Design and Formulation for Speech Interface Based on Flexible Shortcuts

Teppei Nakano¹, Tomoyuki Kumai¹, Tetsunori Kobayashi¹, Yasushi Ishikawa²

¹Waseda University, Tokyo, Japan
²Mitsubishi Electric Corp. Tokyo, Japan

Abstract

This paper proposes a new speech user interface for command selection called “Flexible Shortcuts.” With this approach, users can select any command using continuous keyword input, a voice input method that uses a series of keywords related to the command. The keywords are defined by a hierarchically structured command set, called the functional structure. Two types of interactions are designed to support user friendliness and effectiveness, namely, interaction for exploration and interaction for the resolution of ambiguity. A probabilistic formulation is also considered for this approach. An experiment is conducted to compare the proposed method with the conventional “command and control” interface in terms of efficiency and usability. Experimental results show that the proposed approach is superior to the conventional approach from both objective and subjective points of view.

Index Terms: speech interface, command selection, Probabilistic formulation, continuous keyword input

1. Introduction

Despite the high expectations demanded of speech interfaces, relatively few applications use such interfaces at present. Until today, speech interface is often applied for a command selection method, such as applications for hand-held devices or cellular devices, car navigation systems, and telephony systems[2][3]. However, there are not many reports that speech interface was really used for in these applications. One of the reasons is that existing speech interfaces cannot bring out the full potential, such as quick input and ease of use. It is said that some critical problems in the interface design cause this situation, such as “What should I say” problem and loss of the primacy over other interfaces[1]. To solve this, we clarify the style of speech interface, and consider the effective method for command selection using speech interface.

Speech user interfaces can be classified into the following two styles: a style that does not impose any grammatical constraints on users’ speech inputs, and a style that force users to obey the preconditions. In the former style, which we call “natural spoken language style”, or “NSL Style”[4], systems are designed so that it can handle huge range of user inputs. This style, however, is still being studied and has some unsolved problems. For instance, the variety of actual user’s expressions is usually more complex than that of the acceptable/designed expressions Besides, speech recognition performance for natural spoken languages in the actual environments is not enough yet. On the other hand, in the latter style, which we call “system oriented language style”, or “SOL Style”, systems are usually designed to handle only possible inputs based on their preconditions. This style is more practical in terms of application developments, and most actual products are developed in this style. In this research, therefore, we focus on this SOL Style.

When developing the speech user interface based on SOL Style, subset language[5] is defined, which defines a vocabulary, grammar, and speech style for the target application. Here, we regard the definition of a subset language as including a very small set of resources for speech recognition systems: such as a set of the command vocabulary for the isolated word recognitions and a set of small context free grammars. Because the set of resources of subset language is smaller than the set of resources for NSL usually, higher speech recognition performance rate is expected. In addition, several studies have shown that this style is not always inferior to NSL Style in terms of both subjective and objective assessments [6].

There are applications in which users execute a command via voice input[2]. In these applications, vocabulary is defined as a set of commands that are mapped directly to certain operations. This approach, called “Command and Control” (C&C), is employed by many applications including commercial applications such as car navigation systems, because of its responsiveness in use and ease of development. However, this approach has a serious problem: novices cannot control the application easily. This is because there are mismatches between the vocabulary prepared by the developers and the vocabulary expected of the users, and therefore, it is often difficult for users to find what they can say[1]. Alternatively, several applications use a menu hierarchy[3]. In this approach, called “Menu-based Interaction,” users select menu items interactively to reach the intended command. The advantage of this approach is that users clearly know what they can say because the available words are listed for each menu selection. However, the disadvantage is that the speed is lost because of the increased interactions.

As a means of resolving these problems, a new speech user interface for command selection was investigated. In this study, a method called “Flexible Shortcuts” is proposed. This approach also uses a hierarchy of commands arranged by their functional structure. However, in our approach, users can execute any commands flexibly by using several keywords related to the function. In addition, by using the interaction for exploration, user-friendliness for novices is secured.

