SpeechWear: A mobile speech system

Alexander I. Rudnicky, Stephen D. Reed, Eric H. Thayer
School of Computer Science, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213-3890 USA
Telephone: +1 412 268 2622 *email:* air@cs.cmu.edu

ABSTRACT

We describe a system that allows ambulating users to perform data entry and retrieval using a speech interface to a wearable computer. The interface is a speech-enabled Web browser that allows the user to access both locally stored documents as well as remote ones through a wireless link.

1. INTRODUCTION

The perceived utility of speech systems relies in part on the success with which they compete with more established computer interfaces. With the exception of certain tasks (such as dictation), speech interfaces have not made significant inroads in the desktop domain; on the other hand telephone-based applications are becoming established, as speech provides an effective high-bandwidth channel between human and computer. An emerging and possibly even more important domain is that of "wearable" systems consisting of small computers that can be easily carried on the person. While providing significant computing and communication power such systems have difficulty accommodating conventional interface devices such as keyboards, mouses and displays. An obvious alternative is speech, both for input and for output. The present paper describes an initial attempt to build such an interface in the context of a system for mobile inspection.

The task we chose was initially developed as part of the VuMan[11] project at Carnegie Mellon University. The VuMan has been used for a limited technical inspection (LTI) of an amphibious assault vehicle for the USMC at Camp Pendleton, as a replacement for a clipboard and pencil procedure. The VuMan allows a mechanic to directly enter inspection data into a computer and has been shown to reduce inspection time by a half.

Despite this, the VuMan has a number of limitations, particularly a very low-bandwidth input device, the "rotary mouse". Input activity consists of circularly traversing hotspots on a display using a dial on the device and clicking on spots corresponding to desired inputs. In the worst case, the user is shown the image of a keyboard and needs to enter data character by character using the mouse. Given this, speech seemed like an obvious enhancement to the task.

2. ADAPTING LTI FOR SPEECH

The original VuMan LTI task was implemented using a custom hypertext system, primarily because of processing constraints imposed on that device (a 25MHz Intel 386). As we were primarily interested in the speech interaction aspects of the task, we chose to implement our system using a standard notebook computer with a more powerful processor. The computer, plus a battery power supply and control hardware for the head-mount display were placed in a pack worn on the user's back. This arrangement, although bulkier than the VuMan package (which can be attached to the user's belt), allowed users to freely move about, inspect the underneath of the vehicle, climb to the roof, etc.

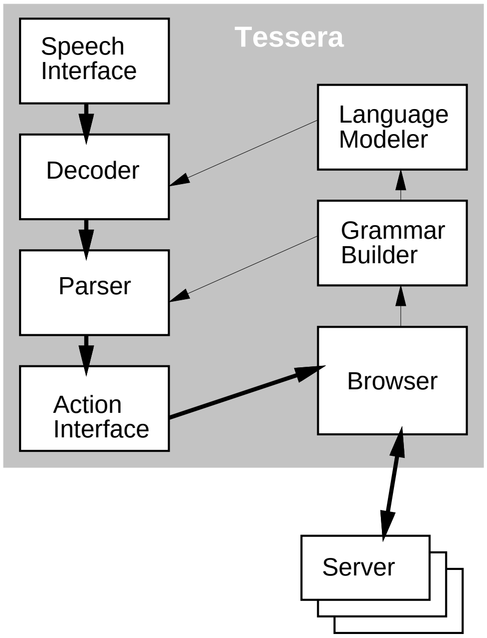
For purposes of the current study the task was recast as a hypertext document using standard `html` format, allowing for a more rapid and flexible design process. The `html/http` framework offers a simple yet powerful mechanism for unifying information resources useful for this task, both for data collection and for access to distributed resources. Using a standard browser also allowed us to incorporate a variety of information, such as a scanned repair manual and video clips keyed to individual steps in repair procedures, all accessible by voice.

3. SYSTEM DESCRIPTION

The SpeechWear system makes use of a Toshiba T4900ct notebook computer containing a 75MHz Pentium processor, 40Mb of RAM and running Windows NT 3.5. Input is through a head-mounted microphone and output through a small head-mounted (grey-scale) VGA display with a speaker attached to its frame. Communications is is by means of a WaveLAN transmitter.

