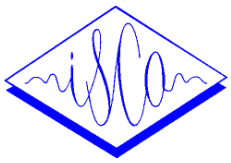

3rd International Conference on
Spoken Language Processing (ICSLP 94)
Yokohama, Japan
September 18-22, 1994

NATURALNESS OF THE INTERACTION IN MULTIMODAL APPLICATIONS

Jean-Claude Junqua[1] and Philippe Morin[1,2]

[1]*Speech Technology Laboratory, Panasonic Technologies Inc., 3888 State Street, Santa Barbara, California, 93105, U.S.A.*

[2]*CRIN-CNRS & INRIA Lorraine, France.*

ABSTRACT

In this paper we focus on the naturalness of the interaction in multimodal dialogue applications. After presenting some characteristics useful to assess naturalness in dialogue applications, we concentrate on the description of several techniques aiming at improving naturalness. We first describe a general algorithm to represent a contextual sub-language and briefly mention how this algorithm can be applied to word prediction and language acquisition for non-frequent users. Then, we focus on several mechanisms aimed at a more human-like interaction. We also report on the integration of these techniques in the multimodal dialogue system *PARTNER* currently under development.

I. INTRODUCTION

In man-machine communication robustness and naturalness are two crucial issues that primarily require the development of 1) flexible discourse models accepting natural input, 2) error management techniques and, 3) sophisticated feedback methods to achieve usability. For a better user acceptance a lively interaction is desirable and techniques are needed to reduce the dialogue monotony and improve its naturalness.

The naturalness of a speech-based dialogue system can be assessed using the following parameters:

- degree of constraints applied to the user. The dialogue can be completely directed by the machine (menu style dialogue) or can provide some flexibility. In this case the system asks highly directed questions but reasonable deviations on the user's part can be handled. At the extreme, fully mixed initiative dialogue allows the users to redirect the dialogue at any point, but this greatly increases the perplexity at every point in the dialogue. Redirection can also come from application events that need to be processed;
- quality and complexity of speech input: spontaneous speech, noisy speech, syntax-directed speech, incomplete sentences;
- context and reference management;
- cooperativeness: error correction, rephrasing capabilities, variability, automatic proposals;
- linguistic complexity and dialogue complexity;
- alternative input modes;
- response time;
- capability to deal with frequent and non-frequent users by adapting the dialogue and the guidance to the user level of expertise;
- management of the speech communication channel. Speech is not just an audible version of text and in human-human communication the transfer of information between talker and listener is going through many levels of abstraction. A man-machine dialogue should take into account these different levels of abstraction.

Some of these issues have already been addressed in the multimodal multilingual dialogue system *PARTNER* [3] and in [2] we already presented several techniques, such as assistance, automatic proposals and speech output variability, aimed at a more habitable man-machine interaction. However, the building of applications as in [1] showed the need for improving the response time, introducing automatic language acquisition for new or non-frequent users, imitating human-like behaviors in the management of the speech communication channel and improving user feedback. To speed up our system we improved the communication between the different modules and implemented a word prediction mechanism. In fact, the word prediction mechanism is derived from the development of a general algorithm suitable for

language acquisition and word prediction. As our system is based on a highly constrained natural language, it was important to guide the user through the use of the application language. Finally, to better manage the speech communication channel we developed human-like behaviors such as selective attention, talk-over, dialogue daemons (to take into account unpredicted application events) together with some techniques aimed at improving user feedback such as the use of an agent and input sensitive feedback. This paper will describe these techniques together with their integration in *PARTNER*.

II. SUPERVISED LANGUAGE ACQUISITION AND WORD PREDICTION

One of the important questions in multimodal dialogue is "*can spoken language be channelled?*" In [2] we presented a mechanism to build valid inputs interactively. We extended this mechanism toward a more general technique which can be applied to different problems such as word prediction and supervised language acquisition called, in [2], "*completion*". This technique builds intermediate structures representing the dialogue context from a dialogue history, a dialogue model and syntactic-semantic grammars containing the operational language. We developed a mechanism for text and speech input which provides the application, at each dialogue turn, with constraint tree structures representing a sub-language. The sub-language contains valid inputs that are syntactically and semantically correct in the current context. Constraint trees reflect syntactic constraints on the grammars that restrict the possible sentences that can be generated. They are expressed as pointers to the rules of the syntactic-semantic grammars.

