MITSUBISHI ELECTRIC RESEARCH LABORATORIES
CAMBRIDGE RESEARCH CENTER

The Audio Interactive Tutor

Richard C. Waters

MERL-TR-94-04 April, 1994

Abstract

The Audio Interactive Tutor (Tait) is an interactive audio/oral computer-aided study device. It is most easily understood in comparison to the familiar notion of self-study audio tapes, which are non-interactive audio study devices. A self-study tape presents information and typically solicits responses from the user; however, it continues with a set pattern of instruction no matter what the user does, since it has no means of even detecting whether the user makes a response. Tait presents the same kind of output, but listens to the responses made by the user and alters the course of study depending on whether the responses are correct or incorrect. Tait supports an efficient approach to study that relies on sparse repetition carefully spread over time, rather than copious repetition.

Contents

1 The Basic Idea

The Audio Interactive Tutor (Tait) is a computer aided instruction system whose output is sound (i.e., speech, but potentially also music, bird calls, etc.) and whose input is sound (i.e., spoken responses from the user). Tait's output consists of explanations and examples along with commands and questions requiring responses from the user. Tait analyzes the user's spoken responses and constructs an evolving model of what the user knows. Tait uses the model to direct the course of further study.

To be of use, Tait must be supplied with course material relevant to some particular domain of study. In the following, learning to converse in a foreign language is used as a continuing example of a such a domain. This is a good example domain because conversing in a language is a quintessentially audio activity. However, it is important to realize that Tait is in no way limited to foreign language instruction. It should be useful for studying anything that can be presented aurally—e.g., music, history, and the law. The only situations where Tait is obviously not appropriate are ones where the domain of study requires visual material—e.g., studying painting or the interpretation of X-rays.

An additional reason why foreign language instruction is a good example domain for Tait is that the well developed concept of foreign language self-study tapes is a useful point of comparison with Tait. At a detailed level, there are many differences between the various self-study tapes available; however, they all take the basic form of scores of hours of interactions of the following type. (1) An instructor says something (either in the foreign language or in the user's native language) that calls for a response (usually in the foreign language) from the user. (2) There is a period of silence for the user to provide a spoken response. (3) A native speaker illustrates a correct response. (4) There is a period of silence in which the user can assess the correctness of his own response, and repeat the correct response.

Self-study tapes have the advantage that they are very cheap (only a few hundred dollars for a set) and can be used in a wide variety of situations (e.g., even while driving). However, they are limited in their effectiveness by inflexibility and a lack of feedback. The fundamental problem is that self-study tapes have no way of assessing whether the user has made a correct response, or for that matter, any response at all. As a result, they cannot provide feedback about correctness to the user. They also have no choice but to follow a single fixed pattern of instruction.

The lack of feedback is troublesome in many ways. First, it can be quite hard for users to decide whether or not they are correct. For instance, it can be hard for a user's untrained ear to hear subtle differences. In addition, there is often more than one correct response. Therefore, the mere fact that a user's response was different from the response on the tape does not necessarily mean that the user was wrong.

The lack of flexibility is an even greater difficulty. Self-study tapes must contain significant repetition to be effective. (In fact, users typically have to listen to a set of tapes several times for good learning to occur.) Unfortunately, since the repetition is fixed once and for all, it cannot be varied based on what the user is and is not succeeding in learning. Inevitably, much of the repetition involves things that were easy for the user to learn and is therefore a waste of time. On the other side of the coin, there are inevitably things that the user finds hard to learn that fail to get repeated often enough.

Tait is similar in conception to self-study tapes, with the addition of simple speech recog-

nition and a study controller. This results in a device that fits into the same basic niche, but can flexibly respond to what the user does and therefore is significantly more effective.

The basic pattern of interaction with Tait is the same as with self-study tapes—i.e., scores of hours of trigger/response pairs. However, when the user makes a response, Tait uses speech recognition to determine whether the user's response is correct. When the user's response is clearly correct, Tait provides positive feedback. When the user's response is clearly incorrect, Tait provides negative feedback followed by an illustration of a correct response. If Tait can recognize some particular common mistake having been made, it points this out. If it is not clear whether the user has made a correct response or not, Tait merely illustrates a correct response and leaves it at that. Most importantly, the next trigger/response pair to present is chosen based on the past history of what the user has gotten right and wrong. As with self-study tapes, all of Tait's output is prerecorded—speech generation is not required.

Another point of comparison with Tait is other computer-aided instruction (CAI) systems (see for example [5]). Tait clearly fits the traditional outline of such systems. In particular, like essentially all CAI systems, Tait creates a model of what the user knows and has a controller that varies its behavior based on the evolving model. However, below this surface description, Tait is as different from other CAI systems as they are from each other. All in all, there are three key features that together make Tait quite different from other current CAI systems.

To start with, Tait supports spoken input from the user, rather than typed input, menu selection, or button pushing. This opens up a new channel of communication that has several important advantages. First, spoken input is convenient and fast. Everyone knows how to talk and can do so at several hundred words a minute. (Many people cannot succeed in typing more than a few dozen words a minute; and the best typist can barely break one hundred words a minute.) Second, because spoken input allows users to give long answers quickly, it makes it easier to avoid the kind of multiple choice interactions that are forced on a system by menu selection, button pushing, and simple typed input interfaces. Third, spoken input is ideally suited to study domains such as learning to converse in a foreign language that are fundamentally oral in nature.

An equally obvious difference between Tait and other CAI systems is that Tait does not have any visual output. On one hand, this can be viewed as a defect of Tait, which limits its applicability. (An obvious direction for extending Tait is to add visual displays of information.) However, on the other hand, the fact that all of the interaction with Tait is via sound is an advantage of Tait, because it allows eyes-free hands-free operation, e.g., in a car.

Finally, as is the case with essentially any CAI system, the heart of what makes Tait interesting is its exact approach to creating a user model and controlling the course of study. Tait is based on a theory of learning and retention that holds that you do not have to practice something a lot of times to learn it, as long as the practice you do is properly spread out in time. The idea is that highly efficient study can be obtained by practicing an item just a few times at first; and then practicing it a few more times after longer and longer intervals in which other items are taught. The goal is to practice the item again just before the time when you would otherwise forget it—practicing earlier is wasteful and boring, practicing later requires reteaching. The key difficulty is determining the proper practice schedule for a given item. In Tait, this is done by observing a user's responses and estimated how well he knows the item in question.

It is interesting to note that the foreign language self-study tapes created by Pimsleur (see

for example [8]) are based on the same basic theory of learning and retention. However, due to the inflexible nature of self-study tapes, the practice schedule used for each item has to be fixed in advance for all users. Therefore, while good on average, the schedule chosen for an item may not be good for a particular user. Tait goes a long way toward fixing this problem by tailoring the practice schedules to each individual user.

The next section (Section 2) presents Tait's architecture and its approach to study in detail. This is followed in Section 3 by a discussion of a proof of concept implementation of Tait and its application to foreign language instruction. Section 4, concludes by discussing various ways that a full prototype of Tait could be implemented.

2 Underlying Concepts

The Audio Interactive Tutor (Tait) is based on two quite separate concepts: a particular system architecture and a particular approach to study. The two concepts are separate, because while they are well suited to each other, you could use many different study models in conjunction with the architecture and many different architectures in conjunction with the study model.

2.1 System Architecture

The architecture of Tait is shown in Figure 1. The system consists of two modules for input and output (in the upper left of the figure), two data stores (in the lower left), and a central controller.

The *audio output* module plays segments of prerecorded audio such as verbal instructions, verbal questions, and music. It need not be more complicated than simple D/A hardware connected to speakers or headphones.

The *speech recognizer* compares utterances by the user with statistical models to determine what the user has said. Speech recognizers are traditionally categorized in terms of five features: (1) whether they support continuous speech, or require unnatural pauses between words, (2) whether they are speaker independent, or have to be trained to the way a particular person speaks, (3) how accurate they are in determining what a person has said, (4) how large a vocabulary they can handle (10 words, 1000 words, 100,000 words), (5) whether they operate in real time (i.e., output a recognition only a second or so after a person has said something) or much slower. It would be nice to have a continuous, speaker independent, very accurate, very large vocabulary, real time speech recognizer. Unfortunately, this is not possible at the current state of the art.

Fortunately, while Tait's speech recognizer must be continuous, speaker independent, and operate in real time, it only needs to handle a tiny vocabulary and does not need to be completely accurate. This reduced level of capability is well within the current state of the art.

