Capti-Speak: A Speech-Enabled Web Screen Reader

Vikas Ashok[1], Yevgen Borodin[2], Yury Puzis[2], I V Ramakrishnan[2]

[1]Dept of Computer Science
Stony Brook University
Stony Brook, NY, USA

[2]Charmtech Labs LLC
CEWIT, 1500 Stony Brook Rd.
Stony Brook, NY, USA

ABSTRACT

People with vision impairments interact with web pages via screen readers that provide keyboard shortcuts for navigating through the content. However, web browsing with screen readers can be a frustrating experience mainly due to time and effort spent on locating the desired content through the extensive use of keyboard shortcuts. This gets even worse if users have limited shortcut vocabulary or are not familiar with the structure of a particular webpage. Augmenting screen readers with a speech input interface has the potential to alleviate the above limitations.

This paper describes the design, implementation, and evaluation of *Capti-Speak*, a speech-enabled screen reader for web browsing, capable of translating speech utterances into browsing actions, executing the actions, and providing audio feedback. The novelty of *Capti-Speak* is that it leverages a custom dialog model, designed exclusively for non-visual web access, for interpreting speech utterances. A user study with 20 blind subjects showed that Capti-Speak was significantly more usable and efficient compared to the regular screen reader, especially for ad-hoc browsing, searching, and navigating to the content of interest.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces And Presentation**]: User Interfaces – *interaction styles, natural language, voice i/o.*

General Terms

Experimentation, Human Factors, Languages.

Keywords

Speech Enabled Web Screen Reader, Web Automation, Spoken Dialog System, Dialog Act Model

1. INTRODUCTION

Blind people typically employ screen readers (e.g., VoiceOver [25], JAWS [10], etc.) to interact with computer applications, including web browsers. Screen readers serially narrate the content of the screen using synthetic voices, and enable users to navigate the content using keyboard shortcuts. This standard (keyboard) press-and-listen mode of interaction has several notable drawbacks [13]. Firstly, it can lead to excessive use of keyboard shortcuts for doing simple browsing tasks. Secondly, one has to remember an assortment of shortcuts and use a variety of browsing strategies. Thirdly, a serial narration of content may cause irrelevant content to be read out while browsing.

These problems become particularly acute while navigating content-rich web pages or doing online transactions spanning multiple pages. As a result, blind users typically end up spending too much time and effort on performing even simple day-to-day tasks such as online shopping, flight reservation, etc. The situation is much worse for novice users due to their limited vocabulary of shortcuts. They mainly use the Down/Up keys to navigate a page line by line, listening to most of the content along the way [4]. This was confirmed in our user study where we also noticed that, on the average, novice users used only five different shortcuts.

Missing and improper labels for page elements such as buttons and links can further exacerbate user frustration, and sometimes prevent users from completing the tasks. For example, during the study, we noticed that the participants struggled to locate the search box on the Amazon website due to a missing label; when the cursor was moved to the search box, the screen reader simply narrated "text box blank", thereby giving no indication that it was the search box. The participants had to first listen to the surrounding content, grasp the context, and then deduce that the text box was indeed the search box. In short, web browsing with screen readers requires a lot of manual effort, guesswork, and time, making the entire experience quite frustrating.

Now imagine that blind users could vocally express their intention to the screen reader, and the screen reader could automate these actions, enabling the user to search for or navigate to some *ad-hoc* content of interest on *any* web page (e.g., "go to search results", "next link", "top of the page", "move to search box", etc.), fill form fields, activate page controls such as links, buttons, and the like (e.g., "click on link about admission", "press the Cart button", or "select the third option"). Augmenting traditional screen reading with such capability could potentially overcome many of the aforementioned issues. Even if the speech interface was limited to just those kinds of actions: (i) users would not need to remember keyboard shortcuts, which is especially beneficial for novice users; e.g., users can simply say "next heading" if they do not remember the corresponding shortcut for moving the cursor to the next heading; (ii) they would not need to spend time and effort manually locating the desired content, which is especially beneficial when the content labels are missing or improperly labeled; this also helps to avoid listening to reams of irrelevant content; (iii) they would be able to complete any task on any website without having to use keyboard shortcuts.

As an illustration, consider the task of online shopping that involves several steps such as browsing through categories of products, searching for products, adding products to the cart, proceeding to checkout, and, finally, making a payment. With the traditional screen reader, the users have to remember and use

W4A '15, May 18 - 20, 2015, Florence, Italy

ACM 978-1-4503-3342-9/15/05…$15.00
http://dx.doi.org/10.1145/2745555.2746660

different kinds of shortcuts, listen through a lot of content, fill forms, and find relevant links and buttons that have to be activated to get through the required steps. Absent labels on certain page controls, such as the search box or checkout button, can make this task even more difficult. With a speech-augmented screen reader, however, the same task can be performed in a more intuitive and easier way by combining speech utterances with keyboard shortcuts as illustrated in Figure 1.

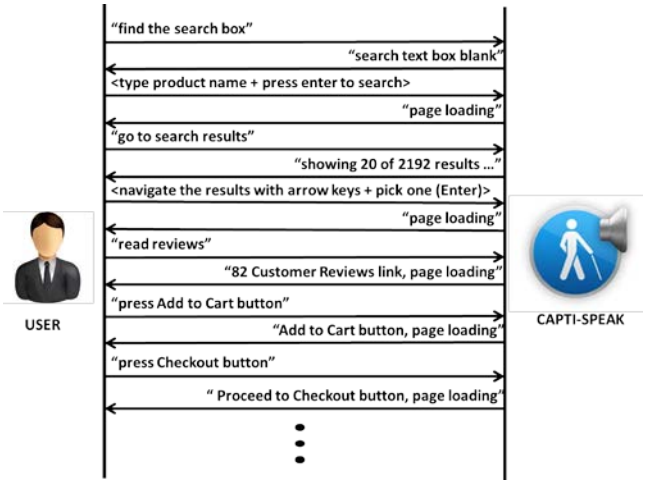

Figure 1. Illustration of how a combination of speech utterances (shown within quotes "") and keyboard shortcuts and text entry (shown within angular brackets < >) are used for online shopping with Capti-Speak. User input flow from left to right and Capti-Speak responses from right to left.

We would like to note that commercial speech-based applications have been around for a while and new ones continue to emerge at a rapid pace; however, these are mainly stand-alone (e.g., Apple's Siri) domain specific systems that are not connected to web browsers, which precludes dialog-based interaction with the Web. Moreover, spoken input modules integrated with web browsers are limited to certain specific functionalities such as search (e.g., Google's voice search) or are used as a measure of last resort (e.g., Siri searching for terms online).

