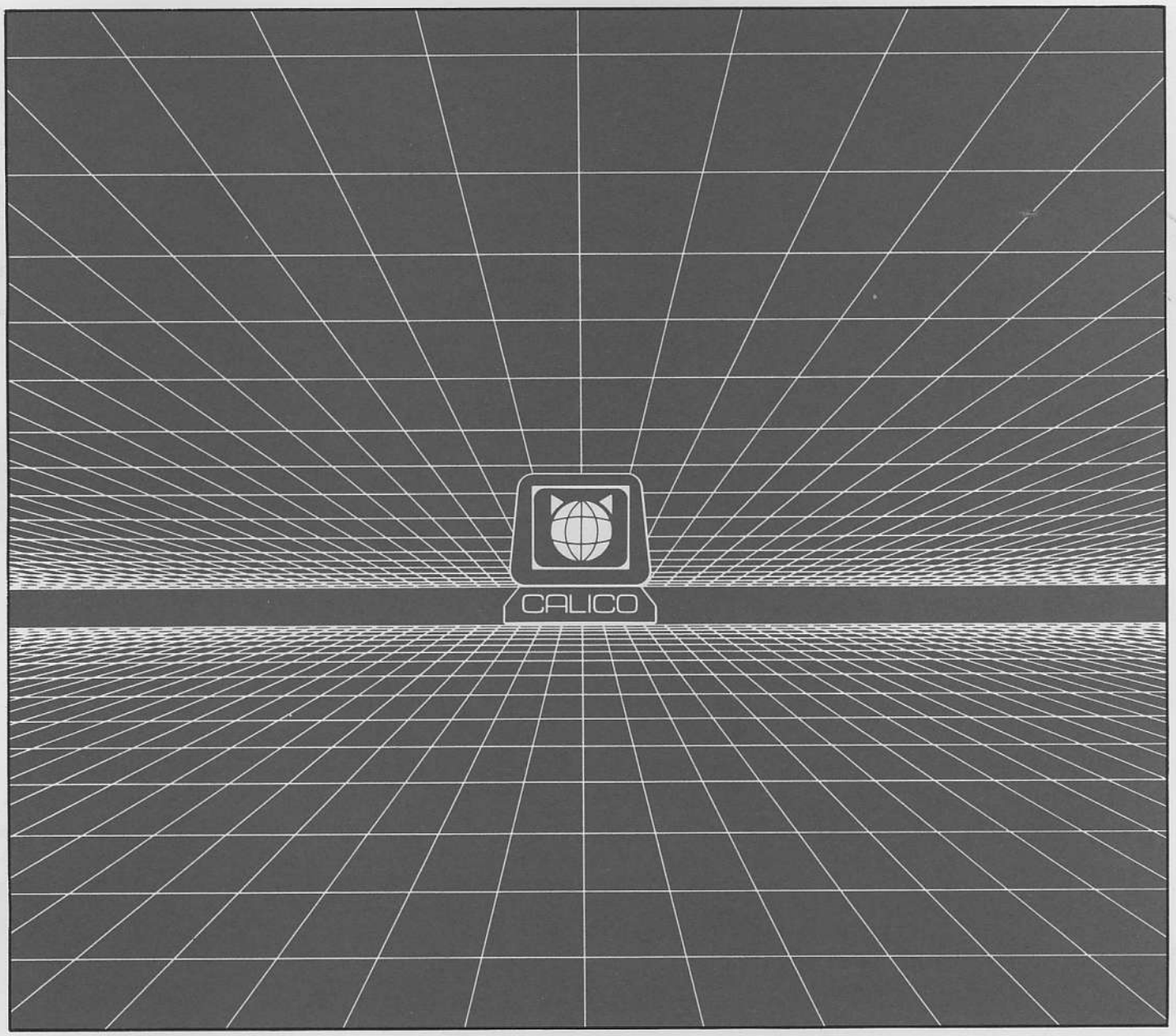
COMPUTER ASSISTED LANGUAGE LEARNING & INSTRUCTION CONSORTIUM

CALICO JOURNAL



VOLUME 3, NO. 4

JUNE 1986

CONTENTS

CALICO JOURNAL

COMPUTER ASSISTED LANGUAGE LEARNING & INSTRUCTION CONSORTIUM

VOLUME 3, NO. 4 JUNE 1986

FEATURES

DEPARTMENTS

OTHER

MASTER(')S VOICE: THE VICTOR 9000 AND HIGH-FIDELITY VOICE REPRODUCTION FOR CALI

William B. Fischer

ABSTRACT

Computerized speech can be incorporated into proficiency-oriented language instruction. The most suitable form is not voice synthesis or linear predictive coding, but rather direct digitization. This article discusses speech digitization with the Victor 9000 computer, including recording procedures, programming techniques, and pedagogical strategies and concerns.

KEYWORDS: computerized speech, voice synthesis, LPC, digitization, proficiency, ILR, ACTFL, ETS, Victor 9000, interactive audio, German, BASIC.

Language teaching in the 1980s has been invigorated by the promise of CALI and by the insistence on practical proficiency as the basis for teaching and testing. There has been some mutual enrichment between the two pursuits, disparate though they may seem initially. Computer-assisted testing of reading and listening proficiency has been discussed by Wyatt (1984). Kossuth (1984) has described interactive, contextual exercise of lower-level writing skills with a German version of *ELIZA*. Quite different but equally appealing is some commercial software, such as that which now accompanies *Allons-y!* and *Puntos de Partida*. Reviewers like Hirsch (1985) are also helping to guide creators of software away from the all-too-tempting concentration on mechanical vocabulary and form drills.

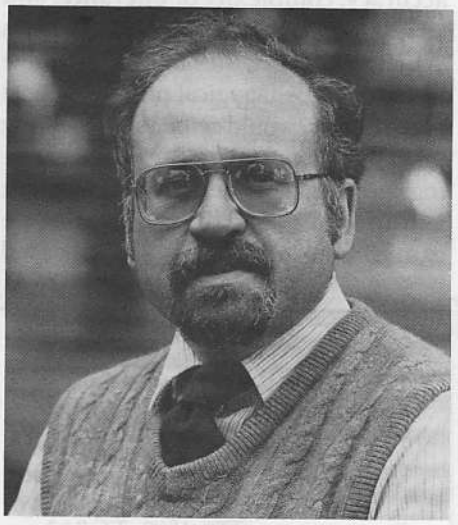
There remains, of course, a troubling discrepancy between common CALI facilities and the ideals of proficiency-oriented teaching and testing. To put the matter more positively, we face a major cybernetic, technological, and pedagogical challenge. In the customary CALI environment, the student views a computer screen and responds through a keyboard; the student reads and writes, but does not need to hear or talk. How—if at all—can and should computers be used to encourage, exercise, and evaluate proficiency in listening and speaking? While the spoken language has been the chief focus of proficiency advocates, it remains by far the most refractory area of CALI development.

Some admirable work has been done. Audio and video cassette-players interfacing with computers is indeed possible, and might well aid in reinforcing and testing certain features of language proficiency. But the sequential storage inherent in tape recording introduces unacceptable delays when pedagogical considerations dictate playback in some other order. Moreover, it has not been possible, at least until recently, to access short segments of linguistic material with the precision and the facility for analysis and interactive response which are taken for granted in programs that display on screen the printed language.[1]

Voice-recognition equipment is also available now. Computers with appropriate peripherals can be used in "voice-based learning systems" as, so to speak, voice coaches that examine and correct pronunciation (Baker 1984; Wagers 1984; Wohlert 1984). But since the sound manipulation is limited to a few short, discrete utterances, such facilities still do not satisfy the insistence on realistic, contextual language which, for advocates of proficiency, is paramount.

Voice synthesizers, now becoming readily available and often touted as spectacular enhancements, are presently a disappointment and are likely to remain so, at least for many years to come. Synthetic speech which is but "reasonably understandable" (Neudecker 1985, 144), though that is a wondrous feat of computerization, simply will not do for language instruction.

