Integration and test environment for an in-vehicle dialogue system in the SIMSI project

Staffan Larsson, Sebastian Berlin
University of Gothenburg
Box 200
SE-405 30 Gothenburg
Sweden
sl@ling.gu.se
sebastian.berlin@gu.se

Anders Eliasson
Mecel AB
Box 140 44
SE-400 20 Gothenburg
Sweden
anders.eliasson@mecel.se

Fredrik Kronlid
Talkamatic AB
Första långgatan 18
SE-413 28 Gothenburg
Sweden
fredrik@talkamatic.se

Abstract

The goal of the SIMSI (Safe In-vehicle Multimodal Speech Interaction) project is threefold. Firstly, to integrate a dialogue system for menu-based dialogue with a GUI-driven in-vehicle infotainment system. Secondly, to further improve the integrated system with respect to driver distraction, thus making the system safer to use while driving. Thirdly, to verify that the resulting system decreases visual distraction and cognitive load during interaction. This demo paper describes the integration of the two existing systems, and the test environment designed to enable evaluation of the system.

1 Background

1.1 Driver distraction and safety

Driver distraction is one common cause of accidents, and is often caused by the driver interacting with technologies such as mobile phones, media players or navigation systems. The so-called 100-car study (Neale et al., 2005) revealed that secondary task distraction is the largest cause of driver inattention, and that the handling of wireless devices is the most common secondary task. The goal of SIMSI is to design systems which enable safe interaction with technologies in vehicles, by reducing the cognitive load imposed by the interaction and minimizing head-down time.

1.2 The Talkamatic Dialogue Manager

Based on Larsson (2002) and later work, Talkamatic AB has developed the Talkamatic Dialogue Manager (TDM) with the goal of being the most competent and usable dialogue manager on the market, both from the perspective of the user and from the perspective of the HMI developer. TDM provides a general interaction model founded in human interaction patterns, resulting in a high degree of naturalness and flexibility which increases usability. Also, TDM reduces complexity for developers and users, helping them to reach their goals faster and at a lower cost.

A major problem with the current state-of-the-art in-vehicle spoken dialogue systems is that they are either too simplistic to be useful to the end user, or alternatively that they are fairly sophisticated but unmanageable for the manufacturer due to the size and complexity of the implementation. TDM offers sophisticated multi-modal interaction management solutions which allow for easy modification and development, allowing interaction designers to easily explore new solutions and reducing overhead for new dialogue applications in terms of code and development man-hours.

TDM deals with several interaction patterns which are basic to human-human linguistic interaction, and offers truly integrated multimodality which allows user to freely switch between (or combine) modalities. All these solutions are domain-independent which means that they need not be implemented in each application. Using Talkamatic technology, dialogue behaviour can be altered without touching application properties, and application properties can be updated without touching the dialogue logic. This makes testing of different dialogue strategies, prompts etc. considerably quicker and easier than when using regular state-machine-based dialogue systems.

In addition, as the dialogue strategy is separated from the application logic, development time for new dialogue applications can be significantly reduced. Furthermore, the developer designing the application does not need to be a dialogue expert as the dialogue design is built into the dialogue manager.

Proceedings of the SIGDIAL 2013 Conference, pages 354–356,
Metz, France, 22-24 August 2013.

1.3 Integrated multimodality in TDM

There are reasons to believe that multi-modal interaction is more efficient and less distracting than uni-modal interaction (Oviatt et al., 2004). TDM supports multi-modal interaction where voice output and input (VUI) is combined with a traditional menu-based GUI with graphical output and haptic input. In cases where a GUI already exists, TDM can replace the GUI-internal interaction engine, thus adding speech while keeping the original GUI design. All system output is realized both verbally and graphically, and the user can switch freely between uni-modal (voice or screen/keys) and multi-modal interaction.

To facilitate the browsing of lists (a well known interaction problem for dialogue systems), Talkamatic has developed its Voice-Cursor technology[1] (Larsson et al., 2011). It allows a user to browse a list in a multi-modal dialogue system without looking at a screen and without being exposed to large chunks of readout information.

A crucial property of TDM's integrated multimodality is the fact that it enables the driver of a vehicle to carry out all interactions without ever looking at the screen, either by speaking to the system, by providing haptic input, or by combining the two. We are not aware of any current multimodal in-vehicle dialogue system offering this capability. Additional information is available at `www.talkamatic.se`.

1.4 Mecel Populus

While TDM offers full menu-based multimodal interaction, the GUI itself is fairly basic and does not match the state of the art when it comes to graphical design. By contrast, Mecel Populus is an commercial-grade HMI (Human Machine Interface) with professionally designed visual output. The Mecel Populus suite is a complete tool chain for designing, developing and deploying user interfaces for distributed embedded systems. It minimizes the time and cost of producing eye-catching, full-featured HMIs.

The Mecel Populus concept has several unique features compared to traditional HMI development. These features, when combined, remove the barriers that traditionally exist between the people working with requirements, system engineering, HMI design and implementation. An HMI is created and verified in Mecel Populus Editor without having to write any software. The HMI is then downloaded to the target environment where Mecel Populus Engine executes it. Mecel Populus has been designed for the automotive industry to deliver high performance user interfaces with a short time-to-market and to enable efficient software life cycle management. Additional information is available at `www.mecel.se/products`.