This paper describes **Capti-Speak**, a speech-augmented Screen Reader (Capti-SR), developed using the Capti Narrator platform [20]. Capti-Speak translates spoken utterances into browsing actions and generates appropriate TTS responses to these utterances. The novelty of Capti-Speak from a web browsing perspective is that it regards each spoken utterance as part of an ongoing dialog. Specifically, Capti-Speak employs a custom dialog-act model [3] that was developed exclusively for "speech-enabled non-visual web access" to interpret every spoken utterance in the context of the most recent state of the dialog, where the state, in some sense, encodes the history of previous user utterances and system responses.

We conducted a user study with 20 blind participants to evaluate Capti-Speak. The study showed that, despite the presence (30%) of speech-recognition errors by Automatic Speech Recognizer (ASR), Capti-Speak was still able to interpret and execute most of the user utterances correctly. The participants rated Capti-Speak to be significantly more usable and efficient compared to the conventional keyboard-operated screen reader, especially for simple tasks involving ad-hoc browsing and searching.

2. RELATED WORK

Over the past decade, researchers and engineers have developed a number of tools and techniques for interpreting human speech and controlling devices via voice. There are now a number of natural language interfaces translating user utterances or commands to intended actions. In this section, we review the state of the art in natural language command interfaces and relate them to non-visual web browsing and Capti-Speak capabilities.

Several commercial voice-activated personal assistants are now available on mobile phones (e.g., Apple's Siri [22], Google Now [6], Microsoft's Cortana [17], etc.). In addition to searching for restaurants, movies, etc., on the Web, these assistants are capable of satisfying simple requests associated with routine tasks such as reminders, alarms, appointments, Facebook/Twitter updates, notes, etc. They also have a limited ability to provide answers to factoid questions posed by the users. Although these assistants can be very handy for people with vision impairments, none of these tools can work within a web browser. And additionally, they are all limited to simple predefined scenarios and tasks.

CoCo [12] is a web automation assistant that offers a conversational interface capable of accepting user utterances in the form of (restricted) natural language commands in order to perform various tasks on the Web. Specifically, CoCo [12] maps natural language commands to macros stored in Co-Scripter Wiki [14] and Co-Scripter Reusable History (ActionShot [15]). Besides a number of problems that arise in using web macros in practice [20], CoCo does not have an actual speech interface. Furthermore, if a blind user wants to enter natural language commands by hand, CoCo is not accessible with a screen reader.

Talking to web browsers is not an entirely new idea. For example, House et al. [8] proposed a modified version of the NCSA Mosaic system [1] that was capable of translating spoken natural language commands to basic browsing actions such as opening URLs, new windows, etc. However, this system [8] only supports commands for interacting with various browser handles/controls such as the URL bar, Menu bar, etc.; it does not support commands for interacting with webpage content. Capti-Speak, on the other hand, supports utterances/commands for interacting with webpage content itself: navigating to an object of interest, searching for an object on the current web page, activating various buttons and links, filling form fields, etc.

Voice browsers like PublicVoiceXML [19] and JVoiceXML [21] have an interactive voice interface for browsing web content. However, browsing the Web with these voice browsers requires conversion of web pages to VoiceXML [26], a document format that operates within a controlled domain.

In some cases, voice navigation is used for improving one particular aspect of browsing, e.g., in [7], the focus is on making menus and submenus voice-accessible; Windows Speech Recognition (WSR) [16] makes it possible to follow a link by speaking its ordinal number and enables a few other basic commands. Alas, neither is accessible to blind users.

An Android accessibility service, JustSpeak [27], can be used with any Android app or accessibility service, and is able to process chains of commands in a single utterance. However, JustSpeak is limited to a few basic browsing-related commands, specifically: activate, scroll, toggle switch, long press, toggle checkbox. Dragon NaturallySpeaking Rich Internet Application feature [18] enables the user to control certain websites by voice. Yet, this system only provides limited support to select parts of

only four (4) websites in specific browsers, and lists many additional caveats and limitations, both general and browser specific. In addition, usage of visual cues and a graphical user interface for listing possible utterances/commands significantly reduces the accessibility of Dragon software for blind people.

To summarize, the existing voice-enabled systems are either limited to a narrow domain that does not enable web browsing, or they support a very limited set of basic browsing actions. In this paper, we present and evaluate Capti-Speak, a speech-augmented screen-reader for voice-activated web browsing.

3. CAPTI-SPEAK ARCHITECTURE

Capti-Speak was implemented as an extension to the Capti Screen-Reader (Capti-SR) feature in Capti Narrator [20]. In addition to standard screen-reading shortcuts, Capti-Speak provides a special keyboard shortcut to activate/deactivate the speech interface; in the first prototype, the user needs to press this shortcut before and after speaking. As shown in Figure 1, the user can interact with Capti-Speak in two ways: (i) by pressing standard screen-reader shortcuts, and/or (ii) by vocally expressing his intention as a free-form natural language speech utterance.

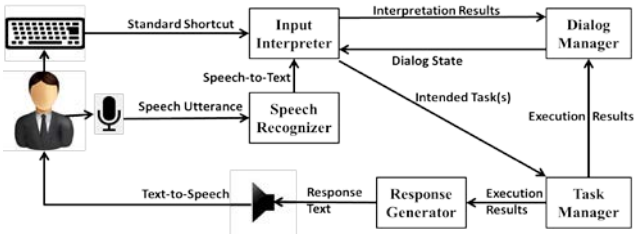

Figure 2. Capti-Speak Information Flow Diagram

The user input (shortcuts/utterances) is processed by the Input Interpreter to determine the intended task. In the case of a standard shortcut, Capti-Speak behaves like any other Screen Reader, and hence the interpretation is straightforward; the Input Interpreter consults a pre-defined shortcut-task knowledgebase (provided by Capti-SR) in order to determine the intended task. In the case of a speech utterance, Capti-Speak first converts the utterance to text using the Speech Recognizer module and passes the recognized text on to the Input Interpreter. From the text, the Interpreter extracts information describing the corresponding intended task: action to be performed and the target object.

The Interpreter determines the desired browsing action (e.g., cursor navigation, object invocation such as link select, button press, etc.) by understanding the high-level communicative intention of the user utterance. In this regard, the Interpreter leverages the dialog state information (maintained by the Dialog Manager) to map the utterance to one of the many pre-defined *Dialog Acts (DAs)*, where each dialog act represents some high-level communicative intent. To identify the target object, the Interpreter employs a custom defined set of rules that are employed for extracting the words that describe the target object in the text of the spoken utterance.

The Interpreter then instructs the Task Manager to perform the action(s) corresponding to the intended task. Depending on the task performed, the Response Generator constructs an appropriate response, which is finally narrated to the user by the Text-to-Speech synthesizing module of Capti-Speak. The implementation details of each of these components are presented next.

3.1 Automatic Speech Recognizer

We enlisted the assistance of publicly available Google ASR web service [9] for speech-to-text conversion. The received text from the Google web service is passed on to the Interpreter for interpretation. We observed latency of about 1-2 seconds between the transfer of the recorded speech command to the Google ASR web service and the receipt of the text response.

