オーディオビジュアル複合情報処理 11－5
（1995. 12. 7）

言語に基づく仮想協調作業空間における仮想物体操作インタフェースシステム

ロペス R ガリバー† ティヘリノ ジュリ A‡ シンガー ハラルド§
吉田 美寸夫‡ 岸野 文郎‡

†国際電気通信基礎技術研究所
‡エイ・ティ・アール通信システム研究所
§エイ・ティ・アール音声翻訳通信研究所
〒619－02京都府相楽郡精華町光台2丁目2番地
gulliver@atr-sw.atr.co.jp

本稿は，仮想協調作業空間において言語に基づいた仮想物体操作のためのコマンド生成について提案したものである。本システムは、入力される言語に基づいてグラフィックスコマンドを生成するモジュールが、神輿の設計や自動車の外観設計といった対象ドメインに特殊な言語解析のための知識を持つことによって、適切なコマンドのみの生成を実現する。さらに、解析ルールを外部ファイルに記述可能な形式として持つことにより、特定のドメインに依存しない柔軟性を有するとともに、予想される音声コマンドの誤認識や言語の多義性を考慮することにより、ロバスト性を持つ音声インタフェースシステムを実現する。また，このコマンド生成モジュールを適用した高度なシステムについて紹介する。

A SPEECH-BASED COMMAND GENERATION MODULE THAT DRIVES INTERACTIONS IN A VIRTUAL ENVIRONMENT

R. López Gúlliver† Yuri A. Tijerino‡ Harald Singer§
Mikio Yoshida‡ Fumio Kishino‡

†ATR International
‡ATR Communication Systems Research Labs.
§ATR Interpreting Telecommunications Research Labs.
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02 Japan
E-mail: gulliver@atr-sw.atr.co.jp

This paper addresses the problem of flexibility and maintainability of systems allowing user's speech to drive interactions in virtual environments. Specifications for semantically valid speech-based commands and their appropriate processing to generate actions in the virtual world may vary depending on the application's domain. In this paper, the design and implementation of a proposed approach is presented. Application's domain independent semantic specifications, robust algorithm to deal with recognition errors and/or ambiguities of speech, easy integration with already existing virtual environment interactive systems are some of the main characteristics of our approach. A Japanese-based application and considerations of its application to more complex systems are also discussed.

1 Introduction

Visual, sound, and force feedback are among the most commonly used methods to present a user with a virtual environment. However, these all are only passive int the sense that the user is not actively modifying the virtual environment. On the other hand, manipulation of 3D objects has been limited to change object's position/orientation by means of a virtual hand.

Recently, speech-based commands in combination with hand gestures has been used to enable the user to not only manipulate, but also create and modify 3-D objects[5] However, valid context-dependent validation of speech-based commands, recognition errors and ambiguities in speech input and others have limited the complexity of its applications[1,4].

In this paper, we present a yet simple but powerful approach to cope with these problems. An application-independent scripting language to specify speech-based commands, a robust and stable algorithm to cope with ambiguities and errors in speech input, as well as guidelines about its general design are discussed here.

The paper also presents and discusses the integration of the module as a part of a Japanese-based application in a virtual space teleconferencing system[6]. The system combines speech-based commands (verbal commands) and hand gestures to give the participants an intuitive means of interaction with the virtual space they are presented. Current status and further plans for using this module in more general virtual envinronments are also discussed.

2 Problems in speech-based interactions in virtual environments

In this section, we identify the main problems related with speech-based virtual environment interactions.

Consider a system designed to drive speech-based interactions in a virtual environment. The general conception of one such system is depicted in Figure 1. Changes in the virtual environment are known to the user via visual, sound, force and other kinds of feedbacks. On the other hand, the user can modify the state of the objects in the virtual world at any time by means of hand gestures (ex. grabbing/releasing) and/or verbal commands. The speech recognition module captures the user's utterances. The recognized part of speech is then processed for validation, in a so called speech semantic validation module. Appropriate commands are sent to the graphics(model) module to update the state of the objects in the virtual world.

It is in the semantic validation module, where we focus our discussion in the paper.

