Interface, Vol. 19 (1990), pp. 155-173

0303-3902/90 /1902-0155$3.00
© Swets & Zeitlinger

A Computer-based Multi-media Tutor for Beginning Piano Students

Roger B. Dannenberg, Marta Sanchez, Annabelle Joseph, Peter Capell, Robert Joseph and Ronald Saul

ABSTRACT

The Piano Tutor provides computer-based instruction to beginning piano students. Intended as a supplement to traditional instruction, the Piano Tutor helps students by correcting mistakes before they become ingrained through practice and by teaching new material as soon as the student is ready. The Piano Tutor combines an expert system with state-of-the-art music recognition software and multimedia output devices to provide a stimulating learning environment that tailors instruction to the student's needs.

INTRODUCTION

The Piano Tutor is a computer-based system for teaching beginners how to play the piano. The goal of the system is to supplement traditional piano teaching with an intelligent computer-based teaching system. The Piano Tutor helps the student to practice correctly and make progress on new material between lessons with his or her regular teacher. This is accomplished though an automated system that is always available to the student and is capable of a high level of interactive instruction. One major part of the Piano Tutor is an expert system that embodies knowledge about teaching the piano. The expert system coordinates an array of hardware and software components that present information to the student and process student input.

The Piano Tutor is innovative along several dimensions. First, the system uses state-of-the-art music recognition software to follow student performances (Dannenberg, 1984; Bloch & Dannenberg, 1985). This allows the Piano Tutor to perform many kinds of analyses of student performances and select appropriate remedial actions.

A second innovation is that the system uses a variety of input and output devices to enhance interactivity and student interest. The devices include an acoustic or electronic piano, a videodisc player for both still and motion video images, computer graphics including music notation, digitized audio for speech playback, and real-time music synthesis. A mouse pointing device is also available for graphical input and selecting options from menus.

A third innovation is our integration of instructional design techniques (Dick & Carey, 1985), formal methods of analysis, production systems and frame-based knowledge representation, and multi-media presentations. To a large degree, this diversity within the project reflects the diversity of the authors and project team: two music professors with expertise in piano instruction, two computer scientists specializing in artificial intelligence, a computer scientist specializing in programming systems, and an instructional designer.

In the next section, we will describe related systems and research. Then, we will examine the range of options with which we were faced at the outset of the project, and we present an overview of the Piano Tutor. The following section describes the implementation of the Piano Tutor. Then, we discuss our methodology for curriculum design and analysis and describe the current status of the project. In the last sections, we present our conclusions and acknowledgments.

RELATED WORK

There have been a number of computer systems designed for teaching musical skills (Hofstetter, 1988). Notable systems include a series of interactive programs for ear training by Hofstetter called the GUIDO System. GUIDO presents students with exercises such as pitch intervals to be identified and rhythms to be notated. GUIDO does not listen to student performances; rather, the student is asked to listen to and understand performances by the computer. Student understanding (the instructional goal) is indicated by selecting from multiple choice answers. GUIDO allows instructors to tailor the system to the needs of individual students through a set of parameters that control the complexity of generated musical exercises. The system does not try to automatically adjust to student needs except by repeating drills when the student does poorly.

Thomas and Monta (Thomas & Monta, 1986) developed MacVoice, a program for harmony instruction. MacVoice checks student harmonizations of four-part chorales and lists errors. MacVoice does not deliver a sequenced curriculum to the student, but it does provide a sophisticated analysis of creative student work.

Sorisio (Sorisio, 1987) has described a more advanced system for harmony instruction that would combine an expert system for teaching with the knowledge about the rules of harmony. This work is still in progress.

In the area of psycho-motor skills, the ACLS Learning System (Currier, 1983) for teaching cardio-pulmonary resuscitation (CPR) is probably the most closely related to the Piano Tutor. Because the Piano Tutor has a much larger body of skills to teach, the Piano Tutor is more hierarchically organized. However, both systems monitor a student's performance and use videodisc to give feedback.

A comparison of the Piano Tutor with intelligent tutoring systems (ITS) is presented below after we describe some motivations for our design choices.

AVAILABLE TECHNOLOGY

When we began the project, our goal was to use technology to supplement and enhance traditional piano instruction, but we did not have a specific methodology in mind. Before describing our project, let us briefly describe the technology available to us:

- *Pianos with computer interfaces.* Both acoustic and electronic pianos are available with standard MIDI (Loy, 1985) interfaces that allow computers to play the piano as well as sense key and pedal position and velocity during a performance. This allows us to use a computer to store, manipulate, analyze, and perform piano music.
- *Expert systems.* Techniques for modeling human-level competence in specific domains are now well-established. Expert system technology implies that most knowledge required to teach piano might be encoded within a computer system and used to supplement and enhance the teaching process.
- *Graphics, Video, Voice, Synthesis.* A variety of computer output media is available at reasonable cost. This includes color graphics displays, videodisc, digitized sound, and synthesized sound. Optical storage for motion video (in the form of the videodisc) and commercial VLSI-based synthesizers for sound synthesis greatly simplify the task of interfacing to and controlling high-bandwidth sounds and images.
- *Score following.* Computer accompaniment is a relatively new development that enables a computer to follow a live performance in a score. By following the music in a score, the computer accompanist can estimate the location and tempo of the performance, and this information can be used to synchronize an accompaniment with the performance and turn pages of music on a computer display. Another result of the score following process is the identification of errors made during a performance so that the teacher knows which notes in the score caused problems for the student.

