

# A Comparison Between Spoken Queries and Menu-Based Interfaces for In-car Digital Music Selection

Clifton Forlines, Bent Schmidt-Nielsen, Bhiksha Raj,  
Kent Wittenburg, and Peter Wolf

Mitsubishi Electric Research Laboratories,  
201 Broadway, Cambridge, MA 02139 USA  
{forlines, bent, bhiksha, wittenburg, wolf}@merl.com

**Abstract.** Distracted driving is a significant issue for our society today, and yet information technologies, including growing digital music collections, continue to be introduced into the automobile. This paper describes work concerning methods designed to lessen cognitive load and distracting visual demands on drivers as they go about the task of searching for and listening to digital music. The existing commercial paradigms for retrieval—graphical or spoken menu traversal, and text-based search—are unsatisfactory when cognitive resources are limited and keyboards are unavailable. We have previously proposed to use error-tolerant spoken queries [26] combined with direct modalities such as buttons mounted on the steering wheel [7]. In this paper, we present in detail the results of an experiment designed to compare the industry standard approach of hierarchical graphical menus to our approach. We found our proposed interface to be more efficient and less distracting in a simulated driving task.

## 1 Introduction

It was estimated in 2001 by the U.S. National Highway Traffic Safety Administration that at least 25% of police reported accidents involve some form of driver inattention. A study by Stutts et al. [24] estimated that at least 13% of the drivers whose state was known at the time of the crash were distracted. Adjusting the audio system of the car accounted for 11% of these distractions. 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. The additional complexity of navigating and selecting music from large music collections while driving is thus a cause of concern.

In general, there are two basic paradigms for retrieving an item from some large set: (1) menu-based traversal and (2) search. For drivers whose hands and eyes are mostly occupied, each of these paradigms has its challenges. Menu-based traversal can be maintained using buttons or touch screens; however, the growing size of the selection set (tens of thousands of songs already) requires hierarchical menus of increasing breadth and/or depth. The need to navigate ever-larger sets of menus may require too much time and visual attention to allow for safe driving.

The combination of speech input with menus might seem promising, and it has been shown to be effective compared to mouse-based menu selections in earlier studies [6][15]. However, speech interfaces need to address not only the issue of misrecognition errors in noisy environments, but also the issue of habitability, i.e., the ease with which a user can stay within the sublanguage understandable by the system. Users need to learn what to say to a speech interface since no speech recognition system can deal with unrestricted language. VoiceXML [25] has been proposed as a rather direct translation of menu selection to the speech domain and addresses the issue of habitability through prompting. However, the enumeration of all menu choices with speech would be time-consuming and frustrating for drivers, and, again, as the size of the set increases, its efficacy diminishes.

More advanced speech interfaces aim to create flexible dialogs in order to avoid the tedium of mechanistic menu traversal and the need for a rigid command syntax. A good example is the automotive interface introduced in [20]. The approach incorporates sophisticated prompting while also allowing for shortcuts once the user has learned the sublanguage of the system. Dialogs are carefully crafted in order to minimize errors and misunderstandings. While promising, a drawback for such approaches is the cost and complexity of development. Each deployment requires extensive language-specific efforts to collect speech natural to the application in question and also requires much iteration to refine the dialogs. Some researchers have suggested that the adoption of universal conventions by speech interfaces could eventually help reduce application and language-specific dialog development [21], but in the meantime, other alternatives should be considered.

What about search UIs? Search interfaces require the entry of a query term and manipulation of a result list. The entry of a query term is the primary challenge in the automobile. For text-based queries, a keyboard is inappropriate for an in-car device, and the entry of text without use of a keyboard is notoriously difficult. Character entry by menu is, again, time-consuming and visually demanding.

A promising approach lies in the utilization of speech in search UIs. Examples proposed previously include Cohen et al.'s system ShopTalk [5]. The historical approach is to prompt the user with a query box, apply speech understanding, and then input the disambiguated result to the query system. The habitability problem is still in evidence here — users need to know the sublanguage of query terms in order for the speech recognizer to achieve successful recognition rates. As the domain becomes less restricted and more “Google like,” the challenge of disambiguating the speech becomes more severe since there are few language constraints that can be imposed on queries.

