A Voice-Driven Web Browser for Blind People

Simon Dobrišek, Jerneja Gros, Boštjan Vesnicer, France Mihelič, and Nikola Pavešić

University of Ljubljana, Faculty of Electrical Engineering,
Laboratory of Artificial Perception, Systems and Cybernetics,
Ljubljana, Slovenia,
simond@fe.uni-lj.si,
WWW home page: http://luks.fe.uni-lj.si/~simond

Abstract. A specialised small Web browser with a voice-driven dialogue manager and a text-to-speech screen reader is presented. The Web browser was built from the GTK Web browser Dillo, which is a free software project in the terms of the GNU general public license. The new built-in screen reader is now triggered by pointing the mouse and uses the text-to-speech module for its output. A dialogue module together with a spoken-command input was also introduced into the browser. It can be used for navigation through a structure of common Web pages. The developed browser is primarily intended to be used with the new Web portal, exclusively dedicated to blind and visually impaired users. All the Web pages at the portal or at sites that are linked from this portal are expected to be arranged as common HTML/XML pages, which complies with the basic recommendations set by the Web Access Initiative.

1 Introduction

Modern information-technology facilities are often not suitable for blind and visually impaired people. Such problems in communication are well known to many disabled persons. If they are unable to use their hands, read or speak, they are forced to use technical aids to overcome their problems. For blind or visually impaired persons the Braille coding of texts is a common aid. This type of coding requires special editions of written corpora or special additional hardware components when used with computers. The solution is relatively costly and requires special skills from the user.

Over the past ten years a considerable advance has been made in the development of automatic text-to-speech and speech recognition systems. Such systems offer a more natural and user-friendly way of communication for the blind or visually impaired persons; the communication goal can be achieved faster and they offer access to large text corpora via modern technical equipment (over computer networks, scanners, etc.) and have a relatively low price [11].

However, these new technologies are very language dependent and general solutions for all languages cannot be applied directly [8]. If speech technologies are to be used with the Slovene language the language-dependent parts of the

systems must be developed for this purpose using knowledge of Slovene-language phonology, syntax and semantics.

Spoken-language technologies have been one of our main research activities for more than twenty years. Our prime interest is to develop a core technology for the Slovene spoken language that could be customised for different kinds of applications. We found the development of a voice-driven Web browser for Slovene-speaking blind people important to our research for several technical and non-technical reasons, among them is the possibility to help the disabled people.

A voice-driven Web browser called Homer was developed for reading Slovenian texts obtained from the Internet site of the Association of Slovenian Blind and Visually Impaired Persons Societies. The Homer Web browser demonstrates how the ways of accessing daily news and other useful information can be improved for disabled users.

1.1 The Kalliope Web Portal

The World Wide Web server called Kalliope has a long-term ambition to become a specialised Web portal for blind and visually impaired people in Slovenia. The Kalliope Web portal is planned to retrieve the Electronic Information System[4] (EIS) of the Association of Slovenian Blind and Visually Impaired Persons Societies (ZDSSS). All the Web pages at Kalliope will comply with the basic recommendations set by the Web Access Initiative [10] and will be tagged with a few additional XML tags, which will enable user-friendly navigation using the presented dialogue module. The portal will also serve as a site that links to other Web sites in Slovenia that are important to the blind and visually impaired community and are accessible via the presented Homer Web browser. The portal will have its access restricted to ZDSSS members since many texts from the EIS database fall under copyright restrictions.

Our first task was to reformat the EIS text corpora and to transfer them to the new portal. The majority of the text files at the EIS database are stored in a plain, non-tagged text format and so a special HTML/XML tagger is needed to convert these texts into a structure of common HTML/XML pages. Virtually all of the available text files at the EIS require a unique tagger function for this conversion as the texts are provided from different sources. Presently, scripting programs for such conversions are being developed.

Initially, we concentrated on Slovenian daily newspapers which are probably the most interesting and the most frequently accessed texts at the EIS. The scripting programs automatically retrieve the original compressed newspaper-text archives from the EIS. These programs are symbolically shown in Figure 1 as a retriever module. A tagger function, specially designed for each of the newspapers, then forms the structure of the HTML/XML pages. The HTML/XML structure is formed and refreshed at the Kalliope server every few hours. The first page contains links to issues for all weekdays and a link to the most recent issue. The sub-pages contain links to the newspaper heading pages with links to the individual article pages. All the pages contain hidden XML tags, which are

required for the dialogue module to make a distinction between different parts of Web pages.

