DOCUMENT RESUME

ED 345 517 FL 020 212

AUTHOR Dickerson, Wayne B., Ed.; And Others

TITLE Issues and Developments in English and Applied Linguistics (IDEAL), 1987.

INSTITUTION Illinois Univ., Urbana. Div. of English as an International Language.

PUB DATE 87

NOTE 111p.; Issue based on selected presentations at a conference on "Computers in Language Research and Language Learning" (Urbana-Champaign, IL, October 25-26, 1986). For individual papers, see FL 020 213-219.

PUB TYPE Collected Works - Serials (022) -- Collected Works - Conference Proceedings (021)

JOURNAL CIT Issues and Development in English and Applied Linguistics (IDEAL); v2 1987

EDRS PRICE MF01/PC05 Plus Postage.

DESCRIPTORS Academically Gifted; *Computer Assisted Instruction; Computer Oriented Programs; *Computers; Courseware; Data Processing; Diachronic Linguistics; Dictionaries; Educational Trends; Error Analysis (Language); High Schools; Humanities; Language Processing; *Language Research; *Lexicology; Microcomputers; Research Methodology; *Research Tools; Second Language Instruction; Second Language Learning; Speech Synthesizers

IDENTIFIERS *Natural Language

ABSTRACT

Articles in this annual serial include the following: "Four decades with the Computer" (Winfred P. Lehmann); "The Uses of the Personal Computer in the Humanities" (James W. Marchand); "Digging in the Dictionary: Building a Relational Lexicon To Support Natural Language Processing Applications" (Martha Evans, Judith Markowitz, Thomas Ahlswede, Kay Rossi); "Vocoders and Speech Perception: Uses of Computer-Based Speech Analysis-Synthesis in Stimulus Generation" (Joseph Tierney, Molly Mack); "Current Trends in Computer-Based Language Instruction" (Robert S. Hart); "Microcomputer Feedback to the Student/Learner" (Patricia A. Mulligan, Gary G. Bitter); "An Example of the Use of Microcomputers in Foreign Language Learning and Teaching from a High School for the Academically Talented" (Constance Curtin, Stanley Shinall); and "Template Construction as a Basis for Error-Analysis Packages in Language Learning Programs" (Stephen C. Helmreich). (MSE)

IDEAL
IDEAL
IDEAL
IDEAL

Issues and Developments in English and Applied Linguistics

Volume 3
1988

Issues and Developments in English and Applied Linguistics (IDEAL)

PURPOSE AND SCOPE

IDEAL is intended as a forum for research into the acquisition and teaching of English as a second language. Articles, review articles and reviews in any of the following areas are welcome: Teaching English as a second language, second language acquisition, varieties of English, neurolinguistics, sociolinguistics, psycholinguistics, pragmatics, discourse analysis, applications of computer technology to second language teaching and research. It is especially important that contributions of a theoretical nature make explicit the practical implications of the research they report.

SUBSCRIPTION INFORMATION

IDEAL is published once each year in the spring. The subscription rate is $8.00 per year. Subscription and business correspondence should be addressed to:

IDEAL Subscriptions
Division of English as an International Language
3070 Foreign Languages Building
707 South Mathews Avenue
Urbana, Illinois 61801
U.S.A.

INFORMATION FOR CONTRIBUTORS

IDEAL invites authors to submit for consideration articles, review articles, and reviews that fall within the purview of the journal. Manuscripts submitted should conform to the general style sheet given on the inside of the back cover. For more information about the range of topics of interest to *IDEAL* and the editorial practices followed by *IDEAL*, please refer to the Editorial, Volume 1, 1986.

Issues and Developments in English and Applied Linguistics (IDEAL)

Computers in Language Research and Language Learning

Editors

Wayne B. Dickerson (Chair), Lyle F. Bachman, Molly Mack

Editorial Assistant

Lubitsa Nenadovich Katz

Published by

Division of English as an International Language
Intensive English Institute
University of Illinois at Urbana–Champaign

IDEAL
Volume 2
1987

Contents

INTRODUCTION

The computer has had a major impact on the study of language and language learning over the course of the past several decades. It has influenced work in psycholinguistics, neurolinguistics, bilingualism, lexicography, stylistics, speech perception and production, first and second language acquisition, and artificial intelligence. It has also profoundly influenced language teaching; today there are literally hundreds of computer programs designed to improve or enhance the study of orthography, sentence structure, word meaning, comprehension, reading and writing.

In spite of a shared interest in language, and a common interest in the use of the computer to investigate or instruct, language researchers and educators only rarely interact professionally, and it often seems as if members of both groups work in virtual isolation from one another. Furthermore, it is axiomatic that changes wrought by the computer occur rapidly, and the proliferation of information which results from such changes seems, at times, overwhelming.

On October 25 and 26, 1986, the Division of English as an International Language, University of Illinois at Urbana-Champaign, hosted a conference entitled, "Computers in Language Research and Language Learning." The purposes of this conference were (1) to provide a forum for the exchange of pertinent information between language researchers and language educators, and (2) to bring into clearer focus some of the recent computer-based developments in language research and language learning.

The contributions to the present issue of *IDEAL* are based upon selected conference presentations, and aptly reflect the range of topics discussed at that conference. In Part I, "History and Humanism," Winfred Lehmann and James Marchand reflect on both the history of computers and their impact on humanistic studies. Lehmann, one of the world's most notable linguists, provides a stimulating and sometimes wry account of changes in the uses of and attitudes toward the computer in his contribution, "Four Decades With the Computer." In his article, "The Use of the Personal Computer in the Humanities," Marchand describes the emergence of new approaches to the study and presentation of literary texts--approaches which have been made possible by advances in personal computer technology.

Part II, "Language Research," includes two articles that report on computer applications in very distinct areas of language research--lexicography and speech perception. In their article, "Digging in the Dictionary: Building a Relational Lexicon to Support Natural Language Processing Applications," Martha Evens, Judith Markowitz, Thomas Ahlswede and Kay Rossi discuss selected problems in the field of lexicography and, in so doing, reveal how the computer has emerged as an indispensable tool in this area of study. Focusing on the auditory aspects of language, Joseph Tierney and Molly Mack discuss several types of speech analysis-synthesis devices in their article, "Vocoders and Speech Perception: Uses of Computer-Based Speech Analysis-Synthesis in Stimulus Generation." They also consider some of the applications of these devices in speech-perception research.

Part III of this issue, "Language Learning," includes four articles that discuss issues in both our understanding of computer-based language learning (CBLL) and the development of more effective CBLL programs. Robert Hart, in his article, "Current Trends in Computer-Based Language Instruction," provides an overview of the field. He identifies three features that most teachers consider crucial to efficient language learning, yet that are lacking in current CBLL courseware. He then discusses what he considers the most promising approaches to incorporating these features into CBLL materials. The utilization of information about student perceptions of interactions with computers in the development of CBLL curriculum and courseware is discussed by Patricia Mulligan and Gary Bitter in their article, "Microcomputer Feedback and the Student/Learner." In particular, they examine student responses to different characteristics of microcomputer feedback as a means of helping assess the relative importance of these features to student learning. In their article, "An Example of the Use of Microcomputers in Foreign Language Learning and Teaching from a High School for the Academically Talented," Constance Curtin and Stanley Shinall discuss a CBLL program for academically talented students, and the use of observational and test data for discerning how to optimally personalize CBLL applications. In the last article of the issue, "Production Systems as a Basis for Error-Analysis Packages in Language Learning Programs," Stephen Helmreich argues in favor of basing error-analysis programs on models of production, rather than on current parsers, which are modeled on natural language understanding. He then outlines the components of a production-based error-analysis system, and discusses the advantages of such a system over parser-based systems.

It is our conviction that the increased use of computers in language-related research and development, and greater familiarity, among social scientists and humanists, with both the potential applications and the limitations of computer technology will lead not only to greater understanding of current problems and issues, but will also lead us to questions never before asked. We also believe that research and development in the future will be increasingly interdisciplinary, reflecting not only the vitality of language and language learning as areas of inquiry, but also their infinite complexity. It is our hope that the articles included in this volume will serve both to reflect the interrelatedness of current research and to stimulate further study, discussion, and interest.

This special issue of *IDEAL* also marks the 40th anniversary of the Division of English as an International Language at the University of Illinois at Urbana-Champaign. Over the past four decades the Division has witnessed myriad changes--not only with respect to its distinguished faculty and multi-national student body, but with respect to the field as a whole and its orientation to this field. The recent renaming of the Division (formerly the Division of English as a Second Language) reflects what we perceive as a broader range of interests and concerns in the role of the English language in an increasingly interdependent international setting. Today, with nearly 80 graduate students, DEIL is one of the largest graduate programs at the University of Illinois at Urbana-Champaign, and its national and international reputation continues to grow as it explores new avenues of language study and research.

We gratefully acknowledge the following campus units, without whose generous support neither the conference nor this volume would have been possible: the Office of the Vice Chancellor for Academic Affairs, the Office of the Dean of the College of Liberal Arts and Sciences, the School of Humanities, the Center for African Studies, the Office of Instructional and Management Services, the Language Learning Laboratory, the Russian and East European Center, and the Departments of Classics, French, Germanic Languages and Literatures, Linguistics, Psychology, Slavic Languages and Literatures, and Spanish, Italian and Portuguese.

Lyle F. Bachman

Molly Mack

FOUR DECADES WITH THE COMPUTER

Winfred P. Lehmann

When computers first became available, scholars in the humanities made little or no use of them. Language applications, such as machine translation, were introduced by non-linguists. By contrast, numerical applications were broadly pursued; hardware and software were rapidly developed for numerical analysis and applications in the physical sciences. engineering, and the business world.

Only recently have computer languages and special computers been developed for managing human language, accompanied by linguistic activities supported primarily by commercial interest in machine translation. The languages were not designed by linguists, and were framed in accordance with logical conventions rather than natural language structures. While more advantageous than the languages designed for numerical processing, they are less congenial for human communication with computers than natural languages would be. Use of these for communication with computers is a widespread interest.

Continued lack of attention by specialists in language, and increasing pressures for applications, are leading to inadequate products, as for computerized language teaching. If specialists in the humanities do not participate in such activities, these will again be unsuccessful. Possibilities are now opening for imaginative applications, as in the teaching of languages and cultures. Humanists concerning themselves with applications will advance their field, while providing greatly improved teaching capabilities with the help of computerized devices.

The title of my paper might lead humanists to expectations of anecdotes about Lord Byron, more probably about his gifted daughter, Ada Lovelace, reputed to be the first programmer. Actually, the kind of computer I first ran into was not unlike hers, or Babbidge's. During World War II, I had the fortune of belonging to a unit whose activities we were never to talk about subsequently, a unit then named the Second Signal Service Battalion. My wing of that unit was concerned with the translation of Japanese documents. Even at that time the usefulness of computers was recognized, as an incident may indicate.

In the early days of our activities there were no handbooks for the type of Japanese with which we were concerned, a type we came to call the Formal Written

Style. There were no dictionaries for it either. But there were rows and rows of machines and printout after printout of Japanese texts. Some ingenious planner made a run of a large number of texts and printed out the words by their frequency. Equipped with this information, our bank of three teachers made an appointment with their Navy counterparts to discuss plans for cooperation in producing the desired handbooks. The Navy Commander in question was dubious about the value of the computer list. To bring him around, the senior Army instructor said that without the information from the computer one might produce a handbook including words like *hoorensoo*. To which the Navy Commander replied: "*Hoorensoo, hoorensoo*, we see that word in our texts all the time." Since *hoorensoo* is the Japanese word for "spinach," the Army group quickly said their farewells and proceeded to prepare their handbooks independently. Only toward the end of the war, when personnel waiting for discharge had few duties, was liaison established with our Navy counterparts.

Four decades ago and even earlier, then, the mechanical computer was useful for specific purposes in the treatment of language. When the first electronic computer, one foot longer than Beowulf's dragon, replaced its mechanical ancestor toward the end of the war, similar uses were proposed for it. The computer is a symbol-manipulating device. Human beings make use of two elaborate systems of symbols: numbers and language. As a manipulator of symbols the computer might deal with either. In proclaiming the capabilities of the new machine, its major proponents recognized these possibilities. Warren Weaver suggested that among its applications was computerized translation, usually called machine translation.

Few probably need convincing that the pragmatic society of today, for all its praise of pure research, demands applications in providing support for scholarly ventures. Physicists may have gloried in their concentration on uncovering the laws of the universe but, apart from the attention paid to one of them annually by the Nobel Foundation, they occupied offices in dim corners of science buildings before advent of the atomic bomb. However much they may proclaim dissociation from applications involving atomic study, few have maintained idealistic attention to pure research, as the proportion of particle to acoustic physicists may demonstrate. In the human sciences, on the other hand, the lure of study untainted by the promise of practical results lives on, probably at least in part because there are no billions pouring out of Washington.

Unlike physical scientists and such social scientists as economists, linguists and other humanists did not concern themselves with computers as they became available, let alone develop new uses for them. Major figures taking up Warren Weaver's challenge, such as Léon Dostert of Georgetown University and Victor Yngve of MIT, did not emerge from linguistics. They and others who joined them concentrated on machine translation, in a manner much like the earlier use of mechanical computers for dealing with language. Matching allowed the identification of the string *hoorensoo* among numerous other Japanese strings. With slightly different procedures this string could be matched with a designated string (for example *spinach* or *Spinat*) in a different language. Only slowly did the computational specialists concerned with language move from lexical matching to syntactic and semantic analysis. Linguists showed little interest in the action, rather disdain.

By contrast, the users concentrating on numerical symbols made huge advances. For their simpler symbolic system, speed of manipulation was important. Accordingly, hardware was developed so that the symbols concerned could be processed with ever-increasing speed. By contrast, hardware was not being developed for the differing requirements of language and data processing.

Similarly, procedures for access to the increased capabilities were developed. Instead of using machine language--long sequences written in the binary system of zeros and ones--so-called higher-level languages were produced. These came to be used, and developed further, in manipulating numerals for economic and business concerns as well as for study in the natural sciences.

One upgrade succeeded another, in hardware as well as software, until production of the fast mainframes of today with their high-level languages. Expense was no concern. When IBM unveiled its 360 series after a long, costly stage of development, it expended another huge sum on discovering the total inadequacy of its software. The sums expended in the subsequent decades may be gauged by looking at the standing of computer firms in the stock market.

What was happening in the meantime to applications in the other major symbolic system, or to attention to this symbolic system with the help of computers? The groups devoted to such research received small amounts of money, attended by high expectations. It is almost ludicrous to recall the publicity given Dostert's experiment carried out in connection with IBM to illustrate how machine translation might be instituted. A few Russian sentences with a vocabulary of a hundred or so items led *The New York Times* to proclaim success on its first page. And why not? Number crunchers were making good use of computers, and their pedigree was scarcely more venerable than that of linguists. The great intellectual center of Alexandria produced Dionysius Thrax a short time after Euclid, both before our era. Even earlier, Pāṇini was more advanced in his field than were the Indian mathematicians who gave us zero and the so-called Arabic numerals. Why shouldn't the specialists in the study of the linguistic symbolic system make comparable advances with those of specialists in the numerical symbolic system?

The answers are not hard to find. First, as noted above, the linguists were not interested. Their major figure of the time regarded language as an ideal system located in an imagined organ of the brain, in a conception leading to abstract mapping of the functioning of that organ rather than attention to language. Second, language scholars in their humanistic tradition were completely unused to costly, large-scale activities. An undertaking of the American Academy of Arts and Sciences published less than a decade and a half ago a supposedly authoritative volume entitled *Language as a Human Problem*, in which one of the editors points to the solicited article whose author "discusses in some detail the débacle of machine translation which wasted millions of the taxpayers' dollars to little purpose and impeded the progress of serious research in the field" (Haugen & Bloomfield, 1974). The sum was "some $20 million on machine translation and closely related subjects...over the past 10 years" (National Academy of Sciences, 1966, p. 29). Even at that time the cost of one military aircraft was far greater than the total amount expended on machine translation. The amount is also considerably less than that required for a Cray X-MP machine. One result of the regard of humanists for adequate support is the discrepancy between the annual budget of NSF and that of NEH. Third, humanists were, and many remain, totally opposed to any attention to technological activities impinging on their area. The results of early machine translation were ridiculed. A day before this section was put on paper I once again heard an eminent humanist repeat the account of how the computer translated into Chinese the aphorism: "Out of sight, out of mind," which on retranslation appeared as "invisible idiot."

In the meantime the users of computers lumbered along with their succession of high-level languages, Fortran through its various upgrades, Cobol, PL-1, Ada, and so on. Visionaries would talk about the glorious future when there would be oral input, when the QUERTY keyboard would finally be deposited beside Whitney's cotton gin and McCormick's reaper in some recess of the Smithsonian Institution. But the visions stopped at that point, and for physical scientists they remained there. However attractive communication may be in the high-level language that human beings have developed, physical scientists, in their concern for harnessing computer technology, disregard it. Of a major funder we learn that it is DARPA's "strategic computing program" to break through "the conventional computing performance growth rate barrier of about 10% to 15% per year," and using hardware and software "achieve a factor of 1,000 improvement every three years" (Fisher, 1986, p. 84).

Moreover, an editorial appearing about a year ago in *Science* extolled the glories of the four National Centers for Supercomputing Applications, singling out the Beckman Institute (Abelson, 1985). Entitled "Computers and Interdisciplinary Research," the editorial lists the "supercomputing applications." We read of its "work stations with good computer graphics." Further, its "Intellectual Center...will bring together some of the nation's best physical and natural scientists and computer engineers, social scientists and computer professionals." In its "key objective..to foster interdisciplinary research...it will consist of two centers: a Center for Materials Science, Computers, and Computation and a Center for Biology, Neuroscience, and Cognition." Inclusion of linguistics among the disciplines, to say nothing of the English Department, which presumably has some acquaintance with one of the human languages useful for computer input, apparently never occurred to Philip H. Abelson, Deputy Editor of *Science* for Engineering and Applied Sciences, nor to his colleague deputy editors, nor to his editor, nor to the long list of reviewing editors.

Before discussing the status of computer control of language today, we might note a recent statement recognizing its importance, in a book of Joseph Deken with the title *Silico Sapiens*, to which I return below. In his introductory chapter Deken says flatly: "Humans are no longer alone on earth as users and generators of language; computers joined us when the mathematician John von Neumann designed a machine that could remember and transform its own instructions." Deken goes on to warn about the "anthropomorphic trap" for anyone "watching a computer communicate in words (and nowadays even speak aloud)...." (1985/86, pp. 14-15). The computer's incipient control of human language apparently triggers the trap. Yet Deken may be too fearful, as those know who have tried the Kurzweil machines, one of which reputedly reads texts, the other of which understands speech. Anyone who has used them almost comes to believe in an ideal language, for they refuse to deal with such forms of texts as are produced by a dot-matrix printer.

Some of us have for some time "watched" computers like Gunnar Fant's Øve communicate not only in words but even in sentences. We have also seen the struggles to maintain research on human language, supported almost entirely by commercial interests, many concerned with machine translation. Information on the status of that research in 1985 may be obtained from Tim Johnson's book *Natural Language Computing: The Commercial Applications* (1985). Among other information we learn that projected sales of natural language processing applications in the United States will move from 15 million dollars in 1985 to 418 million in 1990 and to approximately 1500 million in 1995 (Johnson, 1986, p. 93). Further, the European Community has in operation a system which translates 350,000 words per hour. The system is known as Systran. Its fabricator, Peter Toma--originally chief programmer in Dostert's group at Georgetown University--last year sold Systran for $20 million, the sum which horrified the National Science Foundation and the National Academy of Sciences in 1963, and troubled the editors chosen by the American Acaademy of Arts and Sciences a decade later. A recent news release states that the Japanese-English version of Systran produces translation at the rate of 1.3 million words per hour.

Since the procedures of industrial applications are privileged information, I can give no details on the grammars and lexicons developed for these applications. Some machine translation systems, however, are less secretive. The procedures and requirements are after all straightforward. One may learn improved techniques or ideas by examining other systems, just as General Motors can devise improved gimmicks after tearing a Nissan or a Porsche apart. But like any technological product, translation systems are complex affairs produced by dedicated experts who do not, and possibly cannot, modify one segment of their own product simply on discovering a new device in another. The new device may not fit into the existing scheme. Translation systems and their linguistic bases are not just thrown together, but carefully designed. Like the production of other components of the "(super-)computing development," such as its hardware and software, the production of the linguistic analyses necessary for communicating with computers through human language requires hard work by informed and talented scholars.

The situation can be equated with that arising when speakers of one language wish to communicate with speakers of another in a totally different culture. Linguists know the situation well. Non-linguists probably have run into accounts of it in the Pacific, as through Melville's novels. For such communication, a compromise language known as a pidgin was evolved. Communication with computers differs somewhat in that computers use a binary numeral system rather than a language of different structure. At first the human in the human-computer interchange made all the concessions. The human used the computer's language, so-called machine language. Finding it cumbersome because of the complex syntax necessary when the basic entities on which a language is built are only two, the human soon began constructing higher-level languages. These came to be increasingly like human language, especially when explicitly framed to approximate them, as are LISP and PROLOG. However high in level, they are still pidgins, or possibly by now creoles, since they come to be used by native computer-workers who do not first learn other programming languages. In accordance with the solution sketched, the computational society will proceed to use the highest-level languages, those developed among human social groups. Following the rhetoric of our colleagues, we may call human languages super-high-level languages. In time they may be replaced by so-called artificial languages at a still higher level; no such language has yet been devised.

Without denigrating the national language of Papua, or the common language of Haiti and others like them, we may regard pidgins as inadequate for all-purpose communication. Besides its status as a pidgin, LISP is a VSO language, scarcely the most-favored type. It makes great demands on its users, most of whom use SVO languages natively. McCarthy formed LISP in accordance with the arrangement that had been adopted in modal logic. The problems caused to their users by VSO languages may be indicated by the shift of these to SVO structure when they come into contact with other languages or are used by a wide array of speakers, as in Modern Hebrew. Whether such a shift will apply as further forms of LISP and PROLOG are devised remains to be seen. Whatever happens, LISP and PROLOG still make concessions to the machine.

Nonetheless greater ease for the human con unicator has come with the application of such programming languages, and further with the production of LISP machines. Since their availability about six years ago, we have a favorable situation for language-processing activities. The situation will be almost explosively augmented with the availability of LISP chips, an advance announced by a major manufacturer of computer equipment for two years from the present. And connection machines, allowing parallel processing, should permit computer handling of language in accordance with its multiple structure, with vast improvement over current computers based on von Neumann's logic.

The availability of LISP machines and appropriate languages has speeded the production of large-scale grammars and dictionaries. Like the theoretical proposals of linguists since Jespersen and Koschmieder, these are based on formal systems. They are relatively complete for some languages at the sentence level. Further, some problems in discourse are handled, such as anaphoric relationships. But when languages like German, French, and even English have gender distinctions in anaphors, the problems of translation have only been skirted.

More adequate solutions depend on further semantic analysis. Entities in the lexicon initially had few descriptors, partly because of the demands on computer storage. Storage is now less troublesome than are the demands on lexicographers to determine appropriate semantic descriptors when these become increasingly delicate. Similarly, as grammars increase in size, the addition of new rules requires great care because any new rule may interfere with the application of more general rules. In spite of such problems, translations now produced by some systems are surprisingly good and well received. Some post-editors with long experience in dealing with translation by skilled human translators now evaluate translations by computer systems as superior, in part because of their consistency.

In noting such comments, we must be aware of the type of language texts involved. This is the language of manuals, precise to the point of dullness, as devoid as possible of any stylistic sparkle. Such texts are produced in hundreds of thousands of pages today to guide users of equipment far more complex than cameras with their manuals of fifty or so pages or minicomputers with manuals of several hundred pages. The manuals, however long, have to be ready when one installs an electricity generating plant or a telephone switching system in Brazil, Indonesia, or elsewhere, even in the United States. Concentration on the restricted language of such texts may provide a false idea of the complexity of the problem we have in dealing with human language through computers.

The danger of a simplistic conception of the problem is heightened by the reductionism in linguistics during the past century. The neogrammarians from 1875 or thereabouts restricted their attention largely to the least complex sub-systems of language, as did the structuralists. The generativists restricted their attention to an unreal language, in their words, an ideal language. Any attention to language as treated in the teaching of English, whether as a first or second language, or to language as examined in applied linguistics, must be directed at language as used by speakers in their social activities.

The damage to treatment of language by the reductionism of the major movements in linguistics during the past century is obviously complex, as will be the move to a more adequate treatment. Fortunately, there are now strong movements toward a new approach, away from reducing language to a system which is to be encompassed by procedures of mathematics or elementary logic. I cite only one study, an essay by Robert de Beaugrande to appear in the *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*. The essay is even longer and more complex than its title, which goes as follows: "Determinacy Distributions in Complex Systems: Science, Linguistics, Language, Life." In his final section Beaugrande makes the following statements about language as a dynamic code and about much of the work in linguistics:

> Each speaker has, like any system, a history with several dimensions: the cultural design of the society, the speaker's life, and his or her current actions and experiences. We might see a natural language as a complex system in which spoken and written texts manifest the coded aspect and the use of those texts the dynamic aspect. The problem of linguistics is that we have been analyzing the code in all sorts of ways, but seldom the dynamics of its use. (1987, p. 32)

Beaugrande does not cite Bakhtin, whose criticism of the partial attention which has dominated humanistic study is beginning to make an impact in all areas of humanistic study. Bakhtin sees language as "defining the relation between the self and the other," in a felicitous phrase of Joseph Frank (1986, p. 58). Language in this view is scarcely a code to be comprehended in a linguistic theory, however many sub-theories are proposed.

To deal with natural language adequately Beaugrande sketches activities which he calls "forbiddingly vast," requiring the design and exploration of many models of language "before we can narrow them down to a few plausible ones" (1987, p. 35). Recalling the attractions which model-proliferation seems to hold for linguists, a follow-up to Beaugrande's evaluation raises the possibility that linguists will once again sequester themselves in isolation, funded adequately only by standards which have been engrained among the bovine humanists. In the meantime, applications of computer manipulation of language will be left to non-specialists. I cite examples.

Software modules are alrady being marketed for language teaching, among other applications. They are universally poor. Produced by non-linguists, they are designed for the kind of large-scale market which brings appreciable financial returns. Some even seek to teach the spoken language through the facilities of inexpensive minicomputers. Moreover, while we may say that funds for education are inadequate, even the poorest school districts are being equipped with such minicomputers. It does not require the services of a super-salesman to sell the available teaching modules to these schools. According to Joseph Raben, the well-known authority on uses of computers in the humanities and social sciences, the

> emphasis now seems to be on what the computer does best rather than on what students (and teachers) need. In fact, a substantial portion of microcomputer sales is based on the premise that future generations of students will need to understand the computer itself as the foundation of their success. From this central idea, apparently, come programs that teach foreign languages by drilling students on lists of vocabulary....That is, the machine's ability to present a question, to compare the answer to a stored response, and to present then a signal of approval or disapproval is utilized without any defense of the proposition that this is the preferred method of instruction. (1983, p. 25)

In short, without appropriate attention by teachers in the humanities, including foreign language teachers, we will once again find ourselves in a situation like that of the language laboratories, which were poorly used and supervised not by professors in the humanities but by inadequately paid assistants. Or we may replay the even less attractive situation of machine translation in 1963 when an external committee moved in to cut off all research activities, and pretty well did so.

Fortunately, if a bit tardily, the largest association of humanists in the country, the Modern Language Association, is taking action. Among a broad agenda is the establishment of national centers for language teaching. Even more happily, in spite of the problems with the federal budget, the new Title VI of the Higher Education Act authorizes the Department of Education "to make grants to institutes of higher education to establish and/or operate comprehensive language and area studies centers." The sum authorized for Title VI is $55 million. Less will probably be made available. Yet the seriousness of the intent may be indicated by the reduction of "the number of meetings per year of the Advisory Board from four to two" and the "number of Board members from 23 to 9."

Only through large national centers will the prime specialists be enabled to ensure up-to-date facilities for the use of computers in humanistic activities, comparable to those in the physical sciences and engineering, as an example in the teaching of languages may demonstrate. An illustration which I like to cite deals with the use of videotapes, a somewhat different technology but still not foreign to computer uses. In a series of persistent strokes of good fortune Professor Kimberly Sparks of Middlebury College was successful in persuading one of the German TV networks to provide him with a one half-hour program per week of their Walter Cronkhite newscast. Preparing these for presentation to an elementary group of students required 24 hours of Professor Sparks's time for each half-hour program. Clearly, if such a useful undertaking were to be available throughout the country, including high schools and probably even outstanding elementary schools as well as universities, it would have to be managed from a national center which would assemble the specialists and arrange distribution, in an effort to make similar arrangements for French, Russian, Spanish, Portuguese, Hindi, Arabic, Chinese, Japanese, and so on. The same applies for capable computer software.

Looking a bit to the future, I draw on Joseph Deken's interesting book, *Silico Sapiens*. As robotics, whether or not it is the "ultimate technology," overtakes us (Deken, 1985/86, p. 223), we face once again the new terminology that comes with successive advances. Long ago we learned eventually that the second syllable of *software* was to be spelled like that of *hardware*, rather than like that of *underwear*, which may have been more appropriate. We also made our way through the age of the *breakthrough* to the age of *robustness*, which is still lingering in less with-it centers. The terminology of robotics is virtually an industry in itself. For designations of the various sizes of these creatures we have no trouble with *macrobot* for the monster that digs coal, or with *midrobot* for the variety that welds and paints your Corvette; but is it *microbot* as in Mickey, or *microbot* as in Mike, for the fellow that controls your Corvette's ignition system or for the device in your camera that does all the picture-taking and even tells you about it? Or, since there are *astrobots* and *geobots*, will *robot* herself become a *hydrobot* that competes with Oxford and Cambridge crews on the Thames? Karl Čapek not only gave us a shortened Czech word for puzzlement, but also a suffix rivalling *-burger* in proliferation, as anyone learns who fares forth to provide an appropriate Christmas for the younger set this year.

One prospective form of microbot is the realicorder, which Deken describes as a "robot descendant of the video and high-fidelity systems of the late twentieth century merged with computer *simulation* technology first developed (in extremely expensive and cumbersome form) for computerized aircraft flight trainers." As Deken points out, in the use of such trainers "the human is not simply a listener or viewer, but a *participant*" (1985/86, p. 31). Among proposed uses are "realicorder tours" where participants "will see exotic sights and sounds evolve as they choose an individual itinerary." Further, this device will make it unnecessary "to learn from lifeless books...[but teachers will be able to] summon up a wealth of native speakers and authentic sights and sounds of [differing cultures, say to learn] Russian on a realicorder 'guided tour' of Moscow" (1985/86, pp. 32-33). Those who have seen the language teaching facilities of well-supported schools like the Foreign Service Institute of the State Department know that these have long used models of places like Moscow set on tables around which four or five students discuss in Russian the local scene under the guidance of a fluent speaker/instructor. The realicorder will supplant such teaching through its lively presentation of "authentic sights and sounds" in which students will participate. Students of Chinese will learn control of the Chinese language and culture through simulated tours of Peking. Similarly, students of Italian will have their tours of Rome, as will students of Latin through Rome at an earlier time. Students of Classical Greek will be able to have a dialogue with Socrates; students of Hittite will walk the streets of Boğazköy in 1400 B.C. The study of ancient languages and cultures will be on a totally new basis. Zgusta's *Kleinasiatische Personennamen* (1964) may be among the best-sellers in the year 2006.

But installation of the equipment and preparation of materials will be costly, when these are introduced into high schools and colleges, even if initially perhaps only in universities. Nonetheless, in view of current situations, such as our trade imbalance (caused at least in part by our ignorance of foreign languages and foreign cultures), or even of more tragic events, such as our entry into the Vietnam war or our long-standing inept policy in the Middle East, we would profit more from the development of computerized robots to teach foreign languages and foreign cultures than from determining the activity of the 247th electron of uranium as it rotates around the nucleus, or than from improved understanding of the weather on Mars, Jupiter, or even Earth. Because of such benefits it may even be possible to devote a small fraction of the sums assembled for Supercomputer installations like the Beckman Institute to the study and teaching of language, say $10 million per year. In this way the National Science Foundation might make up in part for its destructive report of twenty years ago which effectively blocked further computational linguistic activity. The Beckman Institute might also fulfill the hopes of the American Association for the Advancement of Science as a center for interdisciplinary research. And the National Endowment for the Humanities might finally recognize that activities in its sphere require funding for matters other than setting up summer seminars for inculcating the procedures of deconstructionism.

