HearSay: Context-Directed Non-Visual Web Browser

Problem & Motivation

The Word Wide Web has come to take an important part in our lives: it has become indispensable for finding information, communicating with others, and performing daily tasks for work, banking, and shopping. Web sites are designed mostly for graphical interaction, limiting access for an entire community of people with visual disabilities. Just to give a sense of the size of this population, in the U.S. alone there are over 10 million people with visual disabilities, of whom approximately 1.3 million are legally blind, as reported by the American Foundation for the Blind. And, according to the WHO's World Health Report, there are approximately 45 million people without sight worldwide.

In 2003, the U.S. Bureau of the Census reported that, among people who are blind or have severe vision impairment, 63% of the adults (ages 25-60) in the labor force and 40% of adults not in the labor force were using the Internet. 23% of the adults over age 60 were also using the Internet. Because blindness affects older people, the total number of users who are likely to benefit from assistive technologies in the U.S. has been increasing due to the aging baby boomer population. Forrester Research projects 70 million users will depend upon various assistive technologies by 2010.

A number of software companies have already realized the need for assistive technologies to help people with visual disabilities use the Web. For example, major operating systems such as Microsoft Windows and OsX provide a choice of accessibility options such as magnifying glass, text narration, etc. There are also other software tools used for Web browsing, such as screen readers. Some well-known and widely used screen-readers and audio-browsers are Freedom Scientific's JAWS, G.W. Micro's Windows-Eyes, Hal and SuperNova from Dolphin Computer Access, and IBM's Home Page Reader. A screen-reader typically reads out the content of the screen and gives other audio or Braille feedback to help navigate Web pages. These tools enable people who have visual disabilities to interact with the Web and perform basic online activities.

Assistive software often relies on Web accessibility guidelines (*e.g.:* W3CWeb Accessibility Initiative - http://www.w3.org/WAI/). Unfortunately, not all Web sites are accessible. Many businesses have already realized a strong business rationale for making their sites accessible, and Web sites run by the government are required to do so by law. Section 508 of the Rehabilitation Act of 1973 was amended in 1998 to address Web accessibility and mandate that "electronic and information technology developed, procured, used, or maintained by all agencies and departments of the Federal Government be accessible both to Federal employees with disabilities and to members of the public with disabilities, and that these two groups have equal use of such technologies as federal employees and members of the public that do not have disabilities". The scope of such laws keeps expanding. For instance, a new rule proposed by the U.S. Dept. of Transportation would require Web sites of airlines be fully accessible.

Unfortunately, even the most accessible Web sites cannot be efficiently browsed with the existing assistive software. Sighted users can visually process Web content, filter out irrelevant data, and quickly identify the information that is most relevant to them. At the same time, most screen-readers process Web pages sequentially, i.e. they first read through the menus, banners, commercials, or anything else that comes before the desired information. Therefore, people with visual disabilities have to process information in the order it is presented in Web pages and can only skip between sections of text with keyboard shortcuts, because screen-readers provide almost no content-analysis to facilitate access to information, or content-filtering to eliminate "noise" (*e.g.:* advertisements). As a result, these users often have to listen to or skip through substantial page content before they get to the information. Thus, browsing becomes time-consuming and causes significant **information overload**.

Although assistive software has made the Internet accessible for people with visual disabilities, current state-of-the-art assistive tools fall woefully short of meeting the intent of the Disability Act. The drawbacks of modern screen-readers pose challenges for blind people to employ the same browsing patterns, as those used by sighted people, making Web browsing time-consuming and strenuous. Thus, there is a clear and pressing need for assistive software that can accelerate browsing and help simulate the browsing behavior of sighted users by analyzing Web content and helping identify relevant information in Web pages.

Figure 1: Product Search Example

Identification of relevant information on any distinct Web page is subjective. However, as soon as the user follows a link, it is often possible to use the context of the link to find the relevant information on the next page and present it to the user first. Consider an example when a blind person is looking for an MP3 player at BizRate.com (Figure 1).

Using a standard shortcut-driven voice-browsing interface, the user finds the MP3 category in the product taxonomy in Figure 1(a). When he or she follows the link, shown by the mouse cursor, we can use the words of the link to find the segment that contains the desired information, surrounded by the box enclosing the two items in Figure 1(b). The voice browser then starts reading the page from that position. After the MP3 player of interest has been found, and the user follows the SanDisk link in Figure 1(b), the words of the link and its context, enclosed by the dotted rectangle, are used to find detailed description of the MP3 player, surrounded by the solid box in Figure 1(c).

