Dialog Systems for Automotive Environments

Julie A. Baca, Feng Zheng, Hualin Gao

Center for Advanced Vehicular Systems
Mississippi State University
baca@cs.msstate.edu, {zheng,gao}@isip.misstate.edu

Joseph Picone

Institute for Signal and Information Processing
Mississippi State University
picone@isip.msstate.edu

Abstract

The Center for Advanced Vehicular Systems (CAVS), located at Mississippi State University (MSU), is collaborating with regional automotive manufacturers such as Nissan, to advance telematics research. This paper describes work resulting from a research initiative to investigate the use of dialog systems in automotive environments, which includes in-vehicle driver as well as automotive manufacturing environments. We present recent results of an effort to develop an in-vehicle dialog prototype, preliminary to building a dialog system to assist in workforce training in automotive manufacturing. The overall system design is presented with focus on development of the semantic information needed by the natural language and dialog management modules. We describe data collection and analysis through which the information was derived. Through this process we reduced the parsing error rate by over 20% and system understanding errors to 3%.

1. Introduction

The Center for Advanced Vehicular Systems (CAVS), located at Mississippi State University (MSU), is collaborating with regional automotive manufacturers such as Nissan, to advance telematics research. Many applications within this domain require speech and other related human interface technologies. While speech-based in-vehicle driver assistance has received the most attention in the media, the automotive manufacturing environment presents many challenging situations for which speech interfaces can provide unique solutions. CAVS, working closely with Nissan, is currently investigating all aspects of the manufacturing environment, including particularly the need to train workers to perform optimally in this environment.

In this paper, we describe the development of a dialog system for in-vehicle navigation which is part of a larger initiative at CAVS in the area of workforce training. There are several commonalities between the in-vehicle and manufacturing tasks, including the need for hands and eyes to be occupied with other tasks, the necessity of operating in a noisy environment, and the need for real-time system response. The in-vehicle task, however, is more clearly defined and understood, making it an attractive start-up application for acquiring a base of knowledge and building an infrastructure for dialog systems development that could be applied to the workforce training or other domains. In addition, we plan to provide the driver assistance capability via telephone to the local community since no such service exists in this region, and this represents an attractive framework in which we can study dialect, accent, and other known problems with contemporary speech to text systems.

Speech-enabled navigational interfaces have been studied in a variety of applications for the last decade [1,2]. While some provide assistance for pedestrians , speech is particularly appropriate for the driver assistance task, where the user's hands, eyes and cognitive state, are more fully occupied. Interfaces for this task can be characterized as giving spoken directions in response to a spoken user request for help in navigating a particular geographical area. Interactions with such systems, however, remain highly constrained by the limitations of the speech understanding process as well as the human-computer interface. The current state-of-the-art for such interfaces offers users a restricted, inflexible interaction at best. Users must adhere to a strict protocol with any variations causing system response to degrade significantly.

True dialog systems that allow users to engage in a natural conversational interaction have yet to be realized for the driver assistance task. Many fundamental problems must be solved to achieve this goal. In particular, this task heightens the need for real-time, accurate responses to complex queries in a noisy environment. This paper presents research exploring these issues in a test-bed developed within CAVS. Challenges in the development of an in-vehicle dialog system are presented and results of current efforts analyzed. Issues to be addressed in future work conducted in both this project and related projects for workforce training are also identified.

2. Dialog System Development

Our prototype dialog system allows users to speak queries using natural unconstrained speech to obtain information needed to navigate the Mississippi State University campus and the surrounding town (Starkville, Mississippi). Queries can be tightly focused: "I'm at the Post Office. I want to go to the cafeteria downtown on Main Street," or more open: "What's the best place to stay near campus?" The system employs speech recognition, natural language understanding, and user-centered dialog control to resolve queries.