Leaning on Deken one last time, we may agree that "Human *natural language* is rich in all the aspects of vocabulary, syntax, and semantics, but for the present at least, it is far beyond the comprehension of robots" (1985/86, p. 148). How better could the Beckman Institute and the faculty of the University of Illinois at Urbana-Champaign, as well as the National Science Foundation, fulfill their aims than in developing the best possible means of teaching human students to teach today's computers and robots to become the more adept computers and robots who will serve the students of the future? Or will it be vice versa?

THE AUTHOR

Winfred P. Lehmann is Louann and Larry Temple Centennial Professor Emeritus in the Humanities. He is also Director, Linguistics Research Center, The University of Texas at Austin.

REFERENCES

Abelson, P. H. (1985, November 22). Computers and interdisciplinary research [Editorial]. *Science, 230*, 4278.

Beaugrande, R. de. (1987). Determinacy distributions in complex systems: science, linguistics, language, life. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*.

Deken, J. (1985/86). *Silico sapiens*. New York: Bantam.

Fisher, E. M. (1986, October 1). The dream machine. *Datamation*, pp. 79-84.

Frank, J. (1986, October 23). The voices of Mikhail Bakhtin. *The New York Review*, pp. 54-60.

Haugen, E., & Bloomfield, M. (Eds.). (1974). *Language as a human problem*. New York: Norton. (Most essays published previously in *Daedalus* 1973, copyrighted by the American Academy of Arts and Sciences.)

Johnson, T. (1985). *Natural language computing: The commercial applications*. London: Ovum.

Johnson, T. (1986, January 15). [Report.] *Datamation*, p. 93.

Lehmann, W. P. (1961). Machine translation: Agent of the humanities. The University of Texas *Studies in Literature and Language, 3*(2), 163-170.

National Academy of Sciences. (1966). *Languages and Machines: Computers in Translation and Linguistics* (Publication 1416). Washington, DC: National Academy of Sciences.

Raben, J. (1983, March-April). Advent of the post-Gutenberg university. *Academe*, pp. 21-27.

Zgusta, L. (1964). *Kleinasiatische Personennamen*. Prague: Tschechoslowakische Akademie der Wissenschaften.

THE USE OF THE PERSONAL COMPUTER IN THE HUMANITIES

James W. Marchand

Humani cs have in the past been tied to the mainframe and the tyranny of the keepers of the mainframe. The personal computer revolution has changed all this. The individual humanist now has available to him computing power far beyond that of the mainframes of the sixties. We cannot afford to let this opportunity pass. This article offers in its first part a survey of some of the applications of computers to the study of the humanities, including both tools and techniques. The second section of the article is a report on what one humanist has done with limited resources.

The most important thing that the new revolution in personal computers has done, and it is a revolution, has been to free the humanist from the mainframe, the keepers of the mainframe, and the tyranny of the programmer, with its eternal question of "What to Tell the Programmer" (Heal, 1971). The humanist, with his inborn tendency to go one-on-one with the texts he deals with, is, in George C. Homans' words, the last academic individualist (Homans, 1972). With the advent of the fifth generation and VLSI (very large scale integration), we may have reached the limits of miniaturization in the ergonomic sense. The keyboard can be redesigned, Dvoraked, or reconfigured, as is the new AT keyboard, but it cannot be made smaller; the miniaturization of the chip and the other components, however, continues at a merry pace. We can at the present time, for not a huge outlay of money, have a desktop with 170 megabytes of storage and 5 megabytes of addressable RAM. DOS 5 promises to give us soft and firmware to run these monsters. For more money, we can go as high as a 1.4 gigabyte hard disk, that is, a disk which will hold 1.4 billion characters. Each mail delivery brings news of dramatic new improvements, a megabit chip, an add-on to allow the AT to run at 4 to 5 MIPS (millions of instructions per second), ever new scanners and printers.

Even at present, I have on my "little" 1984 issue AT the entire Greek New Testament, the King James Bible, and the Gothic Bible, with plenty of room left over for the software to interrogate them. I was able to buy an Oberon International Omni-Reader for $70 which allows me to input without keying in, so long as I stay within the narrow limits of office typewriting. Even on my Seiko UC-2000 watch, I can put enough answers to allow me to cheat through any hour exam, a kind of electronic sleeve, as it were. For the purposes of this article, however, I have for the most part stuck with a minimum configuration. Most of the things I discuss were done on an IBM PC with 256K of RAM and an IBM Graphics printer. Almost all the programming was done in uncompiled BASIC or using batch files with DOS. On the present market, I could do most of it with an outlay of less than $700. The work was done mostly by one humanist, working alone, with no advice from a team of programmers. I say "mostly", since I occasionally uploaded to the mainframe in ASCII for this or that large task which I could have done on a 256K machine, but which would have consumed inordinate amounts of time. Being an individualistic humanist type does not mean that one has to be stupid.

This article is divided into three sections, input, handling, and output. Before I begin, let me make a bow towards the mainframe and describe one project which is being done by a team. Professor Spurgeon Baldwin, Robert Booth and I have a project using computer enhancement to read manuscripts which cannot be read by the naked eye. This consists of obtaining a picture of a manuscript, done by the best photographic techniques available, digitizing it with a digitizing camera and downloading it into the computer's memory. The software assigns to the picture 256 levels of gray, each of which is addressable. One can then ask the machine to do whatever one wants with each level of gray, give it any color, ignore it, etc. If, for example, the original manuscript is a palimpsest, in which one text is written over another, it is possible to separate the scripts. We are frequently able to read letters and words in Gothic manuscripts, for example, which have resisted all previous efforts.[1] This takes, of course, enormous computing power, though with a large memory, PC-EYE and IMiGIT,[2] one can do something like this on a desktop. The recent advent of FFT (fast Fourier transform) programs for the personal computer will lighten our burden tremendously.

INPUT

The first problem one encounters in dealing with textual, visual and auditory materials is the difficulty of getting them into the computer. With texts, one can simply type them in; if one is dealing with a language with a "strange" writing system, one can make up a system of transliterations into ASCII symbols, numbers, etc. Even using BASIC and the draw command, one can have them appear on the screen in the original form. Note the simple screen shown in Figure 1 (slightly retouched, for reproduction purposes), in which the runes (the Old Germanic writing system) welcome conference participants to Chambana, turn upside down, change colors, etc. (cf. Illowsky & Abrash, 1984:166f.). This demonstrates that it is possible, with little effort, to have the output device show the characters in their original form. I have devised cuneiform, hieroglyphic, runic, and Gothic characters. Texts can be input by the lazy or impatient using a scanner, such as the Kurzweil Data Entry Machine,[3] which has been programmed to

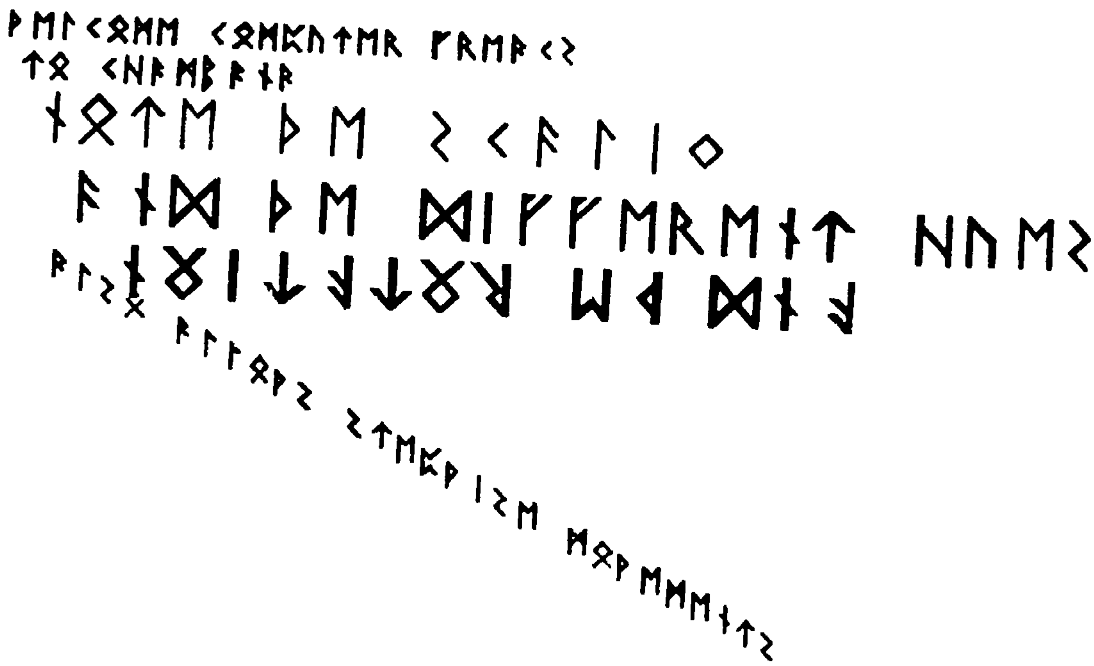


Figure 1. Availability and mobility of a non-Roman orthography on a computer monitor.

read almost any writing system, the PC Scan,[4] which is much faster and reads about 10 fonts, or the Datacopy,[5] which is also trainable and is faster than the Kurzweil, though not so fast as the PC Scan. For those with money, the miraculous self-training Palantir Compound Document Processor rivals the Kurzweil.[6] Even the lowly OmniReader can do some texts. Thus, we have already arrived at the time when a private scholar can expect to be able to buy an OCR (optical character recognition device) and input his own texts.

The private scholar who belongs to the right clubs can also be expected to be able to tap into the burgeoning CD ROM industry. Already available on one CD ROM disk is the Thesaurus Linguae Graecae, with most of the Greek texts from the beginning to 600 AD (Berkowitz & Squitier, 1986). The FCAT group at the University of Pennsylvania, also with Packard Foundation backing, as reported by Bob Kraft (Offline 6, 1986), will have a huge number of Biblical and Patristic texts available on CD ROM for members who share their texts. The Public Domain group has 605 diskettes available on one CD ROM, which Lambert & Ropiequet (1986) have aptly called the new papyrus (cf. also Ropiequet, 1986). For those who are able to afford it, the corpora of London, Louvain, etc., can also be downloaded or obtained by mail. There is a clearinghouse for such corpora at Rutgers. By modem, and one can now buy a half-card Hayes compatible 1200 baud modem for less than $100, one can contact any number of data-bases both here and abroad. There is a new gateway service offered by the venerable Western Union at the cost of $25 per year. In addition, many texts which were typeset by computer are graphic work the humanities scholar must do, the R. R. Bowker Co. has all its catalogues available on one CD ROM disk. Most larger libraries have automated their catalogues; one can even call up and download the Library of Congress Subject Headings. Foreign libraries may also be accessed by modem. In short, we are able to input texts in many different ways, and with the continuing work on OCR, we may be able soon to begin reading manuscripts such as the Codex Argenteus, the major source of our knowledge of Gothic, with its regular handwriting, without benefit of the human finger.

Visual materials such as the Mona Lisa or Swedish church art can at present be input into the machine to be worked on, and scanners are getting better and better. With the advent of EGA (Extended Graphics Adapter) technology, one can have fine resolution 16 color palettes, making the Mona Lisa, the favorite of EGA hackers, look pretty good. Archives of art and such things as runestones should soon be available by remote transfer to any humanities scholar, so that the medievalist will not find it necessary to make his yearly trek to Princeton or the runologist his trip to Sweden. In fact, the libraries and art galleries of Europe ought to be available in machine-readable form. Our present technology permits three-dimensional recording for dealing with runestones and other monumental forms.

As far as auditory materials are concerned, we are still in the infancy of an important field. The MIDI (musical instrument digital interface) explosion, an explosion within an explosion, as it were, has made it possible to do many things with music. Inputting tunes into the computer is no problem for those who can read music only slightly. My computer even plays "She'll be Coming round the Mountain," "Mama's Not Dead, She's Only a-Sleeping," and any number of tunes on the funky piano, using Neil Rubenking's algorithms and devices.[7] Using MIDI techniques, however, with a multi-voiced output, one can input medieval music and play it with different time signatures, etc., whether or not one knows anything at all about music. I can assure you that this is a boon for those who teach medieval minnesong.

In contrast to music, the inputting of speech is much more complex and demanding. Since we do not have at present voice input machines which work sufficiently well, most speech data must be keyed in by hand, an arduous and time-consuming task. However, some progress has been made; IBM has announced a prototype which recognizes 20,000 words, somewhat larger than many people's active vocabulary, and which disambiguates words by searching both the immediate and the subject context.

HANDLING

What does one do once something is in the computer? First, one can catalog things easily, simply by assigning them numbers, as is done by the Motif-Index of Folk Literature of Stith Thompson (1966). The following is an example from my own work: If one looks at the ceiling of a medieval Swedish church, one will notice that it is painted in beautiful colors. A closer look will show that it is divided into individual paintings, and that these are organized into schemes. A further examination will show that these pictures follow the organization of the ribs in the vaults. Medievalists have come to recognize most of the schemes, calling this the science of iconography, and the learned doctors of Leyden have arrived at a numerical scheme which registers these. For example, any medievalist seeing a man with the jawbone of an ass in his left hand would immediately recognize Cain, who, in the medieval view, slew Abel with the jawbone of an ass (Schapiro, 1979). What I have done in my ongoing study of medieval Swedish church art is to organize the ceiling of the church into areas according to the four schemes of ribbing, so that we can now say what occurs and where it is found in the more than 15,000 paintings which Albertus Pictor made in his lifetime. With EGA techniques, enhancement techniques and the possibility of putting text on our pictures, we can now make statements about Swedish medieval painting which were impossible before, even including athetizing one of his churches (Marchand, 1976).

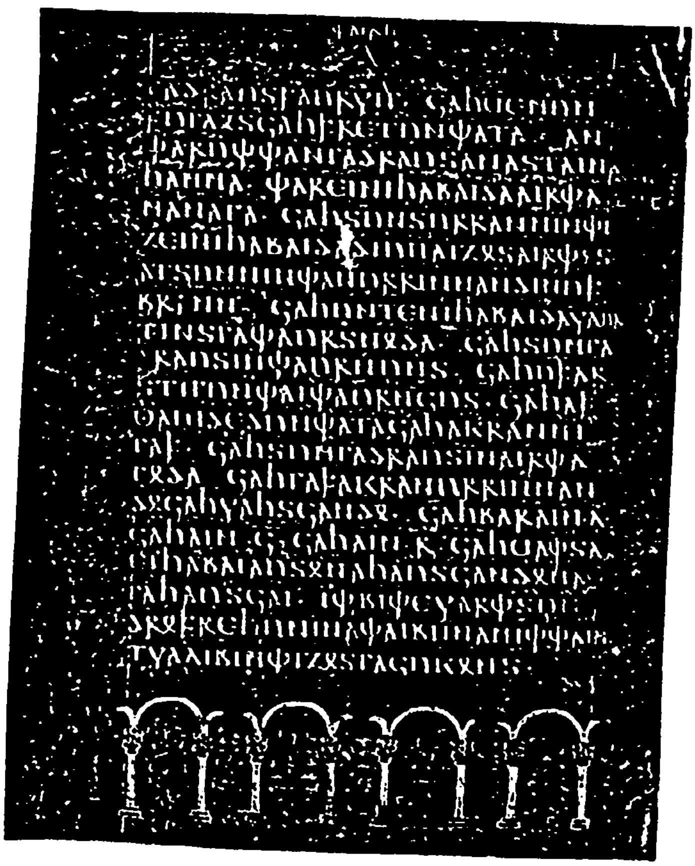


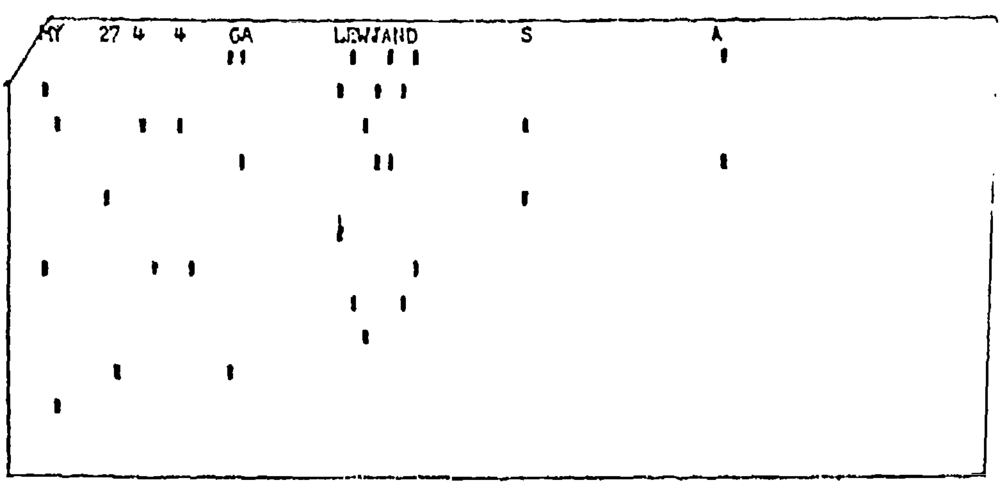
Figure 2. A page of the Codex Argenteus

Let us speak for a moment of the nature of the beast for the medieval philologist. Figure 2 (Uppsala University Library, 1955) is a representation of a page of the Codex Argenteus, one of the most precious manuscripts in the world, presently residing in Uppsala, Sweden, booty from the 17th century wars of Queen Christina. It is written in a unique alphabet, in gold and silver on purple parchment, probably for Theodoric the Great, since the canon tables at the bottom of the page seem to be modeled on his castle in Ravenna. I entered this text on punched cards in 1960, with a grammatical analysis of each word, in a scheme which looked somewhat like Figure 3. In 1960 we did not have mag tape, we programmed in FORTRAN or not at all, and we had no string manipulation capability. One had to input the grammatical codes or one did not have them. Figure 3, from a talk I gave at the Modern Language Association meeting in 1966, shows what we could obtain even then.

I. The fields on a Gothic card:

Book - chapter - verse - number of word in sentence - prefix - stem - ending - particle - grammar code - foreign word - error?

II. Sample card.



III. Questions asked:

1. alphabetized list of tokens.
2. backwards alphabetization of tokens.
3. sentence print-out.
4. comparison of sentences for length.
5. grapheme count.
6. grapheme plus environment.
7. alphabetized concordance.
8. alphabetized grammar concordance (acc. to grammar code).
9. backwards alphabetization of this (gathering cases, etc.).
10. alphabetized list of stems.
11. Syntactical statement of grammar codes.
12. Numerical print-out.
13. Lists of word-types according to various criteria.
14. Gatherings of prefixes and of particles.

Figure 3. Coding a text on a punched card.

Today we can do all kinds of things with the text. We have a concordance, both backwards and forwards, to be discussed below. This means that I can sit at my computer and ask for all the endings of Gothic to be displayed, since the head-words have been alphabetized backwards. Since I have the context, in a grammatical buc (basic unit of concordance) chosen by a person who can read Gothic, namely me, I can reject or accept the samples and arrive at a fairly good grammar of Gothic, including the context of each item talked about, in a short length of time. This means that I have an on-line grammar of a language with a minimum of human interference and the assurance that I have been exhaustive. I am working on a soundex routine (the soundex algorithm is well-known, I do not have to invent it) which will allow a person who knows Gothic only vaguely (a student) to look up a word. Using the King James text and Trinity's Greek text I can parallel the Gothic with its original and a translation and modify those wherever I wish (Figure 4).

```
MY 6.24  Ni manna mag twaim fraujam skalkinon

MAT 6:24  NO   MAN  CAN  SERVE TWO  MASTERS:
          3762 3762 1410 1398  1417 2962

                              7DUNAMAI
MT 6:24 N6OUDEIS12MXNX N7DUNATAI 3SPPI N4DUSI 12MPDX

                             7DOULEUW
         N7KURIOIS   N9 DOULEUEIN9 PAN
```

Figure 4. Lines from a database containing parallel translations in Gothic, English and Greek.

Here the first line represents the Gothic text, the second the King James version in English, as furnished by <u>The Greek Transliterator</u>, by Bible Research Systems. The numbers underneath the line in this version refer to Strong's well-known <u>Concordance</u>. For example, if one places the cursor at 2962, one receives the following entry:

```
STRONG'S REFERENCE NUMBER: 2962
GREEK:         KURIOS
DERIVATION:
DEFINITION:    SUPREME IN AUTHORITY.
```

The final line is from the Greek New Testament, as furnished by Project GRAMCORD. It contains a complete parsing of the Greek, whence its mysterious look. It reads: OUDEIS DUNATAI DUSI KURIOIS DOULEUEIN. Since it represents a one-to-one transliteration, the original Greek is totally recoverable. One can even have a King James version of the New Testament in a memory resident form with the <u>WordWorker</u> from The Way, and a Greek form is promised. With these tools one can make a complete study of the relationship of the Gothic to its

Greek original, including such things as Wulfila's rendering of Greek grammatical features.

With my poor memory, I need to be able to find texts I only vaguely remember, either by searching the English and the Greek, or by using a discontinuous finding routine, such as the batch file displayed here:

```
finder.bat
echo off
FIND/N %2  %1          >PARM1.DAT
FIND/N %3 PARM1.DAT    >PARM2.DAT
FIND/N %4 PARM2.DAT    >PARM3.DAT
```

I remembered the quotation "No man can serve two masters" in Gothic, for example, as <u>ni twaim</u> and <u>skalkinon</u>. Using the simple sieve technique shown here, taken from the DOS manual, I could find the unique sentence which contained those three words:

```
finder mtjn.doc "ni" "twaim" "skalkinon",
```

namely, Mt. 6.24. If there had been more than one such sentence, I could have found them. It is possible, of course, to output to the screen, the printer, a file, a serial port, or any other device one could or would connect up. A simple use for a search is the following: The word <u>gaggida</u> as the past tense of <u>gaggan</u> occurs only once in Gothic; a DOS search reveals instantly where, and sometimes a grammatical hit is possible, even where, as above, the words are not contiguous. Streitberg once wrote on the construction <u>þaim izei</u> 'to those who' that it was the most Gothic of constructions, unparalleled in the Greek. A simple search reveals that it occurs only once. You see that having ones texts "on line" in this way can reveal things impossible to determine otherwise. Our Ambrosian codices contain different versions of the same text. A use of the BASIC <inst> command allows us to compare the two texts and indicate where they differ, and we can output this to any device for more work and perusal. Just by inserting a count loop into our program, we can obtain any kind of statistics we want.

OUTPUT

Let us turn for a moment to the matter of output. I do not want to get into desk-top publishing, but those who tell you that you cannot obtain camera ready copy from a dot-matrix are deceiving you. Figure 5 shows a sample of camera-ready copy of my soon to be published Gothic concordance, which demonstrates the formatting capabilities of the PC. It was done using an IBM Graphics printer. Figure 6 shows a page of a grammatical (retrograde) concordance of Gothic as it came off the page printer. If you have world enough and time, you can develop a font and print it using a dot matrix,

A

a 5 0.00743
1CrA 99:1 du Kaurinþium .a. ustauh
1ThB 99:1 du Þaissalauneikaium .a. ustauh.
1TmA 0:1 du Teimauþaiau .a. dustodeiþ
Nh 5:18 jah was fraquman dagis ƕizuh stiur .a. lamba gawalida .q.
Nh 5:18 jah gaits .a. gamanwida was mis.
aba 6 0.00892
RmA 7:2 aþþan jabai gaswiltiþ *aba*
1CrA 7:14 jah gaweihaids ist *aba* sa ungalaubjands in qenai.
1CrA 11:3 iþ haubiþ qinons *aba*.
1TmA 3:2 ainaizos qenais *aba*.
1TmB 3:2 ainaizos qenais *aba*.
TiB 1:6 ainaizos qenais *aba*:
aban 7 0.0104 aba
Lk 1:34 ƕaiwa sijai þata þandei *aban* ni kann.
Mk 10:12 jah jabai qino afletiþ *aban* seinana. jah liugada anþaramma horinoþ:
1CrA 7:11 jah *aban* qen ni fraletan
1CrA 7:13 jah qens soei aigi *aban* ungalaubjandan
1CrA 7:13 ni afletai þana *aban*.
1CrA 7:16 ƕa nukkannt þu qinon ei *aban* ganasjais.
GlB 4:27 unte managa barna þizos auþjons mais þau þizos aigandeins *aban*.
abans 1 0.00149 aba
1TmA 3:12 diakaunjus sijaina ainaizos qenais *abans*
abba 1 0.00149
GlA 4:6 hropjandan *abba* fadar.
Abeilenes 1 0.00149 Abileini
Lk 3:1 jah Lwsaniaus *Abeilenes* fidurraginja.
Abiaþara 1 0.00149 Abiaþar
Mk 2:26 ƕaiwa galaiþ in gard gudis uf *Abiaþara* gudjin
Abijins 1 0.00149 Abia*
Lk 1:5 was in dagam Herodes þiudanis Iudaias. gudja namin Zakarias us afaram *Abijins*.
abin 8 0.0119 aba
Lk 1:27 du magaþai in fragibtim. *abin* þizei namo Iosef us garda Daweidis
Lk 2:36 libandei miþ *abin* jera sibun fram magaþein seinai.
RmA 7:2 jah auk ufwaira qens at libandin *abin* gabundana ist witoda

Figure 5. Sample page from a Gothic concordance.

in fact, using any bit-mapped device you like, such as a screen. Our students unfortunately usually read Gothic in transliterated form, since it is so expensive to typeset it, given the fact that the fonts can only be used on those rare occasions when one prints Gothic (Fairbanks & Magoun, 1940). It is as if you taught Russian or Greek in Latin letters. Early scholars on Gothic tried to cut or cast their own fonts, with varying degrees of success. What Christopher J.

```
ID00003 BA              \.   5   .00742821
LU 0106      WESUNUH YAN GARAINTA (BA) IN ANDWAIRYJA GUDIS.
LU 0107      JAH (BA) FRAMALDRA DAGE SEINAIZE WESUN.
LU 0507      JAH GEMUN JAH GAFULLIDEDUN (BA) YO SKIPA SWE SUGGUN.
AM 0214      SAEI GATAWIDA YO (BA) DU SAMIN
AI 0214      SAEI GATAWIDA YO (BA) DU SAMIN.

ID00004 ABA             \    6   .00891385
RM 0702      AYYAN JABAI GASWILTIY (ABA)
CN 0714      JAH GAWEIHAIDS IST (ABA) SA UNGALAUBJANDS IN GENAI.
CN 1103      IY HAUBIY GINONS (ABA).
TT 0302      AINAIZOS GENAIS (ABA).
TI 0302      AINAIZOS GENAIS (ABA).
TE 0106      AINAIZOS GENAIS (ABA):

ID00005 GAREDABA        \    1   .00148564
RM 1313      SWE IN DAGA (GAREDABA) GAGGAIMA

ID00006 UNFAIRINGDABA   \    1   .00148564
TH 0210      JAH (UNFAIRINGDABA) IZWIS YAIM GALAUBJANDAM WESUM.

ID00007 FRODABA         \    2   .00297128
LU 1608      JAH HAZIDA SA FRAUJA YANA FAURAGAGGJAN INWINDIYOS UNTE (FRODABA) GATAWIDA.
MK 1234      JAH IESUS GASAIVANDS INA YATEI (FRODABA) ANDHOF QAY DU IMMA

ID00008 HARDABA         \    1   .00148564
CO 1310      EI ANDWAIRYS (HARDABA) NI TAUJAU

ID00009 GAGUDABA        \    2   .00297128
YY 0312      JAH YAN ALLAI YA'EI WILEINA (GAGUDABA) LIBAN IN XRISTAU IESU. WRAKOS WINNAND.
TY 0312      JAH YAN ALLAI YAIEI WILEINA (GAGUDABA) LIBAN IN XRISTAU IESU WRAKOS WINNAND.

ID00010 GABIGABA        \    1   .00148564
KO 0318      WAURD XRISTAUS BAUAI IN IZWIS (GABIGABA).

ID00011 HABA            \   10   .01485641
JN 0849      IK UNHULYON NI (HABA)
JN 1018      WALDUFNI (HABA) AFLAGJAN YO.
JN 1018      JAH WALDUFNI (HABA) AFTRA NIMAN YO.
CN 0725      AYYAN BI MAUJOS ANABUSN FRAUJINS NI (HABA).
CN 1531      BROYRJUS YOEI (HABA) IN XRISTAU IESU FRAUJIN UNSARAMMA.
CO 0204      YOEI (HABA) UFARASSAU DU IZWIS.
CR 0204      YOEI (HABA) UFARASSAU DU IZWIS.
CR 1112      IY YATEI TAUJA JAH TAUJAN (HABA).
```

Figure 6. Section of a reverse (grammatical) concordance of Gothic.

Meyer and I decided to do was a radical departure from 20th century tradition, namely to teach students to read Gothic in the Gothic script. If we wanted to imitate early workers, we could do much better than they, but that would merely be a modernization and one font. We decided early on to put our Gothic out in the original script. What we did was simple. We photographed photographs of the manuscripts, made transparencies of them, enlarged them and projected them on the wall using an overhead projector, traced them, put them on the computer screen and bit-mapped them. We thus arrived at an character set we could use to "type" a manuscript. The process is shown in figure 7. By the same process, we generated fonts for each of the various hands used in our Gothic manuscripts (there are 22; some are shown in figure 8). The results are good enough to fool experts. They were done on a 256K IBM PC with an IBM Graphics dot matrix printer. One can even simulate Gothic print on a dot matrix printer, so that textbooks can again be put out in Gothic (Figure 9). A piece of the Codex Carolinus (Figure 10) and of Hand II of the Codex Argenteus (Figure 11, with appropriate typos) will show you the results obtainable.

