# INTERACTIVE NATURAL LANGUAGE PROBLEM SOLVING: A PRAGMATIC APPROACH

A. Biermann*, R. Rodman**, B. Ballard*, T. Betancourt**, G. Bilbro**, H. Deas*, L. Fineman*, P. Fink*, K. Gilbert*, D. Gregory*, F. Heidlage*

* Department of Computer Science
Duke University
Durham, North Carolina

** Department of Computer Science
North Carolina State University
Raleigh, North Carolina

## ABSTRACT

A class of natural language processors is described which allow a user to display objects of interest on a computer terminal and manipulate them via typed or spoken English sentences.

This paper concerns itself with the implementation of the voice input facility using an automatic speech recognizer, and the touch input facility using a touch sensitive screen. To overcome the high error rates of the speech recognizer under conditions of actual problem solving in natural language, error correction software has been designed and is described here. Also described are problems involving the resolution of voice input with touch input, and the identification of the intended referents of touch input.

To measure system performance we have considered two classes of factors: the various conditions of testing, and the level and quality of training of the system user. In the paper a sequence of five different testing situations is observed, each one resulting in a lowering of system performance by several percentage points below the previous one. A training procedure for potential users is described, and an experiment is discussed which utilizes the training procedure to enable users to solve actual non-trivial problems using natural language voice communication.

## INTRODUCTION

A class of natural language processors is under development which allow a user to display objects of interest on a computer terminal and manipulate them via typed or spoken English imperative sentences. Such a processor is designed to respond within one to four seconds by executing the input command and updating the displayed world for user verification. If an undesired action is observed, a "backup" command makes it possible to undo any action and return the system to a previous state. The domains of interest include matrix computation, where one can display tables of data and manipulate them; office automation, where one can work with texts, files, calendars, or messages; and machine control, where one might wish to command a robot or other equipment via natural language input.

The first such system (Biermann and Ballard [6]), called NLC, provides a matrix computation facility and allows users to display matrices, enter data, and manipulate the entries, rows, and columns. It became operative in 1979 and includes a variety of special purpose features

1 This work was supported by National Science Foundation Grants MCS 7904120 and MCS 8113491, by the IBM Corporation under GSD agreement no. 260880, and by the Universite de Paris-Sud, Laboratoire de Recherche en Informatique during the summer of 1982.

including arbitrarily deep nesting of noun groups, extensive conjunction processing, user defined imperative verbs, and looping and branching features. More recently, a domain independent abstraction of the NLC system has been constructed and now is being specialized to handle a text processing task. In this system, text can be displayed and modified or formatted with natural language commands.

Current work emphasizes the addition of voice input, voice output, and a touch sensitive display screen. Speech recognition is being done on an experimental basis with the Nippon Electric DP-200 Connected Speech Recognizer in both discrete and connected speech modes, and with the Votan Corporation V-5000 Development System. The touch sensitive screen being used is a Carroll touch panel mounted on a 19-inch color monitor. Voice response is also provided by the Votan V-5000 which assembles and vocalizes digitally recorded human voice messages. The work has progressed to the point where our natural language matrix computer NLC is operative under voice control using the DP-200 and the text processing system is beginning to function using the V-5000 speech recognizer. The touch panel interface and voice response systems are still in the design phase.

The goal of the project is to make possible voice and touch interactions of the following kind:

> Retrieve file Budget83.
>
> Find the largest number in this column and zero it. (with touch input)
>
> Add this column putting the result here. (with two touch inputs)
>
> Send this file to Jones and file it as Budget83. (touch input)

That is, imperative sentences are to be processed that operate on domain objects to produce modifications to the existing objects or their relationship to each other. The objects are, for example, rows, columns, numbers, entries, labels, etc. in the matrix domain or sections, paragraphs, sentences, margins, pages, etc. in the text processing domain. The execution of each command is accompanied by an update of the displayed data with highlighting to indicate changes. Prompts and error messages will be given by voice response. System design is aimed at allowing fast interactive control of the objects on the screen while the user maintains uninterrupted eye contact with the events as they happen.

A continuous program of human factors testing has been maintained by the project in order to build a realistic view of potential users and to measure progress in achieving usability. For example, in a test of the matrix computation system with typed input, twenty-three subjects solved problems similar to those that might be assigned in a first course in programming (Biermann, Ballard, and Sigmon [7]). In this test, the NLC system correctly processed 81 percent of the sentences and users were quite satisfied with its general performance. Other tests of the system are described in Fink [14] and Geist et al. [15]. In another test (Fineman [13]), a simulator for a voice driven office automation system was used to obtain data on user behaviors when problem solving is with discrete and slow connected speech. It was found that users quickly adapted their speech to the required discipline of slow, methodical, and simple sentences which can be recognized by machine. Since the data obtained in any system test is heavily dependent on the amount and kind of training given to subjects, it is necessary to have a standardized training procedure. In the current work, a voice tutorial has been developed for training users to use a voice interactive system (Deas [11]).