OPTIONS FOR A PIANO TUTOR

The list of available technologies leaves a great deal of freedom in deciding how to proceed. We identified a number of dimensions along which choices had to be made. We do not believe there is necessarily a best choice along each dimension, and there is still room for the exploration of different approaches and technologies for computer-assisted piano teaching. Even if we adhere to a fairly traditional model of piano instruction, we can still choose what aspects of teaching to automate with our system and the style of interaction between the teaching system and the student. In the next section we will describe the particular choices we made in designing the Piano Tutor.

The first choice is the question of how much "intelligence" to bestow upon the Piano Tutor. Many students find a simple tape recorder to be a valuable practice aid. This represents the low end of the scale. At the opposite end we have expert systems and even systems that learn their own teaching strategies.

Another choice is whether to concentrate on giving qualitative or quantitative feedback. For example, qualitative feedback might be of the form "you are slowing down in the middle of the piece" whereas quantitative feedback might present a graph of tempo versus time with more precise information. The choice is not only in how to present information, but whether the emphasis is on processing information numerically or symbolically.

A third choice determines who controls the student's activity while using the system. At one extreme, the student selects lessons or other activities and can quit at any time. At the other extreme, the system always tells the student what to do next. We note that a trend in modern computer software is to give the user full control over the sequence of processing steps, whereas in traditional piano teaching, the student receives a great deal of direction in terms of what to play and practice.

A fourth choice is whether to restrict hardware to low-cost and/or widely used products in order to make the system more widely available or to use whatever hardware is most appropriate for the task. This is really a question of whether the project is viewed as product development or research.

Finally, there is a spectrum of choices between the approaches of Computer Assisted Instruction (CAI) and Intelligent Tutoring Systems (ITS). At the risk of over-generalizing, CAI systems emphasize high-quality presentation of teaching materials and interactive dialogs with students, while ITS's emphasize the use of cognitive models of students in order to guide the instruction process.

OVERVIEW OF THE PIANO TUTOR

Figure 1 illustrates the main components of the Piano Tutor. The central component is an expert system that was developed in collaboration with piano teachers and through observation of these teachers at work. The expert system operates at two levels. At the lower level, the expert system delivers the chosen lesson, providing instruction, evaluation of student performances, and selection of remedial lessons if necessary. At the higher level, the system selects lessons according to a model of what skills the student has acquired and what skills the student needs to learn.

The Piano Tutor has capabilities for listening to performances and delivering multi-media presentations to the student. In a typical interaction with the Piano Tutor, the student is given some instructions covering new material and is asked to demonstrate mastery of the material by performing an exercise at the keyboard. Based on the student's performance, the Piano Tutor will either point out mistakes, present remedial instruction, ask the student to try again, or select a new lesson for the student.

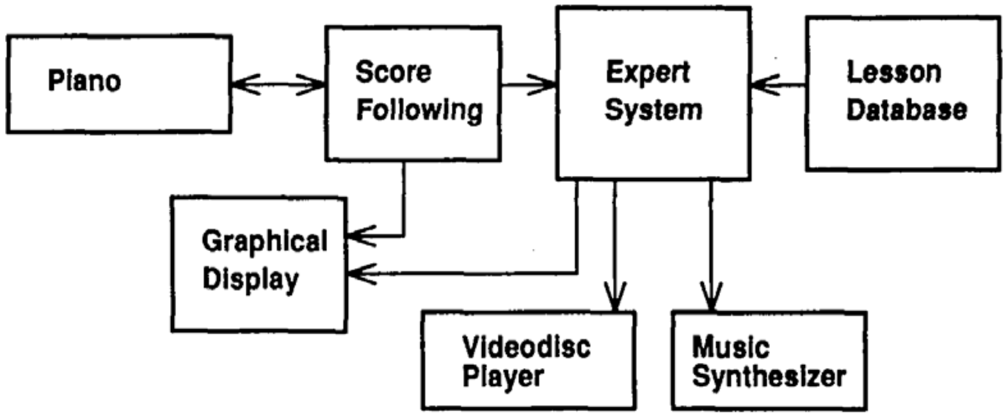


Figure 1. Block diagram of the Piano Tutor's principle components

It is also possible for the student to temporarily break out of this teaching mode in order to practice without Piano Tutor supervision. In this practice mode, the student is not evaluated, but the Piano Tutor software is still available to display music, control a metronome, or accompany the student's performance.

The output devices controlled by the Piano Tutor include video, graphics, speech, and music. A videodisc player provides up to one half hour of video with fast access to any frame and still-frame capability. There is also a computer graphics display and software for displaying text, music, and other symbols. The Piano Tutor can highlight notes on the computer display to indicate where errors occur, and speech, text, and/or video presentations can be used to describe the problem and help the student. For speech output, we use the computer system's built-in digital-to-analog converter and we store voice-quality digitized audio in the computer file-system, providing about an hour of pre-recorded speech in our system. Music output is generated by an external music synthesizer controlled by the computer. Up to 16 voices are available in a variety of traditional and electronic timbres. The computer can also control the student's piano to replay recorded student performances or to give examples of correct performances.