We come to our pièce de résistance. Imitating hands did not suit us, for we wanted more. We wanted to reconstitute a manuscript, to make a replica of the Codex Argenteus which looked more like the codex than it itself did. That is, we

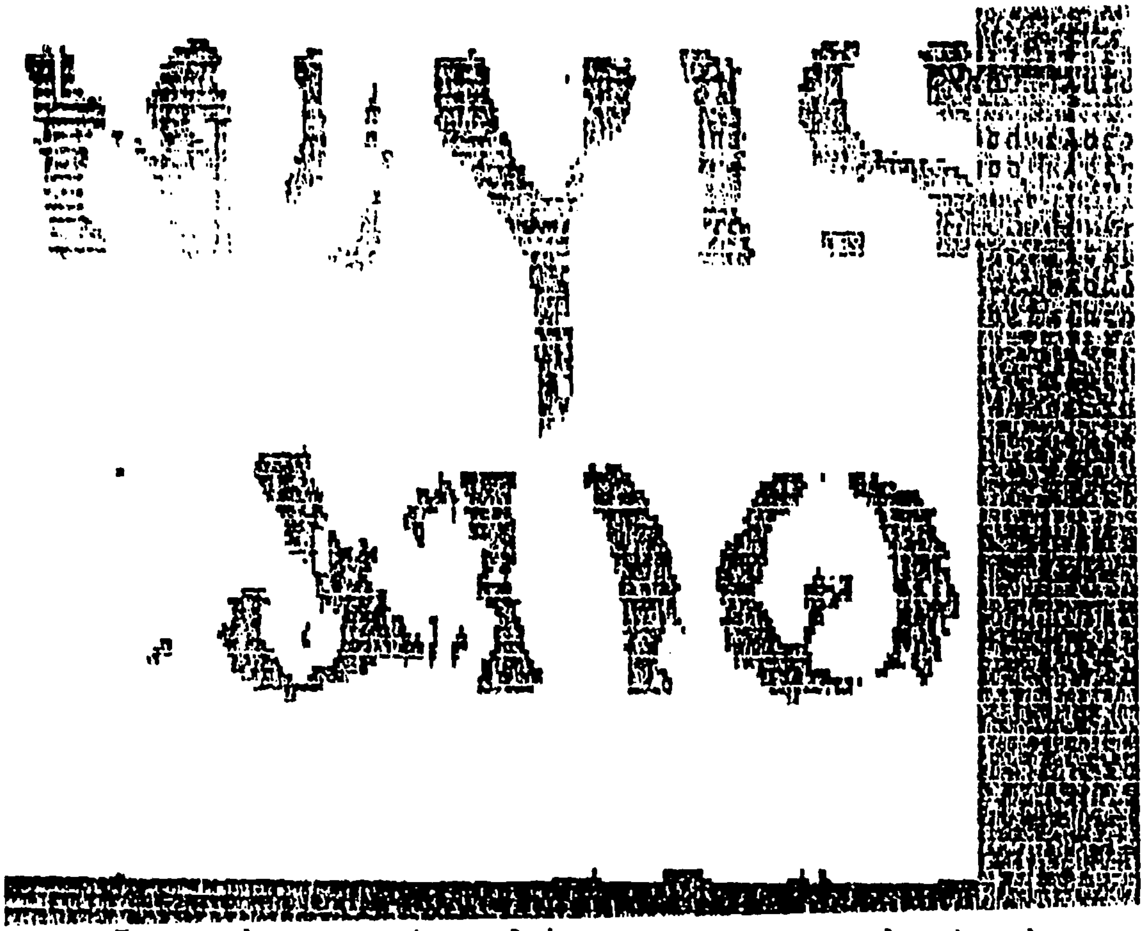
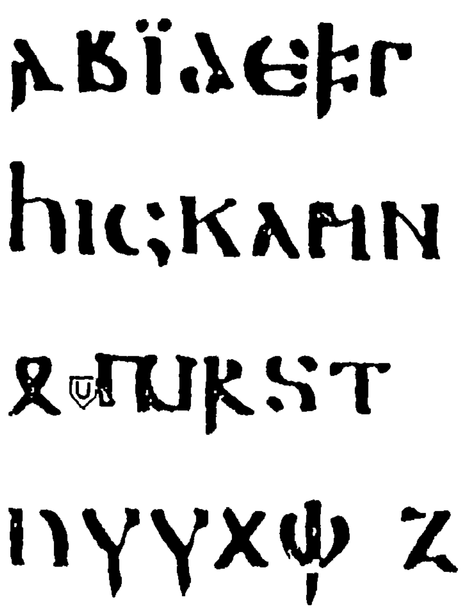


Image by computer driven camera, ready to be smoothed and contour filled, then reduced.



Font produced by electromagnetic process

Figure 7. Stages in developing a Gothic font for a bit-mapped printer.

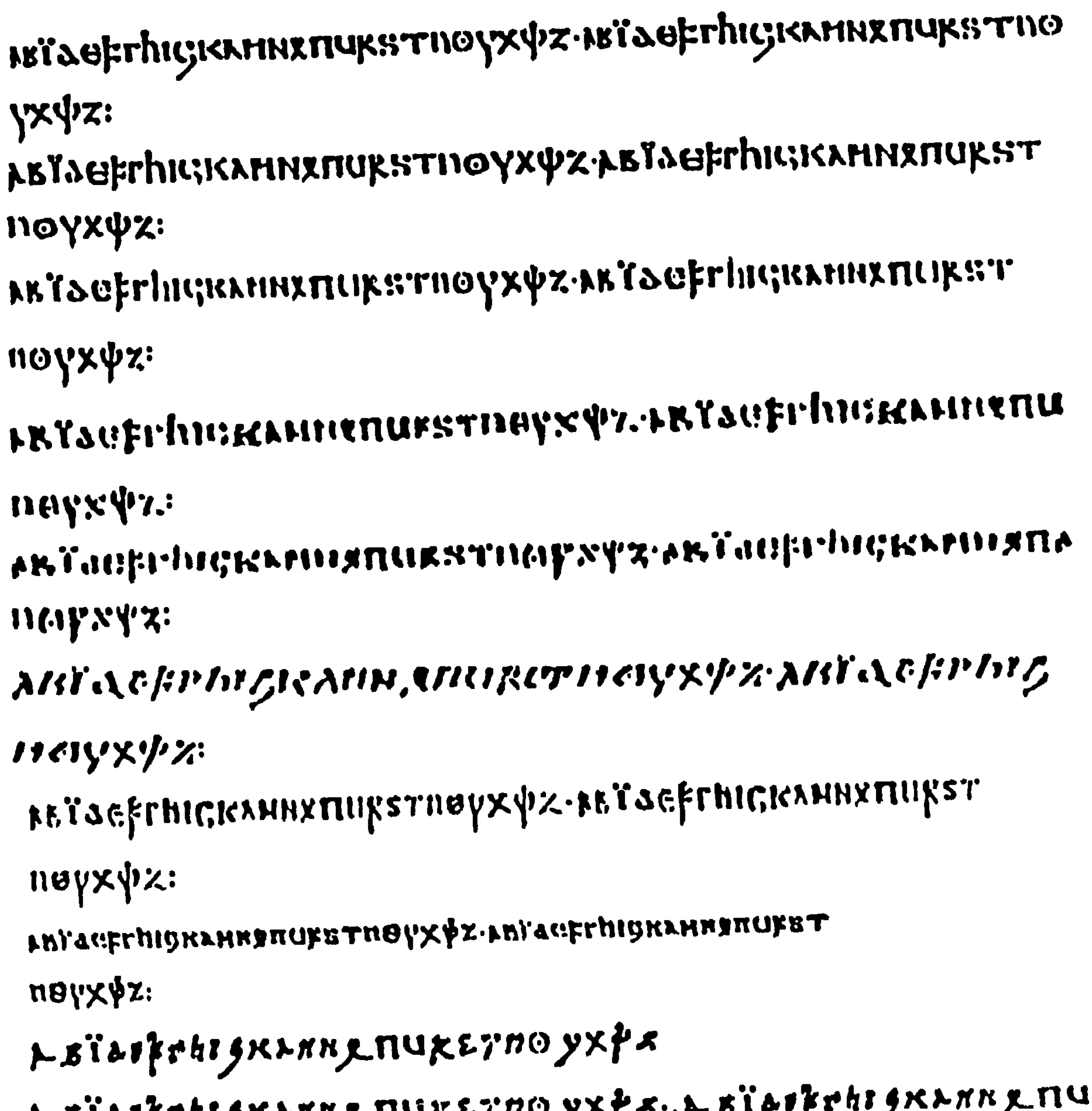


Figure 8. 10 Gothic fonts produced on a dot matrix printer.

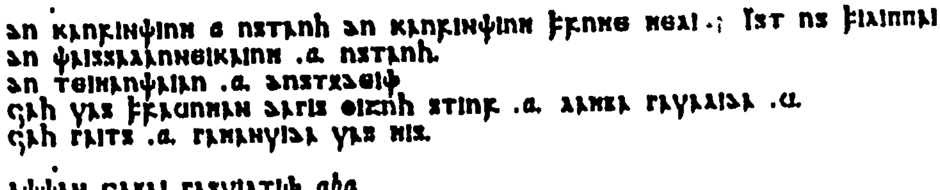


Figure 9. "Printed" Gothic using a dot matrix printer.

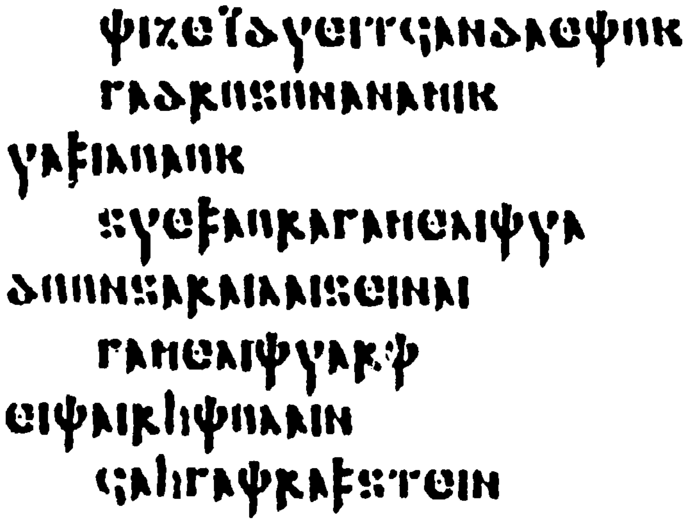


Figure 10. A dot-matrix facsimile of the Codex Carolinus.

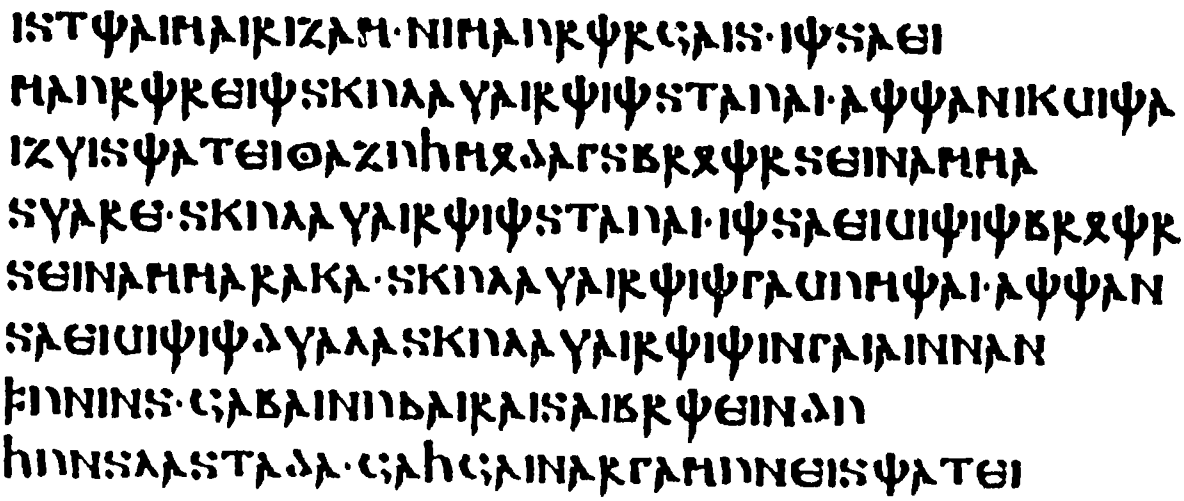


Figure 11. A dot-matrix facsimile of Hand II of the Codex Argenteus.

wanted a manuscript which looked like the original of ca. 520 A.D., one which people could read. What we did was to use a digitizing camera to download pictures of the characters. Using enhancement techniques familiar from our space effort, we contour-rounded and filled in these characters, ending up with a type font which allowed us to type the manuscript, with the appropriate, but unintentional typos (Figure 12). We had paper made which had the right shade of purple, ran our typed manuscript off on the Imagen, a prototype laser printer, made a transparency of this, had it silk-screened onto the paper. For technical reasons, I cannot print the actual result, in gold and silver on purple paper, but it looked some-what like Figure 12. The result could and has fooled experts, who thought we had simply made an excellent photograph of the manuscript, perhaps touched up. You will say that this is a rather expensive project and requires teamwork and money. It could be done, however, with a dot matrix, a video camera and PC-EYE, and one could run it off on a cheap silk-screen device such as a Rex Rotary.

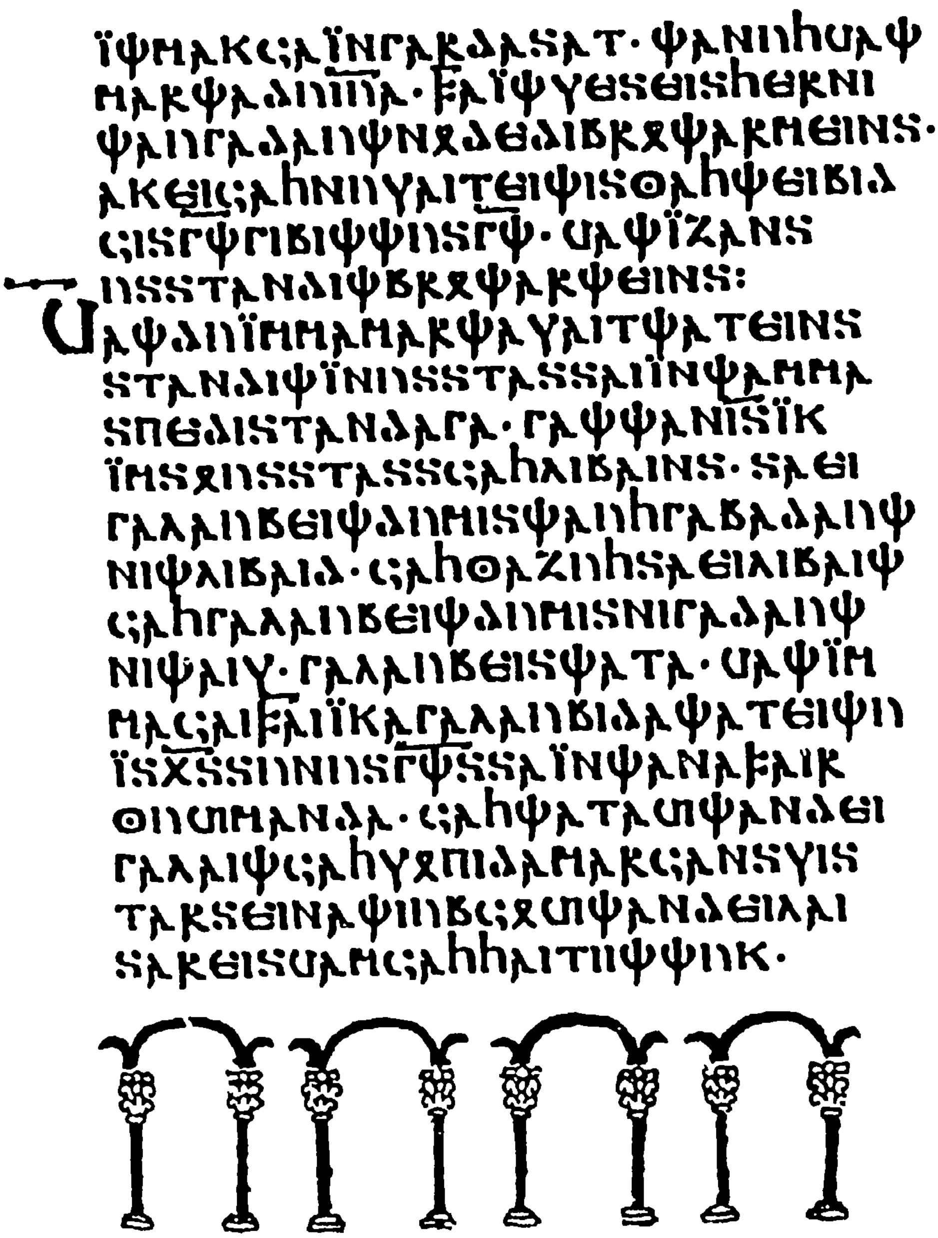


Figure 12. A computer-generated facsimile of the Codex Argenteus. Note the typo in the fourth character.

In conclusion, what I have tried to present here is not in and of itself worth very much. But it is the attempt of a humanities scholar, working very much by himself, with machines and tools he bought himself, in other words, the cottage industry we humanists are accustomed to. If humanists are to use computers, this is pretty much the way they are going to have to go until funding and teams are available. Homo sapiens and homo silicensis really need to be brought together in the individual humanities scholar if real progress is to be made. Se non è ben trovato, è vero.

ACKNOWLEDGEMENTS

I am indebted to Joseph Puryear, Christopher Meyer, and Daniel Ginther for their help in the preparation of this paper.

THE AUTHOR

James W. Marchand is Professor of German, Linguistics, and Comparative Literature, and a member of the Center for Advanced Study at the University of Illinois at Urbana-Champaign. His research interests are language, literature, and the use of computers in their study.

NOTES

[1]The December, 1986 issue of Omni reports on a similar project to read the Syrus Sinaiticus, another Biblical palimpsest (Baker, 1986).

[2]PC-EYE and IMiGIT are trademarks of Chorus Data Systems.

[3]Kurzweil Data Entry Machine is a trademark of Kurzweil Computer Products.

[4]PC Scan is a trademark of Dest Corporation.

[5]Datacopy is a trademark of Datacopy Corporation.

[6]Compound Document Processor is a trademark of Palantir Corporation.

[7]Neil J. Rubenking is the acknowledged king of shareware music. He has produced a number of Shareware programs, including Pianoman.

REFERENCES

Baker, Sherry. (1986). Hidden Gospels. Omni, 9, 49.

Berkowitz, Luci and Karl A. Squitier. (1986). Thesaurus Linguae Graecae. Canon of Greek Authors and Works. 2d ed. Oxford: Oxford University Press.

Coles, John. Archaeology by Experiment. (1973). New York:

Scribner's.

Fairbanks, Sidney and F. P. Magoun, Jr. (1940). "On Writing and Printing Gothic." *Speculum*, *15*, 313-327.

Graham, J. A., R. F. Heizer and T. R. Hester. *A Bibliography of Replicative Experiments in Archaeology*. (1972). Berkeley: Archaeological Research Facility, University of California. With updates.

Heal, J. G. B. (1971). "What to Tell the Programmer," in *The Computer in Literary and Linguistic Research*. Ed. R. A. Wisbey. (pp. 201-208). Cambridge: Cambridge University Press.

Homans, George C. (1972). *Grundfragen soziologischer Theorie*. Ed. by Viktor Vanberg. Opladen: Westdeutscher Verlag.

Illowsky, Dan and Michael Abrash. (1984). *Graphics for the IBM PC*. Indianapolis: Howard W. Sams & Co.

Lambert, Steve and Suzanne Ropiequet. (1986). *CD ROM. The New Papyrus*. Redmond, WA: Microsoft Press.

Marchand, James W. (1976). Early Scandinavian Variants of the Fifteen Signs before Doomsday. *Acta Philologica Scandinavica*, *31*, 117-132.

Ropiequet, Suzanne, with John Einberger and Bill Zoellick. (1986). *CD ROM Volume Two*. Redmond, WA: Microsoft Press.

Schapiro, Meyer. (1979). Cain's Jaw-Bone that Did the First Murder. *Late Antique, Early Christian and Mediaeval Art. Selected Papers*. Vol. 3. (pp. 249-265). New York: George Braziller.

Thompson, Stith. (1955-58). *Motif-Index of Folk Literature*. New Enlarged and Revised Edition. Bloomington: Indiana University Press.

Uppsala University Library. (1955). *Historical Notes. Guide to the Exhibits*. Uppsala: Uppsala University.

DIGGING IN THE DICTIONARY: BUILDING A RELATIONAL LEXICON TO SUPPORT NATURAL LANGUAGE PROCESSING APPLICATIONS

Martha Evens, Judith Markowitz, Thomas Ahlswede and Kay Rossi

Advanced learners of second languages and natural language processing systems both demand much more detailed lexical information than conventional dictionaries can provide. Text composition, whether by humans or machines, requires a thorough understanding of relationships between words, such as selectional restrictions, case patterns, factives and other kinds of verb implicature. For verbs we need to know whether they are action or stative, performative or not, and what kinds of complements they take. It is important to know whether an adjective is non-predicating, non-attributive, action, or stative. For nouns we need relations like taxonomy, part-whole, membership, and modification, and also attributes like count, mass, human, and animate. This paper discusses these and other kinds of lexical information found only implicitly, if at all, in most commercial dictionaries.

INTRODUCTION

Advanced learners of second languages and natural language processing systems both need much more detailed lexical information than conventional dictionaries can provide. Native speakers say 'doctor of medicine' but 'specialist in orthopedics,' even if they have to look up orthopedics to discover the spelling or meaning. Complementizers are especially confusing: wish and want are much alike, but we say 'I wish (that) he would go,' but 'I want him to go,' not 'I want that he would go.' Most conventional dictionaries, even those that explain subtle distinctions of meanings in a sophisticated vocabulary, assume that their users know how to combine the simple words. Natural language understanding and generation programs require even more detailed lexical information and are less well-equipped to learn from examples. It is the designers of dictionaries for advanced learners that have led the way in categorizing the kind of information that is needed and in trying to obtain and organize this information.

DICTIONARIES FOR ADVANCED LEARNERS

The first to propose a design for a radically new type of dictionary were the Soviet linguists Apresyan, Mel'cuk, and Zholkovsky (1970). They proposed an Explanatory-

Combinatory Dictionary that would explain the morphology of the word and its government patterns, describe the lexical universe of the entry word, and the way it combines with other words into phrases. The description of the lexical universe places a term in its semantic field and discriminates between synonyms and near synonyms. The most distinctive and original feature of their proposal was the the list of 'lexical functions.' These functions include the classical relations of synonymy and taxonomy as well as about fifty others, such as:

Son - typical sound	Son(<u>cat</u>) = <u>meow</u>
Liqu - destroying verb	Liqu(<u>mistake</u>) = <u>to correct</u>
Prepar - ready for use	Prepar(<u>table</u>) = <u>to lay</u>
Inc - increase verb	Inc(<u>tension</u>) = <u>to mount</u>
Dec - decrease verb	Dec(<u>cloth</u>) = <u>to shrink</u>

Mel'cuk has published fifty sample entries for French (1984) and a much more complete dictionary of Russian.

Three very interesting dictionaries have been published for advanced learners of English: the Oxford Advanced Learners Dictionary, edited by Hornby (1974), the Collins English Learner's Dictionary (Carver, 1974), and the Longman Dictionary of Contemporary English (Procter, 1978). All three contain detailed information about selectional restrictions, sentential complements, and semantic fields. The Longman Dictionary has a controlled vocabulary of 2,000 words and comes in an American version.

Although none of these dictionaries contains all the features described by Mel'cuk, they provide advanced learners with information not available in other English dictionaries. With great vision the publishers of these dictionaries have made them available in machine readable form for research in lexicography and natural language processing. The Longman tape contains further information too bulky to put in the printed book.

It is clear that lexical knowledge involves not only words but phrases. Becker (1975) argues that people generate text by sticking together large swatches of preformed phrases, some only two or three words in length ('by no means'), some a whole sentence ('I am so glad to see you again'). Table I summarizes Becker's classification of phrasal information needed in the lexicon. If natural language processing systems are to create text that sounds natural, they have to have phrasal lexicons.

If you take a strong lexicalist position, that is, if you believe that much of our linguistic knowledge is stored in the lexicon, then the range of what is considered to be lexical information expands to include case arguments for verbs, generic fillers of functional relation slots like subject and object, and triggers for syntactic rules like dative shift (as in 'Mary gave the ball to me' vs. 'Mary gave me the ball'). Also included are selectional restrictions, collocations, and lexical-semantic relations such as taxonomy and part-whole. Many of these new types of information are as important to computers as to second language learners--along with traditional lexical information like etymology, morphology, and phonology (all being used by programs that read text aloud [Church, 1986]). Furthermore, human lexical knowledge involves not only isolated words and phrases but whole networks of related words. The easiest and most natural way to express this kind of semantic information about the words and phrases in the lexicon is to make extensive use of the lexical functions proposed by Apresyan, Zholkovsky, and Mel'cuk (1970) and of other lexical semantic relations (Evens, Litowitz, Markowitz, Smith, and Werner, 1980; Evens and Smith, 1978).

1. Polywords	to blow up
2. Phrasal Constraints	by pure (sheer) coincidence
3. Deictic Locutions	for that matter
4. Sentence Builders	X gave Y a song and dance about Z
5. Situational Utterances	You are very welcome! How can I ever repay you?
6. Verbatim Texts	When I consider how my life is spent

Table 1. Categories from Becker's phrasal lexicon.

To build a large lexical database by hand would require the resources available to the publisher of a commercial dictionary. The only possible strategy is to extract as much information as possible from a machine readable dictionary. While several British dictionary publishers have made dictionary tapes available for research and other tape sources are available from the Oxford Archive, there is only one American dictionary available to researchers in machine readable form: Webster's Seventh Collegiate Dictionary (W7). John Olney, who produced the original W7 tapes, described his reasons for choosing to transcribe W7 instead of another American dictionary (1968). He was very favorably impressed by the large quantity of citations collected by the staff at the G&C Merriam Company and their systematic analyses of these citations.

W7 is an excellent source for lexical information. Some of that information, such as part of speech, is stated explicitly in each lexical entry, but even more information, particularly information about lexical-semantic relationships, such as taxonomic relationships and typical object of verbs is expressed implicitly and, therefore, must be extracted from definitions. Given the quantity of data available to us in W7 and our goal of building a large lexical database, we decided to try to extract as much as possible automatically. This decision implied that we had to parse the definitions.

After much discussion of possible parsers we chose to use Sager's (1981) Linguistic String Parser (LSP) from the Courant Institute at New York University. Although the theoretical framework on which this parser is based is somewhat out of fashion, the parser is an elegant, modern piece of software, which has been used to parse a large number of scientific papers. Sager and Grishman encourage others to use the LSP and make available a set of well-written manuals. The LSP has a large and sophisticated grammar, a ten thousand word lexicon, and excellent facilities for adding rules to the grammar and for expanding the lexicon. We have used the LSP to parse thousands of W7 definition texts and have found the LSP to be a valuable tool for dictionary research as well as for other natural language processing projects. We would be glad to give copies of our grammar for W7 definitions (and the LSP Mandarin grammar, which we have created for experiments in parsing and text generation) to anyone interested.

In the remaining sections of this paper we will discuss our concept of a lexical database and describe our attempts to extract some of this important lexical information from W7 using Sager's LSP.

LEXICONS FOR NATURAL LANGUAGE PROCESSING

Most existing natural language processing systems attack very specialized problems using handmade lexicons containing only a few hundred words. Before natural language processing systems can expand to understand input from wider domains, they need much larger lexicons containing precise and detailed syntactic and semantic information. Text generation systems require even more knowledge than natural language understanding systems.

We have set out to build a large relational lexicon for natural language processing applications containing as much detailed syntactic and semantic information as possible (Ahlswede, Evens, Markowitz, and Rossi, 1986). Whenever it is feasible, we have extracted information automatically from W7. We began by constructing an interactive lexicon builder (Ahlswede, 1985b) for use when we could not find the information we needed in machine readable form, or when further human input was required to classify entries properly. The interactive lexicon builder includes routines that add an entry, edit existing entries, give a list of all the relations being used in the lexicon with examples, keep track of words that have been used in other entries, but are not yet defined themselves, etc.

All entries contain relational information, regardless of the part of speech of the headword. The other information included depends heavily on the part of speech. Verb entries are the most extensive; they contain case information combined with selectional restrictions, tell whether the verb is active or stative, whether it can be put in the passive voice or not. If the verb is a performative, then the performative class is given. If it can take sentential complements, then the complementizers are listed, along with information about implicature, and whether the verb supports not-transportation. Noun entries list plural forms, factivity, and attributes such as animate, human, concrete, count vs. mass. For adjectives we include selectional information, action vs. stative status. If the adjective cannot appear in predicate position or attributive position that fact is noted. Special classes of adjectives are marked as being ordinal or cardinal, as well as for color, size, time, etc. We are still trying to figure out adverb categories, aside from the obvious time, duration, position, manner, cause, etc.

RELATIONS IN THE LEXICON

Lexical-semantic relations express relationships between words and concepts in the dictionary. They include Mel'cuk's lexical functions as well as case relations like agent, patient, instrument; collocational relations, which identify words that go together like bread and butter, concrete relations such as part-whole, and made-out-of, and various types of grading relations, (as expressed in Monday-Tuesday-Wednesday and hot-warm-cool-cold). Synonymy and antonymy are the only relations expressed overtly in W7, therefore we have had to search for hidden expressions of other relations.

Our greatest success has come from recurring word patterns that signal specific relationships. These patterns are often called 'defining formulae.' Defining formulae consist of one or more specific words in a rigid pattern; sometimes they also involve special punctuation like parentheses (Smith, 1985). Table 2 shows a few of the defining formulae that appear in W7 along with the relations that they identify. The formula "Any" + NP consistently signals a taxonomic relationship between the noun being defined and the head noun of the NP. The similar pattern "Any of a" + NP usually marks a biological taxonomy with the scientific name of the taxonomic superordinate given in parentheses. The formula "to make" + Adj clearly expresses a causative. The formula "To" + VP + ("as" NP) names the typical object of the verb being defined inside the parentheses. More details about defining formulae for nouns in W7 can be found in Markowitz, Ahlswede, and Evens (1986) and Amsler (1980).

Defining formulae often tell us about attributes too. Noun attributes include count vs. mass, concrete vs. abstract, human vs. animate vs. inanimate, and gender. The formula "A member of" + NP tells us about the element-set relation and also signals that the noun being defined is human. The formula "One who" + VP also signals a human noun, while, at the same time, giving us the generic agent for the verb. We hoped that the formula "One that" + VP would signal a non-human noun, but that turned out not to be true. Most, but not all, of the nouns defined in this way are human.

Formula	Relation	Examples
"any" + NP	taxonomy	nectar: any delicious drink
"any of a"		capuchin: any of a genus (cebus) of South American monkeys
"young"	child	puppy: a young dog lamb: a young sheep
"to make" + Adj	cause	heat: to make warm or hot redden: to make red or reddish
"to" + V + ("as" N)	generic object	mount: to put or have (as artillery) in position lay: to bring forth and deposit (an egg)
"one who"	generic	ghost: one who ghost-writes
"one that"	agent	instructor: one that instructs

Table 2. Defining formulae from W7.

VERB CLASSES

The stative/action distinction is important in the generation of dialog. Stative verbs characterize states of being like <u>owning</u>, <u>being</u>, and <u>resembling</u>, while action verbs name acts like <u>moving</u>, <u>thinking</u>, and <u>doing</u>. Not surprisingly, most verbs fall into the action class and are characterized by their ability:

1. to appear in imperative form (e.g., 'Move! Bite that dog!' but not 'Resemble your mother!' and 'Own the house!')
2. to take the progressive aspect (e.g., 'He is moving, he is biting the dog,' but not 'She is resembling her mother.')
3. to serve in sentential complements of verbs of ordering (e.g., 'I told her to bite the dog,' but not 'I told her to resemble her mother.')

The best clue we have found for identifying action verbs in W7 is to look at the definitions of nouns derived from verbs. Those that are defined as "the act of <x>ing," where x is a verb, are typically action verbs. We have taken this route because we have been unable to extract consistent formulae directly from the verb definitions and the verb entries in W7 do not tell us which verbs normally are used in imperative or in progressive forms. Unfortunately, the formula, "the quality or state of <x>ing," is not a reliable signal for stative verbs (e.g., "condensation: the quality or state of being condensed"). Some examples of defining formulae for action verbs are shown in Table 3.

Formula	Examples
"the/an act or process of" + gerund	input: the act or process of inputting
"the/an act or process "of one that" + gerund	firing: the act or process of one that fires

Table 3. Defining formulae for action verbs.

Another important piece of lexical information about verbs is the case frame. For each argument of the verb, we indicate its syntactic role (as subject, direct object, indirect object, or object of a preposition), its case (agent, patient, beneficiary, etc.), whether its occurrence is obligatory, optional, or elliptical (that is, ordinarily required, but capable of being elided if it is understood from the context), and the selectional restrictions imposed on the filler of that slot. Thus the case frame for the sense of the verb promise used in 'I promise you a balloon,' or 'I promise you that I will come back tomorrow' takes the form shown in Table 4. For verbs that can take sentential complements we need much more information, first of all, about complementizers.

Syntactic Role	Case	Occurrence	Selectional Restriction
Subject	Agent	Obligatory	Human or Collective
Direct object	Patient	Elliptical	Concrete or Sentential
Indirect object	Beneficiary	Optional	Human or Collective

Table 4. Case frames for verbs.