This paper presents a non-visual Web browser, HearSay (a.k.a. CSurf), that makes possible the scenario described above. HearSay brings together Content Analysis, Natural Language Processing (NLP), and Machine Learning algorithms to help blind users quickly identify relevant information on following a link, thus, considerably reducing their browsing time.

Background and Related Work

The work described in this paper has broad connections to research in non-visual Web access, Web content analysis, and contextual analysis. The first mention of HearSay appears in [1]. Our initial ideas on context-directed browsing were described in a short paper [2].

Non-visual Web Access. Several research projects aiming to facilitate non-visual Web access include work on browser-level support, content adaptation and summarization [3, 4], organization and annotation of Web pages for effective audio rendition [6, 7, 8], etc. An example of a VoiceXML browsing system (which presents information sequentially) is described in [5]. All of these applications do not perform content analysis of Web pages. BrookesTalk [3] facilitates non-visual Web access by providing summaries of Web pages to give its users an audio overview of Web page content. The work described in [4] generates a "gist" summary of a Web page to alleviate information overload for blind users. However, summarization of the entire page does not help find the relevant information within it. HearSay audio browser goes beyond these systems in scope and approach: it analyzes the content of Web pages and helps find relevant information in them while navigating from one page to another. The works describing organization and annotation of Web pages for better audio rendition typically rely on rules [6] or logical structures [7]. The ASTER system [6] permits visually challenged individuals to manually define their own document-reading rules. Other researchers propose the idea of extracting content using semantics [8]. They describe a framework for manual annotation of the content w.r.t. a schema, representing the task a user wishes to accomplish. These annotation rules are also site specific, and, hence, not scalable over content domains. The essential difference between our work and all of the above-mentioned research is that we do not require any domain knowledge in terms of rules. CSurf dynamically captures the contextual information and uses it to facilitate non-visual Web access.

Web Content Analysis. A critical piece of our context analysis algorithm is in partitioning Web pages into geometric segments (blocks). Substantial research has been done on segmenting Web documents [9, 10, 11]. These techniques are either domain-specific [10], site-specific [9], or depend on fixed sets of HTML markups [11]. Semantic partitioning of Web pages has been described in [12]. These systems require semantic information (e.g. ontologies). In contrast to all of these works, our geometric clustering method does not depend on rules, domain knowledge or ontologies. Web page partitioning techniques have been used for content adaptation [13, 14] and content caching [15]. VIPS [16] algorithm uses visual cues to partition a Web page into geometric segments. This algorithm is used in [17], where the segments are described by a set of features (e.g. spatial features, number of images, sizes, links, etc.). The feature values are then fed into an SVM, which labels the segments according to their importance. HearSay also uses an SVM to rank the blocks on the destination Web page w.r.t. the context of the link in the source page. In contrast to [17], where the SVM model

was learned using features only from the content of Web page segment, our SVM model uses the feature set, Table 1, computed from both the context of the link and the content of the block. The main difference between our research and the above-mentioned techniques is that we exploit geometrical and logical structure of Web pages both to collect context and to identify relevant information on the next Web page.

Contextual Analysis. The notion of context has been used in different areas of Computer Science research. For example, [18] defines context of a Web page as a collection of text, gathered around the links in other pages pointing to that Web page. The context is then used to obtain a summary of the page. Summarization using context is also explored by the InCommonSense system [19], where search engine results are summarized to generate text snippets. The use of contextual information for non-visual Web access is not a well-studied problem. A technique resulting from early efforts at context analysis for non-visual Web access is described in [20], where context of a link is used to get the preview of the next Web page, so that visually disabled individuals could choose whether or not they should follow the link. This idea is used in AcceSS system [21], to get the preview of the entire page. However, presenting a preview does not guarantee the reduction of browsing time. All of these works define the context of the link as an ad hoc collection of words around it. In contrast, our notion of context is based on topic similarity of text around the link. We use a principled approach for context analysis with a simple topic boundary detection method [22], confined to geometric clusters that have semantically related content. HearSay is fundamentally different from all of these works in its application. Specifically, it aims to help visually disabled users quickly identify relevant information on following a link, thus, potentially reducing their browsing time.

