A Multimodal System for Accessing Driving Directions

Ashutosh Morde[1], Ramanujan S. Kashi[2], Michael K. Brown[3], Deborah Grove[1] and James L. Flanagan[1]

[1] CAIP, Rutgers – The State University of New Jersey, Piscataway, NJ 08855
{amorde,dgrove,jlf}@caip.rutgers.edu
[2] Avaya Research Labs, Basking Ridge, NJ 07920
ramanuja@avaya.com
[3] Cydyreal Inc, North Plainfield, NJ
mkb@cydyreal.com

Abstract. The focus of this paper is describing a system that repurposes a web document by a spoken language interface to provide both visual and audio driving directions. The spoken dialog interface is used to obtain the source and destination addresses from a user. The web document retrieved by querying the user on the addresses is parsed to extract the maps and the associated text. Further the system automatically generates two sets of web documents. One of these sets is used to render the maps on a hand-held device and the other set is used for the spoken dialog interface through a traditional phone. The system's user interface allows navigation both through speech and pen stylus input. The system is built on a PhoneBrowser architecture that allows the user to browse the web by speech control over an ordinary telephone.

1 Introduction

As the popularity of the web has increased over the past few years, so has the amount of information available. The conventional method of accessing such information is using desktop web browsers. Access to online information is crucial for professionals on the move. Telephone penetration is still greater than that of laptops. Telephone access to on-line data using touch – tone interfaces is already common. These interfaces have a hierarchical structure; they are essentially menu driven. The user is provided a list of options and he is required to select a particular option from these. A long list of options taxes the users memory while a deep menu structure can frustrate the user. Speech, on the other hand, provides a natural way of capturing spontaneous thoughts and ideas. Voice as an access mechanism allows additional tasks to be performed while the hands or eyes are busy [1].

This suggests the use of speech or spoken dialog to access online information as an alternative to conventional desktop browsers. One of the commonly accessed information is that of driving directions. The user either tries to memorize the driving steps or take a copy of the directions. Obtaining driving directions and help in navigation is often required on the road where access to a desktop computer is not available. So it becomes necessary to have a system that can both query the user using a spoken dia-

D. Lopresti, J. Hu, and R. Kashi (Eds.): DAS 2002, LNCS 2423, pp. 556-567, 2002.

log interface and render driving directions through mobile/conventional phones and hand-held devices.

The application developed at the Center for Advanced Information Processing (CAIP)/Avaya allows the user to access driving directions through a regular telephone connection. Visual feedback is provided to the user by displaying the driving maps and the directions onto the users wireless handheld device. In particular Compaq's iPAQ was used for the system development. The speech interface was provided by PhoneBrowser [2].

2 Need for a Multimodal Interface

In a visual environment, the users can immediately see the information available to them; it persists on the screen as long as the user desires. The users have plenty of time to analyze the information available at a given time before they need to make a decision. In a nonvisual environment, however, a system must list the information serially. Audio is slow for the listener to consume large amounts of speech and the user must rely upon short-term memory since audio does not persist like a visual display [3]. The inherent lack of visual feedback in a speech-only interface can lead users to feel less in control and makes it much harder for the user to absorb spoken information [4].

In the current system these shortcomings associated with a speech-only interface were mitigated by providing the user with an additional visual modality of providing driving directions and maps on a wireless handheld device. This visual display of information gives the user more time to assimilate the data and aids navigation. In addition to using a speech interface to move between driving steps the user also gets the flexibility to navigate using the stylus input of the handheld device.

3 Application for Driving Directions

Figure 1 shows the driving directions application setup. The user calls up the PhoneBrowser using a regular telephone connection. The exchange of information between the PhoneBrowser and the web server hosting the driving direction application is through the regular HTTP protocol over the web.

The user calls up the PhoneBrowser number and logs into the Rutgers Voice portal by saying the login ID "Rutgers Database Search". Once the user logs into the system he is guided in building the origin and destination street addresses with verification dialog turns to remove any misrecognition error.

