Improving clarification dialogs in speech command systems with the help of user modeling: A conceptualization for an in-car user interface

Lars Libuda
Dept. of Technical Computer Science / RWTH Aachen
Ahornstr. 55, 52074 Aachen, Germany
E-Mail: libuda@techinfo.rwth-aachen.de

Abstract: Modern cars contain increasingly complex driver information systems. To handle such a system in a safe and comfortable way new approaches for in-car user interfaces have to be developed. A special focus lies on speech input because it promises the most comfortable operation of the system without distracting the driver from traffic. But because speech input is not reliable enough, an error management for handling recognition errors is needed. This paper introduces a framework for an in-car user interface, presents requirements for an error management and an approach based on user modeling with Bayesian networks.

1. Introduction

Today more and more applications which are supposed to support and entertain the driver are integrated in modern cars (e.g. navigation system, car office, CD player, etc.). This leads to increasingly complex driver information systems. To enable the driver to use all functions in a comfortable and intuitive way, new approaches for an in-car user interface have to be developed. At the department of Technical Computer Science a framework for such a new interface was developed in close cooperation with BMW. The general architecture of this framework is shown in figure 1.

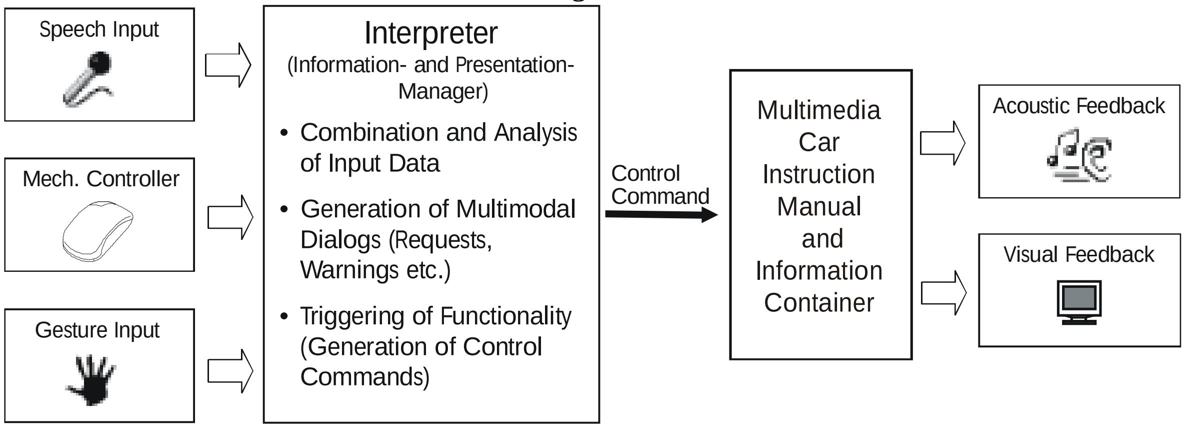
Speech Input
Mech. Controller
Gesture Input
Interpreter
(Information- and Presentation-Manager)
• Combination and Analysis of Input Data
• Generation of Multimodal Dialogs (Requests, Warnings etc.)
• Triggering of Functionality (Generation of Control Commands)
Control Command
Multimedia Car Instruction Manual and Information Container
Acoustic Feedback
Visual Feedback

Fig. 1: Architecture of the developed in-car user interface

We provide three independent input modalities. Speech input (activated via a Push-To-Talk button in the steering wheel), input via a mouse-like mechanical input device, and gesture input.
The core of the system is the interpreter, which collects all messages from the input sources and generates control commands for the connected applications. In a first step two applications were developed: a multimedia car instruction manual and an information container which consists of an email- and a traffic news client. All connected applications have the possibility to present their output visually and acoustically.
A part of this approach is already realized in BMW's iDrive[1]. For input it uses the mechanical device only, which is placed next to the gear lever. A similar framework was also developed by Audi[2].

[1] See http://www.anewwaytodrive.com

2. Requirements for speech input in cars

Since the driving activity should always have the highest priority for safety reasons, current research focuses on speech input in addition to mechanical input. By using speech input it is possible to control an application (e.g. the radio) without taking one's hands off the steering wheel and averting one's eyes from traffic. That's why speech input promises even less distraction than the use of mechanical input.

This, however, can only be reached with a perfect speech recognition which doesn't exist so far. This means recognition errors occur frequently and lead to unexpected system reactions which in turn may confuse the user and distract him from traffic.

So, a speech enabled application mustn't ignore recognition errors in order to reach a robust and acceptable performance. That's because an error management for error prevention and error handling is necessary.

Because driving must have the highest priority there are strong requirements and restrictions for the error management in an in-car user interface:

- The error management has to be as reliable as possible[3] and doesn't have to be obtrusive.
- In case of recognition errors it has to try to keep the interaction time as short as possible in order to minimize the risk of distraction during interactive error recovery.
- The design of the error management has to follow the guidelines found in literature (e.g. [4, 5]). In [3] the most important aspects are summarized in 7 commandments.