Uniqueness of the Approach

HearSay non-visual browser is developed at Stony Brook University in collaboration with the Helen Keller Services for the Blind in Hempstead, NY. HearSay features a flexible dialog interface and innovative context-directed browsing [23] not used by any existing screen-readers. We are striving to implement the best features of existing screen-readers, but go beyond them in usability. HearSay helps visually impaired and blind users browse the Web more efficiently and quickly identify relevant information in Web pages. The technology underlying HearSay brings together the fields of content analysis, natural language processing, and machine learning.

Users interact with the HearSay Web Browser through a dialog system, which is an extended VoiceXML interpreter, VXMLSurfer [24], that I have developed (ACM SRC Finals). The dialog system uses VoiceXML dialogs to communicate with its users, process user input, and present Web page content. The Interface Manager provides both basic and extended screen-reader navigation features, such as shortcuts, voice controls, etc. The system allows both keyboard and voice inputs, and can process text commands along with keyboard shortcuts. Text-to-speech and speech recognition engines are accessed through the Java Speech API (JSAPI), providing a flexible interface capable of supporting different speech engines. In its current configuration, the Interface Manager uses freely available FreeTTS text-to-speech engine and Sphinx speech recognition engine. HearSay now also supports high-quality affordable commercial voices developed by Cepstral.com.

HearSay uses JREX Java API wrapper coupled with Mozilla Web Browser to access the Internet. Mozilla engine takes care of all of the standard browser functionalities such as support for cookies, secure connection, history, pop-up blocking, etc. We have extended JREX to extract a *Frame Tree*, Mozilla's internal representation of a Web page, *after* the Web page has been rendered on the screen. This way, Mozilla takes care of any dynamic content, cascading style sheets, malformed HTML, and other rendering problems. This relieves HearSay browser from having to deal with heavy DOM-tree objects, while giving it even more information about the content and the style. Many screen-readers, such as JAWS, continue using DOM tree information.

We define a *Frame Tree* as a tree-like data structure that contains Web page content, along with its 2-D coordinates and formatting information that specifies how the Web page has to be rendered on the screen. A frame tree is composed of nested *frames*, so that the entire page is a root frame, containing other nested frames down to the smallest individual objects on the page. For example, Figure 2(a), section 1, shows a snapshot of the New York Times front page, with rounded rectangles illustrating the frames. In Figure 2(b), the corresponding frame-tree is partially expanded to demonstrate the types of frames. We distinguish between the following classes of frames: text, links, images, image-links, and non-leaf frames.

We use a collection of VoiceXML dialog templates to convert the frame tree into Voice-XML dialogs, which are then interpreted by the VXMLSurfer. A number of sub-dialogs are also used to present history, help, lists of (un-)visited links in the page, etc. We currently support only basic screen-reading dialogs. More research is needed to determine the optimal structure and representation for the zooming, customizable, and domain specific dialogs, which can be also expressed using VoiceXML and processed by the VXMLSurfer dialog system.