This paper reports on the current status of these projects with emphasis on system design, speech input facilities and their performance, the touch input system and human factors considerations.

## SYSTEM OVERVIEW

The basic system design includes modules to do the following tasks:

(1) token acquisition
(2) parsing
(3) noun group resolution
(4) imperative verb execution
(5) flow-of-control semantics
(6) system output

The token acquisition phase receives typed inputs, word guesses from the voice recognizer, and screen coordinates from the touch panel. These inputs are preprocessed and passed to the parser which uses an augmented transition network to discover the structure of the command and the roles of the individual tokens. Noun group resolution attempts to discover what domain objects are being referred to, and the verb execution module transforms those objects as requested by the imperative verb. The flow-of-control semantics module manages the execution of meta-imperative verbs such as repeat, and handles user-defined imperatives. Finally, system output displays the state of the world on the screen. Any module may issue prompts and error messages via text or spoken output. Backup from any given module to an earlier stage may occur in unusual situations. More details appear in Ballard [1], Biermann [5], Biermann and Ballard [6], and Ballard and Biermann [3].

## SPEECH INPUT

An automatic speech recognizer such as the DP-200 or V-5000 recognizes speech by means of pattern matching algorithms. A subject is introduced to the device for a training session, and asked to repeat the various words of the vocabulary into a microphone. The device extracts and stores bit patterns corresponding to each vocabulary word uttered by that particular speaker. After training, when a speaker wishes to use the device, the appropriate bit patterns are loaded. Each utterance of the speaker is compared with the pre-stored bit patterns and the best match above a threshold limit is presented as the recognized word. Depending on the device being used, the speaker may be required to talk with discrete or connected speech. The results described below were obtained primarily in the discrete mode with a pause of at least 200 milliseconds after each word.

### Error Handling

The major difficulty facing users of automatic speech recognition equipment is the high error rate. Even the best devices in the best of circumstances are not entirely free of error, and when circumstances are less than optimal, and more like the real world, the error rate rises. Thus, a good part of the project effort has gone into coping with errors in recognition. In our view the speech recognition device is a component of the larger natural language computing system, and our goal is to reduce the system error rate as much as possible. We have therefore designed error correction software that corrects for certain kinds of errors, and error messages that elicit repetition from the human subject in less tractable cases.

Error correction essentially functions by starting with a sequence of word guesses from the input system and filtering out the meaningless alternatives at the appropriate stages of processing. Beginning in the token acquisition phase, certain unacceptable word sequences can be disallowed. For example, a noun such as "matrix" or "row" would be disallowed as the first word in the sentence since this is illegal in the system grammar. In the parsing phase, a grammatical sequence of words is selected from the incoming sets of word guesses. Thus all ungrammatical word sequences are eliminated. The parser also disallows phrases containing certain semantically unacceptable relationships such as

the second row in 6.

or phrases containing disallowed operations such as

Add the matrix to 6.

In the noun group processor and later stages, various other semantic errors can be eliminated such as references to nonexistent objects or impossible operations.

For discrete mode operations, errors are classified into four types:

a. Substitutions.
The device reports word B when word A was actually spoken.

b. Rejections.
The device sends a rejection code when a vocabulary word was spoken.

c. Insertions.
The device reports a vocabulary word when a non-vocabulary word, or noise, was uttered;

d. Fusions. Two (or more) words are spoken but only one word is reported.

### Substitution Errors

Substitution errors are the easiest to correct since the substituted word often resembles the actual word phonetically. Some of the substitutions are fairly predictable, e.g. "by" for "five", "and" for "add", or "up" for "of". We have coined the term synophone to describe such sets. Many synophone pairs are symmetrically interchangable; however, some are not. For example, with some speakers, the word "a" is frequently reported as "eight" although the converse seldom occurs.

