Copyright © 1980 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 59, No. 1, January 1980
Printed in U.S.A.

A Conversational-Mode Airline Information and Reservation System Using Speech Input and Output

By S. E. LEVINSON and K. L. SHIPLEY

(Manuscript received June 8, 1979)

We describe a conversational-mode, speech-understanding system which enables its user to make airline reservations and obtain timetable information through a spoken dialog. The system is structured as a three-level hierarchy consisting of an acoustic word recognizer, a syntax analyzer, and a semantic processor. The semantic level controls an audio response system making two-way speech communication possible. The system is highly robust and operates on-line in a few times real time on a laboratory minicomputer. The speech communication channel is a standard telephone set connected to the computer by an ordinary dialed-up line.

I. INTRODUCTION

Recently there has been a substantial research effort in the area of speech understanding. Although the ultimate purpose of this work is clearly that of enabling natural spoken language human/machine communication, most of the work has actually been in the nature of building systems which transcribe speech. The system described in this paper is capable of conducting a complete spoken dialog with its user. The essential system architecture is hierarchical with three levels. These are an acoustic word recognizer, a syntax analyzer, and a semantic processor. The semantic level controls an audio response system which provides the speaking function. There is, of course, a significant interaction of the levels with one another.

The precursor to and incentive for this project was the flight information system built by Rosenberg and Itakura,[1] which consists of only a word recognizer and a voice response unit but which nonetheless is capable of conducting a limited dialog. The first two levels of the system and the voice response unit were taken in toto from previous investigations. The acoustic word recognizer was designed by Itakura[2] and is based on the computation of linear prediction coefficients (LPC), nonlinear time registration with stored reference patterns, and a maximum likelihood decision rule. The syntax analyzer is the maximum likelihood parser described by Levinson[3] and the voice response unit is based on an adaptive differential pulse code modulation (ADPCM) method of coding speech described by Cummiskey et al.[17] and used by Rosenthal et al.[4] The unit uses the hardware coder/decoder of Bates.[5] These building blocks have been explained in detail by their original designers and will not be further described here. The interested reader is encouraged to consult the cited references for a complete discussion.

This paper, then, is concerned with the theory and implementation of the semantic processor, the details of the incorporation of a complete system from the individual component parts, and the performance characteristics and implications of the integrated system.

The performance of the system is highly encouraging. The accuracy of the speech recognition portion of the system was reported previously by Levinson et al.[6] to be over 96 percent on sentences. With the addition of the semantic processor, 6 of the 21 sentence errors encountered in one set of test sentences were corrected without intervention by the user. In the remaining 15 instances, the system recognized the error and it was corrected by the user on his next input sentence. In no case was communication seriously disrupted. This phenomenon has a profound effect on a user of the system. His attention is drawn away from speech recognition accuracy and sharply focused on the exchange of information between himself and the machine. This points very strongly to the conclusion that progress in speech recognition can be made by studying it in the context of communication rather than in a vacuum or as part of a one-way channel.

The response time is currently about five times real time but can easily be reduced. The naturalness of the system is low due to the discipline required in speech input, and we are working to improve it. Overall, we are confident that our continuing efforts will result in increasing accuracy, flexibility, efficiency, and habitability of the system.

The next section of this paper gives the details of the integrated system, task, and architecture including a discussion of our implementation of it on a laboratory minicomputer. Section III is devoted to a description of the semantic processor, and system performance is evaluated in Section IV. The conclusion in Section V contains a brief summary of our results but is largely concerned with the implications of the system and the new directions in which we expect it to take us in further studies of natural language human/machine communication.

II. SYSTEM DESCRIPTION

In this section, we give a detailed description of the domain of the system and its operation in that domain. We present the system first as it appears to the user and then consider its architecture.

2.1 The flight information and reservation task

The goal of this project was to produce an on-line system which permits as nearly natural speech communication as possible. The system was to be robust, to understand the spoken input accurately over the standard telephone channel and respond quickly, and to require only moderate computational resources. To bring the goal within reach, certain constraints were deemed necessary. First, communication was restricted to pertain to a well-defined, limited subject. The flight information and reservation task is ideal for the purpose. It is a paradigm of the general data base information retrieval task for which natural language is appropriate. The tractability of the task domain allows us to impose some necessary restrictions on the flexibility of the language, limiting it to a small subset of natural English, which might otherwise be used for the purpose, generated by a formal grammar over a small vocabulary. Finally, we require that the input speech be disciplined in the sense that brief pauses between words are necessary. At the moment, the system must be trained by each of its users although this last restriction can be relaxed for a small additional increase in complexity.

Specifically, the flight information and reservation task includes 19 different semantic categories. Within each, several alternative and equivalent syntactic structures are permitted. The categories and a typical sentence pertaining to each are given in Table I. The vocabulary of the speech recognizer is 127 words. The language is finite (regular), having 144 states and 450 transitions in its state diagram and contains over 6×10^9 sentences. A detailed specification of the language is given in Ref. 3. The voice response unit has a vocabulary of 191 words. Sentences are generated by a context-free grammar described in detail in Section 3.2. The data base over which the system operates is the subset of the Official Airline Guide (OAG),[7] pertaining to flights from New York to nine American cities.*

2.2 Architecture

A block diagram of the system is shown in Fig. 1. Speech input to the machine is in the form of a sentence, *W*. Brief pauses of approxi-

* The language will actually support a larger data base. We have not yet constructed the necessary files.