To better understand what we might dream of in CALI materials for the modalities of listening and speaking we need only contemplate three popular models of interaction between language learners and language teachers or testers: Krachen's helpful "i + 1" conversant, the ILR/ACTFL/ETS oral proficiency interview, and the *ELIZA* program in its various CALI versions (Kossuth 1984; Kramsch 1985). The three share several features whose



William B. Fischer (Ph.D. Yale, 1979) is Associate Professor of German at Portland State University. His chief teaching interests are lower-level language instruction and proficiency testing; his general research field is the interaction of science and technology with language and literature. Publications include an article on large-scale oral testing (*Unterrichtspraxis*, Fall 1984) and a book about German science fiction. He is presently preparing a proficiency-oriented first-year German text for publication with John Wiley and Sons, Inc.

desirability in language instruction programs is matched by the difficulty of implementing them on computers: sustained, realistic, contextual discourse; interactive response; tolerance of variety and even some error in student language production; and, in the first two instances, emphasis on listening and speaking.

If, then, we wish to employ computers to promote and measure proficiency in managing the spoken language, we need CALI facilities which can emulate real speakers—above all in phonological accuracy, complexity of utterance, and adaptation to context. And, like human language teachers, they must also be able to attend to pedagogical purposes. Most of us, I think, would settle for something far more modest than the famous talking computers of science-fiction cinema; and indeed, we shall have to. But we cannot compromise on the essential standard: natural-sounding speech, conveniently produced and readily integrated into programs having a realistic context.

The main sections of this article describe the Victor 9000 and its "Audio Tool Kit," a virtually unknown computer voice production system which I think meets those standards, at least in the reproduction of speech. Here a caveat is in order. Computers which will process student speech in a manner characteristic of human conversants are simply out of the question for the foreseeable future, that is, at least during our own lifetimes. As language instructors we shall have no magical "HAL" promising to lighten our load or threatening to replace us in what is, in fact, our inalienable function: intensive, free-swinging, adaptive communication in realistic contexts, where the instructor not only produces speech but also responds to it. But if the computer cannot yet listen at all well, it (or at least one computer) can indeed talk, fluently and in any language.

The Victor 9000 computer was introduced in the United States in early 1982. Neither the popular computer publications nor the specialized CAI literature has made any but the barest mention of it, and understandably so.[2] The Victor 9000 is a relatively expensive computer intended for the business market. It is mostly sold abroad, and the company has shown no interest in serving the educational market, not even the foreign-language field to which it might so plausibly appeal.

The Portland State University Department of Foreign Languages acquired its Victor 9000 in the summer of 1983. Its chief appeal, other than a comfortable screen, excellent keyboard, and

DAY OF THE WEEK:

Sunday	SS SS AX AX NN1 PA2 DD2 EY
Monday	MM AX AX NN1 PA2 DD2 EY
Tuesday	TT2 UW2 ZZ PA2 DD2 EY
Wednesday	WW EH EH NN1 ZZ PA2 DD2 EY
Thursday	TH ER2 ZZ PA2 DD2 EY
Friday	FF RR2 AY PA2 DD2 EY
Saturday	SS SS AE PA3 TT2 PA2 DD2 EY

MONTHS:

January	JH AE AE NN1 YY2 XR 1Y
February	FF EH EH PA1 BR RR2 UW2 XR IY
March	MM AR PA3 CH
April	EY PA3 PP RR2
May	IH IH LL MM EY
June	JH UW2 NN1
July	JH UW1 LL AY
August	AO AO PA2 GG2 AX SS PA3 TT1
September	SS SS EH PA3 PP PA3 TT2 EH EH PA1 BB2 ER1
October	AA PA2 KK2 PA3 TT2 OW PA1 BB2 ER1
November	NN2 OW VV EH EH MM PA1 BB2 ER1
December	DD2 IY SS SS EH EH MM PA1 BB2 ER1

Listing 1: *This BASIC program causes the Sweet Talker II to pronounce the word "hello" by sending it the minimal set of phoneme codes.*

```
10 REM SET UP SSI263 FOR TRANSITIONED INFLECTION
20 POKE 50243,255 :REM CONTROL BIT EQUALS 1
30 POKE 50240,192 :REM SET PHONEME DURATION
40 POKE 50243,116 :REM CONTROL BIT EQUALS 0 AND SET AMPLITUDE
50 POKE 50244,231 :REM SET FILTER FREQUENCY TO NORMAL
60 POKE 50242,168 :REM SET SPEECH RATE TO NORMAL
70 POKE 50241,127 :REM SET INFLECTION LEVEL
100 HOME
110 PRINT "HELLO"
120 DATA 44,10,32,17,35,0 :REM PHONEMES FOR WORD HELLO
130 FOR X=1 TO 6
140 READ A
150 POKE 50240,A :REM LOAD PHONEME INTO INPUT REGISTER
160 FOR T=0 TO 50 :NEXT T :REM DELAY TIL NEXT PHONEME TIME
170 NEXT X
180 END
```

Figure 1: *Examples of Voice Synthesis Coding*

Left: Sample coding sequences for common words. The facility, the Radio Shack "Archer" SPO256-AL2 Narrator Speech Processor (list $12.95), provides 59 allophones and 5 different pauses. Coding is sophisticated enough to distinguish, for example, r-colored vowels and to permit doubling for stressed syllables; compare the renditions of the short "e" sounds in "september" (EH, EH EH, ER1), and contrast the first to the use of IY (long "e") in "December. There is, however, no provision for subtle coarticulation ("blending") or adjustment of pitch and amplitude. Source: product manual, catalog number 276–1784.

Above: BASIC program to produce the single word "hello" on the Sweet Talker II voice synthesizer, using the minimal set of phoneme codes, and with no provision for variation in pitch, pace, or intonation. Source: (Ciarcia 1984, 3). Pages 34–5 of the same source provide an assembly-language program which utters the same word with better inflection and intonation; the program consists of 104 lines.

fast operation, was its convenient management of foreign-language characters—features which are now readily available on other computers. But we soon realized that in the realm of computerized voice production the Victor 9000 might well be unique. With its modestly-priced "Audio Took Kit" ($345 list) the computer permits the facile recording, storage, and playback of human speech in any language. The menu-driven recording procedure requires no knowledge of programming—playback fidelity is comparable to that of a tape recorder—and the sound elements themselves are invoked, rapidly and in any sequence, by simple program statements. I am not aware of any comparable system.

The triadic classification, formulated several decades ago, can still be accepted as a description of fundamental types, but it must be amplified by reference both to the earlier stages of computerized speech development and to hybrid products just recently placed on the market. One must also consider certain profound differences among those interested in computerized speech.

Until just a few years ago, computerized speech was the province of computer and electronics specialists, either the individual hobbyist or the market-niche corporation. Industrial users, such as manufacturers of talking clocks or warning devices, encouraged the development of speech chips on which were recorded fixed vocabularies of up to a few hundred words. Customization was extremely expensive, and while the individual voice segments might well sound very human, their concatenation introduced the grievous flaws in coarticulation and intonation which have engendered the common impression of computer speech as "robot-like."

An obvious enhancement of speech synthesizers was the introduction of speech produced by the concatenation of stored phonemes or even allophones, perhaps with provision for variation in speech rate and pitch (Teja 1981,58–60). An assessment of the problems of word or phoneme concatenation, and a brief examination of coding samples and program listings (see Figure 1), lead one to conclude that such speech synthesizers are of no practical use to the proficiency-oriented language teacher.