The primary input to the system is through a piano keyboard. We use an electronic piano with weighted keys to give a feel and sound that is reasonably close to that of an acoustic piano. Just as we have software modules for controlling various output devices, we have a special software module for keyboard performance input. The module records the performance information while following a score and turning pages of music on the computer display. Some low-level real-time analysis is also provided. For example, the system can interrupt the student if he or she starts out playing the wrong note or changes tempo abruptly. All inputs and outputs are controlled by the expert system.

Design Decisions

We can now describe the Piano Tutor in terms of the dimensions presented in the previous section. We based the system on expert system technology in order to support an "intelligent" system. Our goal in this respect is for the system to take on most of the tasks handled by a teacher during a student's first year of instruction. Because little is known about teaching psycho-motor skills (as opposed to purely cognitive skills) by computer, we are using fairly conservative techniques and avoiding topics such as automatic knowledge acquisition.

Because we are basing the expert system on current teaching practice, interaction with the student is qualitative in nature. We do not attempt to provide graphical feedback or graphs that describe errors in detail, but rather present verbal instructions when the student makes mistakes. Verbal instructions can be reinforced through the coordination of text, graphics, and video.

We decided to implement two points along the continuum of the control dimension. In the teaching mode, the Piano Tutor selects lessons and leads the student. In practice mode, the student can select music and decide when to go on to something new. Choosing a middle ground would be desirable. For example, it would be best to let the student and teacher indicate preferences for lesson material or skills to be emphasized during instruction. Since this would involve some negotiation and would complicate our lesson selection strategy, we have not developed this possibility.

Our system is based on off-the-shelf hardware, but we have not made any concessions to lower the system cost. At present, the hardware components cost about ten-thousand dollars ($10,000). Slightly more than half of this cost is for the computer, about a quarter represents the electronic piano, and the remainder is for the videodisc player, video monitor, and synthesizer. It is our belief that future personal computers or home entertainment centers will integrate video, sound, computing, and optical storage, making them available at very low cost. If these components were already available in the home, the cost of the Piano Tutor would be greatly reduced. We hope to be ready for this new technology when it arrives.

CAI, ITS, and the Piano Tutor

Two classes of computer-based instruction activity currently dominate the instructional design/technology literature: computer-assisted instruction (CAI) and intelligent tutoring systems (ITS). At first glance, it would appear that by its namesake, CAI is a superclass over all computer-based instructional systems, but by convention, CAI has come to denote a particular mode, scope, and design of instruction. The following paragraphs are devoted to drawing lines of distinction between these types of system, and to describing the Piano Tutor's position within that spectrum.

It would be nice if having named these two types of system, we could neatly fit all instructional systems into one group or the other, but this is not the case.

Most computer-based instruction (CBI, the conventional name of the superclass of all computerized instruction) can be said to have more or fewer attributes that permit us to place it more in one camp than the other. For example, systems such as Scholar (Carbonell, 1982), or GUIDON (Clancey, 1983), are definitely intelligent tutoring systems, having capabilities far beyond that which is possible using CAI techniques. Other systems having fewer capabilities fall into a grey area between the two types. Still others are most definitely not ITS systems.

The Piano Tutor seems to fall somewhere in the middle between the two positions, but it is an intelligent tutor, at least by virtue of the definition provided here. We have taken a "middle ground" in designing the system, employing techniques in artificial intelligence and expert systems, as well as methods that borrow from basic CAI.

CAI systems are normally planned according to meticulously elaborated series of anticipated student responses. These systems do not try to figure out why the student is succeeding or making errors; rather, they anticipate what the system should say, "if the student does not pick the right choice first time... second time... third..." and so on. In other words, the system may have some consideration that if the student gets the wrong answer a second and third time that the severity of the message should differ from the first error. And of course, with videodisc technology, these messages may now be quite a bit more powerful than simple text messages. We can, for example, engage a student in mixing two chemicals, and branch to an explosion sequence, depending on which chemicals he/she puts together. However, CAI does not look deeper into the nature of the student's errors than to account for them in this manner of *pre-elaboration* with fixed, predictable program branches.

Intelligent tutoring systems, on the other hand, emphasize reasoning about *why* the student took the step that he or she did in doing a task, gaining leverage on the instructional communication by responding with insightful strategies which address these conditions specifically. In terms of the scope of the curriculum and the diversity of possible responses, ITS's are far more complex to build and to understand than CAI systems. While CAI systems can certainly teach complex subjects, their internal structure is simpler than ITS's, and their instructional leverage is gained by virtue of the power of presentation, sometimes employing videodiscs and other high quality media. They rely heavily on the assumption that re-routing the student through material presented earlier is adequate remediation in all cases. Generally too, ITS's are developed for reasons extending beyond instructional delivery.