Table I—Semantic categories

	Category	Sample
1	Information	I want some information, please.
2	Reservation	I would like to make a reservation.
3	Travel plans	I want to go to Boston on Monday evening.
4	General flight departure times	At what times do flights leave Chicago for Denver on Thursday afternoon?
5	Number of flights	How many flights go from Washington to Miami on the morning of the oh one May?
6	Aircraft type	What plane is on flight number five?
7	Fare	How much is the fare from Detroit to Seattle on Sunday?
8	Meals	Is a meal served on the flight?
9	Flight choice	I will take flight six one to Philadelphia.
10	Seat selection	I need two first-class seats.
11	Aircraft choice	I prefer the Boeing seven oh seven.
12	Exact time specification	I want to leave at six a.m.
13	Repeat information	Please repeat the departure time.
14	Specific flight times	When does flight number one to Los Angeles arrive?
15	Method of payment	I will pay by American Express.
16	Phone number	My home phone number is five three six two one five two.
17	Non-stop flight request	I would like a non-stop flight.
18	Elapsed time	What is the flight time from New York to Denver on Wednesday night?
19	Stops	How many stops are there on the flight to Miami?

mately 100 ms between the words permit segmentation of the sentence. Each word is individually recognized by the minimum prediction residual principle of Itakura,[2] which provides an acoustic transcription of the input, $\tilde{W}$, and a distance matrix $[d_{ij}]$ whose ijth entry is a measure of the spectral distance between the ith word in the sentence and the reference template for the jth vocabulary item. The parser takes this information and, using the technique described by Levinson,[3] produces the well-formed sentence, $\hat{W}$, having the minimum total distance. For efficiency, there is some communication between the acoustic and sytactic processing levels. Since the sentence is presegmented, the length of the sentence and the current word position can be given to the parser, which then returns a list of possible words to be matched to the input by the word recognizer. The process is described in detail by Levinson and Rosenberg.[8] The parser also produces an explicit derivation of $\hat{W}$ in the form of a state sequence, Q. Since the grammar which generates the language is unambiguous, Q and $\hat{W}$ suffice (almost) to define the semantic meaning† of the input.

The semantic processor takes Q and $\hat{W}$ and interprets them in the context of the conversation stored in the u-model to generate "actions"

† The precise definition of semantics is given in Section III. For the purpose of describing the system architecture, the reader's own, intuitive or otherwise, is adequate.

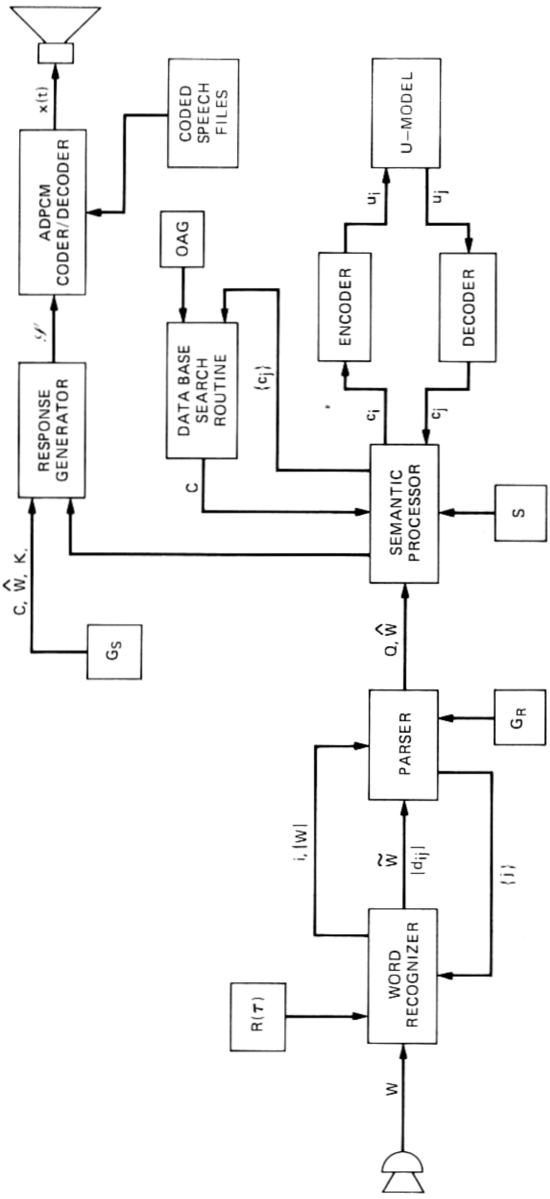

Fig. 1—Block diagram of the system.

which involve searching the data base, altering the context of the conversation, and generating a response. Although it is not conceptually important, for the sake of completeness we point out that the model of the conversation is encoded in an internal representation which is different from the one in the data base. Therefore, a translation process takes place between the semantic analysis and storage in the model. The external form of an item is denoted c_i and its internal code u_i.

The data base search routine can take a set of items $\{c_j\}$ and either match it to a complete flight description, C, or determine in what way the set is insufficient. A complete flight description can be used to answer a question and/or update the u-model.