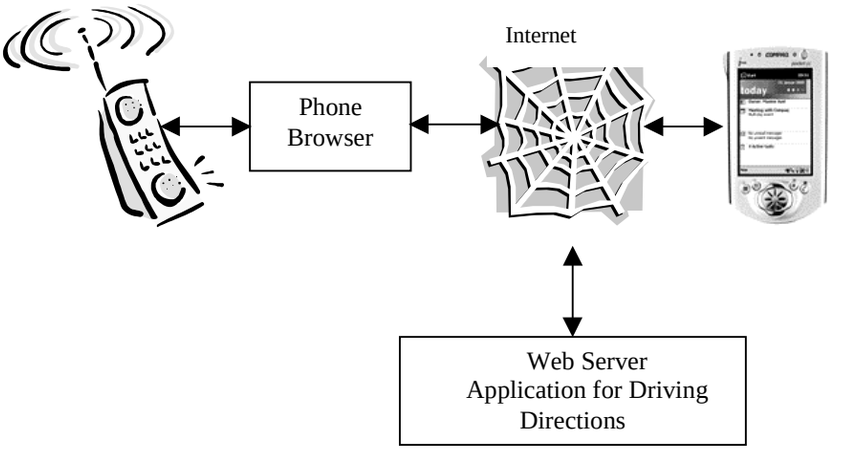

Fig. 1. Application setup

Figure 2 shows the flowchart for the speech only interface aspect of the application. As the address construction process is similar for both the origin and the destination, its flow is shown separately in Figure 3. This figure represents the two modules of “Origin Address” and “Destination Address” in the main flowchart.

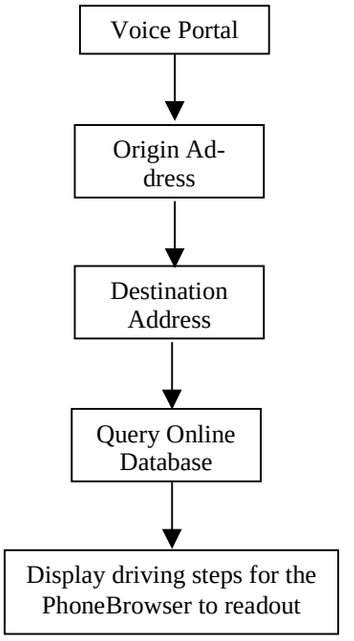

Fig. 2. Application flowchart with telephone interface only

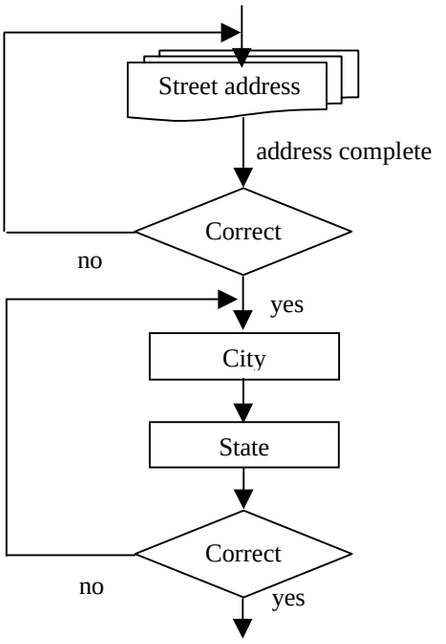

Fig. 3. Address construction

To provide a visual interface web pages are built dynamically, the content being dependent on the driving step requested by the user. Figure 4 indicates the modification required to the application flow for a visual interface. The user has a wireless handheld device connected to the web. The web server sends information to the server running on the handheld device using sockets. Figure 5 shows the user interaction once the online database has been queried and the PhoneBrowser reads out the driving directions. The system can be roughly divided into 4 tasks – dialog for user interaction, information retrieval, information presentation and bootstrapping the grammar.