Synophones of a particular word utterance come from two sources: alternate guesses offered by the recognition device based on its pattern matching computation, and a set of words stored in the system that are known to be confused with the selected word. Whenever a token is collected by the scanner, its synophone list is compiled. Passing the complete set of synophones for each word to the parser would result in excessive parse time so it is desirable to eliminate beforehand any synophones whose occurrence can be determined to be impossible based on grammatical or contextual considerations. For example the syntax of English (and of NLC) prevents certain words from occurring next to each other, or beginning or ending sentences. This information is recorded in a table of adjacencies. If there is a synophone in a word slot that cannot be preceded by any of the synophones in the previous word slot that synophone is deleted. This process is repeated until no more deletions are possible. On average, roughly one-half of the candidate synophones are deleted. Since parsing time may increase exponentially with the number of candidate synophones, and this table driven elimination process is very quick, considerable savings result.

For reasons of individual speech variation some vocabulary words will have synophones peculiar to an individual speaker. The set of synophones of each vocabulary word is therefore augmented to accommodate this situation so that each speaker has personalized synophone sets. Early training includes a tutorial introduction, part of which requires the subject to repeat sentences word for word. In this mode, the software has a priori knowledge of the correct token for each word slot. If a given word slot does not contain the correct token, the substituted word can be added to the appropriate synophone set for that subject. Thereafter, if the same substitution error recurs during a session with that subject, the correct word will be included in the synophone list for that word slot.

## Rejection Errors

The occurrence of one or more rejections in a sentence almost always results in a request for repetition. However, we are designing a number of facilities to handle rejections. In some cases, the rejected word can be determined from context, and processing can continue uninterrupted. Otherwise, the current plan is to handle a single rejection by returning an audio response that repeats all of the sentence with the word "what" in place of the rejected element. The speaker will then be able to choose to repeat the rejected word or, in case other errors are apparent, to repeat the entire utterance.

In cases of multiple rejection errors, the speaker is requested to repeat the entire utterance. In all cases previous utterances will not be discarded. The scanner will merge them, complete with synophones, in an attempt to eliminate rejections and provide the broadest amount of information from which to extract what the speaker actually said. For example, if the actual utterance were

A B C D E F G

and the recognizer returned

A B * Z E * G

where * stands for rejection, the speaker will be asked to repeat. If

A B C * E F H

is then recognized, it will be combined with the first utterance so that the scanner considers the seven word slots to contain:

```
s(A) s(B) s(C) s(Z) s(E) s(F) s(G)
                              s(H)
```

where s(X) is the union of X with its synophones. (Hopefully D is in s(Z).) If subsequent utterances are so different from previous ones that they are unlikely to be word-for-word repetitions (for example, by containing a different number of words), previous utterances will be discarded and processing will be started over.

It may also be possible to predict a rejected word with some degree of certainty based on semantic or pragmatic information. (We consider pragmatics to involve discourse dependent contextual factors.) For example suppose the scanner receives from the recognizer:

Double * nine and add column four to it.

The most likely possibilities for the rejection are entry, row and column. Entry can be eliminated on semantic grounds since it is meaningless to add a column to an entry. Row is semantically possible, but pragmatically less likely than column since adding columns to columns is much more common than adding columns to rows. Thus column may be chosen. Furthermore if the matrix in focus is six by seven, then the nine is a substitution error, and the sentence will be rejected on pragmatic grounds initially. However, since five is a synophone of nine the sentence will be tried with five in the place of nine. Ultimately the user will see displayed on the screen the result from:

Double column five and add column four to it.

The activity described above is transparent to the user. If the results are unsatisfactory to the user, the command "backup" will undo them.

An additional source of pragmatic error correction comes from utterances in historically similar dialogs. We are developing a method for utilizing this type of information. Considering the last example, if the user had been adding columns to rows quite frequently in the current and/or recent sessions, but rarely if ever adding columns to columns, the system would choose row as the rejected word.

## Insertion Errors and Fusion Errors

Most speech recognizers allow the threshold value to be adjusted that determines whether the best match is "recognized" or is rejected. Since rejections are harder to correct for than substitutions there is reason to lower this value. Too low a value, however, aggravates the insertion problem. When the speaker utters a non-vocabulary word, or emits a grunt or uncouth sound, the correct response is a rejection. A non-rejection in this situation may be difficult to deal with.

In our experience users have little trouble in confining themselves to the trained vocabulary. Most insertion errors occur between sentences, rather than between words within a sentence. This results in extraneous "words" in the first one or two word slots. These can often be eliminated because neither they nor their synophones can begin a sentence in the NLC grammar. Timing considerations, too, could be used to eliminate, or at least cast suspicion on, inter-sentence insertions, though we have not found the need for such measures.