The response generator takes the current flight description, C, the recognized input sentence, $\hat{W}$, and a semantic code, K, and generates a reply to $\hat{W}$ from the grammar G_s. The reply is a string of symbols, $\mathscr{S}$, representing a sentence, one symbol per word.

A subroutine which controls the ADPCM hardware uses $\mathscr{S}$ to access a file of pre-coded isolated words and concatenate them into the speech waveform $x(t)$ of the desired reply. Details of the voice response hardware and software are given in Refs. 5 and 4, respectively.

The system requires four distinct pieces of hardware, a laboratory minicomputer which in the present implementation is a Data General S-230, a CSPI MAP array processor which performs some of the computation for word recognition, the ADPCM coder, and a data set which provides the interface between the computer and the telephone network.

All other functions are implemented in software. Since the address space of the S-230 is limited to 32K, the software configuration is an overlay structure with some communication via disk files. The individual overlays, the word reference templates, $R(\tau)$, the data base, and the speech files reside on disk files totaling a few hundred thousand bytes of storage. The input and output grammars and the semantic table are core-resident.

III. HIGHER LEVEL PROCESSING

A well-known description of language due to Peirce[9] distinguishes among four aspects of language: symbolic, syntactic, semantic, and pragmatic. The symbols of a language are virtually arbitrary. Syntax is the relationship among the symbols, semantics is the relationship of the symbols to the objects they represent, and pragmatics is the relationship of the symbols to their users. In our system, the abstract symbols are words identified by the acoustic recognizer. The parser analyzes the formal syntax of the sentences, and the semantic processor relates these sentences to the data base and the task model. What little pragmatic information is used is embedded in other levels. In this section, we give a precise description of the highest level in the system hierarchy, the semantic processor and its associated routines. There are two distinct aspects of our type of semantic analysis, decoding and communication. Their definitions and operation are explained below.

3.1 Semantic decoding

The first phase of the relationship between abstract symbols and real objects is decoding, which we define simply as a mapping

$$S: (Q \times V) \rightarrow A, \tag{1}$$

where Q is the set of states in the state diagram of the language, V is its terminal alphabet or vocabulary, and A is a set of actions which we define precisely below.

The mapping of eq. (1) is used in the following way. Let $\hat{W}$ be the recognized input sentence and Q its state sequence with

$$W = v_1 v_2 \cdots v_n \qquad v_i \epsilon V \quad \text{for } 1 \leq i \leq n \tag{2}$$

and

$$Q = q_0 q_1 q_2 \cdots q_n \qquad q_i \epsilon Q \quad \text{for } 0 \leq i \leq n. \tag{3}$$

Then compute

$$S(q_i, v_i) = \alpha_i \epsilon A \qquad \text{for } 1 \leq i \leq n. \tag{4}$$

Since S is not necessarily defined for all state/word pairs, some α_i may be Λ, the null action. The set of non-null actions determines the response, $R(\hat{W})$ to input $\hat{W}$, which we denote by

$$\{\alpha_i \mid \alpha_i \neq \Lambda\} \Rightarrow R(\hat{W}). \tag{5}$$

The semantic mapping, S, comprises 126 rules of the form of eq. (4).

3.2 The task model

To precisely define the actions, α_i, we must look at the communication aspect of semantics. A well-known abstraction of the communication process is described by Fodor[10] and Minsky[11] as follows. For A to communicate with B, both must have a model or internal representation of the subject. A takes the state of his model and encodes it in a message which he transmits to B. B decodes the message in terms of his subject model and alters its state accordingly. Communication takes place to the extent that B's model is isomorphic to the state A's would be in had he received the same message. This is embodied in the task model, U, which is a finite universe of items which represent the categories in the data base which the system understands. Actions, then, mediate between the input, the data base, and the task model. An action, α_i, is a 4-tuple.

$$\alpha_i = \alpha_i(X, U_j, K, U_k), \tag{6}$$

where $X \epsilon V^*$ (usually $X \epsilon V$), U_j is the present configuration of the task model, K is a response code, and U_k is the new configuration of the task model. Thus the α_i are instructions for a classical finite state machine. The instructions correspond to the following actions: On input X with the present state of model U_j, respond with a sentence of form K and change the state of the model to U_k. The reader is reminded that a real category, c_m, in the data base is internally represented by the code u_m in the task model. The transformation between the two is a simple isomorphism which is shown schematically in Fig. 1.