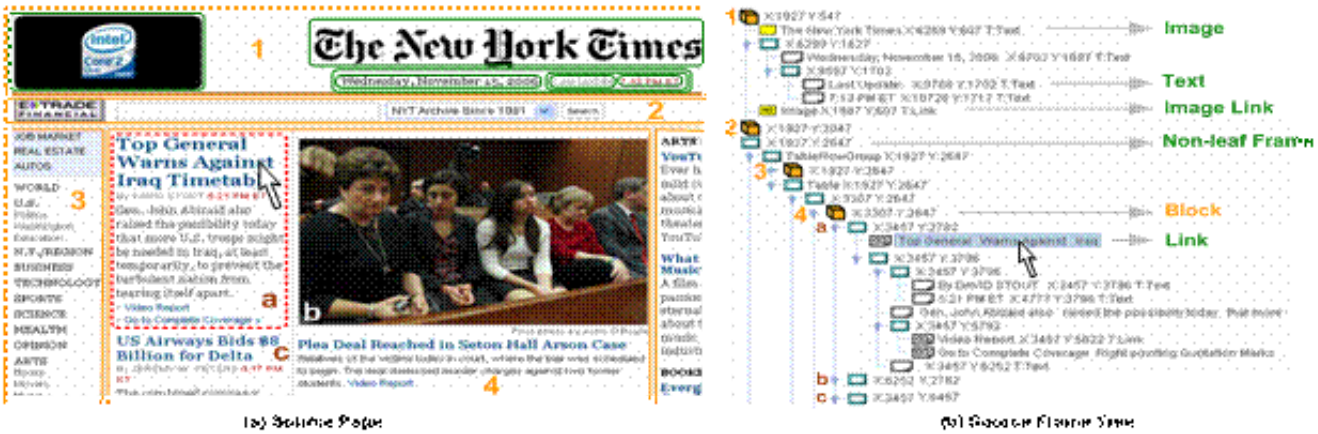


Figure 2: Context Identification

We use our geometric segmentation algorithm to partition the frame tree into sub-trees, which correspond to blocks of the Web page and represent semantic clusters of information. We use an observation that semantically related information exhibits spatial locality and often shares the same alignment on a Web page. Since a frame tree represents the layout of a Web page, we infer that geometrical alignment of frames may imply semantic relationship between their respective content. The geometric segmentation algorithm identifies the largest sub-trees whose children are consistently X- or Y-aligned. Thus, they are likely to be the largest possible clusters containing semantically related items of information. For example, Figure 2(a) shows how the alignment information is used to cluster the New York Times Web page into blocks: banner labeled as 1, search - 2, taxonomy - 3, and news - 4. The frame tree nodes corresponding to these blocks are marked with 3-D block icons in Figure 2(b). The blocks, obtained by the geometric segmentation algorithm, are further used by the context identification and relevant block identification algorithms. The Dialog Generator module of HearSay also makes use of the blocks when structuring its VoiceXML dialogs.

The essence of the context-directed browsing is in using the context of the clicked link to identify what is relevant on the following page. We define the **context** of a link as the content around the link that maintains the same *topic* as the link.[illegible] To collect the context, a cosine-similarity-based topic-detection algorithm is applied to the information surrounding the followed link. Consider Figure 2, showing the front page of The New York Times Web site and the corresponding frame tree. The context of the link, indicated by an arrow, is the text surrounded by the dotted line. Notice how the topic changes from one headline to another.

After the context of the link has been gathered on the source page, we again use our geometric segmentation algorithm to partition the destination page into blocks: 1, 2, and 3 in Figure 3. Then, the relevant block identification algorithm matches the context against every block in the frame tree and uses a support vector machine (SVM) to compute the relevance of each block with respect to the collected context. Subsequently, the Web page is presented to the user starting with the highest ranking block. If the relevant block was not identified correctly, the user can always skip to the beginning of the page.

[illegible]

