Speech-Based UI Design for the Automobile

Bent Schmidt-Nielsen, Bret Harsham, Bhiksha Raj, Clifton Forlines

TR2008-006 February 2008

Abstract

In this chapter we discuss a variety of topics relating to speech-based user interfaces for use in an automotive environment. We begin by presenting a number of design principles for the design of such interfaces, derived from several decades of combined experience in the development and evaluation of spoken user interfaces (UI) for automobiles, along with three case studies of current automotive navigation interfaces.

Finally, we present a new model for speech-based user interfaces in automotive environments that recasts the goal of the UI from supporting the navigation among and selection from multiple states to that of selecting the desired command from a short list. We also present experimental evidence that UIs based on this approach can impose significantly lower cognitive load on a driver than conventional UIs.

Handbook of Research on User Interface Design and Evaluation for Mobile Technology

Volume I

Joanna Lumsden
National Research Council of Canada
Institute for Information Technology – e-Business, Canada

Information Science REFERENCE

INFORMATION SCIENCE REFERENCE
Hershey · New York

Acquisitions Editor:	Kristin Klinger
Development Editor:	Kristin Roth
Senior Managing Editor:	Jennifer Neidig
Managing Editor:	Sara Reed
Copy Editor:	Joy Langel, Katie Smalley, and Angela Thor
Typesetter:	Jeff Ash
Cover Design:	Lisa Tosheff
Printed at:	Yurchak Printing Inc.

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue, Suite 200
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

and in the United Kingdom by
Information Science Reference (an imprint of IGI Global)
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 0609
Web site: http://www.eurospanonline.com

Library of Congress Cataloging-in-Publication Data

Handbook of research on user interface design and evaluation for mobile technology / Joanna Lumsden, editor.

p. cm.

Summary: "This book provides students, researchers, educators, and practitioners with a compendium of research on the key issues surrounding the design and evaluation of mobile user interfaces, such as the physical environment and social context in which a device is being used and the impact of multitasking behavior typically exhibited by mobile-device users"--Provided by publisher.

Includes bibliographical references and index.

ISBN 978-1-59904-871-0 (hardcover) -- ISBN 978-1-59904-872-7 (ebook)

1. Mobile computing--Handbooks, manuals, etc. 2. Human-computer interaction--Handbooks, manuals, etc. 3. User interfaces (Computer systems)--Handbooks, manuals, etc. I. Lumsden, Joanna.

QA76.59.H36 2008

004.165--dc22

2007024493

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter XV
Speech-Based UI Design for the Automobile

Bent Schmidt-Nielsen
Mitsubishi Electric Research Labs, USA

Bret Harsham
Mitsubishi Electric Research Labs, USA

Bhiksha Raj
Mitsubishi Electric Research Labs, USA

Clifton Forlines
Mitsubishi Electric Research Labs, USA

ABSTRACT

In this chapter we discuss a variety of topics relating to speech-based user interfaces for use in an automotive environment. We begin by presenting a number of design principles for the design of such interfaces, derived from several decades of combined experience in the development and evaluation of spoken user interfaces (UI) for automobiles, along with three case studies of current automotive navigation interfaces. Finally, we present a new model for speech-based user interfaces in automotive environments that recasts the goal of the UI from supporting the navigation among and selection from multiple states to that of selecting the desired command from a short list. We also present experimental evidence that UIs based on this approach can impose significantly lower cognitive load on a driver than conventional UIs.

INTRODUCTION AND BACKGROUND

The US census bureau reported in 2005 that the average American spends over 100 hours driving to and from work every year and spends several hundred more driving on errands, vacations, to social engagements, and so on. A significant fraction of this driving is spent while engaged in concurrent activities, such as listening to the radio, listening to music on a personal music player, operating an in-car navigation system, and talking on or accessing information with a hands-free or hand-held cell phone. These secondary activities involve interactions between the driver and a device that can distract the driver from the primary task—that of driving safely to the destination. While it is understood that the safest option is for a driver not to engage in such activities and instead concentrate completely on driving, drivers seem intent on engaging in these distractions; thus, minimizing the impact on safety is a worthy area of research.

It has been estimated that at least 25% of police reported accidents in 1995 involved some form of driver inattention (Wang, Knipling, & Goodman, 1996). A study by Stutts et al. (2001) estimated that, of the drivers whose state was known at the time of the crash, at least 13% were distracted, with adjusting the audio system of the car accounting for 11% of these distractions. Since the advent of cellular phone technology, there has been a great deal of research on the effects of cellular phone use on driving performance (e.g., Ranney et al., 2004); however, only recently have studies begun to address the effects of use of other in-car systems on driving performance. In an analysis of the 100-Car Naturalistic Driving Study, Klauer et al. (2006) found that "Drivers who are engaging in moderate secondary tasks are between 1.6 and 2.7 times as likely to be involved in a crash or near-crash, and drivers engaging in complex secondary tasks are between 1.7 and 5.5 times as likely" (p. 28).

Since these studies, a number of electronics manufacturers have introduced products that incorporate personal digital music collections into automobile audio systems. Some automobile manufacturers have gone as far as bundling a personal digital music player with the purchase of a new car. Recent high-end car models also offer GPS-linked navigation systems. These systems offer functions such as address entry and point-of-interest search, both of which are usually implemented as multi-step tasks requiring significant attention from the user. Navigation and entertainment systems are among the first examples of highly complex automotive interfaces that are available for use while driving. We expect the amount of information available in the car to continue increasing drastically as more and more car systems become networked, and as car makers try to differentiate their products by offering new functionality.