We have observed fusion errors in discrete mode. They arise when the speaker neglects to pause long enough between words. In our experience they occur so infrequently we have not tried to compensate for them. This type of error is more crucial when operating in connected mode. It may be the case that two (or possibly more) words are reported as a single word different from either of the two originally uttered words. It may also happen that two words, A and B, are reported as either A or B. In this case the fusion error takes on the appearance of an omission. Our connected speech parser, currently under construction, will have the ability to guess an omission and insert a correction if sufficient contextual information is available.

## Raw Error Rate

Although a good deal of our interest is in correcting or compensating for the various kinds of errors in recognition, we are also working on ways to reduce the actual number of errors made by the recognition devices (the raw error rate). Careful vocabulary choice and proper tuning of the hardware such as threshold level selections are crucial factors.

It is important to choose vocabulary words as widely separated phonetically as circumstances allow. Additionally, we have found that words containing non-strident fricatives (e.g. the th in fifth), affricates (e.g. the ch in church), liquids (r and l) and nasals (m, n and ng) are more difficult to recognize than words containing other sounds. Monosyllabic words, in general, are not recognized as readily as polysyllabic ones, though words that are long and difficult to pronounce (e.g. anaesthetist) are also to be avoided. Often the domain leaves little latitude for vocabulary choice. If ordinal numbers are needed it is necessary to have fifth and sixth, which are difficult to distinguish. But instead of a word like rate which is easily confused with eight, tax rate or rate-of-pay (pronounced as a single word) might be a better choice.

Correct training procedures are instrumental in reducing the raw error rate as are such factors as whether the user receives immediate feedback from the recognizer, the form and frequency of error messages requesting repetition, and the degree of comfort felt by the user insofar as attitude toward computers is concerned. Some of these are discussed below in the section Measuring System Performance.

## Some Miscellaneous Questions

Apart from error correction, a number of other questions have arisen during our implementation of the voice driven system. Among these are:

a) How is the beginning of a sentence detected?
b) How is the end of a sentence detected?
c) How can a user make a correction in mid-sentence?

Currently a sentence begins with any input after the end of the previous sentence. The instances of inter- or pre-sentence insertions were discussed above.

Sentences are terminated by the meta-word over. This word has few synophones in the current word set and has the advantage of being widely understood to mean "end of transmission." However, we plan to experiment with other kinds of termination such as use of touch input or timing information.

A user may misspeak in instructing the computer to perform a task and may wish to repeat all or part of the command. Also, if the words from the voice recognizer are displayed as they are spoken, the user may desire to correct a misrecognition. The metaword correction is currently used to implement this facility. There are several levels of correction. Some may be accomplished by the scanner, while others require more information than is available to the scanner and must therefore be handled by the parser. The simplest type of correction consists of changing one word at the end of the sentence:

> Add row one to row four
> correction three.

Here the scanner merely deletes the word slot before the metaword. If several words follow "correction" as in

> Add row one to row two correction
> row one to column three.

the scanner detects this fact and scans backward in the sentence, attempting to match the largest possible number of word slots before and immediately after the metaword. In this example the tokens for row, one and to match, so the scanner copies the last part of the sentence into the earlier part of the buffer to arrive at

> Add row one to column three.

In the case of an utterance such as

> Add row one to row two
> correction column three.

it is impossible to match the tokens before and after the metaword. The scanner therefore deletes the token immediately before the metaword, flags the word slot preceding that token and passes the result to the parser. In the example,

> Add row one to row column three.

is passed, with the word slot containing row flagged. The parser attempts to make sense of the set of tokens passed. If it cannot, the flagged word slot is deleted, the word previous to it is flagged and another parse is attempted. The process is repeated until a successful parse is found. If none is found, an error message is issued. Thus in the example, after failing to parse the tokens as passed, the parser tries

> Add row one to column three.

which is parsed successfully.

## TOUCH INPUT

An important aspect of natural language communication is pointing, which is often used in connection with words such as this, that, here and there. Pointing may function as emphasis, as in

> Put the dog out.

where either the dog, the outside, or possibly both are pointed to. Pointing also functions to put objects into focus, allowing subsequent references to use a definite pronoun; for example,

> Move that there and cover it.

with a point to the object to be moved and covered.

A pointing ability would fit in very nicely with voice driven NLC and our project includes a touch sensitive screen so that the user can say "double this", point to a row, and cause the processor to double every element in that row. More complex sentences such as

> Add this row to that row putting
> the results here. (with three
> touches)

also become possible.

Apart from being "natural" in the sense that ordinary language users point often, pointing may increase the efficiency of communication.