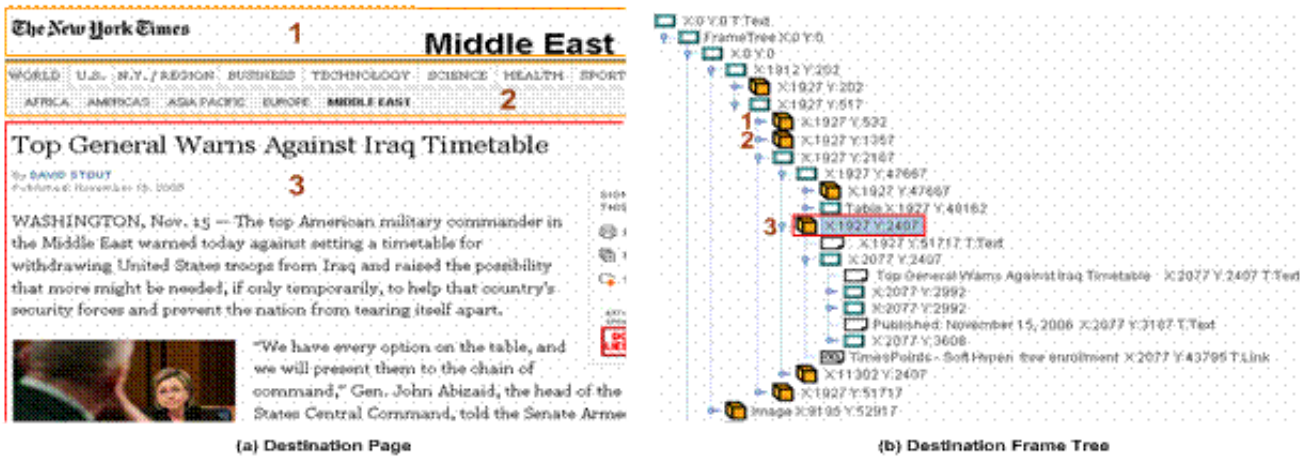


Figure 3: Most Relevant Block Identification

The SVM, as well as the context identification algorithm, was trained on the collection of human selected pairs of source destination Web pages. Each of the pairs had the corresponding link, context, and the most relevant block selected.

We used the collected source pages to statistically compute the accuracy and the threshold for our topic boundary detection algorithm. We used 50% of the page samples to estimate the cosine-similarity threshold value, and the remaining 50% were used to calculate the accuracy of topic identification algorithm. We defined the *accuracy* of context identification as cosine similarity between the human- and computer-selected context, with the cosine similarity value ranging between 0 and 1, where 1 signified a 100% match. We used this measure for threshold estimation, as well as quantitative evaluation of the context-identification algorithm. Finally, we used the remaining 500 collected Web pages to calculate the average accuracy of context identification in each of the 5 domains.

The collected data was also used for SVM training and the evaluation of the algorithm's accuracy. Using the relevant block identification algorithm on each pair of the collected Web pages, we computed a feature vector for each block of the destination Web page. The human-selected most-relevant blocks were labeled with 1's (i.e. relevant), while the rest were labeled with 0's for training the SVM. We divided the training data in two sets: training (90%) and cross-validation (10%). Once the SVM model was learned, we tested it by using 100 Web page pairs of the cross-validation set to predict block labels. The labels were then compared with the human-selected ones to calculate the accuracy of the prediction.

Results and Contributions

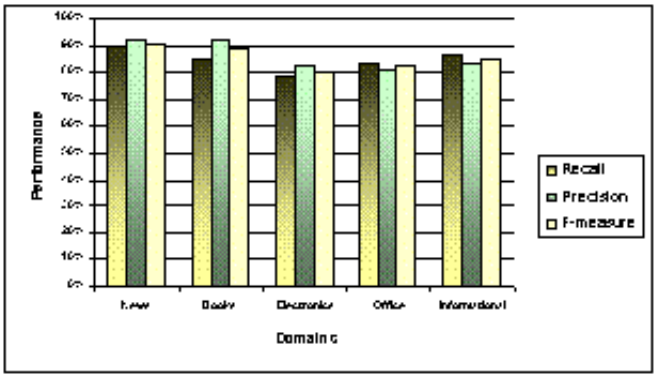
The HearSay non-visual Web browser partitions Web pages into geometrical segments, and uses techniques from Natural Language Processing and Machine Learning to help blind people browse the Web. Using HearSay, visually impaired individuals can potentially imitate the browsing behavior of sighted users, saving time on not listening to irrelevant information. It also uses an extensible dialog system to interact with the users. Thus, HearSay goes beyond traditional screen-readers in the flexibility of its interface and in its ability to combat information overload.

HearSay targets both blind and visually impaired users. Users interact with HearSay through a large input window, which could be useful for individuals with low vision, and an extensible dialog system written in VoiceXML. Non-visual navigation is treated as a dialog between a human and a Web browser. The dialog system is managed by VXML-Surfer [24] - a flexible VoiceXML interpreter that advanced me into the grand-finals of ACM graduate SRC. VXMLSurfer allows users to control HearSay with voice commands, text commands, and keyboard shortcuts. It can also handle multiple layers of dialog that give different views of the same Web page. Currently, HearSay uses only a basic screen-reading view.

HearSay uses our geometrical clustering algorithm [23] to partition Web pages into meaningful segments usually containing semantically related information. The segments often correspond to menus, ads, tables, articles, etc., as can be seen in Figures 2 and 3, making it easy to navigate on most Web pages. HearSay allows users to move between these segments, as well as between the individual items and sentences on the page.

One of the highlights of HearSay is context-directed browsing, which allows users to avoid listening to Web pages from the beginning, and so to skip banners, ads, menus, etc. Instead, when users navigate from one page to another, HearSay starts reading the page from the most relevant section, thus, reducing information overload and potentially saving browsing time. It is a unique and innovative feature not used by any non-visual browsers. If HearSay misses the relevant section, it is very easy to go to the beginning of the page.