Given this situation, it becomes necessary to design effective user interfaces that will enable drivers to operate devices such as radios, music players, and cellphones in a manner that distracts them minimally from driving, while still allowing them to obtain the desired response from their devices.

A compelling choice for UI design in the automotive environment is the speech-based user interface. By "speech-based" we mean an interface which uses utterances spoken by the user as a primary input mode. A speech based interface may also have other input modes, such as dedicated or softkey input, and may also have voice feedback and/or visual feedback. By being largely hands free, a speech-based interface can minimize the need for the driver to disengage their hands from the steering wheel. By presenting information aurally, it can allow a driver to keep their eyes on the road.

These qualities are by themselves not sufficient: automobile UIs must not only allow drivers to keep their hands on the wheel and their eyes on the road, but also must allow them to keep their mind on the task at hand—that of driving safely. Spoken input is typically used as substitute for tactile input. It is frequently unclear how tactile actions such as turning a knob, pressing a button, or selecting an item on a touch screen may best be replaced by simple spoken commands that can be recalled easily by the user. The inability to recall the correct command can lead to poor system response and driver distraction. Further, Automatic Speech Recognition (ASR) engines are error-prone—they will often fail to recognize spoken input correctly, or worse still mistakenly recognize an incorrect command. These problems are magnified in noisy environments such as the inside of a fast-moving automobile.

Common approaches to minimize the adverse effects of ASR errors include detailed dialog mechanisms, help menus, confirmatory prompts, and error correcting dialogs. Unfortunately, these mechanisms are problematic in an automotive environment in which it is important for interactions between a driver and a system to be short in order to create minimal distraction to the driver.

Interface designers must strive to minimize unnecessary cognitive load such as those arising from extended interactions with a system, frustration from poor task completion, and other distractions of the mind. Recognition errors can often be minimized by constraining the choices that the ASR engine must consider at any one time. When using this approach, the UI must be designed to restrict the number of spoken commands available at each state while at the same time ensuring that the currently available commands are evident to a user. Thus, there is a strong bi-directional coupling between ASR performance and usability: ASR problems can manifest themselves to the end-user as usability problems, and interface design or implementation problems can easily lead to reduced recognition accuracy. Our experience has been that speech-based user interfaces are most effective when designed around the constraints of both ASR and UI. It is probably unrealistic to expect application and interface designers to be well versed in the technical details of ASR; however we believe that using a set of reasonable design guidelines could help automotive designers to design more effective speech-based interfaces.

There are a number of published documents that give design guidance for telematics interfaces, (see UMTRI, 2006); however, very few of these give any specific guidelines for the design of speech interfaces for telematics systems. Nor do they specify which general guidelines should apply to speech-based UIs, and which should not. In fact, some sets of guidelines (for instance, Society of Automotive Engineers, 2004) specifically exclude speech interfaces. Of course, there are also many sets of design guidelines for voice user interfaces in general, but few of these were written for the specific issues of an automotive user interface.

In this chapter, we will begin by presenting a number of design principles for the design of speech-based UIs, that have been derived from several decades of our combined experience in the development of spoken user interfaces for automobiles. Our discussion addresses issues such as the cognitive load imposed on the driver, interaction time, task completion rate, feedback, and the appeal of the system to the user. Based on these principles, we then present a brief review of the spoken UIs in a few current car models and highlight the positive and negative facets of these interfaces.

We conclude the chapter with a description of a speech interface paradigm which we call SILO (Divi et al., 2004) that conforms to most of our design principles, as an alternative to highly modal tree structured menus for the selection of a item out of a large number of alternatives. We describe an experiment that indicates that the SILO paradigm has significantly lower driving interference than the menu-based interface for a music selection task (Forlines et al., 2005).

DESIGN PRINCIPLES FOR SPEECH-BASED AUTOMOTIVE UIS

Here, we introduce a short set of design goals and specific recommendations for consumer automotive interfaces. These are based on the collective experience of the authors in implementing speech interfaces for use in automobiles over the last ten years. Most of the work that forms the basis for these recommendations is unpublished; however we feel that it is useful to present these recommendations here in collected form. Many of these guidelines can be found individually in other sources—this combined set is based on the particular constraints of consumer automotive applications.

Note that these guidelines are written with consumer automobile driving in mind. Commercial and military applications have different constraints and thus should be treated differently (for instance, the military user base receives training prior to in-field use, and the risk vs. task success rate balance may be different).

General Design Goals for Automotive Speech Interfaces

- **Reduce driver cognitive load.** In our view, the highest priority in design of automotive interfaces is to reduce the risk associated with performing any secondary task while driving. Since interfaces in use by a passenger can be distracting to the driver, the driver's cognitive load should be considered when designing interfaces that will be used by other occupants.
- **Reduce interaction time.** Reduced interaction time should lead to reduced risk.
- **Increase task completion rate.** Increased task completion rate should lead to reduced risk (fewer repeated interactions) and better user experience. Of course, task completion rate is related to the underlying speech recognition accuracy, but is also strongly influenced by the affordances offered by the user interface.
- The feedback (visual and audio) should reinforce correct use by the user.
- The user should be able to mentally model the system behavior.
- System should be effective for an experienced user (e.g., a long time owner).
- System should be appealing for a new user (especially during a pre-sales test drive).