There are 15 elements of the task model; these are defined in Table II. There are five ways to alter the state of the task model: Information can be directly given by the user; he can, for example, specify his destination, D. We denote this by

$$u_1 \leftarrow D. \tag{7}$$

Next we have default values which can be imposed. For example, unless otherwise specified, the number of tickets, N_t, is assumed to be one, and we have

$$u_{11} \leftarrow N_t = 1. \tag{8}$$

A data base lookup can also alter the state of the u-model as follows. A flight number, N_f, a destination, D, and a class, C, provide sufficient information to look up the fare, F, in the OAG. Thus

$$[u_1 = D \wedge u_6 = N_f \wedge u_7 = C] \overset{L}{\Rightarrow} u_{10} = F. \tag{9}$$

An element of U can be computed from the values of other elements, for example, flight time, T_f, determined by point of origin, O, destination, D, arrival time, T_a, and departure time, T_d. Origin and destination

Table II—Elements of the task model

Element	Symbol	Definition
u_1	D	destination city
u_2	M	meals served
u_3	D_w	day of the week
u_4	T_d	departure time
u_5	T_a	arrival time
u_6	N_f	flight number
u_7	C	flight class
u_8	A	aircraft type
u_9	N_s	number of stops
u_{10}	F	fare
u_{11}	N_t	number of tickets
u_{12}	N_p	telephone number
u_{13}	P	method of payment
u_{14}	T_f	elapsed (flight) time
u_{15}	O	flight origin city

supply time zone information, while arrival and departure time give elapsed time. We say, then, that

$$[u_1 = D \wedge u_4 = T_d \wedge u_s = T_a \wedge u_{15} = 0] \overset{\phi}{\Rightarrow} u_{14} = T_f. \tag{10}$$

Finally, an element of U can be computed from user-supplied information which is not part of a flight description and is not stored as such. For instance, a departure date uniquely specifies a day of the week, D_w, by

$$[n_m \wedge n_d \wedge n_y] \overset{F}{\Rightarrow} u_3 = D_w, \tag{11}$$

where n_m is the month, n_d is the date, n_y is the year, and F is the perpetual calendar function of Robertson.[12]

We can now give an example of a complete action. Suppose W was a request for the fare of a previously selected flight. Semantic decoding would enable action number 14:

$$\alpha_{14} = (\text{How much fare, } u_{10} = F \neq 0, K = 23, u_{10} = F). \tag{12}$$

That is, on a fare request, if u_{10} is some nonzero value, set the response code, K, to 23 and leave u_{10} unchanged. A value of $F = 0$ would indicate that a flight had not been selected as illustrated in eq. (9), and a different response code would be issued, causing a message so indicating to be generated. The complete ensemble of actions which the system needs to perform its task is composed of 37 4-tuples similar to that of eq. (12).

This brings us to consideration of the response generation procedure. Responses in the form of English sentences are generated by the context-free grammar, G_s.

$$G_s = \langle V_n, V_t, \sigma_0, P \rangle, \tag{13}$$

where V_n is the set of nonterminal symbols, V_t is the set of terminal symbols, a vocabulary of 191 English words, σ_0 is the start symbol, and P the set of production rules. The production rules are of two forms

$$\sigma_0 \rightarrow \gamma \in (V_n \cup V_t)^* \tag{14}$$

and

$$B \rightarrow b; \qquad B \in V_n, \qquad b \in V_t \quad \text{or} \quad b = \lambda, \tag{15}$$

where λ is the null symbol.

There are 30 productions of the form of eq. (14) in P. Each one specifies the form of a specific reply and is designated by a response code, K. There are several hundred productions of the type of eq. (15). Their purpose is to insert specific information into the skeleton of a message derived from a production of the other kind. As noted earlier, the specific information comes from the flight description, C, and/or the input, $\hat{W}$.

As an example, consider an input requesting to know the number of stops on a specific flight. The appropriate response code is $K = 26$ and the production rule to which it corresponds is

$$\sigma_0 \rightarrow \text{THIS FLIGHT MAKES } B_1\ B_2 \tag{16}$$

If $u_9 = N_s = 2$, then the following productions will be applied:

$$B_1 \rightarrow \text{TWO}$$

$$B_2 \rightarrow \text{STOPS} \tag{17}$$

resulting in the output string of symbols

$$\mathscr{S} = \text{THIS FLIGHT MAKES TWO STOPS.} \tag{18}$$

In the implementation, the words are represented by codes which are passed to the voice response unit. Each code is translated into an address in the speech file marking the beginning of the ADPCM representation of the corresponding word. These data are transformed into a speech waveform by the ADPCM hardware under software control.

3.3 System operation

To clarify the abstractions of the previous sections, we give a complete example of the system operation. Consider the input sentence, $\hat{W}$, I WANT TO GO TO BOSTON ON TUESDAY MORNING. The state diagram of the sentence is shown in Fig. 2, from which we immediately see that state sequence, Q, is

$$Q = (1, 2, 3, 7, 33, 11, 12, 13, 14, 15). \tag{19}$$

Four state/word pairs from S apply:

$$(33, \text{GO}) = \alpha_1$$

$$(12, \text{BOSTON}) = \alpha_2$$

$$(14, \text{TUESDAY}) = \alpha_3$$

$$(15, \text{MORNING}) = a_5. \tag{20}$$

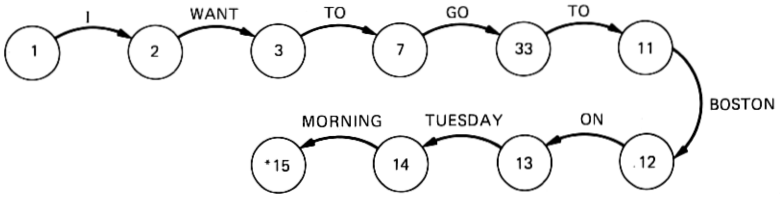

Fig. 2—State diagram of example sentence.