Recognition services are provided by a real-time implementation of the Sphinx-II recognition system [7], a continuous-speech speaker-independent system based on hidden Markov modeling. Spoken language interpretation made use of the Phoenix [2]. The system implements a "continuous listening" protocol[12] that allows the task to be performed hands-free. A modified mouse is provided to turn the system on and off. Figure 1 shows a diagram of the system.

Figure 1: The SpeechWear system



The NCSA Mosaic browser[3] provides the interface to the task hypertext document. It was modified by merging the spoken language code into it to create a single multi-threaded application. Inspection data was recorded through the use of `FORM`s embedded in the task document. As the interface is a speech-enhanced version of the Mosaic browser, communication is through the standard `http` protocol and makes use of servers and `CGI`[4] scripts to implement the inspection system.

4. HYPERSPEECH

To provide speech understanding services, we developed a backward-compatible extension to `html` which facilitates the incorporation of language-specific information into hypertext documents. This approach is somewhat different from that commonly chosen by others [1, 6, 10, 5] which is to store such information in data structures that are parallel to the browser's internal representation of the information on a hypertext page. This is a workable approach in cases where speech is meant to support primarily navigation (i. e., "following links"). However, we were also interested in using native `html` data entry conventions, in particular the `FORM` construct, to capture inspection data in a manner that could take advantage of existing browser mechanisms.

Our extensions to the mark-up language allow direct association of grammar fragments with `html` clauses, specifically anchors and actions inside `FORM`s. The grammar information is extracted by the speech-aware version of Mosaic (Tessera) and is merged into a generic browsing language that allows for voice input of display manipulation commands (such as for scrolling or for traversing the history list). No attempt was made to allow voice control of every aspect of the interface as most were not relevant to the task at hand.

Figure 2: Augmented link `html` used in SpeechWear.

```
*2. <A HREF="/section/ltip7_sec1_a2.html">
<GRAMMAR VALUE="
        ( question two )
        ( towing eyes )
">
 Towing Eyes.
</A>
```

As the browser receives a speech-enabled page, it parses it in its normal fashion. The Grammar Builder component then traverses the parse tree and extracts information from any `GRAMMAR` fields. These are used to dynamically create a grammar fragment that encompasses all speakable items on the page. This partial grammar is then merged with the statically-defined browser grammar to produce the active grammar for that page. This grammar is made available to the Phoenix parser and is also used to derive a bigram language model for the benefit of the decoder. Since the domain language is known beforehand, pronunciations for words can be compiled off-line for efficiency, though these could be obtained as needed from a server (an alternate solution which we have also implemented).

Initial `GRAMMAR` clauses were generated by automatic conditioning of the task hypertext. Where advisable, alternative locutions were generated, as in the example in Figure 2. For the most part, the language was generated automatically from the actual text of the inspection form. Only in the case of free-form inputs was a prespecified grammar used (see Figure 3, which also shows the use of non-terminals built into the language component).

While automatic processing is used to initially populate a document with language information, manual additions can also be made to a `GRAMMAR` clause to reflect arbitrary usage encountered in the field. By this means, the hypertext document can be updated to better approximate the language of the user population.

The above solution is not completely satisfactory as it requires modification of Web pages to make them "speakable". This is somewhat mitigated by the fact that pages can be

Figure 3: Augmented `FORM` `html` used in SpeechWear.

```
Inspector ID :
<INPUT TYPE="TEXT"
NAME="begin.inspector_id.ID"
PHOENIXNAME="begin.inspector_id.ID"
GRAMMAR="( [digit] [digit] [digit] [digit] )">
```

automatically preprocessed to include the necessary information. In principle such processing could be done at the time of page retrieval, allowing the document to be modified without the need to preprocess it for inclusion of speech information. Such an organization would also allow for unrestricted navigation of documents available over the World Wide Web. In the environment we are considering, this would be of benefit, as it would allow the user in the field to consult a variety of sources, such as centrally maintained documentation or even specifications published by manufacturers, not all of which would (or should) be expected to have been preprocessed for the benefit of the speech-based user.