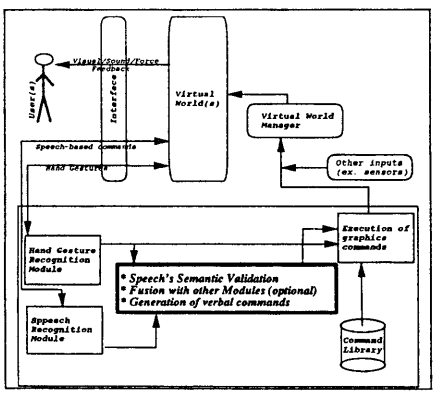

Figure 1: System overview of a prototype to use speech in virtual environments

2.1 Semantic validation of speech-based commands

In what follows verbal command and speech-based commnads are use interchangbly. Also speech considerations will be exemplified by the means of the Japanese language. Consider the problem of transforming verbal commands into actions in the virtual world. After receiving the output from the speech recognition module, the speech semantic validation module accepts or rejects it for execution according to the current status of the speech interaction. Acceptance or rejection means, whether or not the recognized phrases put together in a predefined order make any sense to the application. That is, even saying the valid words/noun-phrases "*migi_ni*" "*ugokasu*" "*teburu_wo*" ("*to the right*" "*move*" "*The table*") they may have no meaning to us if said in this particular order. Therefore we should impose some order to avoid complications when specifying the semantic rules the grammar should obey.

2.2 Specifying semantically valid speech-based commands

When semantically validating speech-based commands, we face the problem of whether accept them as valid or not. Therefore, we are to supply the system with a set of rules specifications about the appropriate semantically valid verbal commands in the domain context of our application. For instance, the sentence "Join object-A to object-B" has to be input in such a way the the system can *understand* what this means in terms of objects and their actions in the virtual world. A flexible and maintainable input procedure should be devised, so that application's domain dependent valid nouns, verbs and their relationships could be easily adapted to the system. We proposed an approach to solve these problems, details are given in Section 3.2.

2.3 User Interaction

Based on previous experiences [1,2,3] , we were able to identify the main problems and requirements of speech-based interactions in virtual environments. Questions as follows raised often for the user's interaction:

1. What sort of words/phrases can I use for interacting?
2. What is the general form of valid verbal command?
3. How to select objects to operate on them?

4. How can I manipulate and modify objects?
5. In case of undesirable errors how can I correct them?
6. Did the system get what I just said rightly?
7. Where am I now in the speech-command's sentence?

Questions 1 to 4 deal with the way the user's speech and computer interface is devised. These are prone to change depending on the application's domain and we take them into account during the design of our speech-based command generation module. Questions 5-7 try to make the interaction more intuitive and robust in case of speech recognition errors or simply user's mistakes.

3 A flexible module for speech-based interactions in virtual environments

In this section we present the design and implementation of a module that allows speech-based interactions in virtual environments. Our module is made as flexible as possible while solving the problems stated in the previous section. Our proposed module assumes the speech recognition system is pause-based[7,8] as shown in Figure 2. The user press a pedal to let the system know the start of the speech interaction. Saying nothing while pressing the pedal is considered a pause. With this feature, the user can have visual feedback after every pause. For example, if the bunsetsu "*teburu_wo*"("*table*") is uttered we can highlight the object which refers to the table in our virtual world, to let the user know that indeed "*teburu_wo*"("*table*") was properly recognized and accepted .

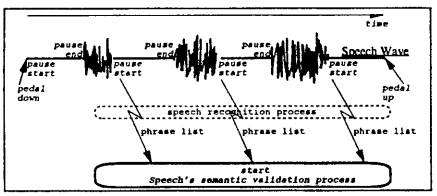

Figure 2: Pause-based Speech interactions

Continuous speech and hand gesture fusioning may need a different approach to the one presented here[9].

3.1 Maintainability of script files for specifications

To cope with the problem of specifying speech-based commands for semantic validation, we provide a simple scripting grammar[1]. The implementer of any specific application's domain has to be acquainted with it and write the specifications in an ASCII format file using a text editor. The scripting grammar consist of symbols and reserved words as follows:

[1]We choose this scripting grammar because it fits perfectly with basic Japanese language grammar rules

```
Symbols:
    ...
    #   ------------------ Comment symbol
    <>  ------------------ Generic symbol
    []  ------------------ Particular symbol
    -   ------------------ Range symbol
    /   ------------------ Logical ''or'' symbol
    !=  ------------------ Exception symbol
    <-> ------------------ Equality symbol
    ...

Reserved Words:
    ...
    meishi ----------- Generic noun(object)
                       used as <meishi_wo>
    shijidaimeishi --- Generic demonstrative
                       pronoun(this/that),
                       used as <shijidaimeishi_wo>
    dooshi ----------- Generic verb(action),
                       used as <dooshi>
    houkou ----------- Generic direction
                       (up, down, right, ...),
                       used as <houkou_he>
    ...
Any single word:

    any-word ---------- Any word, a noun,
                        verb or other,
                        used as [any-word]
```

Table 1. Scripting Grammar for verbal commands specifications

Here, "meishi" and "shijidaimeishi" ("*noun*" and "*demonstrative pronoun*") could have japanese particles attached to them, as "wo", "ni", "kara", "no" and others, indicating the direct or indirect object in the sentence. Also, "houkou" ("*direction*") is usually followed by the particles "he" or "ni".