In the 1960's a group at Indiana University did a major study of verbs that can take sentential complements, resulting in about sixty lists of verbs that can appear in particular contexts (Householder, 1965). We are in the process of combining these lists with our lexical database. We have also written a program to test new verbs to see whether they belong on these lists. It uses our interactive lexicon builder to present verbs in a variety of frames to informants at a terminal and incorporate their judgments into the lexicon. If the informants do not agree substantially then we review the verb ourselves and see if we can figure out what the problem is. This lack of agreement seems to occur principally with relatively uncommon verbs having several senses, in a situation where different informants know different senses.

One important attribute of verbs that take sentential complements is not-transportation. We say that a verb supports not-transportation if not, never, and other adverbs of negation can be moved from the complement clause to the main clause without making a significant alteration in the meaning. The verb want supports not-transportation: 'I did not want to go' and 'I wanted not to go' have essentially the same meaning. The verb promise, on the other hand, does not display this attribute: 'I did not promise to go' and 'I promised not to go' have very different meanings.

Some verbs that take sentential complements display rather complex implication patterns between the main verb and the complement. Verbs like realize, for example, indicate that the speaker presumes the complement to be true, e.g., 'Mary realized that she was wearing magic shoes.' Verbs like pretend, on the other hand, imply that their complements are false, as in, 'Mary pretended that she was wearing magic shoes.' The Kiparskys (1970) gave the name factive to the class of verbs that behave like realize and pointed out that the presumption holds even if the main verb is negated, as in, 'Mary did not realize that she was wearing magic shoes.' Joshi and Weischedel (1973) did a much more complete analysis of implicature relations between verbs and their complements; their results are summarized in Table 5. (Here R stands for the main verb, S for the sentential complement.)

Implicature classes are very important for discourse understanding and generation because they link the discourse to the speaker's view of the world. To date we have not been able to find a satisfactory way of identifying the implicature class of a verb by simply using W7. We are trying to see if we can extract more clues from Householder's verb categories.

Class	Implicational Structure	Examples
Factive	R(S) --> S	Jerry realized that
	~ R(S) --> S	Meg baked the cake.
Implicative	R(S) --> S	We managed to
	~ R(S) --> ~S	finish the job.
Only-if	~ R(S) --> ~S	They allowed Jim to to visit China.
If	R(S) --> S	Larry persuaded Bill to accept the job.
Negative-If	R(S) --> ~S	Larry prevented Bill from winning.
Negative Implicative	R(S) --> ~S	John failed to go.
	~ R(S) --> S	
Counter-Factive	R(S) --> ~S	Mary pretended that
	~ R(S) --> ~S	Ben went home.

Table 5. Classification of main verbs in predicate complement constructions (adapted from Joshi and Weischedel, 1973).

An interesting class of verbs called 'performatives' was first described by Austin (1962) as part of his theory of speech acts. Performatives are action verbs which, when spoken, actually perform an act. When, for example, people say, 'I warn you,' they are simultaneously uttering some words and performing an act of warning. Performative verbs were also studied by Vendler (1972) and then Vendler's classification was reviewed and reorganized by McCawley (1979). We have actually been using McCawley's categories in our lexicon and, therefore, Table 6 represents McCawley's point of view. To date, we have been unable to identify defining formulae for performatives, but we have achieved some success in classifying additional verbs by checking to see if the sense-level synonyms for definitions of a verb appear in our lists of performative verbs.

Class	Description	Examples
Verdictives	"essentially giving a finding as to something." (Austin, 1962, p. 150)	acquit diagnose estimate
Commissives	"promising or otherwise undertaking"	promise espouse agree
Behabitives	"have to do with attitudes and social behavior" (p. 151)	curse thank apologize
Expositives	"make plain how our utterances fit into the course of an argument or conversation"	concede illustrate assume
Operatives	"acts by which the speaker makes something the case" (McCawley, 1979, p. 153)	abdicate appoint levy
Exercitives	McCawley divides in two:	
Imperative	"an imperative act gets the addressee to do the thing in question because it is the speaker's desire"	admonish forbid beg
Advisories	"an advisory act gets him to do it because it is good"	advise exhort

Table 6. Performative verbs.

ADJECTIVE CATEGORIES

We have developed a large list of useful adjective relations (Ahlswede, 1985a), but we are still searching for more information about adjective classes and relevant attributes. The action/stative distinction seems to be as important for adjectives as it is for verbs. There is one important difference, however; adjectives seem to be stative more often than not, while more verbs seem to belong to the action category. Action adjectives behave much like action verbs. They occur after imperative and progressive forms of the verb <u>to be</u>. <u>Kind</u> is an action adjective while <u>tall</u> is stative, as the examples in (1) make clear:

(1) Be kind!
* Be tall!
Sally is only being kind.
* Sally is only being tall

The stative-action parameter seems to be easier to identify in W7 definitions for adjectives than it is for verbs. The many adjectives defined by the formula "Of or relating to" seem to be stative, e.g., "<u>literary</u>: of or relating to books." Adjectives defined as "Being ..." seem to belong consistently to the action class, e.g., "<u>cursed</u>: being under or deserving a curse."

While most adjectives can appear in both attributive and predicate positions, some are not non-predicating and others are non-attributive. It is perfectly appropriate to refer to our neighbor as 'an electrical engineer,' but we do not say 'this engineer is electrical.' The phrase 'a civil engineer' is ambiguous, because it may refer to a person who designs bridges or to a polite engineer. If we say, 'The engineer is civil,' the ambiguity disappears; only the polite sense is possible. Two very common non-attributive adjectives are <u>awake</u> and <u>asleep</u>. I can say 'My class is awake' or 'My class is asleep,' but I cannot refer to 'my awake class' and 'my asleep class.'

Another problem for text generation programs and advanced learners who are trying to write down complex ideas in English is the rule for combining a number of adjectives in attributive position. This rule seems to depend very markedly on the semantic categories of the adjectives in question. One version of this rule (Winograd, 1971) can be phrased:

demonstrative > ordinal >
cardinal > general > size > color

as in 'these first six handsome large red trucks.' In our lexical database we mark adjectives according to the categories, ordinal, cardinal, size, and color, along with time and measure, but we are sure that we are missing many other categories and much important selectional information for adjectives.

CONCLUSION

If we are we are going to do a better job of natural language processing, then we need to make explicit things which are implicit or missing in current commercial dictionaries. In this paper we have only touched on a few types of lexical information that we expect will be available in the dictionaries of the future. We hope that these dictionaries will also serve advanced learners of second languages.

ACKNOWLEDGMENTS

This research is supported by the National Science Foundation under grant IST 85-10069. We are grateful to the G&C Merriam Company for permission to use the W7 dictionary tapes prepared by John Olney. This research would have been impossible without the help and guidance of Naomi Sager and Ralph Grishman who have given us the Linguistic String Parser.

THE AUTHORS

Thomas Ahlswede of R.R. Donnelley and Sons is finishing a lexicography dissertation at Illinois Institute of Technology with Martha Evens, Professor of Computer Science and former President of the Association for Computational Linguistics. Judith Markowitz is an Artificial Intelligence Systems Architect at Navistar International. Kay Rossi is visiting IIT from the University of Pisa.

REFERENCES

Ahlswede, T. (1985a). A linguistic string grammar for adjective definitions. In S. Williams (Ed.) Humans and Machines: the Interface through Language (pp. 101-127). Norwood, NJ: Ablex.

Ahlswede, T. (1985b). A Tool Kit for Lexicon Building. Proc. 23rd Annual Meeting of the ACL, 268-276.

Ahlswede, T., Evens, M., Rossi, K., & Markowitz, J. (1986). Building a lexical database by parsing Webster's Seventh Collegiate Dictionary. Advances in Lexicology. Centre for the New OED, University of Waterloo, 65-78.

Amsler, R. (1980). The structure of the Merriam-Webster Pocket Dictionary. Unpublished doctoral thesis, University of Texas.

Apresyan, Y., Mel'cuk, I., & Zolkovsky, A.(1970). Semantics and lexicography: towards a new type of unilingual dictionary. In F. Kiefer (Ed.), Studies in Syntax (pp. 1-33). Dordrecht: Reidel.

Austin, J. (1962). How to Do Things with Words. London: Oxford University Press.

Becker, J. D. 1975. The phrasal lexicon. In R. Schank & B. Nash-Webber, (Eds.), Theoretical Issues in Natural Language Processing, (pp. 38-41), ACL.

Carver, D.J. (Ed.). (1974). Collins English Learner's Dictionary. Birmingham: Collins Publishers.

Church, K. (1986). Morphological decomposition and stress assignment for speech synthesis. Proc. 24th Annual Meeting of the ACL, 156-164.

Evens, M., Litowitz, B. Markowitz, J., Smith, R., & Werner, O. (1980). Lexical-Semantic Relations: A Comparative Survey. Edmonton: Linguistic Research, Inc.

Evens, M., and Smith, R. (1978). A lexicon for a computer question-answering system. AJCL, 4, 1-96.

Hornby, A.S, (Ed.). (1974). Oxford Advanced Dictionary of Current English. Oxford: Oxford University Press.

Householder, F. (1965). English Verb Classes. Bloomington: Indiana University Linguistics Club.

Joshi, A., & Weischedel, R. (1973). Semantics of predicate complement constructions. Third IJCAI, Los Altos: William Kaufmann.

Kiparsky, P., & Kiparsky, C. (1970). Fact. In Progress in Linguistics, Manfred Bierwisch and K.E. Heidolph, eds., 141-173. The Hague: Mouton.

Markowitz, J., Ahlswede, T. & Evens, M. (1986). Semantically significant patterns in dictionary definitions. Proc. 24th Annual Meeting of the ACL, June 10-13, 112-119.

McCawley, J. (1979). Remarks on the lexicography of performative verbs. In J. McCawley, Adverbs, Vowels, and Other Objects of Wonder, J. McCawley, 151-164. Chicago: University of Chicago Press.

Mel'cuk, I. (1984). Dictionnaire Explicatif et Combinatoire du Francais Contemporain. Montreal: University of Montreal Press.

Olney, J. (1968). To All Interested in the Merriam-Webster Transcripts and Data Derived from Them. Santa Monica: Systems Development Corporation L-13579.

Procter, P. (Ed.). (1978). Longman Dictionary of Contemporary English. Harlow and London: Longman.

Sager, N. (1981). Natural Language Information Processing. Reading, MA: Addison-Wesley.

Smith, R. (1985). Conceptual primitives in the English lexicon. Papers in Linguistics, 18, 99-137.

Vendler, Z. (1972). Res Cogitans. Ithaca: Cornell University Press.

Winograd, T. (1971). Procedures as a Representation of Data in a Computer Program for Understanding Natural Language. MAC TR-84. Cambridge, Massachusetts: MIT Artificial Intelligence Laboratory.

VOCODERS AND SPEECH PERCEPTION: USES OF COMPUTER-BASED SPEECH ANALYSIS-SYNTHESIS IN STIMULUS GENERATION

Joseph Tierney and Molly Mack

Work on the perception of the speech signal has produced much useful psychophysical data. In previous years, stimuli used to produce these data have been obtained from simple filtering and distortion of the speech waveform, sometimes accompanied by noise. For purposes of more complex stimulus generation however, the parameters of speech can be manipulated--after analysis and before synthesis--using various types of algorithms in order to generate stimuli of interest to the experimenter.

In this article, we discuss several computer-based analysis-synthesis techniques which make different assumptions about the speech production and perception mechanisms. We also describe experiments in which synthesized speech has been or easily could be utilized.

INTRODUCTION

Experiments in psychophysics and speech perception which use speech or speech-like signals as stimuli are not always easy to design and administer because of the variability of the signal and the difficulty of isolating and controlling single features. In spite of the challenges posed by using speech stimuli, experiments utilizing natural speech which has been filtered and distorted have produced valuable data over the past several decades (Miller & Nicely, 1954; Pisoni & Hunnicutt, 1980; Luce et al., 1983; Pisoni et al., 1985).

Coexistent with this activity in auditory phonetics and speech generation has been work in speech coding for communications applications. The computer-based techniques used in this work can be adapted readily to generate interesting speech stimuli for perception research. More specifically, transforming the acoustic signal into a descriptive stream of binary digits suitable for transmission over telephone lines, satellite links, and packet communications has been a major activity of communications engineers for many years (Flanagan et al., 1979). Many techniques have been developed for dealing with this type of speech coding. All of these techniques can be implemented with programs running on current or proposed digital processors, and therefore they can be used in speech-perception and psychophysical research.

SPEECH ANALYSIS-SYNTHESIS AND APPLICATIONS

Speech encoding devices may be classified into three categories according to their data rate--i.e., the rate at which they transmit a binary signal. High data-rate devices, such as those in the 16-to-64 kilobits/sec range, are not very complicated. They work directly on the speech waveform and they utilize very little knowledge about the underlying speech mechanisms. These devices are used as standards in the telephone industry and in the military. Mid data-rate devices, such as those in the 4.8-to-16 kilobits/sec range, are more sophisticated than the high data-rate devices, and they have some potentially interesting applications in speech-perception research. It is the low data-rate devices, however, which are perhaps of greatest interest from the speech researcher's point of view. These devices, often referred to as vocoders or analysis-synthesis devices, have data rates in the range of .6 to 9.6 kilobits/sec. Vocoders provide the greatest potential for speech signal manipulation because they assume a speech-production model in analysis-synthesis.

For speech to be usefully intelligible, as in standard telephone communication, it must contain acoustic information up to about 3.5 kilohertz (kHz).[1] (As a point of reference, a human with good hearing can perceive frequencies of approximately 20 Hz to 20 kHz--i.e., from 20 to 20,000 cyles/sec.) In order to encode and transmit a speech signal with a bandwidth of 4-5 kHz at 2000-3000 bits/sec, it becomes necessary to model the speech production mechanism and to measure parameters of the model. This process is called analysis. At the receiving end of the transmission, the measured parameters are used with the model to recreate or synthesize the speech signal. This process of analysis-synthesis is the basis for several voice-coding algorithms which allow a speech signal to be encoded and transmitted in the range of 1000-4000 bits/sec.

The analysis process presents us with parameters which describe the speech signal and which can be modified to alter certain of its properties. Almost all of the systems in use at present, and many of those proposed for future use, assume that the speech-production mechanism can be modeled as in Figure 1. This mechanism has a sound source and a filter. The sound source may be the vocal cords which produce voiced (periodic) sounds and which determine the fundamental

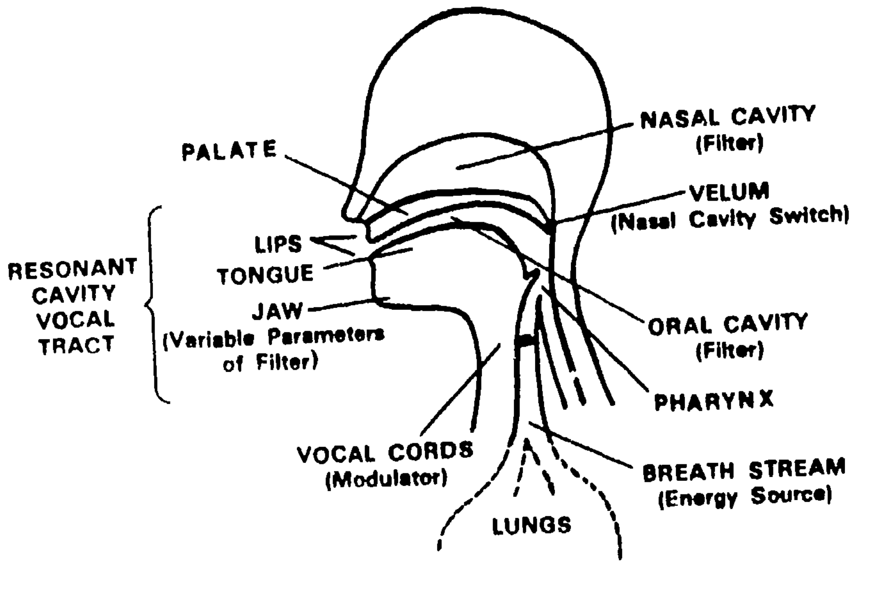


Figure 1. Lateral view of the speech-production mechanism.

frequency--recognized perceptually as pitch. The source may also be some type of intraoral constriction, such as that used to produce voiceless (aperiodic) sounds, or it may be a combination of voiced and unvoiced sounds, such as that used to produce a voiced fricative like [v]. The filter is the supralaryngeal vocal tract which includes the nasal, oral, and pharyngeal cavities. It may properly be considered a variable acoustic filter whose shape specifies the formant frequencies (also called the vocal tract resonances) of a sound. In Figure 2, note the display of a simple input waveform (a waveform generated by the energy or sound source) on the left. The period of the waveform is represented by T. To the right is the complex output waveform as it appears after being filtered by the supralaryngeal vocal tract (resonant cavity). Below these waveforms are spectra displaying the fundamental frequency (F0) and harmonics of the sound source, the spectrum envelope with formant frequency peaks, and the actual output signal, with the spectrum envelope superimposed upon the source signal.

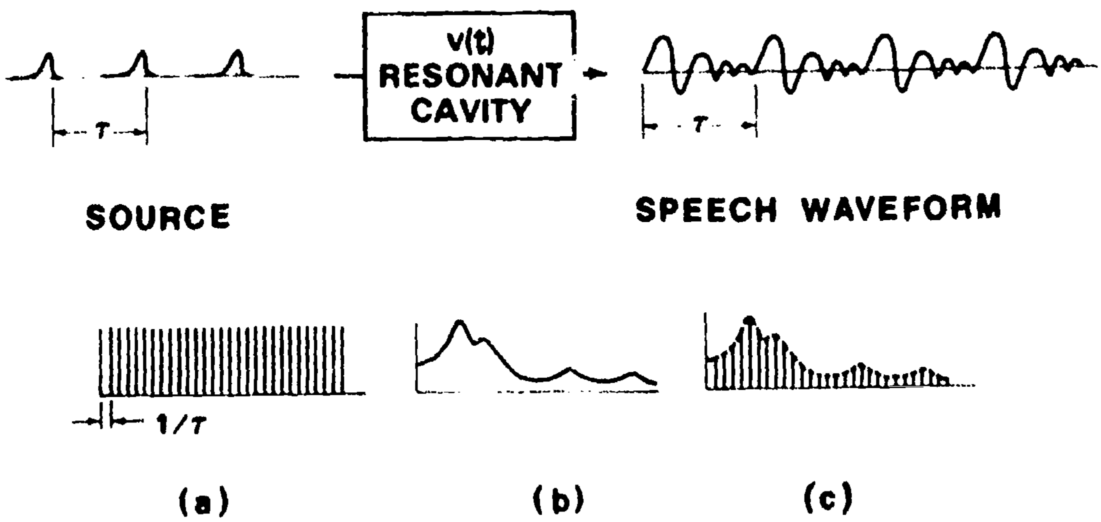


Figure 2. Top line: Input signal before and after filtering by supralaryngeal vocal tract (resonant cavity). Bottom line: Spectra showing fundamental frequency and harmonics generated by sound source (a); spectrum envelope with filter formant frequency peaks (b); and combined source and filter spectrum (c).

The challenge in speech analysis is to measure accurately these speech attributes using only the transduced acoustic signal available as the electrical output of a microphone. Assuming that the production model is accurate and that the articulators vary as a function of time, we can expect the measured spectra of speech signals to resemble the examples presented in Figure 3. The spacing of the harmonics (the vertical lines) in each spectrum reflects the properties of the sound source, with more widely spaced lines characterizing a signal of higher frequency, as in (a), and more narrowly spaced lines characterizing a signal of lower frequency, as in (b). The vocal tract filter determines the envelope of the spectrum whose peaks, at or near formant frequencies, specify a given sound. The height of the peaks reflects their relative amplitude, usually measured in decibels. In this figure the spectrum envelope is of a different shape in (b) and (c) because the configuration of the vocal tract used to produce the sounds in (b) and (c) was different. Thus, the spectrum in (a) could be the vowel [a] produced by an adult female, and the spectrum in (b) could be that of the same vowel produced by an adult male. The spectrum in (c) could be that of the vowel [i], also produced by an adult male. Once the relationship between the source-filter mechanism and the spectrum is understood, we can estimate the acoustic properties of the source and vocal tract filter from the measured spectrum. In fact, this is one way of deriving measurements for vocoded speech.

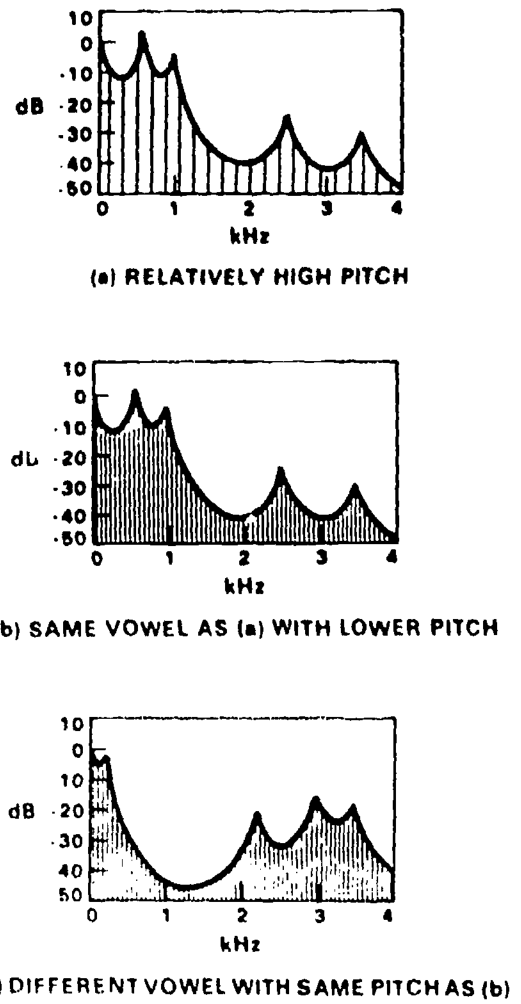


Figure 3. Line spectra showing fundamental frequency, harmonics, and formant frequencies for higher- and lower-pitched vowels.

In the sections which follow, we shall discuss several types of vocoders, and we shall consider specific experiments which have been, or could be, conducted using these vocoders. Although the channel vocoder will receive the greatest amount of attention, we do not mean to suggest that it is better suited to speech research than are the other types of vocoders. In reality, a number of the experiments discussed below could be carried out with any of the vocoders under consideration. (In fact, the selection of one type of vocoder over another may simply reflect the personal preference of the researcher.) However, to the extent that each vocoder utilizes different algorithms for analysis and synthesis, each will produce speech sounds of somewhat different quality. And, more importantly, each vocoder will control the synthesized output in a specific way, permitting different types of manipulation. Thus, it may be that one type of vocoder is somewhat more effective for the exploration of certain research issues (e.g., perturbation of the spectrum or speech produced in noise) than another.

Channel Vocoder

One typical analysis-synthesis speech coder is the channel vocoder, presented schematically in Figure 4. In the channel vocoder, an analyzer consisting of a bank of filters splits the speech spectrum into discrete frequency ranges whose energies are measured often enough to keep up with the speaker's rate of articulation (Tierney et al., 1964; Gold et al., 1983). When the signal is synthesized, its spectrum is recreated using a similar filter bank. In the measurement of the signal needed to recreate the sound source of the original signal, the waveform can be examined directly, or the signal spectrum can be analyzed. If the waveform is used, only a low-frequency portion is needed to determine if the source is the vocal cords (as in a voiced sound, such as [d] or [a]), if it is a noise-producing constriction (as in a voiceless sound, such as [t] or [s]), or if it is an unconstricted egressive air flow (as in [h]).

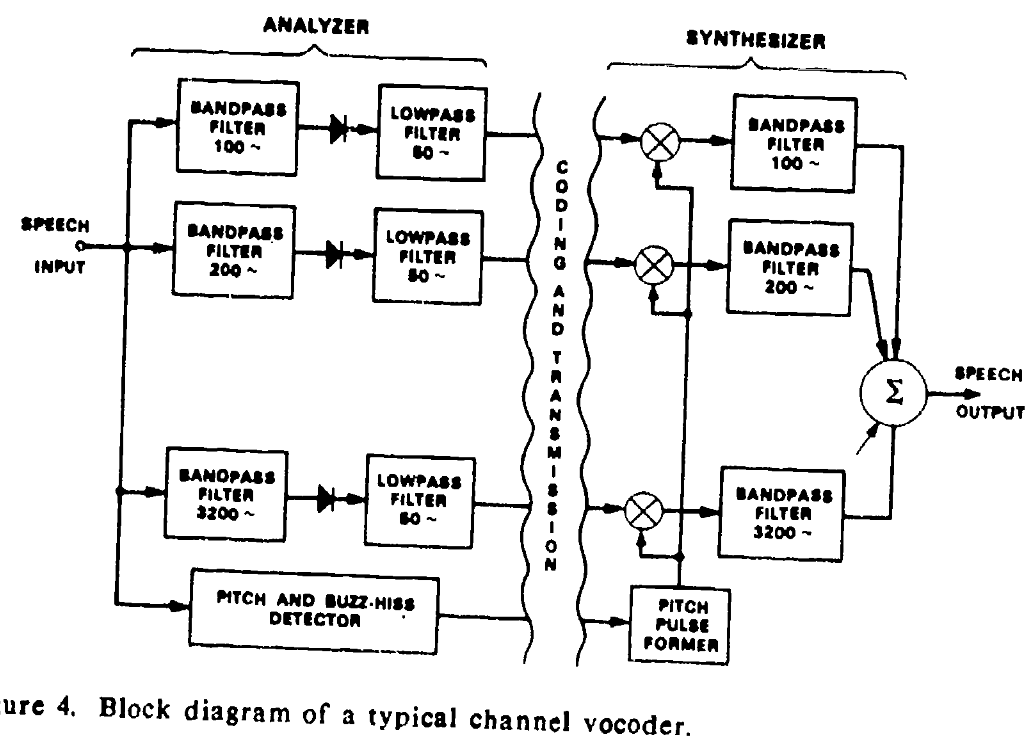


Figure 4. Block diagram of a typical channel vocoder.

With the spectrum envelope (whose frequencies approximate those of the vocal tract) and the spectrum fine structure (whose frequencies approximate those of the sound source), it is possible to recreate a speech signal by measuring the pertinent parameters rapidly (every 10-15 msec). What we cannot retain and recreate with this model is the detailed timing relationship between the different frequency components; that is, we measure only the energy in bands, not the relationship between the harmonics which constitute the signal. However, if the filter bank has sufficient resolution--i.e., if the frequency range of its filters is narrow enough--and if the source determination is sufficiently accurate, we can still obtain recreated speech of excellent quality.

The channel vocoder has a number of interesting applications in speech research, and several experiments have been conducted recently which illustrate some of these applications. For example, the sound source itself is of interest. The pitch signal used in the synthesizer contains very little information about the vocal tract, but it characterizes the fundamental frequency (F0) of the speaker's voice. The waveform of this signal can be used as a basis for pitch discrimination experiments in which the fundamental frequency is altered in various ways. A simple distortion can, for example, be obtained by adding or multiplying the F0 by a constant. The resulting pitch signal can then be passed through the vocal tract filter to determine what the speech sounds like with this distortion. One obvious application of this procedure involves the separation of source and filter. A researcher might wish to examine the perceptual effect of a high-pitched (e.g., female) voice being produced by a large (e.g., male) vocal tract. In doing so, one could thus ascertain the relative perceptual salience of fundamental and formant frequencies in signaling the gender of a speaker and one could determine what kind of trading relation exists between these two acoustic cues.

Another application involving manipulation of the sound source may be found in experiments in which listeners' ability to discriminate pitch is observed. Such an experiment has been conducted by Mack and Gold (1984; 1986). In this experiment, listeners were presented with three different types of stimuli consisting of simple pulse trains (non-speech stimuli with some acoustic complexity), monotone sentences, and sentences with natural intonation. For the generation of the sentence stimuli, Mack and Gold used a high-quality channel vocoder to produce sentences whose F0 differed from 0 to 6 Hz in .5 Hz increments. Subjects were presented with pairs of stimuli and were required to indicate whether the two members of each pair had identical or different pitch. Results are presented in Figure 5.

As is apparent from the figure, pitch discrimination, as determined by the percent of stimulus pairs judged as different, proved most accurate in pulse trains, and least accurate in naturally intoned sentences. These results suggest that the presence of linguistic information results in a relative decrement in listeners' sensitivity to fine differences in overall F0. Moreover, when a speech stimulus has a constantly changing F0 (as natural speech does), overall differences in the pitch of sentence pairs are especially difficult to detect. For, as may be seen in the figure, subjects performed near chance (with only about 50% of the pairs judged as different) in the natural-intonation condition, even in response to those stimulus pairs whose pitch was most dissimilar.

Mack and Gold used the same vocoder in a pilot experiment to determine listeners' responses to another type of permutation in the sound source. In this experiment, they generated pairs of sentences in which one member of the pair had

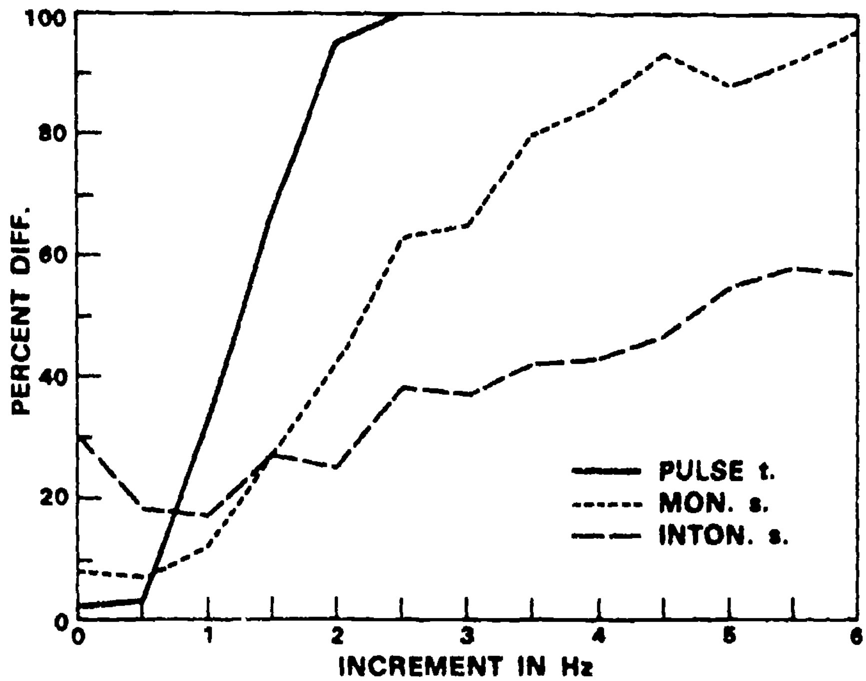


Figure 5. Discrimination functions for three types of stimuli--monotone pulse trains (PULSE t.), monotone sentences (MON. s.), and naturally intoned sentences (INTON. s.).

natural intonation contour and the other had a contour which was linearly interpolated in increments of 20 to 140 msec. In Figure 6a, a schematic of this interpolation process is presented. As this figure illustrates, the F0 of the signal was linearly interpolated between two points, rather than being allowed to fluctuate in a natural manner. In a discrimination task using these stimuli, subjects were required to indicate whether they perceived a difference between the members of a stimulus pair in which one member had a natural intonation contour and the other an interpolated one. The objective of this study was to determine whether or not subjects would find sentences with interpolated pitch segments acceptable. Results of this experiment were, unfortunately, somewhat unclear due to the fact that some of the listeners found it extremely difficult to discriminate between the stimuli. Still, the trend was that subjects perceived a difference between the two types of stimuli, but only when longer interpolations were used (see Figure 6b). Thus, this experiment suggested that listeners seem to be relatively insensitive to changes in F0 provided that the overall pitch contour of a sentence is maintained--a finding in agreement with that of Collier (1975), t'Hart (1981) and de Pijper (1983).