The LISP Tutor (Anderson & Reiser, 1985), for example, is an intelligent tutoring system specializing in teaching LISP programming at the college introductory level. It was created out of a long history of cognitive research by Anderson et al. (Anderson, 1983). Aspects of its "lineage" in cognitive theory which are directly implemented in the system include its *immediacy of feedback* and in its assumption that knowledge is *proceduralized.* Proceduralization is the idea that our ability to perform complex tasks is developed first by learning *declarative* knowledge, rote *descriptions* of procedures telling *how* to perform a task, followed by practice of procedural knowledge, in other words, the *doing* of the task itself. The basic assumption underlying this theory is that once having acquired the initial declarative and procedural background to perform a task, we assimilate this knowledge into higher level cognitive structures, perhaps even forgetting the specific lower-level procedures we learned earlier. So the new ability becomes a compound of its subparts, a non-discrete amalgam, and an "automatic" ability of our repertoire of skills. It is this character of proceduralization that often makes it difficult for experts to explain to a novice precisely what steps need to happen to arrive at a given solution (Anderson, 1985).

As its instructional strategy, the LISP Tutor analyzes student performances by determining how each word of input (atom) changes the state of the student's attempt at solving a problem. The Tutor has a library of known student errors, and as the student adds more to the solution, the system compares each entry of input against this library. The library is a compendium of common "bugs" known to plague students in their programming, therefore allowing the system to make incisive commentary or other remediation as is appropriate. This is a much more elaborate process than simply preprogramming things to do when the student gets one wrong, or two wrong, and so forth. The Tutor actually attempts to determine when it is that the student gets off the track and in what way and then to provide remediations appropriate for each of the "recognized" error conditions. The Tutor does well at emulating the teacher that, upon finding an error, would say: "Aha! My student is making the *x* mistake, well then I know what to do for that ... I'll show her the *y* remediation." Of course the *x* error represents some form of well-known error that teachers recognize, and the *y* remediation is some instructional strategy with which the teacher has been successful.

The Piano Tutor does things in a similar manner to the LISP Tutor to the extent that we have a library of known student performance problems and responses that correspond to those problems. Our work also bears some resemblance to GUIDON (Barr & Feigenbaum, 1982), particularly its *heuristic classification system* (HERACLES) (Wenger, 1987), in that the Piano Tutor's diagnostic routines derive abstract higher-level, taxonomized error types from lower-level data.

A specific difference in our approach is that we have designed our system to permit incremental adaptation of diagnostic rules and teaching strategies, so that we maintain flexibility with regards to either our analysis of student performance, or our responses. In other words the system has been developed with hypothesizing and adaptation in mind, with an eye to basic theories of instruction, in combination with continuous guidance from subject matter experts, the music professors. Beyond this however, the Piano Tutor makes no presumption of a basis in a complete theory of instruction; rather, our view is to create an environment that permits systematic modification of the instructional environment, along the dimensions of instructional analysis, diagnosis, and strategy.

IMPLEMENTATION OF THE PIANO TUTOR

We now turn to describe the inner workings of the Piano Tutor. Figure 2 illustrates the structure of the system. At the lower level of interaction (the box labeled "Deliver Lesson"), the Piano Tutor is engaged in teaching a small number of specific skills to the student. Music teachers and students usually think of a lesson as the entire half-hour or hour-long session with the teacher; however in the context of the Piano Tutor, skills are taught in modules that we call "lessons," which take less time to finish. Usually the student will master a short musical exercise in a lesson. At a higher level of organization (the figure as a whole), the Piano Tutor selects lessons for the student based on skills the student needs to learn. As each lesson is delivered to the student, the system's model of the student's knowledge is updated. The lesson selection process then repeats. We will start with a detailed look at the lower level of interaction with the student and then work up to the higher level of lesson selection.

The Performance/Remediation Cycle

The performance/remediation cycle is used to deliver lessons to the student. A "lesson" in the Piano Tutor consists of a presentation of new material to be mastered, performances by the student to demonstrate mastery, and remedial instruction to help the student with difficulties when appropriate.

The first step in the performance/remediation cycle is to present instructional material to the student. The material can be in the form of text, video, speech, music, or any combination. Presentations are represented in a descriptive form that can be easily edited. This allows us to change the presentation without reprogramming the Piano Tutor. Since our videodisc and lessons are being developed in parallel, the system can be tested without a final version of the videodisc.

At the end of the presentation, the student is prompted to perform some music. (Part of the presentation includes showing music on the computer display.) At this point, the Piano Tutor listens to the student performance, recording the pitch, starting time, duration, and loudness of every note. While the performance is recorded, the Piano Tutor follows along in the music and turns pages on the computer display as needed.

At the end of the performance, the Piano Tutor performs an evaluation. Figure 3 illustrates the evaluation steps taken by the Piano Tutor in order to select remediation material. In the first step, an algorithm similar to the score-following algorithm is used to perform a low-level analysis of the student performance. For every note in the score, the analysis computes whether or not a matching note was played in the performance, and if so, how accurately. The resulting "Primitive Errors" computed include pitch, duration, tempo, and loudness. Extra notes are also indicated as primitive errors.