2. Basic approach

The basic goal of this research is to give users the ability to select a command flexibly out of the set of commands by using continuous keyword input. In this approach, multiple keywords and their synonyms are defined and associated to each command. Users can select any commands flexibly by specifying arbitrary series of keywords that can uniquely distinguish a certain command. For instance in Fig.1(Bottom), several keywords (from A to E) and their synonyms (from A’ to E’) are defined, and associated to the commands (from X to Z). In this example, users can select the command X by using several keyword sequences, such as A-B-C and A’-C. This type of voice input method, with which users can specify a series of related...
keywords at a time, is called continuous keyword input. When comparing this approach to the conventional approach, command and control, or C&C (Fig.1(Top)), several advantages are found. Firstly, it is possible to use fine-grained keywords that can be easily associated from a function of a command. Secondly, synonyms can be defined for each keyword, not for each command. Thirdly, the probability of keyword sequence selection can be taken into account to speech recognition systems so that the problem of speech recognition got easier. In order to apply this approach, however, the following two problems need to be considered:

- The decision method for keyword-command relation.
- The behavior when keywords that are not enough in identification are given.

As a solution to these problems, we employed the approach using “functional structure” and “two types of interactions.” The remainder of this section describes the details of the solution.

### 2.1. Functional Structure

In order to define keyword-command relationships, we use the functional structure, a hierarchically structured command set. In this approach, we define a set of keywords for each command from its node name, its parent node names, and their synonyms. An example of the structure is illustrated in Fig.2. In this example, for instance, a command to enlarge the text size in the email application is associated with the following keywords: “Email”, “Thunderbird”, “Mailer”, “View”, “Text Size”, “Font Size”, and “Increase”. Users can select this command by using a series of keywords that can uniquely distinguish this command. Users can select this command by using flexible shortcuts, which consists of these keywords. The followings are the examples of flexible shortcuts for this command.

- “Email” – “View” – “Text Size” – “Increase”
- “Email” – “Increase” – “Text Size”
- “View” – “Increase”
- “Mailer” – “Text Size” – “Bigger”
- “Increase” – “Text Size”
- “Increase” – “Font Size”

### 2.2. Two Types of Interactions

Selecting a command uniquely by using given keywords is not always possible. For instance, in the previous example, it is not possible to select any commands when user input is “View” only. Also, more than one commands can be selected when user input is “View–Normal.” To handle these problems, we extend the target of the flexible shortcuts to the intermediate nodes in the structure, and introduce two types of interactions: an interaction for the exploration and an interaction for the resolution of ambiguity.

The interaction for exploration is used when keywords that cannot uniquely distinguish a certain leaf node but can uniquely distinguish an intermediate node are given. To help users find their intended command, this interaction follows the fundamentals of Menu-based Interaction. That is, the system lists the available items based on the current “context” and a user select an item. The “context” is decided based on the location of the user within the menu hierarchy. The method of listing depends on the available devices, such as graphical feedback and vocal feedback. When the user selects an item, the context is updated. This interaction continues until users reach the leaf. An example is shown in Fig.3. This example shows that the user can also skip a couple of interactions by using continuous keyword input at step 5 in the figure.

The interaction for the resolution of ambiguity is used when keywords that can select more than one node are given. This interaction follows the fundamentals of the interaction for exploration. But this is different in that it lists not child nodes but candidate nodes. For instance, when user input is “View–Normal” in the above example, system lists “Text Size–Normal” and “Toolbar–Normal.” In addition to the ambiguity of node location, this interaction can handle ambiguities of speech recognition results. Ambiguity of the speech recognition results is unavoidable in speech recognition applications. Most applications solve this ambiguity at the input method level by prompting users to select the exact inputs[7]. In the proposed approach, however, both the ambiguity of the speech recognition result and the ambiguity of the intended node can be solved at the same time. When results are ambiguous, systems can list candidate nodes that are selected for each ambiguous result, as well as candidate nodes caused by the node location ambiguities.