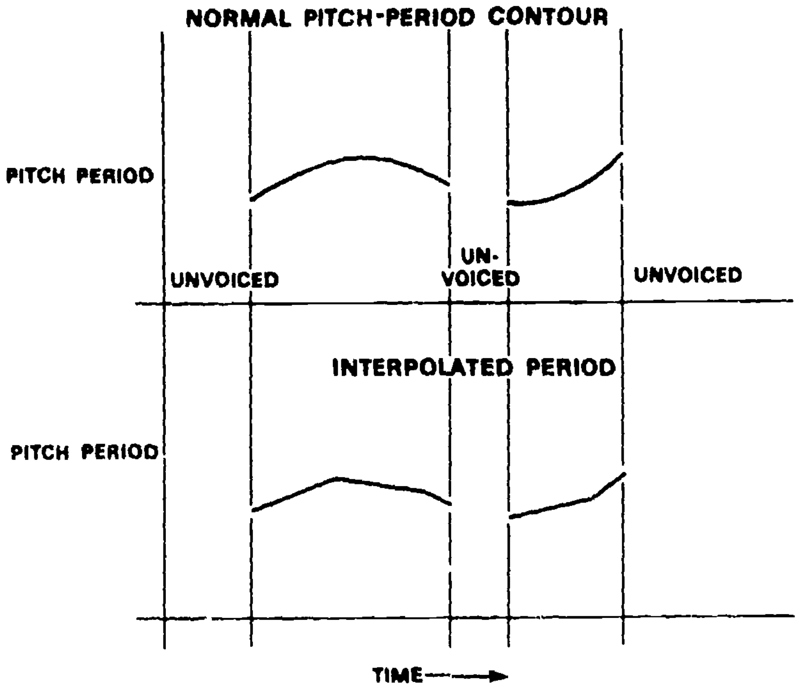


Figure 6a. Schematic of normal pitch-period contour and linearly interpolated pitch-period contour.

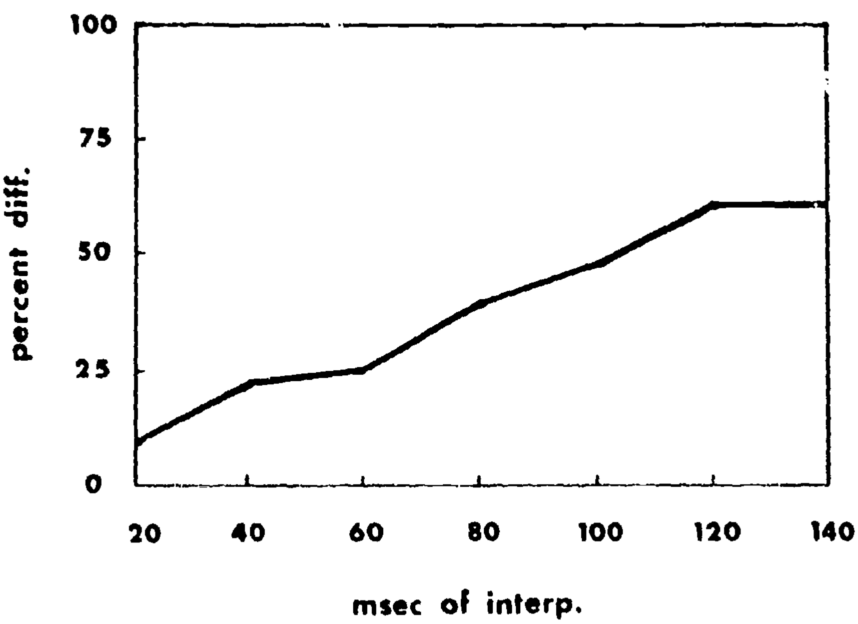


Figure 6b. Average discrimination function for pitch-interpolation experiment.

An additional experimental application of vocoders involves examination of the actual effect of vocoding itself. In a recent study (Mack and Gold, 1985), the intelligibility of speech in four conditions was analyzed. The conditions included (1) natural (non-vocoded) speech; (2) vocoded speech; (3) vocoded speech with noise added to the pitch track; and (4) vocoded speech with noise added to the spectrum. Speech stimuli consisted of 57 sentences presented auditorially to English monolingual subjects. All sentences were semantically anomalous--i.e., grammatical but meaningless. (An example of such a sentence is, "Simple rocks fear crooked hats.") Subjects were required to write each sentence as accurately as possible immediately after they had heard it. Analysis of the errors revealed that subjects made over twice as many errors in the vocoded as in the natural-speech condition, in spite of the fact that the vocoded speech was of extremely high quality. Moreover, they made numerous errors in response to vocoded speech with noise added to the pitch track, and they made even more errors when noise was added to the spectrum. Mack (1987) also used natural and vocoded anomalous sentences in a comparative study of English monolinguals and German-dominant German-English bilinguals. Analysis of the two groups' performance revealed that both found the vocoded sentences less intelligible than the natural ones. However, the bilinguals had an especially high error rate (even in response to the natural sentences, on which they made an average of six times as many errors as the monolinguals did). These findings reveal, in some detail, the effect of vocoded speech on perception--effects which might not be apparent in tests whose perceptual demands are slight.

Due to the fact that the analysis-synthesis vocoding procedure permits researchers to extract and manipulate source and filter information independently, there is another type of signal permutation--time expansion and compression--which has considerable potential in perception research. Specifically, an interesting consequence of time expansion and compression is that the temporal quality of an utterance can be changed without altering its fundamental or formant frequencies. To appreciate the significance of this, it is only necessary to consider what usually happens when a speech signal is played on a tape recorder at a faster or slower rate than that at which it was recorded. The speed of all speech events is changed and, as a result, the fundamental frequency of the speech is changed as well, resulting in an unnatural and high-pitched "chirpy" quality in the rapid speech and in a low-pitched "drunken" quality in the slow speech. While vocoding does permit one to make often dramatic speech rate changes in the output, the unnatural effects just described do not obtain, as is demonstrated in Figure 7. That is, the speed of the speech articulators is effectively decreased, but the frequencies are not changed. The resulting speech remains highly recognizable and the F0 is unperturbed. This procedure has considerable research potential and can be used in studies in which speech rate must be varied independently of frequency changes (which may, at the perceptual level, interact with subjective judgments about rate).

Homomorphic Vocoder

Another type of vocoder is the homomorphic vocoder. This vocoder derives source and vocal tract parameters from the spectrum signal alone (see Appendices 1a and 1b[2]). Assuming that the sound source and vocal tract spectra do not overlap badly, the logarithm of the spectrum can be used to turn a product into a sum. That is, if the parameters of the vocal tract (the variable acoustic filter) do not change rapidly relative to the sound source, then the vocal tract filter can be separated from the overall speech signal (Oppenheim, 1969). The synthesis procedure for this kind of analysis-synthesis is somewhat different from that of the channel vocoder. A filter bank is not used for recreating the vocal tract filter.

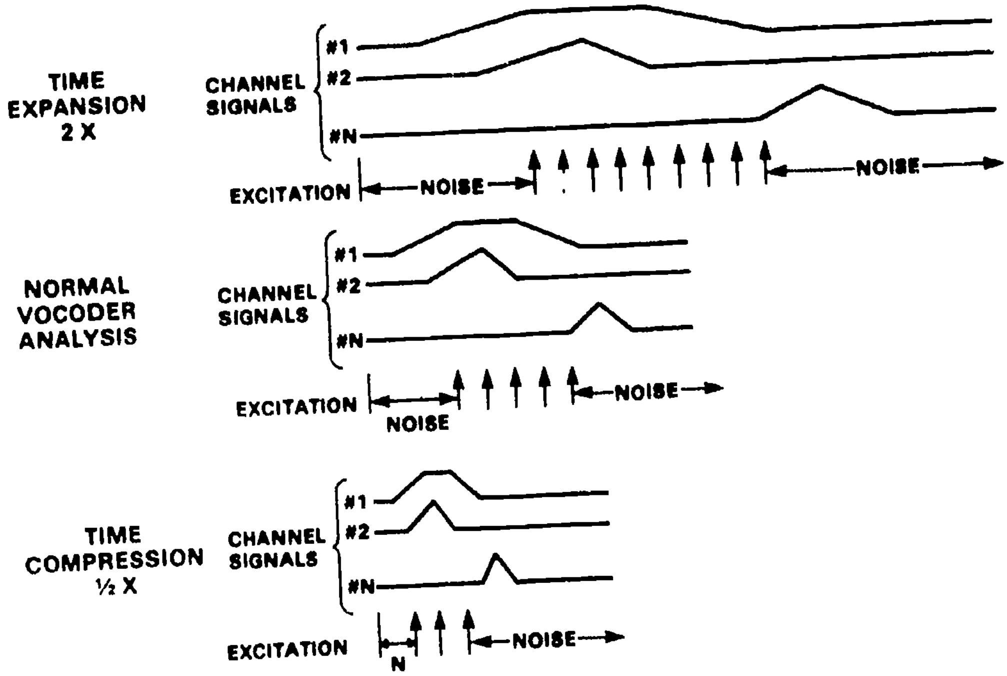


Figure 7. Channel vocoder time expansion and compression.

Instead, a filter based upon transforming the vocal tract spectrum is used to operate on the energy source. This type of analysis-synthesis vocoder has been used to produce acceptable synthetic speech and can thus be used in the same ways as a channel vocoder to produce transformed speech outputs for various communications and research applications.

Linear Predictive Coding Vocoder

Currently, the vocoder which is used most often for speech transmission is the linear predictive coding (LPC) vocoder (see Appendix 2). With this vocoder, the speech signal is generated from the same mechanism and with the same source determination as in the channel and homomorphic vocoders. But with the LPC vocoder, the vocal tract measurement is made by fitting the speech waveform in some interval (usually a frame about 20 msec wide) to the output waveform of a linear filter (Makhoul, 1975).

Fitting a linear filter to the speech waveform can be an efficient process from a computational point of view. As a result, the LPC vocoder is most popular for encoding and transmitting speech at the standard data rate of 2400 bits/sec--the rate used, for example, on a telephone-line modem. One drawback of the LPC vocoder is that, because of the way in which the vocal tract filter is described, the frequency spectrum is not explicitly present for modification. This means that only a small set of transformations can be implemented or that additional calculations are needed to compute the vocal tract spectrum parameters. Instead of presenting the frequency spectrum explicitly, this vocoder describes the inverse filter impulse response which is used to compute the vocal tract spectrum. This filter description can be used to manipulate certain parameters of the signal directly. For example, it can be utilized to "sharpen" formant frequencies (i.e., to reduce their effective bandwidths) or to reverse the phase of a waveform. If it is necessary to modify only the source signal (as is done in a pitch-perturbation experiment), then the fact that the spectrum is not explicitly present causes no problems. The LPC vocoder may represent the simplest way to implement an analysis for generating modified speech signals. Hence, its application in speech research is potentially great.

Sinusoidal Transform Coding Vocoder

Unlike the vocoders described above, a sinusoidal transform coding (STC) vocoder does not treat the sound source as a periodic pulse train (for voiced sounds) or as a noise signal (for voiceless sounds). Rather, it characterizes the sound source as a sum of sinusoidal waves which vary with respect to one another (Quatieri and McAulay, 1985). In analysis, the peaks of a short-term spectrum measurement are used in specified time-frame intervals to characterize the speech signal sinusoids (see Appendix 3). This "peak picking" occurs in both voiced and voiceless signals. Thus, this coding technique does not separate the sound source into "buzz" (voicing) and "hiss" (voicelessness), one of the more difficult measurement decisions to implement. At all times, the sound source is represented as a sum of sine waves. Provided that a sufficient set of peaks is selected, the set of parameters measured at every frame time characterizes the speech signal completely.

One problem which must be solved in the application of this technique involves matching the measurements made from one frame to the next. It is necessary to match these measurements so that continuous parameter tracks will be generated, i.e., so that the synthesized speech will sound continuous, as natural speech does. A solution to this problem has been developed. It involves an algorithm for a frame-to-frame matching procedure in which the frequencies in frame k are lined up as closely as possible with the frequencies in frame k+1. If there is a peak in frame k that is not matched with a peak in frame k+1, the peak in frame k is allowed to "die." (That is, a zero-amplitude peak of the same frequency is assigned to frame k+1.) Likewise, if there is a peak in frame k+1 that has no counterpart in frame k, its "birth" is assumed. That is, it will be assumed that a zero-amplitude peak of the same frequency started in frame k.[3]

As mentioned in the discussion of the channel vocoder above, it is possible to utilize time expansion and compression with the STC vocoder. However, with this vocoder, time expansion and compression are somewhat more accurate than with the channel vocoder. With such a procedure, it is possible to use speech features themselves to control the time warping so that specific segments of the signal are automatically perturbed or left intact. For example, time expansion (resulting in a slowed speech signal) could occur everywhere in a speech signal except during the production of plosive sounds, such as the stop consonants /t/ and /k/. If these sounds _are_ included in the time expansion, the result is speech which sounds as if the speaker is having difficulty articulating (see discussion of time expansion in the section on channel vocoders above). However, if the time expansion does not include these sounds, the speech sounds normal but deliberate. This kind of selective transformation of the signal can also be effected by hand-editing a displayed waveform of an utterance. This method would probably be employed in gen rating stimuli for a perception experiment, since it provides greater control of the transformed utterance than automatic feature-driven time expansion does.

One obvious application of the STC coding procedure in speech research involves utilizing time expansion and compression to determine listeners' sensitivity to rate changes in different segments, such as vowels versus consonants, and in different parts of utterances, such as sentence-initial and sentence-final position. One advantage of using vocoded stimuli here is that fully elaborated sentential and contextualized stimuli could be generated quite easily, enabling one to assess the effect of rate changes in relatively naturalistic utterances.

CONCLUSION

In the present article, it has not been possible to discuss all types of vocoders in use.[4] However, we have presented brief descriptions of four of these vocoders--four which have already been used or could be used in psychophysical and speech-perception research. For, although vocoders were originally developed for the coding and transmission of speech signals, the techniques employed for deriving and transforming these signals can easily be used to produce stimuli for probing listeners' perception. The low data-rate vocoders described above seem especially well suited to applications in language research. Some of the speech-processing techniques we have described are more complex than others, but all have been simulated and implemented in current digital computing facilities--and all are potentially accessible to the interested researcher.

ACKNOWLEDGMENTS

This work was sponsored by the Department of the Air Force. The views expressed are those of the authors and do not reflect the official policy or position of the U.S. Government.

THE AUTHORS

Joseph Tierney is a staff member at Lincoln Laborabory, Massachusetts Institute of Technology. His major interests are in speech coding, signal processing, and auditory perceptual mechanisms.

Molly Mack is Assistant Professor in the Division of English as an International Language at the University of Illinois, Urbana. She has worked as a speech researcher and consultant for Lincoln Laboratory, Massachusetts Institute of Technology. Her primary interests are in speech perception and production and in the psycholinguistic and neurolinguistic aspects of bilingualism.

NOTES

[1]Hertz (Hz) is a unit of frequency equal to one cycle per second. For example, if an object vibrates 256 times per second (as does the string on a piano to produce a note of middle C) its frequency is 256 Hz. This value is the fundamental frequency (F0) of the sound. In addition to the F0, a complex sound

such as a piano note (or the human voice) has frequency components or harmonics, each of which has a frequency which is n times that of the F0. Thus, middle C has harmonics at 512 Hz, 768 Hz, 1024 Hz, etc. Further, any complex periodic waveform can be analyzed as the sum of a set of harmonically related sinusoidal (sine) waves. A sine wave appears on an oscilloscope as a simple smooth wave, such as that produced by a tuning fork. The analysis of a complex waveform as the sum of harmonically related sine waves is a fundamental principle of the mathematical procedure called Fourier analysis.

While the fundamental frequency may be considered an acoustic property of the sound source, formant frequencies may be considered acoustic properties of the filter. Formant frequencies (or resonances) are those frequencies at which maximum energy passes through a filter. For example, the first three formant frequencies for [a] produced by an adult male vocal tract are, on the average, 730 Hz, 1090 Hz, and 2440 Hz (Peterson & Barney, 1952).

[2]These and the following figures appear in the appendices. These figures consist of block diagrams of particular vocoders and they are designed to demonstrate (e.g., to a speech-systems computer programmer) how the vocoder algorithms are implemented.

[3]The frame-to-frame frequency matching technique does not, of itself, solve the entire problem of providing a continuous synthesized speech signal. The phase relations must also be interpolated smoothly. Discussion of the procedure for doing so is beyond the scope of this article.

[4]Other types of vocoders being utilized at the MIT Lincoln Laboratory and elsewhere include mid data-rate systems such as the sub-band vocoder and the voice-excited vocoder. In addition, several new techniques for manipulating speech have been developed (Seneff, 1982; Portnoff, 1981). Although the resulting systems produce speech of excellent quality, they are costly in terms of computational complexity compared to the systems described in this article

REFERENCES

Collier, R. (1975). Perceptual and linguistic tolerance in intonation. International Review of Applied Linguistics, 13, 293-308.

Flanagan, J.L., Schroeder, M.R., Atal, B.S., Crochiere, R.E., Jayant, N.S., & Tribolet, J.M. (1979). Speech coding. IEEE Transactions on Communications, COM-27, 710-737.

Gold, B., Lynch, J., & Tierney, J. 1983. Vocoded speech through fading channels. Proceedings of ICASSP, 83 101-103.

Luce, P.A., Feustel, J.C. & Pisoni, D.B. (1983). Capacity demands in short-term memory for synthetic and natural speech. Human Factors, 25, 17-32.

Mack, M. In press. The intelligibility of natural and vocoded semantically anomalous sentences: a comparative analysis of English monolinguals and German-English bilinguals. M.I.T. Lincoln Laboratory Technical Report.

Mack, M. & Gold, B. (1984). The discrimination of pitch in pulse trains and speech. M.I.T. Lincoln Laboratory Technical Report, 680, DTIC No. AD-142996/8.

Mack, M. & Gold, B. (1985). The intelligibility of non-vocoded and vocoded semantically anomalous sentences. M.I.T. Lincoln Laboratory Technical Report, 703, DTIC No. AD-A160401/6.

Mack, M. & Gold, B. (1986). The effect of linguistic content upon the discrimination of pitch in monotone stimuli. Journal of Phonetics, 14, 333-337.

Makhoul, J. (1975). Linear prediction: A tutorial review. Proceedings of the IEEE, 63, 561-580.

Miller, G.A. and Nicely, P.A. (1954). An analysis of perceptual confusions among some English consonants. Journal of the Acoustical Society of America, 27, 338-352.

Oppenheim, A.V. (1969). A speech analysis system based on homomorphic filtering. Journal of the Acoustical Society of America, 45, 459-462.

Peterson, G.E. & Barney, H.L. (1952). Control methods used in a study of the vowels. Journal of the Acoustical Society of America, 24, 175-184.

de Pijper, J.R. (1983). Modelling British Intonation. Dordrecht: Foris Pub.

Pisoni, D.B. & Hunnicutt, S. (1980). Perceptual evaluation of MITalk: The MIT unrestricted text-to-speech system. IEEE International Conference Record on Acoustics, 572-575.

Pisoni, D.B., Nusbaum, H.C. & Greene, B. (1985). Perception of synthetic speech generated by rule. Proceedings of the IEEE, 73, 1665-1676.

Portnoff, M.R. (1981). Time-scale modification of speech based on short-time Fourier analysis. IEEE Transactions, ASSP-29, 3, 374-390.

Quatieri, T.F. & McAulay, R.J. (1985). Speech transformation based on a sinusoidal representation. Proceedings of ICASSP, 85, 489-492.

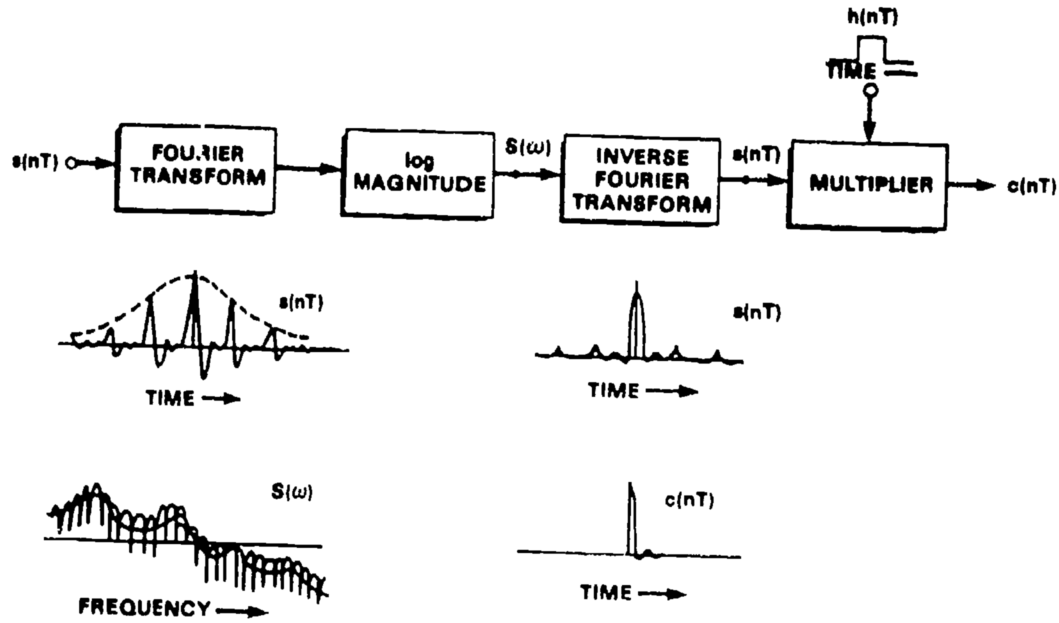
Seneff, S. (1982). System to independently modify excitation and/or spectrum of speech waveform without explicit pitch extraction. IEEE Transactions, ASSP-30, 4, 566-578.

t'Hart, J. (1981). Differential sensitivity for pitch distance, particularly in speech. Journal of the Acoustical Society of America, 69, 811-821.

Tierney, J., Gold, B., Sferrino, V., Dumanian, J.A., & Aho, E. (1964). Vocoder with digital pitch extractor. Journal of the Acoustical Society of America, 36, 1901-1905.

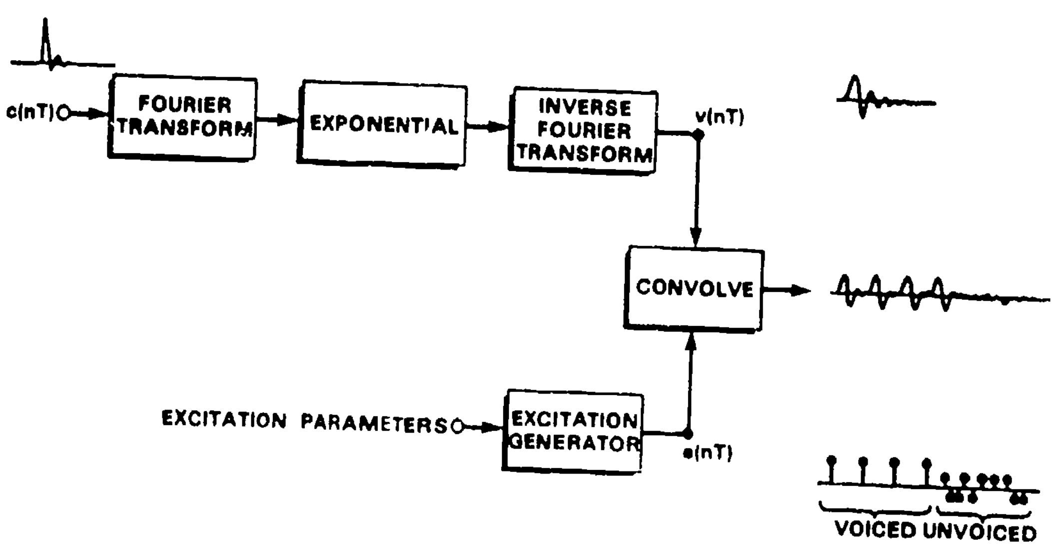
APPENDIX 1a

Homomorphic vocoder: Speech analysis (spectrum separation).



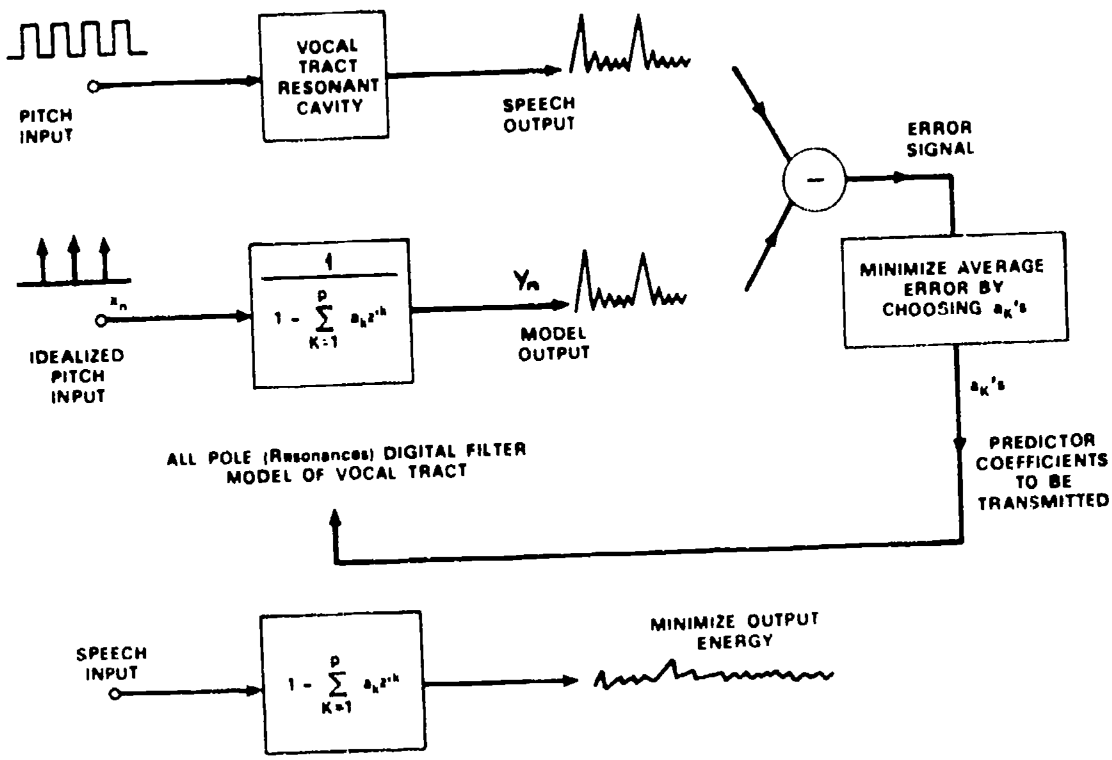
APPENDIX 1b

Homomorphic vocoder: Speech synthesis.



APPENDIX 2

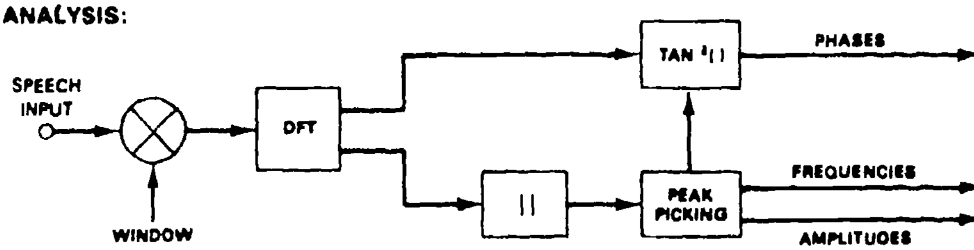
Linear predictive coding (LPC) vocoder: Filter determination.



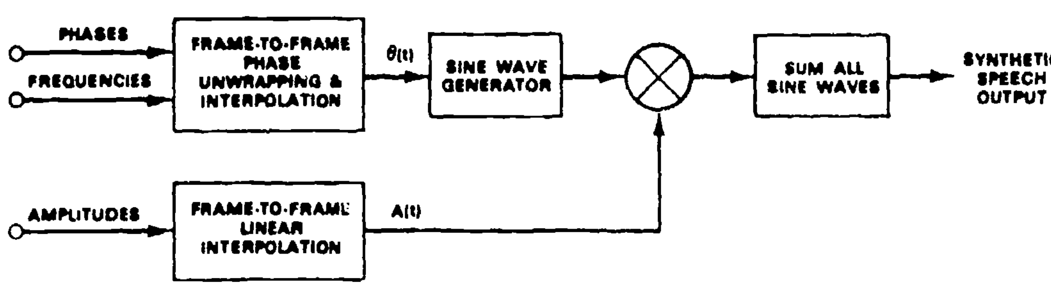
APPENDIX 3

Sinusoidal tranform coding (STC) vocoder: Analysis and synthesis.

ANALYSIS:



SYNTHESIS:



CURRENT TRENDS IN COMPUTER-BASED LANGUAGE INSTRUCTION

Robert S. Hart

Current computer-based language lessons are weak in the areas of error analysis and feedback, communicative realism, and convenient lesson authoring. This paper reviews research underway to improve lesson quality in each area. The best hope for systematic grammar-sensitive error analysis seems to be based on parsing techniques borrowed from natural language processing. Both simulation designs and current work with interactive video offers greater communicative realism. One approach to convenient lesson authoring involves adaptation of word processing techniques to create analogies with textbook authoring.

It is not news that the attempt to make computer-based language instruction into a national educational reality faces a wide range of problems. Most schools now have computers, but sometimes only one or two per hundred students. When there are machines, foreign language students are often denied access in favor of mathematics and science classes. Textbook publishers are jumping into the language courseware market, but they want to keep costs down, and the bulk of what they do falls into the category of electronic workbook exercises. Freelance authors write for the lowest common denominator of hardware in order to increase their market. There is powerful new hardware, but it is not clear that most schools will be able to afford it any time soon. And, to complete this litany of difficulties, many language teachers simply do not believe that computer-based instruction really improves on the textbook and paper-and-pencil techniques it is designed to supplant.

This skeptical attitude of teachers, which arguably accounts in large part for the remaining problems, cannot be shrugged off simply as the results of ignorance, conservatism, or technophobia, since those same teachers have been quite willing to adopt computer technology (for example, word processing) when it is clearly superior to the pre-computer alternative. Most foreign language teachers, by inclination as well as necessity, are connoisseurs of new teaching methodologies, and if they are not convinced that computer-based language instruction is a superior tool it is probably because the evidence just is not there.

Current courseware does in fact lack features which most teachers consider crucial to efficient language learning, notably:

Error analysis and feedback: Teachers want programs that identify students' written grammar errors, and that mark them the way that paper-and-pencil exercises are marked.

Communicative realism: Teachers would like more realistic lessons which make interaction with the lesson seem like a real language use situation.

Convenient lesson authoring: Existing options for creating lessons are cumbersome. Teachers need an "electronic ditto machine," useful for quickly manufacturing exercises targeted directly at their own classes and syllabi.

As long as these features are lacking, most teachers are simply not going to fight to put language courseware into their classrooms, no matter how hard the sales pitch. Fortunately, new microcomputer hardware and software now commonly available provide an improved set of tools for attacking these problems and a great deal of work intended to remedy these shortcomings is now underway. The remainder of this paper is devoted to reviewing some of this work.

ERROR ANALYSIS

Whenever the student has to type a response longer than a word or two, the possibility of multiple grammar errors exists. Feedback which locates and identifies the errors actually present should (presumably) lead to more efficient learning than the ever-popular "no, try again," or some generic hint at the correct answer. Instructors have been requesting this capability nearly as long as language courseware has existed, but attempts to provide it have all fallen short in various ways.