The vocabulary to be recognized is effectively small, because at any one moment, Tait need only determine whether or not the user has said something; and whether this utterance is, or is not, one of at most a dozen things. Tait can tolerate errors in recognition, because the study

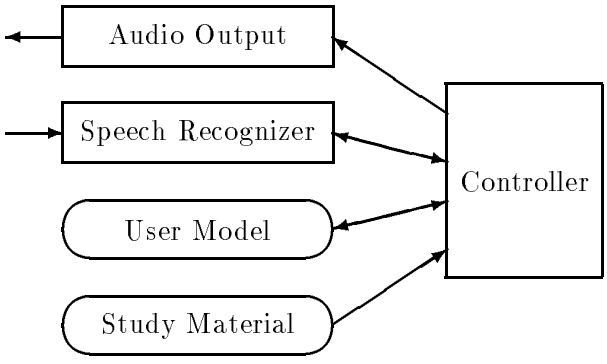

Figure 1: Architecture of Tait.

approach it uses effectively averages groups of observations together. As a result, the speech recognizer only needs to be right most of the time (i.e., 80–90%), rather than all of the time. Both of these simplifications are a result of Tait's study approach (see Section 2.2).

Even in the simplified form required by Tait, the speech recognizer is by far the most complex and computation intensive part of the system. It consists of a microphone connected to A/D hardware and a considerable amount of numerical calculation.

The *user model* records information about the user. The most important part of this information is a profile that records what the user has learned and how well. This information is frequently modified by Tait, but does not take up much space.

The *study material* is a large amount of data that specifies what is to be studied. Conveniently, this data never needs to be modified, so it can be stored on a dense read-only medium such as CD-ROM.

The study material consists of three kinds of information: prerecorded chunks of sound output, statistical descriptions of possible user responses, and study control information. The study control information is the most interesting, but takes up very little space. The other information is simple but voluminous.

The amount of memory required to store prerecorded sound depends on the quality desired. As an upper bound, an hour of CD-quality stereo sound takes up 600 megabytes of space (a standard CD). However, mono sound that is acceptable for most purposes can be stored in 1/10 the space.

The amount of memory required for the statistical descriptions required for speech recognition is harder to assess, but should not be significantly larger than what is required to represent the sound itself.

Together the estimates above suggest that Tait's study material requires something in the neighborhood of 100-200 megabytes per hour. This indicates that the material probably must be stored on something like a CD-ROM rather than a floppy disc, but that a single 600 megabyte CD-ROM should hold an ample amount of study material. (Note that due to the repetition and pauses required, a typical 10–15 hour set of self-study tapes probably only holds an hour or two of material.)

The *controller* is the central module of Tait. Based on the user model, it decides what pieces of the study material to present when. To present a piece of material, it transmits the associated prerecorded sound to the audio output module and statistical descriptions of the expected responses to the speech recognizer. When the user responds (or fails to respond), the controller updates the user model based on what the speech recognizer determines the user has said.

The controller is not specialized to any one domain of study. All the information about any particular domain is recorded in the study material. This material can be viewed as a study program that runs on the controller.

2.2 Study Approach

Tait's study approach is oriented toward situations where the user needs to learn a large number of relatively simple things. The interaction between Tait and the user is a series of stimulus/response pairs, where Tait explains something and/or poses a question and the user follows with a response.

Tait's study approach is based on a model of memory and retention that emphasizes the

ability of people to learn very rapidly if information is presented in a way that is spread out appropriately over time. In particular, Tait's design assumes that there is no benefit (and perhaps harm) in practicing something 30 times rather than 3 times when you first encounter it. What is valuable is to practice it a few times spread out in time to help cement it in memory. Then, once you know something, you only need to practice it occasionally to keep it current in your mind.

Items. A set of study material is organized around a set of primitive items to be learned such as "the Spanish word for left is izquierda" or "the capital of Maine is Augusta". The only explicit piece of information about an item is an identifying name, which is referred to by the other kinds of information described below.

Knowledge levels. The bulk of the user model consists of two pieces of information about each item in the material the user is studying. The first piece of information is the date and time the item was last practiced by the user (if ever). The second is an estimate, called the knowledge level, of how well the user knows the item.

Practice intervals. Each knowledge level is associated with a practice interval that specifies how often an item should be practiced. For instance, the knowledge level 1 might be associated with a practice interval of 24 hours.

Tait uses practice intervals when deciding what to ask the user to practice when. For instance, suppose that it has been one week since the user last practiced some particular item. If the user's estimated knowledge level for that item is associated with a practice interval of one week or less, Tait will select study material that practices the item. Otherwise, Tait will avoid material that practices the item.

Estimating knowledge levels. When a user first begins to use a set of study material, the knowledge level estimate for each item is set to a special null value that indicates that the item needs to be taught by Tait. As the user demonstrates knowledge of the item, its knowledge estimate takes on a range of integer values indicating levels of greater familiarity. An item is said to be *current* if its estimated knowledge level is positive.

Whenever the user responds to a question posed by Tait, the response demonstrates knowledge of some items, and if it is an incorrect response, lack of knowledge of others.

If a response indicates knowledge of an item, the knowledge level estimate is increased as follows. If the knowledge level was previously set at the special null value it is set to is lowest integer value. Otherwise, if the current time is more than three quarters of the way through the practice interval associated with the item's previous level, then the estimated knowledge level is incremented (unless it is already at the maximum level).

If a response indicates lack of knowledge of an item, the knowledge level estimate is reduced as follows. If it is greater than 1, it is reduced to 1. Otherwise, if it is greater than the lowest level, it is reduced to this level. Otherwise, it is set to the special null value indicating that the item should be taught again.

The fact that changes in estimated knowledge levels based on a user's responses are gradual is one reason why Tait is relatively insensitive to errors in its speech recognizer. As long as the speech recognizer is correct much more often than it is incorrect, the knowledge levels will converge on appropriate values. The only error that could occur that the user could reasonably notice is if the speech recognizer erroneously determines that a correct response for a current item is incorrect twice in a row for the same item. This will push the knowledge level down to

Level	Practice Interval
-2	3 minutes
-1	10 minutes
0	16 hours
1	20 hours
2	1 week
3	2 weeks
4	1 month

Figure 2: An example of knowledge levels and practice intervals.

the minimal value, which will cause the item to be practiced a lot even though the user knows it well.

An example. The knowledge levels and practice intervals to use in conjunction with a particular set of study material are specified as part of the study material. As an example, Figure 2 shows the knowledge levels and practice intervals used by the current implementation of Tait when studying foreign languages.

The values in Figure 2 were chosen under the assumption that the user would study once or twice a day for approximately 30 minutes at each session. Correct response to the initial teaching of an item changes the item's practice level to -2. If the user continues to practice the item correctly, then the knowledge level raises rapidly as follows. The level will reach 0 during the same lesson in which the item was first taught. The item will reach level 1 (and therefore currency) after being practiced successfully the next day. If the user does not recall the item the next day, intensive practice will resume.

Features. It may be the case that the material to be studied needs to be tailored to the characteristics of the user. These characteristics may be preferences or inherent properties of the user.

The characteristics of a user are represented in the user model as *features*. Each feature has an identifying name and can take on one of a few specific values. What the relevant features and values are for a given set of study material is specified as part of the material.

As an example of the use of features, consider that in many languages, gender agreement means that male and female speakers have to say somewhat different things. As a result, the foreign language study material discussed in Section 3.3 contains a feature 'Gender', with the potential values 'Male' and 'Female', which is used to specialize the material for different users.

Pairs. Tait's interaction with the user is oriented around stimulus/response pairs. Each pair contains the following information.

Name - a unique identifying name used when referring to it from other pairs.

Stand-alone - the value True if the pair can be used in isolation, as opposed to as part of a unit (see below).

Features - a list of zero or more features that must be true of the user in order for this pair to be applicable.

Requires - names of the items that must be current for the user, if this pair is to be used. (It is assumed that it is confusing to be teaching or practicing too many

things at once. Rather, each pair should teach or practice only a couple of things, but can do so utilizing arbitrarily many other things the user knows well.)

Teaches - names of the items taught, if any. (Many pairs are just for practice and don't teach anything.)

Practices - names of the items (other than any required items) that will be practiced in correct responses to this pair.

Stimulus - a segment of prerecorded sound to present to the user. (I.e., a description, or question.)

Secondary-stimulus - a segment of prerecorded sound that will be presented to the user if the primary stimulus evokes no response.

Continuation - an optional action to perform immediately after the pair has been presented to the user and the user has responded. (This is either one of a few special system actions like stopping the lesson, or the name of another pair to present to the user. The primary use of continuations is to chain sets of pairs together into groups.)

Responses - a list of responses (see below) expected from the user. (Typically only a few and never more than 10 or so.) It is possible for there to be no responses. In that case, Tait assumes that the stimulus merely imparts information to the user and no response is expected.

The fact that only a small number of very specific responses is allowed is important for two reasons. First, it makes speech recognition straightforward in the current state of the art. Second, it allows Tait to determine whether a given response is correct without understanding what the response means, by merely comparing it with the allowed responses.