In previous work, we have proposed using speech input for search UIs without direct conversion to disambiguated text [7][26]. In this approach, the spoken query is converted to a probabilistic query vector that is input directly into the search process. Instead of requiring the spoken input to be converted to an unambiguous form, the spoken input is converted to a probabilistically scored “bag of words” that serves the purpose of defining the query. The user never sees the query specification, but only the list of query results. We call this approach “Speech In, List Out” or SILO.

Our proposal for an automotive UI for digital music selection utilizes the SILO approach in combination with a set of simple buttons for manipulating query results [7].

The buttons can be mounted directly on the steering wheel. 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 when the user is simultaneously involved in other attention-critical tasks. In the remainder of this paper we review the basic elements of the proposed SILO-based interface for an automotive digital music player. We then relate the full details of an experiment in which we compare our interface to a graphical menu-based interface that is today's industry norm. Our preliminary findings support the claims that a SILO-based interface can be more efficient and impose less cognitive overhead than a menu-based graphical one. We finish by discussing the ramifications of this work on future study.

## 2 SpokenQuery

The enabling technology for our experiments is the SpokenQuery speech-based search engine [26]. While a detailed description of SpokenQuery is outside of the scope of this paper, we present a summary here for the convenience of the reader. SpokenQuery is similar to familiar web-based information retrieval engines such as Google, AltaVista, etc., except that the user speaks his or her query instead of typing it. The user may speak whatever words he/she thinks best describe the desired items and there is no rigid grammar or vocabulary. The output of SpokenQuery is an ordered list of items that are judged to be pertinent to the query. As with other IR techniques, there is no guarantee that the desired item(s) will be the top choice(s) in the output list.

Contrary to the conventional approach of using speech recognition to convert the spoken input to disambiguated form, SpokenQuery uses the probabilities in a word lattice returned by a speech engine as input to a probabilistic document index. SpokenQuery uses speech to input a set of words with associated probabilities, which can in turn be used to return a list of best matches.

SpokenQuery stands in contrast to conventional recognition techniques that convert the spoken input into a single disambiguated phrase, or as is often the case, a list of the N-best phrases for the speaker to choose among. Instead, the speech recognizer converts the spoken query into a graph of words, called a lattice, where the nodes correspond to possible words and the edges correspond to the probability of transition between the words. All the words in this lattice are weighted by their probability and used in the search. The output of the system is a list of the N-best documents found by this search. Unambiguous identification of the query words is never required.

A noteworthy feature of recognizers is that the actual words uttered by a user are usually included in the recognition lattice and have high expected counts, even when they are not included in the final word sequence output by the recognizer. As a result, SpokenQuery is able to perform remarkably well in highly noisy conditions (such as automobiles) under which conventional speech UIs that depend on the recognizer's text output break down [7]. Table 1 lists some example phrases and their (often poor) interpretation by the speech recognizer along with the performance of the SpokenQuery search.

## 2.1 A SILO Interface Model

Here we consider a SILO-based UI for an application we refer to internally as MediaFinder. It is intended for retrieving music from large collections using multimodal input. Digital music players designed for automobiles currently allow for 10s of thousands of songs (and growing). The UIs provided on these devices today provide a display of up to about 10 lines of text and a few buttons or touch surfaces to navigate a tree of choices in order to play a desired piece of music. The collection may be navigated by artist/album/song as well as by genre and other organizations. In contrast, MediaFinder is not menu-driven. Instead, it recasts the music retrieval problem as information retrieval: the system responds to spoken requests with a list of songs that are deemed pertinent to that request. Here are four design principles we followed:

**1. Appropriate use of speech.** Speech input is used only for choosing from a very large set when the use of buttons (for scrolling and selection) is inefficient or impossible. All choices from a small set are performed with direct manipulation [13].

**2. Speech in, graphics out.** While speech input is used by the user to describe the desired item(s), the result of the query is rendered as graphics or text. Speech output is not the most efficient way to communicate a list of results to the user—text or graphics is quick to display and convenient to browse.

**3. Pure pull model.** The user interface never prompts the user. It always waits for the user to initiate the next interaction. The pace of the interaction is entirely set by the operator, which is an important feature of an automotive system.