The PLATO system "error markup" (Figure 1) typifies one approach to feedback. This scheme describes errors in terms of editing operations such as insertion of extra words, omission of required words, movement of words to the right or left of their correct condition,

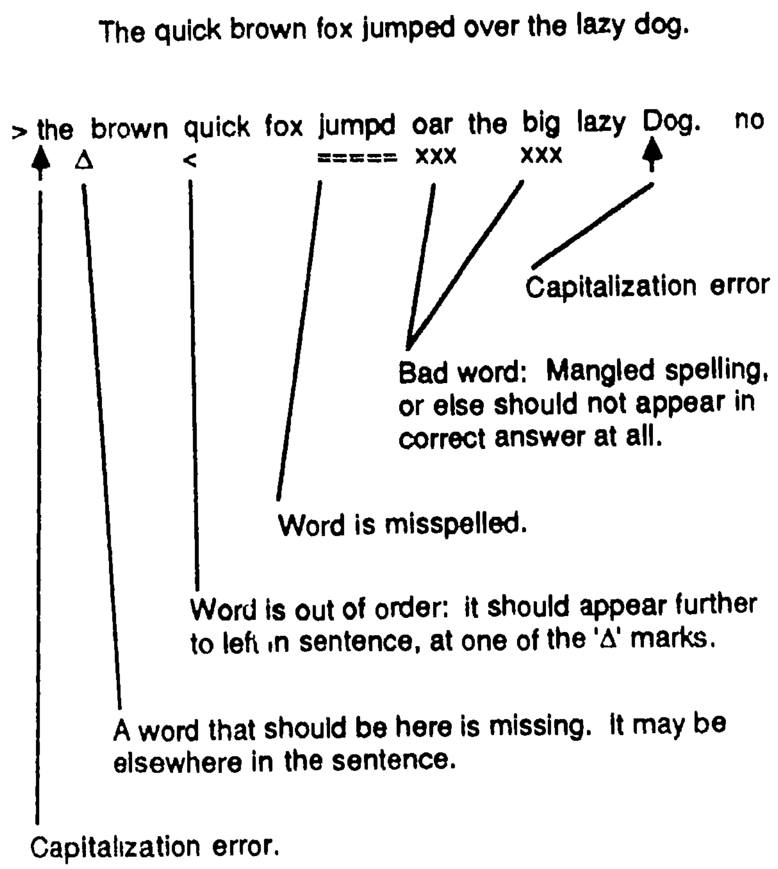


Figure 1. PLATO error markup

incorrect upper- or lower-case letters. The kinds of errors present are shown by a standard system of graphic markup symbols. Other programming languages go even further in this direction. For example, EnBASIC[1], a dialect of BASIC designed explicitly for application to computer-based instruction, provides special markup symbols for accent errors and for insertions, omissions, or transpositions of individual letters. The popular DASHER[2] authoring system uses a scheme similar in spirit, but points of disagreement are indicated by dashes or carets.

In all these systems, the error markup is done automatically as a normal part of response evaluation, requiring no attention from the lesson author. The meaning of the graphic symbols can be learned quickly. A good student can often get sufficient information from the markup to diagnose and correct the grammar errors. In some instructional contexts, this is sufficient. For language teachers who teach explicit grammar rules, however, this style of error analysis has a severe drawback, since it makes no reference at all to the rules implicated in each error, and in fact marks errors just the same for Swahili or Russian as for English. The student is left to infer which grammatical rule was violated, and this is often expecting too much even of a good student.

One obvious way to accomplish this is to anticipate the incorrect structures which might occur in the student's response and search for each of them. In the following example, adapted from Marty's (1981) *Written French* series on PLATO IV, the student is asked t translate the English sentence

She translated the last ten pages for us.

Here the corrected answer is

Elle nous a traduit les dix dernières pages.

Suppose that the student has responded by typing

Elle a traduit le dernier dix pages pour nous.

Marty's program conducts the error analysis by searching through the student's response for each of the strings at the left, and by giving the error message on the right if it is found:

dernières dix	*Order of Adjective / cardinal number*
pour nous	*Prep + pn --> indirect object pn*
dernier	*Gender error: "page" is feminine*

Notice that the items searched for are literal strings. The error messages are telegraphic descriptions of grammar rules used in Marty's textbook. A student who finds them too cryptic can press the HELP key to receive a more detailed explanation.

From a language teacher's viewpoint, the anticipated error approach provides some nice features lacking in the PLATO error markup, since grammar errors can be identified according to whatever taxonomy the author prefers, and feedback messages can be given whatever phrasing he or she wishes. However, this way of proceeding also entails some serious disadvantages. For one thing, the author must hand-analyze each item individually in order to specify anticipated errors--a tedious and expensive undertaking if the volume of material is large. More fundamentally, different kinds of errors within a single item can interact in such a way that specifying all possible combinations causes a combinatorial explosion. As a result, string search is practical only when the answer is very short or when the author specifies just a few of the most important errors.

The risk of combinatorial explosion is alleviated when pattern descriptors of greater generality are available, as is the case, for example, when a string processing language like SNOBOL is used to program the analysis. But this is at best an evasive maneuver. The real problem with string matching is that a program written this way does not have enough knowledge about the grammar or morphology of the target language, or about typical student errors, to produce an intelligent error description.

The ideal error analysis algorithm would mark an item, as much as possible, the way a teacher would mark it on paper, but do it automatically, without requiring a hand analysis. This implies that the algorithm should have the following three properties:

1. It should be *grammar specific*, identifying the particular grammar rules implicated in various types of errors;

2. The error messages actually shown to the student should be *adaptable to the teacher's preferred terminology* and presentation conventions.

3. It should be *completely automatic*, requiring no item-by-item programming.

The third property is not strictly necessary, but it is very desirable since it makes implementing new materials much faster and easier, hence less expensive.

Although Levine (1973) did some early exploratory work on this approach, no one seems to have followed his lead. Grammar sensitive error analysis has been successfully implemented in working instructional programs only for rather limited domains of of morphology, for example, Cully's (1979) Latin materials, originally done on the Delaware PLATO system but since adapted to the IBM-PC, and Davidson's Russian verb lesson[3] on the IBM-PC. Although several projects are under way, there is still no program which handles a large set of morphology and syntax.

The Language Learning Laboratory at the University of Illinois, Urbana-Champaign, is now building a system intended to approximate this ideal. A general solution clearly requires incorporating information about the target language grammar, which we accomplish by borrowing techniques from natural language parsing and adapting them for the case of ill-formed input.

We require (as do the authors of all the other schemes discussed above) that a model or correct answer be available beforehand. Analysis of free responses, where no correct answer is available, would require the program to construct an appropriate model before performing the error analysis proper. Determining "what the student was trying to say" generally requires the integrated use of grammatical, semantic, and pragmatic information and presents very difficult theoretical problems. In addition, to do such computations dynamically will likely require computing resources far exceeding those available to individual students in the foreseeable future. To obtain further simplification, we also require that the desired error descriptions be obtainable from grammatical analysis alone, possibly supplemented by some knowledge about what kinds of written errors language learners are likely to make; no semantic information is utilized.

The system, schematized in Figure 2, is now being implemented on an IBM-AT using GOLDEN COMMON LISP. The target language in this pilot version is French. The two inputs are the model, that is, the correct answer, and the student's response. The correct answer, which is assumed to be well formed, is first submitted to a morphological analysis, then parsed using standard parsing techniques. The output is a parsing tree with feature-labelled nodes, plus a table of role registers which label the grammatical functions of various constituents.

The student's response is also morphologically analyzed, then parsed, but since it may be ill-formed, the parsing must be highly error tolerant. Several means are employed to obtain such tolerance. First, the parser ignores most inflectional morphology and parses mainly in terms of roots; second, deviant rules which represent common syntactic misconstructions are included in the grammar. (When an incorrect construction is identified, it is of course marked as such.)

Morphological analysis utilizes a root dictionary and an inflection dictionary, both containing morphological and syntactic features. To assure that incorrect inflections can be identified and remediated, these dictionaries must contain more than the usual amount of information about inflectional morphology. Specifically, relational links are introduced "upward" to root (citation) forms, and "downward" from each root to its various stems and suppletive forms, including (when appropriate) ill-formed variants. This allows, for example, recognition that the verb *allent* in *Ils allent au cinéma* is an analogical (but incorrect) formation, and that it corresponds to the correct form *vont* in the correct answer *Ils vont au cinéma*. The latter information is necessary for successful matching at the next stage.

The two parsing trees are then submitted to a matcher which is responsible for matching up constituents from the two trees and for identifying and describing deviations from a perfect match. Since the trees may vary greatly in structure, matching proceeds bottom-up, starting with the heads of the various constituents to secure reliable correspondences. When a direct, word-for-word match cannot be made, the matcher can try various rearrangement rules, each of which corresponds to a particular type of error. For example this matching rule

((?X pn io) = (pp ("pour" prep) (?X pn));

("*pour* should not be used to indicate a pronoun io")) .

states that an indirect object pronoun in the model may be matched with a suitable prepositional phrase in the student's response, thus identifying errors of the type

...nous a traduit ...	(model)
a traduit ... pour nous	(student response)

A suitable error message can be associated with each matching rule. For example, the prepositional construction is judged by some French speakers to be not truly incorrect. If the instructor chooses to take this position, the error message could be modified to read *"Nous a traduit* is also acceptable and might be preferable," or something of the sort. Intra-sentence errors, such as concordance errors, which do not require reference to the model, can be handled by examining patterns within the response parsing tree.

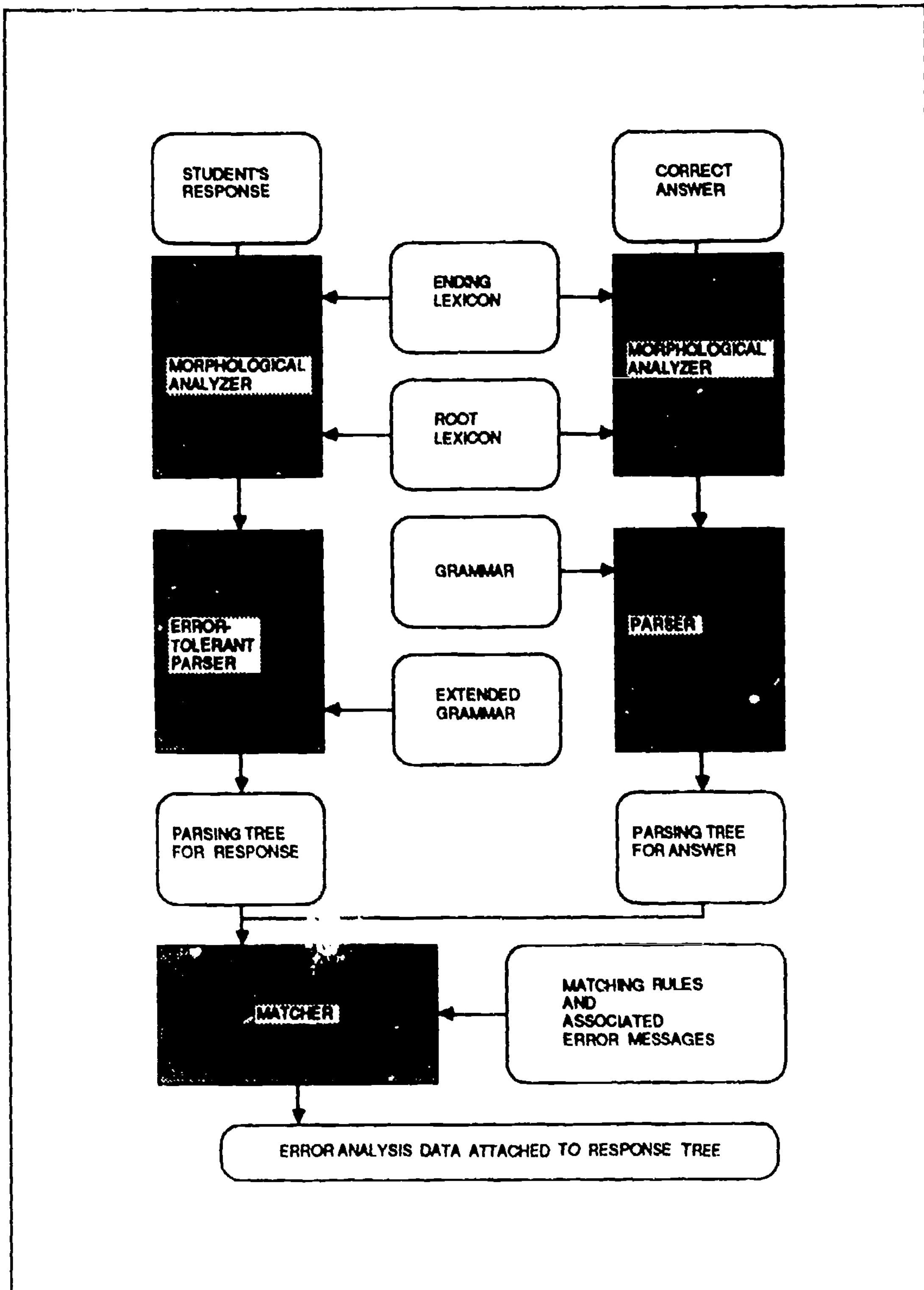


Figure 2. Block diagram of error analysis program.

The approach outlined here has decisive advantages over simple string-matching techniques. The fact that constituent structure is available to control matching makes mismatches less likely, and pattern descriptions of greater abstractness can be specified to the matching algorithm, allowing for a more compact and perspicuous statement of relevant grammar. A dictionary and grammar must of course be created, something not required for string matching, but they need be implemented only once for each language. Considerable flexibility in the phrasing of feedback can be obtained by simply changing the messages attached to the various matching rules. If this does not provide sufficient tailoring to individual pedagogical approaches, the error taxonomy can be redefined by changing the matching rules, without rewriting the dictionary or parser. A reasonable degree of modularity in the program design is thus maintained.

Several groups, for example those involved with the Athena project (Kramsch et al., 1986; Morgenstern, 1986), are working on the problem of providing grammatically sensitive error feedback. It is virtually certain that the next few years will see functioning error analysis systems.

A REALISTIC COMMUNICATION ENVIRONMENT

The phrase "realistic communication environment" simply means that computer lessons should use language in the same w, a correspondence which has several aspects:

> *Conversational use*: The computer responds to and uses language the way a human being would, meaningfully, and with regard for the rules of conversational interaction.
>
> *Contextual use*: Language is used within a rich auditory and visual environment.
>
> *Communicative use*: Language is used in a meaningful and instrumental way, as part of some ongoing activity.

Encouraged by the communicative competence philosophy, attempts to produce communicatively realistic programs are currently a major theme in language courseware development. Some ambitious projects, such as MIT's project Athena, are trying to work toward all aspects of communicative realism in parallel. A few reservations are in order, however, about the feasibility of conversational language. First, getting a program to really converse within anything other than a trivial domain of discourse, if it can be done at all, will require very advanced artificial intelligence techniques, and hardware resources far in excess of anything likely to be generally affordable for standard instructional use in the foreseeable future. Nor are speech understanding programs likely to accommodate anything close to normal speech any time soon. Writing is not normally a conversational activity, and any program which depends on typing as the sole mode of inputing language is far from ideal as a conversational medium.

Obviously, this does not imply that we should give up trying to develop conversational programs. It does imply however that they will probably remain showcase items for a long time to come. For the moment, other aspects of communicative realism, for instance, meaningful and instrumental use, seem more tractable to computer implementation, and are is likely to yield greater payoff if pursued.

Simulation designs offer a promising approach to the meaningful and instrumental use of language. One well-known foreign-language example is *Cartels & Cutthroats*[4], a French-language economics simulation which has marketing and finance as its domain of discourse. The user is made president of a manufacturing corporation which must compete in the open market. The president is given an initial set of resources and must react to economic events outside his or her control, such as fluctuating interest and inflation rates, price changes by suppliers and competitors. etc. He or she in turn makes budgeting and pricing decisions that will either make or break the company. Several students can play at once, and converse (presumably in French).

Information is both tabular and verbal, the way it would actually reach an executive officer's desk. The user produces very little French, since input is generally numerical, but is exposed to a modest subset of commercial French. And the eighteen page instruction manual is also written in French. Failure to comprehend this French-language information prevents the user from functioning successfully in the simulation and soon results in bankruptcy.

Another example of this sort is *The Would-be Gentleman*[5], which simulates some of the sociology and economics of the France of Louis XIV. The user plays the role of a young bourgeois trying to climb, by judicious management of his resources, into the ranks of the rich and noble. Although written in English, the program could easily be converted to French, and would be an excellent vehicle for learning about French culture through the medium of the French language.

Although simulations can be very complex programs, they do not have to be. *Cartels & Cutthroats* is actually quite crude, and there is no reason that it could not be more highly developed, even within the very constrained APPLE II environment. Of course, the program was not designed for teaching French, and would be better suited to language teaching if it were provided with a glossary, paraphrase options, or other language teaching functions. This redesign would face a problem endemic to simulation programs, namely, that realistic simulation works against graded introduction of vocabulary and grammar constructs. As a result, keeping control of the language content without trivializing the simulation is difficult.

Context, both oral and visual, are obviously an important aspect of realism, and the authors of language courseware usually try to supply it by incorporating computer graphics into their lessons These, however, are generally not very satisfactory in terms of cost, space requirements, or final quality. High quality animation is even more difficult.

The new videodisc technology appears to offer a neat, relatively cheap solution to all these difficulties. There are language courseware authors who believe that micro/video interfacing will revolutionize language lessons by allowing presentation of culturally authentic material, nuances of body language, etc.

Given adequate time and money, it is certainly possible to assemble plenty of interesting "authentic" video material. How to use it in a way that exploits the interactive nature of the computer is another matter. It is useful to remember that a videodisc is a fixed data base, and that one can retrieve on request only from among a fixed set of visual items. The effectiveness of the lesson is a function not only of the video content, but also of the retrieval strategy employed. The strategies in use so far seem to be rather simple. The student makes choices to branch to different scenes, as in *Montevidisco* (Gale, 1983), or asks a question which the program answers by retrieving a video scene with the relevant information, as with some work done by project Athena.

These designs do not exhaust the possibilities of videodisc; perhaps they are not even the most useful ones. Presently the Language Learning Laboratory is investigating the use of video graphics as a context for efficiently teaching concrete vocabulary. We have done a little pilot work in this direction. In some formats, the computer discusses a visual scene and, while doing so, points to the objects being discussed. In others, the computer orally describes objects and asks the student to point them out. A game format called *Cherchez l'homme* is directed at teaching the vocabulary of clothing and personal appearance. The program displays pictures of several French men, then gives an oral description of one of them in terms of clothing and general physiognomy. The student's task is to point out the man being described. Since most of the details in the description could apply to more than one of the men pictured, a student must listen carefully and understand well to make a correct identification. When a wrong choice occurs, the program gives remedial feedback, in French, by explaining orally why the choice is wrong (e.g., "No, this man has blond hair, not black, and his hands are not in his pockets").

These formats are still rather crude. Our goal is to make them like some classroom exercises: very fast, very interactive, with constant exposure to audio and video, and with as much repetition as is necessary to internalize the vocabulary and learn the grammar. The next step, of course, is to introduce action scenes as a device for teaching verb vocabulary and verbal concepts such as tense and aspect.

We also plan to investigate designs which depend on categorizing and interrelating the library of scenes in various ways. For example, a brown book could be contrasted with a blue one, or one girl with several girls. Words could be taught by generalization from a number of specific examples, and temporal subordinators could be taught using sequences of action scenes. If the student indicates that he can not understand what is being said about a scene, we could use a common classroom technique and retrieve simpler paraphrase messages or video scenes which are simplified or differently focused. (This kind of paraphrasing is not practical, of course, unless we have a way to select the audio message independently of the picture. It is an important technical problem to provide this capability; either speech synthesis or CD-ROM may offer a solution.) If the program also maintains a learning history for the student and uses it to control the difficulty level of presentation, we might end up with instruction which looks somewhat like a computer-based version of the natural method.

The power of this sort of design comes from having a lot of knowledge about the scenes, and a program which can use that knowledge to select video information according to a strategy which will lead to efficient comprehension learning. Determining what kind of knowledge to include and how to represent it is an open problem which will require input from both instructional design theory and linguistics.

EASY-TO-USE AUTHORING FACILITIES

Some expert opinion holds that lesson authoring, like textbook writing, is too complex and time-consuming a task for the average teacher, and should be left to individuals who have special expertise in producing computer-based instruction. In this view, really good materials can result only from a long-term (and very expensive) development effort. This view ignores the fact that microcomputers are ideal devices for doing desktop publishing of both on-line and off-line materials. Teachers want authoring facilities for the same reason they want access to mimeo machines: because they frequently find it desirable to produce ephemeral materials which are hand-crafted to individual syllabi, students, or teaching styles, and which can be redone or modified to reflect changes in current events, student interests, or classroom techniques. Such materials typically forego complex instructional design in favor of simple formats already familiar to both instructor and student. But authoring must be quick, because the materials intrinsically have a short lifetime, and because teachers typically have little free time to devote to lesson authoring.

General purpose programming languages cannot satisfy this need for speed and efficiency because their flexibility makes them complex and difficult to learn and use. Some language-specific authoring systems attempt to avoid these difficulties by adopting a template-based, fill-in-the-blank approach. The author selects one from a small set of exercise formats and types in some language items. The system then takes care of presenting the items according to the selected format. In this case, authoring simplicity is achieved at the cost of flexibility. The formats are more or less fixed, and modifying the program or adding new formats is likely to be difficult or impossible.

Certain courseware authoring systems attempt to maintain flexibility by compromising between general programming languages and template systems. Unfortunately, currently available systems of this type are typically oriented toward programmed learning designs and stress screen layout. Even when they provide for truly easy screen design (which is not always the case), control of sequencing remains primitive. Screens must generally be implemented in small pieces. Getting all the pieces together in a desired sequence may involve several menus and editors--at best an awkward process, at worst, impossible. Using such an authoring tool to create a simple vocabulary drill turns into a sizable project.

In part, these problems result from actual limitations of capabilities, but clumsy user interfacing plays an even greater role. The study of human-machine interaction has shown that an effective interface must provide a *user model* of how the interface works. This means, in essence, that the designer should find an activity the user can already do well, and construct the interface so that it functions, as much as possible, in the same way. This idea may seem like common sense, but a quick survey shows that it is not much applied by the designers of authoring systems.

Teachers who are beginning to write computer-based lessons naturally enough use as a model the writing of non-computer lesson materials such as classroom exercises and textbooks. At this point, it will be tempting for the computer sophisticate to sigh and patiently explain once more the radical differences between programming a lesson and writing a textbook. But human factoring suggests that stressing this distinction may be bad strategy. Instead, we should perhaps do just the opposite, namely, facilitate the identification by making the authoring process resemble ordinary writing as closely as possible.

The Language Learning Laboratory is now exploring this possibility by building a language authoring system designed along these lines. As described by Cheng (1986), the user interface will be a word processor which is function, not procedure, oriented interactive, not reactive, and direct manipulation not command, driven

The word processor will provide simple sequencing control in the form of an outliner facility. Questions, answers, and feedback will be interactively prompted without requiring the user to leave the word-processing environment. Various drill and exercise formats may be specified and their data typed directly into the text, just as would be done with textbook exercises. The intent is that all authoring be done by creating text in a (heavily prompted) word-processor environment. All the specifications for the lesson will appear as text, which is a familiar and non-threatening medium for new authors.

Of course the lesson itself will generally be highly interactive. The completed lesson text is submitted to an interpreter which generates appropriate screen displays, collects and judges responses, keeps scores, status, and learning history information, and controls branching and routing. This system is being implemented on the IBM-PC. Display is done in bit-map graphics to support a full variety of non-Roman writing systems, pictorial material, and a variety of type sizes and fonts.

I do not want to give the impression that a word-processor model solves all the problems of author/program interaction. As in any direct-manipulation approach, branching presents a troublesome issue. For many common instructional operations, variables and conditionals seem to be the most natural way of controlling program flow. Introducing such control mechanisms into the text reintroduces aspects of a procedural programming language and destroys the "what you see is what you get" property of direct manipulation. Devising a representation which will be natural both for screen display and for flow of control information remains a central problem for the proposed authoring language.

CONCLUSION

The research reviewed above suggests the outline of a new generation of language courseware. While not yet reaching the level of truly "intelligent" instruction, it will be substantially more sophisticated than the drill-and-practice materials we have now, and it will invalidate many of the criticisms leveled at current lessons. Micro hardware and software presently on the market or under development will provide delivery systems for these new lessons at something approximating an affordable price. The net result will be a much stronger case for classroom deployment of computer-based foreign language instruction.

THE AUTHOR

Robert S. Hart is Assistant Professor of Humanities and Associate Director of the Language Learning Laboratory at the University of Illinois at Urbana-Champaign. Address correspondence to G70 Foreign Languages Building, 707 S. Mathews St., Urbana, IL 61801.

NOTES

PLATO is a registered trademark of the Control Data Corporation. EnBASIC is a registered trademark of the Computer Teaching Corporation. GOLDEN COMMON LISP is a registered trademark of Golden Hill Computers, Inc. Courseware Authoring System is a registered trademark of McGraw-Hill, Inc. Cartels & Cutthroats is a registered trademark.

[1]EnBASIC, authored by A. Avner, S. Smith, and P. Tenczar, is distributed by Compress, PO Box 142, Wentworth, NH 03282.

[2]DASHER, authored by J. Pusack, is distributed by Conduit, PO Box C, Oakdale, IA 52319.

[3]The BMCAI program for error analysis of Russian verbs is presently under development by Dan Davidson of Bryn Mawr College. For information contact BMCAI, Bryn Mawr College, Department of Russian, Bryn Mawr, PA 19010.

[4]Cartels & Cutthroats is distributed by Le Professeur, PL Box 301, Swanton, VT 05488.

[5]The Would-be Gentleman, authored by Ruth Lougee for the Apple Macintosh, is distributed by Kinko's Academic Courseware Exchange, 4141 State Street, Santa Barbara, CA 93110.

REFERENCES

Cheng, C. C. (1986). An authoring system for computer-assisted instruction in Chinese. In *Proceedings of the 1986 International Conference on Chinese Computing*, 40-46.

Culley, G. (1979). Two pronged error analysis for computer-based instruction in Latin. In *University of Delaware Symposium on language and linguistics*. University of Delaware.

Gale, L. (1983). Montevidisco: an anecdotal history of an interactive videodisc. *CALICO Journal, 1:1*, 42-46.

Kramsch, C., Morgenstern, D., & Murray, J An overview of the MIT Athena language learning project. *CALICO Journal*, June, 1985, 31-34.

Levin, D. (1973) Computer-based analytic grading for German grammar instruction. Stanford University Computer Science Department Technical Report #STAN-CS-73-343, 1973.

Marty, F. (1981). Reflections on the use of computers in foreign language acquisition. *Studies in Language Learning*, 3:1, 26-52.

Morgenstern, D. (1986). The Athena language learning project *Hispania*, 69, 1986, 740-745.

MICROCOMPUTER FEEDBACK TO THE STUDENT/LEARNER

Patricia A. Mulligan and Gary G. Bitter

Little research to date has reported student assessment of the relevance or usefulness of software used in instructional programs. In the present study students were interviewed about the feedback provided by such programs. It was found that students react to stimuli in ways adults do not. Students think along channels that adults have learned are non-productive, and can therefore no longer remember and activate. Therefore, although the educator might make a special effort to put himself/ herself in the student's place to see problems in the way that the student sees them, this kind of intellectual empathy can really be achieved only indirectly through observation and interviews. When the student's point of view is incorporated into the curriculum, then a program that is in harmony with the student's limitations and growth potential can be assimilated Students are capable of identifying characteristics of software and their recommend-ations should be valued as equal to those made by adults. Students should use and evaluate software with an adult observer.

INTRODUCTION

One aspect of microcomputers is their ability to interact with humans. This feature enables the student to respond to questions, and the computer provides "feedback," so that an interaction, a kind of conversation, is established between person and machine. Microcomputer feedback is defined as any message or display that the microcomputer presents after the student inputs information. Steinberg (1984) defines feedback; it may be as simple as yes or no; or as elaborate as an explanation of why the student's answer is incorrect with directions for how to get the correct answer. Feedback may also be animated graphics or a statement of the student's score.

As in any learning process, learning through the micro-computer provides immediate response, and in this case, that feedback reflects or assesses student responses and encourages student performance in a systematic and consistent way.

Ideally, any computer curriculum is developed by teams of users, with multiple perspectives: the classroom managers, the microcomputer operators, and the learners who will implement the curriculum. It is true that microcomputers have been used in classrooms and laboratories in a variety of ways, and that they have been used as programming tools, for word-processing, and as individualized tutors for software for two general types of lesson design: drill and practice exercises and simulation exercises. However, it is unfortunate that curriculum designers, educational specialists, school administrators, classroom teachers, and students have not collaborated in developing effective programs (Komoski, 1984). More disconcerting is the observation that students, the major consumers of educational software, have played only a minor role in the development process. Students have had little input; teachers have contributed what they conjecture is the student perspective because they have informally observed students. More succinctly, little research to date has reported student assessment of the relevance or usefulness of the programs used in instruction.

The purpose of this study is threefold:

1) to formulate propositions and to form a grounded theory base for understanding student/learner computer inter-action,
2) to extract from that theory, the relevant features of educational curriculum,
3) to incorporate these principles into the development of educational curriculum and software.

As part of the study, student interaction with selected microcomputer software was observed, described, analyzed, and interpreted. Students were observed and interviewed in the process of using software programs. Students were asked to discuss the feedback provided by the programs. Students were asked to express their likes and dislikes, to give recommendations for software improvement, and to explore potential uses in their school curricula with the interviewer.

To gather data for a theoretical base, the researchers pursued answers to the following questions:

What characteristics of feedback in software are important to students/learners?

Do media characteristics such as color, sound, animation, and the ability to generate both text and graphics increase appeal to students?

Do learner-related characteristics such as the type of required cognitive processing, the degree of help (mediation), motivating factors, and reduced anxiety producers (affective qualities) change learners' attitudes?

Are user friendly characteristics such as clarity of screen display, positiveness, variety, interactivity, and humor important to learners?

What recommendations can be made to curriculum designers, educational specialists, school administrators, and classroom teachers for putting the findings into practice?

NEED FOR THE STUDY

Educators hear about the amazing potential of computers and how their use can enhance learning by individualizing instruction. Software program descriptions convince educators that computerized learning is an efficient means for transferring knowledge, but students frequently find programmed learning boring; many software programs are merely academic exercises that lack the motivational force of good classroom instruction. Lack of space, lack of memory, and lack of hardware capability surely do account for part of the problem; but, much of the weakness of software resides in the limitations of the basic instructional design (Cohen, 1985).

One reason for the discrepancy between potential and actual use of educational software has been in the way feedback has been utilized. Feedback is a powerful yet little understood feature in instructional design. Decisions on the use of feedback are presently being based on a small amount of applied research that has been done in this area. Little information from learning theory research has been translated into instructional design.

Instructional designers were swayed by the behaviorist research of the 1960's; and the software programs reflect that learning theory which encouraged immediate feedback following a correct stimulus-response bond. Stimulus-response models resulted in sterile drill and practice excercises. Supporters of the theory applied it to classroom learners by designing learning programs such as programmed instruction and teaching machines. These programs reinforced the student for every correct response, using immediate feedback to shape and maintain behavior as illustrated in Figure 1.

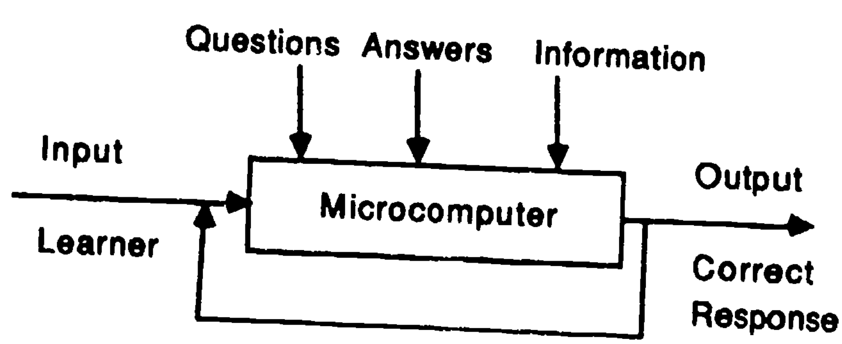


Figure 1 Feedback to learner

This reasoning was a natural outgrowth of behaviorism; however, there has been little classroom evidence that feedback following positive responses acts in a reinforcing manner(Cohen,1985; Bardwell,1981; Barringer and Gholson, 1979; Hartley and Lowell,1978; Anderson, Kulhavy, and Andre, 1972).