It is important to note that the way Tait handles responses places significant limits on the material Tait can be effectively used to study. One of the goals of Tait is to require users to come up with responses entirely on their own, rather than picking them from multiple choice lists. However, to keep its internal operation practical, Tait operates on the inside more or less exactly the same as if the users responses were selected from multiple choice lists. This can only work for stimuli where only a small number of very specific correct responses are possible. Some material can be studied with questions of that character, and some cannot. For instance, Tait can easily work with questions like "What is the Spanish word for left?" and "What is the capital of Maine?", but it cannot work with questions like "Why did Maine separate from Massachusetts?".

Each response in a stimulus/response pair contains several pieces of information about the response:

Validity - an indicator specifying whether the response is valid, invalid, or neutral. (Neutral responses are neither correct nor incorrect. They are used only in special situations—e.g., when determining features of the user, see below.)

Phrase - a statistical model of the expected response that can be used by the speech recognizer. If there are any responses for a pair, exactly one is given a special null model, which indicates that the response is a catchall that is selected when none of the other responses for a pair can be recognized.

Correct items - names of items the response indicates correct knowledge of. (The practices field of a pair is typically the intersection of the correct items fields of its

valid responses.)

Incorrect items - names of items the response indicates incorrect knowledge of.

Action - an optional action to perform if the user makes this response. (This is either one of a few special system actions like terminating the session, or the name of another pair to present to the user—e.g., to reinforce the response or make a correction.)

For Tait to be tolerant of errors in its speech recognizer, it is important that the actions associated with responses be chosen so that they are not jarring in situations where a response is incorrectly recognized. In particular, if the action for an invalid response merely focuses on demonstrating a correct response, it is unlikely to upset the user even if incorrectly recognized. In contrast, an action that is critical of the user could be quite upsetting.

Presenting a pair. When Tait presents a stimulus/response pair to the user, it goes through the following steps. Tait plays the stimulus for the user. It then waits a few seconds for a response. If the user does not respond in that period, Tait plays the secondary stimulus. If there is still no response forthcoming after a few more seconds, the catchall response is assumed to have been received, and Tait continues on without waiting any longer for the user. (The amount of time to wait is specified by the study material.)

The study material should be arranged so that the user always has the option of telling the system to pause for a while. This is illustrated in Section 3.3.

If the speech recognizer believes that two or more responses are more or less equally likely to have been given by the user, it picks the most likely valid response. This is done on the theory that it is better to pick a valid response by mistake than an invalid one.

When the response made by the user has been determined, Tait upgrades the knowledge level estimates and practice times of the items used correctly in it as discussed above. Also as discussed above, Tait downgrades the knowledge level estimates of the items used incorrectly in the response. Tait then performs the action associated with the response, if any. Finally, Tait performs the continuation, if any.

Querying features. As illustrated in section 3.3, any features that are relevant are defined by a set of study material. The study material queries the user to find out appropriate values for the features. This is done with stimulus/response pairs with neutral responses specially set up for the purpose. The actions associated with the responses store the selected values in the user model.

Units. Much of the teaching material is represented as stand alone pairs. However, sometimes it is useful to group together a number of pairs into a *unit* (e.g., in foreign language instruction, centered around a dialog). A unit contains the following information.

Name - a unique identifying name.

Pairs - an ordered list of the names of the pairs in the unit. (Typically, many of these pairs will not be stand alone pairs; however, many may be.)

Requires, teaches, practices - names of items just as in a pair. (These fields are the union of the corresponding fields of the pairs in the unit.)

A unit is presented by presenting its pairs in order, one after the other. Unless the user makes a great many errors, Tait will push right on through the whole unit. The idea here is that even if the user got an answer wrong, hearing the correct response and then repeating it may have been sufficient teaching. Even if it is not, just plowing right on is often a better thing

to do than demanding perfection at each step. Further, this approach makes Tait less sensitive to speech recognition errors.

Error rate. Tait maintains a moving average of the user's error rate. For example, if the user has gotten two out of the last ten responses wrong, the error rate is 20%. This is useful for deciding what to do next. If the error rate suddenly shoots up, it is advisable to switch to easier material or stop the lesson.

A user settable parameter is provided for controlling the maximum error rate. Some people prefer to advance slowly with a very low error rate. Others prefer to press on even with a significant error rate. Tolerating an error rate of at least 10–20% is probably a good idea.

The basic study cycle. Tait's basic study cycle is to find a pair or unit, present it, find another pair or unit, and so on, until a reasonable amount of time (say 30 minutes) has passed or the error rate suddenly rises. This is repeated daily, or more often, at the user's convenience.

To allow the algorithm for picking pairs and units for presentation to be tailored to a particular set of study material, it is included as part of the study material. Any such algorithm has to balance a number of criteria.

A pair/unit cannot be performed unless all its required items are current. Teaching should only apply to items that need to be taught. Practice should focus as much as possible on items that are due for practice and away from items that are not due.

All things being equal, it is better to prefer units over single pairs, because they are more coherent. However, it is important to choose pairs that practice material that is not well practiced by units. It is particularly important to practice items that are near to becoming current and are required by many other pairs.

Since different users will make different mistakes, the course of study for two users will be very different. Beyond this, random variation should be introduced whenever possible to reduce user boredom.

An example of a specific algorithm for controlling Tait's study cycle is given in the next section.

2.3 Prior Uses of Speech Recognition for Tutoring Systems

It is only in the past couple of years that speech recognition of anything other than a tiny fixed vocabulary has been possible in real time. As a result, it is not surprising that there has been little if any attempt to apply speech recognition to computer aided tutoring systems until very recently. However, in the past couple of years a few such systems have been developed. Most of these systems focus on foreign language education.

The Summit Literacy Tutor: Members of the Spoken Language Systems Group at MIT have been working on a tutor for teaching people to read English [6]. This tutor is based on the group's Summit speech recognition system [9]. Summit is a research prototype of a workstation based, real-time, moderate size vocabulary, speaker independent, continuous speech recognizer.

The goal of the Summit Literacy Tutor is to monitor a person who is reading aloud. The tutor can detect pronunciation errors when they occur and, using a speech synthesizer, can pronounce words and sentences when requested by the user.

Even though the tutor requires continuous speech recognition, the speech recognition is relatively straightforward, because the tutor knows exactly what the user is supposed to be reading. The difficulty is that the system has to be sensitive to subtle differences in pronunciation, in

order to guide the user to correct pronunciation.

Although similar in a number of ways, the Summit Literacy Tutor is more limited than Tait in four critical ways. First, the Summit Literacy Tutor is specialized solely for the task of monitoring someone who is reading aloud, rather than being general purpose in nature like Tait.

Second, the Summit Literacy Tutor does not have a study controller like the one in Tait. In particular, the tutor does not create a model of what the user knows, and does not vary the study material based on what the user does and does not know. It just monitors what the user chooses to read.

Third, the tutor uses a speech synthesizer instead of prerecorded sound as output. This prevents it from outputting anything other than speech. In addition, due to the fact that current speech synthesizers are not capable of producing high quality (i.e., really correctly pronounced) output, the Summit Literacy Tutor is unable to make a really good presentation for the user of how something should be pronounced.

Fourth, the Summit Literacy Tutor is based on a research prototype continuous speech recognizer that runs on a scientific workstation, rather than on a commercially available PC-based speech recognizer.

The Summit Language Tutor: Recently, the Spoken Language Systems Group at MIT has investigated using the technology underlying their Literacy Tutor as the basis for Foreign Language instruction [7]. In their initial prototype of a Language Tutor, the user is presented with written sentences in English or Japanese. The Tutor can read these sentences aloud for the user using a speech synthesizer and can monitor and comment on the user's pronunciation if he/she chooses to read the sentences. In addition, the Tutor can operate in a testing mode where the sentences are chosen at random and displayed to the user to be read.

Given that it is based on the Literacy Tutor, it is not surprising that the Language Tutor falls short of Tait in the same four ways as the Literacy Tutor. Namely, it is special purpose in nature; it neither maintains any sort of user model nor attempts to vary the course of study in any way based on what the user knows; it is restricted to using a speech synthesizer; and it uses a workstation-based speech recognizer.

The SRI Autograder system: Researchers in SRI International's Speech Research and Technology Program have also been investigating the application of speech recognition technology to foreign language education. This has lead to the development of two systems based on their DECIPHER [2] speech recognition system. Like Summit, Decipher is a research prototype, workstation based, real-time, moderate size vocabulary, speaker independent, continuous speech recognition system.

The Autograder system [1] extends DECIPHER so that it can be used to determine how well a person has spoken a foreign language utterance. As used in Autograder, DECIPHER is tuned to determine how accurately and how fluently a phrase has been uttered. Experiments have shown that the scores produced by Autograder come quite close to the scores produced by expert human listeners.

Autograder and Tait have very different goals and almost totally non-overlapping capabilities. Autograder focuses exclusively on determining how fluently someone is speaking. Other than providing simple feedback by means of scores, it makes no attempt to teach anything. Further, autograder does not maintain any kind of user model and does not adjust its operation based on what the user does or does not do.