3.1 Dialog

The user-system interaction consists of dialog for accessing the origin address, which is broken up into following components.

- dialog turns for origin street address, origin city and origin state
- similarly, there is dialog for accessing the destination address from the user
- dialog turns for verifying that the system has understood the user correctly
- dialog for providing the user with context sensitive help
- dialog for the case of no recognition of user input by PhoneBrowser
- dialog to control the way information is provided to the user

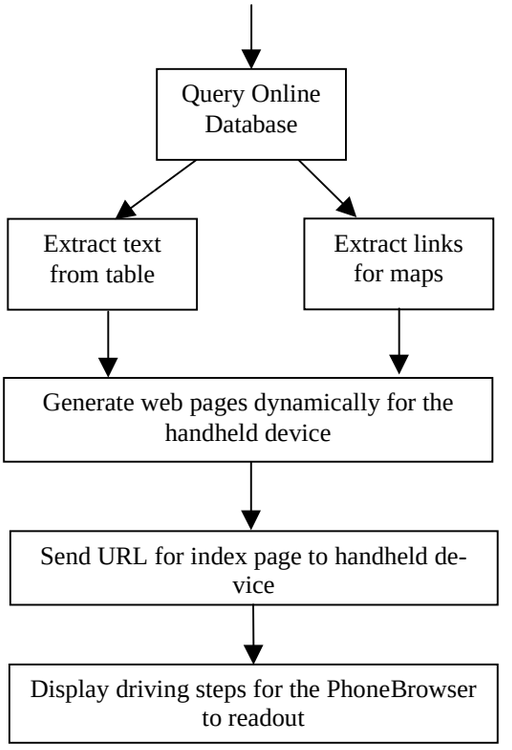

Fig. 4. Modification required to application flow for visual interface

If the users are to be given flexibility as to what they can speak, the appropriate grammar has to be designed for the system. The speech interface is achieved using the PhoneBrowser, which is a tool that allows a user to access web information through a telephone line. The system grammar is embedded in web pages, which are read by the PhoneBrowser. The user input as recognized by the PhoneBrowser is passed onto the system cgi-script, which then takes appropriate action based on user utterances.

The general syntax for embedding the grammar in web pages is

```
<a> href  = “URL” special_tags > URL title </a>
```

where the special tag is rule-based grammar written in the Java Speech Grammar Format (JSGF) and URL is the URL to the cgi-script that acts on the user utterances or just a static web page to be read out to the user.

An example of the grammar is

```
<a> href  = “URL” JSGF=([can you|could you](go to
the)(previous|next|last)(step))
> URL title </a>
```

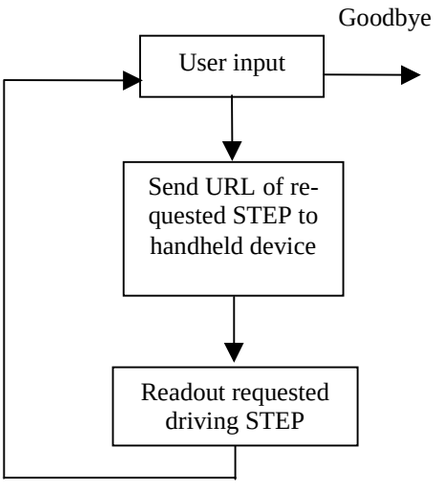

Fig. 5. Driving direction readout control

The finite state representation of this grammar is shown in figure 6. The user can go to the web page pointed to by the URL by saying, “Can you go to the last step”, “go to the next step” or any other phrase or sentence defined by the finite state grammar.

