DJ GoDiS: Multimodal Menu-based Dialogue in an Asynchronous ISU System

Staffan Larsson, David Hjelm
GU Dialogue Systems Lab
Göteborg University, SE 405 30 Göteborg, Sweden
{sl, dhjelm}@ling.gu.se

Abstract

abstract

1 Introduction

In this demo we present work on multimodal menu-based dialogue currently being carried out at the Göteborg University Dialogue Systems Lab as part of the TALK project. We are adding support for multimodality to GoDiS and TrindiKit, as well as making use of recent simplifications and improvements on TrindiKit. For modality fusion and multimodal generation we are using multimodal grammars written in GF and compiled into an OAA agent.

The basic idea is to use a menu-based spoken dialogue (Larsson et al., 2001) together simultaneously with a graphical menu system. Input is accepted in either modality. Output is presented in either or both modalities (spoken interaction and menu-based GUI) depending on whether each modality currently provides an information channel between system and user. We believe this is a simple yet very useful way of exploring the benefits of multimodal dialogue. In addition, it has the advantage of subsuming and extending the already familiar menu-based-GUI-style interface.

2 GoDiS, TrindiKit and GF

GoDiS (Larsson, 2002) is an experimental dialogue system implementing a theory of Issue-Based Dialogue Management based on Ginzburg’s concept of Questions Under Discussion (QUD). GoDiS is implemented using the TrindiKit, a toolkit for implementing dialogue move engines and dialogue systems based on the Information State approach (Traum and Larsson, 2003). GoDiS has been adapted to several different dialogue types, domains, and languages, including menu-based action-oriented dialogue when acting as an interface to a mobile phone (Larsson et al., 2001) or VCR. To enable multimodal interpretation and generation, we have recently integrated GF (Ranta, 2004) into OAA, TrindiKit and GoDiS.

3 Asynchronicity in TrindiKit

In TALK, all partners have agreed to use OAA as a common interface. Previous versions of TrindiKit were compatible with OAA but required dialogue systems to be run by TrindiKit agent which would then call other agents when necessary. No solvables were offered by TrindiKit to the OAA agent community.

TrindiKit4 offers the possibility to distribute system components across several OAA agents. This means that a number of
input and output modules can run simultaneously, and update and inspect the TIS independently. For instance, a module can listen continuously for speech and update the TIS only when speech has been detected without blocking the rest of the system. The capabilities of each TrindiKit component are published as solvables to the OAA community.

System coordination is done by a special purpose control module, which can be set up to monitor certain TIS variables and execute an associated control algorithm when they are set to a certain value. This can be used e.g. to notify the interpretation module that the input module has updated the TIS. The control algorithm can contain calls to TrindiKit modules or OAA agents as well as TIS updates and checks. Any number of control algorithms can be run in parallel. As OAA enables asynchronous processing, a previous implementation of asynchronicity in TrindiKit has been removed, thus radically simplifying the TrindiKit core.

In DJ GoDis, this architecture is exploited by having multiple input and output modules to allow for spoken and GUI-based (as well as written) communication in parallel.

4 Menu-based multimodal dialogue

This approach offers what we believe to be a very flexible and intuitive multimodal interface to any device that can be operated using a standard menu-based GUI interface. Several modes of interaction emerge from a simple setup, including the following:

- The user may use the menu-based GUI in the normal way, without bothering with speech
- The user may use the spoken interface without bothering with the GUI
- The user may make menu choices using speech; these will have the same effect on-screen as making the menu choice by pointing and clicking
- The user may exploit GoDis' flexible dialogue management to bypass the menu system and give commands and/or provide information as (s)he sees fit. Again, any spoken interactions will result in the corresponding menu options appearing on-screen, thus enabling the user to freely switch modality or combine modalities as desired at any point in the interaction.
- No user-modeling is done apart from keeping track of the shared dialogue state; user adaptation emerges instantly as an effect of the user's choice of modality from utterance to utterance

5 A sample interaction with an mp3 player application

This short example shows how GoDis tries to figure out the user's goal by posing a clarification question. (This process is described in Larsson (2002) as "dependent accommodation").

usr> (GUI: CLICKS ON A SONG IN THE PLAYLIST)

sys> Do you want to play or remove from playlist? (GUI: A MENU APPEARS WITH TWO BUTTONS: [PLAY] AND [REMOVE])

usr> Play

sys> OK, play. (GUI: [PLAY] BUTTON IS HIGHLIGHTED)

Acknowledgements

The research reported here was funded by TALK (Talk And Look, Tools for Ambient Linguistic Knowledge), EC Project IST-507802.
References