3.2 Input Interpreter

The Interpreter is responsible for processing both standard screen-reader shortcuts and natural-language speech utterances. As mentioned before, the Input Interpreter uses a custom knowledgebase of shortcut-task mappings available in Capti-SR [20] to interpret standard screen-reader shortcuts. To interpret speech utterances, the Interpreter utilizes the following two sub-components: a Dialog-Act Identifier and a Target Extractor.

3.2.1 Dialog Act Identifier/Recognizer

In the area of spoken dialog systems, the ability to label any speech utterance with a functional tag (the utterance's dialog act which represents the communicative intentions behind each utterance), is acknowledged to be a useful and important preliminary step in dialog processing and deep dialog understanding [23]. Such annotation, or tagging, leverages the user context in semantic interpretation of user utterances, and, hence, can be useful for assisting Capti-Speak in choosing the right task to execute and, subsequently, in producing an appropriate response. For example, imagine that the user is filling out a form, and the cursor is currently at the "Phone Number" field. If the user utters "Skip", then the trained Dialog Act (DA) Recognizer will assign the tag that represents cursor navigation instead of the tag that represents form-field filling.

In our work, we used the decision-tree-based DA Recognizer [3] that we developed using the dialog corpus collected in a Wizard-of-Oz study [2]; the WOZ study was an examination into how blind people used speech utterances for web browsing. The DA Recognizer is based on an annotation scheme that is exclusively designed for non-visual web browsing and comprises the following two groups of tags:

i. *Local*: Command-Task, Command-Navigation, Information-Task, and
ii. *Global*: Command-Intention, Command-Multiple, Question-Task, Help-Task, and Self-Talk.

The *local* dialog acts represent all simple utterances that request some basic actions (e.g., *click*, *select*, *go to*, etc.) to be performed on the currently open web page. On the other hand, the *global* dialog acts represent those user utterances that require a generation of complex execution plans containing a sequence of basic actions. Additionally, the execution of high-level utterances may span multiple web pages. For example, the high-level utterance "Buy this product" involves the following sequence of steps – press the *add to cart* button, press the *proceed to checkout* button, fill the *personal information* form, fill the *enter card details* form, and, finally, press the *place order* button. For this paper, we supported only the simple speech utterances, belonging to the following subset of the aforementioned *local* dialog acts:

- Command-Task – utterances for basic actions, such as *click*, *select*, *enter*, *submit, press, choose,* etc.; e.g., "*Click the xyz link*", "*Press the abc button*", "*I would like to choose the second option*", etc.

- Command-Navigation – utterances for directing the movement of the virtual cursor, such as *go to*, *stop*, *next*, *continue*, *skip*, *move*, etc.; e.g., "*Go to search box*", "*please find ABC link*", "*previous link*", "*last heading*", "*I want to go back to the previous item*", "*move to top of the page*", etc.
- Information-Task – utterances for filling form fields, e.g., "*John"* for the first name field, "*Last name Doe"* for the last name field, "*Fourth November"* for the date field, etc.

All utterances belonging to the remaining unsupported global tags (Command-Intention, Command-Multiple, etc.) were categorized under the label "Other".

The DA recognizer uses a wide variety of features to identify the associated dialog act of a user utterance. The feature set includes, among other features, syntactic features (e.g., parse tree production rules), contextual features (e.g., previous user DA, previous system DA, current cursor location, etc.), and lexical features (e.g., unigrams, presence of certain keywords, etc.). Contextual features were mostly obtained from dialog state variables (see section 3.4) stored by the Dialog Manager and the syntactic features were obtained using the Stanford Parser [11]. A detailed description of the entire feature set can be found in [3].

3.2.2 Target Extractor

Once the high-level action (cursor navigation, object invocation, and form-field filling) of the intended task is identified by determining the dialog act, the next step is to find the "target" object involved in that task. The target object can be a link, a button, or simply a text node in the current web page. Therefore, the Target Extractor obtains the following information, if available, from the text of every speech utterance and provides them to the Task Manager:

- The **type** of the target object (e.g., link, button, etc.). Knowledge of the object type aids in improving accuracy of locating the desired object.
- The **descriptors** associated with and describing the target object. For example, in the case of the utterance "Click on the admissions link", the object type is *link* and the associated descriptor is *admissions*.

We used the Stanford Part-of-Speech tagger [24] to assist this task of target information extraction. The tagger assigns a Part-of-Speech (POS) tag (e.g. "NN" for singular proper noun, "JJ" for adjective, "VB" for verb, etc.) to each word in the utterance.

Extracting object type: The analysis of the non-visual spoken dialog corpus obtained in the prior study [3] revealed that, in every utterance, the object type is always a singular common noun with the "NN" POS tag. Therefore, to determine the object type, the extractor first extracts all the words with "NN" tag from the utterance and then checks if any of them belong to a predefined set of keywords (e.g., "link", "button", etc.) representing the various object types. The constraint regarding "NN" tag is important since certain keywords (e.g., "link") can sometimes appear as verbs in an utterance. In case more than one keyword (with NN tag) are present in the utterance, the object type is based on the keyword closest to the end of the utterance - this rule was based on our analysis of the utterance corpus from the earlier Wizard-of-Oz study [2].

Extracting object descriptors: The descriptors are extracted by removing target-unrelated words from the text of the utterance. Specifically, the following words, if present, are discarded:

- The verb (VB) indicating the intended task (e.g., click, press, go, etc.) at the beginning of the utterance, along with the immediately following prepositions (IN) and determiners (DT), if any (e.g., "click on the…").
- The words preceding the verb (VB) indicating the intended task, if any (e.g., "I want to press…").
- The singular noun (NN) indicating the object type, along with immediately following prepositions (IN) and determiners (DT), if any, (e.g., "link about the…").

The remaining words, i.e. the words that are not discarded, are considered to be the descriptors. For example, consider the utterance "I would to like to select the cart button". For this utterance, the following words are removed: (i) the task verb "select", along with preceding words "I", "would", "like", "to", the determiner "the" immediately following "select", and lastly, the object type "button". Therefore, the target object descriptor for this example is the word "cart". The descriptors can also be *relative* (e.g., "next", "previous", "top", "last", etc.) to the current screen-reader cursor location. Note that there can only be one relative descriptor, if any, in utterances; this was confirmed in the earlier Wizard-of-Oz study [2].

The Input Interpreter then supplies the required information (the utterance's dialog act, target object type, and target object descriptors) to the Task Manager for execution.