The next step, labeled "Analysis", refines primitive errors using a set of rules structured as a discrimination net. The primitive errors are examined to decide which errors are most significant. The layers of this net can also substitute higher-level errors for lower level ones. For example, the performance analysis may make an indication (among others) that the note corresponding to note-index 17 of the score was played early by a certain percentage. (Musically determined rhythmic tolerances are used to decide whether to signal the primitive

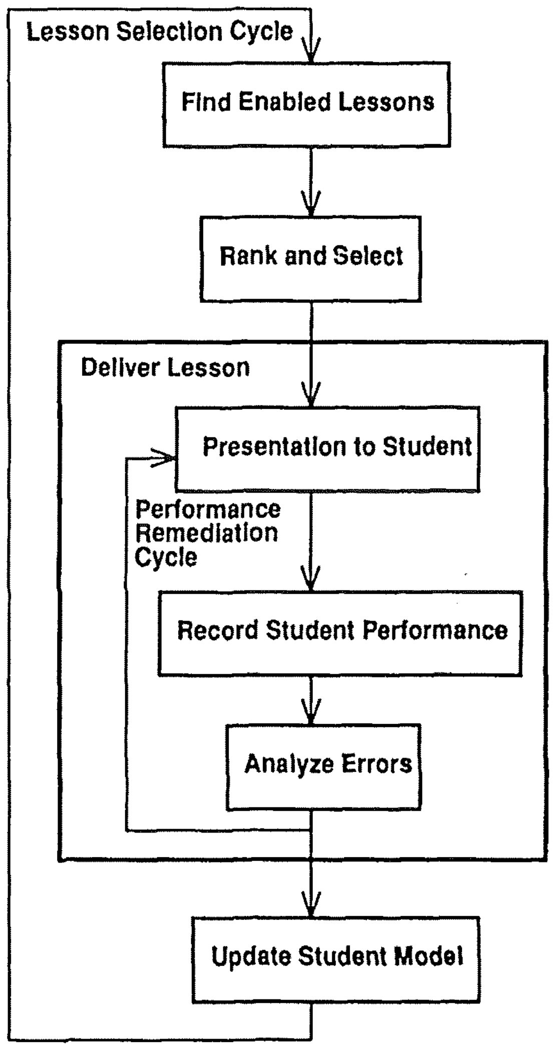


Figure 2. The basic control structure of the Piano Tutor.

error in the first place.) One discrimination rule may look for a string of such early rhythms to diagnose "rushing" the tempo.

In our example, let us say that the internal description of the score indicates that note 17 is preceded by a half rest. The rule which looks for the student playing rests too short will succeed on this error and pass it on for further refinement. A successive test will use the time values of the preceding note and rest and compute the appropriate ratios to find that (within a small tolerance) the student counted a quarter rest rather than a half. By incrementally adding such tests to the network, we hope to build a set of qualitative error descriptions that mimic the concepts used by piano teachers.

The output of this error analysis is a set of "Correctable Errors" as shown in figure 3. In the presence of multiple errors, The Piano Tutor must decide which errors to point out. We use some general heuristics and a subjective priority scale to do this. For instance, piano teachers will usually correct pitch mistakes (ex-

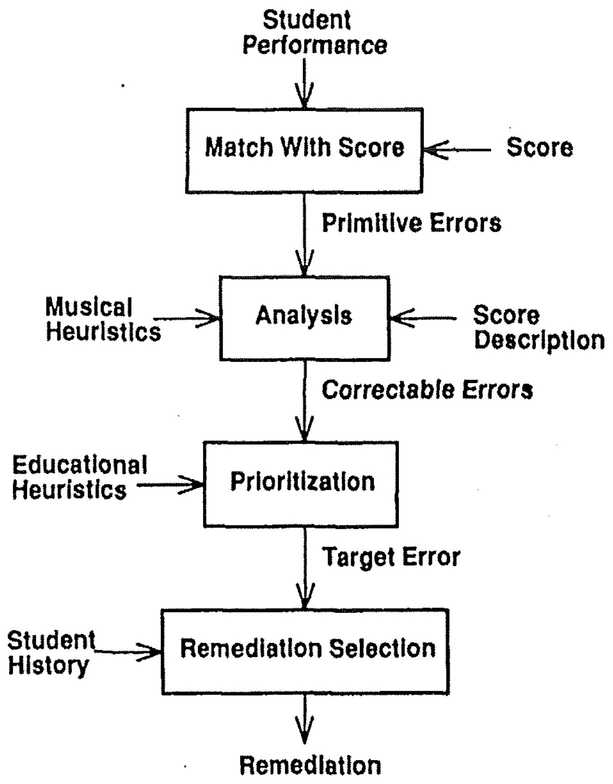


Figure 3. Evaluation of a student performance and selection of remediation.

cept for minor slips) before insisting on correct rhythm. Similarly, rhythm (usually) precedes articulation.

Once the most important error is identified, the Piano Tutor must decide what remedial actions to take to help the student correct the error. In general, the remediation will be a function of the specific error (called the "Target Error") and the student's history. The first strategy is to briefly point out a mistake to allow for self-correction. If an error appears repeatedly, a remediation is selected, and the Piano Tutor gives the student the corresponding presentation after which the performance/remediation cycle is repeated. If it seems prudent to withdraw or suspend the lesson, the relevant skill level is decremented to influence subsequent lesson selection.

Eventually, the student either passes the lesson or the Piano Tutor decides the material is too difficult. In either case, the lesson is ended.