There are simple rules for writing specifications into a file, say: 1) Use any of the above symbols and words, separated with blanks, in the desired order to form a valid verbal command in the context of your application's domain. 2)A newline or a "#" will mark the end of the sentence-command. We also need to give the system information about the vocabulary we want to use, this depending on the application's domain. 3) The vocabulary part of the file would be a list of "word < − > part-of-speech" pairs, and for nouns and verbs the actual name of the object or action as known by the virtual world manager module. A typical specification file using this scripting grammar and rules would look as the one below:

```
...
# Specifying our valid commands
    <meishi_wo> <meishi_ni> [kutsukeru]
    <meishi_wo>             <verb!=kutsukeru>
    <meishi_wo> <houkou>    [ugokasu/kaitensuru]
    ...
# Defining our vocabulary
# This could be as big as needed...!!!
    yane        <-> meishi          <-> ROOF
    katsugiboo  <-> meishi          <-> BAR
    this/that   <-> shijidaimeishi
    ...
    kutsukeru   <-> dooshi          <-> JOINT
    ugokasu     <-> dooshi          <-> MOVE
    kaitensuru  <-> dooshi          <-> ROTATE
    ...
```

```
migi        <-> houkou  <-> RIGHT
hidari      <-> houkou <-> LEFT
...
barabara    <-> any-word
...
```

Table 2. A verbal command specifications file

The first three examples in Table 2 would mean:

- "Join any two given objects"
- "Do anything to a given object except joining"
- "Move or rotate any given object in the given direction"

The user during his/her interaction with the virtual environment, *must* give his verbal commands following this same predefined order. Also in Table 2, "roof, bar, joint, move, rotate" are the actual names of objects and actions as in the virtual world manager module. To be used to generate the verbals commmands acting on the grafic objects in the world.

We should not here that this scripting grammar is for specifying *sentence-based verbal commands* and that their word order is relevant to have any meaning to the application's domain. In this sense this grammar differs with those for specifying the parts of speech in the speech recognition module, where there are only isolated single phrases and order and meaning is not relevant.

3.2 Semantic Validation of verbal commands

After specifications for semantically valid commands in the context of the application's domain are given, the module reads these specifications and form, what we call a *semantic network*. This semantic network is used as part of a general algorithm to accept or reject any verbal input from the user.

3.2.1 The semantic network

The verbal command module constructs the *semantic network* based on the specifications as described in the previous section. Figure 3 shows an example of a part of such network:

In this example the user have said "yane_wo migi_ni" ("*roof to the right*"), the bold lines describe the flow of the currently accepted part of the verbal command, that is "meishi_wo −− > houkou". Here the network's status depth is 2. Then according to the so constructed semantic network, "ugokasu" and "kaitensuru" ("*move*" and "*rotate*") are the only expected phrases. That means that these and only one of these two phrases are to be accepted as the following speech input from the user.

Note that this semantic network is separated from the specifications and the implementer of the specifications does not need to know about its details. This gives flexibility to the system, by separating application's domain dependent parts from our main speech processing module.

3.2.2 The semantic validation algorithm

Here we present a simple but useful algorithm to validate sentence-based verbal commands. It is intended to make the flow of speech during the interaction more

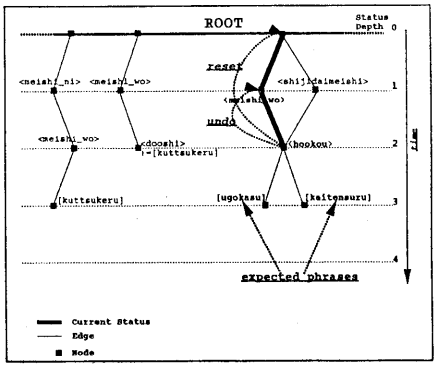

Figure 3: Speech flow and the semantic network

natural and flexible. It assumes we get, from the speech recognition, a 10-word list and their respective recognition probabilities. The algorithm uses this fact, to look down for a *suitable word* up to a certain limit of accuracy. Figure 4 shows how this is done.