Elimination of cumbersome coding, or rather, provision for simple text-to-speech conversion, has long been an evident desideratum, one addressed early by another special-interest group, suppliers of aids to the handicapped. Recently text-to-speech facilities for popular personal computers have

MATCHING STRING	ALLOPHONES	EXAMPLE
<[E]<	=IY	"E"
#:[E]<	=""	FADE
'*[E]<	=""	I'VE, WE'RE, YOU'RE
*[E]<	=IY	SHE, ME, WE
#[ED]<	= PA2, DD1	TRIED, GREED, SPEED
#:[E]D<	=""	DRAGGED
[EV]ER	=EH,VV	EVERYTHING
#*[EL]	=EL	GRAVEL, TRAVELER
[ERI]#	=YR, IY	EXPERIENCE
#:[ER]#	=ER1	GENERAL, OBLITERATE
[E] %	=IY	DELETING
[ERI]	=EH, EH, RR1, IH	PERILOUS, PERIMETER
[ER]#	=EH, XR	
[ER]	=ER1	OBSERVE, UNDERSTAND
<[EVEN]<	=IY, VV, IH, NN1	EVEN
<[EVEN]	=IY, VV, EH, EH, NN1	EVENT, EVENTUAL
#:[EW]	=YY1, UW2	SINEW
@[EW]	=UW2	DEW, CHEW, NEW
[EW]	=YY1, UW2	FEW, SPEW

\# = One or more Vowels (A, E, I, O, U, or Y)
. = One or B, D, G, J, L, M, N, R, V, W, X
% = One of ER, E, ES, ED, ING, ELY (A suffix)
& = One of S, C, G, Z, X, J, CH, SH (A sibilant)
@ = One of T, S, R, D, L, Z, N, J, TH, CH, SH preceeding a "u" sound. (Ex. rule, mule)
= One consonant (B, C, D, F, G, H, J, K, L, M, N, P, Q, R, S, T, V, W, X, Z)
\+ = One of E, I, Y (A front vowel)
: = Zero or more consonants
* = One or more consonants
> = One of O or U
< = Anything other than a letter
? = Zero or more vowels
Brackets [ ] surround the pattern to be substituted.

Each rule is applied by first matching the characters enclosed in brackets, then working outward from these characters, first to the left and then to the right, to determine if the character pattern matches the rule being tested.

Looking at a specific rule:
#:[E]< = ""
would be read as:
The letter "E" which is preceeded by zero or more consonants preceeded by one or more vowels and followed by any non-alphabetic character (space, punctuation, etc...) is to be replaced by a null string (the "E" will be silent). In the example word "FADE", the "E" is preceeded by one consonant ("D"), which is preceeded by one vowel ("A"), and it is followed by a non-alphabetic character (end of the word), therefore the "E" is silent.

One other example:
#:[ER]# = ER1
The letters "ER" which are preceeded by zero or more consonants preceeded by one or more vowels and which are followed by one or more vowels is replaced by the allophone "ER1". In the example word "GENERAL", the "ER" is preceeded by one consonant ("N") which is preceeded by one vowel ("E") and is followed by one vowel "A", therefore the "ER" is pronounced as the allophone "ER1".

Note that the sequence of the rules is not arbitrary. It is important to realize that a very general rule may be a subset of a more specific rule, and therefore care was taken in its placement so that a more specific rule preceeded a more general rule, since the rules are searched sequentially.

Figure 2: *Excerpt from an algorithm for text-to-speech conversion, showing part of the rules for pronouncing the letter "e" in English texts used as computer input. Source: product manual 276–1786 for the Radio Shack "Archer" CTS256A;AL2 Code-to-Speech Processing Chip (list $16.95), which is used in conjunction with the SPO256 Speech Processor described in Figure 1.*

become available. An example is First Byte's "SmoothTalker" ($149.95 list) for the Macintosh. One reviewer states that the "impressive" software package "allows you to produce reasonably understandable speech by typing English text" (Neudecker 1985, 143-44). The present writer found unacceptable for language-teaching purposes the speech sample offered through First Byte's telephone demonstration (213)427-0178. Accent and intonation were both false. The speech was virtually monotone, consonants were often garbled, and the voice was accompanied by a persistent ring or echo. Yet the claims of suppliers and the praise of enthusiasts are probably sincere. Kuecken (1983) has observed that those who are exposed to synthetic speech—by now most of us—steadily gain facility in comprehending it (141). The improvement is in the human listener, not the talking machine. To demand such conditioning of language students is, of course, unthinkable.

The problems posed by text-to-speech conversion are evident. Aside from dealing with the orthographical peculiarities of human language, which in some instances (like English or French) can be daunting, the facility must include at least some simple parsing rules which identify and isolate phonemes—no easy task if the text consists of continuous discourse (see Figure 2). Moreover, if word and sentence intonation is even remotely to resemble that of human speech, there must also be higher-level analysis of vocabulary and structure. Or, as Witten puts it, "In effect, the system must examine the text and understand it before it can generate a realistic reading" (vii). Those who are familiar with the work of Winograd (1972; 1984) and others (Schank 1973; Simmons 1984) with computer analysis of natural language merely as keyboarded text will understand why even the optimists in the field consider the solution of the technological and linguistic problems of voice synthesis to be decades away.

For good reason the foregoing survey makes no mention of computerized speech systems based on waveform digitization. Experimenters and commercial suppliers alike have paid almost no attention to it. Ciarcia (1983, 35) speaks for many other authorities when he remarks that waveform digitization and playback provide by far the greatest fidelity in computerized speech. Still, the experimenter–enthusiasts and the commercial developers have preferred to work with formant synthesis, LPC, or—more recently—text-to-speech. The predilection has nothing to do with the actual digitization of speech. Converting analog signals to digital form is a relatively easy process, and has recently, in combination with the laser, become the foundation of a glamorous commercial venture, digital musical recording. I suspect that for "hackers" speech digitization, the use of the computer as a simple recording and playback device, lacks the mystique and challenge of true voice synthesis: how to employ abstract formulae and highly compressed data to coax a machine to simulate complex human behavior.

Be that as it may, in their explicit expressions of preference and estimates of practicality, experts like Teja, Cater, Veltri, Witten, and Kuecken have cited two criteria: intelligibility and memory demand. The first, intelligibility, highlights the difference between two groups interested in computerized speech: teachers and learners of language, who have had very little effect on the development of computers; and—everybody else—the ordinary users of language, including those who create and market computers. The latter, understandably enough, seek economy and flexibility. For them, voice synthesis, especially LPC or text-to-speech, is attractive since, as Teja puts it bluntly, not high linguistic accuracy but rather mere "Intelligibility is all you are after anyway" (119). Witten voices the matter in even clearer terms:

> Even with 'ordinary' cooperative friendly conversation, the need to find words and somehow fit them into an overall utterance produces great diversity of prosodic structures. Applications for speech output from computers do not, however, call for spontaneous conversation, but for controlled delivery, which is like that when reading aloud . . . . Unfortunately for us, linguists are (quite rightly) primarily interested in living, spontaneous speech rather than pre-prepared readings." (38–9)

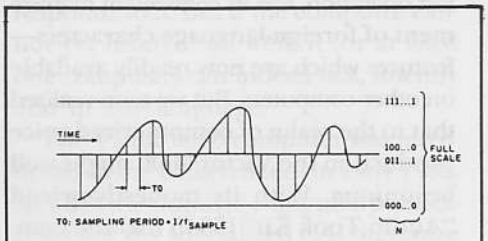


Figure 3a: *Illustration of waveform digitization at two sampling rates. Source: (Ciarcia 1978, 144).*

Indeed we are.

Of course no user of computerized speech prefers false accent and intonation over natural-sounding speech. But until recently it has appeared that the direct recording and playback of high-fidelity digitized speech in any but the smallest quantities—a few words, a few seconds of speech—was impractical or, with small computers, impossible. Thus Teja, in a work published in 1981, expands his assertion about intelligibility: "For high fidelity you won't want to [can't?] use a personal computer. You'll be working with at least a 16-bit microprocessor" (119). Soon thereafter, of course, 16-bit personal computers like the Victor 9000 did appear, though they are still by no means common. Witten, in his 1982 book, emphasized the elaborate recording facilities and inordinate processing time which high-fidelity speech digitization would seem to require (19-20).

Other researchers, writing when 48K or 64K of internal memory were considered capacious and quick-access disk storage was not yet common, observed that formant synthesis and LPC permitted the reasonably intelligible utterance of all sounds, not just prerecorded ones, and that both techniques required far less memory than digitized recording and playback. Thus Cater (1983) provides a chart showing that a 48K computer might handle up to 64 minutes of speech created by phonetic synthesis, over five minutes of LPC-created speech, but only 3.2 to 24 seconds of digitized speech (82). And indeed, as even early work showed, high-fidelity speech digitization was costly in terms of

memory; 2K per syllable was and is a reasonable estimate and, it might once have seemed, an extravagant price.