The actions invoked are the following:

$$\alpha_1 = (\text{GO}, U, 0, U_0)$$

$$\alpha_2 = (\text{BOSTON}, U_0, 0, U_1 \leftarrow U_0 + u_1 \leftarrow D)$$

$$\alpha_3 = (\text{TUESDAY}, U_1, 0, U_2 \leftarrow U_1 + u_3 \leftarrow D_w)$$

$$\alpha_5 = (\text{MORNING}, U_2, 1, U_3 \leftarrow U_2 + u_4 \leftarrow T_d;\ U_3 \overset{L}{\Rightarrow} C). \qquad (21)$$

Action α_1 causes the task model in any state to be initialized to state U_0 and no response to be made.

Next, α_2 changes the state from U_0 to U_1 by fixing the destination; no response is generated. Similarly, α_3 causes the day of the week to be defined. Finally, α_5 fixes an approximate hour of departure permitting a data base lookup which gives a complete flight specification. The response code is set to 1. The state of the task model after the lookup is shown in Table III.

The response code $K = 1$ causes application of the production rule

$$\sigma_0 \rightarrow \text{FLIGHT NUMBER } B_1\, B_2 \text{ LEAVES } B_3 \text{ AT } B_4\, B_5\, B_6\, B_7$$
$$\text{ARRIVES IN } B_8 \text{ AT } B_9\, B_{10}\, B_{11}\, B_{12} \qquad (22)$$

From the part of C corresponding to u_6, we have

$$B_1 \rightarrow \lambda$$
$$B_2 \rightarrow \text{THREE} \qquad (23)$$

From u_{15} we get

$$B_3 \rightarrow \text{NEW YORK} \qquad (24)$$

Table III—State of the task model after processing the input sentence of Fig. 2

U	C
$u_1 = 1$	Boston
$u_2 = 0$	no meals
$u_3 = 2$	Tuesday
$u_4 = 1000$	ten a.m.
$u_5 = 1047$	ten forty seven a.m.
$u_6 = 3$	flight number three
$u_7 = 1$	coach (by default)
$u_8 = 208$	DC-9
$u_9 = 0$	no stops
$u_{10} = 56$	\$56.00
$u_{11} = 1$	1 seat (by default)
$u_{12} = 0$	phone number unknown
$u_{13} = 0$	method of payment unknown
$u_{14} = 0$	flight time not calculated
$u_{15} = 7$	New York (by default)

From u_4,

$$B_4 \rightarrow \lambda$$
$$B_5 \rightarrow \lambda$$
$$B_6 \rightarrow \text{TEN}$$
$$B_5 \rightarrow \text{A.M.} \tag{25}$$

From u_1,

$$B_8 \rightarrow \text{BOSTON}$$

And finally, from u_5,

$$B_9 \rightarrow \text{TEN}$$
$$\mathrm{B}_{10} \rightarrow \text{FORTY}$$
$$B_{11} \rightarrow \text{SEVEN}$$
$$B_{12} \rightarrow \text{A.M.} \tag{26}$$

Thus, $\mathscr{S}$ is FLIGHT NUMBER THREE LEAVES NEW YORK AT TEN A.M. ARRIVES IN BOSTON AT TEN FORTY SEVEN A.M. The voice response unit then utters the sentence.

Clearly, then, the essence of the system is contained in the mapping, S, and the set of actions, A. S is implemented as a table of 126 entries of the form of eq. (4), and A is partially tabulated in the form of eq. (6) with auxiliary subroutines to perform the necessary operations. Neither table is uniquely determined by the language or the task. Both are constructed by a laborious and ad-hoc hand calculation. The algorithms implemented by the subroutines are well defined, but subject to the usual coding variations.

The above description is not intended to be complete and is given merely as an example of the operation of the system. It would be virtually impossible to list all the modes of operation of which the system is capable. Indeed, even for the example given, certain details are omitted for the sake of conceptual simplicity. From the sample dialogs given in the next section, the reader will be able to infer some of these details and other modes of behavior. One particular mode of operation is of particular interest and significance and will be discussed later. It occurs when the input is incomplete, or in disagreement with or logically in conflict with the current state of the task model.

3.4 Implementation

An important design criterion for the system was that it run on-line on a minicomputer with a 32-K word address space. Since the programs required are much larger than that, some form of memory management is required. Our first approach was to chain together as many independently executable modules as necessary, rolling the programs into memory from disk as needed. Although this implementation worked, it was quite slow for several reasons. First, the chaining process itself is slow. Second, every time a program is brought in, it must be reinitialized including reading in all necessary data files. Finally, all communications among programs must be via disk files.

The solution was to implement the system by an overlay structure consisting of a main program and four overlays for word recognition, parsing, semantic analysis, and voice response, respectively. The main program reads all the data tables in from disk, stores them in a common area, and then acts as a master which invokes overlay programs. Only the communication between the recognition and parsing subroutines is via disk. This memory management method is much more successful, as system overhead time is imperceptible.