3. An error management using dynamic Bayesian networks

This approach is a first initial sketch and doesn't represent the solution necessarily. It is based on ideas realized in a similar system for a desktop application [1].

First of all, this approach doesn't use a vocabulary management like other systems do (e.g. the VODIS project [3, 8] and the error management for the introduced framework above [6]). The vocabulary management adapts the recognizer's vocabulary to the system's state to improve the recognition performance. This reduces the quality of feedback in case the user uttered a command which isn't valid in the current state because this command is currently out of vocabulary. In some cases this method may also restrict the user in his free choice of functionality.

Instead, several instances of the speech recognition engine are created and run in parallel. Each instance handles the vocabulary of a system state and returns its result with a corresponding confidence measure every time the user uttered a command (fig. 2). In this way, the error management can give detailed help if the user uttered a command which isn't valid in the current state.

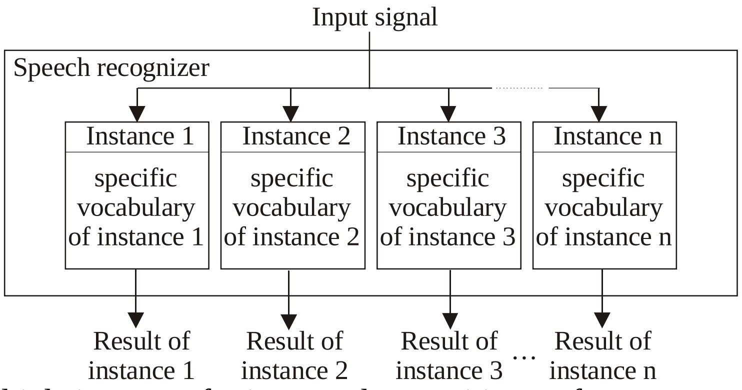

Fig. 2: Using multiple instances for improved recognition performance

[2] See http://www.spiegel.de/auto/news/0,1518,144582,00.html

[3] This depends directly on the reliability of the speech recognition engine.

A disadvantage of this approach is the increasing processing time and memory consumption with each new instance.
After obtaining all results each of them is mapped onto a corresponding action (e.g. “yes”, “ok”, and “do it” are mapped to “confirmation”). This is done because each action can be triggered with several synonyms. Possible actions are: confirmation, rejection (undo), navigation, and activation[4]. These actions are fed with their confidence into a dynamic Bayesian network[5] (fig. 3).

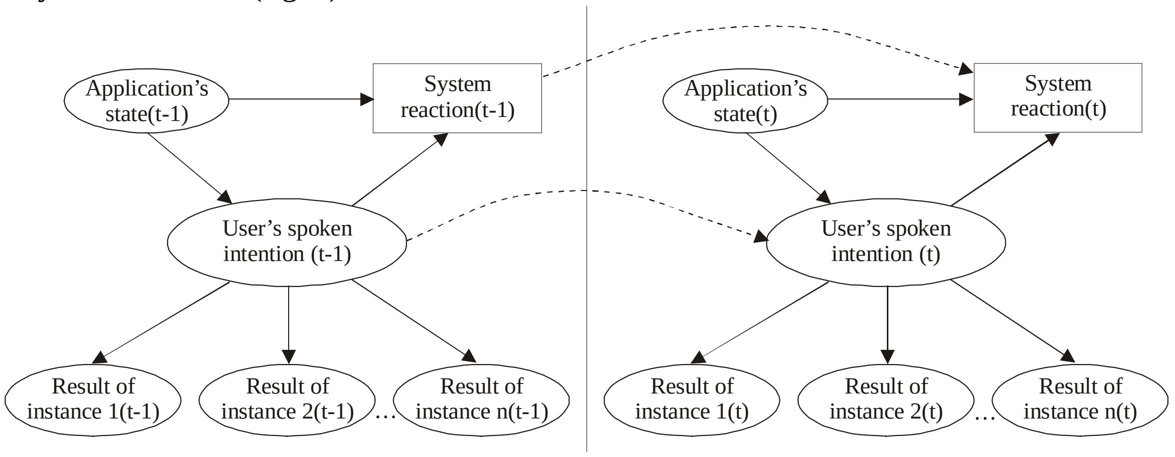

Fig. 3: Two time slices of the dynamic Bayesian network for reasoning about the system reaction. Dashed arcs show the temporal dependencies.

From all actions the user’s spoken intention is derived. The reaction of the error management (decision node “system reaction” in fig. 3) depends on the user’s spoken intention, the application’s state and previous reactions. Possible reactions of the error management are:

- Execute the recognized action directly, which is the best case.
- Execute with delay to give the user the possibility to abort the action.
- Ask for confirmation before executing the action.
- Tell the user that the action is invalid in the current state.
- Troubleshoot.