HearSay has been evaluated by both blind and sighted users. It was impractical to get quantitative measurements of the accuracy of our algorithms with blind users. Therefore, we used sighted students to obtain those metrics, while blind evaluators helped get qualitative feedback. The algorithm for context identification scored in the range of 80% to 90% accuracy compared with human-selected context (Figure 4). The SVM model for relevant block identification showed an average of 91% accuracy in its prediction of the most relevant block (Figure 5).



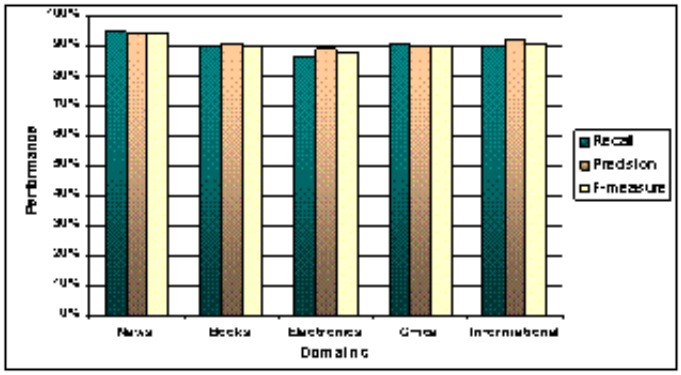


Figure 4: Performance of Context Identification

Figure 5: Performance of Relevant Block Identification

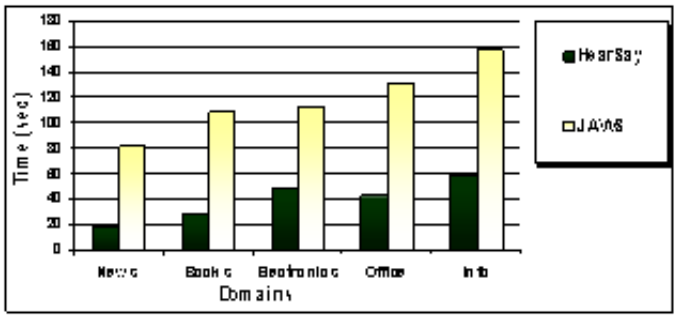


Figure 6: HearSay vs. JAWS

We also performed preliminary evaluation of the advantages introduced by context-directed browsing vs. simple screen reading in five different Web site domains. Since our goal was to provide faster access to relevant information, we measured the time taken to reach the desired information after following a link using HearSay and a state-of-the-art screen-reader, JAWS (Figure 6). Our preliminary results show promise that context-directed browsing can substantially improve browsing efficiency for blind people. The complete description of the algorithms, experiment, and the results appear in [23].

HearSay research resulted in several publications described in the Related Work section. The reviewers of the 16th International World Wide Web Conference (Banff, Canada, May 2007) have unanimously rated our paper [23] as a "strong accept". The paper was also nominated for best paper award. The HearSay Web Browser was also entered in the W4A Web Accessibility Challenge Competition at the International Cross-Disciplinary Conference on Web Accessibility taking place in Banff, Canada, in May 2007. We were also able to use the ideas of context-directed browsing in mobile devices, where full Web pages get deformed and finding relevant information takes a lot of scrolling. The paper describing our prototype context-directed browser, CMo, will appear in MobiSys 2007 [25].

HearSay is an ongoing research project with plenty of room for usability improvements. While we are striving to introduce innovative ideas to facilitate non-visual browsing, we also understand the importance of being in touch with the end-users. We are consulting with the instructors of Helen Keller Services for the Blind and conducting evaluations to make sure that we meet the needs of the end-users. Among our current projects there are handling JavaScript menus, analysis of HTML forms and tables, pattern-based partitioning, domain specific dialogs, transaction support, personalization, spell-checking, multilingual interface, support of Braille keyboards, etc. The official release of HearSay is scheduled for May 2007.

As the manager of the HearSay project, I would like to acknowledge my colleague, Jalal Mahmud, who takes a share of the HearSay research and is a co-author of many HearSay papers. I am also grateful to my co-advisors Dr. I.V. Ramakrishnan and Dr. Amanda Stent without whom this research would not be possible. Finally, I would like to thank NSF for supporting this research (Award IIS-0534419) and I extend my appreciation to the HearSay development team.