These goals are given in order of decreasing importance. Not all of these design goals are achievable at the same time, and in fact, some may be in opposition to each other for various tasks. For instance, it can be difficult to make a system that is effective for both a new user and a long-time user. Below are some design recommendations that we believe follow from these design goals

Design Recommendations

1. **Interactions should be user paced:** Speech interfaces for use in automobiles should be *entirely* user paced. The primary task of the driver is the safe operation of the automobile. The operation of other equipment in the vehicle is a secondary task. The driver must be available to respond to changing traffic conditions. Driver responses to the system must be timed when the driver can spare attention for the secondary task. Thus every voice response by the driver should require its own push-to-talk event. Systems which ask for further voice input without waiting for a signal from the driver can cause the driver to feel pressured to respond even under difficult traffic conditions. Many state of the art systems include dialogs that violate this recommendation.
2. **Use a Push and Release button with a listening tone:** After the user activates the Push-to-talk, the system should promptly produce a short pleasant listening tone to indicate that it is listening. Studies have indicated that in the absence of a listening tone, some users will start speaking prior to activating the push-to-talk and other users will delay speaking for a variable amount of time. Either of these changes in timing can confuse the speech recognizer, resulting in an overall reduction of recognition accuracy.
   In the presence of a listening tone, most users quickly learn to wait for the listening tone and to start speaking promptly after they hear the tone. However, the time between activation of the push-to-talk and production of the listening tone must be short and consistent; otherwise the cognitive load on the user is higher.
   Push and Release interfaces, where the user releases the PTT button immediately, usually impose a lower cognitive load on the user

than Push and Hold style interfaces, where the user is required to hold the button for the entire utterance. This is because the user's physical actions are sequential (press, then speak) in a Push and Release interface rather than simultaneous. It can be useful to indicate with a different tone that the system is no longer listening. This affordance helps the user to adapt to system constraints such as listening timeouts.

3. **Use physical input instead of voice for simple things:** There are numerous actions for which there are existing effective physical interfaces. In almost all instances keeping those physical interfaces is superior to the substitution of voice commands. For example, using up and down buttons to navigate through a list of options is much easier than saying commands such as "scroll down". In addition, the users are already familiar with such physical interfaces. There is also less new learning involved to be able to use the interface.
4. **Provide always-active commands including voice help:** A "Help" or "What can I say" command should always be available for users who are unsure as to what functions are currently available. Systems that have modal behavior should have mode-specific voice help, as well as commands to cancel the current mode and to backup.
5. **Use consistent grammars with minimal modality:** Consistency and predictability of the grammar is very important so that the user will have less to remember. Also the grammar should have minimal modality so that users will have access to the functions of the system with fewer interactions and will not need to remember the state of the system. Modal behavior should be associated with audio and visual cues so that the user can easily understand what mode the system is in.
6. **Visual cues should be consistent with the active grammar:** The use of visual cues that do not model the grammar usually causes an increased number of out-of-grammar utterances.
7. **Feedback should indicate the recognition result:** The visual and audio feedback given to the user should indicate what was heard by the speech recognizer. This helps to reduce confusion when the system does not behave as expected, either due to misrecognition, or to user confusion about the effect of the spoken command. Users have a strong tendency to mimic the sentences that they hear; thus, some systems echo the recognized utterance back to the user as a confirmation of what the system heard or echo the preferred form of the command to help the user learn the grammar.
8. **Reasonable behavior for out-of-grammar utterances:** Systems should attempt to detect out-of-grammar utterances and indicate that the last utterance was not understood. It is far better for the system to respond that the command was not understood than it is to perform an unexpected action. Unexpected actions cause user confusion (e.g., was the utterance in the grammar and misrecognized, or not in the grammar?). Lack of rejection raises the cost of an out-of-grammar utterance as the user must take whatever action is necessary to undo the undesired action, resulting in significantly longer task completion times.
9. **Provide reasonable backoff strategies:** In some cases a speech interface will consistently fail to recognize certain voice commands from the user. For example, a system may allow the entry of a street name as part of an address, a very difficult voice recognition problem for a locality with many streets. Thus, if the voice system fails to get the correct street after a couple of tries, it should offer an alternative method of entry, such a spelling. As another example, many systems allow the entry of long telephone numbers in one long utterance by the user. For some users the error rate for strings of ten or more digits can be too high. For such users the system should allow the entry and correction of the digits forming a long telephone number in smaller chunks.

EVALUATION OF RECENT AUTOMOTIVE SPOKEN USER INTERFACES

In April 2006, we conducted a review of three automotive speech-based interfaces from model year 2006. Rather than give an exhaustive review, here we highlight the system attributes that had the most impact on interface usability. In order to focus on the attributes of the interfaces rather than the identities of the manufacturers, we will call these interfaces "Model A," "Model B," and "Model C."

The state of the art for speech-based navigation systems at the time of the review was:

- Push and Release interface with listening tone
- Grammar based systems with varying degrees of modality
- Navigation Entry of part or all of an address by voice

There was significant variation in these features between the tested systems. There was also significant variation in the accuracy of the underlying automatic speech recognition engines, but we found the usability to be more affected by the UI design than the underlying ASR.

Model A

Model A had good ASR engine accuracy. However, there was no rejection of out-of-grammar utterances, so ASR errors (misrecognitions) frequently led to unexpected actions.

User Pacing

This system was entirely user-paced, with a push-to-talk button which was required for every interaction, and a prompt, pleasant listening tone.

Grammar Consistency and Modality

Model A had a fairly simple modal grammar structure, with some common always-active commands. The grammar format was consistent, and the audio feedback modeled the grammar. Each mode was associated with a visual state. Overall it was easy to understand what mode the system was in, and to guess what voice commands were available.