By using these interactions together with the continuous keyword input method, this interface is expected to be useful for all levels of user, from novice to expert: First, when the user cannot find a desired function, functions can be searched by simply entering a couple of keywords related to the function. As with the menu-based interaction technique, this makes it possible for novices to execute commands easily. In addition, by using continuous keyword input, the user can execute commands directly without the need for unnecessary interactions. This makes it possible for the user to execute commands as efficiently as by the C&C approach, which is regarded as the most efficient command selection method for speech interface, unless
1. Initialize

3. Speak: "Email"

5. Speak: "View Text Size"

6. Change context to [Email-View-Text Size], and its child nodes: e.g. Increase, Decrease, ...

7. Speak: "Increase"

8. Execute the command:

"Enlarge the text size at the email application"

Figure 3: A sample of an interaction for exploration

the user is a novice. Finally, users can easily learn the keyword sequences needed to execute a certain command directly as a result of everyday operation. This is because shortcuts are defined dynamically by the structure explored by the user. As a result, a smooth transition from novice to expert is possible with this interface.

3. FORMULATION FOR CONTINUOUS KEYWORD INPUT

The proposed interface is characterized by the use of a speech recognition system that uses a functional structure as a linguistic resource to accept continuous keyword input. Using the provided keyword sequence, the system must decide the desired command. In this model, the command estimation problem can be replaced by a path determination problem on the functional structure. Therefore, we formulated this model as a path determination problem using the voice input method.

When a voice input, a current node, and a destination node are defined as $X$, $n_s$, and $n_g$, respectively, (Fig. 4), this problem can be formulated as a problem to find $n^*_g$ that maximize $P(n_g|n_s,X)$.

$$n^*_g = \arg\max_{n_g} P(n_g|n_s,X) \quad (1)$$

Using Bayes theory, (1) can be transformed to (2).

$$n^*_g = \arg\max_{n_g} P(X|n_g,n_s)P(n_g|n_s) \quad (2)$$

We now define a set of word sequences $\Omega = \{W_i\}$, ($W_i$ is a word sequence, $i \in C$, where $C$ is a number of all combinations of defined words) as all available word sequences that can be defined based on $n_s$ and $n_g$. Using $W_i$, (2) can be transformed to (3).

$$n^*_g = \arg\max_{n_g} \sum_{W_i \in \Omega} P(X|W_i)P(W_i|n_g,n_s)P(n_g|n_s) \quad (3)$$

where the first term is the acoustic model, the second term is the language model dependent on the state (current node) and the intention (intended node), and the third term is the state dependent intention model (the model of intended node). Here, we use (4) given by Bayes theory.

$$P(W_i|n_g,n_s) = \frac{P(n_g,W_i,n_s)P(W_i|n_s)}{P(n_g|n_s)} \quad (4)$$

Then (3) can be transformed to (5).

$$n^*_g = \arg\max_{n_g} \sum_{W_i \in \Omega} P(X|W_i)P(W_i|n_s)P(n_g|W_i,n_s) \quad (5)$$

Figure 4: A model of formulation. (Each $n_{0-9}$ is a node of the structure. Each $k_{0-9}$ is a keyword associated with the node. $k^'$ is a synonym of k3. $W_{16}$ denotes a set of keyword sequences between $n_1$ and $n_6$.)

where the first term is the acoustic model, the second term is the state dependent language model, the third is the state procedure (word sequences) dependent intention model. Now, let us consider $\Omega' \subset \Omega$ determined by the first and second terms in (5).

$$\Omega' = \text{nbest} P(X|W_i)P(W_i|n_s) \quad (6)$$

Using $W_i$ given by (6), (5) can be approximated as (7).

$$n^*_g \simeq \arg\max_{n_g} \{ \max_{W_i \in \Omega'} P(X|W_i)P(W_i|n_s)P(n_g|W_i,n_s) \} \quad (7)$$

Here, let $f(n_g,n_s,W_i)$ be a proper function of the third term of (7), and $L(W_i)$ as a likelihood function of $W_i$. Then $n^*_g$ can be calculated approximately using (8).

$$n^*_g \simeq \arg\max_{n_g} \max_{W_i \in \Omega'} L(W_i)f(n_g,n_s,W_i) \quad (8)$$

That is to say, the most likely destination node is approximately given as the node that maximize (9).