2 The Homer Web Browser

The Homer Web browser was not built from scratch. New modules were just introduced into the source code of one of the publicly available Web browsers, which is now considered to be yet another module of the whole Homer system.

When seeking the appropriate Web browser we considered the following criteria:

- The source code has to be written entirely in C.
- It has to be a multiplatform browser.
- It has to be small, stable, developer-friendly, usable, fast, and extensible.
- It has to be a free software project in the terms of the GNU general public license.

We found that the GTK Web browser Dillo [1] was perfect for our needs.

2.1 The Homer system structure

The whole system consists of four main modules. The system structure is shown in Figure 1

Input to the system is performed via a keyboard, with some specially selected keys or by using the speaker-independent spoken command recognition module that runs in parallel with the other modules. The voice control of the system additionally facilitates working with the system as there is no need to use mechanical interfaces. The dialogue module manages dialogues with users and performs access to the Web pages via the Web browser.

The most important part of the system is the first fully developed Slovenian text-to-speech system [5], which is essential for blind users. It enables automatic generation of Slovene speech from an arbitrary Slovenian text written in a non-tagged plain-text format using one of the standard character encodings, like the Slovenian version of the 7–bit ASCII coding or the WIN-1250 and the ISO-8859-2 codings.

The Homer browser was designed to run on a standard PC with a minimum of 64 MB of RAM with a built-in standard 16–bit sound card and a standard headset with a close-talking microphone. Initially it was developed for Linux platform and later ported to the Microsoft Windows 9x/ME/NT/2000/XP operating systems. For the best performance it uses multi-threading and other advantages of the 32–bit environment. It requires approximately 15 MB of disk space for the program code and for the text-to-speech and speech recogniser module inventory.

2.2 Screen reader Module

Our first step was to add a screen-reader function to the existing Dillo source code. The built-in screen reader is now triggered by pointing the mouse and uses the text-to-speech module for its output. When a user stays for a moment at a certain position on the Web page the text beneath the pointer is sent to the output text-to-speech module. The output module works in a separated thread with a time-out function that prevents the user from overfilling the synthesis buffer with a fast pointer motion when browsing through the Web page.

An important feature of the screen reader is that it generates special distinctive non-speech sounds which inform users about changes to the current positions of the mouse pointer. Such sounds are generated when users leave or enter a particular area of the displayed Web page.

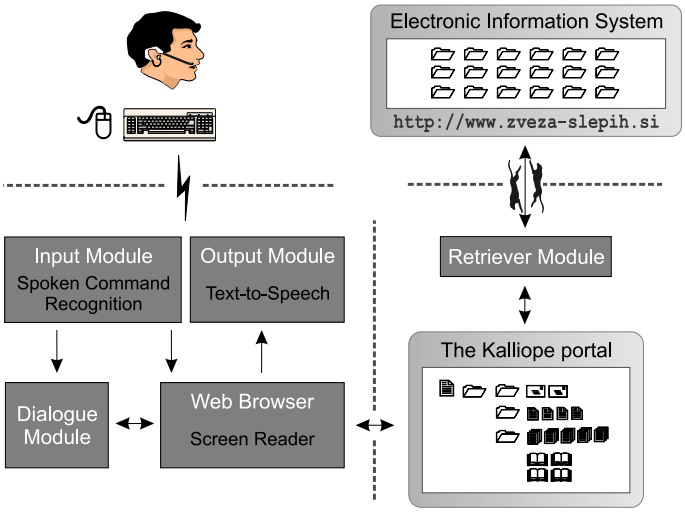

Fig. 1. The structure of the Homer Web browser

The screen-reader function supports not only text parts of common Web pages but some basic graphic objects as well, such as non-animated images, lines, bullets, buttons and input text fields. When a user stays for a moment with a mouse pointer pointing at such graphic objects then the system sends a short description of the object to the text-to-speech module. An example of such a description would be: "*Button labelled 'send', sized 60x20 pixels.*".

The screen reader works in several different modes. It can read individual words, sentences, lines and paragraphs of the displayed Web page. It can read page headings and the whole page as well. The reading mode can be changed by using the function keys on a standard PC keyboard.

2.3 Input Module

The input interface manages the keyboard entry and/or spoken-command recognition. Each of the spoken control commands is associated with its accelerator key on a standard PC keyboard.