However, there were some modes with inconsistent design. Two interesting examples:

- **Voice help modes:** Each voice mode (v_n) had a corresponding help mode (vh_n) which was invoked through the always-active help command. Each help mode visually displayed the available commands for its parent mode in a numbered list.
  However, the help modes had their own very limited grammar. Thus in any help mode (vh_n) the commands for v_n were displayed but not active. The result of this was to induce users to utter unavailable commands in help mode, resulting in frequent and confusing misrecognition in help mode. In order to utter one of the displayed commands, users had to first dismiss the help screen, at which point the commands were no longer visible. This design was almost consistent with design recommendation 6 (*Visual cues should be consistent with the active grammar*), but the seeming minor detail, that the visual cues for these modes matched an inactive grammar, led to a major usability issue.
- **Setup modes:** This system had several preference modes for manipulating system settings. Typically, the system displayed a set of buttons and sliders when in one of these modes. There were voice commands for manipulating the values, but the commands were inconsistent and there was no visual indication of which objects could be manipulated by voice or what command language to use. There were physical/softkey methods

to change the settings, and it appeared that the speech interface had been grafted onto these modes as an afterthought. The interface could have been more effective without the additional voice commands, consistent with design recommendation 3 (*Use physical input instead of voice for simple things*).

Other Usability Issues

The interface included a 'Back' button, which was probably designed to move the interface to the previous state after a recognition event, but had inconsistent behavior. This was an interesting attempt to provide a reasonable backoff strategy for misrecognition. Prior speech user interface (SUI) implementations have shown that a 'Back' button or an "Undo that" command that undoes the previous action can be a very effective affordance for an ASR system, mitigating the inevitable ASR errors. In this particular case, the behavior was unreliable.

Model B

Model B had poor ASR engine accuracy, especially in high noise conditions typical of highway driving.

There was no rejection of out-of-grammar utterances, however, some modes accepted unavailable commands and then reported that that command was not available in the current mode.

User Pacing

This interface had a push-to-talk button which was required for every interaction. As noted in design recommendation 2 (*Use a Push and Release button with a listening tone*), this interface pattern usually improves usability, but not in this case.

Model B's implementation of push-to-talk degraded not only the usability of the interface, but also the ASR performance, all while increasing cognitive load: Activation of the push-to-talk was followed by a visual cue which indicated that the user could begin speaking. This visual cue was followed by a highly variable delay and then a listening tone. Sometimes the listening tone was produced promptly, other times there was a delay of 1-2 seconds after the visual cue. Our testing indicated that the variable listening tone delay frequently caused recognition errors (when the users spoke too quickly) and higher cognitive load (the user had to actively think about waiting for the tone). Furthermore, the listening tone was hard to hear in high noise conditions.

Grammar Consistency and Modality

Model B had a fairly simple modal structure. However, the voice modes were not well associated with visual cues. This made it very difficult to determine which commands were available at any given time. For instance, there were several modes where a map was visible on the screen, but only one of these was "map mode." Therefore the system sometimes responded "That command is only available in map mode" even when there was a map showing on the screen.

Model B also had a confusing lack of consistency between visual cues, voice commands, and voice feedback. Recognition feedback was aligned with, but did not match the grammar. When the user spoke a command of the form "action object," the voice response was "<action>-ing object." For instance:

- **User:** *"raise temperature"*
- **System:** *"raising temperature"*

This was a clear indication of what action the system was taking, but an indirect indication of what command had been recognized. The use of different language sometimes induced users to copy the response language (e.g., to say "raising temperature" by accident).

In those cases where users mimicked the response language, the lack of rejection of out-of-grammar utterances usually caused an unexpected action to occur, forcing the user to attempt to undo the unexpected action before re-trying the original task.

Model C

Model C had reasonably good ASR performance. It also had rejection of out-of-grammar utterances. Some out-of-grammar utterances were misrecognized, but many were rejected.

User Pacing

Model C was not user paced. An interaction with the system could consist of a single (conversational) turn by the user or an entire sequence of turns by user and system. Push-to-talk was used to initiate a task, but then the system controlled the turn-taking and pace until the end of the task. To mark the user's subsequent turns, the system generated a voice prompt followed by another listening tone and immediately expected more voice input from the user. This design was confusing to new users because they did not know when a conversational sequence would be finished.

When used during driving conditions, the interface created a high cognitive load on the user, who had to pay enough attention to the voice prompts to be able to respond to demands for more input. If the user ignored a prompt, the prompt and listening tone were repeated, thus during a task the system constantly demanded attention from the user regardless of the traffic conditions. Our experience indicates that this can be hazardous—see design recommendation 1 (*Interactions should be user paced*).

Grammar Consistency and Modality

In contrast to the other interfaces tested, Model C consisted of a tree of menus 3-5 levels deep. At any given time, the active grammar commands were based on the active node of the menu tree. In addition, the voice commands for the top two levels of the tree (including voice commands for navigation through the menu tree) were always active. Visually, up to three levels of the menu tree were displayed on a single screen.

The command language for Model C was inconsistent and hard to remember. The voice feedback for a command sometimes echoed the recognized command and sometimes told the user what action the system had taken. However, there was a visual display of the command recognized from the last voice utterance, which was useful. There was a good correspondence between the text of on-screen objects/icons and the command language used to select them, but there was no clear visual indication of which objects could be manipulated by voice.

The combination of menu complexity and grammar inconsistency made it difficult for the user to determine the active menu node and to infer what voice commands were available in a particular state. This system had a very high cost for recognition errors because a misrecognition would usually cause a transition to a state very far away in the menu tree, with no way to recover.

Summary of Evaluation of Current Interfaces

Model A was superior to the other interfaces that we tested. The consistency of the UI design and adherence to good design principles was the primary reason this was the best of the three tested systems, although the fact that this system had good ASR performance also contributed. In those areas of the system where the design had inconsistent grammars and/or visual cues, the resulting misrecognitions combined with the lack of rejection of out-of-grammar utterances to significantly degrade the usability. The 'back' functionality provided by this interface could have been a significant feature if it had worked consistently.