Troubleshooting means that the system gives hints in case all recognition results are too bad to safely apply one of the other reactions. The reasons for applying this method can be noisy background, unknown vocabulary, or speaking not loud enough.
The feedback of each reaction is very important because good feedback ensures the comprehensibility of a system. In this application acoustic feedback is preferred, because too much visual feedback may distract the user from traffic. In addition, this feedback is only needed once so that the transient character of spoken feedback can be neglected.
The success of this approach mainly depends on a good specification of the conditional probability tables (cpts). With a Wizard of Oz simulation the initial data will be gained to learn the cpts. The subjects will have to evaluate the reactions of the systems afterwards. A remaining problem will be, that the evaluation is subjective. So, it is hardly possible to develop a system which will be accepted by every driver but by the majority of drivers in case of a representative sample.

[4] In case of a rejection, the recognized utterance will be disabled in the corresponding instance so that it won’t be recognized until the user has confirmed an utterance. The actions “navigation” and “activation” are parameterized to distinguish how to navigate (up, down, etc.) and what to activate.
[5] Dynamic Bayesian Networks are probabilistic dependency models which also include temporal dependencies among random variables at different times [7, 2].

4. Example

Let’s consider a simple example: The mentioned information container (fig. 1) includes a client for traffic news and an email client, which is currently active. Here the user can delete a mail, listen to it, and send a voice mail. In the following, the user’s utterances are marked with “U” and system reactions with “S”. There are four instances of the recognition engine (for email, traffic, confirmation, and rejection). Their results including confidence measures are printed in brackets after the user’s utterance.

<U1>: “Jim’s mail.” (“Tim’s mail”:82, “Next traffic news”:34, “undo”:7, “yes”:4)
The system decides to read Tim’s mail and gives appropriate feedback.
<S2>: “Reading Tim’s mail.”
<U3>: “No.” (“no”:67, “yes”:23, “New message”:4, “Delete traffic news”:3)
The user rejected the utterance. The utterance “Tim’s mail” is excluded from recognition.
<U4>: “Jim’s mail.” (“Jim’s mail”:87, “Next traffic news”:33, “abort”:11, “Do it”:7)
The user repeated the utterance. Because of the previous rejection, the system asks for a confirmation.
<S5>: “Read Jim’s mail?”
<U6>: “Yes” (“Yes”:74, “cancel”:13, “Delete mail”:9, “Next news”:6)
The user confirms. All previous excluded utterances are activated again.
After listening to the mail, the user drives some time and forgets that the system is still in the email-state.
<U7>: “Next traffic news.” (“Next traffic news”:82, “Tim’s mail”:42, “undo”:10, “yes”:6)
<S8>: “Do you want me to switch to traffic news?”
The system understood the utterance and offers to switch to the traffic news container.

5. Summary

In this paper a new approach for an error management which handles recognition errors of a speech recognition engine in an in-car user interface is presented. It is based on a dynamic Bayesian network to reason about the appropriate system reaction. The use of multiple speech engine instances which don’t restrict the vocabulary allows the system to give feedback of a higher quality. In combination both methods increase the system’s comprehensibility and translucency.

References

[1] E. Horvitz and T. Paek (2001). Harnessing Models of Users’ Goals to Mediate Clarification Dialogs in Spoken Language Systems. In *Proceedings of the Eighth International Conference on User Modeling (UM2001)*. Sonthofen, Germany. 3 - 13

[2] K. Kanazawa, D. Koller, and S. Russel (1995). Stochastic simulation algorithm for dynamic probabilistic networks. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence (UAI-95)*. Montreal, Canada. 346 - 351

[3] E. Krahmer, J. Landsbergen, and X. Pouteau (1997). How to obey the 7 commandments for spoken dialogue?. In *Proceedings of the Workshop on Interactive Spoken Dialog Systems: Bringing Speech and NLP Together in Real Applications*. Madrid, Spain. 82 - 89

[4] W. Lea (1994). Developing usable voice interfaces. *Journal of the American Voice I/O Society*, 16

[5] R. Leiser (1993). Driver-vehicle interface: dialogue design for voice input. In A. Parkes, S. Franzen (eds.): *Driving future vehicles*. Taylor & Francis. 275 – 294

[6] L. Libuda (2001). Adaptive Error Management for Speech Command Input in Cars. *Proceedings of the Workshop on Context-Aware Applications*. In press

[7] A.E. Nicholson and J.M. Brady (1994). Dynamic belief networks for discrete monitoring. In *IEEE Transactions on Systems, Man, and Cybernetics*, 24(11). 1593 – 1610

[8] X. Pouteau, E. Krahmer, and J. Landsbergen (1997). Robust Spoken Dialogue Manager for Driver Information Systems. In *Proceedings of the fifth European Conference on Speech Communication and Technology* (Eurospeech ’97), Volume 4. Rhodes, Greece. 2207 – 2210