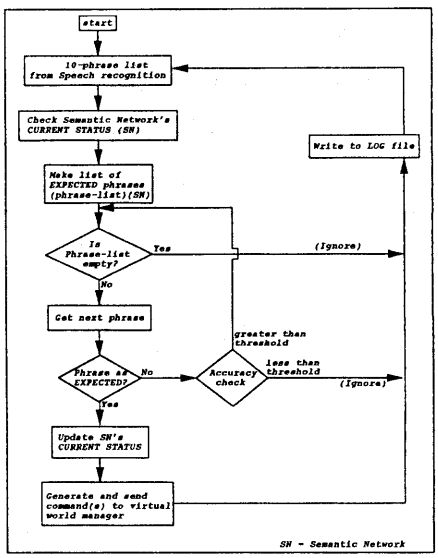

Figure 4: Validating speech-based commands

The algorithm checks the semantic network current status and also of the 10-phrase list of possible recognized words from the user. If the phrase with the highest probability is not accepted for validation we keep looking down up to a certain limit of accuracy(high probability) for a *now-expected* phrase and, if appropriate, generate the feedback to the user or send a command to the virtual world manager module. If nothing

appropriate is found we simply ignore what the user said and leave the current status of the user's interactions as is, as well as making the virtual world blink to let the user know that something went wrong.

3.2.3 Generating commands for the virtual world

After a verbal command is completed, the module sends a message of the form "object action [arguments]". Where *object* and *action* refers to the name of the object and the action, as known by the virtual world manager module, respectively. *arguments* are sent only if necessary to indicate for instance the direction or grade of deformation.

With the help of the validation algorithm and the semantic network, the generations of commands for interacting in virtual environments is reduced to the rather simple task of packing and sending messages to objects in the virtual world. These objects are to respond to the messages sent to them according to their own behavior. The speech-processing module is not supposed to know whether or not an action is valid for an object. That is, we assume that the virtual manager is designed object-oriented in this sense.

3.3 Other characteristics

3.3.1 Off-line debugging

As the last step in the semantic validation algorithm we write all the messages sent or the rejected ones to a log file for off-line debugging purposes of the user's session. This is particularly good for analyzing users' session common errors or tendencies a posteriori, and a means to identify new functionalities to improve the module.

3.3.2 On-line error correction

In order to make it user-friendly, we added some default functionalities to the module. The user can say "*chigau*"("*undo*") to correct his verbal command in course one step back. Also he can use "*risetto*"("*reset*") to restart a new command from scratch. This is illustrated in figure 3. Optionally, the module provides a visual text window feedback to let the user know what is the current status of the verbal command in course.

3.3.3 Supporting other modules

The module can be used to fuse other modules information with the speech input to make the degree of interaction more accurate. In order to support demonstrative pronouns, and/or allow the user to specify the degree of deformation or movement of an object, our module requires to have access to the hand gesture recognition module and extract from it the necessary values to execute the task. For instance, "*katsugiboo_wo* **konogurai** *nagakusuru*" or "**kore_wo this** *ugokasu*" ("*Make the bar* **this long**" or "*Move* **this**") requires the speech recognition module to merge, at the appropriate point of time, the information of the user's hand gesture with the speech information . Figure 5 illustrates this. Although this functionality is now supported by the module it is dependent on the hand gesture recognition module.

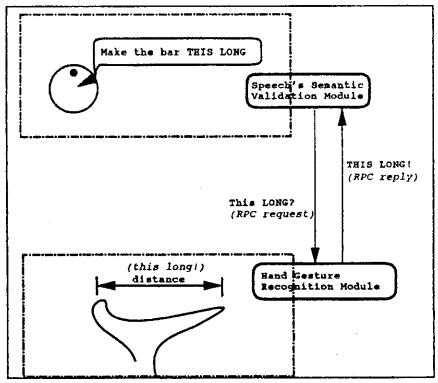

Figure 5: Speech and hand gesture modules communication using remote procedure calls(rpc).

4 Flexible integration with a virtual environment

The above discussed speech-based command generation module was implemented on a SGI Indigo-2 under IRIX 5.2, using remote procedure call libraries, and the C programming language.

The module was included as one part of the speech-based interaction module with our ATR Virtual Space Teleconferencing System to realize the WYSIWYS[5] framework.