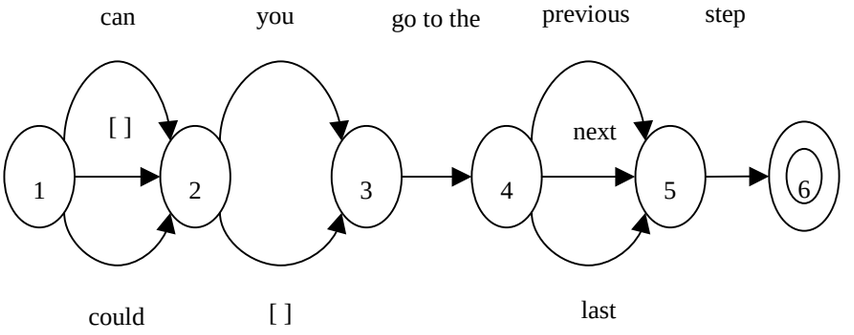

Fig. 6. Finite state grammar

The processing of user input is achieved using the URLINSERT tag with the URL pointing to a cgi-script. Its syntax is

```
<a>href =”http://host/path?userip=%s” URLINSERT
JSGF=((next|previous)(step)) > title </a>
```

When PhoneBrowser sees the URLINSERT tag, instead of just going to the URL location, it replaces the %s with its interpretation of user input. Further details of the various PhoneBrowser tags can be found in [5].

It is tempting to provide the user greater flexibility in building his sentences by increasing the grammar supported by the application. This generally results in an increase in misrecognition as similar sounding words (like eight and ate) are introduced in the grammar, not to mention increased search times. So there are tradeoffs involved in the grammar size and recognition accuracy. For the application at hand, for large databases, restrictive grammar size can be achieved by querying the user for the state. Then depending upon the state the grammar for cities is restricted and finally the user is provided with the street names grammar for the city he is interested in.

When the users log into the system they are given a very brief overview of the system. The system at this stage could have very well asked the user to provide the entire origin and destination street address explicitly or its grammar could have been made complex enough such that the system could have understood a sentence like "I want to go from 96 Frelinghuysen road in Piscataway, New Jersey to 100 Commercial Avenue in New Brunswick, New Jersey" or "I am at 96 Frelinghuysen Road in Piscataway, New Jersey and would like to go to 100 Commercial Avenue in New Brunswick, New Jersey".

The problem with such an approach is that there are no delimiters for street address, city and state as the address can be anything – the user may want to go to Old New Brunswick Road in New Brunswick. So in spite of recognizing what the user has said the system can't process the information provided to it into a meaningful form. As a result a guided approach was chosen in which the entire process is broken up into smaller dialog turns with each turn acting as delimiter for an address component.

Moreover special consideration is required for the street address as the user may want to say just Biel road or 54 East 32 North. As there is no standard format for street address an additional delimiter "address complete" was added to signify completion of the street address building process. Also street numbers could be any number in the range 1 to 9999. Instead of having all the numbers in grammar the number itself was built using individual digits. The street address could now contain any combination of up to 4 digit numbers and street names. On being prompted to begin building the street address the user may say "Five Four (wait for an acknowledgement) East (wait for an acknowledgement) Three Two (wait for an acknowledgement) West (wait for an acknowledgement) address complete". Though this may seem as a very elaborate procedure for building a street address it gives the user the flexibility to build any street address. Dialog turns for verifying the correctness of the recognized information were introduced to remove any misrecognition errors. A sample interaction between the user and the system for an origin address is shown in Figure 7.

A confirmation for each component of the street address is given by the system as it can misrecognize individual digits. If there is an error the user can say "address complete" and then rectify the error. The user can ask for help at any stage by saying "direction help". The user is then provided with help for the pertinent dialog turn. If the user feels that he needs the help for entire application he can say "full help" or he can say, "return to application" and return to the dialog turn through which he initiated help with all his state information intact. The context sensitive help is initiated by the system if the user does not respond for a time period of 20 seconds.