Flexible dialog management, focused on the user's goals, is critical to understanding and properly responding for such an application. For example, the first query, "I'm at the Post Office. Tell me how to get to the cafeteria downtown on Main Street" requires the system to know or determine to which Post Office the user is referring, the one in town or on campus.

Related efforts have discussed issues in the development of in-vehicle dialog research prototypes [3-5]. As noted in [6], language model and grammar development remain costly endeavors for any dialog system, requiring either handcrafting or automatically learning from large amounts of training data. The DARPA Communicator program, however, has facilitated overall system design and development by providing a standard architecture for building dialog systems, thus enabling the exchange of components and evaluation of results across sites [7]. Our prototype system uses this Hub architecture, shown in Figure 1, with base components derived from other publicly available system modules [8,9]. A hub and a set of servers comprise the Communicator architecture. The hub functions as a central router through which the servers

pass messages to communicate with each other. The hub accepts these messages from the servers in a specific format.

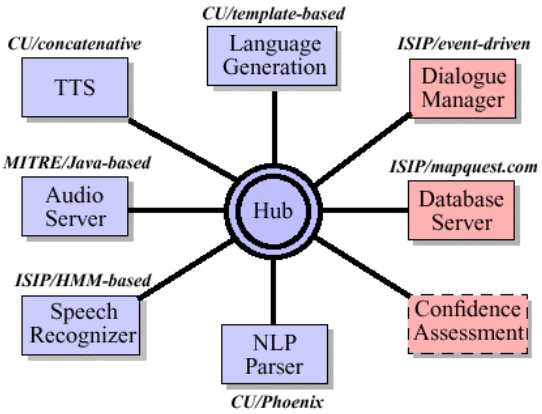

Figure 1: The DARPA Communicator Architecture

This enforces a standard protocol for all servers to communicate, thus encouraging a "plug-and-play" approach to dialog system development. Designing the server modules to follow the message-passing protocol improves modularity. A description of each of the major components of our system follows.

2.1. Audio Server

The audio server controls the hardware which samples and digitizes acoustic signals from the microphone. It accepts these incoming speech signals and passes the result via the hub to the automatic speech recognition (ASR) module. It also sends synthesized speech through the hub to the audio device. Our system uses the MIT/MITRE audio server [7] provided to DARPA Communicator participants.

2.2. Speech Recognition

The ASR module uses a publicly available recognition toolkit [9] that implements a standard HMM-based speaker independent continuous speech recognition system. It supports cross-word context-dependent phonetic models and employs a generalized hierarchical time-synchronous Viterbi beam search for decoding. This produces a single most likely word sequence that is passed through the hub to the natural language understanding (NLU) module.

The recognizer processes acoustic samples received from the audio server. For our application, we began with acoustic models trained on Wall Street Journal. We chose this set initially because it is a medium size general-purpose set of acoustic models upon which we could build. We then collected minimal training data for our application and retrained these models.

To create the language model for our application, we first collected over 2,000 business and other special names specific to our application. We then used the SRI Language Modeling Toolkit [10] to interpolate the new domain-specific words into the WSJ language model (LM). We first trained a class-based language model for the new words using phrase patterns observed in collecting a pilot corpus described in Section 3. We then expanded the class-based LM to the word-based LM, where the probabilities are recomputed by:

$$\text{Unigram: } p(W_i) = p(W_i|C_i) * p(C_i) \quad (1)$$

$$\text{Bigram: } p(W_i|W_{i-1}) = p(W_i|C_i) * p(C_i|C_{i-1}) \quad (2)$$

Next we interpolated the word-based LM with the original WSJ LM, adjusting the weight of the interpolation to ensure the perplexity of the new LM was close to the original LM. The resultant vocabulary increased from 4988 words to 7132 words, while the perplexity of the LM on the pilot corpus increased by 70. Further details of the pilot corpus are given in Section 3.

The ASR module produces the most likely word sequence and sends this string via the hub to the NLU module, described in the next section.