Rule # Grammar Entries

```
Define <Color>  as
  {
#0  -> red            @:="Color=Red()" ;
#1  -> black          @:="Color=Black()" ;
#2  -> green          @:="Color=Green()" ;
#3  -> dark green     @:="Color=Green(Intensity=Low())" ;

    Default @:="Color=?()" ;
  }

 Define <Name>  as
  {
#4  -> alpha          @:="Name=Alpha()" ;
#5  -> beta           @:="Name=Beta()" ;

    Default @:="Name=?()" ;
  }

 Define <Item>  as
  {
#6  -> cube           @:="Cube" ;
#7  -> sphere         @:="Sphere" ;

    Default @:="?" ;
  }

 Define <NounPhrase>  as
  {
#8  -> the <Item>                          @:="Item=$(Attributes{})"(@1) ;
#9  -> the <Color> <Item>                  @:="Item=$(Attributes{$})"(@2,@1) ;
#10 -> the <Item> <Name>                   @:="Item=$(Attributes{$})"(@1,@2) ;
#11 -> the <Color> <Item> <Name>           @:="Item=$(Attributes{$,$})"(@2,@1,@3) ;

    Default @:="Item=?()" ;
  }

 Define <TopLevel>  as
  {
#12 -> <NounPhrase>           @:=@1 ;
#13 ->  Reduce <NounPhrase>   @:="Pred{Com=Reduce{Case1{Obj{$}}}}"(@1) ;
#14 -> <Color>                @:= @1 ;

    Default @:="?" ;
  }
```

FIGURE 1. Simple subset of a syntactic-semantic grammar for the manipulation of graphic objects.

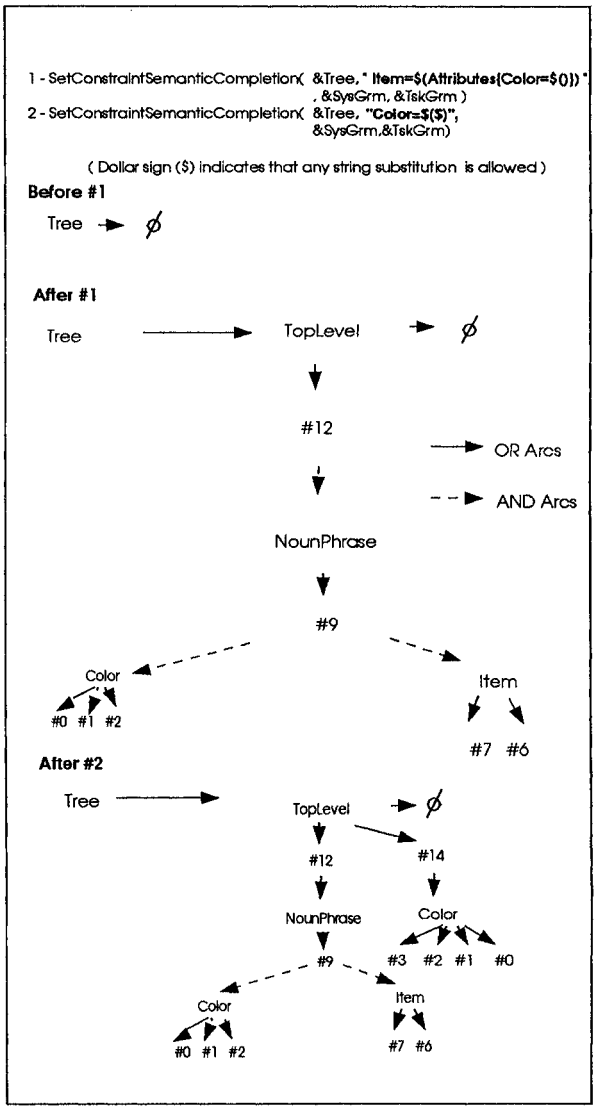

FIGURE 2. Building of a constraint tree for the two semantic expressions "Item=$(Attributes[Color=$0]) and "Color=$($)".