**4. A recognition of the limitations of speech recognition.** In a command and control system, poor speech recognition often results in the system responding with “I did not understand that” or, worse still, with the system issuing an incorrect command. With MediaFinder, the result of poor speech recognition is the same as that of a poor query — a degraded result list (i.e., the desired item(s) may be further from the top or missing all together). As with IR (e.g., Google), it is the responsibility of the user to speak a query that distinguishes the desired item.

## 3 Experiment

While driving, perception and attentional resources are highly constrained. Recent studies have shown significant driving impairment by cell phone use and navigational aids [8]. Any search mechanism that interferes with driving's complex balance of motor, cognitive, and perceptual skills may result in unacceptable performance penalties, leading to unsafe conditions. A successful search in this environment not only means finding the desired information quickly, but also means generating less interference while doing so.

Our initial plan was to compare the SILO speech interface to a command and control speech interface for in-car music selection. At the time of this experiment, we surveyed available in-car and handheld voice activated music players and found no systems that we felt could be used in a meaningful comparison. High error rates and limited storage capacity eliminated all contenders. We considered developing our own using an off-the-shelf speech recognition system, but this proved to be problematic, as

shown in Table 1. The lack of a successful commercial product became in our minds comparison enough.

**Table 1.** While the disambiguated phrase output by the speech recognition system is often wildly inaccurate, SILO manages to return the desired song near or at the top of the list

Driver 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 credence 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

One might pose the question, “*Why compare a speech-based system to a menu-driven system? Clearly, any speech-based system that allows drivers to keep their eyes on the road should “beat” any menu-driven system that demands lots of visual attention, right?*” The answer to this question is unclear in our minds. Cell phone use while driving is now a well known cause of distraction, and even hands-free, eyes-on-the-road dialing does not eliminate the cause for concern. A purely voice-based music-retrieval system with no visual display or manual input might seem like a good idea at first glance, but such a system requires that the operator keep track of the state of the system in their working memory. 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. In-car systems must strive to not only keep their user’s eyes on the road, but also keep their minds on the driving task.

With this in mind, we designed an experiment to test the multimodal SILO interface against what is widely available today, an interface based on hierarchical menus. We compared quantitative measurements of simulated steering and braking while searching for music with the two different interfaces. Our hypotheses were:

**H1:** Subjects will more accurately track a moving target with a steering wheel while searching for songs using the SILO interface than while searching for songs using the menu-driven interface.

**H2:** Subjects will react faster to a braking signal while searching for songs using the SILO interface than while searching for songs using the menu-driven interface.

**H3:** Subjects will be able to find songs faster while using the multimodal interface than while using the menu-driven interface while driving.

### 3.1 A “Driving-Like” Task

Although testing subjects in a real automobile while they engage in highway driving would lead to a more accurate study, ethical considerations prevented us from

exposing subjects to a potentially lethal combination of activities. We relied on a simple driving simulator, such as those in [4][9], that mimicked two important facets of driving — steering and braking. An experiment using a high-fidelity simulator or real automobile is left for future work. Eyes-off-the-road gaze time is difficult and expensive to measure; however, steering and braking measurements are good proxies for gaze since both require visual attention, and may well be a better proxy for driving performance.

The simulator (Figure 1) had both a “windshield” and “in-dash” display. Subjects steered, braked, and controlled the searching interfaces with a Microsoft Sidewinder [16] steering wheel and its gas and brake pedals. A microphone (not shown) 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 [23]. 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.