2.3. Natural Language Understanding

The NLU module uses a publicly available semantic case frame parser [8]. It employs a semantic grammar consisting of case frames with named slots. A context free grammar (CFG) specifies the word sequences for each slot. The grammars are compiled into Recursive Transition Networks, against which the recognizer output is matched to fill the slots. A semantic parse tree is generated for each slot with the slot name as root. A simple example frame for a request for driving information is shown below

```
FRAME: Drive
    [route]
    [distance]
```

The first slot in this frame, [route], allows asking for directions along a specific route, while the second slot, [distance], allows asking for the distance from one location to another. A subset of CFG rules for the route slot are shown below:

```
[route]
    (*IWANT *[go_verb]  [arriveloc])
IWANT
   (I want *to) (I would *like *to) (I will) (I need *to)
 [go_verb]
    (go) (drive *to) (get) (reach)
[arriveloc]
    [*to [placename] [cityname]]
```

This type of grammar is advantageous for dialog systems because it can accept the ungrammatical inputs likely to occur in spontaneous speech. For example, the rules shown above would accept the input, "I would like…I.. need to go to the Post Office on campus." This flexibility reduces the need for users to adhere to strict syntactic correctness in their requests.

As previously discussed, grammar and language model development are highly domain specific, making these costly efforts for dialog system development. The semantic grammar supplied with the parser toolkit was developed for a general purpose travel domain; thus, development of frames and slots specific to our application was required. Our current semantic grammar consists of approximately 500 rules and over 2000 words. It was developed, along with the language model, from a corpus of 276 sentences spontaneously entered by users over a series of three pilot tests, described more fully in Section 3.

Once parsing is complete, the NLU module sends a list of possible sentence parses via the hub to the Dialog Manager to determine meaning and resolve the query. The semantic frames defined for the parser are integral to this resolution.

Semantic frame development as well as the Dialog Manager structure are described in the following section.

2.4. Dialog Manager

The Dialog Manager (DM) determines the nature of the interaction between the user and system. First, it initiates the dialog, prompting the user for an initial query. The user speaks a request that is received by the recognition module, and parsed by the NLU module. The NLU module then passes a list of possible sentence parses to the DM, which selects the best possible parse, based on the scoring of the slots, and maps this to its own internal set of frames. It then merges this result with a set of context frames it maintains and determines what action to take in response. If there is no missing or conflicting information, the DM forms a database query, obtains the result from the database, and passes it to the NL module for output. Otherwise, it asks the user to clarify missing or conflicting data and attempts again to resolve the request.

Our DM was derived from the toolkit provided in [8]. It follows a declarative design which means that, similar to the NLU module, the bulk of the development effort lies in the construction of domain-specific frames, forms, and grammars. Modifications to the DM code base consisted primarily of those needed to process domain-specific information. This approach offers many advantages, in particular, a reduction in the complexity of the DM code base. Nonetheless, design of the semantic frames can critically affect system performance. Using the fewest frames and slots provides efficiency but does not yield a system robust to unexpected input. Conversely, using too many frames and slots can degenerate to keyword spotting. Finding the proper balance between these two extremes requires careful experimentation and analysis.

Our application provides both general-purpose information to queries such as "Is there a hiking trail here?" as well as information specific to drivers, e.g., "Can I drive to the coliseum from the drill field?" We began with a single Drive frame with multiple slots to represent all types of queries a user might ask about driving in a particular area and a single Info frame with multiple slots for general information. The relative simplicity of the driving task warranted the initial choice of a single frame with carefully selected slots, subslots and associated CFG's. Analysis of the pilot corpus, however, indicated the single Drive frame did not provide sufficient robustness to unexpected input. We replaced the single frame with 9 separate frames, each related to different types of queries a driver might pose. Example frames and the associated queries the frames can handle are shown in Table 1.