References

[1] I. Ramakrishnan, A. Stent, and G. Yang. Hearsay: Enabling audio browsing on hypertext content. In *WWW*, 2004.

[2] J. Mahmud, Y. Borodin, D. Das, and I. Ramakrishnan. Combating information overload in non-visual web access using context. In *IUI*, 2007. Short paper.

[3] M. Zajicek, C. Powell, and C. Reeves. Web search and orientation with brookestalk. In *Proceedings of Tech.and Persons with Disabilities Conf.*, 1999.

[4] S. Harper and N. Patel. Gist summaries for visually impaired surfers. In *Assets '05: Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility*, pages 90-97, 2005.

[5] http://www.internetspeech.com.

[6] T. Raman. Audio system for technical readings. *PhD Thesis, Cornell University*, 1994.

[7] M. Hori, G. Kondoh, K. Ono, S. Ichi Hirose, and S. Singhal. Annotation-based web content transcoding. In *WWW*, 2000.

[8] A. Huang and N. Sundaresan. A semantic transcoding system to adapt web services for users with disabilities. In *ASSETS*, 2000.

[9] H. Takagi, C. Asakawa, K. Fukuda, and J. Maeda. Site-wide annotation: Reconstructing existing pages to be accessible. In *ASSETS*, 2002.

[10] D. Embley and L. Xu. Record location and reconfiguration in unstructured multiple-record web documents. In *WebDB*, 2000.

[11] Y. Yang and H. Zhang. HTML page analysis based on visual cues. In *Intl. Conf. on Document Analysis and Recognition (ICDAR)*, 2001.

[12] S. Mukherjee, I. Ramakrishnan, and A. Singh. Bootstrapping semantic annotation for content-rich html documents. In *ICDE*, 2005.

[13] O. Buyukkoten, H. Garcia-Molina, and A. Paepcke. Seeing the whole in parts: Text summarization for web browsing on handheld devices. In *WWW*, 2001.
[14] S. Baluja. Browsing on small screens: recasting web-page segmentation into an efficient machine learning framework. In *WWW*, pages 33-42, 2006.
[15] L. Ramaswamy, A. Iyengar, L. Liu, and F. Douglis. Automatic detection of fragments in dynamically generated web pages. In *WWW*, 2004.
[16] S. Yu, D. Cai, J.-R. Wen, and W.-Y. Ma. Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In *WWW*, 2003.
[17] R. Song, H. Liu, J.-R. Wen, and W.-Y. Ma. Learning block importance models for web pages. In *WWW*, pages 203-211, 2004.
[18] B. B.-M. J.-Y. Delort and M. R. Enhanced. Enhanced web document summarization using hyperlinks. In *HYPERTEXT'03: Proceedings of the fourteenth ACM conference on Hypertext and hypermedia*, pages 208-215, 2003.
[19] E. Amitay and C. Paris. Automatically summarizing web sites - is there a way around it? In *Proceedings of ACM 9th CIKM*, pages 173-179, 2000.
[20] S. Harper, C. Goble, R. Stevens, and Y. Yesilada. Middleware to expand context and preview in hypertext. In *Assets '04: Proceedings of the 6th international ACM SIGACCESS conference on Computers and accessibility*, 2004.
[21] B. Parmanto, R. Ferrydiansyah, A. Saptono, L. Song, I. W. Sugiantara, and S. Hackett. Access: accessibility through simplification & summarization. In *Proceedings of the International Cross-Disciplinary Workshop on Web Accessibility W4A'05*, pages 18{25, 2005.
[22] J. Allen. Topic detection and tracking: Event-based information organization. Kluwer Academic Publishers, 2002.
[23] J. Mahmud, Y. Borodin, and I. V. Ramakrishnan. Csurf: A context-driven non-visual web browser. In *International WWW Conference*, 2007.
[24] Y. Borodin. A flexible VXML interpreter for non-visual web access. In *ACM ASSETS*, 2006.
[25] Y. Borodin, J. Mahmud, and I. V. Ramakrishnan. Context Browsing with Mobiles - When Less is More. In *MobiSys* 2007.