3.3 Task Manager

The Task Manager is responsible for performing the intended task – specified action(s) such as navigation, invocation (e.g., clicking, pressing, option select for checkboxes and radio buttons, form submit, etc.), and form-field filling on the specified target object. Recall that we represent a task as a pair <action, target object>. The Task Manger can determine the action to be performed directly from the utterance's dialog act. In order to locate the target object, the Task Manager uses the target information (type and descriptors) and other contextual details (e.g., the dialog act, current screen-reader cursor location, object type where the cursor is, etc.) to search for a matching object (or a node) in the DOM tree of the currently opened webpage. A node in the DOM tree is considered a match if the following conditions are satisfied:

- The object type of node is the same as that of the target, if available.
- The action indicated by the dialog act can be performed on the node.
- If the descriptor is relative (e.g., "next", "previous", "top", "bottom", "first", "second", etc.), the node's placement in the DOM tree corresponds to the descriptor, relative to the current screen-reader cursor position.
- If the descriptors are absolute (e.g., "Proceed", "to", "Checkout", "Attack", "in", "Syria", etc.), at least one descriptor is present either in the textual content of the node or in the values of one of the following node attributes: *class*, *id*, *value,* or *type*.

The Task Manager then assigns a score (the number of the target descriptors present in either that node's content or attribute values) to each of the matching nodes or candidates. Finally, the Task Manager performs the intended action (e.g., move the cursor) on the candidate with the highest score; in case of a tie, the Task Manager picks the one that it encountered first when traversing the DOM in a depth-first-search manner.

Notice that the target-locating scheme described above offers some resilience against speech recognition errors; even if one or two descriptors are incorrect (or missing), the Task Manager is still able to find the desired target in the DOM tree. For example, during the user study, we noticed that even when one of the descriptors "Proceed" in the utterance "Press Proceed to Checkout button" was incorrectly recognized as "Process", the DOM node corresponding to the "Proceed to Checkout" button still got the highest score among matching candidates, and therefore the Task Manager was able to perform the desired action of pressing the "Proceed to Checkout" button. More examples are provided in subsequent sections.

To further mitigate the effects of speech recognition errors, and promote reliability, we perform *approximate searches* while checking for the presence of descriptors in a node's text (or attribute values). Specifically, a descriptor is assumed to be present in a node's text (or attribute values) if the normalized Levenshtein distance between that descriptor and one of the words (in either the node's text or attributes values) is less than 0.3 (the maximum error seen in the Wizard-of-Oz study corpus [2]). The benefit of using this approximate search can be illustrated as follows. Once again, consider a scenario where the utterance "Press Proceed to Checkout button" is incorrectly recognized as "Press Process to Checkout button". In this case, the error ("Proceed" recognized as "Process") will not affect the outcome, since the approximate search will consider the words "Process" and "Proceed" to be equivalent (normalized Levenshtein distance between "Process" and "Proceed" is 0.28 which is less than the chosen threshold of 0.3).

Finally, if the search is unsuccessful, the Task Manager sends a *failure* notice to the Response Generator. If the dialog act of the user utterance is "Other", the Task Manager by default sends a failure notice along with "unsupported utterance" message to the Response Generator.

3.4 Dialog Manager

The Dialog Manager maintains the dialog state information associated with the dialog. This state information includes user interaction history containing previous utterances, dialog acts, shortcut presses, target objects, Capti-Speak actions, cursor locations, etc. The Dialog Manager assists interpretation of an utterance by providing the state information to the Interpreter. For example, the dialog act classifier in the Interpreter uses the prior dialog acts as features for accurately identifying the current dialog act. Also, for some utterances such as "next", the state provides the context to determine the target object, e.g., whether it is the next link, button, search result, etc. Both the Interpreter and the Task Manager report their outcomes to the Dialog Manager.

3.5 Response Generator

As explained earlier, the Response Generator produces appropriate text responses based on execution results received from the Task Manager, and forwards them to the Text-to-Speech (TTS) module for narration. We used the Microsoft Windows built-in TTS with voice "Anna" speaking at the "Normal" speech rate, which is about 120 words per minute.

In case of failures, the Response Generator generates a "Please rephrase your command" text message. In case of successful executions of tasks involving just the navigation of the screen-reader cursor (Command-Navigation dialog act), the Response Generator narrates the text of the target object and, additionally, the object type of the target if it is not a text node. For example, a successful execution of "Go to admissions" utterance may result in the Response Generator producing a "*Graduate admissions link*" text message. In the case of invocation tasks (Command-Task dialog act) where an actionable object, e.g., link, button, etc., is pressed or selected, the Response Generator initially generates a "<target text> <target type>" message (e.g., "Add to Cart button") followed by an optional "page loading" message if the browser starts loading a new page. Each generated text message is forwarded to the Text-to-Speech (TTS) module for narration.

3.6 Illustration: Putting It All Together

Consider a scenario where the user utters "Press the Proceed to Checkout button" after adding an item to the shopping cart on Amazon.com. In order to process this speech utterance, Capti-Speak performs the following sequence of operations:

i) Convert the speech utterance to text;

ii) Determine the dialog act and hence the action to be performed ("Command-Task" - Invocation);

iii) Determine the target object type ("button");

iv) Extract target descriptors ("Proceed", "to", "checkout");

v) Find matching candidates in the DOM tree and compute their scores ("Proceed to Checkout" button – Score 3, "Add to Cart" button – Score 1);

vi) Perform the intended action on the candidate with the highest score (Invoke or press the "Proceed to Checkout" button); and

vii) Narrate an appropriate response, using TTS ("Proceed to Checkout button, page loading").

4. USER STUDY WITH CAPTI-SPEAK

We conducted a user study to evaluate and validate the usefulness and utility of Capti-Speak for assistive web browsing. Specifically, the study aimed to confirm the following hypotheses:

- **H1**: The users are more efficient, i.e. complete tasks faster, with Capti-Speak than with a regular screen reader.
- **H2:** Capti-Speak is more usable than a regular screen reader; web browsing is easier and more intuitive with Capti-Speak than with a regular screen reader.

In this section, we describe the setup of the user study.

4.1 Study Setup

In the study, the users were asked to perform the following tasks:

- *Shopping* – purchase any product on amazon.com, starting from the home page and ending with *Login to Checkout* page.
- *Admissions* – locate the graduate admissions application on the Stanford University website and fill out the first basic form starting from the home page and ending with the "submit" button on the application page.
- *Advertisement* – find an advertisement for a piano in the price range of $100-$500 on craigslist.org, starting from the home page and ending with the opening of one of the listings that meets the price range requirement.
- *Email* – send an email to a friend on gmail.com, starting from the inbox page and ending with pressing the "send" button.

These tasks were identified as representative and "useful" for people with vision impairments by our web accessibility

consultants. The tasks were also selected to be of comparable difficulty. For each participant, the tasks were randomly split among the following two conditions, with two tasks per condition:

1. *Capti-SR:* Standard Shortcuts only – the users had to complete the assigned tasks using only the standard screen-reader shortcuts such as those available in JAWS [10].

2. *Capti-Speak:* Speech Utterances & Standard Shortcuts – the users were allowed to use both speech utterances and standard keyboard shortcuts with no restrictions; a user could potentially ignore either of the two input modalities completely.

Capti-SR (a feature in Capti Narrator [20]) supports all the standard screen-reading shortcuts; no attempts were made by the participants to use unsupported shortcuts during the study. The participants were given sufficient time to practice and become comfortable with Capti-SR.