IV. SYSTEM PERFORMANCE

4.1 Accuracy

In a previous study,[6] it was established that the recognition portion of the system is quite robust. On one test set of 351 sentences, only 21 were incorrectly recognized. The sentences were chosen at random and were not part of a dialog but, assuming that they were parts of normal conversations, six of the errors would have been corrected by the semantic processor. For example, when the word LEAVE in the sentence I WANT TO LEAVE AT NIGHT was rejected by the acoustic recognizer, the parser provided the word GO. From the standpoint of a transcription, this is an error. From the standpoint of the response of the system, no error occurred. In the remainder of the cases, the error would cause an inconsistency to occur in the task model. An appropriate response would be issued and the error corrected by the user. For example, in the sentence I WOULD LIKE TO RETURN TO NEW YORK ON WEDNESDAY, NEW YORK was recognized as SEATTLE. Since the origin of the flight is NEW YORK, the return cannot be to SEATTLE. The system responds WHAT DID YOU SAY, and the error is corrected on the next input. In none of the errors encountered would communication have broken down.

4.2 Sample dialogs

In fact, it is difficult to test our system exhaustively because, in addition to the large number of possible conversations, each input can occur while the machine is in a variety of states. We did, however, test the system with over two hundred sentences containing inputs from each semantic category and, while the machine was in three common states, u-model initialized, u-model containing the flight description of the flight being discussed, and *u*-model containing the description of a flight different from the current one. In all cases, the dialogs were perfectly understood. Some sample transactions are listed below.

The first is the simplest kind of interchange, a simple flight request.

Machine: THIS IS THE BELL LABORATORIES FLIGHT INFORMATION SYSTEM MAY I HELP YOU
User: I WOULD LIKE TO MAKE A RESERVATION PLEASE
M: PLEASE SPECIFY YOUR TRAVEL PLANS
U: I WANT TO GO TO BOSTON ON TUESDAY MORNING
M: FLIGHT NUMBER THREE LEAVES NEW YORK AT TEN A.M. ARRIVES IN BOSTON AT TEN FORTY SEVEN A.M.
U: I WANT TO LEAVE AT ELEVEN O'CLOCK
M: FLIGHT NUMBER FOUR LEAVES NEW YORK AT ELEVEN A.M. ARRIVES IN BOSTON AT ELEVEN FORTY FIVE A.M.
U: I WILL TAKE FLIGHT NUMBER FOUR
M: YOUR RESERVATION ON FLIGHT NUMBER FOUR TO BOSTON ON TUESDAY IS CONFIRMED THANK YOU.

A slightly more interesting conversation is the following:

M: THIS IS THE BELL LABORATORIES FLIGHT INFORMATION SYSTEM MAY I HELP YOU
U: I WOULD LIKE TO GO TO CHICAGO
M: PLEASE SPECIFY THE DAY OF YOUR DEPARTURE
U: I WANT TO LEAVE AT FOUR P.M.
M: PLEASE SPECIFY THE DAY OF YOUR DEPARTURE
U: I WANT TO DEPART ON THE TWO FIVE MAY
M: FLIGHT NUMBER NINE TWO LEAVES NEW YORK AT FOUR P.M. ARRIVES IN CHICAGO AT FIVE TWENTY FIVE P.M.
U: WHAT PLANE IS ON THE FLIGHT
M: THE AIRCRAFT IS A BOEING SEVEN TWENTY SEVEN
U: HOW MUCH IS THE FARE
M: THE ROUND TRIP COACH FARE IS ONE HUNDRED FORTY TWO DOLLARS
U: I WANT A FIRST CLASS SEAT
M: ONE SEAT THANK YOU
U: WHAT IS THE FARE
M: THE ROUND TRIP FIRST CLASS FARE IS ONE HUNDRED EIGHTY FOUR DOLLARS
U: PLEASE REPEAT THE ARRIVAL TIME
M: THE ARRIVAL TIME IS FIVE TWENTY FIVE P.M.
U: I WILL TAKE THE FLIGHT
M: YOUR RESERVATION ON FLIGHT NUMBER NINE TWO TO CHICAGO ON FRIDAY IS CONFIRMED THANK YOU

The following set of user inputs was carefully chosen to give the machine some problems.

M: THIS IS THE BELL LABORATORIES FLIGHT INFORMATION SYSTEM MAY I HELP YOU

U: I WANT ONE FIRST CLASS SEAT ON FLIGHT NUMBER THREE ONE TO DENVER ON SUNDAY

M: I AM SORRY THE FLIGHT IS NOT AVAILABLE ON SUNDAY

U: I WANT TO GO ON SATURDAY

M: FLIGHT NUMBER THREE ONE LEAVES NEW YORK AT NINE FORTY FIVE A.M. ARRIVES IN DENVER AT ONE TEN P.M.

U: WHAT IS THE FLIGHT TIME

M: THE FLIGHT TIME IS FIVE HOURS AND TWENTY FIVE MINUTES

U: HOW MANY STOPS ARE THERE ON THE FLIGHT

M: THIS FLIGHT MAKES ONE STOP

U: I WANT A NON-STOP FLIGHT

M: I AM SORRY THERE ARE NO MORNING NON-STOP FLIGHTS AVAILABLE

U: I WANT TO GO AT FOUR P.M.

M: FLIGHT NUMBER ONE SIX LEAVES NEW YORK AT FOUR FIFTY P.M. ARRIVES IN DENVER AT SEVEN P.M.