To allow complete automation, three operations need to be available: the conditioning of text into speakable form (e.g., transforming *35* into *thirty five*), establishing pronunciations for the resulting words and creating a suitable language model for the decoder. Such a protocol would be sufficient to support most forms of navigation, but might not be adequate for specifying language for certain FORM elements which (for efficiency) might benefit from manual specification, as in the example above (a large vocabulary language could always be attached implicitly to an input field). Presumably workable solutions could be developed for specific applications once the details are known.

5. LTI TASK DESCRIPTION

The inspection consists of a checklist of 467 items. The checklist is divided into eleven sections, grouped into four major vehicle subsystems and in its typical version normally takes about 3 hours to complete. The check-off procedure consists of inspecting an item and noting its condition (*Serviceable*, *Unserviceable*, *Missing* or *On ERO*). If the condition is not deemed *Serviceable*, the user is required to comment on the condition of the item. The VuMan implementation of the task followed this structure more or less exactly, except that the Comment section was implemented as a multiple-choice question rather than a free-form comment (due to the limitations on the input channel). The items in the multiple-choice sets were chosen as representative of the most common faults encountered (based on an interview of maintenance personnel). The current implementation follows this design.

In terms of the maintenance process, the inspection serves as a tool for the mechanic to fill out a comprehensive work order; the work-order notations are used to prioritize the repair work. The work order is used to initiate the ordering of new parts and to track the progress of the repair work.

The framework provided by CGI permits the use of a flexible control structure and allows the implementation of different interaction protocols. The inspection task allows both for user control of the sequence of items visited (through standard browser navigation features) and for the imposition of certain contingencies by the data collection script. For example, indicating that a part is not in operable condition automatically places the user on the comment page for that item. Similarly, the system can be configured to either request explicit confirmation for each item or to step through the inspection list automatically.

Table 1: Error Analysis for field trial data

source of error	*amount*
Signal processing / mic	30%
Language coverage	35%
Instructions	12%
Other	23%

6. FIELD TRIAL

A prototype of the system was tested during the course of a field trial that took place at Camp Pendleton in June 1995. During the course of the trial, three (male) mechanics performed partial LTI inspections. (Excluded were inspections of the engine plenum, a physically demanding procedure.) Participants were assigned to the study by their supervisor and were individually introduced to the system in a structured training session.

The training approach used a combination of modeling an experienced user and explicitly instructing the novice in proper use. Thus first the user observed the experimenter using the system (on a separate notebook computer), then was invited to use it himself and become comfortable with its operation. At that point, the wearable system was given to the user to try out and questions were entertained. The training process was limited to 10 minutes and was paced by the individual's progress (no participant needed the entire period). At the conclusion of training, all proceeded to the vehicle and the inspection was carried out. Upon completion, the mechanic participated in a structured interview that assessed their impressions of the device.

The system was instrumented to collect a variety of data, including: the actual utterances produced by the user, their decodings, decoder and task timings and the sequence of links traversed. System response was at a median of 4.2 xRT, producing a corresponding lag of 3.8 s per input (utterances were 0.8 s median duration). Recognition word error ranged between 12%–15% across subjects. Detailed analysis of the errors (Table 1) suggests that the majority of the recognition errors were due to factors that can be brought under control through additional development. This includes a better choice of microphone, a more complete domain language and more focussed user training.

User interviews indicated that the participants came away with a favorable impression of the novel inspection device and indicated they would be willing to use it in regular work. At the same time, the users pointed out a number of deficiencies: the device appeared subjectively slower than the traditional paper-and-pencil system. There is reason to believe that some of this impression may be based on a simple lack of experience with the system (users will typically experience long-term improvement in task completion time while using a speech system, e.g. [9]). It also became apparent that an interface that is capable of actively guiding users when they exhibit difficulties would also be of value. We have since explored strategies for monitoring the input stream and detecting patterns that suggest the user is in trouble (for example, a sequence of identical inputs). This in turn can be used to trigger a separate clarification dialog.