The early, sometimes wistful conclusion that digitized speech demanded inordinate memory assumed storage in what is now regarded as a small RAM. Only seldom was there mention of storage of sound libraries on disk. Thus Teja (1981, 127) alluded to a speech processor and specialized software which would permit storage of about 120 seconds of speech on a single diskette. Witten (1982) simply notes that "the cost of storage is dropping so fast that this is not necessarily an overriding factor" (18) working against digitized recording and direct playback.

This article would be pointless had not technological advances greatly expanded computer memory capacity and decreased operating time, and had not some language teachers a healthy appreciation of what it means to have readily available even a few seconds of realistic computerized speech. The native user of a language needs such speech simply for communication with other fluent users; the learner of a language needs targeted practice, and thus works within a much smaller linguistic space. Were we actually limited to a few seconds of genuine speech in readily accessible memory, we could still do much with it, for even a small capacity, divided among many individual segments, can yield a wealth of varied recombinations. Each software program, too, might employ its own collection of speech elements.

In short, one might well gain the impression that waveform digitization and playback, especially if the speech could be stored in external memory, would be the speech computerization technique best suited for CALI, but the designers and marketers of small computers and specialized voice systems, developing their products for the language user rather than the language learner, have had little interest in anything but formant synthesis, LPC, and text-to-speech converters. As I prepared the text of the present article I was therefore most heartened to read, in the September 1985 issue of this journal, Richard Barrutia's remarks about the desirability of digitized speech for CALI.

The Victor 9000 Audio Tool Kit uses an advanced form of speech digitization known as "Continuously Variable Slope Delta" (CVSD) modulation, or simply "delta" modulation. As in all such digitization, the sound wave signal to be recorded is sampled at regular intervals (see Figure 3a). The amplitude and frequency of each sample are recorded or "encoded" in digital (binary) form, along with information which makes it possible to store and retrieve in proper sequence the codes of the many samples which are taken from a single sound. For playback, a digital-to-analog converter transforms the digitized information back into an electrical signal which can then be passed to a loudspeaker.

As one might expect, the higher the sampling rate is, the more accurate the sample will be. According to the Nyquist sampling theorem, for decent fidelity the wave should be sampled at a rate equal to twice the highest frequency anticipated. For human speech, a sampling rate of several thousand times a second (6 to 8 kHz) is advisable. In terms commonly used to describe the memory capacity of small computers, each second of sound would require several kilobytes of storage; thus a 64K computer would accommodate only a few seconds of sound in RAM at any time (see Ciarcia 1983, 36). Storage on disk would of course provide considerably greater capacity, perhaps up to a few minutes of solid sound per disk.

Several ways to compress or simplify the digital data without greatly compromising fidelity have been devised. One such is "delta" modulation (see Ciarcia 1983; Cater 1983, 94–100). In the case of human speech it can be assumed that in such a short interval as a millisecond there will be no great change in amplitude or frequency. Thus not absolute readings or many bits in length are recorded for each sample, but rather a very few bits indicating the small differences ("delta") in level between one sample and its predecessor (Figure 3b). The saving in memory may multiply the capacity of the system considerably. Such technical information is not a manner of ordinary concern to the user of the Victor 9000 and its Audio Tool Kit, since the facility operates with simple English inquiries about sampling, recording, editing, and storing.

The Audio Tool Kit consists of a small conventional microphone, a microphone jack, a disk of application files, and an integrated circuit board to be installed in the main unit. Standard features on the Victor 9000 are not only a small speaker but also—here the Victor appears to be unique—a CVSD coder/decoder or "CODEC." Thus speech programs created with the Audio Tool Kit can be run on any Victor 9000.

To begin a recording session, the user enters the command "VOICE," and then specifies the disk drive for storage of sound libraries. Thereafter recording, editing, and filing are accomplished by menu selections made with dedicated cursor and function keys. No knowledge of programming or electronics is necessary. Individual recorded sound segments are termed "keys," while collections of keys are "files" or "libraries"; their capacities will be discussed below.

The basic ATK working screen consists of a brief filename header, a large display of highlighted horizontal bars upon which the sound will be represented, and at the bottom various menu items assigned to dedicated function keys. The user can and should use the "CONTROL" option to select a speech sampling rate, which will then be maintained automatically throughout the voice file. Sampling rates, which represent approximately the maximum length in seconds of the individual speech segment or "key," vary from 1 to 240. The default rate of 32 produces a fuzzy, choppy reproduction unacceptable for CALI. A rate of 8 yields sound whose fidelity compares favorably with

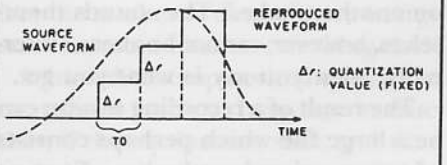


Figure 3b: *Waveform sampling by delta modulation. Each sample of the source waveform is tested to see if its amplitude is higher or lower (within the resolution of a fixed quantization value Δr—delta-r) than that of the previous sample. If the amplitude is higher, the single-bit delta-modulated encoding value is set to 1; if lower, the encoding value is set to 0.*

a cassette recording of decent quality; 8 seconds are enough time to recite, at moderate pace, the Gettysburg Address up to the word "proposition" (34 words or 41 syllables plus rhetorical pauses).

The selection of the menu-option "SOUND" produces another menu, which includes the item "RECORD." That choice, when selected, displays the standard screen with a header and the prompt: "space bar to begin, < A > to abort" (see Figure 4). Speech uttered into the microphone will be recorded in temporary memory until the RETURN key is pressed or the screen is filled. As the user speaks, patterns of dots fill the highlighted horizontal bars, of which there are twenty on the scrollable screen. With practice one can associate patterns with sounds. The present writer has no idea what system of analog correspondence is used, except that the pattern is not that of a standard voice spectrogram. But it at least furnishes linear reference marks which make editing of the raw speech segment very convenient.

The sound may be audited by the menu-item "PLAY." An unsatisfactory "take" can either be erased or recorded over. If the segment is deemed satisfactory, other menu options permit it to be stored on disk with a keyname solicited by a keyboard prompt. Usually, however, some editing is desirable. The cursors which mark the start and end of the active sound field can be moved to clip off unnecessary leading or trailing silence and thus save memory. They can also be used to divide the segment into smaller parts which can then be named and stored as separate keys to be recombined later in some other sequence, or perhaps with keys recorded on another "take." The sounds themselves, however, cannot be altered internally; what you say is what you get.

The result of a recording session can be a large file which perhaps consists of many rather lengthy keys. Storage of the full sound field of a key requires 51.3K of memory, or more than 6K per second at the sampling rate of 8. The size of a file is limited by the capacity of a single diskette, which for the Victor 9000 is 620K per side. A single program run from one disk may, however, call many files from either disk. Thus, even allowing for the memory required by the operating system (100K), BASIC interpreter (53K), and the program itself (typically 2-4K), one might have available for retrieval, repetition, and varied combination more than two megabytes of voice files, or at the sampling rate of 8, about 300 seconds of raw speech material—perhaps a thousand different words of various lengths.

Once the desired speech segments are created and filed, they can be summoned for playback within programs written in assembly language, Pascal, or—as demonstrated here—BASIC. That is, the sounds are equivalent to the strings of linguistic material in conventional CALI programs, and thus can be used similarly in, for example, branching, loops, and subroutines.

Essential to any such program is a preliminary section which initializes the coder/decoder or "CODEC" in its various functions (see Figure 5, lines 200-370). Since the initialization is invariable in general form, its details are of no practical concern to the ordinary teacher-programmer or, much less, the language student. This writer simply constructs all his voice programs around a standard skeleton program.

Before sounds may be played during a program run, the file or "library" which contains them must be named as a string and then loaded into RAM. Thus, in line 410 of Figure 5 the assembly language statement "CALL CODEC (FILE.NAME$, LOAD.LIB%)" instructs the coder/decoder to load the voice file or library named in line 400. The LOAD.LIB and FREE.LIB (line 1990) functions permit one to use more than one set of voice segments within a single program, as long as the desired libraries are present on one of the disk drives. The feature makes for greater flexibility in branching within a single program, and permits one to construct many different programs which draw from a collection of nuclear sound libraries. Loading is automatic and requires little time: freeing is almost instantaneous.