U: WHAT STOPS ARE THERE ON THE FLIGHT

M: THIS FLIGHT MAKES NO STOPS

U: IS A MEAL SERVED ON THE FLIGHT

M: DINNER IS SERVED

U: I WILL TAKE FLIGHT SIX FIVE

M: I AM SORRY THERE IS NO FLIGHT SIX FIVE TO DENVER

U: I WILL TAKE FLIGHT ONE SIX

M: YOUR RESERVATION ON FLIGHT NUMBER ONE SIX TO DENVER ON SATURDAY IS CONFIRMED THANK YOU

Questions need not occur in a logical sequence, as illustrated below.

M: THIS IS THE BELL LABORATORIES FLIGHT INFORMATION SYSTEM MAY I HELP YOU

U: AT WHAT TIME DOES FLIGHT FOUR FOUR TO LOS ANGELES DEPART

M: THE DEPARTURE TIME IS NINE P.M.

U: WHAT IS THE FARE FROM NEW YORK TO SEATTLE ON SUNDAY MORNING

M: THE ROUND TRIP COACH FARE IS THREE HUNDRED EIGHTY EIGHT DOLLARS

Finally, we observe that the machine can, to some extent, deal with nonsequiturs.

M: THIS IS THE BELL LABORATORIES FLIGHT INFORMATION SYSTEM MAY I HELP YOU

U: I WANT SOME INFORMATION

M: WHAT DO YOU WANT TO KNOW
U: I WILL TAKE THE FLIGHT
M: WHAT DID YOU SAY
U: IS A MEAL SERVED ON THE FLIGHT
M: FOR WHAT FLIGHT ARE YOU REQUESTING INFORMATION

This last dialog has some important implications which we shall discuss shortly.

From the above, the reader can easily observe that the dialogs which the system can sustain are not highly natural or sophisticated. The fact remains that speech communication, however stilted, has been achieved. We feel that two results of this achievement are of significance. First, when one interacts with a system which communicates in a merely vaguely natural way, his perception of the machine is changed. Conventional notions of speech recognition accuracy and algorithms for data retrieval assume a secondary importance as attention is sharply focused on transmission of information. It is quite clear that the state of the art in speech recognition is advanced enough to support research in complete human/machine communication systems. Second, the synergistic effect of integrating several crude components into an interactive system is to produce a machine with greater capacities than might otherwise be expected. As the sophistication of the components increases and as their interaction becomes more complex, their behavior will at some point become a surprise even to their builders.

As we noted in the previous section, the ability of the system to deal with nonsequiturs is of some interest. The system currently has no feedback paths from the highest level to either the acoustic or syntactic levels (see Fig. 1). If we assume that the system will be used by cooperative "customers," that is, those who want to communicate with it and would therefore not deliberately make confusing statements, then an inconsistent input can only be caused by an error of the word recognizer which could not be corrected by the parser. When an input appears to be semantically invalid, the sentence can be rejected and the next best one constructed by the parser. Thus we have a way to feed semantic information back to the lower processing levels.

This potential use of semantic information is not so important when the front end of the system is as reliable as the one we are using. However, when the input is continuous speech rather than strings of isolated words, recognition accuracy drops. For the same language and vocabulary, Levinson and Rosenberg[13] have reported an accuracy of 75 percent for continuously spoken sentences. In that case, the semantic information is necessary to improve the overall performance of the system.

4.3 Habitability

The system was not designed with habitability as a criterion but rather, as stated in the introduction, for robust communication. Nonetheless, it is important to evaluate the system from the standpoint of naturalness and ease of use. The single most unpleasant feature of the system is the requirement that spoken sentences have pauses between words. Clearly, continuous speech is a preferable mode. Recognition of continuous speech, however, is not so reliable as is recognition of strings of isolated words but, as we have observed, semantic processing increases recognition robustness. Thus we are in a position to deal with the more complex task of continuous speech recognition which, in turn, increases system habitability.

A similar, though less severe, problem exists in the speech output of the system. Output sentences are assembled from pre-recorded, coded, isolated word utterances. Although the intelligibility of the responses is unaffected, the loss of natural prosody degrades the quality. The deleterious effect of concatenation of isolated words on quality is somewhat mitigated by recording the words with a neutral or downward inflection, but this is not the ultimate answer. A more satisfactory result might be obtained by synthesizing the responses directly from their written form, $\mathscr{S}$, by a method similar to that used by Coker, Umeda, and Browman.[14] An appealing consequence of text-to-speech synthesis would be that some of the linguistic information required by the synthesizer can be used in the recognition phase. This is especially attractive in light of the motor theory of speech perception of Liberman,[15] which says that we are aided in our ability to perceive speech by our knowledge of how to produce speech.

One aspect of the semantic processor which detracts from system habitability is the inadequate memory of the u-model [see eq. (6)]. Since the present state of the model is dependent only upon the past state and the present input, there may be lack of continuity in the conversation. The unpleasant consequence is that a user must often repeat information. Consider the dialog on page 132. In the context of the previous question, the first-class seat request should cause the first-class fare to be given immediately. Clearly, several previous inputs and states should be retained.

Other areas in which habitability needs improvement are more technical than developmental. The response time is currently about five times real time, with most of it being devoted to the time registration procedure in the word recognition algorithm.[2] By performing this computation in the array processor, a capability which we presently possess, responses can be given in "psychological real time."

