. The film can then branch to another sequence that might explain an error made in the answer or go to new material, depending on the student's response.

With the MRPT, this type of educational or assessment program can be prototyped, viewed, and evaluated instantly. The videodisc player is controlled by an interface on the computer screen that looks very much like a remote control unit for a home videocassette recorder. In this way, control functions can be created that switch the videodisc player on or off, replay a segment, or move to a new scene in a film possibly containing a hint or some feedback.

Other features of the videodisc player include the provision of two separate audio tracks. This feature has been used in the past to create bilingual discs but could also be used very advantageously by special educators. For example, by changing tracks the program can switch the program from an expository presentation to questions, followed by feedback to the student based on the correctness of answers.

When integrated with a computer system, videodisc technology allows the user to annotate pictures by overlaying text or graphics. The MRPT software makes this overlay technique simple. A button activates a dialog box, and the user enters text into a field automatically created for this purpose. The data are then sent to the videodisc player via the serial port on the Macintosh computer. Videodiscs can also be created to contain up to 54,000 still frames, thereby providing an enormous range of pictures that can support or expand explanations made through other media and live video action.

Digitized Speech

Educational programs, particularly those designed for students with severe learning problems, can be greatly enhanced by the use of natural-sounding digitized voice. An interface to a speech digitizer is built into the rapid prototyping tool. During a design session, voice segments can be recorded, incorporated, and evaluated. A voice segment can be moved from one part of the program to another to decide at what stage it would be most effective.

Optical Scanner

The MRPT can incorporate bit-mapped graphics into system prototypes. These can be transferred from graphics programs that operate on the Macintosh. Also, by using a scanner, it is possible to incorporate any printed picture into prototype educational software.

Benefits

The MRPT is being used to design all of the component programs in a family of CAI software to help learning-disabled children improve their reading skills. This software system includes two modules in comprehension skills, one in word analysis (analogies), one in textual analysis, and one in sequencing. All of these programs can be authored; thus, a user interface is required for both the lesson designers and the students. In the future, the system will include extensive database capabilities to keep track of lessons and student performance.

The most tangible benefits of the MRPT are the greater involvement and contributions it encourages of members of the interdisciplinary team. In the past, team members were reluctant to submit ideas that had not been thoroughly considered before meetings because they realized the trouble and cost involved in translating ideas into computer code. Spontaneous ideas, however, often develop into valuable features. The prototyping tool encourages this kind of creativity. The tool also helps new or visiting members of the team become knowledgeable about issues surrounding the program in a very short time. Familiarity with the technical design and context contributes to their critique in vital subject matter and content areas. Perhaps the most dangerous pitfall of interdisciplinary teaming is when an educator spends time, effort, and energy working not as a professional educator, but as an amateur software designer. The use of online rapid prototyping helps avoid that trap.

Not only has the level of creativity increased significantly with the use of the MRPT, but the potential to respond to that creativity has also increased. Complicated ideas have been demonstrated almost instantly by using the building blocks provided by the MRPT.

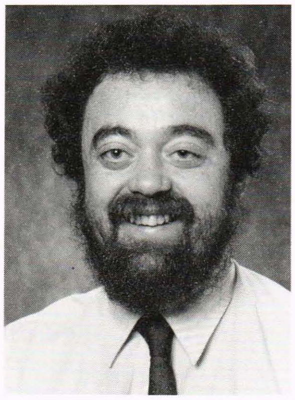
Because it has become easier to be involved in the design of the system model, the team has been able to use the advice of a broad community of experts. As a result, the latest programs of the APL-JHU team were designed with advice from specialists in a wide range of disciplines: learning disabilities, reading, mental retardation, hearing impairments, physical handicaps, emotional disturbances, and educational research. Input from many areas makes the resulting software attractive to more end users.

Finally, the benefit of the tool extends to the applicability of its techniques to the design of other kinds of software. Because MRPT has been used to design a very complete CAI software system, it can handle a wide variety of software types. For example, in the record-keeping component of the program, the tool is being used to design charts and graphs to assist in the analysis of student performance. The techniques used in this phase are more typical of the design of a business graphics package than of traditional CAI. Similarly, the design of the library system has much in common with database management system design.