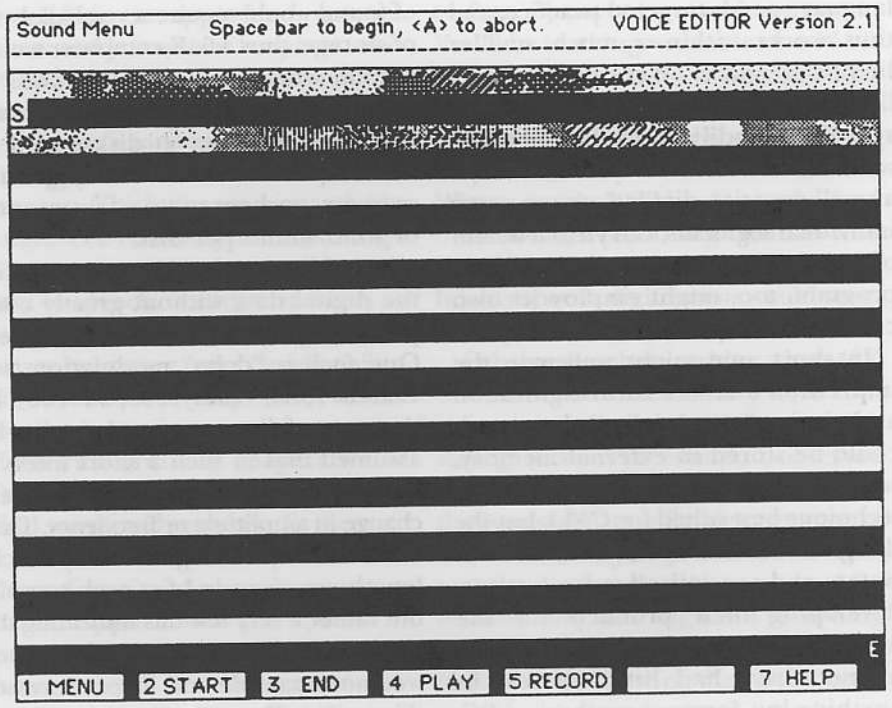


Figure 4: *Victor 9000 Audio Tool Kit screen display in recording mode. The dotted patterns in the first two white fields represent a speech segment ready for editing and saving to disk. "S" and "E" mark variable start and end cursor positions. Scrolling permits a total of twenty lines of sound. The bottom line maps function keys.*

The student user, of course, remains oblivious to such operations.

The heart of ATK voice programs is the statement "CALL CODEC," combined with an actual string name for a sound key or a related string variable, plus the function-specification SAY% (Figure 5: lines 575, 720, 726, and 740). The command has the same effect aurally that the conventional PRINT statement has visually. The string which the SAY% function command operates may be either a literal or a variable. If it is a literal, somewhere in the program it must be declared equal to a keyname (Figure 5, line 100). If the string is a variable, then simple READ and DATA statements specify which keyname is to become the present variable (Figure 5: lines 575, 640, 726, and 1610–1910). That is, indeed, usually the better practice, for it permits the creation of loops which can present item after item. String-name and key-name do not have to be identical; while longer, more descriptive keynames help one keep track of what has been recorded, short string names make for easier, more flexible programming.

The "Audio Tool Kit" has certain limitations, insufficiencies which occasion a consideration of recording, programming, and pedagogical strategies whose validity and usefulness are not confined to the Victor 9000 speech facility. An inherent weakness of all present computerized voice systems is encountered in concatenation, the combination of smaller speech units into longer utterances. In formant or phoneme synthesis the problem becomes evident at a very low level. For even crude verisimilitude syllables and words must be constructed by a tedious coding procedure which for the basic phonemes specifies variants (allophones) to suit the surrounding phonemes, and then adjusts volume and pitch.

Newer text-to-speech converters offer far easier coding, of course; but, as I noted earlier, they still lack the realism necessary for CALI. Where entire words are stored individually on chips, their combination into phrases is likely also to be unsatisfactory. Sustained simulation of genuine diction is virtually impossible unless vocabulary and structures are drastically limited and variations of the same word are recorded separately, as for example in devices which recite telephone numbers. The same would be true of the Victor digitized speech if sound-keys were limited to a fixed stock of single phonemes, syllables, or words. That, however, is not the case.

In my early programs I was concerned not to squander memory. I was also intrigued by the prospect of constructing, from a small stock of elements, a wealth of utterances which had not been spoken as such before. I therefore attempted concatenation at too low a level. Thus the single word-syllable *drei* was supposed to do multiple duty as a number in itself and as part of *dreizehn, drei und zwanzig*, etc. That and similar ventures aiming at even longer concatenations failed. Over the many record ing "takes" I could not maintain sufficiently even pitch, timing, and volume. Therefore, my several programs which involve numbers now use a library in which each number, at least from 0 to 100, is stored as a separate key.

The *Rechnung* program, whose pedagogical motivations will be discussed below, illustrates more advanced concatenation strategies. I could have created for it a special library in which each number-key also included the currency unit *Mark*. I chose instead to make *Mark* a separate element, since I wanted to be able to change the currency unit and also to create the fractional prices (e.g., *zwei Mark achtzig*) from the same library of numbers. In that and similar programs I found it advisable to record, on one "take," in one breath, and at level pitch and amplitude, patterns such as *eine, . . . zwei, . . . drei, . . . vier, . . . funf Mark zehn, . . . zwanzig*." I then saved each number and the central word *Mark* as separate keys. The many recombinations with *Mark* placed between unit and fractional prices sound adequately natural.

It should be observed that a human speaker who in a real conversation utters a compound phrase like *dreizehn Mark funfzig* may very well pause between the elements, or may even change pace and pitch as mental calculation is conducted and as body-language transactions (shifts of glance, etc.) between speaker and listener occur. Modulation is even more evident in larger contexts, a point whose pedagogical implications will be explored shortly.

Two observations followed. Programs like *Rechnung* show that the illusion of complex continuous speech can be created by careful management of context and pacing. It is advisable, therefore, to select conversational contexts in which human speakers might well build up longer utterances by a process of measured concatenation that include considerable modulation of pace, pitch, and emotion. Happily, it turns out that the types of conversation typical of the lower proficiency levels—simple financial transactions, requests for basic information, etc.—are often just such interchanges.

Secondly, one gains a heightened appreciation of skill in elocution. Thus when a colleague, Sandra Rosengrant, and I created a preliminary Russian version of *Rechnung*, we felt fortunate to work with a native speaker who could maintain, from take to take, extremely consistent pitch, pacing, and amplitude and who, nevertheless, did not produce artificial, excessively accentuated speech. As Professor Rosengrant thereupon observed, and as I, a teacher of German, should have recalled, some cultures emphasize more than others the cultivation of the voice as a part of ordinary education.

The language teacher skeptical about computers might ask simply, Why do all this? The proficiency-oriented teacher favorably disposed to CALI might inquire instead, How would it contribute to acquiring language competence? Still better are other questions. Why should the computer *talk*? Why should *the computer* talk? Can the computer now do something which the language teacher cannot do as well or as easily? Can it free the teacher to devote more energy and classroom time to something else?[3]

I think that computerized speech can contribute much to language instruction, if the equipment, skill, and time to develop the resource are available. My particular thesis is that Victor 9000-style voice reproduction programs might be very useful in encouraging and evaluating listening proficiency at the lower levels, from ACTFL/ETS Novice-Low to Intermediate-High. With care in design such materials could also be created for higher levels, perhaps up

to Advanced-Plus. In ordinary language, the equipment and procedures described here would be of value at levels ranging from the recognition of common words and phrases to that of comprehending the subtleties of extended discourse.