[1] Patent Pending

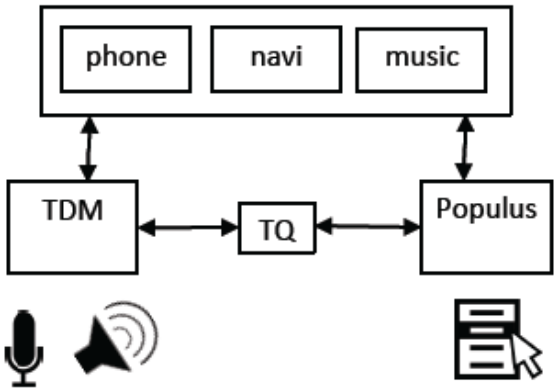

Figure 1: SIMSI system overview

2 System integration

The goal of this part of SIMSI is to provide a project-specific integration of TDM and the Mecel Populus platform. In this way, we establish a commercial-grade HMI for experiments and demonstrations. At the same time, the integration of TDM and Populus increases the commercial potential of both platforms, since it integrates a state-of-the-art HMI tool without voice capabilities and a dialogue manager with limited graphical capabilities.

The major problem in integrating Populus and TDM is that both systems keep track of the current state of the interaction and manage transitions between states resulting from user or system actions. Hence, there is a need to keep the systems in sync at all times. This is managed by a Transition Queue (TQ) module which keeps a lock which can be grabbed by either system at any time, unless it has already been grabbed by the other system. The systems then enter into a master-slave relation where the master is the system which owns the lock. The master tells the slave how the interaction state is to be updated, and the slave only waits for messages from the master until the lock has been returned to the TQ.

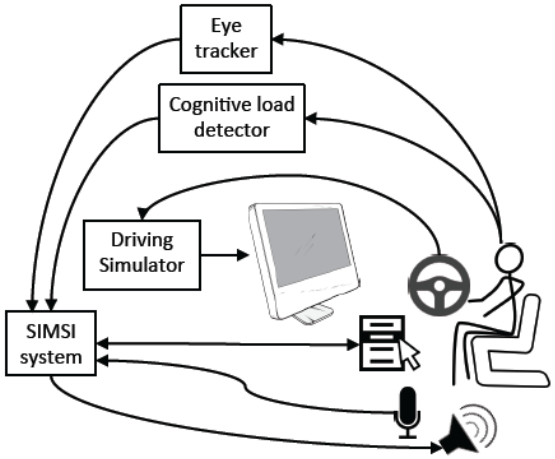

Figure 2: SIMSI test environment overview

3 Test environment

The purpose of this part of the project is to conduct ecologically valid test of the applications, and to begin and continue an iterative development cycle of testing - evaluation - development. We want to find the best interaction solutions in cases where it is not intuitively clear what is best. This involves implementing variants of a behaviour, testing them on naive users, collecting data from these interactions, and establishing statistically significant results based on the collected data.

The test environment consists of two parts, apart from the dialogue system: a driving simulator (SCANeR from Octal) and an eye tracker (Smart Eye Pro from Smarteye). In later tests we will also include instruments for measuring cognitive load.

In our setup we have three monitors, giving the user a wide field of view. We also have a gaming steering wheel, including pedals, gear lever and a driver's seat. These are used mainly to control the driving simulator, but there are also a number of buttons on the steering wheel which are used to browse the menus in the HMI and as Push-to-talk (PTT). An Android tablet (Asus Eee Pad Transformer TF101) showing the HMI GUI is placed in front of the user, trying to match the position of a display in a car. Both TDM and Populus run on the same desktop computer as the driving simulator, and a Populus Android app runs on the tablet. The app allows the user to select items by tapping them, as well as scrolling in lists in normal smart phone fashion. The eye tracker runs on a separate desktop computer, as it requires a substantial amount of processing power.

Figure 3: SIMSI test environment in action

Studio software that comes with the driving simulator is used to design and run scenarios. The scenarios govern how autonomous traffic should behave and events, such as weather change and the state of traffic signals. The simulator logs data for the environment and each vehicle. Data like lane deviation (where in the lane the vehicle is) and how the user handles instruments, e.g. steering wheel and pedals, can be used to measure cognitive load. At a later stage this kind of data can also be used to trigger behaviour in the dialogue system.

The eye tracker uses three cameras to track the user's eyes and head at 60 Hz. The cameras are spaced to give good tracking in the middle of the scene, where you typically look when you're driving, and at the same time capture head movement to the side. As we are interested in when the user is looking at the tablet, we placed one of the cameras specifically to improve eye tracking in this area.

References

Staffan Larsson, Alexander Berman, and Jessica Villing. 2011. Adding a speech cursor to a multimodal dialogue system. In *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy, 2011*, pages 3319–3320.

Staffan Larsson. 2002. *Issue-based Dialogue Management*. Ph.D. thesis, Göteborg University.

Vicki L. Neale, Thomas A. Dingus, Sheila G. Klauer, Jeremy Sudweeks, and Michael Goodman. 2005. An overview of the 100-car naturalistic study and findings.

Sharon L. Oviatt, Rachel Coulston, and Rebecca Lunsford. 2004. When do we interact multimodally?: cognitive load and multimodal communication patterns. In *ICMI*, pages 129–136.