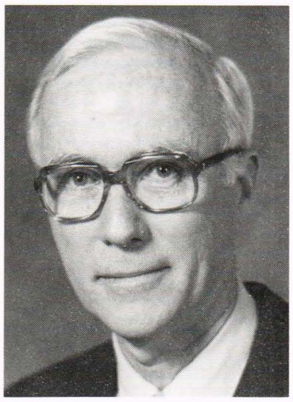
CONCLUSIONS

The benefits of the MRPT are believed to have greatly enhanced the value of the CAI that has been developed with its use. In addition, the MRPT is expected to be even more significant as the work moves into the stage of merging video and voice technologies into the authoring systems for instruction and assessment.

THE AUTHORS



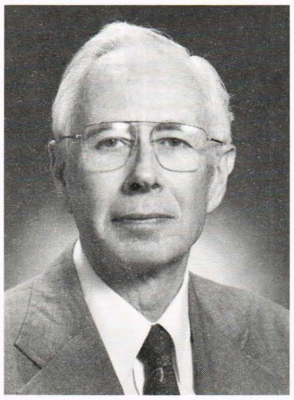
A. V. LOUIS BIGGIE was born in Glen Cove, N.Y., in 1953. He was educated at Lancaster University, England, where he received a B.A. degree in linguistics in 1977. From 1977 to 1978, Mr. Biggie was a visiting lecturer in the Philosophy Department at the University of Seville, Spain, where he taught courses in transformational grammar and sociolinguistics. In 1978, he moved to London, where he spent three years as Deputy Director of Language Link Ltd., a group of language schools. Mr. Biggie returned to the United States in 1981 and worked for The American Language Academy as Coordinator of Computer-Assisted Instruction. Since coming to APL in 1984, Mr. Biggie has worked as a programmer and designer of educational materials for handicapped, learning-disabled, and mentally retarded children.



WILLIAM E. BUCHANAN is supervisor of the External Relations Group and serves as project engineer of APL's multisensory authoring computer system development, which is funded by the U.S. Department of Education. A native West Virginian and Johns Hopkins University graduate (B.S., 1950; M.A., 1951), Mr. Buchanan joined APL in 1955 as the recruiter of professional staff. After working in several areas of personnel management, he was appointed to the external relations post in 1961. His duties have included public and community relations and audiovisual productions. He earned an M. Ed. degree in communicative disorders at Johns Hopkins in 1978, and in 1982–83 served as an M. A. Tuve Fellow in the Division of Education at Homewood.

PAUL L. HAZAN is Assistant to the Director for Advanced Computer Technology at APL. He received his B.Sc. degree in electrical engineering from The Royal College of Science and Technology in Britain in 1952, and did graduate work toward his M.S.E.E. degree at the University of Maryland (1964–66). Before joining APL in 1975, he was Technical Director of the Singer Company's Link Division in Maryland. His current technical interests include design automation, VLSI/VHSIC technologies, and computer visualization.

Mr. Hazan headed The Johns Hopkins National Search for applications of computing to aid the handicapped. He has served on the board of governors of the IEEE Computer Society for four years, as director for micro- and minicomputers, and was founding chairman of the Technical Committee on Personal Computing. Mr. Hazan is presently Chairman of the IEEE Defense R&D Committee and Director for Interdisciplinary Technology Development and Transfer. He is the author of numerous papers on computing.



ALEXANDER KOSSIAKOFF received a B.S. degree in chemistry from California Institute of Technology in 1936 and a Ph.D. from The Johns Hopkins University in 1938. He taught at The Catholic University of America (1939–42), served with the wartime Office of Scientific Research and Development, and was Deputy Director of Research at the Allegany Ballistics Laboratory, Cumberland, Md., from 1944 to 1946.

Dr. Kossiakoff joined APL in 1946 and served as head of the Bumblebee Launching Group until 1948, when he was appointed Assistant Director. He became Associate Director in 1961 and was appointed Director on 1 July 1969. In 1980 he stepped down as Director and was appointed Chief Scientist, a position he currently occupies. He is also Program Chairman of the Master of Science Program in Technical Management, G.W.C. Whiting School of Engineering.

In recognition of his work on national defense during World War II and at APL, Dr. Kossiakoff was awarded the Presidential Certificate of Merit, the Navy's Distinguished Public Service Award, and the Department of Defense Medal for Distinguished Public Service. He is a Fellow of the American Institute of Chemists and a member of the American Association for the Advancement of Science, the Cosmos Club, and the Governor's Scientific Advisory Council.