Encouraging and evaluating listening proficiency has been a problematic area in language instruction. The grammar-translation approach ignores it, Total Physical Response treats it as a way station on the path to speaking, and until lately, the proficiency experts were much more occupied with oral proficiency than with skill in the other modalities. Several problems are involved even when listening skills are explicitly addressed. We are learning that reception and production modalities should not be confused. Listening proficiency cannot be checked by written dictation exercises or, at least not directly, by an oral proficiency interview. We are coming to understand the methodological dangers attendant on improper input—for example, the use in oral form of material originally intended to be read, or of speech samples whose features are not indexed to proficiency levels and which, when used for teaching, do not present Krashen's "i + 1" language.[4] We are also beginning to realize the consequences of inattention to the vital low levels of proficiency, where to the teacher the target language seems so easy to understand and so boring to exercise. A society replete with commercial audio products underscores the inadequacy of sound resources in the typical language program, specifically the poor supply or inept use of recorded material or, at worst, the inadvertent or willful limitation of speech to that produced by a single teacher using nonrealistic pace and intonation. And—this the last and saddest point of all—we must reckon with just plain teacher fatigue. One can talk only so long to only so many students.

I will illustrate my several points with some programs which are still quite primitive and without lines of others not yet carried out. For three reasons many of my programs concentrate on comprehension of numbers. My own experience bears out the ACTFL/ETS *Provisional Proficiency Guidelines'* implication that many introductory courses skip too quickly over such supposedly trivial features as numbers, colors, dates, sizes, and spatial directions. It is surprising how many students cannot reliably comprehend even simple numbers if they are incorporated into larger utterances rather than presented in isolation. But evidence of comprehension of numbers can also be used to evaluate more subtle comprehension. To know which numbers are significant, and how they fit together, the student must understand the context, often in considerable detail. Moreover, CALI programs which limit student input to simple keyboarding avoid unintentional input errors which, though linguistically inconsequential, may be interpreted as serious errors. Such programs also preclude the need for the elaborate parsing routines necessary for the management of extended discourse. Thus they enable one to concentrate on measuring listening proficiency rather than skill in some other modality.

The *Rechnung* program, mentioned above in the discussion of recording strategy, drills and scores listening comprehension of low-level numbers within a realistic, commonly-encountered context: the presentation of a restaurant bill.[5] The pseudo-randomly generated prices are presented within larger utterances. The first field, an introductory remark of the kind often heard from waiters, may be any of four segments: *Das ware* . . . ("That would be . . . "), *Das macht zusammen* . . . ("All together that will be . . . "), *Also* . . . ("So . . . "), or a "dummy" segment of recorded silence, so that the student cannot count on encountering the price only after something else is said. The second field, as the program presently stands, contains a number which may be any integer between 1 and 30, or any of the even tens between 30 and 100 (37 items in all). The third field is the currency unit, *Mark*. The fourth field consists of a concluding remark: *bitte* ("please"), *Hats geschmeckt?* ("dit taste good?"), *Zusammen oder getrennt?* ("All together or separate checks?"), or the "dummy." Presently the program data will produce 592 ($4^4 37^4 1^4 4$) different items. Thus a few seconds of random-access disk-stored speech actually yield what would, on tape, require lengthy footage, most likely played only sequentially. Lines 900 on permit the student to rerun the program; unlike taped material, which must be rewound for simple replay, the exercise begins again immediately, and the sequence of items is changed (lines 1100–1520).

Expansion of either the number of items in each field, or the number of fields in the pattern, would vastly multiply the number of unique utterances readily available, while requiring very little additional memory. Thus the enlargement of the number set to include all integers between 1 and 100 and the addition after *Mark* of another field which would include the even tens and the "dummy" would yield a more sophisticated and very much larger collection of fractional-unit prices—16,000 ($4^4 100^4 1^4 10^4 4$) different items would be available. At two seconds per item, that would be about 10 hours of speech—not that any human student could tolerate that much at one sitting, of course, but neither do we ask our students to listen to the full battery of a tape set without a break. We merely hope that our technological aids will provide resources which will improve the student's linguistic competence without placing undue strain on the human instructor.

With more elaborate but not particularly difficult programming and the addition of a small supplementary sound library, the *Rechnung* program might be expanded to include subroutines that would present what the proficiency experts term a "situation with complication," a ploy often used to check performance at the Intermediate level. Every so often, especially if the student's score showed good comprehension of simpler utterances, the cybernetic speaker might pause, sound flustered, and then announce a miscalculation: "Let's see now, that's eighteen—no, wait a second, I goofed—nineteen *Marks* sixty-five. Sorry." Or the totaling of the bill might include an itemization of dishes and beverages and their prices, with a request for verification: *Stimmt das/Gelt?* ("zat right?"), and a direction to the student to press Y or N. Perhaps the program might request the student to tender an appropriate currency note, after which a discussion of change returned might ensue ("A hundred? Do you have anything smaller?"). A useful elaboration in any case would be variable

scoring which would weigh the responses according to the difficulty of the items presented and then yield a score which might be indexed to a momentary proficiency estimate or "level check." Thereupon the program might continue, with subsequent items selected accordingly, either to confirm the rating or to offer exercises adequately challenging rather than too easy or too difficult for the individual student.

Essentially the same sound library and programming strategy might exercise aural comprehension of numbers in other realistic situations. Obvious "contexts" are weather reports, broadcast schedules, or railroad-station announcements, including perhaps complications involving delays or changes in services. Similar programs, requiring scarcely more keyboard sophistication, could address other important low-level elements like spatial directions, colors, days of the week, clothing items, or even spelling. A perusal of some of the better conventional taped materials will show how exciting such material can be made, and will suggest as well how such exercises could be improved by the computer's facility for interactive feedback and branching.