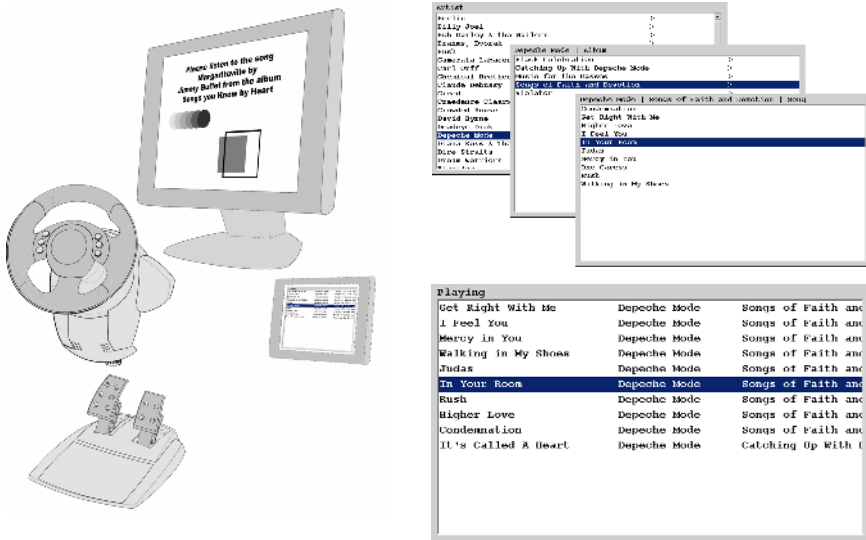


Fig. 1. The hardware setup (left). Subjects controlled the interfaces using the buttons on the steering wheel. The three levels of the menu-driven interface (top right). The SILO interface (bottom right). Three columns display song name, artist, and album for the search results.

### 3.2 Two Music Searching Interfaces

We built two interfaces for this study. The first interface was based on a sampling of currently available MP3 jukeboxes and used a traversable menu organized by artist, album, and song; the second was a SILO multimodal 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.

**The Menu-driven Interface.** The menu-driven interface was designed to be familiar to anyone who has used an MP3 jukebox (such as the Apple iPod [1]) that uses an artist/ album/song hierarchical structure. When searching for a song, the user first scrolls through a list of the artists with music on the device. The user then selects an artist from this list, and the device displays a list of albums by that artist. After traversing and selecting an album, the user is presented with a list of the songs on that album. Finally, the user selects and plays a specific song from that list. By moving in and out within this three-level tree, the user is able to traverse to any song stored on the device. This interface was controlled with four buttons — one for scrolling up in a list, one for scrolling down in a list, one for moving “up” a level (such as moving from a list of an artist's albums to the list of all artists), and one for selecting the currently highlighted artist or album. To simplify the interface, the selection button doubled as a play button when in the song listing. A picture of the menu-driven interface is shown in Figure 1.

Many music jukeboxes can present their content in alternative fashions such as user defined play lists, favorites, anti-favorites, etc., but a comparison between the SILO interface and these methods is left for future study.

**The SILO MediaFinder Interface.** To search for a song using the SILO interface, the user first presses and holds the “push-to-talk” button while speaking a combination of the name of the song, the artist, and the album in any order. The interface then performs the SpokenQuery search, and displays a list of the ten most promising results. The user then scrolls through this list to the desired song, and selects the song to play it. This interface was controlled with four buttons - the “push-to-talk” button, a button for scrolling down, a button for scrolling up, and a play button. A screenshot of the visual portion of the SILO interface is shown in Figure 1.

### 3.3 Method and Procedure

Fourteen subjects participated in this experiment and were paid \$20 for about 45 minutes of their time. Of the fourteen subjects, eight were male and six were female, and their ages ranged from 18 to 37. Four of our subjects spoke English as a second or third language. All but one were regular automobile drivers.

Subjects were first given instructions on how to correctly perform the steering and braking tasks and were given as much time as they wanted to practice “driving”. They were then asked to perform a four-minute driving-only trial during which they performed no music searches.

Next, subjects were instructed to search for and playback specific songs while performing the driving task. Subjects used both the SILO and menu-driven interface, and completed 8 search trials with each condition. Before each block of 8 trials, subjects were given instructions on how to use the current interface and allowed to practice searches while not driving. The order that the interfaces were used, and the order of

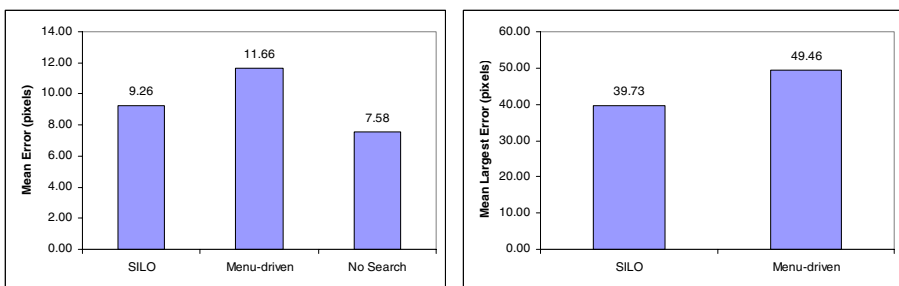
the songs that were searched for were randomized. During each trial, the testing application displayed the steering and braking signals along with instructions that the user would read asking them to search for a specific song (e.g. “Please listen to the song Only the Good Die Young by Billy Joel from the album The Stranger”). Subjects were allowed to take a break between trials for as long as they wished.