It was clear that the design of the system could be improved in a number of ways. In particular, a better microphone (which we have since identified) and a more comprehensive coverage of the domain language (the task was designed without first-hand experience of the domain) can reduce the number of errors by a factor of two-thirds. The excessive response lag could also be reduced by more careful exploitation of the constraints available in this domain and by tailoring the properties of the speech system to conform more closely to the task language (our current implementation runs at 2.6 xRT and continues to be improved).

7. GENERAL OBSERVATIONS

The development of the SpeechWear system was a success: a working system was produced and was tested in the field under conditions of actual use. At the same time an extensible infrastructure was created (SpeechWare) that can be applied to a variety of domains based on hypertext multimedia documents.

The experience also revealed a number of problems with this approach. For example, the form of the task as designed followed quite closely that used in the original VuMan implementation and was implicitly constrained by the characteristics of the rotary mouse interface. Analysis of the task structure, for example, suggests that a different protocol (implicit confirmation [8]) could eliminate approximately half the steps in the original task, by implicitly channeling the dialog along the most likely path and relying on the user to indicate deviations. An analysis of the data showed that about 90% of items were judged *Serviceable*, yet the protocol required the user to input this item explicitly, then confirm it. A simple confirmation of a suggested default input (*Serviceable*) would have been sufficient to enter the inspection outcome.

8. SUMMARY

The system we have implemented uses speech to increase the input bandwidth for a wearable computer used in hands-busy environments. The original hypertext structure of the inspection task was enhanced by recasting it into a conventional `html` format, allowing the user interface to be used not only to access the inspection document, but also to provide access to a variety of task-relevant documents, both local to the device and available remotely through a wireless LAN. Finally, we have specified a speech extension to `html` which allows specialized browsers to accept voice equivalents of standard browser inputs.

9. Acknowledgements

We would like to thank Sunil Issar, Ravi Mosur, Dick Martin and Asim Smailagic for their invaluable help at different stages of this project. This work was supported by the Advanced Research Projects Agency under Grant N00014-93-1-2005.

10. REFERENCES

1. Charles T. Hemphill and Philip R. Thrift. Surfing the Web by voice. In *Proceedings of ACM Multimedia'95, San Franscisco, CA*. ACM, November 1995.
2. Sunil Issar and Wayne Ward. Flexible parsing: CMU's approach to spoken language understanding. In *Proceedings of the Spoken Language Technology Workshop*, pages 53–58, San Franscisco, CA, March 1994. ARPA, Morgan Kaufmann.
3. Mosaic Home Page. `http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/`.
4. NCSA. The common gateway interface. `http://hoohoo.ncsa.uiuc.edu/cgi/`.
5. Bill Noon. ListenUp! Speech recognition plugin for Netscape. `http://snow.cit.cornell.edu/noon/ListenUp.html`.
6. David G. Novick and David House. Spoken-language access to multimedia (SLAM): A multimodal interface to the World Wide Web. `http://www.cse.ogi.edu/SLAM/slam-paper.html`, 1995.
7. Mosur K. Ravishankar. *Efficient algorithms for speech recognition*. Phd, Carnegie Mellon University, Pittsburgh, PA, May 1996.
8. A. I. Rudnicky and A. G. Hauptmann. Models for evaluating interaction protocols in speech recognition. In *Proceedings of CHI*, pages 285–291, New York, April 1991. ACM.
9. Alexander I. Rudnicky. Mode preference in a simple data-retrieval task. In *Proceedings of the Arpa Workshop on Human Language Technology*, pages 364–369, San Mateo, CA, March 1993. Morgan Kaufmann.
10. Shocktalk. `http://www.emf.net/~dreams/shocktalk/`.
11. Asim Smailagic and Daniel P. Sieworek. Modalities of interaction with CMU wearable computers. *IEEE Personal Communications*, 3(1):14–25, Feb 1996.
12. Reed Stephen D. Utterance end-pointing in the presence of noise using Gaussian classification of cepstral coefficients. Technical Report TR 1995-8, Information Networking Institute (Carnegie Mellon University), Pittsburgh, PA, 1995. Masters' Thesis.