In contrast to the strong behaviorist tradition a second school of thought developed forwarding the idea that the main function of feedback is neither to strengthen nor to reinforce correct responses,but to locate errors and provide information so that the learner can correct them (Cohen, 1985; Bardwell, 1981; Barringer and Gholson, 1979; Anderson, Kulhavy, and Andre, 1972).

Whereas the behaviorist model resulted in a right or wrong checking system, informational feedback acts primarily to correct errors, and the number of errors subsequently corrected is directly related to the amount of information offered. Providing only positive feedback during instruction is less effective than giving negative feedback. In addition, by merely indicating "correct" or "wrong," the message is not providing the student with enough informational content to be instructionally effective. The "correct" message alone has been shown to have no significant influence on learning. Thus, there are two separate effects feedback will have on each response students make: 1) to let the students know when they are right or wrong, which can be accomplished with a simple "right" or "wrong" feedback message; and 2) to correct students when they are wrong by providing informational content to help facilitate understanding and comprehension of the material (Cohen, 1985).

The value of both functions in instructional design is obvious. The first function, described as knowledge of results, engages students in corrective activity when they have made an error. It allows them to judge the accuracy of their responses in comprehending and mastering the material. The second function, informational feedback, identifies errors and allows a learner to correct them by providing enough information to help the students locate the error in the response and construct an alternative response. Informational feedback has been acknowledged in research to be one of the most important aspects of feedback (Cohen,1985).

Although researchers have identified what feedback is, they have not developed guidelines on how to use feedback in instructional design and/or curriculum development. There are few practical suggestions in the literature on how to use feedback as an instructional tool. In addition, the directions that are part of the software itself on the use of feedback strategies are often simplistic and thus make feedback ineffective (Cohen, 1985).

If feedback is an important constituent of any learning process, as is evident from the previous discussion, feedback must play a crucial role in the implementation of instructional design using software programs. Incorrect or inadequate feedback can subvert a well-planned curriculum. In order to study learner feedback appropriately, the investigators selected a grounded theory approach.

METHODS

Grounded theory, which provides a means for investi-gating previously unresearched areas, was the method used in this study. Grounded theory, a form of field methodology, aims to generate theory constructs which explain the action in the social context under study. That is to say, it con-sists of a series of propositions linked together in such a way so as to explain the phenomenon. It is a combination of inductive and deductive approaches; the investigator focuses the research according to a conscious selective process. The way the researcher collects and codesthe data, integrates the categories, generates memos, and constructs theory are all part of the processes of generating theory and research. These are guided and integrated by the emerging theory. The outcomes of the research, therefore, evolve continually; this evolving outcome leads to Glasser and Strauss's view of theory development as a "process" and an "ever-developing entity," not as a perfected product (Glasser and Strauss, 1967).

Procedures

Triangulation, the use of multiple data structures, permits values and perceptions to emerge and forms a basis for the interpretation of results and the confirmation of theory.

Continuous comparative analysis, the process of checking by alternating between phases of inductive and deductive inferencing, involves constant comparison of data with other data, creating propositions about patterns that emerge from the data, testing and evaluating those propositions against further observations, and generating a synthesis from that process. This kind of analysis can be seen from Figure 2.

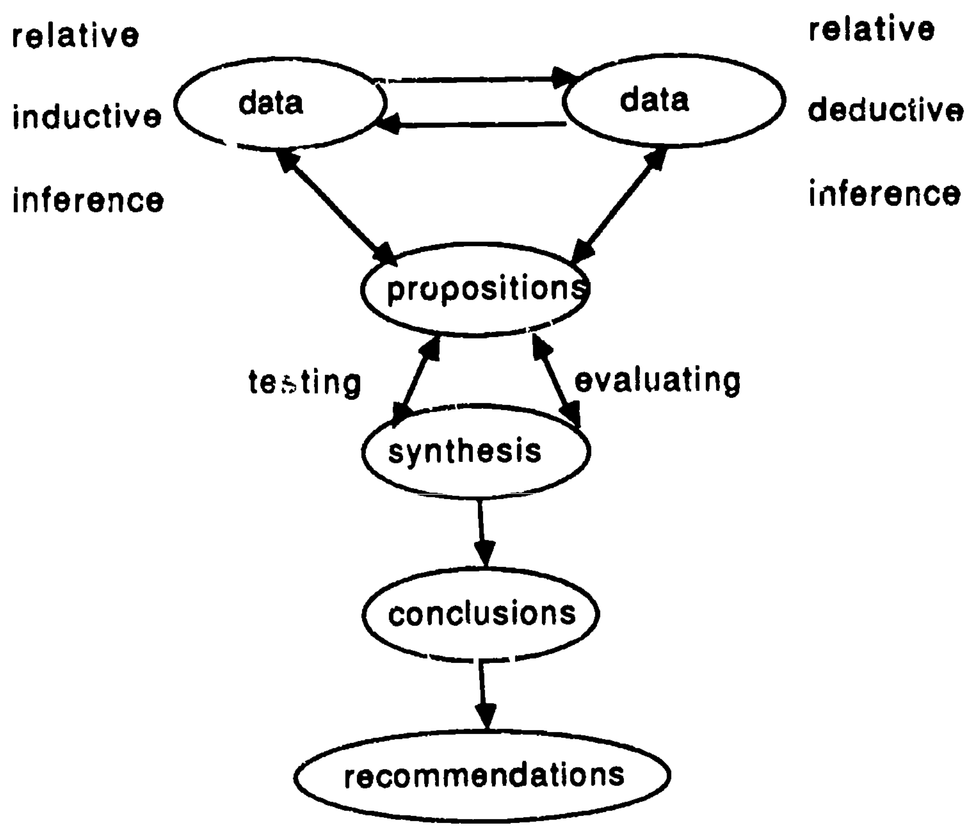


Figure 2 Continuous comparative analysis

Because the process is ongoing, the continuous comparative analysis must be pursued consciously throughout the study. This method is well suited for this particular study because it examines concerns and issues as they emerge from the investigation. At the conclusion of the study the researcher is then able to make recommendations. The following sections describe the language arts studies.

Data Collection

Data collection involved observing, taping interviews, and distributing questionnaires at a northwest Phoenix elementary school which has a Commodore 64 microcomputer laboratory. Two teachers volunteered to participate in this study, one teaches sixth grade and the other runs the microcomputer laboratory. It was decided after consultation with them that there would be two data collections and that the first data collection would begin with the fifth and sixth grade after-school computer program. The teachers have been teaching the students Logo and some word processing with <u>Bank Street Writer</u>. The researcher, who participated in the regular classroom activities for a two-month period, observed the students daily while they were working on their

Logo projects and Create With Garfield, the language arts software selected for this study.

The Create with Garfield software used in this study was provided to the school by the software company, Developmental Learning Materials. The program features are described by the publisher in the following way:

> Create with Garfield offers you
>
> o your favorite Garfield characters, plus colorful backgrounds and props for creating your own cartoons
>
> o easy-to-use word processing capabilities for writing your own captions--or use Garfield's own comic quotes
>
> o the ability to use your own backgrounds and props for an unlimited variety of cartoon settings
>
> o an electronic comic feature, which displays your cartoons continuously frame after frame
>
> o the opportunity to print your creations as cartoons, posters, or labels and use them to personalize letterheads, books, name tags, announcements--anything you like!
>
> The objectives are
>
> o to stimulate writing and artistic skills
>
> o to encourage humor for creative thinking
>
> o to give practice writing dialogue
>
> o to help develop a sense of design, balance, sequencing, and spatial relationships.

Both teachers introduced the program to their students. They let the students explore its capabilities for a week. They then brought in cartoons from the newspaper and discussed putting cartoon strips together. The teachers asked each student to develop a plan on paper and show the plan to them before continuing with the project.

The students then worked on their own cartoon strips for the next two weeks. The teachers allowed them to interact and comment on one another's work.

At the end of the unit the researcher tape recorded interviews with the students, asking them to give their impressions of the software.

The second data set collected was with the sixth-grade teacher's regular class. They used Create with Garfield the last three weeks of school. At the end of the three weeks they were asked to evaluate the program by filling in a written questionnaire.

DATA ANALYSIS

The following coding scheme categories emerged as the data were analyzed:

Cognitive aspects (appealing to the intellectual side, learning, making, doing, and previous computer experience)

Instructional mode (teaches content and students can show its use to others)

Interaction (can do things on the computer that cannot be done with paper and pencil)

Degree of help (student needed help from the teacher or researcher)

Machine capabilities (using the Commodore 64, judgments of slow/fast, or whether it works/does not work)

Graphics (pictures, in Create With Garfield are described as background, stick-ons, and props)

Sound (music and beeps)

Color (authentic or phony)

Variety (more than one activity)

Clarity (clear directions)

Positiveness (pleasant computer responses, fun)

Humor (funny, enjoyable)

Analysis of the classroom observations, oral interviews, and responses to questionnaries becomes difficult as it is necessary to separate any one category from the others because they overlap into one another. Therefore, the categories will not be described separately in sections, but will be integrated into one succinct report.

Students identified aspects of using a computer (aspects of computer literacy, computer experience, or computer skills) which they felt they were taught and learned by using Create With Garfield:

- how to make a cartoon using backgrounds, props, and captions

- o how to use windows
- o how to use menus
- o how to initialize and use a data disk
- o how to use new vocabulary
- o how to read and follow directions
- o how to save and retrieve (load) a cartoon
- o how to edit
- o how to make people laugh

The students liked the graphics which were described as backgrounds, props, and stick-ons. There is a choice of seven different backgrounds: plain white, room, table, line, tree, fence, and label outline. The stick-ons are Garfield, Odie, Jon, Friends, Props, Quotes, and Write Captions. When friends are selected there is a choice of Arlene, Nermal, and Pooky The Props contain pictures such as lasagna, refrigerator, TV, etc., and cartoon words such as "CRASH", "BARK", "WHAM". Examples of quotes that can be used are: "NAP ATTACK!," "IT'S HARD TO BE HUMBLE WHEN YOU'RE AS GREAT AS I AM", "WHY ME?" "Write Captions" allows you to write your own statements in stick-on caption bubbles.

The students liked the variety. Many of the students would like to use the Koala pad to generate their own backgrounds and props. One student said that she would "like to change the characters and props so that they could be different sizes." In addition, the students liked creating cartoons that they couldnot draw by themselves with paper and pencil.

There was a great deal more computer experience required of the students in using computer software than was indicated by the identified objectives. Sample lesson plans adding the following objectives to the previously mentioned ones might have helped the teachers plan more effectively:

- o To learn the "new" computer vocabulary, a computer terms dictionary would be helpful to teachers and students alike.
- o To learn how to use menus and "toggle" back and forth between them; overhead transparencies allowing the teachers to present the concepts would have been helpful.
- o To understand the purpose of a data disk and how to intialize one, the teachers needed to have a blank disk for each student.

Both of the teachers participating in this study have had computer education courses and have taught computer applications for six years. Because of their experience and knowledge of a variety of software programs, they were able to teach the students as problems arose.

> Brian: "Whenever I started the little white screen, I didn't know what to do with that...And then I had to ask Mr. Charles what to do and he told me how to get a background and all the stickers to put on."

Student criticisms may have been expressed as "the directions were not clear," but usually it was the case that they had simply not read the directions.

> April: "I thought it was hard because when I was making a cartoon, I would forget to press something that makes them stay on the screen."
>
> Interviewer: "So you lost them?"
>
> April: "Yeah. I didn't read the directions."

The "Create Menu" was the hardest menu for them to master because from it they would go to other menus and sometimes find themselves in the wrong place and would forget which keys to press to get back to this menu.

CREATE MENU

SEE PICTURE	SPECIAL ARTWORK
GET BACKGROUND	DATA DISK
GET STICK-ONS	CARTOON PRINTER
ERASE STICK-ONS	SEE INSTRUCTIONS

Editing or erasing stick-ons presented problems. The option is available so that one can eliminate characters, props, and captions after they have been placed on the background. The procedure is to select an outline large enough to enclose the entire stick-on. The problem that arises is that any portion of any stick-on that extends into the outline will be erased too. So if two pictures overlap and you erase one, part of the other will be erased as well. The students found it easier to start all over than to use this editing feature.

The students ran into difficulties learning the operations with this menu and requested teacher assistance to walk them through the steps until they learned them. For example, they would see a data disk menu that looks like this:

DATA DISK

SEE INSTRUCTIONS

SEE THE CURRENT CARTOON

LOAD A CARTOON

SAVE THE CURRENT CARTOON

DELETE A CARTOON

INITIALIZE A DATA DISK

CREATE M[illegible]

Students confused not only the words "intializing, loading, and saving," but also the operations that went along with them. If the teachers had better understood the level of computer expertise that would be required of the students the teachers would surely have spent more time talking about and teaching the students about these operations in their lessons. [illegible]ch information might have been included in the documentation.)

The color graphics capabilities of the Commodore 64 pleased both teachers and students. Rhoda said, "The colors are nice and look real." She explained when discussing the background with the fence, "It looks like a real fence."

The only problem brought up by the students was color bleeding. Two students were disturbed when they put two graphics side-by-side or overlapped them and the colors bled.

> Chris explained, "You set it next to something, it will mess it up. Like if you have a little dish of dogfood and put a bone on top of it, the bone will cover up part of the bowl with the color."

Overall, the students liked the use of color and hoped some day to print their cartoons in color using a color printer.

The music that introduces Garfield is best described by Mike, one of the teachers.

> There were kids who were barely able to sit in their chairs. The sound was so exciting. We told them where the volume knobs were so that they could adjust them [so that the sound wasn't blaring]. They wanted to be sure they heard the music. I remember the first couple of times we were trying to talk [give directions] to them and load at

> the same time. And the minute you hit that music, you just saw, "All right! This thing sings!" It was great!

The extended-day students didnot comment on the speed of the Commodore 64 disk drive, but three students from the regular sixth grade class did.

> Melanie: "I didn't like how it took so long for some things to load. Sometimes I didn't even finish my picture because it took so long to load."

The microcomputer laboratory printer could not print out the students' cartoons. A printer that was compatible with the software was borrowed, but the printing took a long time. The teachers found that they could not use the software for the writing workshop they had planned because the editing process was so slow. Instead, they ended the cartoon unit after each student had completed one cartoon strip.

When the students were asked what their impressions were of Create with Garfield, their responses were quite positive:

> Manuel and Nathan said that,"the character Garfield is "funny and neat" and that "it is fun to make cartoons."

> Frankie expressed that "the other kids in his regular sixth-grade class should have the opportunity to use the program."

> Missy thought that "it is better than any other program I've ever used."

> April said, "It is just like a Garfield cartoon, and it is fun making a cartoon on the computer."

Data Interpretation

In their evaluations, the students reported aspects of both the hardware and software that relate to the feedback characteristics of the software and to the purpose of this study. The results from the data analysis have been compared to identify as many characteristics of feedback as possible, and they have been integrated into the discussion as highlighted propositions.

Students know the cartoon character Garfield and enjoy interacting with him. The students' familiarity with Garfield's behavior (for example, they all know that he eats great quantities, consumes all with great relish, and relates with other characters in a rather sarcastic mode) eliminates the need for the students to create patterns of behavior for the character, thus following current rhetorical teaching recommendations. In other words, students are not required to supply, from their own creative banks, more than they are willing to borrow from already established sets. Because the characters are established, the students do not need to create more than the action, the dialogue, and the excitement. They can concentrate on the most exciting part of the creating process: this already complete picture needs only the elements of language and expression that the students want to give the characters, thus eliminating all the anxiety-producing elements of invention.

The work with Garfield amused the children at the same time that it motivated them. The humor involved in the process of composition blurred the fact that the students were learning as they were composing: that is, prewriting, visualizing, reconsidering, planning, revising, checking, and testing. Therefore, they were involved in the project at two levels. Consciously they were occupied with surface issues, while subconscious by they were learning to manipulate language, reinforce computer operating skills, communicating, and composing. In this way the students were freed to be creative, encouraged to explore their artistic abilities and to exploit the potential of the program, and encouraged to find new limits.

The experience of using the Create with Garfield program demanded the use of more computer know-how than most of the students possessed. Thus, the lessons indirectly involved acquisition of skills in the use of both the hardware and the software. Furthermore, the students learned valuable academic skills such as setting objectives, working within a timeframe, refining results, and following directions (If they did not read the directions, they did not succeed.) In addition, the students' reactions and evaluations pointed out the need for several kinds of software improvement.

The language propositions evolved into the grounded theory that follows.

GROUNDED THEORY

Students in elementary and secondary schools react to stimuli in ways adults no longer do. Because they think along channels that adults have learned are non-productive, have weeded out, and can therefore no longer remember and activate, students make errors that adults (teacher, curriculum specialists, administrators, and software specialists) can no longer predict naturally. Therefore, although the educator might make a special effort to put himself/herself in the student's place so as to see phenomena and problems in the way that the student sees them, this kind of intellectual empathy can really be achieved only indirectly through observation, interviews, and questionnaires. When the student's mindset is incorporated into the curriculum, then a program that is in harmony with the student's limitations and growth potential can be developed.

Students are capable of identifying characteristics of feedback which will aid in software adoption review. For this reason, students should be considered for membership into their school district's software review boards. However, taking into account the fact that students lack mature adult ability to objectify, the students should be required to use and evaluate the software with an adult observer. In other words, their recommendations should be evaluated along with those made by the adults.

RECOMMENDATIONS

The following recommendations are made to curriculum designers, educational specialists, school administrators, and classroom teachers:

- For purposes of lesson clarity and orientation, specific objectives for each lesson should be incorporated into each lesson introduction. Research results emphasize such objectives to orient students and reinforce stimuli. (Cohen, 1985; Hartley and Davies,1976; Duchastil and Merrill,1973).

- Function keys which allow the student to ask routine vocabulary and concept review questions should be included in software programs.

- Special care needs to be taken in software selection to include programs that will be meaningful to the children. It is recommended that students be included in the planning phases so that what they like and choose for themselves is included in the curriculum. There are many things we as adults donnot think of as fun any more that children really enjoy and need to have as part of their growing up experience.

- Sound, animation, and response mechanisms stimulate and sustain attention. They should be incorporated into lessons to make them more compelling.

- Language arts writing software, which must be adaptable to the workshop approach, needs to include an easy editing feature. If text editing can be done easily, then the writing and revision process is the students' focus and not the hardware or software functions.

- This study employed the use of two different software programs. It is recommended that other software programs in different content areas with different grade levels of students be studied.

- It is also recommended that a national survey and interview be conducted of schools which include student feedback in their program and curriculum design process.

ACKNOWLEDGMENTS

A special thanks to Bernie Valdez, Mike Charles, Steve Van Herpen and the fifth and sixth grade students at Pendergast Elementary School who enthusiastically participated in this study. We also owe a debt of gratitude to Donna Craighead of Developmental Learning Materials who sent us the software for our study.

AUTHORS

Patricia A. Mulligan is an interdisciplinary doctoral student at Arizona State University. Her areas of expertise include bilingual education, computer-based education, and second language pedagogy.

Gary G. Bitter, a professor in Computer-based Education at Arizona State University. He is editor of the School Science and Mathematics Journal and author of many books and articles on the use of microcomputers in education.

REFERENCES

Anderson, R. C., Kulhavy, Raymond, Thomas, Andre.(1972). Conditions underwhich feedback facilitates learning from programmed lessons. Journal of Educational Psychology, 63, 186-188.

Bardwell, R. (1981). Feedback: How does it function? The Journal of Experimental Education, 50, 23-26.

Barringer, C. and Gholson,B. (1979). Effects of type and combination of feedback upon conceptual learning by children: Implications for research in academic learning. Review of Educational Research. 49, 459-478.

Cohen, Vicki Blum. (1985). A reexamination of feedback in computer-based instruction: Implications for instructional design. Educational Technology. 25, 33-37.

Duchastel, P. C. and Merrill,P. F. (1973). The effects of behavioral objectives on learning, a review of empirical studies. Review of Educational Research. 43, 53-69.

Glasser, B.G. and Strauss, A.L. (1978). Theoretical Sampling San Francisco: Sociology Press.

Golton, R. (1976). The effects of student interaction on computer-assisted instruction in mathematics at the sixth grade level. Unpublished doctoral dissertation. University of California, Berkeley.

Hartley, J. R. and I. Davies. (1976). Preinstuctructional strategies: The role of pretests, behavioral objectives, overviews and advance organizers. Review of Educational Research. 46, 239-265.

Hartley, J. R. and K. Lowell. (1978). The psychological principles underlying the design of computer-based instructional systems. In James Hartley and Ivor Davies (Eds.), Contributions to Educational Technology, 2, London: Kogan Page Limited.

Komoski, P. Kenneth. (1984, April). Educational software and other learning materials: A look at the differences. Paper presented at the Annual Meeting of the American Educational Research Association, New Orleans, Louisiana.

Kulhavy, Raymond, White, Mary, Topp, Bruce W., Chan L. Ann, and James Adams. (1985). Feedback complexity and corrective efficiency. Contemporary Educational Psychology. 10, 285-291.

Kulhavy, Raymond W., Yekovich, Frank R., Dyer, Frank W. (1979). Feedback and content review in programmed instruction. Contemporary Educational Psychology. 4, 91-98.

Kulhavy, Raymond W. Yekovich, Frank R., Dyer, Frank W. (1976). Feedback and response confidence. Journal of Educational Psychology. 68, pp.522-528.

Sutter, E. and J. Reid. (1969). Learner variables and interpersonal conditions in computer-assisted instruction. Journal of Educational Psychology. 60. 153-157.

AN EXAMPLE OF THE USE OF MICROCOMPUTERS IN FOREIGN LANGUAGE LEARNING AND TEACHING FROM A HIGH SCHOOL FOR THE ACADEMICALLY TALENTED

Constance Curtin and Stanley Shinall

This paper is based upon observations of secondary school students using foreign language programs on the microcomputer. The authors commend computer-assisted language instruction as an excellent tool in the promotion of independent, self-directed study and review which can result in better or more efficient learning. Learner characteristics seen as contributing to achievement are discussed in relation to previous research studies done at the University of Illinois. The best software seems to be that which capitalizes on positive learner characteristics and attributes, motivational factors. In addition to programs which permit self-direction, appropriate feedback, and encourage discovery about the language, those which expand student awareness of the target culture are expecially commended.

The purpose of this paper is to provide observations gained from curricular applications of the microcomputer in the foreign language program of the University High School at the University of Illinois, Urbana, Illinois. The school serves a population of academically talented students from Champaign and Urbana, Illinois, and the surrounding area with occasional enrollments of visiting foreign nationals. As an autonomous unit of the University under the directorship of Dr. Russell Ames, University High School serves as a laboratory site for research and development activities related to curriculum appropriate to students in middle and high school programs. In recent years, a major interest at the school has been the development of instructional materials for use on the microcomputer by students in the foreign language curriculum. Through grants by the Apple Education Foundation and the NEH (C. Curtin, principal investigator) review packets in French, German, Russian, and Spanish have been prepared in Apple Pilot for use on the Apple II family of microcomputers.[1] More recently, through an award by the IBM Corporation in conjunction with a local EXCEL grant (S. Shinall, principal investigator), a microcomputer laboratory has been established. This laboratory enables whole classes to employ IBM PC's at the same time and permits the investigation of a variety of formats. The French and Russian programs have now been converted to IBM Pilot for use in that facility. The development of the programs and their use by students in various instructional settings have led to the observations contained in this paper.

There is no doubt that the pool of academically talented students observed tends to contain some strong language performers. In our discussion of computer-assisted language instruction (CALI), we do not mean to imply that it is the preferred mode of instruction. Neither do we wish to claim that strong student performance in this population is due to CALI. However, our observations lead us to believe that CALI is an excellent addition to the battery of available educational resources and thus should be exploited for what it brings.

In considering student achievement, two studies done at the University of Illinois, Urbana-Champaign, led to further investigation. One done by Avner, Curtin and

Smith (1983)) directly concerns the student population. All students entering University High School take the Pimsleur Language Aptitude Battery (1966) which measures of aptitude based upon phonetic coding ability, verbal and auditory sensitivity and also provides a measure of individual motivation. In each problem of the sound-symbol section of the Pimsleur Aptitude Test, students choose the word which they hear from a list of four written nonsense words. A comparison of the University High School students' performance with national norms for college-bound students (Pimsleur 1966) showed that the local population had a mean grade point average of 14.73 against the national grade point average of 11. In addition, the Pimsler sound-symbol score averaged 21.05 in the local group against the national college bound average score of 16.9. Our high average score suggests that students who have the ability to score higher (as measured by grades in our program) appear to have the ability to integrate visual and aural cues more efficiently than can the average or below-average performer.

A second study by Curtin, Avner and Provenzano (1981) based upon student responses in a main-frame (PLATO) program prepared for University of Illinois students of Russian. Student keyboard responses were collected and analyzed for several semesters to determine among other things, review time, error rate and interaction rate. It was found that students with both low error rates andhigh interaction rates had high grades and spent the most time reviewing at the computer. We are led to conjecture that the more successful language learners (where success is measured by grades) will also be those most capable of independent, self-directed review and that for them such review results in better or more efficient learning. If this is so, CALI programs developed for students are best when they capitalize upon the characteristics which seem to contribute to foreign/second language learning. Moreover, CALI authors should attempt to appeal to positive attributes which may be inspired, developed and reinforced in the learners, indeed enjoyed by them. Let us consider some of the things we have learned from observing students using CALI at the University High School.

First, they learn faster than others and so presumably can attain proficiency more rapidly. All students feel happier when they are engaged in the learning task. They do not particularly enjoy feeling pushed, but they do enjoy active learning. Therefore the pace or rate of instruction needs to suit the student. One way to fit the pace to the student in CALI is to provide a complete index (Figure 1).

```
                 Verb Tense and Aspect

        Type the number of your choice and press ENTER:

 1. Introduction to the Lesson    8. Which Aspect?  (in English)
 2. Vocabulary Practice           9. Which Aspect?  (in Russian)
 3. Past Tense                   10. Vocabulary Quiz on Verbs
 4. Future Tense                 11. Soviet Republics Game
 5. Aspect Discussion            12. Final Test
 6. Forms of Aspectual Pairs     13. Disk Index
 7. Aspect Contrast in English
```

Figure 1. An example of a complete index allowing for choice in a range of areas progressing from the more simple to the more complex.

As we developed our programs, which are primarily independent review materials, we learned to structure the index using a laddered approach, that is, one which proceeds from more simple concepts and problems to more complex ones. Because most students enjoy the motivational aspects of a game, we placed our games at the end of the instructional sequences, so that to succeed in the game, the student would need to have learned the content of the lesson. We learned that students would frequently attempt the game first and would then be impelled to enter other parts of the program if only to be able to return to the game with the assurance of a better performance. At the end of each lesson index, we placed a final test, which could also serve as a pre-test. If students are able to score well on this test, they may elect to move to another review and proceed more rapidly or they may elect another activity. By giving students the freedom to choose, we have attempted to meet their need for independent decision-making, to allow them to proceed at their own rate, and to feel responsible for their own progress.

A second observation is that our successful foreign language students are sensitive to the process of learning taking place within themselves and are quick to ask questions about what they are learning. Changes take place both in the understanding and control of the language. Achievement paves the way for new learning, and the successful learner seems eager to apply what is learned and becomes impatient when application is not possible. At the same time, talented students are often quite sensitive to their perceptions of "lack of success." Just as teachers in oral communicative situations in the classroom must encourage students without demolishing self-esteem, CALI must provide appropriate feedback and help to build self-confidence through facilitating an understanding of concept and application. Talented students need both inspiration and challenge so that they are continuously drawn ahead in the learning process. An example of CALI designed to do this is found in the Russian Review Packet lesson treating the Accusative and Locative Cases.1 In one lesson segment, after they have had the opportunity to gain control of the case forms in isolation, students are required to apply their understanding not only of form but of usage in answering a question. The preposition and case required in the response show either that the verb in the question indicates a static location (locative) or motion toward a location (accusative). The cue noun is in the nominative case, and helps are available. The inspiration to apply what has been learned comes in the form of the game-like format which consists of a map of Moscow. Successful responses cause a blinker to show the location of the site named in the question. This approach--combining task, appropriate helps and feedback with tangible reward (new knowledge)--has been one of those most appreciated by talented students (Figure 2).

The foregoing observation leads directly to the third: talented language students enjoy discovery. Successful language learners we have observed use their ability to analyse and to apply logic to problem solving. Such students are adept at following logical developments of concepts to grasp and to retain structural concepts which may hinge upon contrasts of English and the language studied. CALI can be an important tool for such students as they see displayed, and are given opportunity to interact with, examples which show how language conveys meaning. Case, verb, tense, and form are made immediately accessible. Teachers of more commonly taught languages can well appreciate such applications (subject-verb agreement, pronoun form, idiomatic tense uses, contrasts between preterite and imperfect, for example). Teachers of highly inflected languages can see even more subtle applications. In Russian, for example, the teaching of aspect poses particular difficulties to the learner. Here again the computer provides a magnificent tool for teaching logically and incrementally. In the Russian Review Packets, students are helped through a series of steps (first in English, then in Russian) designed to help them "discover" aspect (Figure 1).

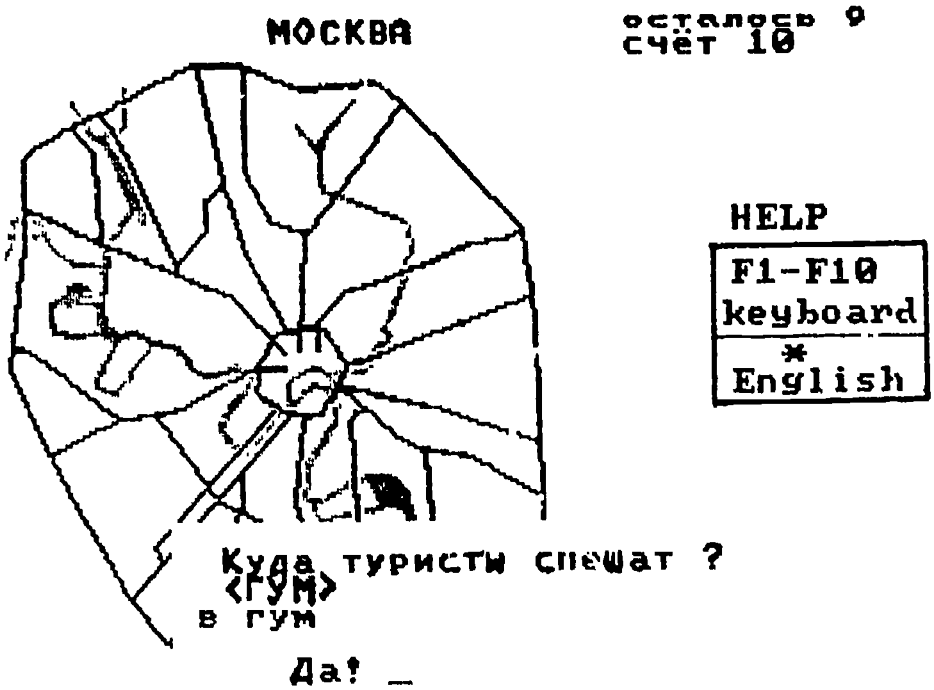


Figure 2. An example of a game requiring a conceptual understanding of case usage. Places of interest in Moscow are shown by a blinker as a reward for supplying the correct answer.

Discovery in the foreign/second language curriculum extends beyond a growing awareness of language for personal communication. It involves an appreciation of different societies and cultures directly related to the target language and, beyond them, of the world. CALI offers opportunites to use target language forms to instruct in daily life, traditions, and customs as well as in the politics, history and economy of other nations or language groups. The students we have observed especially enjoy encountering such material in game format in which they may learn things related to geography (rivers of the Soviet Union), tourist interest and travel (Moscow, Paris, Vienna), or combinations of these features (Figure 3). In such lessons, instead of learning isolated facts the students are applying their language skills in a context which allows them to reach a broader understanding of the world in which they live.

As an extension, at present the authors are developing programs designed to enable students to work their way through sequences of authentic cultural settings. Students select a response from among several alternatives containing idiomatic language appropriate to a particular setting. Sequences of scenes develop a situation likely to be encountered in the target culture.