The 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. At the end of the session, subjects were asked to fill out a questionnaire designed to measure subjective preference between the two interfaces. Subjects rated their agreement with a collection of statements on a 7- point Likert scale, and listed the “best three things” and “worst three things” about the SILO interface.

### 3.4 Results

Our data supports hypotheses H1 and H3 and rejects H2.

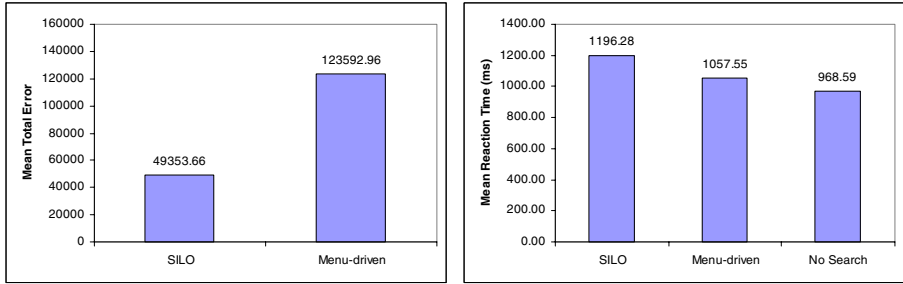
**H1: Subjects were significantly better at the steering task while using the SILO interface than while using the menu-driven interface.** During each trial, the testing application recorded the average distance between the moving target and the subject-controlled frame. Subjects were able to steer more accurately while searching for music using the SILO interface than while searching with the menu-driven interface (on average, 9.2 vs. 11.6 pixels of error respectively,  $t(13) = 3.15$ ,  $p=0.003$ ). Additionally, nine out of our fourteen subjects listed variations of “it was easier to keep your eyes on the screen” as one of the “best three things” about the SILO interface. Searching with the SILO interface was not without its penalties; subjects steered more accurately while driving without searching than while using the SILO interface (on average, 7.4 vs. 9.2 pixels,  $t(13)=2.5$ ,  $p=0.013$ ). The average error for each condition is shown in Figure 2 (*left*).



**Fig. 2.** The SILO interface had both a significantly lower mean steering error (*left*) and a significantly lower mean largest steering error (*right*) than the menu-driven interface

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





**Fig. 3.** The SILO interface had a significantly lower mean total steering error (*left*). There was no significant difference in mean break reaction times between the search conditions (*right*).

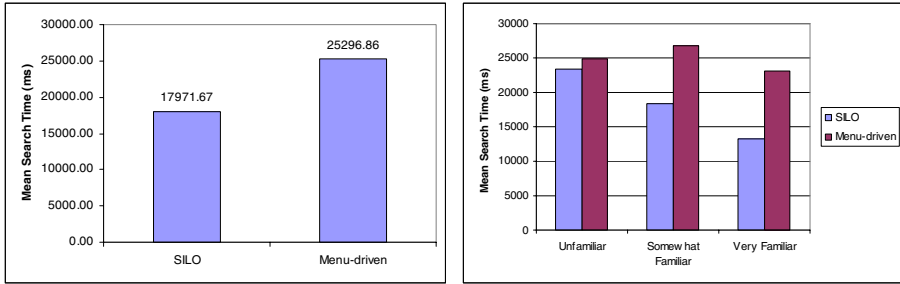
corresponds to the point when the subject was most distracted from the steering task. If actually driving, this point would be the point of greatest lane exceedence. The average maximum error for the two interfaces is shown in Figure 2 (*right*).

Finally, to measure the total steering error attributable to each search, we first subtracted the average no-search error from each of the average search errors and then multiplied these differences by the respective task times. On average, subjects accumulated 2.5 times the error while using the menu-driven system than while using the SILO interface (49,300 vs. 123,500,  $t(13)=1.95$ ,  $p=0.03$ ). The total errors for the two interfaces are show in Figure 3.