…
System: Please start dictating the origin street address. Say address complete when you are finished dictating.
User: nine six
System: Ninety-six
User: Frelinghuysen Road
System: 96 Frelinghuysen Road
User: address complete
System: You want to start from 96 Frelinghuysen Road. Is it correct?
User: yes
System: Please tell me the name of the city
User: Piscataway
System: Street is 96 Frelinghuysen Road. You are in Piscataway. Now tell me the name of the state.
User: New Jersey
System: Street address is 96 Frelinghuysen Road. You want to start from Piscataway in New Jersey. If this is correct say start?
User: Start
System: Please start dictating the destination street address. Say address complete when you are finished dictating
…

Fig. 7. Usage scenario

3.2 Information Retrieval

Once the origin and destination address have been verified the system queries the Avis Rental Company site (http://www.avis.com). The result of the query is a regular web document with maps and table of direction steps. This information can be directly provided to the PhoneBrowser, it will ignore the images and start reading out the directions. But there will be no control as to how the information is presented to the user. So after the system grabs the results page from the web it parses the html document to extract the driving steps text from the tables. This text is formatted to introduce longer pauses between steps. A special control character, which disables the spelling of capitalized words by the PhoneBrowser, is inserted at the beginning of each step. This is required as the street names and left and right turns are capitalized. The formatted text is then presented to the PhoneBrowser. The steps are also stored as state information so that they can be used later to present the information to the user in the format he desires. At this step grammar for context sensitive help is also provided to the PhoneBrowser.

3.3 Information Presentation – The Wireless Handheld Device

In an earlier version of this project all the information was presented to the user through the voice interface only. Usability experiments showed that the users had

trouble memorizing the driving steps [6]. These usability experiments guided us to add the visual modality – the turn-by-turn maps and the driving steps were rendered onto a wireless handheld device.

The spoken dialog interface obtains the origin and destination addresses from the use and is used to retrieve an appropriate web document. The driving steps and the URL's to the turn-by-turn maps was obtained by parsing this web document. The extracted driving steps were used in the creation of web pages dynamically on the fly. This approach avoids the download of images from the web onto the web server. The content of these web pages depends upon the user input and the driving step content. Figure 8 shows a typical driving direction web page as viewed on a handheld device.

Our implementation includes a server running on the wireless handheld device to synchronize the information displayed with that requested by the user over the phone. It is implemented using two threads – the dialog thread and the worker thread. The dialog thread is responsible for monitoring the user stylus input for starting and stopping the server while the worker thread is responsible for monitoring the socket for URL information from the web server.

The user starts the web server by clicking on the start button of the dialog. This causes the worker thread to be created, which initializes the socket and starts listening for input from the web server. When the PhoneBrowser starts reading out the driving directions the web server sends the URL, of the index page of the set of web pages created for the user, to the server running on the wireless handheld device. The server then launches Internet Explorer. Whenever the user requests to go to any specific step,

Fig. 8. Driving directions displayed on a handheld device

in addition to reading out the step through PhoneBrowser, the web server sends the URL for that step to the server running on the wireless handheld device. This URL is passed on to the launched pocket Internet Explorer. A sample interaction between the user and the application while the driving directions are being read out is as shown in Figure 9.

The user can jump to the web page of any step either through spoken input to the telephone or using the stylus input on the hand-held. The grammar available to the user is

```
[can | could][you][repeat | go to](step (0 | (1 | 2 | 3
| 4 | 5 | 6 | 7 | 8 | 9) [(0 | (1 | 2 | 3 | 4 | 5 | 6 |
7 | 8 | 9))])|(next | previous | last) step).
```

So the user can say go to step 3, repeat step 3, go to next step, next step and so on. The other set of grammar available to the user is for finding the total travel time and distance. The user can ask the system for the total travel time or the total travel distance. The user can stop the web server by clicking on the STOP button of the dialog. This causes the worker thread to be terminated and the sockets to be cleaned up. All the web pages visited by the pocket Internet Explorer are stored in a temporary cache on the wireless handheld device. By doing so these pages can be retrieved and viewed later if so desired.