Model B was the worst of the interfaces tested. In spite of the poor ASR performance, the main weakness of this system was the implementation of the Push-and-Release with listening tone. This system provides an example of how an interface implementation problem can manifest as an ASR problem. Here, recognition performance was seriously degraded by the variable delay of the listening tone. In addition, the lack of rejection, and the relatively poor underlying ASR performance (especially in noise), all combined to make this interface unacceptably poor.

Model C had the most complicated modal structure. It also had system-paced turn-taking, which usually results in higher cognitive load on the user. Its speech recognition was reasonably good, with some notable exceptions.

In general, current state of the art systems show very distinct design philosophies. While we are in favor of innovative design, we hope that the design of future systems will more thoroughly consider the driver's cognitive load. It is interesting to note that the underlying speech recognition performance, while significant, was far from the most important factor in the usability of these interfaces—this shows the importance of good interface design in this application space.

A SPEECH-IN LIST-OUT APPROACH TO IN-CAR SPOKEN USER INTERFACES

Many in-car applications for which spoken UIs may be used deal with the selection of one of an enumerable set of possible responses, e.g., selecting one of a number of radio stations, retrieving a song from a music collection, selecting a point of interest (from the set of all points of interest), etc. The most common UI for these applications is through a hierarchy of menus. Even when the UI is speech-based, speech is used primarily as an input or output mechanism for the underlying menu-driven interface.

At Mitsubishi Electric Research Labs we have developed an alternative speech UI for selecting an element from a set, which we refer to as the "Speech-In List-Out" interface or SILO. In this section, we briefly describe the SILO interface. We also describe an experiment that indicates that the SILO interface can result in lower driving interference than the menu-driven interface.

Selection from a Set

The most common paradigm for retrieving a specific response from a large set is through menus; however, as the size of the selection set increases (e.g. for a UI to a digital music player with an ever increasing repertoire of songs), the tree of menus increases in depth and width, and can become problematic, particularly for users who are also simultaneously involved in other attention-critical tasks such as driving.

While the problem may be alleviated to some degree through voice output and spoken input (Leatherby & Pausch, 1992; Cohen et al., 2000), this by itself is not a solution. Spoken enumeration of menu choices cannot fully replace visual display; a deeply nested menu-tree, presented aurally, is very demanding in terms of cognitive load. Knowing that a quick glance at a screen can recover forgotten information relieves the user from having to keep close track of the system's state in their mind. Spoken-input-based interfaces must address the problem of misrecognition errors (e.g., in noisy environments) and, more importantly, the "what can I say" problem—users must be able to intuit what to say to the speech recognition system (which typically works from rigid grammars for such tasks). This latter issue can be particularly difficult when selecting from long lists.

The SILO interface (Divi et al., 2004) recasts the UI as a search problem. The set of all possible responses are viewed as documents in an index. The user prompts the system with a single spoken input, which is treated as a query into this index. The system returns a short *list* of possible matches to the query. The user must make the final selection from this list. While search-based speech UIs have previously been proposed (e.g., Cohen, 1991), SILO differs from them in that it places no restrictions on the user's language. The user is not required to learn a grammar of query terms. The simplicity of the resulting interactions between the user and the system is expected to result in a lower cognitive load on the user, an important consideration in the automotive environment.

The enabling technology for SILO is the SpokenQuery (SQ) speech-based search engine (Wolf & Raj, 2002). SQ is similar to text-based information retrieval engines except that users speak the query instead of typing it. Users may say whatever words they think best describe the desired items. There is no rigid grammar or vocabulary. The output is an ordered list of items judged to be pertinent to the query.

A major problem for speech UIs is misrecognition error, which can derail an interaction. The reason is that they attempt to convert the user's spoken input to an unambiguous text string, prior to processing. In contrast, SQ converts the spoken input to a set of words with associated probabilities, which is then used to retrieve documents from an index. The set of words and their associated probabilities are derived from the *recognition lattice* that represents candidate words considered by the recognizer. The lattice often includes the actual words spoken by the user even when they are not included in the disambiguated text output. As a result, SQ is able to perform well even in highly noisy conditions (such as automobiles) in which speech UIs that depend on accurate recognition fail. Consequently, SILO interfaces are able to perform robustly in noisy environments (Divi et al., 2004). Table 1 lists some example phrases and their (often poor) interpretation by the speech recognizer along with the performance of the SQ search.

Though the disambiguated phrase output by the speech recognition system is often wildly inaccurate, SQ manages to return the desired song near or at the top of the list. The right-most column shows the rank of the desired result in SQ's output.

The design of the SILO interface follows several of the design principles enumerated above. Specifically:

- **Interactions should be user paced:** The system always waits for the user to initiate the next interaction.
- **Appropriate use of speech:** Speech input is used only for choosing from very large sets where the use of buttons (for scrolling and selection) is inefficient or impossible. All choices from small sets are performed by direct manipulation.
- **Non-modal:** SILO is not modal, therefore the user does not need to remember the system state.