In contrast, Tait's ability to judge fluency is very weak. Rather, it focuses on introducing new material, leaving it to users to judge their own level of fluency by comparing what they say with prerecorded native speakers. For the application of foreign language education, it would be quite profitable to combine the capabilities of Tait and Autograder. However, it is not clear whether the capabilities of Autograder can be achieved with commercially available, PC-based speech recognizers.

The SRI VILI system: The second foreign language education system developed at SRI, called VILI [1], bares more resemblance to Tait. Based an a database of stored utterances, and a 'dialog grammar', VILI can engage a foreign language student in a simple dialog. This allows the student to practice listening to native speakers and replying. At a typical moment, the student can choose between a variety of spoken responses. Using DECIPHER, VILI determines which response has been made and then continues the dialog appropriately. This allows interaction that is more flexible than rote drills, but still limited by the fact that the student must stay within the confines of the dialog grammar.

VILI is designed to be used in conjunction with Autograder, so that a students performance can be readily evaluated.

Using Tait's study material definition language (see Section 3.3), one can easily create a 'dialog grammar' essentially identical to the grammars used by VILI and thereby achieve interaction essentially identical to the interaction with VILI. However, using Tait one can do much more as well. In particular, VILI falls short of Tait in most of the same ways as the tools based on Summit.

First, VILI is specialized solely for a single task: engaging in simple foreign language dialogs, rather than being general purpose in nature like Tait.

Second, VILI does not have a general study controller like the one in Tait. In particular, VILI does not create a model of what the user knows, and does not vary the study material based on what the user does and does not know.

Third, VILI is based on a research prototype continuous speech recognizer that runs on a scientific workstation, rather than on a commercially available PC-based speech recognizer.

The MIT R/L Perception/Pronunciation Trainer: Prior to the time when real time speech recognition was possible, efforts were made to use simpler speech signal processing to aid foreign language study. An interesting example of this is the MIT R/L Perception/Pronunciation Trainer[1]. This system was designed to help a native Japanese speaker learn how to correctly pronounce and discriminate between the English phonemes R and L. The system used signal processing and the display of sound spectrograms to enable a user to see the difference between these two phonemes as a first step to learning how to hear the difference between them.

While interesting, the MIT R/L Perception/Pronunciation Trainer, bares little resemblance to Tait, because it does not make use of speech recognition, and neither maintains a user model nor varies the course of study based on user performance.

[1] Janet Murray, personal communication

3 Proof-of-Concept Demonstration

To demonstrate the technical feasibility of The Audio Interactive Tutor (Tait), a rapid prototype was constructed out of readily available components. While this was being done, studying a foreign language was used as the guiding application. The implementation was tested using some illustrative study material for Spanish and German.

3.1 Realization of the Architecture

Figure 3 shows how the generic architecture in Figure 1 was realized in the proof-of-concept prototype. For the most part, the prototype was implemented on an HP-700 series scientific workstation. However, the prototype makes use of a speech recognizer that runs on a PC. Using two different computers introduced the need for inter-machine networked communication; however, it made it possible to utilize a readily available commercial speech recognition system, without losing the benefits of a powerful programming environment for the rest of the implementation.

Audio output. The HP-UX Audio Application Program Interface [4] in conjunction with the A/D and D/A hardware built into recent HP-700 series machines is used to support prerecorded sound in the Tait prototype. The HP Audio API represents chunks of sound as disk files that can be created (by recording from a microphone), played (through speakers or headphones), and edited (with a simple sound editing program).

Because the HP Audio API expects each chunk of sound to be a separate disk file, the Tait prototype represents the prerecorded utterances required as part of its study material as a directory full of files, rather than as an integral part of the main body of a set of study material. This would be inappropriate for a commercial implementation, but is quite convenient in an experimental prototype.

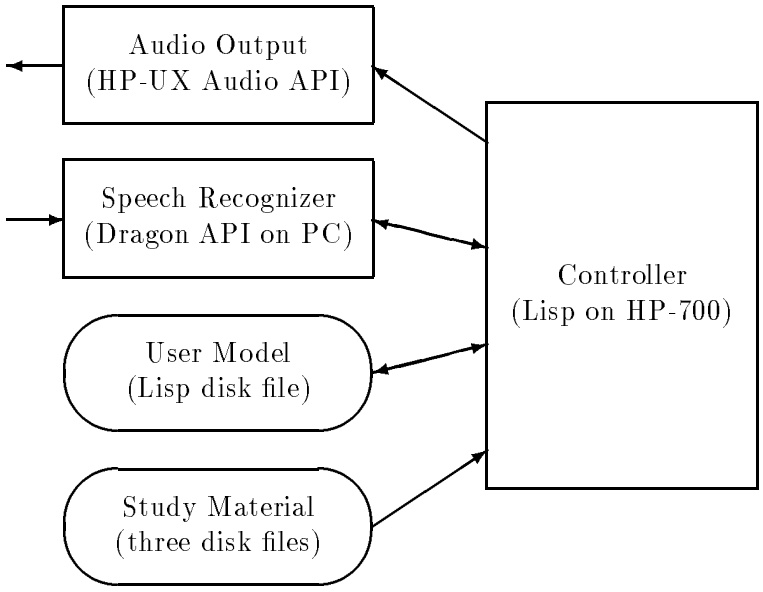

Figure 3: The proof-of-concept implementation of Tait.

As discussed in Section 3.3, a list of the utterances that need to be recorded is automatically generated from the definition of the study material. The required sound files are then created by hand using the HP Audio API sound editor. When the Tait prototype wishes to output a chunk of prerecorded sound, it simply uses the HP Audio API to play the corresponding sound file.

The HP Audio API is sufficient for supporting the Tait prototype, but falls far short of what would be needed in a commercial implementation. In particular, the fidelity is poor, the sound editing facilities are crude, and there is an annoying delay of 2–3 seconds between the time a program requests that a sound be played and the time that play actually begins. (This latter problem could be fixed by operating directly on the underlying audio device driver, rather than through the HP Audio API server.)

Since A/D and D/A sound hardware for computers is well developed and the sound processing required for Tait is trivial. It would be easy to obtain high levels of sound quality in a commercial implementation of Tait.

Speech recognizer. The beta test version of the DragonDictate 1k Speech Recognition Application Program Interface (Dragon API) [3] is used as the speech recognizer in the Tait prototype. The Dragon API is more or less typical of the state of the art of commercially available speech recognizers. In particular, it is a speaker dependent, isolated word, speech recognition system that operates in real time given a vocabulary of 1000 words or so. Like most such systems, it is designed to operate on an IBM PC compatible machine and utilizes a special A/D board.

The Dragon API has two key features that make it particularly well suited for use in Tait. First, it is delivered not as a monolithic program, but rather as a set of C subroutines, that can be used as the basis for constructed a speech recognition module interface that is tailored to a particular application. Rather than providing one particular interface, the Dragon API provides the tools for building a very wide range of interfaces.

The second key feature is that the Dragon API operates based on a *finite state grammar* (FSG) that specifies what words to look for. An FSG is organized as a collection of groups of words called 'states'. The system can be rapidly switched from one state to another, and will analyze an utterance only with regard to the words in the current state. Given an utterance, the Dragon API returns a list of the most likely words in the current state that correspond to the utterance along with scores of how well each one matches.

The use of states in an FSG has two important advantages. First, states make it possible for the application program to communicate information about expectations to the Dragon API. If an individual state is small compared to the size of the grammar as a whole, this greatly increases the recognition accuracy. Second, as long as no state contains more than 1000 words or so, there is no limit on how many words can be in an FSG as a whole.

Given a set of study material, Tait creates a Dragon FSG that has states corresponding to each stimulus/response pair. The state corresponding to a given pair, indicates all the responses the user might make. To determine what response the user makes when the stimulus is played, Tait switches the Dragon API into the appropriate state, and then triggers it to listen for and analyze the user's response.

Because the Dragon API expects an FSG to be specified in a particular kind of file in a particular format, the Tait prototype represents FSGs as separate files rather than as an integral part of the main body of a set of study material. This would be inappropriate for a commercial

implementation, but is adequate in the experimental prototype.

As discussed in Section 3.3, an FSG specifying the responses that can be made by a user is automatically generated from the definition of a set of study material. The required speech recognition information is then created manually using the speech training facilities provided as part of the Dragon API.

There is one aspect of the Dragon API that is quite inconvenient—the fact that it runs on a PC while the rest of the Tait prototype runs on an HP workstation. This necessitates remote communication between the HP and PC machines to specify what states to use and what words are recognized. This introduces significant delays and a host of complexities that would be eliminated in a commercial implementation of Tait by locating all the modules of the system on a single machine.

The points in the last few paragraphs are best viewed as basically being technical details. The key question is: how can a speaker dependent, isolated word, speech recognition system support the speaker independent continuous speech recognition required by Tait? This question is discussed at length in Section 3.2.