……
User: Go to next step
System: Step 2. Turn RIGHT onto TUNISON ROAD. 0.6 miles
(Send URL for step 2 to Handheld device)
User: Go to step 5
System: I think you asked me to go to step 51 but there are only 7 steps
User: Go to step 5
System: Step 5. Merge onto NJ-18 north. 0.1 miles
(Send URL for step 5 to Handheld device)
User: Repeat step 4
System: Take the exit on the left towards US-1 south/NEW BRUNSWICK. 0.2 miles
(Send URL for step 4 to Handheld device)
User: Jump to last step
……

Fig. 9. Read out control usage scenario

3.4 Bootstrapped Grammar

It may not always be possible to provide all possible street names in the application grammar. In such an instance the application can start off with a few street names in the grammar and then build up its own grammar.

While the HTML document is being parsed to extract direction steps the extracted text is also analyzed for street names. Most of the web sites providing driving directions capitalize the street names and the left or right turns to be taken. The automatic grammar update algorithm first replaces all street name abbreviations with their full forms, e.g., dr by drive and pl by place. It then chooses a sequence of capitalized words ending with one of the predefined street name endings as its initial guess. This eliminates the selection of words for turns to be included in the grammar. The algorithm then determines whether the particular sequence of selected words lie in a single sentence. This check is necessary as it is quite possible that a step ends with a street name while the immediate next step begins with another street name. Requiring that the words belong to a single sentence eliminates the inclusion of such concatenated street names into the grammar. Once it has been decided that the selected sequence of words is a valid street name it is compared with the existing street names in the grammar to avoid duplication and only then inserted in the grammar. The next time the grammar is used; it can be in the same session, e.g., if the user wants a second set of directions, the user can say any new street name that has just been added to the grammar. Thus the application grammar is bootstrapped.

4 Summary

A multimodal system for accessing driving directions, whose user interface allows navigation both through speech and pen stylus input has been implemented. The system is built on a PhoneBrowser architecture that allows the user to browse the web, by speech control over an ordinary telephone. The user-system interaction was tested only for the speech interface. The users showed a marked preference for jumping between driving steps. The usability experiments also confirmed that the user finds it difficult to memorize driving directions without any visual aid.

The future directions for this application involve enhancing the grammar for a more natural user interface and making the call through the users handheld device instead of a telephone. Also to minimize the errors resulting from an increased street name grammar size the application flow can be reversed. The user can be asked to provide the information for the state first followed by city and then the street. This will restrict the amount of grammar loaded onto the PhoneBrowser thereby minimizing the chances of recognition error. Usability studies for the application with both the speech and visual interface need to be conducted to study the user-system interaction.

References

[1] Martin, G.L. “The utility of speech input in user-computer interfaces.” *International Journal of Man-Machine Studies*, 30:355–375, 1989

[2] Brown, Michael K., Stephen C. Glinski, Bernard P. Goldman, and Brian C. Schmult. "PhoneBrowser: A Web-Content-Programmable Speech Processing Platform." *Position Paper for The W3C Workshop on Voice Browsers*, Cambridge, Massachusetts, October 1998

[3] C. Schmandt. “Multimedia Nomadic Services on Today's Hardware.” *IEEE Network*, September/October 1994.
[4] Stifelman, Lisa, Barry Arons, Chris Schmandt, and Eric Hulteen, “VoiceNotes: A Speech Interface for a Hand-Held Voice Notetaker.” *ACM INTERCHI ‘93 Conference Proceedings*, Amsterdam, The Nether-lands, April 24–29, 1993
[5] Brown, Michael K. “PhoneBrowser User’s Guide.” *Avaya Communication Inc.* October 2000
[6] Morde, Ashutosh. “An Application for Voice Controlled Driving Directions”, *Master of Science Thesis, Rutgers – The State University of New Jersey*, May 2002.