This system put up to three people remotely located in a shared virtual room to conduct a teleconference. As a demonstration, the system presents the participants with the parts of a Japanese portable shrine (*mikoshi*) and let them cooperatively assemble it and/or modify its shape by means of verbal commands. Japanese was used as the natural language for interactions. The user can generate, manipulate and change the shape of the virtual components using verbal commands like:

- "*heya_wo haichisuru*",
- "*kamisori_wo hidari_ni kaitensuru*",
- "*yane_wo takakusuru*"

("*make a chamber*", "*rotate the base to the left*", "*make the roof taller*")

Integrating our module to the whole system was rather simple. We only had to give a list of specifications as the ones described in section 3 in form of ascii text files. Small changes in interactions were possible by only modifying these files. Users of the system find it rather easy to edit and manipulate the objects using verbal commands. This is due to the flexibility of the semantic validation algorithm and the "*chigau*" and "*risetto*" ("*undo*" and "*reset*") facilities. On the other hand, every time the system is used, we can collect and, afterwards analyze, user's interactions data by just looking at the log file written by the off-line debugging facility. With this in hand, later considerations for improvements to the human-machine interface of the system can be made.

5 Conclusions

In this paper, we introduced the design and implementation of a flexible and easy maintainable module to drive speech-based interactions in virtual environments. Its generality and application's domain independent features make it a promising approach to be used in simple to moderately complex interactions in virtual reality applications.

This approach proved its flexibility as being easily integrated in an already existing virtual reality application. In order to use the module in other applications' domain , only that application domain dependent files that can be written using a simple scripting grammar, are necessary to provide or maintain. The semantic validation algorithm proved to be a good way to handle smoother and more intuitive speech-based interactions.

Even though we have no plans for the moment, we would like to generalize our ideas to use this module in not only Japanese but also English-based interactions.

We plan to make use of expert systems to build a *reasoning* network instead of the semantic one, fusing the semantic network and the validation algorithm in a single process. This to cope with more ever-growing and complex types of interactions.

We are also considering objects' ontologies and machine learning to realize virtual environments interactions more intelligent-like. Eliminating with this the connection of nouns and objects names in the implementation phase.

6 Acknowledgments

This work would not have been possible without the generous support of N. Terashima and the members of ATR CSRL, ATR-I and the CSK group.

References

[1] Takahashi, T., Hakata, A., Kobayashi, Y. and Yamashita, K.(1988). User interface using language and visual information, in *Proc. of Computer-World'88*, Kobe, Japan.

[2] Takahashi, T., Hakata, A., Shima, N. and Kobayashi, Y.(1989). Scene description using spatial relationships derived from visual information, in *SPIE/Advances in Intelligent Robotics Systems*

[3] Tijerino, Yuri A., Mochizuki, Kenji., and Kishino, Fumio. Interactive 3-D Computer Graphics Driven through Verbal Instructions: Previous and Current Activities at ATR. In *Comput. & Graphics.*, volume 18, pp. 621–631. Elsevier Science, Great Britain, 1994.

[4] Bolt, R.A. (1980) Put that there: voice and gesture at the graphics interface, *Computer Graphics*, Vol. 14, No. 2, pp. 262.

[5] Tijerino, Y.A., Abe, S., Miyasato, T. and Kishino, F. (1994) What you say is what you see–Interactive generation, manipulation and modification of 3-D shapes based on verbal descriptions–. To be published in the *AI Review Journal*, Vol. 8, issues 1, 2, 3. Also ATR Technical Report TR-C-0093.

[6] Yoshida, M., Tijerino, Y., Abe, S. and Kishino, F.(1995) A Virtual Space Teleconferencing System that Supports Intuitive Interaction for Creative and Cooperative Work. In *Proc. of 1995 Symposium on Interactive 3D Graphics*, Monterey, CA.

[7] Singer, H. Beppu, T., Nakamura, A., and Sagisaka, Y. (1994). A Modular Speech Recognition System Architecture in 日本音響学会講演論文集 pp. 39-40, Tokyo, Japan. Oct 1994

[8] Sagayama, S., Takami, J., Nagai, A., Singer, H., Yamaguchi, K., Ohkura, K., Kita, K., and Kurematsu, A. ATREUS:a speech recognition frontend for a speech translation system. in *Proc. EuroSpeech* pp. 1287-1290, Berlin, 1993

[9] Wexelblat, A.D. (1994) A feature-based approach to continuous-gesture analysis, Master of Science thesis in Media Arts and Sciences, Massachusetts Institute of Technology, May 1994.