The speaker-independent spoken commands recognition module is based on tied-mixture continuous HMMs of fundamental phone transition-like units [3]. These models are used as the fundamental models in a silence / commands / silence HMM graph. A number of improvements to the acoustic modelling were introduced. Variable HMMs structures were implemented and a unique initialisation of the model parameters using a Slovene diphone speech database was used. The parameters of the fundamental models were estimated from the Gopolis spoken-language database [2], which contains several hours of speech from 50 speakers. Using this large database the fundamental HMMs were made speaker-independent.

The whole speech recognition module is designed in an open manner, enabling fast adaptation to different applications with isolated spoken-command recognition input, and also for larger vocabularies of up to several hundred words. The recognition procedure also offers the unrecognised spoken-command category classification, which activates a request for repetition of the command. The current version of the speech recognition module allows use of a spoken command grammar which is translated into an HMM graph.

A preliminary off-line evaluation of the spoken-command recognition accuracy, using a clean speech database of ten test speakers, yielded an average recognition error rate lower than 2%. However, the actual recognition rate is strongly dependent on the spoken-command grammar and the user's behaviour while interacting with the system. In practise, the online recognition error rate increases, but remains below 5%.

2.4 Dialogue Module

The dialogue module manages dialogues with users, accesses Web pages via the Web browser module and performs the system-control function. Its design is based on our experiences with the design of a similar dialogue module in another speech recognition system [7].

Since a structure of common Web pages can always be presented as a tree structure, the dialogue module enables transitions between all the tree nodes at any stage of the processing and navigating through a list of sub-nodes at each of the tree nodes as well. There are only three main actions that the dialogue module takes or offers to user. These actions are: opening of a Web page tree node, navigating through a list of sub-nodes, and closing a Web page tree node. Each of the tree nodes represents a link, a page itself, or a part of a page.

All the actions can either be just offered to the user or are taken immediately after the dialogue module successfully interprets a recognised spoken command or a keyboard entry, even though the dialogue is still in the process of

describing the current position in the Web page tree or is in the process of reading/describing a part of the Web page. As a result, these two different dialogue strategies were implemented. When an action is explicitly offered then a *yes/no* answer is expected from the user. This dialogue strategy is called *passive-user behaviour*. On the other hand, an *active-user behaviour* strategy means that the dialogue module expects the user to interrupt the dialogue process with spoken commands.

The first strategy is more suitable for beginners; the second is more suitable for expert users since it enables faster navigation. Both strategies have to be combined when the dialogue is in the process of reading a selected text. In this case the user is allowed to occasionally interrupt the reading process with commands. As an example a typical *passive-user* dialogue with the system would be something like:

system: "*Welcome to the Kalliope Web portal!*"
system: "*Would you like to open the ZDSSS messages ...*"
user: "*Skip!*"
system: "*Would you like to open the daily newspaper section?*"
user: "*Yes.*"
system: "*Would you like to open The Independent?*"
user: "*Yes.*"
system: "*Would you like to open the News ...*"
user: "*Next!*"
system: "*Would you like to open the Sports page?*"
user: "*Yes.*"
system: "*Would you like to read the article: 'Mika won again'?*"
user: "*Yes!*"
system: "*The selected article has 456 word. The subtitle is ...*"
user: "*Skip!*"
system: "*Mika had a brilliant race and succeeded to burst ...*"
user: "*Close!*"
system: "*Would you like to read the article: 'Keep the gold coming'?*"
... "

The dominantly *passive* dialogue strategy requires only *yes/no* answers and a small number of easy-to-remember navigation commands. We found that this dialogue strategy provides a very comfortable interaction with the system, even though it has proved to be rather time consuming and even annoying to expert users.

The *active* dialogue strategy requires some additional navigation commands. In the best case we could use a dynamic spoken-command grammar instead of using just a static list of commands. The current version of the speech recognition input module did not allow us to use the dynamic spoken-command grammar, thus we carefully selected a small number of navigation commands. We found that the navigation speed is very comfortable when using eight basic commands. Their English translations are: "*Open!*", "*Close!*", "*Skip!*", "*Previous!*", "*Repeat!*", "*Restart!*", "*Pause!*", and "*Resume!*". With these commands a user can open and close Web page tree nodes and navigate through a list of sub-nodes at any position in the tree.