**H2: There was no difference in subjects' brake reaction times for the SILO and menu-driven interfaces.** During each trial, the testing application recorded the reaction time of the subjects to the randomly occurring brake stimulus. The mean reaction times were indistinguishable between the multimodal 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 3 (*right*).

**H3: Subjects were significantly faster at finding and playing specific songs while using the SILO interface than while using the menu-driven interface.** For each trial, the test application recorded the time taken from the moment that the instructions appeared on the screen to the moment that the correct song started playing. 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 4 (*left*). 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.

Six out of our fourteen subjects listed variations of “it often put the song you were looking for at the top of the list” as one of the “best three things” about the interface. 35 out of the 56 SILO trials 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.



**Fig. 4.** Subjects were significantly faster at finding songs with the SILO interface (*left*). Familiarity with the music affected the SILO condition, but not the menu-driven condition (*right*).

### 3.5 Non-hypothesized Findings

**Subject's familiarity with the music significantly affected the speed with which they were able to locate a song using the SILO interface, but did not affect the time taken to find a song using the menu-driven interface.** At the end of each session, we asked subjects to rate their familiarity with the music that they had been asked to search for. Being familiar with the search content lead to faster searching using the SILO interface ( $F(2,53)=8.25$ ,  $p=0.0008$ ), but not with the menu-driven system ( $F(2,53)=1.13$ ,  $p=0.32$ ). We speculate that this difference is largely due to knowing the correct pronunciation of names in the database, and we would expect that searching through a familiar set would increase the hypothesized performance differences between the two interfaces. The average time for each familiarity group for each interface is shown in Figure 4 (*right*).

### 3.6 Experimental Discussion

We are pleased to be able to report evidence that our SILO multimodal interface for music finding does have measurable advantages for users operating a simulated automobile over the standard menu-based approach. Although a few other studies have shown advantages in task performance for speech input over mouse and keyboard [15], this is the first as far as we know that has found an advantage for speech input for search tasks. The findings of this preliminary study were encouraging; however, as is often the case, many questions arose.

We were surprised by the lack of any statistical difference between mean brake reaction times between the SILO and menu-driven search conditions. A closer inspection of the results shed light on this issue. Because the time between brake stimuli was randomized, many trials were finished without the subject encountering any brake signals. Because SILO trials were faster than menu-driven trials, subjects were 20% more likely to finish a SILO trial without encountering a single brake stimulus than with the menu-driven interface. These cases, in which no brake reaction times were recorded, did not factor into the mean brake reaction times; however, one would think that shorter tasks times would lead to safer driving. Additionally, after each brake signal and at the beginning of each trial, the amount of time between brake signals

was reset to a random number between 5 and 15 seconds. This minimum of 5 seconds may have unfairly penalized the SILO trials, as the period at the very beginning of the trial during which the subject speaks their query (and can keep their eyes on the road) never contained a brake signal. Only after the query was spoken, and the subject's eyes turned to the in-dash display (where they were focused from the first moment of menu-based trials) did brake signals occur.

The instructions for each task included all of the available information for a particular song. While a song title is enough to perform a SILO search, it would not have been fair to ask subjects to find the song "Never Die" without telling them it is by the artist "Creed" on the album "Human Clay" while using the menu-based interface. An informal evaluation of the SILO interface found that it performed quite well using incomplete and even partially incorrect information. A future study might incorporate searching with incomplete information.

The song library we used for the study contained only 2124 songs. It is now possible to buy a handheld music player that holds over 10,000 songs, and we can count on this number increasing. As the number of available artists, albums, and songs grows, we would expect the time needed to search through a menu-driven interface to grow as well. Since this experiment was conducted, we have increased the size of the database in our prototype. An informal evaluation of the SILO interface searching a database of 250,000 songs shows no noticeable differences in search time.

Several subjects noted in their questionnaires that they would like to be able to search for and play an entire album rather than an individual song. MediaFinder is easily modifiable to handle this additional task by including a heterogeneous collection of individual songs and playlists in its database. Playlists would be indexed by the artist and album name, as well as keywords like "album" and "record". A driver searching for "Beatles, Yellow Submarine" would receive a listing for both the song and the album.