There has been a good deal of interest among human factors scientists as to the efficiency of various modes of communication. Past experiments, for example, have compared the efficiency of typed versus voice messages (voice messages are more efficient). We carried out an experiment to verify the hypothesis that voice input together with touch input is more efficient than voice input alone, and we attempted to quantify the results. We solved eight different types of matrix problems including Gaussian elimination, divided differences and matrix inversion, using NLC without touch. We then went back and rewrote the solutions using the touch facility, but without any other changes. On the average 29% fewer words were needed to solve the problem, and individual sentences were shortened by 23%.

A number of interesting problems arise when a touch facility is implemented. One is how to pair up tactile and verbal input in the way intended by the user. Another problem is identifying the actual object the user intends to refer to once the tactile and verbal input have been resolved.

An example of the latter problem would be the command

Double this

accompanied by a touch of element <3,2> of a displayed matrix. Does the user want to double element <3,2>, double row 3, double column 2, or even double the entire matrix? The same touch paired with

Double this entry.

Double this matrix.

Double this column.

or

Double this matrix.

would be unambiguous. If the demonstrative is not accompanied by a nominal some strategy is needed to process the sentence. We opt for the smallest possible noun group encompassed by the touch (the <3,2> entry in the above case), and rely on our "backup" facility in case the user's intentions are not fulfilled. If the utterance "double this" is accompanied by a touch of the displayed name of a row, column or matrix, then the named object will be referenced.

Pairing up touches with spoken phrases is straightforward when a single noun group is used with a single touch, as in "double this entry." In a more complicated case we might have

Add this entry to that row
and put the result here.

accompanied by three touches. The strategy here is to pair touches and utterances in the order given by the user.

In the last example all touches functioned to establish focus or resolve noun group reference. If the emphasis function of touch is mixed in, a more difficult situation arises. If three touches accompany

Add this entry to the first row
and put the result here.

then the second touch was presumably to emphasize the first row or even to establish a rhythm of touching. In any case the facility to match touches with non-deictic expressions is needed. If only two touches accompany this last sentence then the focusing function should take precedence, and the touches should be matched with "this entry" and "here."

The situation is made even more complex by the ability to establish focus verbally. In NLC the user can say

Consider row four.

Double that row.

and the expression "that row" will refer to row four. If the same utterance is accompanied by a touch to a row other than four a potential conflict results. Our strategy is to give precedence to touch, since it is the more immediate focussing mechanism. Thus the sequence

Consider row four.
Double that row. (touching row three)

will result in the doubling of row three.

When both verbal and touch focus are present, nearly unresolvable ambiguities may result. The sequence

Consider row four.
Add this row to that row.

accompanied by one touch, gives rise to the problem as to which demonstrative noun group to associate with row four, and which to associate with the touch. One strategy is to associate with a demonstrative noun group the touch that occurred closest to the time of utterance. Another possible strategy is to assume that the expression with <u>that</u> refers to the more distant element in focus (the one established verbally in this case). This takes advantage of the fact that <u>this</u> and <u>that</u> can be distinguished in <u>English grammar</u> by the feature ±NEAR. Unfortunately by a simple change in stress pattern a speaker can undo this fairly weak regularity. Thus the sequence

Consider row four.
Add thís row to thàt row.

plus a single touch, where <u>this</u> bears primary stress and <u>that</u> bears secondary stress, should find the touch referring to "this row." If the stress pattern were

> Ádd this row to that row.

with primary stress on <u>Add</u>, the touch would more likely be associated with <u>that row</u>. It is unfortunate that to date we know of no voice equipment sensitive enough to distinguish between two such stress patterns.

Somewhat more complicated cases are possible:

> Consider row three.
> Add this row to that row and
> put the result in the first row.

accompanied by two touches. Since we allow a touch to occur with expressions such as "the first row," and since it is possible to disregard the element in verbal focus altogether, such a case produces multiple ambiguities. Although we foresee being able to resolve these ambiguities effectively, and can always fall back on our "backup" facility in case of mistakes, we also believe that such complex cases will be extremely rare. No sentence of such complexity was produced in our solutions to the eight problems mentioned above. With a voice and touch facility, sentences tend to be shorter and simpler.

NLC has implemented plurals, but we have not considered their use in touch input. Such sentences as

> Multiply these elements by
> this element.

or

> Add these elements up.

with multiple touches, would be useful. In the trial run of eight problems, the introduction of plurality resulted in up to fifty percent reduction in number of words needed and sentence length.