To summarize, the DM controls the interaction of the user with the system, determining what data the user requires, obtaining the data, and finally presenting it to the user. Once the DM has determined the data needed by the user, it must obtain this data via the hub from the application back end.

2.5. Application Back End

Our application back end houses a database storing information about the MSU/Starkville area. Initially we used an Internet map routing program to obtain the data and store it in a local relational database. We used this as an interim tool only, however, in order to focus more effort on developing other system modules. Though expedient, its coverage of the campus and local area were not sufficient. We have since incorporated a Geographic Information System (GIS) module, which stores data captured from a global positioning system (GPS). The GIS provides the route information, which is then stored in the local database. We are incorporating real-time GPS data capture in the next release of the system.

Drive_Direction:	"How can I get from Lee Boulevard to Kroger?"
Drive_Address:	"Where is the bakery located?"
Drive_Distance:	"How far is China Garden from here?"
Drive_Turn:	"I'm on the corner of Nash and Route 82. What's the next turn to get to campus?"
Drive_Quality:	"Find me the most scenic route from LJ's to Scott Field."
Drive_Intersect:	"Does Lynn Lane intersect Academy Road?"
Drive_Special:	"Can I bypass Highway 12 to get to Bryan Field?"

Table 1: Example frames and the associated queries.

3. Pilot Experiments

To obtain the domain-specific data needed to develop the semantic grammar and language model, we conducted a series of three pilot experiments, during which users were asked to spontaneously enter requests for information about the university campus and surrounding town. Each of the three pilot experiments consisted of two phases, 1) initial data gathering and system testing, followed by 2) retesting the system on the initial data after enhancements were made to the grammar and language model. Initial efforts focused on reducing the rate of out of vocabulary (OOV) utterances and parsing errors for the NLP module. The initial lexicon contained over 2000 words. The initial semantic grammar contained a Drive frame with two slots, each expandable to multiple subslots as shown:

```
FRAME Drive:
   [Depart_Loc]
   [Arrive_Loc]
```

each of which can expand to [City_name], [State],[Place_Type], [Place_Name], [Address]. Each of these subslots can be further expanded, for example, [Place_Name] can expand to [(Restaurant_Name)], [(Hotel_Name)], [(Campus_Name)], [(Building_Name)].

After each test, new domain-specific words occurring in the pilot data were added to the lexicon. In addition, new slots and CFG rules were added to the Drive frame to increase the robustness of the parser. For example, to handle a request such as "What's the best way to get to the courthouse?" a new slot and associated CFG's were added. The slot and an excerpted subset of the rules added include:

```
[query_best]
   (WHATS * THE * SPEC_WAY  WAY)
   SPEC_WAY
         (best) (shortest) (better)(nearest)(closest)(fastest)
   WHATS
         (what's)(what is)(where *)(which *is)(which *are)
```

The results for each of the three pilot experiments are given in Table 2:

Vers.	**1.0**		**2.0**		**3.0**	
Test	Pre	Post	Pre	Post	Pre	Post
OOV	25%	0%	36%	0%	4%	0%
Parser	80%	3%	60%	5%	46%	11%

Table 2: A summary of the results from three pilot studies to refine the NLU system.

After completing the pilot experiments, two overall system tests on the entire corpus were conducted for refinement. Results of these tests are given in Table 3:

Test No.	NLP Parser Error Rate	Dialog Manager Error Rate
1	43%	49%
2	6%	3%

Table 3. The results from two pilot studies on the overall system performance.

Prior to the first test, the single Drive frame was implemented. As described in Section 2.5, analyses of the first overall test results indicated the number and type of semantic frames should be increased to reduce the parsing error rate. Even with multiple slots, subslots and carefully crafted associated CFG rules, the single Drive frame did not provide sufficient robustness. The addition of the new semantic frames reduced the parsing error rate from 43% in the first overall test to 6% in the second overall test as shown in Table 3.