Experimental Evaluation of the SILO Interface

We designed an experiment to compare the SILO interface to a menu-based UI for in-car music selection. An effective in-car UI must not only allow users to find desired information quickly, but also affect their driving performance minimally. To evaluate both factors, we compared quantitative measurements of simulated steering and

Table 1. Example of SQ search to retrieve songs from a collection

User says…	**System hears…**	**SILO search result**
"Play Walking in my shoes by Depesh Mode"	layla [NOISE] issues [NOISE] [NOISE] load	1
"Depesh Mode, Walking in my shoes"	E [NOISE] looking [NOISE] night shoes	1
"Walking in my shoes"	law(2) pinion mae issues	1
"Walking in my shoes by Billy Joel" (partially incorrect information)	walking inn might shoes night billie joel	1
"um, uh, get me Credence Clearwater Revival… um… Who'll stop the Rain" (extra words)	fall(2) [UH] dead beat creedence clearwater revival [UM] long will stop it rains	1
"Credence Clearwater Revival, Who'll stop the Rain" (very noisy environment)	[NOISE] [COUGH] clearwater revival [COUGH] down [COUGH] [BREATH]	6

braking while searching for music with the two interfaces.

Our hypotheses were:

- **H1:** Subjects will more accurately track a moving target with a steering wheel while searching for songs using the Mediafinder SILO interface than while using the menu-driven interface.
- **H2:** Subjects will react faster to a braking signal while searching for songs using the SILO interface than while using the menu-driven interface.
- **H3:** Subjects will be able to find songs faster while using the SILO interface than while using the menu-driven interface while driving.

Experiments were conducted on a simple driving simulator, such as those in Beusmans et al. (1995) and Driving Simulators (2006), that mimicked two important facets of driving—steering and braking. The simulator had both a "windshield" and "in-dash" display. Subjects steered, braked, and controlled the interfaces with a steering wheel and gas and brake pedals. A microphone was placed on top of the main monitor. Steering was measured with a pursuit tracking task in which the subject used the wheel to closely frame a moving target (Strayer et al., 2001). The simulator recorded the distance in pixels between the moving target and the user-controlled frame 30 times a second. Braking was measured by recording subjects' reaction time to circles that appeared on screen at random intervals. Subjects were asked to only react to moving circles and to ignore stationary ones. Moving and stationary circles were equally probable.

We built two interfaces for this study. The first was a menu-based interface based on a sampling of currently available MP3 jukeboxes; the second was the SILO interface. Both interfaces ran on the same "in-dash" display and were controlled using buttons on the steering wheel. Both interfaces searched the same music database of 2124 songs by 118 artists, and both were displayed at the same resolution in the same position relative to the subject. Additionally, both interfaces displayed the same number of lines of text in identical fonts. Neither interface dealt with many of the controls needed for a fully functional in-car audio system, such as volume, power, and radio controls.

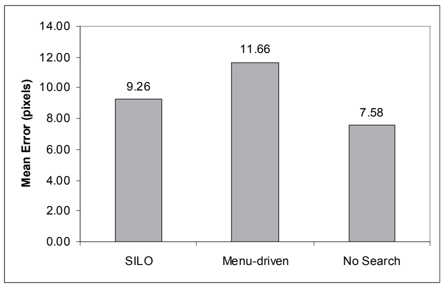
Fourteen subjects, eight male and six female, of ages ranging from 18 to 37, participated in this experiment. All but one were regular automobile drivers. Subjects were first instructed on how to correctly perform the steering and braking tasks and were given as much time as they wanted to practice "driving." Next, they were instructed to search for and playback specific songs while performing the driving task. Subjects completed 8 trials each with both the SILO and the menu-driven interfaces. Before each set of trials, subjects were instructed on how to use the current interface and allowed to practice searches while not driving. During each trial, the testing application displayed the steering and braking signals along with instructions asking the user to search for a specific song (e.g., "Please listen to the song Only the Good Die Young by Billy Joel from the album The Stranger"). Subjects were allowed to take a break between trials for as long as they wished. The order that the interfaces were used was balanced among participants, and the order of the requested songs was randomized. The application logged the distance between the moving target and the subject-controlled black box, as well as the reaction time to any brake stimulus presented during each trial. The task time was also logged, measured from the moment that the instructions appeared on the screen to the moment that the correct song started playing. To reduce learning effects, only the last 4 of each set of 8 trials contributed to the results.

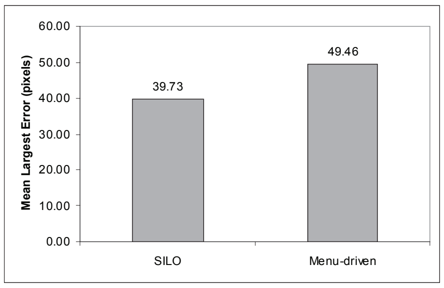
Results

Our data supports hypotheses H1 and H3 and rejects H2.

- **H1:** Subjects were able to steer more accurately while searching for music using the SILO interface than with the menu-driven interface (on average, 9.2 vs. 11.6 pixels of

Figure 1. The SILO interface had both a significantly lower mean steering error (below left) and a significantly lower mean largest steering error (below right) than the menu-driven interface (Source: Forlines et al., 2005)





error respectively, t(13) = 3.15, p=0.003). However subjects steered most accurately while driving without searching (on average, 7.4 vs. 9.2 pixels for SILO, t(13)=2.5, p=0.013). The average error for each condition is shown in Figure 1 (*below left*). The SILO interface had a significantly lower *maximum* steering error as well (39.7 pixels vs. 49.4 pixels, t(13)=2.27, p=0.02). This measurement of error roughly corresponds to the point when the subject was most distracted from the steering task. If actually driving, this point would be the point of greatest lane exceedence. The average maximum error for the two interfaces is shown in Figure 1 (*below right*).