We did not create a separate condition exclusively for speech utterances because our goal was not to replace the existing screen reader per se, but to augment its current functionality by providing speech-input support. Therefore, while browsing with Capti-Speak, users would still be able to use the standard screen-reading functionalities that they are familiar with.

Before doing the study tasks, the participants had to perform the following practice tasks to get acquainted with the user interface:

- Read the latest tweet from a popular celebrity on twitter.com.
- Read news articles on different topics on abcnews.com.

The time limit to complete each task was set to 10 minutes. In case a participant was unable to complete any task within 10 minutes, the experimenter was instructed to stop that task and record the reason for failure; however, no such event occurred during the experiment.

4.2 Recruitment and Additional Details

We recruited 20 visually impaired participants for our user study. Gender representation was equal (10 men and 10 women). All ages were represented, but the average age of the participants was 47 years (std.dev 16.4, median 50). This age group represents the typical age of blind screen-reader users, as many people lose sight in later stages of life. The racial composition for this study was approximately 50% African-American, 45% Caucasian, and 5% Hispanic. Exactly 40% of the participants indicated that they were very comfortable with screen readers – we will refer to them as "experts", while the remaining 60% said they were not comfortable with computers – beginners. The experiment was conducted on a Windows 7 laptop with Firefox web browser.

4.3 Procedure

The experimenter first introduced Capti-Speak to the participants and explained how Capti-Speak was different from a regular screen reader. The experimenter then explained to the participants that the goal of the study was to evaluate the effectiveness and usability of Capti-Speak. The participants had a keyboard and a microphone placed in front of them, while the experimenter was facing the screen to visually observe the progress in the browser.

Since speech-utterance based browsing was a new concept to all participants, the experimenter provided a demonstration at the beginning of the experiment by performing a simple task using a wide range of speech utterances. The demo task involved searching and reading news articles on various topics on the CNN website; the utterances included various actions (e.g., clicking on an article link, navigating to the sports section, etc.).

The participants were allotted a sufficient amount of time (approximately 10-15 minutes) to get acquainted with the interfaces for each condition (i.e. Capti-SR and Capti-Speak). In each of these two practice sessions, the experimenter assisted the participants in completing the sample tasks, and clarified all their doubts. During practice, the experimenter clearly instructed the participants to avoid using speech utterances belonging to the unsupported *global* dialog acts by providing various examples of such utterances.

The order of conditions and tasks was randomized for each participant to minimize the impact of the learning effect. At the end of the experiment, the participants were asked to complete a set of questionnaires so that we could collect subjective feedback and compare the usability of the two conditions used in the study. These questions included the standard System Usability Scale (SUS) [5], a modified SUS [20] allowing us to compare subjects' opinions regarding the two conditions, and open-ended questions prompting the participants for comments and suggestions. We next present the results of the user study.

5. RESULTS

In this section, we present our findings, analyze the results, and discuss their implications. We also compare and contrast our results with some of the findings of an earlier Wizard-of-Oz (WOZ) user study [2] that required participants to perform similar browsing tasks, using speech utterances that were interpreted and executed by an experimenter (wizard) instead of an actual system. The other important aspect distinguishing the WOZ study from the study described in this paper was that the WOZ study allowed the participants to use speech utterances belonging to both *global* and *local* dialog acts, whereas this study restricted the participants to using only those speech utterances that belonged to the *local* dialog acts. Notably, there was no overlap between the set of participants in this study and the set of WOZ study participants.

5.1 Objective Analysis

In order to validate the hypothesis that the users can complete tasks faster with Capti-Speak than with a regular screen reader, we compared the task-execution times of Capti-Speak with that of Capti screen reader (Capti-SR). The summary of these results for each group is presented in Table 1.

Table 1. Task completion time statistics (in seconds) (faster and significant results ($p < 0.05$) are in bold)

Group	Capti-SR	Capti-Speak	Diff. in timings significance
Overall (20 subjects)	μ: 284.3 σ: 151.8	μ: **191.9** σ: 119.3	p = **0.0002**
Age < 50 (10 subjects)	μ: 256.8 σ: 147.5	μ: **197.0** σ: 137.0	p = **0.0464**
Age ≥ 50 (10 subjects)	μ: 311.8 σ: 151.1	μ: **186.9** σ: 098.4	p = **0.0025**
Males (10 subjects)	μ: 250.1 σ: 139.2	μ: **186.2** σ: 097.6	p = 0.0572
Females (10 subjects)	μ: 318.5 σ: 156.2	μ: **197.7** σ: 137.5	p = **0.0017**
Beginners (12 subjects)	μ: 303.0 σ: 152.4	μ: **210.6** σ: 118.4	p = **0.0069**
Experts (8 subjects)	μ: 256.2 σ: 140.8	μ: **163.9** σ: 115.2	p = **0.0188**

As can be seen from Table 1, overall and for all identified groups, using Capti-Speak was faster than using the screen reader (Capti-SR). In addition, a t-test for statistical significance revealed that the differences in task-completion times between Capti-Speak and Capti-SR were statistically significant for all identified groups in Table 1, except for the group *Males* ($p = 0.057$). This differs from the corresponding results in the WOZ study [2], where no significant differences were observed between completion times in these conditions for all identified groups. This discrepancy can be attributed to the observation that, compared to the participants in this study, the participants in the WOZ study spent a significant amount of time exploring the interface: randomly browsing websites using unrestricted (*local* and *global*) speech utterances before completing the assigned task. In other words, the freedom to use unrestricted speech utterances, coupled with the perfect interpretation/execution of these utterances by a human wizard, *distracted* the participants in the WOZ study. On the contrary, the participants in this study were restricted to use only *local* speech utterances and, hence, they were more *focused* on the task at hand.

Table 2. Statistical significance of t-test assuming unequal variances between groups for each condition

Groups	Capti-SR	Capti-Speak
Age < 50 vs. Age ≥ 50	$p = 0.263$	$p = 0.795$
Males vs. Females	$p = 0.162$	$p = 0.768$
Beginners vs. Experts	$p = 0.344$	$p = 0.235$

Also, with respect to task completion times, there were no significant differences between different group pairs (Table 2) for either Capti-Speak or Capti-SR, which is in agreement with the corresponding results of the WOZ study.

It is also notable that the execution of each speech utterance in Capti-Speak involved an additional overhead of 1-2 seconds per utterance, since the speech recognition was performed using a Google web service. In spite of this overhead, browsing with Capti-Speak was still significantly more efficient than browsing with Capti-SR, thereby validating the hypothesis **H1**.