$$\text{score}(n_g) = \max_{W_i \in \Omega'} L(W_i)f(n_g,n_s,W_i) \quad (9)$$

In particular, when we use only 1-Best, $n^*_g$ is given by following equations:

$$W^* = \arg\max_{W_i \in \Omega'} P(X|W_i)P(W_i|n_s) \quad (10)$$

$$n^*_g \simeq \arg\max_{n_g} f(n_s,n_g,W^*) \quad (11)$$

4. Experiment: Flexible Shortcuts vs. C&C

In order to evaluate the efficiency and usability of the proposed interface, an experiment is designed to compare two different interfaces, "Flexible Shortcuts" and C&C. An experimental application, "Application Controller", was developed which supported the both interfaces. This application provides both graphical and speech user interfaces so that users can execute commands to control registered applications. The GUI is used to present the internal state, such as the current node and keywords of child-nodes. The user can select nodes by touch pen or speech input. When the user selects a certain leaf node, a function associated with the leaf is executed.

Using a given interface, each subject performs various tasks. A task involves executing a command in response to a question. The task is successfully completed only when the exact correct command is selected. If a subject selects an incorrect command, the subject must "undo" the command by clicking the undo button. There is no time limit. However, some subjects
might not be able to determine the command name. Therefore, the subjects were allowed to give up after 45 seconds. The task completion time is calculated as the time from when the question is shown until the task is completed. This includes both the time to consider command names or keywords and the operation time. Users were encouraged to use speech input, and therefore, they used speech input as a main input method actually. The subjects are required to participate in four sessions: two consecutive sessions using one interface, and two consecutive sessions using the other interface. A total of 20 tasks are prepared for each session. The order of the two interfaces is switched evenly among the group of subjects, but the target applications always follow the same order. During the first two sessions, the subjects perform the tasks for the media player application using a given interface. During the next two sessions, the subjects perform the tasks for the voice chat application using the other interface. After each set of two sessions, the subjects complete a subjective assessment [8].

A total of 20 graduate or undergraduate students from the Engineering Departments participated in this experiment. All of the participants were native Japanese students of 18 to 24 years of age. Students majoring in computer science were allowed to participate in this experiment. The result is shown in Fig. 5. The objective result is shown in Fig. 5(A) as a “Time-Achievement Graph”, a graph that represents an achievement rate of tasks when a time limit was given in x-axis. This result shows that the max achievement rate, which is the achievement rate when the time limit is infinity, increased 20 points in the 1st session and 10 points in the 2nd session. Also, by observing the result for the second session, it is concluded that the experienced users did not lose any efficiency with the proposed speech interface. These results show that using the proposed interface is an effective solution that does not sacrifice efficiency to the critical problem of the C&C interface, which is referred to as the “What should I say?” problem. Fig. 5(B) shows that all of the subjective scores for the proposed interface except Speed were superior to those of the C&C interface. In particular, the scores for Likeability and Annoyance for the proposed interface (5.2 and 4.9, respectively) were far superior to those of the C&C interface. This means that subjects were able to enjoy the interface, which is one of the most important features. The Habitability for the proposed interface was slightly less than neutral (3.9 / 7). This may be one reason why some tasks are so difficult or unfamiliar that the subjects cannot imagine any keywords. In addition, the structure, which is based on the menu hierarchy of the application, might not be well designed. The score for Speed for the proposed interface was approximately the same as that of the C&C interface. This means that the subjects felt that they were able to use the proposed interface as quickly as the C&C interface, which is significant because speed is a key feature of the C&C interface. As a result, the proposed interface is verified to be superior, both objectively and subjectively, to the conventional interface, i.e., the C&C interface.

5. CONCLUSION

This paper presented a new speech user interface called Flexible Shortcuts. This approach is designed to give the user the ability to select any command using continuous keyword input, which is a voice input method that uses a series of keywords related to commands. A basic model and probabilistic formulation for this approach are considered. Using this method, an experiment was conducted to compare the proposed interface with the conventional approach, C&C. The experimental result shows that the proposed approach achieved high usability for novices without losing the efficiency for experts.

6. Acknowledgements

This work was supported by the METI: “Development of Fundamental Speech Recognition Technology,” and by the project research at Waseda University: “Basic Research on Speech Recognition.”

7. References