Lesson Selection Cycle

We now turn to the topic of how lessons are selected. Lesson selection is closely tied to and interacts with the student model, which maintains knowledge about the student's current abilities. The student model consists of a large set of skills, each of which has a value representing the degree of mastery of that skill.

Lessons have *prerequisites* that a student must have in order to take the lesson. Lessons also have *objectives* that the lesson attempts to teach. Both *prerequisites* and *objectives* are skills and correspond directly to components of the student model.

To select a lesson, the Piano Tutor first forms a list of all enabled lessons. A lesson is enabled if the student has met the prerequisites for the lesson and it has at least one objective the student has not yet met. Once the set of enabled lessons is determined, the Piano Tutor can use a set of rules to rank the candidate lessons and make a selection. The rules can encode knowledge such as: "if the previous lesson taught a rhythm skill and the student passed, then favor another lesson that teaches rhythm skills," or "try to teach lesson A before lesson B."

After a lesson is selected, the performance/remediation cycle described in the previous section is started. The lesson is stored with associated presentations, remediation rules, and other lesson-specific information required by the performance/remediation cycle.

At the end of the lesson, the student model is updated according to the objectives specified by the lesson. In cases where the student does not pass the lesson, it is possible for the remediation rules to specify a special set of updates. For example, if the student has apparently forgotten some skills, then the value of these skills in the student model can be decremented. This will have the effect of disabling advanced lessons with these skills as prerequisites, leaving only simpler lessons for the Piano Tutor to choose from.

CURRICULUM ANALYSIS

One development that we feel is unprecedented is our use of formal systems analysis techniques in determining dependencies among skills, prerequisites and instructional objectives. As Briggs states, the purpose of instructional design is to ensure "congruence among objectives, teaching strategies and performance evaluation." (Briggs, 1977). Thus far, in the field of instructional design, the methods and strategies for carrying out this purpose are mostly illusory because the links between skills are not always clear. Often, the designer is left to use informal strategies and "best guesses" to develop what appears to be a reasonable curricular progression. The Piano Tutor *has not* solved this problem, but by virtue of the precision required to properly analyze and respond to student performances on a computer, the Tutor has become a framework within which to create tools to understand and refine the dependencies among skills. Thus we are making progress in ensuring "instructional integrity" in a much stricter and precise sense than has ever before been possible. The following paragraphs provide the details of this effort.

In a large network of interconnected lessons and skills, it is difficult to make sure the network expresses the intended relationships and constraints. One of the benefits of our representation of lessons and the student model is that it is possible to use formal methods to analyze the curriculum design.

For example, one might want a student to understand the concept of half note before teaching about half rests. To express this concept, the lessons that teach half notes must have *half-note-skill* in their list of objectives, and the lessons that teach half rests must have *half-note-skill* in their prerequisites. This guarantees that the student will have demonstrated the skill *half-note-skill* before proceeding to take a lesson on half rests.

To assist with the specification of skills and lessons, we have implemented analysis software that derives information from the lessons and skills. The analysis algorithms used are either standard graph algorithms (Aho & Ullman, 1974) or related to algorithms used by optimizing compilers for code analysis (Wulf et al., 1975). The intent of the analysis is to help the designer discover inconsistencies in the specification of skills, lessons and their interdependencies. Once exposed, these specification errors can be corrected. The following information is computed:

- Implication of skills. Does the possession of a skill imply that the student has other skills as well? The instructional designer can study these derived implications to see if they seem reasonable. For example, it is logical that understanding eighth notes would imply an understanding of quarter notes, since quarter notes would always be taught first. On the other hand, if an understanding of eighth notes implies understanding major triads then there is probably a mistake in the specification of some lesson.
- Lessons that use a skill. This is the inverse of the prerequisites of a lesson,

allowing the lesson designer to find all lessons that use a given skill as a prerequisite.
- Lessons that teach a skill. This is the inverse of the objectives of a lesson, allowing the lesson designer to find all lessons that teach a given skill. Skills that can only be taught by one lesson indicate a lack of alternative teaching strategies.
- Irrelevant lessons. A lesson that does not teach a skill used by another lesson is irrelevant in that it does not help the student progress towards the ultimate goal (represented by a distinguished lesson). Usually, irrelevant lessons indicate that other lessons are missing or that an objective was omitted or specified incorrectly.
- Irrelevant skills. Skills that are not prerequisites of some lesson are useless in the same sense that lessons can be useless. Again, irrelevant skills point out missing lessons or incorrect lesson specifications.
- Unobtainable skills. A skill that is not an objective of any lesson cannot be taught. An unobtainable skill indicates an incomplete or incorrect specification of lessons.
- Equivalent skills. If two skills imply one another, then they are equivalent. The existence of equivalent skills usually means that there there are two conceptually distinct skills, but the curriculum always treats them identically. The instructional designer should either coalesce the skills into a single skill, or new lessons should be added, for example, to teach one skill but not the other.
- Unlearnable lessons and skills. Suppose a student passed lessons until there were no more lessons that were enabled but not taken. Any untaken lessons or unlearned skills are said to be unlearnable, indicating an error in the curriculum design.
- An example curriculum. The analysis software finds an approximately "shortest path" through the lessons to the designated ultimate lesson. This helps to identify unintended "shortcuts" through the lesson structure and helps to verify the overall curriculum design.