It is well known that speech recognizers have more trouble with female and heavily accented voices than with male and native English speaker voices. We were surprised that the SILO interface performed just as well for female and non-native speakers as it did for our male and native English speakers. A formal exploration of the differences between these groups is left for future study.

Finally, in preparing the study, we found that the menu-driven interface was more susceptible to inconsistencies in the music files metadata. Because the music files had been generated over a long period of time using several conversion tools, the metadata itself was not uniform. For example, music by the group "The B-52s" was erroneously split into many artists: "The B-52s", "B-52s", "The B52s", etc. While problematic for the menu-driven interface, these types of errors do not affect the performance of the SILO interface. For the purpose of this study, we cleaned up the metadata in the database, but these issues must be taken into consideration for any menu-driven interface.

Limitations of this study include the fact that an actual in car environment that included environmental noise was not used. Other tests have shown that the Spoken-Query information retrieval engine is very robust to high levels of environmental noise [7]. We are therefore optimistic about the performance of the SILO interface in real in-car environments, but will have to confirm this expectation with future

experiments. Finally, we look forward to a comparison between SILO and other speech based audio selection systems.

## 4 Conclusion

In this paper we have advocated for a particular model of using speech in interfaces when text-entry methods are unavailable or inadvisable. The paradigm of including spoken queries in SILO interfaces narrows the role of unstructured speech to a search subtask while employing direct manipulation for other functions in the interface. We believe that such a model can address the problems of misrecognition and habitability that limit the use of speech interfaces today, particularly in stressful or cognitively demanding environments such as driving an automobile. As evidence, we conducted an experiment in which we measured task-related performance on a simulated driving task as well as task completion time while asking users to find songs from a music collection. We compared an interface designed with our SILO paradigm with an industry standard graphical menu-based interface and were able to show statistically significant superiority for the SILO approach in two of the measures.

Future work will include the consideration of more complex user interfaces than the one we presented here. Our model was restricted to the simplest type of query in which the users' input is considered to be a "bag of words". It is an open question as to whether this model of direct-manipulation and speech can be extended to include more complex types of queries, which could include negation and other operators. Also, there are other challenges for interfaces in the automobile when more precision is required than in the examples we discussed here. We would like to consider, for example, whether our model can be extended to encompass destination entry in navigation systems, in which a particular street number, for instance, must be specified.

From our perspective, one may consider both search-based and menu-based systems as generators of lists of plausible responses from which the user must choose. The smaller and more accurate the list (where accuracy may be defined as the probability that the desired response is actually in the list), and the smaller the number of intermediate steps to getting there, the more user-friendly and less taxing the system is likely to be. As the set of choices gets larger, menu based systems often tackle the problem by setting up hierarchies. The user must generally navigate several levels of menus to get to the appropriate item. A search-based interface, on the other hand, returns a dynamically configured list of responses to the user's request. If the desired response is not on the presented list, the next list of possible choices can be immediately presented based on the scores attributed to the various choices by the system. In the worst case, the user must repeat the query in order to generate a fresh list of choices; a single step as opposed to the navigation of menus required by a menu-based system.

Safe interfaces for operation of communications and entertainment systems while driving are of concern as the complexity of such systems grows. The multimodal Speech-in List-out (SILO) paradigm shows promise over conventional GUI-based approaches for accommodating the inevitable introduction of large personal music collections into the automobile. The paradigm itself is applicable for retrieval of digital information other than music — recorded news, weather, and other radio pro-

gramming, for instance. An intriguing idea is whether the SILO model that we have presented here can be generalized to handle many more of the functions in off-the-desktop interfaces than just the ones that we think of today as search. It will be interesting to see whether our approach provides an alternative to the mixed-initiative dialog approach, which has captured the attention of most of the researchers in our field, or whether some sort of integration of the flexible dialog approach with SILO will prove most effective.