We also measured DM performance in the overall tests. Changes to the DM code base to include the new semantic frames reduced the system understanding errors from 49% to 3%. The query, "Is Dean's still on University Drive?" provides an example of the type of error the DM still could not resolve even after including the new frames. The correct answer, "Dean's is located at 134 E. Amite St." may not be given. "Dean's" is a proper noun represented in the grammar as both a place of business in town and the location of the academic dean for a college. The DM resolves such ambiguities by examining context of place, i.e., town or campus. If the previous context is campus, "Dean's" is interpreted as an academic dean rather than a place of business. If no previous context exists, any address given in the query is used to establish context. In this case, "University Drive" resolves to campus context. Such issues must be addressed and are problematic since they affect more basic issues of system usability, e.g., the level of clarification required by the user as well as the level of user initiative.

4. Conclusions and Future Work

We undertook the in-vehicle driver assistance project to explore the challenges in dialog system development and build a capability for further research in the workforce training and other related domains. For the ASR module of our final prototype, we anticipate a real-time system with a vocabulary of approximately 5000 words, perplexity of 50, and WER of 10%. For the NLU module, we anticipate a semantic grammar of approximately 15 frames and a parsing error rate of less than 10%. One interesting finding of our work concerned verification of the cost incurred by the domain-specific development required for dialog systems. An area of future research will explore methods for automating this process using statistical techniques. Advances in this area could greatly reduce development time. We intend to investigate this as we transition to the workforce training domain.

We are also implementing several enhancements to the in-vehicle system. As noted, we have incorporated a GIS database to contain the routing information and are developing the capability for real-time GPS data capture. To enhance the recognition module, we are collecting further data to refine the acoustic models and the language model. We plan to analyze this data to improve the robustness of the NLP and the DM modules as well. Our current prototype provides information in NL text output only. Our final prototype will integrate NL generation with speech output. Finally, we intend to make the current system available to the local community via telephone access, which would enable drivers to obtain assistance in navigating the local area through cellular telephones.

5. References

1. Loomis *et al.*, "Personal Guidance System for the Visually Impaired," *ASSETS '94, ACM Conference on Assistive Technologies*, Los Angeles, CA, USA, November 1994.
2. J.R. Davis and C. Schmandt, "The Back Seat Driver: Real Time Spoken Driving Instructions," *First Vehicle Navigation and Information Systems Conference*, Toronto, Ontario, Canada, September 1989.
3. D. Buhler, W. Minker, J. Haubler, S. Kruger, "Flexible Multimodal Human-Machine Interaction in Mobile Environments," *Proceedings ICSLP*, Denver, CO, USA, September 2002.
4. B. Pellom, W. Ward, J. Hansen, K. Hacioglu, and J. Zhang, X. Yu, and S. Pradhan, "University of Colorado Dialog Systems for Travel and Navigation," *Proceedings of the 2001 Human Language Technology Conference (HLT-2001)*, San Diego, CA, USA, March 2001.
5. P. Geutner, M. Denecke, U. Meier, M. Westphal, and A. Waibel, "Conversational Speech Systems for On-Board Car Navigation and Assistance," *Proceedings ICSLP*, Sydney, Australia, December 1998.
6. S. Young, "Talking to Machines (Statistically Speaking)" *Proceedings ICSLP*, Denver, CO, USA, pp. 9-16, September 2002.
7. "DARPA Communicator," http://fofoca.mitre.org/, The MITRE Corporation, 2003.
8. B. Pellom, W. Ward, S. Pradhan, "The CU Communicator: An Architecture for Dialogue Systems," *Proceedings ICSLP*, Beijing China, November 2000.
9. J. Picone, *et al*., "A Public Domain C++ Speech Recognition Toolkit,", ISIP, Mississippi State University, Mississippi State, MS, USA, March 2003 (http://www.isip.msstate.edu/projects/speech).
10. SRILM – The SRI Language Modeling Toolkit: http://www.speech.sri.com/projects/srilm.