## MEASURING SYSTEM PERFORMANCE

Progress in any endeavor is greatly aided if the level of accomplishment can be measured in some meaningful way. It is desirable to give a figure of merit for a system both so that a project can indicate to the world the degree of the achievement and also so that the project can internally judge its own improvements over time. In voice language processing, one can attempt to measure performance by the word and sentence error rates. However, experience shows that these measures are highly dependent on two factors and that almost any level of performance can be reached if those factors are appropriately adjusted. Those factors are

(a) the environment and type of test within which the measurement is made, and

(b) the level of training of the system user.

### <u>Type of Testing Environment</u>

Considering (a), we tend to classify the type of test for a recognizer into one of the following five categories and we expect significant differences in device response in each case.

(1) Lists of words are read in tests performed by the manufacturer.

(2) Lists of words are read in our laboratory.

(3) Sentences are read in our laboratory. (discrete or connected)

(4) Sentences are uttered in a problem solving situation in our laboratory. (discrete or connected)

(5) Sentences are uttered in a problem solving situation in the user environment. (discrete or connected)

In the first situation, a manufacturer is interested in advertising the best performance achievable. Tests are performed in controlled conditions with microphone placement and all system parameters set for optimum performance, and an expert speaker is used. In our laboratory, we are not interested in the best possible system performance but rather what we can realistically expect. The parameters are set at medium levels, there is some ambient noise, the microphone may move during the test, and the user will be anyone we happen to bring in regardless of their speech characteristics.

As soon as the sequential words become organized as sentences, situation (3), the speaker begins to impose inflections on the utterance that will affect recognition. Certain words may be stressed, and intonation may rise and fall as the sequential parts of each sentence are voiced. Training samples based on reading lists of vocabulary items tend to be inaccurate templates for words spoken in context. When sentences are spoken in a problem solving environment, situation (4), these effects increase and other aspects of word pronunciation change. When voice control stops being the central concern of the speaker, larger variations in speech are bound to occur with accompanying larger error rates.

The most difficult situation of all occurs in situation (5) where the user might not even be a person who could be brought into a voice laboratory. In this case, the user has only one concern, achieving the desired machine performance. Encouragement to speak carefully could be met with impatience, and a few system errors could result in even worse speech quality and further degraded performance.

Our experience has been that word error rates increase from about three to seven percent as one moves to each more difficult situation type depending on the vocabulary, the equipment, and other factors. Consequently, we tend to distrust any figures gathered in the easier classes of environments and attempt to do our own testing in the more difficult and more interesting situations. Most of our recent data is of type (4) and we hope to gain some type (5) experience in the coming year.

## Training the System User

The second major factor affecting voice recognition performance is the level of training of the system user. Humans are extremely adaptive and capable of learning behaviors to a high degree of perfection. Thus the designer of a voice system might, over the years, learn to chat with it like an old friend whereas others might not be able to use the system at all. Again, almost any level of system performance can be observed depending on the quality of training of the user.

Our approach to controlling this factor has been to develop a standardized training procedure and to only report statistics on uninitiated users whose experience with the system is limited to this procedure. Ideally this procedure would be administered by machine to obtain maximum uniformity in training but this has not yet been possible.

The training procedure has two parts. The first part is an informal session in which the user is told how to speak individual words to the system and examples of the complete vocabulary are collected by the recognition system. The second part is administered very mechanically by reading a tutorial document to the user and requesting the utterance of trial sentences. This portion of the training introduces the user to the interactive system's capabilities and is specifically designed to be administered by the machine.

## Some Performance Data

An experiment was run during the summer of 1982 to obtain DP-200 performance data in an environment of type (4) as described above. Because no voice interactive system was yet available, a system simulation was used. After the first part of the training session in which the voice samples were collected, the subject was placed in a room behind a display terminal with a head mounted microphone. The voice tutorial was read to the subject through a loudspeaker at the terminal introducing the capabilities of the simulated system and the types of voice commands that could be executed. The subject's commands were recognized by the DP-200 and executed by the simulation. Thus each user command resulted in either appropriate action visible on the screen or a voice error message. In the final portion of the experiment, the subject was asked to solve an invoice problem that involved computing costs for a series of individual items and finding the tax and total. The experiment gave a reasonably accurate simulation of the expected NLC system behavior when it becomes completely voice interactive. The experiment attempted to simulate a syntactic level of voice error correction but nothing deeper.

It was found that the DP-200 word error rate rose to about 20 percent in this test with about 14 of the 20 percent being automatically correctable. The vocabulary size was 80, with three samples of most words, and six samples of a few of the difficult words, stored in the DP-200. This means that roughly every two to four sentences will have a single word error not correctable at shallow levels. This data comes from the first two hours of usage for these subjects and we expect significant improvement as usage experience increases over time.