## References

1. Apple iPod, <http://www.apple.com/ipod/>.
2. Baeza-Yates, Ricardo, Ribeiro-Neto, Berthier, (1999) *Modern Information Retrieval*. Addison Wesley, p. 27.
3. Bolt, R.A. (1980) Put-that-there: Voice and Gesture at the Graphics Interface, *Computer Graphics* 14,3, 262 - 270.
4. 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.
5. Cohen, P. (1991) Integrated interfaces for decision-support with simulation. In *Proceedings of the 23rd Winter Conference on Simulation*, Phoenix, Arizona, pp. 1066 - 1072.
6. Cohen, P. R., McGee, and D. R., Clow, J. The efficiency of multimodal interaction for a map-based task. *Proc. Applied Natural Language Processing Conference (ANLP'00)*, Seattle, WA, April 29-May 4, 2000, Morgan Kaufmann, pp. 331-338.
7. Divi, V., Forlines, C., Van Gemert, J., Raj, B., Schmidt-Nielsen, B., Wittenburg, K., Woelfel, J., Wolf, P., and Zhang, F. (2004) A Speech-In List-Out Approach to Spoken User Interfaces, in *Proc. Human Language Technologies 2004*, Boston MA.
8. Driver distraction with wireless telecommunications and route guidance systems. [DOT-HS-809-069]. Washington, DC: USDOT, NHTSA.
9. Driving Simulators, [http://www.inrets.fr/ur/sara/Pg\\_simus\\_e.html](http://www.inrets.fr/ur/sara/Pg_simus_e.html)
10. Geller, V.J., and Lesk, M.E. (1983) User Interfaces to Information Systems: Choices vs. Commands, in *Proc. ACM SIGIR*, 130-135.
11. Green, P. (1997). Potential safety impacts of automotive navigation systems, in *Proceedings of the Automotive Land Navigation Conference*, June 18, 1997.
12. Hauptmann, A. G. (1989) Speech and Gestures for Graphic Image Manipulation. In *Proc. CHI '89*, pp. 241-245.
13. Hutchins, E.L, Hollan, J.D., and Norman, D. A. (1986) Direct Manipulation Interfaces, in D. A. Norman and S. W. Draper (eds), *User-Centered System Design*, Lawrence Erlbaum, 87-124.
14. Kaljuvee, O., Buyukkokten, O., Garcia-Molina, H., and Paepcke, A. (2001) Efficient Web form entry on PDAs, *Proc. World Wide Web Conf.*, 663 - 672.
15. Leatherby, J.H., and 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.
16. Microsoft Sidewinder Wheels - Precision Racing Wheel, <http://www.microsoft.com/hardware/sidewinder/PrecPro.asp>.
17. Oviatt, S.L. (2003) Multimodal Interfaces, in J. Jacko and A. Sears (eds.), *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, Lawrence Erlbaum, 286-304.

18. Oviatt, S.L. (2002) Breaking the Robustness Barrier: Recent Progress on the Design of Robust Multimodal Systems, in M. Zelkowitz (ed.), *Advances in Computers*, 56, 305-325.
19. Oviatt, S.L., Cohen, P., Vergo, J., Duncan, L., Suhn, B., Bers, J., Holzman, T., Winograd, T., Landay, J., Larson, J., and Ferro, D. (2000) Designing the User Interface for Multimodal Speech and Pen-based Gesture Applications: State-of-the-Art Systems and Future Research Directions, *Human Computer Interaction* 15,4, 263-322.
20. Pieraccini, R., Dayanidhi, K., Bloom, J., Dahan, J.-G., Phillips, M., Goodman, and B. R., Prasad, K. V. Multimodal Conversational Interface for a Concept Vehicle. *Proc. Eurospeech 2003*, pp. 2233-2236.
21. Rosenfeld, R., Olsen, D., and Rudnicky, A. (2000) Universal Human-Machine Speech Interface: A White paper. Technical Report CMU-CS-00-114, School of Computer Science, Carnegie Mellon University.
22. Silfverberg, M., MacKenzie, I.S., and Korhonen, P. (2000) Predicting text entry speed on mobile phones, in *Proc. SIGCHI*, 9-16.
23. 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*.
24. Stutts, J.C., Reinfurt, D.W., Staplin, L.W., and Rodgman, E.A. (2001) *The Role of Driver Distraction in Traffic Crashes*. Washington, D.C.: AAA Foundation for Traffic Safety. Full text at: <http://www.aaafoundation.org/pdf/distraction.pdf>.
25. VoiceXML Forum, <http://www.voicexml.org>
26. Wolf, P.P., and Raj, B. (2002) The MERL SpokenQuery Information Retrieval System: A System for Retrieving Pertinent Documents from a Spoken Query, in *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, Vol. 2, 317-320.