The amount of information that can be derived from the lesson specifications is considerable. It is important that the analysis can be performed without actually implementing the lesson content and without testing the system on real students. In practice, we have been able to develop much of the Piano Tutor software and the curriculum independently and concurrently. Although extensive testing will still be required, the lesson analysis helps us consider lesson sequences that will rarely arise in practice. We expect this to decrease the number of problems encountered by students.

THE PIANO TUTOR AT PRESENT

The Piano Tutor is presently operational with about 25 lessons in place. We have multimedia input and output as described (except for speech) including a videodisc produced expressly for the Piano Tutor. We have a complete curriculum design for about one year of piano instruction, including about 50 lessons and 25 skills. We are currently modifying the Piano Tutor's lesson selection and performance analysis strategies to conform to the descriptions presented above, after which we will implement more lessons to complete the system. We expect to begin testing the system with students in the summer of 1990.

CONCLUSIONS

The Piano Tutor has been through many design revisions in response to our experience with prototypes. The current design strikes a balance between implementation complexity and the ability to deliver intelligent instruction.

The structure of the Piano Tutor seems to be fairly general and we can envision using the system for other instructional tasks. The key elements of the Piano Tutor are:

- Skills are used to model student knowledge. The model is simple to update and interpret because it consists of a number of independent elements with numerical values. The model can be extended easily by introducing new skills.
- Lessons are used to modularize instructional tasks. This has several benefits. The system performance is enhanced because lessons restrict the amount of knowledge that must be managed at any point in time. Analysis is simplified because at any point in time, the student's task is well defined, making the likely trouble spots few in number. Finally, lessons make the system modular; it is fairly simple to add new lessons or modify lessons without affecting the rest of the system.
- The performance/remediation cycle within lessons is an effective way to manage an interactive dialog with the student. Once the student is given instructions, the system does not need to do any sophisticated decision-making while the performance is in progress. This frees the processor to manage music, graphics, and video.

One of the key constraints on the design of the Piano Tutor has been real-time performance. It is essential that the system be able to interact rapidly with the student. Typically, expert systems are not interactive because they are too slow. The Piano Tutor largely solves the performance problem by using a hierarchical structure. The higher level is concerned with lesson selection, and because lessons are selected only on the basis of skills they require and teach, the lesson selection process does not need to consider the content of lessons in any detail. Once a lesson is selected, attention is focused on a restricted task, again minimizing computation. During presentations and performances, nearly all computing resources are dedicated to managing input and output, minimizing the computational load even further at the times when performance is most critical.

Even during a student performance, our software is able to follow the performance, turn pages of music, and play an accompaniment. In addition, some low-level analysis of the performance is performed in real time. If the student starts on the wrong note, makes a large change in tempo, pauses, or misses too many notes, the Piano Tutor can interrupt in the middle of a performance. The conditions under which the system will interrupt the student are determined in advance based on the student model and the current lesson. This provides effective real-time remediation while minimizing the amount of real-time decision-making required.

The Piano Tutor is programmed in the Common Lisp (Steele, 1984) and C (Kernighan & Richie, 1978) programming languages. Lisp is used for lesson selection and performance evaluation, and C is used for nearly all real-time interaction with the student. A knowledge representation system called Framekit (Nyberg, 1988) is implemented in Lisp and is used to represent lessons, rules, and presentations. We have found Framekit, Lisp, and their associated interactive programming environments to be extremely valuable for prototyping and knowledge representation.

For all of us, the Piano Tutor has been our first experience working within an interdisciplinary group on a major project. Although we have spent an enormous amount of time learning each other's language and establishing a common ground on which to coordinate our efforts, the results have been rewarding. It is hard to imagine how the Piano Tutor could have been developed without contributions from all disciplines.

ACKNOWLEDGMENTS

The Piano Tutor has had many contributors, including John Maloney, Hal Mukaino, David Pugh, and Michael Miller. We would especially like to thank the Center for Art and Technology and Phil Miller for their support in the critical beginning phase. The Piano Tutor project is currently supported by the Markle Foundation.

REFERENCES

Aho, Hopcroft, & Ullman (1974). *The design and analysis of computer algorithms.* Addison Wesley.

Anderson, J.R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.

Anderson, J. (1985). *The implications of cognitive science.* NY: Houghton-Mifflin.

Anderson, J.R., & Reiser, B.J. (1985). The LISP tutor. *BYTE,* 159-175.

Barr, A., & Feigenbaum, E. A. (1982). Guidon. *The Handbook of Artificial Intelligence.* William Kaufman, Inc.

Bloch, J.J., & Dannenberg, R.B. (1985). Real-time computer accompaniment of keyboard performances. In Truax, B. (Ed.), *Proceedings of the International Computer Music Conference,* (p. 279-290). Computer Music Association.

Briggs, L.J. (Ed.) (1977). *Instructional design: principles and applications.* Englewood Cliffs, NJ: Educational Technology Publications.

Carbonell, J.R. (1982). Scholar. *Artificial intelligence and tutoring systems.* William Kaufmann, Inc.

Currier, R.L. (1983). Interactive videodisc learning systems. *High technology,* 3(11), 51-59.