More recently, the NLC system has become operative in a voice driven mode and subject testing has begun using the same training procedure. It is too early to report results but it appears that the performance predicted in the simulation will be approximately achieved. This experiment will include longer usage by the subjects and thus indicate how much error rates decrease over time.

In conclusion, we have at this time only fragmentary information regarding what levels of performance can be achieved. However, we have developed some tools for making measurements and will report the results as they become available.

## OTHER WORK

Much of the applied work in natural language processing has concerned database query (Bronnenberg et al. [8], Codd[9], Harris[17,18], Hendrix[22], Mylopoulos[27], Plath[29], Thompson and Thompson[32], Waltz[35], and Woods et al. [36]). At least one such system is being marketed (namely INTELLECT [18]), while several others have been successfully used in pilot studies. (Damerau[10], Egly and Wescourt[12], Hershman et al. [24], Krause[25], Tennant[31]).

As described in this paper, our initial work with NLC involved programming as an application area, while our more recent interest has shifted toward office domains. However, as Petrick[28] observes, many of the same technical problems arise regardless of application area. For the most part, the imperative sentence structures we are dealing with are simpler than the question forms recognized by the database systems cited above, while our noun phrases tend to exhibit more elaborate structures. Furthermore, whereas typical database systems process each input separately, or perhaps seek to handle ellipsis by consulting the immediately preceding input, we build up a richer semantic context as a session proceeds to be used in handling matters such as focus and pronoun resolution.

The most distinctive features of our present work are (a) the inclusion of voice input and output facilities, and (b) an attempt to deal with relatively "deep" relationships among domain objects. A more detailed discussion of the domain-independent mechanisms appears in Biermann[5], and as described in Ballard [2] the related LDC project being conducted in our laboratory is built around many of these techniques. Similar research projects which are moving away from a fixed database setting include work by Haas and Hendrix[16], Heidorn[20], Hendrix and Lewis[23], and Thompson and Thompson [33].

During the 1970's a number of speech understanding systems were developed under ARPA support (Lea [26], Reddy [30], Walker [34], Woods [37]) and currently some systems are being built in other countries, for example [19]. However, none of these systems has been refined to the point that it could actually support user interactions in real time as we are attempting to do. Our project uses well developed speaker dependent voice recognition equipment with a small enough vocabulary to achieve usable accuracy rates.

## REFERENCES

[1] B.W. Ballard, "Semantic and Procedural Processing for a Natural Language Programming System," Ph.D. Dissertation, Report CS-1979-5, Dept. of Computer Science, Duke University, Durham, NC, 1979.

[2] B.W. Ballard, "A Domain-Class Approach to Transportable Natural Language Processing," Cognition and Brain Theory, 5, pp. 269-287, 1982.

[3] B.W. Ballard and A.W. Biermann, "Programming in Natural Language: NLC as Prototype," Proceedings of the 1979 ACM National Conference, October, 1979.

[4] A.W. Biermann, "A Natural Language Processor for Office Automation," Proceedings of the 1982 Office Automation Conference, San Francisco, California, April, 1982.

[5] A.W. Biermann, "Natural Language Programming," to appear in Computer Program Synthesis Methodologies (Eds. Biermann and Guiho), Reidel, 1983.

[6] A.W. Biermann and B.W. Ballard, "Towards Natural Language Computation," American Journal of Computational Linguistics, Vol. 6, No. 2, pp. 71-86, 1980.

[7] A.W. Biermann, B.W. Ballard, and A.H. Sigmon, "An Experimental Study of Natural Language Programming," to appear in International Journal of Man-Machine Studies, 1983.

[8] W. Bronnenberg, S. Landsbergen, R. Scha, and W. Schoenmaker, "PHLIQA-1, A Question-Answering System for Data-Base Consultation in Natural English," Philips Tech. Rev., 38, pp. 229-239 and 269-284, 1978, 1979.

[9] E.F. Codd, "Seven Steps to RENDEVOUS with the Casual User," IBM Report J1333, 1974.

[10] F.J. Damerau, "Operating Statistics for the Transformational Question Answering System," American Journal of Computational Linguistics, Vol. 7, No. 1, pp. 30-42, 1981.

[11] H. Deas, M.Sc. Thesis, Dept. of Computer Science, Duke University, Durham, N.C., November 1982.

[12] D. Egly and K. Wescourt, "Cognitive Style, Categorizations, and Vocational Effects on Performance of REL Database Users," Joint Conference on Easier and More Productive Use of Computing Systems, Ann Arbor, Michigan, May 1981.