The limited knowledge of screen-reader shortcut vocabulary was one of the primary reasons why the users took significantly more time to complete tasks with Capti-SR than with Capti-Speak. On average, the beginners used only five different shortcuts (popular ones included the arrow keys, TAB, SPACEBAR, and ENTER), whereas the experts used ten different shortcuts (popular ones included arrow keys, TAB, SPACEBAR, ENTER, CTRL, SHIFT, H – heading, A – link, E – text box, and B – button). Still, all participants spent a considerable amount of time listening to irrelevant content while trying to find the web page controls and information necessary to complete the tasks.

Another reason why the users could complete tasks faster with Capti-Speak than with Capti-SR was that some of the important task-related objects were improperly labeled, and therefore it was time-consuming for the users to locate those objects using only screen-reader shortcuts provided by Capti-SR. For example, for the Shopping task involving the amazon website, with Capti-SR, the participants spent a considerable amount of time locating the *search box*, since the appropriate label was missing (at the time when this study was conducted). The screen reader simply narrated "textbox blank" when the cursor was on the search box, thereby giving no indication that it was the search box. The participants had to first listen to the surrounding content to deduce that the text box was indeed the search box. On the other hand, with Capti-Speak, no such extra effort was required from the participants; most of the participants issued simple speech utterances such as "Go to search box" and Capti-Speak undertook the responsibility of locating the search box, thereby saving a considerable amount of time. Specifically, the descriptor-based approximate-search feature of Capti-Speak was able to quickly and easily locate the search box, since the "id" attribute of the DOM node corresponding to the search box contained the descriptor "search".

On a side note, the participants were significantly more efficient with Capti-Speak even in the *Admissions* task where all content was properly labeled. This was because most participants struggled to navigate to the "admissions form" page from the home page using just keyboard shortcuts.

Also with Capti-Speak, almost all participants started off with a speech utterance, which was typically a navigation utterance aimed at locating task-related links/buttons or a search input box. We noticed that this behavior was another reason why participants were more efficient with Capti-Speak than with Capti-SR. For example, for the 'Admissions' task on the Stanford University website, we observed that, with Capti-SR, most participants, and especially beginners, initially spent a significant amount of time navigating the home page trying to locate the 'Admissions' link. With Capti-Speak however, the participants first tried to find out if there was a link related to admissions by issuing an utterance such as "Go to admissions link", thereby saving time.

Although every participant reported having had experience with speech-to-text dictation, especially with Apple's Siri and Nuance's Dragon, no participant chose to use speech utterances for form filling, even though Capti-Speak was capable of processing such utterances. This makes sense since anyone who had experience even with the best ASR knows that it often fails to correctly recognize incomplete sentences and proper nouns such as names, addresses, etc. The keyboard interface, on the other hand, is much more reliable for data entry.

With Capti-Speak, the participants mostly used the keyboard when they had to type something (e.g., form fields) or when they knew how to navigate a simple list of links, paragraphs, etc. When asked why they used the keyboard, they said that (i) the keyboard was more accurate and reliable for filling form fields, and (ii) for basic back and forth navigation, keyboard shortcuts such as "DOWN" key, "A" key were sufficient and, additionally, more efficient than speech utterances such as "next link", "next", etc.

5.2 System Usability Scale Analysis

In order to validate the **H2** hypothesis, namely that Capti-Speak was more usable than a regular screen reader, we administered a System Usability Scale (SUS) [5] questionnaire for both Capti-Speak and Capti-SR at the end of the study. The SUS questionnaire asked the participants to rate positive and negative statements about the system on a Likert scale from 1 – strongly disagree to 5 – strongly agree, with 3 – neutral.

The summary of the overall scores, as well as the scores for different groups are presented in Table 3. As seen from the table, overall and for all identified groups, Capti-Speak was rated much higher than Capti-SR. In addition, the variation in the participants' scores for Capti-Speak was much lower than that for Capti-SR. These results are in agreement with the corresponding findings of the WOZ study [2], where the 'Combination' condition, which supported both speech utterances and screen-reader shortcuts, received much higher usability scores than the 'Keyboard' condition which supported only screen-reader

shortcuts, while having a low variation in the participants' answers. Also, as can be seen in Table 3, a t-test for statistical significance revealed that the difference in usability scores between Capti-Speak and Capti-SR was statistically significant ($p < 0.05$) for all examined groups, including the group of experts.

Table 3. SUS score statistics (higher and statistically significant results ($p < 0.05$) in bold)

Groups	Capti-SR	Capti-Speak	Diff. in scores significance
Overall (20 subjects)	μ: 47.0 σ: 24.4	μ: **83.5** σ: 11.9	p < **0.001**
Age < 50 (10 subjects)	μ: 50.0 σ: 19.9	μ: **81.2** σ: 11.8	p = **0.002**
Age ≥ 50 (10 subjects)	μ: 44.0 σ: 27.8	μ: **85.7** σ: 11.5	p = **0.002**
Males (10 subjects)	μ: 49.5 σ: 32.6	μ: **88.7** σ: 13.3	p = **0.010**
Females (10 subjects)	μ: 44.5 σ: 10.7	μ: **78.2** σ: 07.1	p < **0.001**
Beginners (12 subjects)	μ: 37.0 σ: 19.7	μ: **81.2** σ: 11.3	p < **0.001**
Experts (8 subjects)	μ: 61.8 σ: 23.1	μ: **86.8** σ: 11.9	p = **0.018**

It should be noted that the scores for the 'Combination' condition (that supported both speech utterances and screen-reader shortcuts) in the WOZ study [2] were considerably lower than those obtained in the present study. This difference can be attributed to the presence of a third condition "Voice" in the WOZ study, where the participants had to complete tasks "hands-free" using unrestricted speech utterances. In that study, the participants clearly preferred the Voice condition to all other conditions, which resulted in the Combination condition receiving lower scores despite 100% accuracy of speech recognition.

Also, as shown in Table 4, there were no significant differences in the SUS scores for Capti-Speak between various group pairs (age, gender, and experience). However, it is notable that the male group (not correlated with age or experience groups) gave comparatively higher scores to Capti-Speak, than the female group; the difference is almost statistically significant. In the case of Capti-SR, as expected, the scores from expert participants were much higher than those from the beginner participants – the difference was found to be statistically significant (Table 4).

Table 4. Statistical significance t-test assuming unequal variances between groups for different conditions (significant values ($p < 0.05$) are in bold)

Groups	Capti-SR	Capti-Speak
Age < 50 vs. Age ≥ 50	p = 0.606	p = 0.425
Males vs. Females	p = 0.670	p = 0.056
Beginners vs. Experts	**p = 0.035**	p = 0.337

Overall, it is clear from the SUS scores that the participants indeed found the multimodal interface of Capti-Speak, which supported both speech utterances and screen-reader shortcuts, to be much more convenient to use than the traditional keyboard-controlled screen-reading interface.

To test if some of the above, as well as subsequent, observations are a consequence of the correlation between different groups of participants, we conducted Fisher's exact test on contingency tables of these aspects and found no statistically significant correlation between age, experience, or gender of participants.