A typical *active-user* dialogue would be now something like:

system: "*Welcome to the Kalliope Web portal!*"
system: "*I have the ZDSSS messages – daily newspapers – ...*"
user: "*Open!*"
system: "*I have The Daily Star – The Independent – ...*"
user: "*Open!*"
system: "*I have the News – the Sports – ...*"
user: "*Open!*"
system: "*I have the articles: 'Mika won again' – ..*"
user: "*Open!*"
system: "*The selected article has 456 word. The subtitle is ...*"
user: "*Skip!*"
system: "*Mika had a brilliant race and succeeded to burst ...*"
user: "*Repeat!*"
system: "*Mika had a brilliant race and succeeded to burst ...*"
user: "*Close!*"
system: "*I have the articles: 'Keep the gold coming' – 'Hughes Takes Gold' – ...*"
user: "*Restart!*"
system: "*I have the articles: 'Mika won again' – 'Keep the gold coming' – ...*"
user: "*Open!*"
... "

Please note that the newspaper titles in the above examples do not really exist in the EIS database. The Slovenian newspapers are *Delo*, *Dnevnik*, *Večer*, etc.

The current version of the dialogue module supports only the two described dialogue strategies. The presented list of basic navigation commands was extended with the names of the most frequently accessed sections at the Kalliope portal. Thus, the navigation commands: "Open daily newspapers!", "Open The Daily Star!", and similar, are now supported. By using of these commands the speed of navigation is increased even more.

2.5 Output Module

For the automatic conversion of the output text into its spoken form the first Slovenian text-to-speech system called **S5** [5] based on diphone concatenation was applied. The non-tagged plain text is transformed into its spoken equivalent by several modules. A grapheme-to-allophone module produces strings of phonetic symbols based on information in the written text. A prosodic generator assigns pitch and duration values to individual phones. The final speech synthesis is based on diphone concatenation using TD-PSOLA [9].

3 Conclusions and Future Work

The development of the Homer system and the Kalliope portal is still in progress. We expect the system to evolve towards a specialised Web browser with a mouse-driven Text-to-speech screen reader and a voice-driven dialogue manager that

handles all the Web pages arranged at the Kalliope portal or at sites that are linked from this portal.

Improvements in the sense of more accurate and robust speech recognition and a user-friendly system to control high-quality speech synthesis are planned for the future. Work on speech recognition that incorporates a larger dynamic spoken-command grammar is already under way. We are expecting further suggestions from the blind and visually impaired community, especially with regards to the design of the strategy for communication with the system and, of course, remarks on the Slovene speech synthesis quality. Many measurements and research in the field of micro and macro prosody modelling of Slovene speech should be done as well as recordings of new diphone databases with different speakers.

References

1. The Dillo Web browser. (2002). `http://dillo.sourceforge.net/`.
2. Dobrišek, S., Gros, J., Mihelič, F., Pavešić, N. (1998). 'Recording and Labelling of the GOPOLIS Slovenian Speech Database'. *Proceedings ot the First International Conference on Language Resources and Evaluation.* Granada, Spain. **2**: pp. 1089–1096.
3. Dobrišek, S. (2001). *Analysis and Recognition of Phones in Speech Signal.* Ph.D. Thesis (in Slovene), University of Ljubljana.
4. EIS - the Electronic Information System. (2002). `http://www.zveza.slepih.si/zdsss/eis/`.
5. Gros, J., Pavešić, N. and Mihelič, F. (1997). 'Text-to-speech synthesis: A complete system for the Slovenian language'. *Journal of Computing and Information Technology.* **CIT-5, 1**:11–19.
6. Huang, X.D., Ariki, Y. and Jack, M.A. (1990). *Hidden Markov Models for Speech Recognition.* Edinburg Information Technology Series. Redwood Press Limited, London.
7. Ipšić, I., Mihelič, F., Dobrišek, S., Gros, J. and Pavešć, N. (1995). 'Overview of the Spoken Queries in European Languages Project: The Slovenian Spoken Dialog System'. *Proceedings of the Scientific Conference on Artificial Intelligence in Industry.* High Tatras, Slovakia. pp. 431–438.
8. Jelinek, F. (1998). *Statistical Methods for Speech Recognition.* The MIT Press. Cambridge, Massachusetts.
9. Moulines, E. and Charpentier F. (1990). 'Pitch-Synchronous Waveform Processing Techniques for Text-to-Speech Synthesis Using Diphones'. *Proceedings of the National Academy of Sciences of the United States of America.* **92. 22**:, pp. 9999–10006.
10. WAI - Web Access Initiative. (2002). `www.w3.org/TR/WAI-WEBCONTENT`.
11. Zajicek, M., Powell, C. and Reeves, C. (1999). 'Ergonomic factors for a speaking computer interface'. In M. A. Hanson, E. J. Lovesey and S. A. Robertson (Eds.), *Contemporary Ergonomics - The proceedings of the 50th Ergonomics Society Conference, Leicester University.* Taylor and Francis, London, pp. 484–488.