User model. In the Tait prototype, the user model is stored as a simple disk file that contains: the user's name, the name of the material being studied, the features that have been determined to be appropriate for the user, and for each item in the study material, its estimated knowledge level and the next time at which it should be practiced. This is read in whenever the user resumes studying and written out whenever he/she stops.

Study Material. The study material for a particular course of study is stored in three pieces. The information about each possible audio output is stored in a separate disk file as outlined in the discussion of the audio output module. The information about how to recognize user responses is stored in a Dragon FSG as outlined in the discussion of the speech recognizer.

The control information about items to be studied and their associated stimulus/response pairs is stored in a separate file as well. This information is loaded into Tait whenever a user starts or resumes studying the material.

Section 3.3 describes the language that is used to define a set of study material in the Tait prototype. A compiler (written in Lisp) converts this definition into the separate files above. Information about the exact sounds to output and recognize is then added manually.

The controller. As discussed at length in Section 2.2, the heart of Tait is its controller. A prototype controller has been written in Lisp, which runs on an HP workstation and controls the other components of the system. The controller is discussed at length in Section 3.5. However, before going into the details of the controller, it is useful to look at the speech recognizer and study material definition language in greater depth.

3.2 Speech Recognizer

State-of-the-art commercial speech recognizers support: discontinuous speech (i.e., recognize only one word at a time), moderate accuracy (i.e., at most 90% correctness or so), small vocabularies (i.e., 100 words to at most 1000 or so), and real time performance (i.e., return a result only a second or so after a user has finished speaking). Typically, systems with very small vocabularies (less than 100 words) are speaker independent while systems with larger vocabularies require training for each user.

Tait only requires very small vocabulary speech recognition and can tolerate a significant

number of speech recognition errors. However, Tait requires speaker independent operation. (The object being to train the user how to speak a foreign language correctly, not to train the speech recognizer to recognize the user's incorrect pronunciation.)

Fortunately, as noted above, as long as speech recognition is restricted to very small vocabularies, speaker independent recognition is easily achievable. For instance, even though the Dragon API is billed as a speaker dependent system, if it is only required to discriminate between a dozen words, it operates satisfactorily as a speaker independent system.

So far so good, but there is one place where on its face, there is an obvious mismatch between the commercial state-of-the-art and the needs of Tait—Tait requires continuous speech recognition. However, this is not really a problem, because while Tait requires the recognition of continuously spoken sentences, rather than the recognition of individual isolated words, this is really not the full problem of continuous speech recognition.

The full continuous speech recognition problem consists of taking a vocabulary of n words and determine what possible combination of words corresponds to an utterance U. If U consists of 10 words, this requires determining which of n^{10} possible word combinations corresponds to U. The key difficulty in this is that since there are no pauses between words in natural continuous speech, the recognition system must simultaneously determine where the words begin and end and what the words are. This requires the statistical comparison of an enormous number of possibilities.

In contrast to the above, Tait might, in a typical situation, be trying to determine whether the user said one of 6, 10 word sentences. Suppose that these sentences contain a total of 15 words. This means that Tait is only looking for 6 possible word sequences rather than $15^{10} = 576{,}650{,}390{,}625$. The problem still remains that the speech recognizer has no a priori information about where the words in each of the 6 sequences begin and end. However, this is no different from the fact that it never has any a priori knowledge about where the phonemes begin and end when looking for an individual word. The speech recognizer can just treat each sentence Tait is looking for as a very long word.

For instance, the Dragon API is happy to let you call anything you want a word. Suppose a particular stimulus/response pair requires the response "The Pacific ocean is west of the United States". There is no trouble training the Dragon API to recognize this as if it were a single word. Suppose that a possible incorrect response to this stimulus is "The Pacific ocean is east of the United States". There is no trouble training the Dragon API to recognize this as if it were a single word as well.

The key question is, are commercially available speech recognizers good at discriminating between 36 phoneme words that differ by only a phoneme or two? The answer, at least for the Dragon API, is that they seem to be.

Informal experimentation with sentences like the ones above showed that the Dragon API seems to have no trouble at all discriminating between quite long sentences that differ by only a single phoneme. In fact, its ability to discriminate between long phoneme sequences seems just as good as its ability to discriminate between short ones.

It should be noted that treating each whole response as a single vocabulary item leads to the need for recognizing a very large vocabulary. For example, suppose that a set of study material has 1000 stimulus/response pairs with an average of 5 responses each. If these 5000 responses were all different, they would lead to a vocabulary of 5000 vocabulary items even if the responses together only contained say 400 different words. However, using a focusable

system like the Dragon API, this is not a problem since one only has to be able to discriminate between 5–10 vocabulary items at a time, and therefore, the total vocabulary size is irrelevant.

3.3 Study Material Definition Language

Tait's study material definition language provides a convenient method for defining user features, study items, and stimulus/response pairs. The use of this language is illustrated in Figure 4, which shows the definition of a small bit of material that an English speaker could use to study Spanish.[2] Each part of this definition is discussed in detail below.

The study material definition language is defined as a macro extension of Lisp. (One of Lisp's greatest strengths is that such extensions are almost trivially easy to create.) A macro expansion compilation process converts the definition of a set of study material into an internal representation that is convenient for Tait's controller.

The first form in Figure 4 defines a stimulus/response pair teach-left that teaches the Spanish word for left. The macro `define-pair` provides a convenient syntax for specifying the various pieces of information that Tait needs to know about stimulus/response pairs (see pages 7–9). The definition is rendered more concise through the use of intelligent defaults and macros that expand into groups of stimulus/response pairs. This is illustrated by Figure 5 which shows the five stimulus/response pairs that result from the first form in Figure 4.

Starting at the top of Figure 5, it can be seen that it is assumed by default that a pair can be used alone unless explicitly stated otherwise. The features, requires, teaches, and practices fields default to empty lists. The stimulus must be present. The secondary stimulus defaults to the stimulus unless it is explicitly specified. By default there is no continuation.

The remaining part of a `define-pair` form specifies the responses that are expected. They are collected into three groups: *valid*, *invalid*, and *neutral*. Each group is a list of response specifications, which defaults to the empty list. Each response specification is a phrase the user might say and an action to perform if the response is encountered. Four interesting kinds of defaulting occur.

First, it is assumed by default that a valid response has no incorrect items and has as correct items everything the pair as a whole teaches or practices. It is assumed by default that an invalid response has no correct items and has as incorrect items everything the pair as a whole teaches or practices.

Second, it is assumed that every pair allows "pause" as a neutral response. The purpose of this is to ensure that the user can allows get Tait to pause in such a way that study can be resumed later without any discontinuity. The function `suspend-tait`, interrupts Tait and waits for the user to either say "resume" in which case Tait starts up again or "stop" in which case Tait halts. If the user says "resume", then Tait resumes where it left off by repeating the stimulus/response pair that was interrupted.

Third, the keyword `:otherwise` is used to indicate the response (if any) that has a null speech recognition model and will be assumed to have been given if no other response can be recognized. If no `:otherwise` response is specified, it is assumed that the `:otherwise` case is an

[2]There is considerable debate in the foreign language education community about whether foreign language study material should make use of the user's native language. The material in Figure 4 assumes that this is a perfectly good idea. However, Tait itself makes no such assumption. One could easily construct study material that consisted solely of Spanish, without any English. If desired, one could include non-verbal sounds as language independent cues.

```
(define-pair teach-left
  :teaches (left)
  :stimulus "The word for left is ‘izquierda’."
  :secondary-stimulus "Say ‘izquierda’."
  :valid (("izquierda"))
  :invalid ((:otherwise (pronounce left "iz-quier-da"))))
(define-pair practice-left
  :practices (left)
  :stimulus "Say ‘left’."
  :valid (("izquierda")))
(define-feature :gender
  (query "Are you male or female?"
         (("male" '(add-feature :gender :male))
          ("female" '(add-feature :gender :female)))))
(define-unit teach-north-american-male-and-I-am
  (define-pair teach-north-american-male
    :teaches (north-american-male)
    :stimulus "A man from the United States is a ‘norte americano’."
    :secondary-stimulus "Say ‘norte americano’."
    :valid (("norte americano"))
    :invalid ((:otherwise (pronounce north-american-male "norte -ameri-cano"))))
  (define-pair teach-I-am
    :teaches (I-am)
    :stimulus "I am is ‘soy’."
    :secondary-stimulus "Say ‘soy’."
    :valid (("soy")))
  (define-pair male-practice-I-am
    :features (:male)
    :stand-alone t
    :practices (I-am north-american-male)
    :stimulus "Say ‘I am from the United States’."
    :valid (("Soy norte americano.")
            ("Yo soy norte americano."
             (usually-no-pronoun "Soy norte americano.")))
    :invalid (("Soy norte americana."
               (refers-to-woman "Soy norte americano."))
              ("Yo soy norte americana."
               (refers-to-woman "Soy norte americano.")))))
```

Figure 4: Example Spanish study material for Tait.

invalid response.