To restrict or constrain the generation, the dialogue manager builds semantic masks from the immediate and long-term context. Semantic masks are generative expressions based on the semantic language representation used by the dialogue system. They are contained in the parsing tables of the script instructions used for data input and are also built from the dialogue history (see [3] for a description of the dialogue system). Figure 1 shows a simple subset of syntactic-semantic grammar and figure 2 shows the generation of a constraint tree for the two semantic expressions *"Item=$(Attributes{Color=$()})"* and *"Color=$($)"*. Given a mask, a sub-tree is created. This sub-tree represents the constraints that must take place to generate the associated syntactic form(s) in relation with the grammar. For all the semantic masks which are valid in the current context the constraint tree is updated. After the processing of all the semantic masks, the constraint tree represents all the inputs that are syntactically and semantically correct for the current context.

This general algorithm has been used to generate a graphical interface for language acquisition and for word prediction towards a speech recognizer. In the case of language acquisition, inputs are built interactively by the user until a valid command is obtained. Then this command is executed at the user's request by the dialogue manager. The graphical interface consists of a series of menus representing the possible words (or remaining letters of a word in case of text input) available to the user in the building of a syntactically and semantically correct command at this point in the dialogue. The possible words (or word endings) are transferred to the application by an algorithm whose role is to provide a new word list after each user input.

In the case of word prediction the constraint tree is processed by an algorithm which generates a lexicon composed of all the words that can be used to build all the valid inputs. This lexicon is sent to the recognizer server which transfers it to the current active recognizer (see figure 3 for the dialogue-recognition interface).

III. MANAGEMENT OF THE COMMUNICATION CHANNEL

III.1 Selective attention

In human-machine communication as in human-human communication, it is often desirable to establish a connection between speaker and listener before starting a dialogue. In human-machine communication, this feature allows the machine to process user inputs only when the machine knows that the intention of the user is to communicate with the machine.

To activate this selective attention mechanism we defined a customized keyword in the system syntactic-semantic grammar. When the user wants to have a dialogue with the machine he needs to address the machine with the keyword before any dialogue can begin. Any other input is still being recognized by the active recognizer but it is filtered by the recognizer server. When the machine is addressed with the customized keyword a session is opened between the machine and the user and an agent is displayed on the user interface to indicate that a dialogue is going on. The session ends after an explicit user request of no user input for a customized period of time.

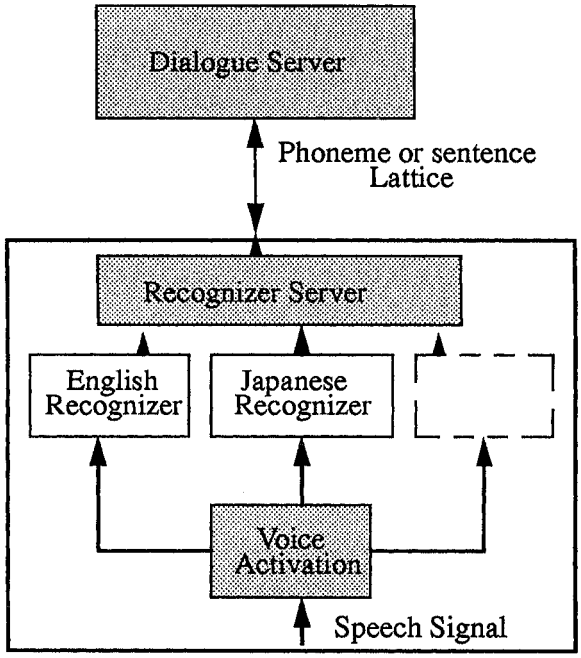

FIGURE 3. Block diagram of the speech recognition-dialogue interface

III.2 A talk-over mechanism

To improve the response time and diminish the constraints placed on the user we have developed a talk-over mechanism at different levels of our dialogue system. First, we use a frame synchronous voice-activation system which generates a message to the recognizer server as soon as a beginning speech boundary has been found. The voice activation algorithm uses the variance of band-limited energy [4] to detect beginning and ending boundaries. If a session between the user and the machine has been opened, the recognizer server propagates the message to the dialogue manager. If the dialogue system is executing a command it will complete the execution of this command but cancel all the speech outputs contained in the speech output FIFO resulting from the command being executed. As this mechanism is event-oriented, a real-time interaction can be

obtained. Preliminary evaluation of our demonstration system indicated that this feature is highly suitable for experienced users.