Dannenberg, R.B. (1984). An on-line algorhithm for real-time accompaniment. *Proceedings of the International Computer Music Conference,* (p.193-198). San Francisco: Computer Music Association.

Dick, W., & Carey, L. (1985). *The systematic design of instruction.* Glenview, IL: Scott, Foresman and Company.

Clancey, W. (1983). Guidon. *Journal of Computer-Based Instruction,* 10(1 & 2), 8-15.

Hofstetter, F.T. (1988). *Computer literacy for musicians.* Englewood Cliffs, NJ: Prentice Hall.

Kernighan, B.M., & Richie, D.M. (1978). *The C programming language.* Englewood Cliffs, NJ: Prentice-Hall.

Loy, G. (1985). Musicians make a standard: The MIDI phenomenon. *Computer Music Journal,* 9(4), 8-26.

Nyberg, E.H. (1988). *The FRAMEKIT user's guide version 2.0.* Unpublished manuscript, Computer Science Department, Carnegie Mellon University.

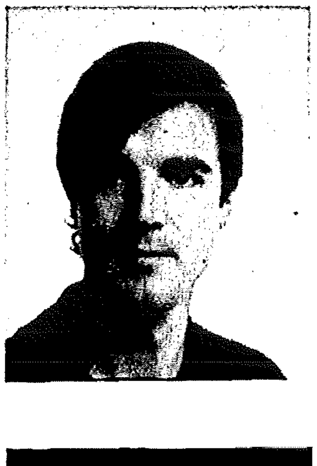
Sorisio, L. (1987). Design of an intelligent tutoring system in harmony. *Proceedings of the 1987 International Computer Music Conference,* (p. 356-363). Computer Music Association.

Steele, G.L. Jr. (1984). *Common lisp.* Digital Press.

Thomas, M.T., & Monta, P. (1986). MacVoice 1.0 user's manual. Kinko's Academic Courseware Exchange.

Wenger, E. (1987). *Artificial intelligence and tutoring systems.* Morgan Kaufmann Publishers, Inc.

Wulf, W., et al. (1975). *The design of an optimizing compiler.* New York: American Elsevier.



Dr. Roger B. Dannenberg is a Research Computer Scientist on the faculty of the Carnegie Mellon University School of Computer Science, where he received a Ph.D. in 1982. In addition to the Piano Tutor, his current work includes research on computer accompaniment of live musicians, and the design and implementation of Arctic, a very high-level functional language for real-time control.

Dr. Marta Sanchez is Professor of Music and former Head of the Music Department at Carnegie Mellon University where she teaches Eurhythmics and Piano Improvisation and directs the Dalcroze Training Center. As an educator and musicologist, she lectures and consults in the United States, Latin America, Europe, Australia and New Zealand. She participates in national and international conferences and is actively involved in musicological research and educational projects. Dr. Sanchez has written numerous articles in various languages and her recently published book, *The Spanish Villancico of the Eighteenth Century,* has been reviewed as an authoritative and innovative contribution to the field of Spanish music. As a pianist she specializes in four-hand and two piano American music as well as in music by Latin-American composers and she also maintains a private studio.

Peter Capell completed his Bachelor of Arts Degree at the University of Pittsburgh in 1978. His major was self-designed, focussing on Psychology and Sociology, with a minor in writing. In 1982, he completed his Master of Arts Degree with his concentration in instructional systems design. His thesis was an examination of cognitive styles in relation to strategic thinking ability. In 1988, Capell earned his Ph.D. at the University of Pittsburgh, School of Education. Research for his doctorate was conducted at Carnegie Mellon University, where he is currently employed as a Research Associate in the Center for Art and Technology. The focus of his research is instructional design as it relates to intelligent tutoring systems. His dissertation is a study of the characteristics of instructional design in intelligent tutoring systems. Capell's primary responsibilities for the Piano Tutor project include instructional and videodisc design.

Robert L. Joseph is currently a Ph.D. student at Carnegie Mellon University. In the coop program at MIT he received a dual Bachelor and Master's Degree in 1984 from the EE/CS department. During his coop work at AT&T Bell Laboratory he completed his Master's thesis "An Expert System for Completing Partially Routed Printed Circuit Boards". Along with his current research with Piano Tutor, Robert is also involved as a student of Jaime Carbonell in graphical knowledge acquisition for expert systems.

Ronald Saul, Research Programmer for the Piano Tutor Project, received degrees in music from Northwestern University and the Ohio State University. During the past eight years he has studied and worked in Computer Science and Expert System development.

Dr. Annabelle Joseph, Associate Professor of Music at Carnegie Mellon University, teaches Ear-training, Dalcroze Pedagogy and Piano Improvisation in the Music Department and continues research on the Piano Tutor in the Center for Art and Technology. She is former Coordinator of Eurhythmics at Duquense University and past president of the Dalcroze Society of America. Dr. Joseph consults, lectures and gives workshops in Dalcroze Eurhythmics throughout the United Stated and in England. A forthcoming article, "The Importance of Rhythm in Music Education" will appear in the new MENC edition of *The Eclectic Curriculum in American Music Education.* As a pianist, she specializes in four-hand and two piano American music and maintains a private piano studio for students of all ages.