[13] L. Fineman, "Preliminary Results on the Voice Driven Information System Simulation Experiment," Report to IBM Corporation, Dept. of Computer Science, Duke University, Durham, N.C., 1981.

[14] P.K. Fink, "Conditionals in a Natural Language System" (Master's Thesis), Report CS-1981-8, Duke University, Durham, N.C., 1981.

[15] R. Geist, D. Kraines, and P. Fink, "Natural Language Computing in a Linear Algebra Course," Proceedings of the National Educational Computing Conference, June, 1982.

[16] N. Haas and G. Hendrix, "An Approach to Acquiring and Applying Knowledge," First National Conference on Artificial Intelligence, 1980.

[17] L.R. Harris, "User Oriented Data Base Query with the ROBOT Natural Language Query System," International Journal of Man-Machine Studies, pp. 697-713, September 1977.

[18] L. Harris, "The ROBOT System: Natural Language Processing Applied to Database Query," Proceedings of the 1978 ACM National Conference, pp. 165-172, 1978.

[19] J.P. Haton and J.M. Pierrel, "Data Structures and Organization of the MYRTILLE II System," Fourth T.I.C.P.R., Kyoto, Japan, 1978.

[20] G. Heidorn, "Natural Language Dialogue for Managing an On-Line Calendar," IBM Research Report RC7447, 1978.

[21] G.G. Hendrix, E.D. Sacerdoti, D. Sagalowicz, and J. Slocum, "Developing a Natural Language Interface to Complex Data," ACM Transactions on Database Systems, Vol. 3, No. 2, pp. 105-147, 1978.

[22] G.G. Hendrix, "Human Engineering for Applied Natural Language Processing," Fifth International Conference on Artificial Intelligence, pp. 183-191, 1977.

[23] G. Hendrix and W. Lewis, "Transportable Natural Language Interfaces to Databases," Annual Meeting of the Assoc. for Computational Linguistics, 1981.

[24] R. Hershman, R. Kelly, and H. Miller, "User Performance with a Natural Language Query System for Command Control," NPRDC TR 79-7, Navy Personnel Research and Development Center, San Diego, California, January 1979.

[25] J. Krause, "Results of a User Study with the 'User Specialty Language,' System and Consequences for the Architecture of Natural Language Interfaces," Technical Report 79.04.003, IBM Heidelberg Scientific Center, May 1979.

[26] W.A. Lea (Ed.), Trends in Speech Recognition, Prentice-Hall, 1982.

[27] J. Mylopoulos, A. Borgida, P. Cohen, N. Roussopoulos, J. Tsotsos, and H. Wong, "TORUS - A Natural Language Understanding System for Data Management," Proceedings of the Fourth International Conference on Artificial Intelligence, 1975.

[28] S.R. Petrick, "On Natural Language Based Computer Systems," IBM Journal of Research and Development, Vol. 20, No. 4, pp. 326-335, 1976.

[29] W.J. Plath, "REQUEST: A Natural Language Question-Answering System," IBM Journal of Research and Development, Vol. 20, No. 4, pp. 326-335, 1976.

[30] D.R. Reddy, "Speech Recognition by Machine: A Review," Proceedings of the IEEE, Vol. 64, No. 4, pp. 501-531, 1976.

[31] H. Tennant, "Experience with the Evaluation of Natural Language Question Answerers," Working Paper 18, Advanced Automation Group, Coordinated Science Lab., Univ. of Illinois, January 1979.

[32] F.B. Thompson and B.H. Thompson, "Practical Natural Language Processing: The REL System as Prototype," in Advances in Computers, Vol. 13 (Eds. M. Rubinoff and M.C. Yovits), Academic Press, New York, 1975.

[33] F. Thompson and B. Thompson, "Shifting to a Higher Gear in a Natural Language System," AFIPS Proc. of the National Computer Conf., Vol. 50, pp. 657-662, 1981.

[34] D.E. Walker (ed.), Understanding Spoken Language, Elsevier North-Holland, New York, 1978.

[35] D.L. Waltz, "An English Language Question Answering System for a Large Relational Database," Communications of the ACM, Vol. 21, No. 7, pp. 526-539, 1978.

[36] W.A. Woods, R.M. Kaplan, and B. Nash-Webber, "The Lunar Sciences Natural Language Information System: Final Report," Report 2378, Bolt, Berenek, and Newman, Cambridge, MA., 1972.

[37] W.A. Woods, "Motivation and Overview of SPEECHLIS: An Experimental Prototype for Speech Understanding Research," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 1, pp. 2-10, 1976.