Fourth, if there is no explicitly specified action to go with a response, one is generated as follows. If there is no action for an invalid response, a default action is created that demonstrates the first valid response to the user. If there is no action for a valid response, a default action is created that reinforces the response. The latter default can be seen in Figure 5 where the second pair, reinforce-izquierda, is used to reinforce the correct response “izquierda”. (The function `do-pair` causes Tait to present a pair to the user.)

The final interesting feature of the first form in Figure 4 is its use of the macro `pronounce`. This macro takes a word or phrase containing hyphens that break it into chunks and creates a sequence of stimulus/response pairs that guide the user through the pronunciation of the phrase one part at a time. In Figure 5 it can be seen that this macro has generated a chain of

```
(define-pair teach-left
  :stand-alone t :features nil :requires nil :teaches (left) :practices nil
  :stimulus "The word for left is ‘izquierda’."
  :secondary-stimulus "Say ‘izquierda’."
  :continuation nil
  :neutral (("pause" '(suspend-tait)))
  :valid   (("izquierda" (do-pair 'reinforce-izquierda)))
  :invalid ((:otherwise (do-pair 'repeat-izquierda-in-parts))))
(define-pair reinforce-izquierda
  :stand-alone nil :features nil :requires nil :teaches nil :practices nil
  :stimulus "izquierda"
  :secondary-stimulus nil
  :continuation nil
  :neutral (("pause" '(suspend-tait)))
  :valid   (("izquierda" nil))
  :invalid ((:otherwise nil)))
(define-pair repeat-izquierda-in-parts
  :stand-alone nil :features nil :requires nil :teaches nil :practices nil
  :stimulus "Repeat 'izquierda' in parts. da [in izquierda]"
  :secondary-stimulus nil
  :continuation (do-pair 'repeat-quierda-in-izquierda)
  :neutral (("pause" '(suspend-tait)))
  :valid   (("da [in izquierda]" nil))
  :invalid ((:otherwise nil)))
(define-pair repeat-quierda-in-izquierda
  :stand-alone nil :features nil :requires nil :teaches nil :practices nil
  :stimulus "quierda [in izquierda]"
  :secondary-stimulus nil
  :continuation (do-pair 'repeat-izquierda)
  :neutral (("pause" '(suspend-tait)))
  :valid   (("quierda [in izquierda]" nil))
  :invalid ((:otherwise nil)))
(define-pair repeat-izquierda
  :stand-alone nil :features nil :requires nil :teaches nil :practices (left)
  :stimulus "izquierda"
  :secondary-stimulus nil
  :continuation nil
  :neutral (("pause" '(suspend-tait)))
  :valid   (("izquierda" nil))
  :invalid ((:otherwise nil)))
```

Figure 5: Fully expanded version of the stimulus/response pair teach-left.

three pairs that lead the user through the pronunciation of izquierda starting with just the last syllable and ending with the word as a whole. If the user succeeds in pronouncing the whole word at the end, this is counted as having correctly practiced the word.

The second form in Figure 4 defines a pair practice-left that can be used to practice the word izquierda. It expands into the following:

```
(define-pair practice-left
  :stand-alone t :features nil :requires nil :teaches nil :practices (left)
  :stimulus "Say 'left'."
  :secondary-stimulus "Say 'left'."
  :continuation nil
  :neutral (("pause" '(suspend-tait)))
  :valid   (("izquierda" (do-pair 'reinforce-izquierda)))
  :invalid ((:otherwise (do-pair 'reinforce-izquierda))))
```

This makes use of the second pair in Figure 5. It illustrates the creation by default of a secondary stimulus and an invalid `:otherwise` response.

The third form in Figure 4 defines a feature called *gender* that is used to parameterize the study material. It expands into the following:

```
(def-feature :gender '(do-pair 'are-you-male-or-female))
(define-pair are-you-male-or-female
  :stand-alone nil :features nil :requires nil :teaches nil :practices nil
  :stimulus "Are you male or female?"
  :secondary-stimulus "Are you male or female?"
  :continuation nil
  :neutral (("pause" '(suspend-tait))
            ("male" '(add-feature :gender :male))
            ("female" '(add-feature :gender :female)))
            (:otherwise (redo-pair)))
```

The function `def-feature` defines a new kind of feature and specifies an action that should be performed to determine what feature value applies to the current user. The first time the user uses a set of study material, he/she will be queried to determine what features are appropriate. After that time, these features are remembered as part of the user model.

The macro `query` generates a stimulus/response pair that determines what feature is appropriate. It uses neutral responses and forces the user to specify a feature value before study can be continued. (The function `redo-pair` causes the current pair to be presented again.)

The last form in Figure 4 is an instance of the macro `define-unit`, which groups three pairs together as a unit. Within the scope of `define-unit`, it is assumed by default that pairs cannot operate in a stand-alone way. In Figure 4 it is explicitly specified that the last of the three pairs in the unit can nevertheless be used by itself outside of the unit.

The first pair in the unit teaches the word used to refer to a man from the United States. The second pair teaches how to say "I am". Both pairs are closely analogous to the pair teach-left.

The third pair in the unit, male-practice-I-am, practices the two items taught earlier in the unit. The most interesting aspect of the pair is its use of features. In Spanish, gender agreement between subjects and predicate adjectives means that men and women have to say many things slightly differently. In current foreign language self-study tapes, this kind of problem leads to an unavoidable awkwardness since a fixed set of output is being presented to every user. However, in Tait, one can parameterize the study material so that a given user only hears appropriate sentences.

The pair male-practice-I-am specifies that it should only be presented to male users. (Although not shown in the figure, there should be an analogous pair exists that is only presented to women.)

```
(the-word-for-left  "The word for left is ‘izquierda’.")
(say-izquierda      "Say ‘izquierda’.")
(izquierda          "izquierda")
(quierda            "quierda [in izquierda]")
(repeat-izquierda   "Repeat ’izquierda’ in parts. da [in izquierda]")
```

Figure 6: List of Tait outputs corresponding to Figure 5.

```
VOCAB "spanish.voc"  STATE MAIN
  STATE "PAUSE"                WORD "pause";;
  STATE "IZQUIERDA"            WORD "izquierda";;
  STATE "QUIERDA-IN-IZQUIERDA" WORD "quierda [in izquierda]";;
  STATE "DA-IN-IZQUIERDA"      WORD "da [in izquierda]";;
  STATE "TEACH-LEFT"                  INCLUDE STATE "IZQUIERDA";
                                       INCLUDE STATE "PAUSE";;
  STATE "REINFORCE-IZQUIERDA"         INCLUDE STATE "IZQUIERDA";
                                       INCLUDE STATE "PAUSE";;
  STATE "REPEAT-QUIERDA-IN-IZQUIERDA" INCLUDE STATE "QUIERDA-IN-IZQUIERDA";
                                       INCLUDE STATE "PAUSE";;
  STATE "REPEAT-IZQUIERDA-IN-PARTS"   INCLUDE STATE "DA-IN-IZQUIERDA";
                                       INCLUDE STATE "PAUSE";;
  STATE "REPEAT-IZQUIERDA"            INCLUDE STATE "IZQUIERDA";
                                       INCLUDE STATE "PAUSE";;
```

Figure 7: A the Dragon API grammar corresponding to Figure 5.

The macro `usually-no-pronoun` generates an explanation that subject pronouns such as yo are usually omitted when speaking Spanish. The macro `refers-to-woman` generates a pair that explains that a response is inappropriate for a man to make. The use of these macros here illustrates that Tait study material can contain explicit information that explains errors and is presented only if the errors actually occur.

An important part of the study material compilation process is the creation of files that specify what the audio output and speech recognition modules have to do. The instructions for the audio output module are in the form of a list of utterances to be recorded, along with symbolic names, which are referred to by the internal compiled form of the study material. Figure 6 shows the list of utterances corresponding to the pairs in Figure 5. This list consists of every stimulus and secondary stimulus in the pairs. Note that while `"izquierda"` only appears once in Figure 6, it is the stimulus for two different pairs.

As discussed in Section 3.1, the instructions to the Dragon API speech recognizer are in the form of a finite state grammar (FSG). The FSG segment corresponding to Figure 5 is shown in Figure 7. It contains 5 primitive word states corresponding to the 5 different things the user can say. It contains 5 compound states corresponding to the 5 stimulus/response pairs in Figure 5.

As an example of how the information in Figures 6 & 7 is used, consider the following. When presenting the stimulus response pair reinforce-izquierda, Tait first asks the audio component to play the utterance `izquierda`. It then asks the Dragon API to listen for a one of the responses in the state `REINFORCE-IZQUIERDA`. Based on whether the user gives one of the expected responses, a response that cannot be recognized as one of the expected responses, or no response at all,

Tait then decides what pair to present next.