I confess to a healthy bias against CALI software which debases language learning into artificial games like "Hangperson." Nevertheless, I can appreciate serious games which include an element of practical communication. That is, it should be possible to create speech programs which have as their context the games actually played in the target culture. I am presently designing, for use initially in French, a voice program called "Roulette." The student would be one of the gamblers; voices of the croupier, other gamblers, and a curious onlooker or "kibitzer" would be provided by the computer. Bets would be solicited and announced, and a randomizing feature would spin the wheel, with appropriate sound effects. From time to time the curious onlooker would intervene to quiz the student—aurally of course—about the progress of the game (*M. Blanc, combien a-t-il gagne cette fois?*). Correct answers would add to the student's score and perhaps increase his wagering capital; the program might even propose side-bets to

```
REM program name "rechnung," VOICE program (draft)
15 REM by William B. Fischer, Portalnd State University, Portland, OR
16 REM begun 9 July 1985
18 REM
20 REM screen instructions to student (outline)
25 REM
30 PRINT "Rechnung: An Experimental VOICE Program" : "PRINT
35 PRINT "for Drilling and Testing Listening Comprehension" : PRINT
40 PRINT "of numbers in German" : PRINT : PRINT
50 PRINT "Instructions, etc.—hear number, enter digit, RETURN"
80 REM
90 REM declaration of sound keys as BASIC strings (if not variables)
100 MARK$ = "MARK"
110 REM
200 REM : initialize the CODEC
210 PLAY.NOTE% = 1
212 RECORD.SND% = 2
214 PLAY.SND% = 3
220 LOAD.LIB% = 4
230 SAY% = 5
240 FREE.LIB% = 6
250 PAUSE% = 7
260 PAUSE.LEN% = 500
300 DEF SEG = 0
310 LOWOFF = PEEK (880)
320 HIOFF = PEEK (881)
330 LOWSEG = PEEK (883)
340 HISEG = PEEK (883)
350 CODECSEG = (256 * HISEG) + LOWSEG
360 CODEC = (256 * HIOFF) + LOWOFF
370 DEF SEG = CODECSEG
390 REM
395 REM load sound library
397 REM
400 FILE.NAME$ = "ZAHLENDM"
410 CALL CODEC (FILE.NAME$, LOAD.LIB%)
430 REM
440 REM Main loop (derived from Kemeny)
460 RESTORE
500 S = O : REM sets score at zilch
505 REM responses to user input
510 L$ (O) = "TRY AGAIN"
520 L$ (1) = "THE CORRECT ANSWER IS:"
530 L$ (2) = L$ (1)
545 PRINT:PRINT
550 INPUT "How many items do you want?",D
560 FOR X = 1 to D
565 RESTORE 1100
570 GOSUB 1100
575 FOR F = 1 TO R : READ F$ : NEXT F : CALL CODEC (F$,SAY%)
580 REM creates front string and says first phrase (or dummy)
610 RESTORE 1695 : REM skips to number data
620 GOSUB 1500 : Z = R : REM FETCHES RANDOM NUMBER R—for Z see subroutine
630 FOR Y = 1 TO R : REM reads R pairs for items and answers
635 REM adn retains the R-th pair as the current pair
640 READ Q$,A
650 NEXT Y
710 T = O : REM sets number of tries at zilch
715 PRINT Q$«A« "Will be removed in final version."
716 REM above line for programmer's convenience in checking data
719 REM next few lines utter price phrase
720 CALL CODEC (Q$« SAY%) : CALL CODEC (MARK$, SAY%)
722 RESTORE 1900 : REM skips to end string data
724 GOSUB 1100
726 FOR E = 1 TO R : READ 3$ : NEXT E : CALL CODEC (E$, SAY%)
728 REM creates back string and says end phrase (or dummy)
730 INPUT B : REM elicits user guess
735 GOTO 750
740 CALL CODEC(Q$, SAY%) : CALL CODEC (MARK$, SAY%) : INPUT B
750 if A = B THEN 820
760 PRINT L$(T);
770 T = T + 1
780 IF T = 1 THEN 740
790 PRINT A
800 T = 2
810 GOTO 740
820 S = S + 2·T
830 PRINT
840 NEXT X
850 S1 = S/ (2*D)
860 PRINT "Your score is"; INT(S1*100 + .5); "%"
900 INPUT "Do you wish to repeat the exercise? Enter Y or N.", Y$
910 IF LEFT$ (Y$,1) = "Y" THEN 440
920 IF LEFT$(Y$,1) = "Y" THEN 440
1000 GOTO 1990
1100 REM subroutine to generate integer random number between 1 and 4
1105 RANDOMIZE (D)
1110 R = INT(RND*4 + 1)
1120 RETURN
1500 REM subroutine to generate integer random number between 1 and 22
1502 P = D + 1 : REM If user repeats drill, increments P to reseed randomizer
1505 RANDOMIZE (P)
1510 R = INT(RND*38 + 1) : REM 22 + 1 for 22 data pairs, etc.
1520 IF R + Z THEN 1510 : REM avoids immediate repetition of same item
1590 RETURN
1600 REM front strint data
1610 DATA daswaere, dasmachtzus, alsopause, dummy
1690 REM string data for prices—can be caps or lower case
1695 DATA ZIILCH, O
1697 DATA ONE,1, TWO,2 THREE,3, FOUR,4, FIVE,5
1700 DATA SIX,6, SEVEN,7, EIGHT,8, NINE,9, TEN,10
1710 DATA ELEVEN,11, TWELVE, 12, THIRTEEN,13, FOURTEEN,14, FIFTEEN,15
1720 DATA SIXTEEN,16, SEVENTEEN,17, EIGHTEEN,18, NINETEEN,19, TWENTY,20
1730 DATA TWENTYONE, 21, twentytwo,22, twentythree,23, twentyfour,24, twentyfive, 25
1740 DATA twentysix,26, twentyseven,27, twentyeight,28, twentynine,29, thrity,30
1800 DATA forty,40, fifty,50, sixty,60, seventy,70, eighty,80, ninety,90, hundred,100
1900 REM end-string data
1910 DATA bitte, geschmeckt, zusgetrennt, dummy
1990 CALL CODEC (FREE.LIB%)
2000 END
```

Figure 5: *The author's experimental voice program* Rechnung, *which drills listening comprehension of german numbers within the context of a waiter's presentaiton of a restaurant bill.*

check how confident the student was of her comprehension.

One soon envisions many other programs built around realistic situations, for example money-changing or arrange ments for travel. Appealing also are decorative but effective subroutines which might use the student's own prerecorded name or politely issue answer-prompts. Such features, and the main programs just described, have their cousins in conventional software now readily available, and thus may seem trivial or not very innovative. And indeed it is virtually pointless, say, to spell out numbers on the screen in the target language and request the student to enter their digital equivalents. But the effect of the change of modality can be astounding; the *Rechnung* program is not at all easy, even for some students in advanced courses. We are realizing that much traditional language instruction neglects exercise in comprehension of rudimentary information, on the assumption that the student masters such supposed trivialities readily, as is "proved" by the ability to recite and recognize numbers in isolation and to carry out—usually rather laboriously—noncontextual arithmetic calculations. Moreover, systematic exercising and testing of such material in conversational interchange is burdensome to the human teacher. But the careful observer will find that students are surprisingly inept at handling even such "simple" content as numbers and colors within larger contexts.

One might ask, however, what use might be made of convenient digitized speech at higher levels of proficiency instruction and testing. Here I can remark only that much remains to be said, programmed, and heard, and that the convention of student keyboarding of responses is a two-edged sword. Keyboarding of simple responses imposes evident limitations on linguistic creativity, and yet it precludes the illegitimate mixture of modalities. That is, voice programs which demand only simple keyboard responses can for that very reason help one to pinpoint listening proficiency by precluding contamination introduced by lack of ordinary typing skills or compensatory language proficiency in the modality of writing. Here the Victor 9000 speech system poses no great linguistic or technological obstacle, since it permits the presentation, at considerable length, of utterances which in their vocabulary or structure might challenge even the "well-educated native speaker" who represents the ultimate in language proficiency.

An example would be a program that might be called, in German, *Autowerkstatt* (Repair Shop). In a manner reminiscent of the crucial point in an oral proficiency interview situation, the student must deal with an automobile mechanic who discusses repairs. Though the low-level language user might ultimately handle the gross features of the situation, typical of such an interchange would be quite specialized vocabulary ("carburetor adjustments"), high-level structures (careful tense distinction, passive, subjunctive, etc.), and speech-tailoring. The sound library might include, in the target language such statements as on a relatively low level, "Sure, I'll get you the keys," or, on much higher levels, "Yep. We just took it down off of the rack," "Virgil's still tinkering with the ignition," "We were all set to do it but you didn't return our call about the estimate," or "It'll have to be taken out and road-tested one more time."

Here German, the present writer's specialty, is especially difficult for the English-speaking student. While English uses the auxiliary "to be" in passive construction which are both "genuine" and "statal" or "false," German insists on the distinction of the auxiliaries *werden* and *sein*. Moreover, in German the present tense commonly has future sense. Thus *Das Auto ist repariert* means "The car is fixed" (ready to go), while *Das Auto wird repariert* can mean "The car is being fixed," "The car is fixed [whenever or by whomever fixing takes place]," or even "The car will be [is being?] fixed" (but NOT " . . . will be in a state of good repair"). The linguistic points, some might say, are subtle, but in real situations the differences they represent are enormous—certainly well into the ACTFL/ETS "Advanced" or even "Superior" range. Nevertheless, a voice program could still evaluate the student's comprehension of the mechanic's naively complex language by posing a simple choice which requires minimal keyboard ing skill: "Is the car ready to go?" Type "Y" for YES, "N" for NO.

Working with computerized speech for CALI on the Victor 9000 is both an exhilarating and a frustrating experience. The facility I have described here is admirably suited to CALI. It could provide a convenient and effective means of exercising and evaluating proficiency in listening, a modality which has often been slighted. It can do that at levels which—as is so often ignored—are either difficult or onerous to manage. In short, the Victor 9000 offers resources which might help fill a huge gap in language instruction.

Yet at the same time one wistfully contemplates several technical and economic obstacles. The ATK is proprietary, and the Victor 9000 is—to say the least—not a common computer. While voice synthesizers are now readily available for many popular computers, voice synthesis is not presently suitable for CALI. Facilities offering the convenient speech digitization and direct reproduction of sound necessary for the production of CALI materials are not easily to be had. While textbook publishers seem interested in the concept of interactive audio programs, and have even begun to offer software which interfaces computers and cassettes, marketing considerations strongly favor the development of such supposedly ancillary materials only as parts of larger textbook packages, with software designed to run on the Apple IIe, of course, and perhaps on the TRS-80, Commodore, IBM, or—recently—Macintosh. No one seems interested in developing, for its own sake, digitized-voice CALI equipment and software.

There seems to be no solution to the quandary, unless a language department which possessed, or which contemplated, acquiring a large number of computers decided that the same sum that would purchase, say, 24 Apple IIes might better be spent on 18 Apple IIes and two or three Victors. Suitably qualified staff might then be assigned to create voice programs until such time that an outside supplier might furnish ready-made programs.