III.3 Dialogue daemons

In real-world applications situations can be encountered where the actual processing must be interrupted to handle specific conditions related to the application universe and its dynamic. Such conditions are time-out tasks, unexpected or abnormal reports that may require user intervention, warning messages, etc. Those conditions are initiated by the application.

In our new version of *PARTNER*, when specific conditions occur in the application universe, an application can interrupt the ongoing dialogue by sending a daemon request. A daemon tells the dialogue manager that a condition that requires dialogue interaction has been met. To process the daemon request the dialogue server suspends the ongoing dialogue, preserves its context and executes the daemon script that is linked to the event. Daemons have priorities attached to them. In order to be accepted by the dialogue server a daemon must be registered first in the scripts. The objective of the registering procedure is to make a link between the daemon request (represented by a daemon function) and the daemon scripts that represent the scenario to be executed. Daemons provide support for taking into account unpredictable application-oriented events. They proved to be very useful for handling the telephone line events in the voice activated telephone assistant that we are currently developing.

III.4 Toward an agent as a new modality

To convey to the user the idea of a conversation we associated to the machine an animated facial display synchronized with the machine speech (see figure 4). This involved the provision of speech output status from the dialogue to the application to animate the agent. We are currently extending this protocol to convey other kinds of informations such as nonverbal outputs representing the current understanding of the dialogue system. The visual agent represents dialogue states and expresses global information.

III.5 Input sensitive feedback

In a multimodal system every input media has its specificity. When speech input is used the machine should answer using speech output. However, when another media such as a pointing device is used graphical output may be better than speech output. To allow the customization of this application-dependent feature we developed an input sensitive feedback which can be enabled by the user providing that the application scripts have been programmed to take into account this feature.

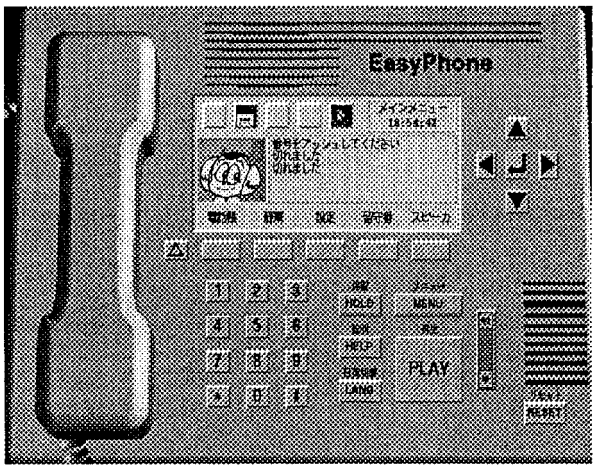

FIGURE 4. User interface (for Japanese) of a voice activated telephone assistant currently under development. For the agent, a robot type image was selected.

IV. CONCLUSIONS

To develop usable real-world applications it is necessary to focus on the development of techniques which will decrease constraints imposed by the machine to the user and improve the naturalness of the interaction. In this paper, we presented several mechanisms developed along these lines. They are currently being applied to the development of a voice activated telephone assistant (figure 4) whose goals are to emphasize customization and delegation.

Bibliography

[1] J-C. Junqua and P. Morin. Towards successful and usable applications using speech technology. In *ESCA-NATO RSG.10 Workshop, Applications of Speech Technology* , September 1993.

[2] P. Morin and J-C. Junqua. Habitable interaction in goal-oriented multimodal dialogue systems. In *EUROSPEECH-93*, pages 1669-1672, 1993.

[3] P. Morin, J-C. Junqua, and J-M. Pierrel. A flexible multimodal dialogue architecture independent of the application. In *ICSLP-92*, pages 939-942, 1992.

[4] B. Reaves and J-C. Junqua. Robust real-time preprocessing for speech recognition. In *Acoustical Society of Japan, Fall*, October 1992.