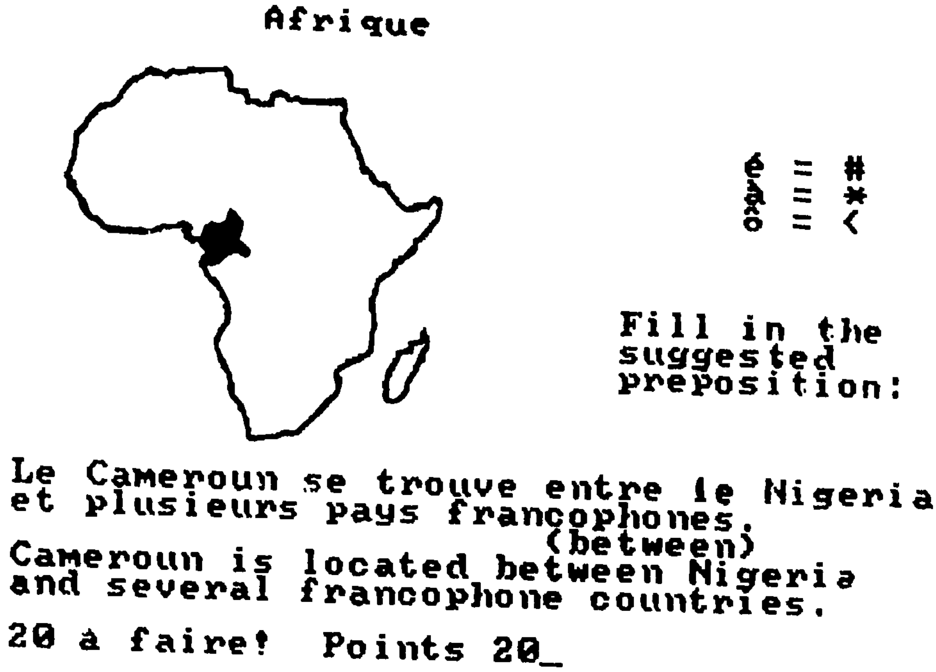


Figure 3. An example of "discovery" about franco- phone Africa. The typing of the correct expression is rewarded with a fiil of the country's location and a translation of the sentence.

In conclusion, what we have learned from observing talented language students is informative for the preparation of CALI for language students in general, including future teachers. Software which allows for self-direction, which provides appropriate help and feedback, and which uses what has been learned as a building tool to increased knowledge can enlist language to expand student understanding of the target culture. It follows that CALI materials, prepared with such syntheses of language and culture as its aim, can provide an efficient and economical tool capable of making an instructional impact both on the learner and on the curriculum.

NOTE

[1]The Review Packets are published by COMPress, a Division of Wadsworth, Box 102, Wentworth, NH 03282.

REFERENCES

Avner, A., Curtin C., and Smith, L. (1983). The Pimsleur Battery as a predictor of student performance, MLJ, 67, No. 1, 33-40.

Curtin, C., Avner, A. and Provenzano, N. (1981). Computer-based analysis of individual learning characteristics. In Robert S. Hart (Ed.), The PLATO System and Language Study. Studies in Language Learning, 3 (pp. 201-213). University of Illinois, Urbana: Language Learning Laboratory, University of Illinois at Urbana-Champaign.

TEMPLATE CONSTRUCTION AS A BASIS FOR ERROR-ANALYSIS PACKAGES IN LANGUAGE LEARNING PROGRAMS

Stephen C. Helmreich

In this paper, I describe a set of programs which work interactively with the teacher/instructor to create a template from which an indefinite number of stimulus sentences can be generated. The instructor also specifies exactly how the stimulus sentence is to be transformed into the target sentence. Using this set of transformation steps, the student's response is judged and feedback is supplied to the student. Feedback is also given to the stimulus-generating system, allowing it to tailor the construction of further stimulus sentences to the difficulties encountered by the student in previous attempts.

INTRODUCTION

Computerization of language learning lessons is still in its early stages. In particular, the analysis of student responses to drill items is generally simplistic and the feedback often counter-productive. In this paper I describe a computer program that I believe represents a substantial advance over syst ms currently available. It permits pedagogically oriented, drill specific feedback and alters the presentation of stimulus material in response to errors encountered in analysis of previous student responses.

Error analysis is the main focus of the approach I am proposing here. It is in this area that the program offers the most substantial improvement over other approaches. However, the program itself encompasses more than just an error analysis package. It offers, in fact, a complete computerization of a certain type of language drill, namely, transformation exercises, or pattern drills. Once it has been provided by the instructor with a pattern for the drill and a procedure for deriving the correct response from the stimulus, the program will construct appropriate stimulus sentences, correctly determine the target sentences, analyze the student's response, and provide feedback to both student and teacher. It may also handle a small group of highly constrained translation exercises. However, by itself, it cannot provide a complete set of computerized language-learning lessons.

In the following section I provide background information, pointing out some of the difficulties of intelligent error analysis, describing alternative approaches, and placing template construction within the larger framework of language generating systems. Subsequent sections provide a detailed description of the system and a conclusion, which summarizes both the advantages and some disadvantages of this approach.

BACKGROUND

Problems of Intelligent Error Analysis

We as teachers are continually involved in error detection, error correction, and error analysis. These tasks are performed routinely and are well within the capabilities of most teachers. One might think, therefore, that computerization of this process would be fairly simple. However, what we are doing is something extremely complex and subtle. In analyzing errors, we are trying to ascertain the invisible, internal cognitive processes which produced the error. The identification of these processes is vital to the task of providing appropriate feedback to the student to aid in correction of the error. Moreover, these processes are not even open to conscious introspection, since it is usually the student's conscious intention to produce the correct answer. However, the main difficulty in intelligent error analysis is that the interpretation of the objective error is ambiguous.

First of all, it is ambiguous linguistically. The same physical error may represent a cognitive error at several linguistic levels. It may be an error at a fairly low level (typing, spelling), at an intermediate level (inflection, agreement, word order), or at a high level (word choice, idiomaticity, appropriateness). For instance, finding the word "chevaux" instead of "cheveux" in a French sentence may be a spelling error, a morphology error, or possibly a word choice error.

Second, there are many possible causes for any particular error. For instance, one and the same error may represent a simple oversight, a temporary lapse, a gap in the student's knowledge, interference from the student's native tongue, a misunderstanding of the instructions, or a major misconception about the target language. Our choice often depends on many other things we know about the student: general level of knowledge, past performance, etc.

Another factor which makes the computerization of error analysis difficult is that it is task dependent. That is, an error in one situation may not be an error in another. For instance, vocabulary tests are generally geared to a particular set of items. Using another word with the same meaning might get the student marks for ingenuity, but would still be a wrong answer. Even if errors "count" in different situations, they may have different weights. For instance, a spelling error would count heavily on a spelling test, and probably for little in an extended essay.

Thus, what makes intelligent error analysis difficult to computerize is the fact that physical errors in the text are ambiguous. Their interpretation and evaluation depends on an analysis of the linguistic import of the error and of its probable cause. Even if this can be done, the weight given to errors varies from exercise to exercise.

Other Approaches to Computerized Error Analysis

There have been two major approaches to error analysis in computer-assisted learning programs. The first is simple pattern matching. With this approach, the student's input is treated as a flat (unstructured) string of characters. The input is searched for certain words or phrases. This method requires a pattern (a correct answer) against which the student's input can be matched. As such it is a fairly unintelligent approach to error analysis. However, in its favor, it can serve as a general purpose approach, workable for almost any input in any language.

Parsing the input is a more intelligent approach to error analysis. Because of work done both in computer science and linguistics on the nature of grammars and parsers for natural language, some of the principles of parsing and some of the basic structures of natural languages are fairly well understood. The parsing approach, therefore, tries to analyze the student's input in structural terms, not simply as a flat string of characters. The structures it looks for are those of the foreign language, so it must "know" something about these structures. In this sense, it is more intelligent. In addition, it is not bound to situations where a clear-cut pattern is available against which to judge the student's input. With an adequate grammar of the target language, a parser can handle almost any input in the language. In this respect, it appears to mimic human analysis of language. Since we do not always know what it is we will be hearing or reading, we must analyze the input we receive and assign it a structure and meaning.

It would be tempting to think that this work could be taken over directly into error-analysis programs. However, almost all such work deals only with analyzing correct input. Bad or incorrect input is generally discarded or ignored. Incorrect input, unfortunately, is exactly what must be examined and analyzed in error analysis. In addition, a parser that can analyze most natural language input and operate at a reasonable speed will be quite complex and unintuitive. This is because the tasks of computationally optimizing a parser and of providing a linguistically sophisticated grammar are both highly complex and technical. This means that it is difficult to make such programs interactive and task dependent. It also is not an easy job to take the complex analysis provided by the parser and turn it into helpful and non-technical feedback to the student and teacher. "Violation of Binding Principle B" is not always helpful feedback to a student trying to learn reflexives.

I do not want to suggest that the parsing approach should not be pursued in the development of computer assisted language learning programs. In fact, it is clearly necessary if these programs are to deal with free input from the student. However, as outlined above, there are several large problems that must be solved in the process of applying what we know about language structure in general, and computerized parsing routines in particular, to computerized language-learning programs. Therefore, I am suggesting another approach that does not seem to pose quite so many technical problems. Although it is applicable to a more limited domain than the parsing approach, it may prove quite useful as an intermediate stage in the development of "intelligent" computer programs. This approach involves the use of templates or generating systems.

Generating Systems and Templates

From the standpoint of generative linguistics, a generating system, or language generator, is simply the inverse of a parser. That is, a generative grammar is a formal system that associates each "phonetic form" (sentence) of a natural language with a "logical form" (meaning). It is similar to a mathematical function in that it is simply a set of ordered pairs, <sentence, meaning>. A parser is also a function in the less technical sense of a procedure which can be applied to an input (namely, a sentence of a language) and produces as output a structured meaning for the sentence. A generating system is simply the inverse function or procedure. That is, it takes as input a meaning and produces as output a sentence of a natural language that encodes that meaning.

It is easy to see why parsers have been a more popular area of research than generating systems. First of all, there is disagreement about what a "meaning" is or looks like. By contrast, a "sentence" is a fairly objective thing. So it is certainly more feasible to start with a sentence and work toward some sort of logical/semantic representation than vice versa. Second, in most cases, the practical goal toward which researchers are aiming is a program that can "understand" natural language input. Generally, computers have very little that they need to communicate, and this can be achieved with canned responses and simplified statements that do not require sophisticated processing. Therefore, a parser would be much more useful than a generating system.

Seen from this point of view, generating systems pose even more difficult problems than parsers, because there is no clear consensus on what the input (a meaning) should look like. The problems of adapting such a system to the analysis of incorrect student input seem even more formidable than for a parser. And the complexity of the program would certainly rival that of parsers.

There is another type of program that is generally referred to as a language generator. Given a formal grammar, with phrase structure rules (and possibly transformations), this program randomly generates grammatical sentences of the language defined by that grammar. This type of generator is completely divorced from any semantic considerations. It is fairly simple to implement technically, but its usefulness in error analysis systems is unclear. How could a list of grammatical sentences, randomly generated, help in analyzing student errors?

The approach described here, which I call a "template approach," falls somewhere in between these two technical types of generating systems. It is grounded in the fact that a constrained list of randomly generated sentences would be useful in the construction of a particular type of language-learning exercise, namely the transformation drill.

In this type of exercise, the student is presented with a set of sentences and told to perform some specified action to each of them. For instance, typical transformational drills include pluralizing noun phrases, changing the tense of sentences, turning statements into imperatives or questions, or replacing a noun phrase with an appropriate pronoun. In most cases, the meaning of the sentence is not of paramount importance. Although the sentences should not be semantically anomalous, the purpose of the drill is to inculcate certain syntactic transformations. The stimulus sentences are generally similar in syntactic form, and the student response is usually obtained by a straightforward syntactic manipulation of the stimulus sentence.

What I am suggesting then is a program which would allow the teacher to specify a syntactic template (using terminology of a non-technical nature, such as that found in most pedagogical grammars) for the generation of stimulus sentences. The teacher then specifies exactly how this stimulus sentence is to be transformed into the response sentence. Using this information, the program can generate any number of stimulus sentences and also automatically produce the correct response. Then, since it has an understanding of the steps to be followed to create the correct answer, it can easily check the student's response to see if those steps were followed. A program like this might be viewed as a rough, non-technical application of early transformational grammar to the construction of language-learning drills.

If this were all the program did, it might be a time-saver for the teacher, that is, if constructing the template and the transformation took less time than typing in the stimulus sentences themselves. However, using this approach, informative and relevant feedback can also be provided to the student by the teacher. This feedback is geared to the detection of errors at each step in the transformation of the stimulus sentence into the target sentence. In addition, by gathering information about the errors committed by the student in the course of a drill, immediate feedback to the generating system results in the generation of more sentences of the type which posed some problems for the student.

DESCRIPTION OF THE SYSTEM

In this section, I will explain in more detail the construction and operation of the proposed program. A prototype model of most of the system has been written, though with minimal implementation of each section and with no regard for user-friendliness. In order to facilitate an understanding of how the system works, I will also describe the functioning of the program in the construction of a particular transformation drill. This is a simple French pattern drill requiring the student to pluralize simple sentences in the present tense. For example, the stimulus sentences in (1) should prompt the responses in (2).

(1) Je vais au cinéma.
Le chat mange le poisson.

(2) Nous allons au cinéma.
Les chats mangent le poisson.

As can be seen in Figure 1, the system consists of four major parts: the template construction module, the transformation procedure module, the stimulus creation program, and the error analysis program. In addition there are some built-in, language-specific functions that provide short-cuts in the construction of templates and transformations. There is also a large "reverse lexicon." These will be discussed at the appropriate place as they are called upon by the four main programs.

Template Construction

Template construction can be visualized as the creation of a tree structure which reflects the various syntactic shapes that a stimulus sentence can take. Figure 2 shows a possible template for the construction of our example sentences. The tree is constructed in a top-down, nearly depth-first manner. Each node (branch-point) of the tree is named, generally with a mmemonic code representing the syntactic category of the leaves (terminal nodes) of the tree. For instance, the root node of the template constructed for our example might be called SENT, since the template will create sentences. Naming each node is important, since they can then be referred to when describing the transformation necessary to produce the target output.

The branches under each node must be one of three possible types: they may lead to options in stimulus construction, they may lead to required parts of the stimulus, or they may

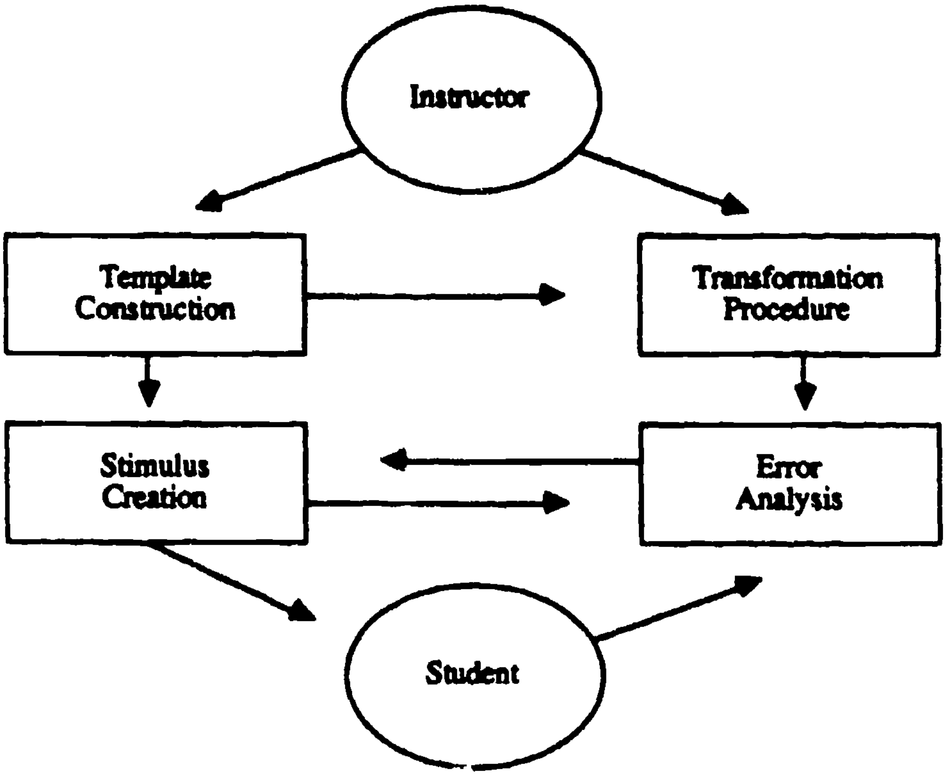


Figure 1. Model of a template construction system for error analysis

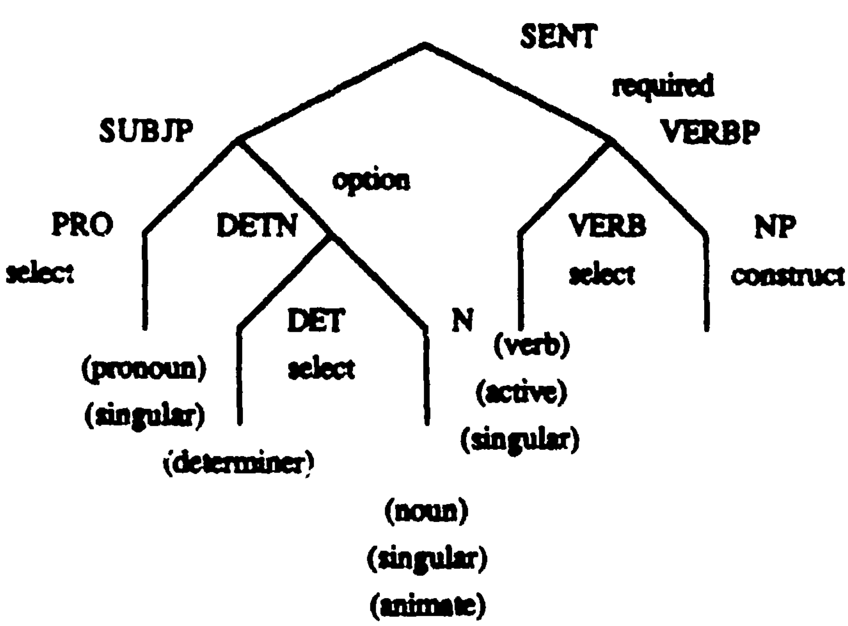


Figure 2. Template construction

lead to the selection of a lexical item to fill a spot in the stimulus sentence. The instructor must specify which type of branching occurs beneath each node. For instance, in our example, beneath the node SENT, the instructor specifies two other nodes, SUBJ and VERBP. These will subsume the subject and the verb phrase of each sentence. Since both a subject and a verb phrase are required for each sentence, each branch must be taken when constructing a stimulus sentence.

However, under the node labelled SUBJ, we find nodes labelled PRO and DETN. These nodes represent the construction of a subject consisting of either a pronoun or a determiner noun sequence. Since either is a complete subject and both together would be ungrammatical, the branches to these nodes are options for the stimulus constructing program--it must take one and only one of these paths.

Finally, a branch may lead to a terminal node (leaf) of the tree. In this case, a lexical item must be selected to fill the terminal node. The instructor is asked to specify the lexical category and the features that the lexical item must bear. In this case, the features "singular" and "animate" have been selected. In the PRO branch the lexical category "pronoun" has been selected and in the DETN branch the lexical category "common_noun." When a lexical item is to be selected, the instructor must also specify whether the lexical item selected must agree with any other item in the template. In our example, the determiner must agree with the subject noun and the subject noun with the verb. Both the feature system and morphology will be discussed in detail below.

In addition to the completely teacher-specified templates, there are additional built-in functions which can generate random constituents such as noun phrases, prepositional phrases, verb phrases, adverbial phrases, etc. If the content of these constituents is not relevant to the exercise at hand, the teacher need not specify completely their construction. For instance, in our example, the form and content of the object noun phrase is not relevant to the task of pluralizing the sentence.

Features

Features play an important part in this template system, since it is by means of features that much of the flexibility of the system is obtained. Most lexicons are organized as shown in (3), with each lexical item attached to a list of its own features. In this system, however, the feature names are entered with a list of the lexical items bearing that feature attached, as shown in (4).

(3)	chat	(noun masc animate common)
	plume	(noun fem common)
	manger	(verb active)

(4)	noun	(chat plume)
	common	(chat plume)
	verb	(manger)

This organization means that the selection procedure involves taking the sets associated with the features selected, intersecting them, and randomly choosing one lexical item from the intersection. This item will bear all the features selected by the teacher.

It is important to note that neither the lexicon nor the set of features is static. Words can be easily added and features can be easily assigned. This means that the selection process for lexical items can be closely controlled by the teacher. For instance, a feature could be created such as "noun from the vocabulary list for lesson nine." Using this feature, the teacher could limit selection to certain vocabulary items. The features could also be semantic in nature. A possible feature might be "articles of clothing" or "things to eat." Thus by the use of this flexible feature system, semantic constraints can be ensured.

It is also important to note that these features, whether syntactic or semantic, are intended to be inherent features of the lexical item itself. Morphological variation (primarily the agreement features) would be taken care of by independent pre-programmed modules. Therefore, to enter a regular lexical item it would not be necessary to enter more than the citation form. Morphological variation would be handled automatically. (Irregular variation would need to be entered into the appropriate computation module.) These functions, when provided with a lexical item and a set of morphological features, return the correct morphological form as their result. If a feature is left unspecified (say, for example, tense), it would be supplied at random. In addition, this functional approach to morphology allows closely related and even incorrect forms (a regular stem in place of an irregular correct stem, or a future stem with a present ending) to be generated and stored with the correctly generated form. These are used later to perform a primitive sort of morphological error checking.

Transformation Procedure

Once the template is constructed, the teacher must then specify exactly what actions are to be performed in order to create the correct target sentence. The transformation program operates from left to right, so changes must be specified in sequence through the sentence. For instance, in our example, the instructor must specify first changes to be made to the subject noun phrase and then to the verb phrase.

Any node of the stimulus template may be specified as subject to transformation. The instructor has four options to choose from in specifying the appropriate transformation of the stimulus sentence: copy, insert, delete, or alter. **Copy** results in a verbatim copy of whatever lexical items lie under the node selected. **Insert** involves the insertion of specified material. New material may be specified, or another node from a different part of the stimulus. Insertion must occur **between** constituents of the stimulus sentence. **Delete** is simply the removal of the specified material. Thus, **insert** and **delete** together provide a movement transformation. **Alter** involves changing the feature matrix of a specified lexical item, such as tense, number, and other syntactic/morphological features. In addition, there are available several pre-programmed functions which can provide certain commonly-performed feature alterations. These would be available to apply to larger scale constituents than lexical items. Such functions as changing tense, number, or case would be available, allowing the teacher to avoid directly manipulating the feature values themselves.

After specifying the constituent and the action to be performed on it, the teacher may provide an error message to be used if the student fails to execute properly this part of the transformation. Default messages for each kind of transformation are also available.

As can be seen from the above specifications of the stimulus template and the transformation construction, there is more than one way in which the same drill could be formulated. The instructor can choose how detailed a template to construct. The more detailed a template, the more detailed can be the specification of the transformation. The more steps in the transformation, the greater the number of "errors" the student may commit--each "error" being related to one step of the transformation. Thus the teacher controls how detailed the analysis of the student's work is.

Stimulus Creation

Given the template structures for a drill as specified by the teacher in the foregoing sections, the actual creation, presentation, and judging of a transformation exercise is fairly routine.

First, using the template, a stimulus sentence is constructed. That is, starting at the root node, the program traverses the stimulus template tree. If the node it is examining is has optional branches, only one branch (randomly chosen) will be traversed and the others ignored. If the branches are specified as part of a construction, each branch will be traversed in turn. If the node is a terminal node, an appropriate lexical item is selected. Agreement routines are executed. The tree-structure for the stimulus is stored and the lexical items are strung together and presented on the screen to the student.

Once the student has typed in a response, the judging sequence begins. Following the transformation schema specified by the teacher, the program checks each step to see that it has been successfully completed by the student. Thus, a copy step merely involves checking to see if the identical material in the stimulus sentence is in the student's response, and at the appropriate place. Insertion and alteration steps check for the appropriate inserted or altered information. In addition, if an error is found in an alteration step, morphologically similar forms generated earlier are checked to see if the student's response is among them.

Deletion of material cannot be directly checked since there is no way to see if something is not there, so evaluation is deferred. If the following step succeeds, it is presumed that the prior deletion was properly executed. If not, the program checks to see if the deleted material is still present in the student's response.

In each situation, if evaluation fails, the appropriate error message is printed. Thus, the student is led through the necessary transformations step by step.

Error Analysis

In addition to providing appropriate response to student errors, the program allows for a feedback loop from error analysis to stimulus generation, since stimulus sentences are spontaneously generated. This means that the stimulus sentences the student is presented with can be weighted to favor those types of sentences that proved difficult in earlier attempts.

For instance, data about which branch of the option node of a template was chosen is stored. Information about the number of errors a student made for each constitutent of each sentence is also correlated with the branches present in that sentence. This allows for the weighting of option nodes, so that options which resulted in sentences causing difficulty appear more frequently.

I want to emphasize the importance of this feedback loop, since it represents one of the significant advantages of this approach over a parser-based approach. Given an adequate parsing and error-analysis routine, a teacher would need only to type in the stimulus sentences, and the correct response. The parser could easily parse the correct response and use that as a target against which the student response would be judged. However, such a program could not generate new stimulus sentences, and thus could not individualize the drill appropriately for each student.

CONCLUSION

The "Intelligence" of the System

At this point, it is appropriate to summarize and highlight those aspects of the proposed approach which permit it to be somewhat "intelligent" in the creation of transformation drills and analysis of student responses.

In the first place, the teacher interacts with the program to create the drill. The teacher's intelligence is, therefore, integrated into the construction of the drill. This is made possible by several features of the program. First, the flexibility of the template construction process allows the teacher to specify in greater or lesser detail the structure of the drill template. Second, the ease with which features may be added to the system permits the teacher to tightly constrain the random choice of lexical items for the drill. Third, the allowance in the program for extended, specific error messages written by the teacher incorporates the teacher's knowledge of probable student errors.

In the second place, there is the built-in knowledge of the language that allows for "intelligent" drill construction. This knowledge includes the built-in lexicon and feature system, the construction functions which contain information about the grammar of certain constituents of the language, and the morphological functions which contain stored information about the regularities and irregularities of morphological systems in the language. This built-in knowledge interacts with the teacher's input to allow the instructor to concentrate on specifying the significant constructions for the purpose of the exercise and allows peripheral areas to be handled automatically, with minimal attention.

Finally, there is the feedback loop from error analysis programs to the construction of stimulus sentences for the drill. This allows "intelligent" construction of stimulus sentences.

That is, sentences are presented that help the student learn by concentrating on areas where the greatest difficulty lies.

Problems of Implementation

A few problems that may arise in the complete implementation of this system should be mentioned. These are mainly problems of scale. That is, there are questions of how well the system would work with a large lexicon, a large set of features, and a significant set of built-in functions.

The first concern is whether or not the response speed of the system can be maintained with such a larger data base. I suspect that a reasonable speed can be maintained, particularly if compiled code is used.

The second concern is whether or not the set of features and constructions necessary to handle a wide variety of constructions in the target language would make the system too unwieldy for the average teacher to work with. As with most systems, unforeseen problems will arise as it expands, demanding exceptional or ad hoc solutions. This may make the system too large to be handled by a typical foreign-language teacher without a great deal of linguistic or computational background.

The third concern is about the nature of the stimulus sentences generated by the program. There is currently a great deal of interest in communicative competence and the simulation of appropriate conversational situations. This program, with its limited semantic knowledge, will probably generate grammatically correct, but semantically odd sentences. How odd, it would be difficult to tell. To some extent, these odd sentences might be amusing to the student/learners. However, care must be taken that the oddness does not blend into "wrongness". It might be acceptable for the program to generate sentences like "Sally ate the stone". However, it should not generate a sentence like "Sally ate the sincerity." Avoiding sentences like the latter might overburden either the system or the teacher.

Finally, there is a concern about the current left-to-right procedure in template-construction, transformation, and response judging. A brief look at typical pattern drills shows that it would not pose unnatural constraints on the instructor to specify transformational changes in a strict left-to-right order. However, in some cases, it might be unintuitively or pedagogically better to work through the transformation in some way other than left to right.

The advantage of this strictly sequential method is that it avoids one of the major problems of a pattern matching approach. That is, in matching up the student input with the correct response pattern, it is very difficult for pattern matching to deal with misplaced constituents or scrambled word order. Of course, it also can match sequentially left to right, stopping if it fails to match exactly. But because there is no structure to the pattern, there is no way it can know more about the error than that it is not an exact match with the correct response. With the template approach, the program "knows" what each constituent in the answer should be. If it fails to match at a particular point, an appropriate error message is generated, helping the student correct the error at that point.

Summary

In this paper I propose an "intelligent" system of computerized drill construction. It operates interactively with the instructor to create a template for stimulus sentences and a sequence of transformations which change the stimulus sentence into the target sentence. Using a built-in system of features, morphological functions and constructions, the program presents stimulus sentences generated according to the template and judges the student response based on the steps in the transformation process. A feedback loop permits the generation process to be sensitive to errors committed by the student.

The positive features of such a system include the following: (1) it allows the instructor flexibility in constructing transformation drills that are specific to the instructional goals of the teacher and the learning difficulties of the student; (2) it encourages clarity on the part of the teacher in specifying clearly what the possible shapes of the stimulus sentences are and what must be done to produce the correct responses; (3) it enables flexible (goal-related) error analysis; (4) it permits the construction of an indefinite number of drill sentences, differentially for each student in relation to prior errors.

ACKNOWLEDGMENTS

I would particularly like to thank Dr. Robert Hart for his assistance with this article. Many of the ideas underlying this approach to error analysis I owe to him.

THE AUTHOR

Stephen Helmreich is a graduate student in the Department of Linguistics at the University of Illinois. He works as a research assistant in the Language Learning Laboratory.

Issues and Developments in English and Applied Linguistics (IDEAL)

Style Sheet for Contributors

1. Typing area should be 6.5 x 9.0 inches (16.5 x 23 cm). No text should exceed these margins.

2. All text material should be double spaced. The first line of each paragraph should be indented five spaces.

3. An abstract of not more than 200 words should follow the author's name. The entire abstract should be indented five spaces from left and right margins. Indent five additional spaces for the first line of each paragraph.

4. The title is centered and all capitals. The author's name is centered and in mixed capitals. The first level of headings is centered and all capitals. The second level is mixed capitals and flush with the left margin. The third level is underlined, on the same line as the paragraph opening, and indented five spaces. Only the first word and proper nouns of the third level are capitalized.

5. Footnotes should be numbered serially throughout the text using raised numerals. Each footnote number refers to a note at the end of the paper.

6. Notes, references, examples, tables, charts and figures are typed as illustrated in *IDEAL* (or *TESOL Quarterly*), except that all are double spaced.

7. The last line of the text is followed by ACKNOWLEDGMENTS—a statement of not more than fifty words, THE AUTHOR—a statement of not more than fifty words, NOTES, and REFERENCES. Each heading is centered and capitalized.

8. In general, articles and review articles should not exceed twenty pages in length in the form described above. Reviews should not exceed ten pages in the form described above.

9. Manuscripts should be sent in triplicate to the following address.

   Editor, *IDEAL*
   Division of English as an International Language
   3070 Foreign Languages Building
   707 South Mathews Avenue
   Urbana, Illinois 61801 U.S.A

10. Articles accepted for publication are returned to the author(s) copy-edited. The editor asks contributors to prepare the revised typescript in a form that can be photoreduced and printed without futher editorial work. To insure consistency of format, the editor will enclose a copy of the *IDEAL* Style Sheet for Preparing Articles in Photo-Ready Form to be followed carefully.

IDEAL
Volume 2
1987

Contents