3.4 A Transcript of Tait in Action

The best way to get a feel for how Tait study material works is to see it being used. Figure 8 shows a transcript of a user, who we will call Dave, studying the material in Figure 4. Out of necessity, Figure 8 presents the output from Tait and the input from the user as text. However, the real interaction is all via prerecorded sound output from Tait and spoken input from Dave. Commentary (in italics) is included to indicate what part of Figure 4 each part of Figure 8 comes from.

Figure 8 assumes that Dave is starting to use the study material for the first time. Pursuant to this, the first thing that Tait does is to ask Dave what his gender is (see part 1 of the figure).

In part 2, Tait presents the pair teach-left. Dave does not repeat izquierda correctly, so Tait takes Dave through the pronunciation of the word in sections.

In parts 3 through 7, Tait presents the unit teach-north-american-male-and-I-am. This begins (in part 3) by teaching the phrase 'norte americano'. When first presented with the phrase, Dave fails to respond. After a few seconds, Tait prompts Dave a second time.

Flustered, Dave tells Tait to pause while he collects his wits (see part 4). After a minute, Dave tells Tait to resume. In part 5, Tait presents the phrase 'norte americano' again. No longer flustered, Dave responds correctly and Tait reinforces his response.

Following this, Tait presents the next pair in the unit (see part 6). Dave muffs his response totally at first. After hearing the word again, Dave gets closer, but still mispronounces the word.

Tait concludes the unit in part 7 of the figure by presenting the pair male-practice-I-am. Note that Tait does these even though, in its estimation, Dave does not know the verb soy. This is done on the theory that it is better to push on through a whole unit, rather than over micro-manage the course of study.

In part 7, Dave responds almost correctly. Tait comments that it is better not to use the subject pronoun yo in this situation, and Dave repeats the correct response.

The unit completed, Tait presents the pair practice-left (see part 8). Even with two tries at it, Dave is unable to produce the correct response. Tait therefore concludes that Dave does not know izquierda. (After part 2, Tait had thought he did.)

In part 9, Tait proceeds to teach Dave about izquierda again. Tired of deliberately making a lot of mistakes in order to demonstrate the flexibility of Tait's study material, Dave stops the study session in part 10 of Figure 8.

So that Dave can resume studying where he left off at a later time, Tait saves its model of what Dave knows. The contents of this model are shown in Figure 9. The first three items record Dave's name, what he is studying, and his features relative to that material. For each item in the study material, the model contains Tait's estimate of Dave's knowledge level and the time at which repractice should occur.

The knowledge level integers are explained in Figure 2. Because Dave got the item I-am right in part 7 of Figure 8, but had previously gotten it wrong, the model rates Dave's knowledge level for this item at the minimal value of -2. In contrast, Dave got north-american-male right in both parts 5 and 7, with a modest time interval in between. As a result, the model rates Dave's knowledge of this item at -1. Since Dave has repeatedly gotten the item left wrong, it is assigned the null knowledge level, indicating that it should be taught again.

1: *Tait asks what Dave's gender is.*

```
Tait: Are you male or female?
Dave: male
```

2: *Tait presents teach-left.*

```
Tait: The word for left is 'izquierda'.
Dave: izqu...
Tait: Repeat 'izquierda' in parts.  da
Dave: da
Tait: quierda
Dave: quierda
Tait: izquierda
Dave: izquierda
```

3: *Tait presents teach-north-american-male.*

```
Tait: A man from the United States is a 'norte americano'.
Dave:
Tait: Say 'norte americano'.
```

4: *Dave has Tait pause.*

```
Dave: pause
Tait: Say 'resume' to continue working with Tait.
Dave: resume
```

5: *Tait resumes presenting teach-north-american-male.*

```
Tait: A man from the United States is a 'norte americano'.
Dave: norte americano
Tait: norte americano
Dave: norte americano
```

6: *Tait presents teach I-am.*

```
Tait: I am is -- 'soy'.
Dave: s...
Tait: soy
Dave: soys
```

7: *Tait presents male-practice-I-am.*

```
Tait: Say 'I am from the United States'.
Dave: yo soy norte americano
Tait: That is correct, but the subject pronoun is usually omitted.
      It is better to say: 'soy norte americano'.
Dave: soy norte americano
```

8: *Tait presents practice-left.*

```
Tait: Say 'left'.
Dave: izquierdo
Tait: izquierda
Dave: izqui...
```

9: *Tait presents teach-left.*

```
Tait: The word for left is 'izquierda'.
```

10: *Dave has Tait stop.*

```
Dave: pause
Tait: Say 'resume' to continue working with Tait.
Dave: stop
```

Figure 8: A transcript of Tait in action.

```
:USER Dave
:STUDY-MATERIAL Spanish
:FEATURES (:gender :male)
:CURRENT-TIME 11/10/1993 14:56
:KNOWLEDGE-LEVEL-ESTIMATES
  I-am                -2   11/10/1993 14:59
  north-american-male -1   11/10/1993 15:06
  left                nil
```

Figure 9: The user model at the end of Figure 8.

In Figure 9 it is assumed that Dave terminated the session at 2:56 in the afternoon. The practice times specify that I-am should be practiced within the next 3 minutes and north-american-male should be practiced in the next 10. The item left should be taught as soon as possible.

3.5 Controller

As noted in Section 2.2 the control algorithm used by Tait should properly be considered part of the study material rather than as a static part of Tait. However, to date only one example control algorithm has been implemented. This algorithm was designed with foreign language study in mind and was tested on small samples of study material for Spanish and German. (Part of the Spanish material is shown in Figure 4.)

Inasmuch as Tait's control algorithm is intended to be fluid, and will undoubtedly change in the future even as applied to foreign language study, it is not appropriate to discuss the algorithm in minute detail here. However, it is illuminating to discuss the basic concepts that are used to decide what pairs to present to the user when.

Utility of practicing items. For each item, Tait computes the utility of practicing the item. If an item has not been taught, or its practice time is far in the future, the practice utility is negative. The utility rises as the practice time approaches, becoming very large if the desired practice time was missed a long time ago in comparison with the associated practice interval. For instance, if the desired practice interval was one day, and three days have passed since the last practice, the practice utility will be very high. Random noise is introduced into the utility values in order to break ties and introduce greater variety into Tait's actions.

Utility of teaching items. For each item, Tait also computes the utility of teaching the item. This utility is negative if the item does not need to be taught. Otherwise, the utility depends on how many pairs refer to the item—an item that is used a lot is considered more important to teach. As above, random noise is introduced.

Utility of presenting pairs. Based on the above, the controller computes the utility of presenting each pair. To start with, the utility is negative if the pair cannot be presented because some required item is not current (Section 2.2). It is also negative if the pair teaches an item that should not be taught or practices an item that does not need to be practiced. Beyond this the utility of a pair is more or less the sum of the teaching utilities of the items it teaches and the practice utilities of the items it practices.

Utility of presenting units. Based on the utility of the pairs, the controller computes the utility of presenting the various units in the study material. To start with, the utility is

negative if the unit cannot be presented because some pair in it cannot be presented. Beyond this, the utility of a unit is more or less the sum of the utilities of the pairs in it.

Incremental computation. As befits a prototype, Tait's current controller recomputes all the information above every time it looks for new pairs to present. In study material of realistic size where there are likely to be hundreds of not thousands of items, this would be prohibitively costly. Fortunately, it would be relatively straightforward to alter the computations so that they are computed incrementally—i.e., so that the utilities of items and pairs were recomputed only when some significant relevant change occurs. For instance, if an item becomes current, one has to reconsider the utility of any pairs that require it to be current. However, in the absence of this item becoming current, none of these pairs ever has to be considered.

Selecting some pairs to present. Given the information above, Tait's controller selects what items to present as follows. To start with, it considers the unit with the highest positive presentation utility (if any). If there are stand-alone pairs that have higher individual utilities than this best unit, the controller selects up to ten of them for presentation. Otherwise, it presents the best unit (if any). If there are no units or pairs with positive utility, the controller terminates study. (With study material of realistic size, this situation should never arise.)

It should be noted that at a given moment, you would expect that relatively few units will have positive utility. In particular, many will be blocked because they teach things that do not have to be taught, or depend on items that are not yet current, or practice items that do not need practice. In contrast, the practice utility of recently learned items rises very rapidly, because the practice intervals are short. The only way that a unit can rise to the top is if it happens to practice many recently learned things, or there is nothing that is much in need of practice.

For all these reasons, the controller often picks stand-alone pairs rather than units. However, whenever the user catches up on practice, the controller pushes on with a unit that teaches new material.