- **H2:** The mean breaking reaction times were indistinguishable between the SILO and menu-driven conditions (on average, 1196 ms vs. 1057 ms, t(13)=1.66, p=0.12); however, subjects were significantly faster at braking while not searching for music than while searching using the SILO (p=0.008) or the menu-driven (p=0.03) interface. The mean reaction time to the brake stimulus for each condition is shown in Figure 2.
- **H3:** Subjects were significantly faster at finding and playing a specific song while using the SILO interface than while using the menu-driven interface (on average, 18.0 vs. 25.2 sec., t(13)=2.69, p=0.009). The mean search time for each interface is shown in Figure 3. It is important to note that it was not unusual for the SILO interface to have a computational interruption of 3-6 seconds, which was included in the SILO search time. A faster CPU or better microphone could decrease this time.

Figure 2. There was no significant difference in mean break reaction times between the search conditions (Source: Forlines et al., 2005)

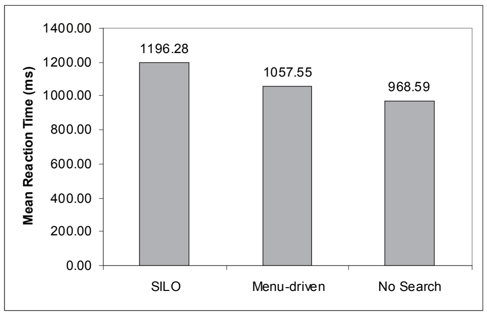
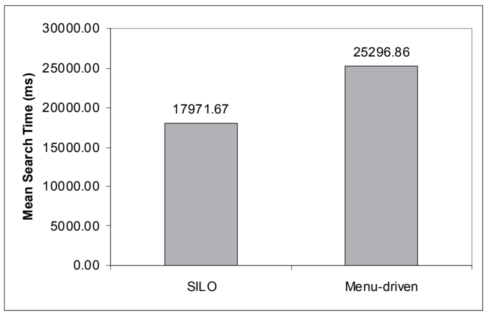


Figure 3. Subjects were significantly faster at finding songs with the SILO interface (Source: Forlines et al., 2005)



Since SILO returns a list of songs from which the user must select, an important factor is the position of the correct song in the returned list. In 35 out of the 56 trials SILO returned the correct song at the top of the list on the first try. The average position for the correct song for all SILO trials was 5.1.

Experimental Discussion

The evidence indicates that our SILO interface for music finding has measurable advantages over the standard menu-based approach for users operating a simulated automobile. The SILO interface scores better than the menu-based system on several of the design goals mentioned outlined above. It poses lower cognitive load, as evidenced by the improved driving accuracy, has lower interaction time, and is more effective for the experienced user. Although we observed no statistical difference in mean break reaction times between the SILO and menu-driven interfaces, closer inspection revealed that subjects were less likely to encounter a brake stimulus while using SILO due to the faster task completion. SILO effectively results in fewer opportunities for braking error.

The SILO interface has several additional advantages that were not explicitly evaluated in the experiment. We expect that these will be subjects of future study:

- **Flexibility:** The SILO interface was able to retrieve songs from only partial information, such as the song title. On the other hand, for the menu-based interface, it would not have been sufficient to ask subjects to find the song “Never Die” without telling them it is by the artist “Creed” on the album “Human Clay.”
- **Scalability:** The song library used for the study contained only 2124 songs. The latest handheld music players can hold over 10,000 songs. As the number of available artists, albums, and songs grows, we expect the time needed to search through a menu-driven interface to grow as well. An informal evaluation of the SILO interface searching a database of 250,000 songs shows no noticeable differences in search time.
- **Robustness:** The metadata in the music files in our library was not always consistent. For example, music by the group “The B-52s” was erroneously split into many artists: “The B-52s,” “B-52s,” “The B52s,” etc. While these inconsistencies were problematic for the menu-driven interface, they do not affect the SILO interface.

The experiments reported here only evaluated SILO in a limited music selection task. Many music jukeboxes can present their content in alternative fashions such as user defined playlists, favorites, etc. Mediafinder is easily modifiable to handle playlists and personalization; however a rigorous evaluation of its capabilities in this direction remains to be performed.

Other limitations of this study include the fact that an actual in-car environment that included environmental noise was not used. Other tests using automotive speech data have shown that the SpokenQuery information retrieval engine is very robust to high levels of environmental noise (e.g., Divi et al., 2004). We are therefore optimistic about the performance of the SILO interface in real in-car environments, but must confirm this expectation with future experiments. Finally, we look forward to a comparison between SILO and other speech based music selection systems.

FUTURE TRENDS

The current trend toward increasing the number and complexity of secondary tasks for automobile drivers is both worrisome and accelerating. The risks associated with these complex tasks raise two opportunities. First, standards bodies, government agencies, and the public must demand less distracting, safer interfaces for drivers. Second, manufacturers must provide these interfaces, and perhaps market not only their additional functionality, but also their safety advantages. Researchers and manufacturers must conduct studies measuring the degree of interference between the driving task and existing and future automotive user interfaces. Safety and liability dictate that these experiments be conducted in a driving simulator environment.

CONCLUSION

The advent of complex user interfaces in automobiles raises many issues relating to safety and usability, some of which can be mitigated by the appropriate use of speech in the UI. We have presented a set of design principles that can help mitigate some of the problems cited, and have applied these principles in a review of several existing automotive speech interfaces. Finally, we presented an in-car interface for the selection of items from a large collection and have shown that this method interferes less with driving than the current status quo. Much work remains to maximize the safety and usability of complex devices in automobiles, and we hope that this writing will aid the UI designer in this endeavor.

REFERENCES

Beusmans, J., & Rensink, R. (Eds.) (1995). *Cambridge Basic Research 1995 Annual Report* (Tech. Rep. No. CBR-TR-95-7). Cambridge, MA: Nissan Cambridge Basic Research.