Meanwhile, time and technology march on. It is conceivable that the Victor 9000 sound system—as well as cassette interfacing—will shortly be

made obsolete by other computers or by interactive video systems which also support audio, although problems exist with the presently primitive versions of such facilities. Nevertheless, a convenient speech digitizer and authoring program like that of the Victor 9000, and the experience that teachers could gain from using it, would still be useful in CALI. The exercise of listening skills without the aid of visual cues is likely beneficial in itself, even were it only a preparation for genuine use of language, when such cues are usually present. Secondly, even or especially at the lower levels of proficiency much language listening in genuine contexts does takes place under aural constraints: telephone conversations, radio broadcasts, etc. Thirdly, many of the subtle pedagogical strategies and recording techniques developed with the Victor 9000 would still be valid with interactive video-audio.

But lastly, in an awkward period when computer hardware and software come and go, and when those interested in CALI often find themselves neglected in a market which considers their intersts and needs to be odd and sees little promise in terms of sales, the existence of the Victor 9000 and its ATK provides a moral bargaining chip, one which should be exploited by our profession. It gives us a benchmark, a notion of what we should demand of the suppliers—and of ourselves—when we contemplate how the computer can, or might, or should serve our purposes as teachers of language for practical proficiency.

ENDNOTES

1. For descriptions of two new systems which offer at least some interactive audio facility see John Lawler et al. 1985. Interactive audio in a videodisc system. *BYTE* 10 (June):108–117, and Harry S. Wohlert and Martin McCormick. 1985. An algorithm for controlled integration of sound and text. *CALICO Journal* 3 (December): 9–21, 37. Of interest too are the several Tandberg products and the CAX-50 interace for Sony Series 5000 cassette recorders.
2. For a detailed product review see Jerry Willis and Merl Miller. 1984. *Computers for Everybody: 1985 Buyer's Guide.* Beaverton: Dilithium Press. (pp. 412–17). Technical data: Intel 8088 16-bit CPU operating at 5MHz; 128K internal memory, expandable to 768K; MS-DOS and CP/M (mutually readable); 12-inch green screen with 25-line by 80-character or 50-line by 132-character text display (monochrome graphics displayed at 800 by 400 pixels); 103-unit standard keyboard (10 programmable function keys, all keys reconfigurable, character sets "soft-loading"); two disk drives standard (each 600K single-sided or 1.2M double-sided); built-in coder/decoder (CODEC) with 6852-SSDA chip as bit-stream interface. Our department purchased its 256K unit in July 1983, with educators' discount, for $4145.85 (list price $4745). The current list price for the same unit is $3295. The corporation underwent financial reorganization and has now added to its product line an IBM-compatible portable version, the "VI" or "Vicky" (with no CODEC), a straight IBM-clone ("VPC"), an AT-clone, and a retrofit kit which makes the Victor 9000 also IBM-compatible or which, as some Victor users put it, makes it possible to "downgrade" a Victor into an IBM.
3. Two articles in **CALICO** have addressed with particular eloquence and insight the proper roles and functions of the computer and the human teacher in language programs: Robert L. Blomeyer, Jr. 1984. Computer-based foreign language instruction in Illinois schools. *CALICO Journal* 1 (March):35–44, and Rex C. Dahl and Paul F. Luckau. 1985. VIDEO-DEUTSCH: A computer-assisted approach to verbal and nonverbal cultural literacy. *CALICO Journal* 2 (June):13–19.
4. See Theodore V. Higgs. 1985. The input hypothesis: An inside look. *Foreign Language Annals* 18 (May):197–203.
5. The core of the *Rechnung* program (much of lines 500–860) was lifted from a string-matching routine written by John G. Kemeny, one of the creators of BASIC; see Daniel L. and Joan K. Slotnick. 1973. *Computers: Their Structure, Use, and Influence.* Englewood Cliffs: Prentice-Hall. (pp. 127–28).

BIBLIOGRAPHY

Ariew, R. 1984. *Utilisons l'ordinateur.* Software for first-year French text *Allons-y.* Boston: Heinle and Heinle.

Baker, R. L. 1984. An experience with voice-based learning. *CALICO Journal* 1 (March):17–19.

Barrutia, R. 1985. Communicative CALL with artificial intelligence: Some desiderate. *CALICO Journal* 3 (September):37–41.

Bristow, G., ed. 1984. *Electronic Speech Synthesis: Techniques, Technology and Applications.* New York: McGraw-Hill.

Cater, J. P. 1983. *Electronically Speaking: Computer Speech Generation.* Indianapolis: Howard W. Sams and Co.

Ciarcia, S. 1984. Build a third-generation phonetic speech synthesizer. *BYTE* 9 (March):28–40, 42.

Ciarcia, S. 1983. Use ADPCM for highly intelligible speech synthesis. *BYTE* 8 (June):35–44, 46, 48–9.

Ciarcia, S. 1978. Talk to me! Add a voice to your computer for $35. *BYTE* 3 (June):142–51.

Dixon, N. R., and Martin, T. B., eds. 1979. *Automatic Speech and Speaker Recognition.* New York: IEEE Press/John Wiley and Sons.

Flanagan, J. L., and Rabiner, L. R., eds. 1973. *Speech Synthesis.* Stroudsburg: Dowden, Hutchingson, and Ross.

Hirsch, B. 1985. Review of *French Achievement* software. *Modern Language Journal* 69 (Summer): 199–200.

Holmes, J. N. 1972. *Speech Syntheses.* London: Mills and Boon.

James, C. J. 1984. Experimenting with voice recognition in German. *Unterrichtspraxis* 17 (Spring):140–141.

Bassein, R., and Underwood, J. 1985. *Juegos Communicativos: Games for Communicative Practice in Spanish.* For Knorre et al. *Puntos de Partida.* 2nd ed. New York: Random House.

Kaplan, G., and Lerner, E. J. 1985. Realism in synthetic speech. *IEEE Spectrum* 22 (April): 32–37.

Kossuth, K. C. 1984. Suggestions for comprehension-based computer-assisted instruction in German. *Unterrichtspraxis* 17 (Spring):109–115.

Kramsch, C. et al. 1985. An overview of the MIT Athena language learning project. *CALICO Journal* 2 (June):31–34.

Kuecken, J. A. 1983. *Talking Computers and Telecommunications.* New York: Van Nostrand Reinhold.

Lerner, E. J. 1982. Products that talk. *IEEE Spectrum* 19 (July):32–37.

Linggard, R. 1985. *Electronic Synthesis of Speech.* Cambridge: Cambridge University Press.

Neudecker, T. 1985. The Mac talks back. *Macworld* (August) :143–44.

Rabiner, L. R., and Schafer, R. W. 1978. *Digital Processing of Speech Signals.* Englewood Cliffs: Prentice-Hall.

Reveaux, A. 1986. Digital sound for the Mac. *A + Magazine* (May):106–113.

Schank, R. C., and Colby, K. M., eds. 1973. *Computer Models of Thought and Language.* San Francisco: Freeman.

Simmons, R. F. 1984. *Computations from the English: A Procedural Logic Approach for Representing and Understanding English Texts.* Englewood Cliffs: Prentice-Hall.

Teja, E. R. 1981. *Teaching Your Computer to Talk.* Blue Ridge Summit: TAB Books.

Veltri, S. J. 1985. *How to Make Your Computer Talk.* New York: McGraw-Hill.

Wagers, W. D. 1984. Voice-based learning. *CALICO Journal* 1 (June):35–38.

Winograd, T. 1984. Computer software for working with language. *Scientific American* 251 (September):130–45

Winograd, T. 1972. *Understanding Natural Language.* New York: Academic Press.

Witten, I. H. 1982. *Principles of Computer Speech.* London: Academic Press.

Wohlert, H. S. 1984. Voice input/output speech technologies for German language learning. *Unterrichtspraxis* 17 (Spring): 76–84.

Wyatt, D. H. 1984. Computer-assisted teaching and testing of reading and listening. *Foreign Language Annals* 17 (September):393–407.

Author's Address

William B. Fischer
Department of Foreign Languages and Literatures
Portland State University
P.O. 751
Portland, OR 97207