The effects of restrictions such as limited vocabulary size and rigid grammatical structure are manifest as increased requirements on user discipline. As recognition becomes more robust, these constraints can be relaxed. In fact, our design criteria were conservative so that, even with the present levels of reliability, we can tolerate a larger vocabulary and a more flexible syntactic structure.

Presently the system is speaker-dependent and, as such, requires a short training session by the user. Levinson et al.[16] have devised techniques whereby a word recognizer can be made speaker-independent without altering its structure. This technique, based on careful construction of reference templates, can be directly incorporated into our system at the expense of a small increase in response time.

V. CONCLUSION

We have described a conversational mode speech understanding and response system. By the addition of a new semantic processor to previously existing word recognition, syntax analysis, and voice response programs, we have constructed a system with which robust, on-line, two-way speech communication is possible over dialed-up telephone lines.

The results of two recognition tests show that the semantic level is capable of correcting errors in lower level processing and provides a method for interaction by means of which "misunderstanding" can be resolved.

Two important conclusions can be drawn from this study. First, one can and should consider speech recognition in the context of a communication task. Second, the "systems" approach of combining devices into larger and more complex structures produces new devices which are more sophisticated than the rudimentary nature of their components might indicate.

The system we have described has several shortcomings. It should be extended to accept connected speech; it would benefit from the incorporation of a speech synthesizer for response generation and it needs a more sophisticated task model. On the technical side, response time and speaker dependence must be reduced while vocabulary size and grammatical flexibility should be increased.

Despite the deficiencies which are manifest in the habitability of the system, its behavior is readily identifiable as natural language communication. We find this encouraging enough to warrant further investigations in the directions indicated by this study.

REFERENCES

1. A. E. Rosenberg, and F. Itakura, "Evaluation of an Automatic Word Recognition System over Dialed-up Telephone Lines." J. Acoust. Soc. Amer., 60 supp. 1, S12, 1976.
2. F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," IEEE Trans. Acoust. Speech Sig. Proc., *ASSP-23* (1975), pp. 67–72.
3. S. E. Levinson, "The Effects of Syntactic Analysis on Word Recognition Accuracy," B.S.T.J., *57*, No. 5 (May–June 1978), pp. 1627–1644.

4. L. H. Rosenthal, L. R. Rabiner, R. W. Schafer, P. Cummiskey, and J. L. Flanagan, "A Multiline Computer Voice Response System Using ADPCM Coded Speech," IEEE Trans. Acoust. Speech Sig. Proc., *ASSP-22* (1974), pp. 339–352.
5. S. L. Bates, "A Hardware Realization of a PCM-ADPCM Code Converter," Massachusetts Institute of Technology, M.S. unpublished thesis, Cambridge, Massachusetts, 1976.
6. S. E. Levinson, A. E. Rosenberg, and J. L. Flanagan, "Evaluation of a Word Recognition System Using Syntax Analysis," B.S.T.J., *57*, No. 6 (May–June 1978), pp. 1619–1626.
7. *Official Airline Guide*, Vol. 1, No. 11, Oak Brook, Ill.: Reuben H. Donnelley, March 1975.
8. S. E. Levinson and A. E. Rosenberg, "Some Experiments with a Syntax Direct Speech Recognition System," Proc. IEEE, *ICASSP-78* (1978), pp. 700–703.
9. C. S. Peirce, *Collected Papers of Charles Sanders Peirce*, C. Hartstone and P. Weiss, eds., Cambridge, Mass.: Harvard U. Press, 1935.
10. J. A. Fodor, *Language of Thought*, New York: Crowell, 1975, pp. 103 ff.
11. M. L. Minsky, "Matter, Mind and Models," *Semantic Information Processing*, M. L. Minsky, ed., Cambridge, Mass.: MIT Press, 1968, pp. 425–432.
12. J. D. Robertson, "Tableless Date Conversion," Comm. ACM, *15* (1972), p. 918.
13. S. E. Levinson and A. E. Rosenberg, "A New System for Continuous Speech Recognition—Preliminary Results," Proc. IEEE, *ICASSP-79* (1979), pp. 239–244.
14. C. H. Coker, N. Umeda, and C. P. Browman, "Automatic Synthesis from Ordinary English Text," IEEE Trans. Audio And Electroacoustics, *AU-21* (1973), pp. 293–298.
15. A. M. Liberman, F. S. Cooper, K. S. Harris, and P. F. MacNeilage, "A Motor Theory of Speech Perception," Proc. Stockholm Comm. Seminar, R.I.T., Stockholm, Sweden, 1962.
16. S. E. Levinson, L. R. Rabiner, A. E. Rosenberg, and J. G. Wilpon, "Interactive Clustering Techniques for Selecting Speaker Independent Reference Templates for Isolated Work Recognition," IEEE Trans. Acoust. Speech Sig. Proc., *ASSP-27* (1979), pp. 134–140.
17. P. Cummiskey, N. S. Jayant, and J. L. Flanagan, "Adaptive Quantization in Differential PCM Coding of Speech," B.S.T.J., *52*, No. 7 (September 1973), pp. 1105–1118.