5.3 Modified SUS and the final Questionnaire

Questions in Table 5 asked the participants to choose the better or the worse condition, with positive and negative questions, which were a simple modification of SUS questions [20]. For each question, the user had to choose one of the two conditions or select "none of these" or "both of these". It can be seen in Table 5 that, overall, the participants preferred Capti-Speak over Capti-SR for all the questions. A few participants who preferred Capti-SR said that they did so because they were skeptical about the speech recognition in Capti-Speak being consistently accurate. Additionally, the participants who answered either "None of these" or "Both of these" for some of the questions in Table 5 were mostly expert users. Surprisingly, among the 6 participants who preferred Capti-SR over Capti-Speak in at least one question, only 2 were expert users, thereby indicating that the majority of participants (75%) in both the expert and beginner groups preferred Capti-Speak.

Table 5. Modified SUS statistics (expressed as % of users) Winning condition for each question is marked in bold.

Question	Capti-SR	Capti-Speak	None	Both
Which system do you prefer to use?	10%	**90%**	0%	0%
Which system did you find the most complex?	70%	**10%**	20%	0%
Which system did you find the easiest to use?	15%	**85%**	0%	0%
With which system would you need the most help from a technical person?	70%	**20%**	10%	0%
Which system was the most well-integrated?	15%	**75%**	5%	5%
Which system was the least consistent?	70%	**15%**	15%	0%
Which system do you think people will learn to use the quickest?	5%	**90%**	5%	0%
Which system was the most cumbersome?	75%	**15%**	10%	0%
Which system did you feel the most confident using?	15%	**80%**	0%	5%
Which system required you to learn the most before you could use it?	65%	**25%**	10%	0%

Finally, at the end of the study, we asked the participants to rate a few statements (Table 6) to obtain subjective feedback on their experience with the speech interface of Capti-Speak.

Table 6. General statements about voice browsing (Likert Scale from 1 - Strongly Disagree to 5 - Strongly Agree)

Statement	Avg	Med	Dev
1. I would like to use speech utterances for performing tasks on the web	4.30	5	1.05
2. The accuracy of speech recognition affects my decision to use the speech interface of Capti-Speak	3.65	4	1.15
3. I accept the current speech recognition accuracy of Capti-Speak	3.90	4	0.94

As can be seen from Table 6, the participants were in strong agreement with the first statement about performing tasks with speech utterances (Median 5.0, Average 4.3). Their response to the last statement concerning the accuracy of speech recognition, however, was just "Agree" because some participants (mostly experts) wanted speech recognition to be more accurate and reliable. Indeed, speech recognition is prone to errors and therefore requires multiple dialog turns in case of inaccuracies. Finally, as expected, most participants agreed that their decision to use speech utterances was heavily influenced by speech recognition accuracy (a more detailed analysis on errors follows).

Given this and other results of the study, it is safe to conclude that Capti-Speak is i) desirable, (ii) usable, and iii) a productivity enhancer that helps people perform browsing tasks faster.

5.4 Dialog Act and Speech Recognition

The accuracy of Capti-Speak is directly dependent on the accuracy of the Dialog Act Recognizer (DA Recognizer) in the Input Interpreter. The accuracy of the DA Recognizer was measured by comparing the actual dialog acts of speech utterances to that predicted by the Interpreter. To obtain the actual dialog acts of speech utterances, we manually annotated the recorded speech utterance corpus using the previously described dialog-act annotation scheme (Sec. 3.2.1). The recognition accuracy is reported in Table 7. As an example, a total of 18 speech utterances were annotated as 'Command-Task' by the human expert, and our DA Recognizer was able to correctly identify/predict 14 of them, thereby producing a recall of 77.7%.

Table 7. Individual Dialog Act identification accuracy

	Number of Commands + Accuracy			
Dialog Act	**Total**	**Correct**	**Precision**	**Recall**
Command-Task	18	14	100%	77.7%
Command-Nav.	39	33	100%	84.6%
Info-Task	0	0	100%	0%
Other	4	4	28.5%	100%
Overall	61	51	82.1%	83.6%
Overall w/o Other	57	47	100%	82.4%

Notice in Table 7 that all dialog acts except 'Other' have an absolute precision of 100%. The low precision of 'Other' was primarily due to speech recognition errors that caused utterances of the type 'Command-Task' and 'Command-Navigation' to be incorrectly classified as 'Other'. Note, however, that none of 'Command-Task' utterances were incorrectly classified as 'Command-Navigation' and vice versa. Still, it can be observed in Table 7 that, overall, our DA Recognizer performs reasonably well with accuracies comparable to those reported in [3].

As can be seen in Table 7, a large portion (~64%) of all issued utterances belonged to the 'Command-Navigation' dialog act. For example, navigation-related utterances such as "Go to search box" or "Go to graduate admission" were popular among the participants. All speech utterances annotated as 'Other' belonged to the unsupported *global* dialog acts such as Command-Intention, Command-Task-Multiple, and Question-Task in [3]. Examples of the 'Other' category included "Add to Cart", "Find me King James Version of the bible", and "Compose an email".

As mentioned earlier, the performance of the DA Recognizer is dependent, to a considerable extent, on the accuracy of the ASR module. Table 8 presents detailed per-task speech recognition/ dialog-act recognition accuracy observed during this study.

Table 8. Accuracy of speech recognition (SR) and dialog act recognition (DA) for each task

	No. of utterances + Accuracy				
Task	**Total**	**Correct SR**	**Correct DA**	**Acc. SR**	**Acc. DA**
Shopping	20	15	18	0.75	0.90
Email	10	7	10	0.70	1.00
Admission	19	14	15	0.74	0.79
Advertisement	12	8	8	0.67	0.67
Overall	*61*	*44*	*51*	*0.72*	*0.84*

Recall that we used a third-party service for automatic speech recognition (Google Speech API), and therefore we did not have control over this component. However, notice that our Dialog Act Recognizer is more tolerant to speech recognition errors as evidenced by a higher overall DA classification accuracy (84%) compared to speech recognition accuracy (72%).

Prediction of correct dialog act in the presence of speech recognition errors proved to be useful in some cases. For example, the user utterance "Go to Stanford Admission Link" was incorrectly recognized by ASR as "Go to Stafford Admission Link", but correctly identified by DA Recognizer as belonging to the 'Command-Navigation' dialog act. Using the descriptor-based approximate search algorithm of the Interpreter, Capti-Speak was able to correctly navigate the cursor to the 'Admission' Link.

Recall from Table 6 that most participants accepted the current speech-recognition accuracy of Capti-Speak. Many participants pointed out that repeating an utterance in case of incorrect speech recognition was still better than using keyboard shortcuts to navigate to the desired content on web pages, especially the ones with a lot of advertisements and other irrelevant content. Given the fast improvements in speech technology, we expect Capti-Speak to become more usable and accurate in the near future.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented Capti-Speak, a speech-augmented Screen Reader for web browsing, capable of interpreting spoken utterances and performing browsing actions on behalf of the user. A study with 20 blind subjects demonstrated that Capti-Speak was able to interpret and execute spoken user utterances with fairly high accuracy. In addition, the study also showed that Capti-Speak was significantly more efficient and usable compared to the traditional keyboard-controlled screen reader.