Cohen, P. (1991). Integrated interfaces for decision-support with simulation. In *Proceedings of the 23rd Winter Conference on Simulation* (pp. 1066-1072).

Cohen, P., McGee, D., & Clow, J. (2000). The efficiency of multimodal interaction for a map-based task. In *Proceedings of the Applied Natural Language Processing Conference (ANLP'00)* (pp. 331-338). Seattle, WA: Morgan Kaufmann.

Divi, V., Forlines, C., Van Gemert, J.V., Raj, B., Schmidt-Nielsen, B., Wittenburg, K., Woelfel, J., Wolf, P., & Zhang, F. (2004, May 2-7). A speech-in-list-out approach to spoken user interfaces. In *Proceedings of the Human Language Technology Conference (HLT 2004)* (pp. 112-116). Boston, MA.

Driving Simulators. (2006). Retrieved October 24, 2006 from http://www.inrets.fr/ur/sara/Pg_simus_e.html

Forlines, C., Schimdt-Nielsen, B., Raj, B., Wittenburg, K., & Wolf, P. (2005). A comparison between spoken queries and menu-based interfaces for in-car digital music selection. *IFIP TC13 International Conference on Human-Computer Interaction (INTERACT 2005)*

Klauer, S.G., Dingus, T. A., Neale, V. L., Sudweeks, J.D., & Ramsey, D.J. (2006). *The Impact of Driver Inattention on Near-Crash/Crash Risk: An Analysis Using the 100-Car Naturalistic Driving Study Data*. (DOT HS 810 594) Washington, DC: United States Government National Highway Traffic Safety Administration, Office of Human-Vehicle Performance Research.

Leatherby, J.H., & Pausch, R. (1992) Voice input as a replacement for keyboard accelerators in a mousebased graphical editor: An empirical study. *Journal of the American Voice Input/Output Society*, *11*, 2.

Ranney T., Watson, G. S., Mazzae, E. N., Papelis, Y. E., Ahmad O., & Wightman, J. R. (2004). *Examination of the distraction effects of wireless phone interfaces using the national advanced driving simulator—Preliminary report on freeway pilot study* (DOT 809 737). East Liberty, OH: United States Government National Highway Traffic Safety Administration, Vehicle Research and Test Center

Society of Automotive Engineers. (2004). *SAE recommended practice navigation and route guidance function accessibility while driving* (SAE 2364). Warrendale, PA: Society of Automotive Engineers.

Strayer, D.L., Drews, F. A., Albert, R. W., & Johnston, W. A. (2001). Cell phone induced perceptual impairments during simulated driving. In D. V. McGehee, J. D. Lee, & M. Rizzo (Eds.), *Driving assessment 2001: International symposium on human factors in driver assessment, training, and vehicle design.*

Stutts, J.C., Reinfurt, D.W., Staplin, L.W., & Rodgman, E.A. (2001). *The role of driver distraction in traffic crashes.* Washington, DC: AAA Foundation for Traffic Safety. Retrieved October 24, 2006, from http://www.aaafts.org/pdf/distraction.pdf.

US Census Bureau News. (2005). Americans spend more than 100 hours commuting to work each year. *Census Bureau Reports*, Retrieved October 24, 2006, from http://www.census.gov/Press-Release/www/releases/archives/ american_community_survey_acs/004489.html.

UMTRI Driver Interface Group. (updated 2006). *All the major telematics guidelines we've seen.* University of Michigan Transportation Research Institute, Driver Interface Group. Retrieved October 26, 2006, from http://www.umich.edu/~driving/guidelines/guidelines.html

Wang, J., Knipling, R. R., & Goodman, M. J. (1996). The role of driver inattention in crashes; new statistics from the 1995 crashworthiness data system (CDS). In *proceedings of the 40th Annual Proceedings: Association for the Advancement of Automotive Medicine* (pp. 377-392).

Wolf, P., & Raj, B. (2002). The MERL Spoken-Query Information Retrieval System: A System for Retrieving Pertinent Documents from a Spoken Query. In *Proceedings of ICME, Vol. 2* (pp. 317-320).

KEY TERMS

Cognitive Load: A measure of the mental effort required to carry out a given task.

Driver Distraction: A measure of the degree to which attention is taken away from the driving task.

Listening Tone: A sound generated by a speech-based user interface when it is ready to accept spoken input

Lombard Effect: The specific changes in style of speech caused by the presence of noise. In particular the speech gets louder and higher frequencies are emphasized

Misrecognition: A speech recognition result which does not accurately represent what was spoken by the user. In spoken command recognition, recognizing the exact words spoken is not necessary to avoid a misrecognition as long as the correct command is recognized.

Push and Hold: A type of speech interaction where the user must hold down a button while speaking to the system. This kind of system is familiar to most users as it is reminiscent of a walkie-talkie.

Push and Release: A type of speech interaction where the user must depress a button prior to the start of speech. This type of interaction is unfamiliar to some users, but provides an easy learning curve with the proper affordances.

Recognition lattice: A directed graph of candidate words considered by a speech recognizer. This graph will often contain alternate words with similar phonetics. It will also contain confidence weights.

SILO: A speech-based user interface which returns a shortlist of possible responses, from which the user must make a final selection. We refer to such interfaces as Speech-In List-Out, or SILO.

Speech-Based [User] Interface (SUI): A user interface which uses utterances spoken by the user as a primary input mode. A speech based interface may also have other input modes, such as dedicated or softkey input, and may also have voice feedback and/or visual feedback.

Telematics: Broadly, telematics refers to the combination of telecommunication and computation. More specifically telematics has come to refer to mobile systems which combine wireless data communications with local computation resources. Voice communication and/or location information provided by GPS are often assumed.