Monitoring the user's error rate. Once a unit or a group of high utility pairs has been selected, the controller presents them one by one. In general it presents all the selected pairs no matter how well the user does in responding to them. This is done on the theory that it is not wise for the user to strive for total perfection at all times, and also because the inevitable inaccuracies of the speech recognizer mean that the controller should not take any one mistake too seriously.

However, while presenting pairs, the controller maintains a moving average of the number of errors the user makes. If the error rate rises sharply while presenting a group of pairs, the controller stops presenting the pairs and selects a new group of pairs. By means of a parameter, the user can control how great an error rate Tait will allow.

If the user's performance deteriorates to the point that a high error rate continues through several attempts to present groups of pairs, the controller suggests that the user stop studying for a while.

Limiting the length of a lesson. On the theory that it is not good for a user to study too long at once, the controller monitors the total time spent with by the user. As currently implemented, it stops the lesson after half an hour unless the user requests that the lesson continue.

4 Future Directions

The proof of concept implementation of Tait presented in Section 3 demonstrates the technological feasibility of the system. However, much more work would be required to create a full prototype that could be used for experimentation with real users. This work is required in two principle areas: developing a much fuller set of study material and reimplementing the system to be more robust.

Developing study material. The greatest obstacle to be overcome before user experimentation with Tait can begin is the design and production of a reasonable set of study material. To date, only a few minutes of study material have been developed. To do real experiments, at least a half hour or so would be required. (Each minute of material supports many minutes of study, due to the repetition required.)

Unfortunately, developing study material is far from easy. The material has to be designed based on a good understanding of the domain of study and the kind of learning that Tait supports. A particular difficulty is that a lot of effort has to be expended thinking about exactly how the various parts of the study material will fit together. After design has been completed, the stimuli have to be recorded, and statistical models created for the responses.

Judging from the effort that seems to be required to produce current foreign language self-study tapes, it would not be surprising if a week or more of effort was required to produce a single minute of good study material. Further, this effort has to come primarily from domain experts, not computer scientists.

One way to greatly simply the task of creating prototype study material for Tait would be to link up with an organization already making related material. For example, it might be possible to put together adequate Tait study material based on the material currently used in a set of foreign language self-study tapes.

Reimplementing Tait. Tait should be reimplemented so that it runs entirely on a single computer. This would remove delays from the system, make the system more robust by eliminating network difficulties, and make the system easily portable. As outlined in the following sections, consolidation of Tait onto one machine could be done in three principle ways depending on the goals of the desired prototype.

Tait's controller should be reimplemented to make it more useful in the face of unusual user input. Beyond this, experimentation with real users would undoubtedly point out many areas where the controller could and should be further improved.

Conveniently, since the initial implementation of Tait only required about a couple of man months of effort, there is no reason to believe that more than a year would be required for the implementation of a full prototype.

4.1 A Workstation-Based Implementation

An obvious way to move Tait to a single computer would be to consolidate it on a scientific workstation. This is logical given that the current prototype already locates everything but the speech recognizer on such a machine. In addition, it would allow the control program to continue to be written in Lisp and maintain access to a high quality debugging environment. Both of these factors would promote the continued rapid improvement of the system.

However, finding an appropriate speech recognizer that runs on a scientific workstation is not an easy task. The problem is that commercial speech recognition efforts have focussed

on PC-class machines. More or less the only speech recognizers that are available for UNIX machines are research prototypes. These systems are at one time: not well productized, much more powerful than necessary for Tait, and due to the way they are implemented much harder to adapt for use in a foreign language. It would be interesting to experiment with the greater power, but it might not be practical to get acceptable foreign language performance.

4.2 A PC-Based Implementation

Another approach would be to consolidate Tait on a PC-class machine, by rewriting the controller in C and using a PC sound output board. Since the computation required by the controller is modest, it would not stress the computation or memory limits of a PC.

The principle advantage of consolidating the implementation of Tait on a PC would be that this would allow experimentation with what would very likely be the final delivery platform for a commercial implementation of Tait. The only difference between an experimental PC-based Tait prototype and a likely commercial implementation is that it would probably be wise to continue to store study material on magnetic disk rather than a CD-ROM in order to allow for easy modification of the material.

It is worthy of note that the kind of hardware support required by Tait is rapidly becoming standard on PCs. For instance, more and more PCs come with reasonably good A/D and D/A hardware built in. In addition, CD-ROM drives for PCs are rapidly falling in price, and PCs are beginning to appear with built-in CD-ROM drives.

Finally, speech recognizers of the kind required by Tait are available for PCs from many sources. Due to rapid evolution of these systems and fierce competition between various vendors, the day is rapidly approaching when they will only cost a few hundred dollars including the required A/D hardware.

4.3 A Special-Purpose Device Implementation

A key advantage of using PCs as a final delivery platform for Tait would be that Tait could be sold purely as software that runs on hardware the user already owns. Tait would be delivered in the form of a CD-ROM containing software and study material, rather than as a hardware device. However, using PCs as the delivery platform has some drawbacks as well.

To start with, if the user does not already own a PC, the user will have to buy one—a somewhat expensive proposition. Further, if a user wants to take advantage of the full hands-free, eyes-free, portable nature of Tait, the user will have to buy a lap top PC, A/D hardware, D/A hardware, and a CD-ROM drive—a quite expensive proposition.

These problems could be avoided by building a special-purpose device that supports Tait. Such a device would include: a 386-level processor, A/D and D/A hardware, floppy disk drive, and a CD-ROM drive. Floppy disks would be used to store user models and, via these models, to identify users. To use a Tait device, the user would turn it on, insert his floppy to identify himself, and a CD-ROM containing the study material. All further interactions would be verbal. To stop, the user would merely turn the Tait device off.

A special-purpose Tait device would not include either a keyboard or a video display. This would have two beneficial effects. First, it would be easy for a Tait device to be small—perhaps no more than twice the size of a Sony Discman. Second, since keyboards and displays are significant parts of the cost of PCs, it should be easy for a Tait device to be much less inexpensive than a general purpose PC.

5 Conclusion

There are two principle ways to look at Tait: as research on computer aided instruction (CAI) in general and as a practical step forward in comparison to the current state of the art of self-study tapes.

From the perspective of research on CAI, Tait is deliberately modest in scope. In particular, Tait's focus on rote memorization renders it closer in spirit to programmed instruction than to advanced AI-based CAI systems that attempt to understand what the user is thinking or provide intelligent corrections.

However, Tait's study approach is interesting, because of its potential for highly efficient memorization. As discussed at length above, this approach relies on sparse repetition carefully spread over time, rather than copious repetition.

All in all, the best way to view Tait is not as an advance to far horizons, but rather as a pragmatic advance over current self-study tapes that is of significant value and yet well within the current technical state of the art. Tait achieves this by starting with the kind of prerecorded material that is embodied in current self-study tapes, and adding commercially available speech recognition and a controller that tailors the course of study to what the user knows.

A commercial version of Tait running on a PC could easily be produced in a year or two at most. There is every reason to believe that such a system would be significantly more effective than current self-study tapes.

Acknowledgments

A number of people assisted with the proof of concept implementation of Tait presented in Section 3. Eric Conrad implemented the communication channel between the workstation and the PC. Michal Roth implemented much of the interface to the Dragon API system. This was extended by Leejay Wu, who also helped implement the sound output module and construct initial study material. The Center for Excellence in Education provided support for Leejay Wu during his month at MERL, through their Research Science Institute.

References

[1] J. Bernstein, "Speech Recognition Technology for Language Education", Speech Research & Technology Program note, SRI International, Menlo Park CA, November 1993.

[2] M. Cohen, H. Murveit, J. Bernstein, P. Price, and M. Weintraub, "The DECIPHER Speech Recognition System", *Proc. International Conference on Acoustics, Speech, and Signal Processing*, 77–80, 1990.

[3] Dragon Systems inc., "DragonDictate Speech Recognition Application Program Interface", beta release documentation, Dragon Systems inc., 1992.

[4] Hewlett-Packard Co., "Using the Audio Application Program Interface", user's manual, Hewlett-Packard Co., 1991.

[5] G.P. Kearsley (ed.), *Artificial Intelligence & Instruction: Applications and Methods*, Addison Wesley, Reading MA, 1987.

[6] M. McCandless, *Word Rejection for a Literacy Tutor*, SB Thesis, MIT, Cambridge MA, May 1992.

[7] M. Phillips, *et.al.*, "Language Tutor: An Interactive Aid for Teaching English and Japanese", in V. Zue (editor), *Annual Research Summary, Spoken Language Systems Group*, MIT Laboratory for Computer Science, Cambridge MA, November 1993.

[8] P. Pimsleur, *Speak & Read Essential Spanish*, Heinle & Heinle Enterprises Inc., publishers 1988.

[9] V. Zue, *et.al.*, "The SUMMIT Speech Recognition System; Phonological Modeling and Lexical Access", *Proc. International Conference on Acoustics, Speech, and Signal Processing*, 49–52, Albuquerque NM, April 1990.