In the future, we plan on extending Capti-Speak to support unrestricted speech utterances belonging to the *global* dialog acts that will require automated planning and dynamic generation of elaborate execution scripts. Also, we intend to support spoken dialog so that desirable features such as disambiguation and conversation-based error correction, can be offered to the users. We also plan to investigate how domain-specific knowledge and other contextual information can be leveraged to further improve interpretation accuracy, and hence make Capti-Speak more robust.

7. TESTIMONIALS

We received overwhelmingly positive responses about Capti-Speak; the following are some of the notable testimonials:
"It is like speaking your mind."
"A lot of stress is removed."
"Voice will help people with motor impairments."
"You couldn't have made it simpler."
"Good for people who are beginners and who cannot type well."
"It will be cool to use it on *FreshDirect* grocery website."
"I can use this while giving talks and presentation."

8. ACKNOWLEDGEMENTS

Research reported in this publication was supported by the National Eye Institute of the National Institutes of Health under award number 1R43EY21962-1A1. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. We also thank our accessibility expert, Glenn Dausch, for his invaluable feedback on the non-visual usability of Capti-Speak.

9. REFERENCES

[1] Andreessen, M., *NCSA Mosaic technical summary*. National Center for Supercomputing Applications, 1993. **605**.

[2] Ashok, V., Y. Borodin, S. Stoyanchev, Y. Puzis, and I.V. Ramakrishnan, *Wizard-of-Oz evaluation of speech-driven web browsing interface for people with vision impairments*, in *Proceedings of the 11th Web for All Conference*. 2014, ACM: Seoul, Korea. p. 1-9.

[3] Ashok, V., Y. Borodin, S. Stoyanchev, and I.V. Ramakrishnan, *Dialogue Act Modeling for Non-Visual Web Access*, in *to appear in the 15th Annual SIGdial Meeting on Discourse and Dialogue, SIGDIAL*. 2014: Philadelphia, PA, USA.

[4] Borodin, Y., J.P. Bigham, G. Dausch, and I.V. Ramakrishnan, *More than meets the eye: a survey of screen-reader browsing strategies*, in *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*. 2010, ACM: Raleigh, North Carolina. p. 1-10.

[5] Brooke, J., *SUS-A quick and dirty usability scale.* Usability evaluation in industry, 1996. **189**: p. 194.

[6] Google. *Google Now*. 2012; Available from: http://www.google.com/landing/now/#utm_source=google&utm_medium=sem&utm_campaign=GoogleNow.

[7] Han, S., G. Jung, M. Ryu, B.-U. Choi, and J. Cha. *A voice-controlled web browser to navigate hierarchical hidden menus of web pages in a smart-tv environment*. in *Proceedings of the companion publication of the 23rd international conference on World wide web companion*. 2014: International World Wide Web Conferences Steering Committee.

[8] House, D., D. Novick, M. Fanty, and J. Walpole, *Spoken-Language Access to Multimedia (SLAM): Masters Thesis.*

[9] J.A.R.V.I.S. *Java Speech API*. 2014; Available from: https://github.com/The-Shadow/java-speech-api.

[10] JAWS. *Screen reader from Freedom Scientific*. 2013 [cited 2013]; Available from: http://www.freedomscientific.com/products/fs/jaws-product-page.asp.

[11] Klein, D. and C.D. Manning., *Accurate Unlexicalized Parsing*, in *Proceedings of the 41st Meeting of the Association for Computational Linguistics*. 2003. p. 423-430.

[12] Lau, T., J. Cerruti, G. Manzato, M. Bengualid, J.P. Bigham, and J. Nichols, *A conversational interface to web automation*, in *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, ACM: New York, New York, USA.

[13] Lazar, J., A. Allen, J. Kleinman, and C. Malarkey, *What frustrates screen reader users on the web: A study of 100 blind users.* International Journal of human-computer interaction, 2007. **22**(3): p. 247-269.

[14] Leshed, G., E.M. Haber, T. Matthews, and T. Lau. *CoScripter: automating & sharing how-to knowledge in the enterprise*. in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2008: ACM.

[15] Li, I., J. Nichols, T. Lau, C. Drews, and A. Cypher, *Here's what i did: sharing and reusing web activity with ActionShot*, in *Proceedings of the 28th international conference on Human factors in computing systems*. 2010, ACM: Atlanta, Georgia, USA. p. 723-732.

[16] Microsoft. *Common commands in Speech Recognition*. 2014 [cited 2014]; Available from: http://windows.microsoft.com/en-us/windows/common-speech-recognition-commands#1TC=windows-7.

[17] Microsoft. *Cortana contextual awareness*. 2014 [cited 2014]; Available from: http://www.bing.com/dev/en-us/contextual-awareness.

[18] Nuance. *Dragon NaturallySpeaking Rich Internet Application*. 2014 [cited 2014]; Available from: http://nuance.custhelp.com/app/answers/detail/a_id/6940/~/information-on-rich-internet-application-support.

[19] Public Voice Lab, S., *PublicVoiceXML*. 2002.

[20] Puzis, Y., Y. Borodin, R. Puzis, and I.V. Ramakrishnan, *Predictive Web Automation Assistant for People with Vision Impairments*, in *To appear in proceedings of the 22th international conference on world wide web*. 2013, ACM: Rio de Janeiro, Brazil.

[21] Schnelle. *JVoiceXML*. 2013; Available from: http://webdesign.about.com/gi/o.htm?zi=1/XJ&zTi=1&sdn=webdesign&cdn=compute&tm=171&f=00&tt=14&bt=3&bts=31&zu=http%3A//jvoicexml.sourceforge.net/.

[22] Siri. *The Personal Assistant on Your Phone*. 2013 [cited 2013]; Available from: http://siri.com/.

[23] Stolcke, A., K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. Van Ess-Dykema, and M. Meteer, *Dialogue act modeling for automatic tagging and recognition of conversational speech.* Computational Linguistics, 2000. **26**(3): p. 339-373.

[24] Toutanova, K., D. Klein, C.D. Manning, and Y. Singer. *Feature-rich part-of-speech tagging with a cyclic dependency network*. in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. 2003: Association for Computational Linguistics.

[25] VoiceOver, *Screen reader from Apple*. 2010.

[26] VoiceXML. *W3C - Voice Extensible Markup Language*. 2009 [cited 2010]; Available from: http://www.w3.org/TR/voicexml20.

[27] Zhong, Y., T.V. Raman, C. Burkhardt, F. Biadsy, and J.P. Bigham, *JustSpeak: enabling universal voice control on Android*, in *Proceedings of the 11th Web for All Conference*. 2014, ACM: Seoul, Korea. p. 1-4.