A Multimodal Interaction System for Navigation

Dennis Hofs, Rieks op den Akker, Anton Nijholt & Hendri Hondorp
Centre of Telematics and Information Technology
University of Twente, Enschede, the Netherlands
{infrieks,anijholt}@cs.utwente.nl

1 Introduction

To help users find their way in a virtual theatre we developed a navigation agent. In natural language dialogue the agent assists users looking for the location of an object or room, and it shows routes between locations. The speech-based dialogue system allows users to ask questions such as “Where is the coffee bar?” and “How do I get to the great hall?” The agent has a map and can mark locations and routes; users can click on locations and ask questions about them.

In an earlier version (Luin et al., 2001) no underlying dialogue model was used. Now we have a generic architecture and dialogue model allowing multimodal interactions, including reference modelling and backward/ forward looking tags to determine and model dialogue structure.

The architecture of the system can be conceived as a box containing a dialogue manager (DM) and world knowledge, to be connected with input and output processors. The processors can be plugged into different dialogue systems. E.g., our speech processor and generator have been used with some adjustments for both a navigation agent and a tutor agent application. The implementation supports streaming at the recording side as well as the playback side to decrease the delay between user utterances and system responses. The architecture enables the user to interrupt when the system is speaking.

2 The Multimodal Dialogue Manager

Speech input is sequentially processed by the dialogue act determiner (that selects a dialogue act), the parser, the reference resolver and the action stack. The dialogue management module itself is not implemented as a system of asynchronous distributed agents: there is a strict logical order in the execution of the updates of dialogue information, the selection of goals and the execution of actions after the system has received user input. The mouse input reaches the DM through a map, which is a visual 2D representation of the world (the virtual theatre). The user can point at objects or locations on the map and the world notifies the DM of these events.

In our dialogue and action management module we allow mixed-initiative dialogues and several types of subdialogues. An Action stack stores the system’s actions that are planned and a subdialogue stack keeps track of the current dialogue structure. All dialogue acts are also kept in a history list to be retrieved for later use.

The input queue of the DM receives the user’s utterances in the form of lists of possible acts. The relation between word sequences and acts is specified in the grammar for the speech recogniser. A dialogue act contains the original sentence and a forward and backward tag, based on the DAMSL scheme. In addition, a dialogue act may have a domain-specific argument.

When the DM gets a list of acts from the input queue, it passes it to the dialogue act determiner together with the history. Information in the history that the dialogue act determiner uses, are the forward tags of the last utterance in the current subdialogue and the last utterance in the underlying subdialogue, if present. For every possible forward tag, the dialogue act determiner holds an ordered list of preferred backward tags that can follow it. The dialogue act determiner selects the preferred dialogue act and returns it to the DM.

Our dialogue act determiner also helps to determine the dialogue structure with respect to subdialogues. If the user could end the current subdialogue – that is if the last dialogue act in the underlying dialogue was performed by the system and the user started the current subdialogue – the dialogue act determiner will always try to end the current subdialogue by connecting the user's dialogue act to the underlying dialogue.

3 Parser and Reference Resolver

The DM can start processing the selected dialogue act. It starts with parsing the phrases that occur in parameters in the dialogue act's argument. The feature structure that is obtained is stored together with the original parameters in the dialogue act. The next step is to bind the parameter to a real object in the navigation agent's world. That is where the reference resolver comes in. The reference resolver is used for all references to objects in the world. References can be made by the user or the system, by talking about objects or by pointing at them. The reference resolution algorithm is a modified version of Lappin and Leass's algorithm that assigns weights to references, based on a set of salience factors. Although language used in the dialogue system is rather simple, compared to complex structures in written texts, resolving referring expressions in multimodal interaction is far from trivial. The modification makes the algorithm suitable for multimodal dialogues. For details of the adapted algorithm see (Hofs et al, 2003).

After resolving references, the set of objects found is added to the parameter in the dialogue act where the reference occurred. So now we have a dialogue act that consists of a forward tag, a backward tag and an argument. The parameters in the argument have a value that was taken directly from the recognition result as well as a feature structure received from the parser and a set of objects received from the reference resolver.

4 Dialogue and Action Stacks

The history contains all dialogue acts that occurred during the dialogue. Besides the user's dialogue acts, it also contains the system's dialogue acts. A subdialogue stack is used for the currently running subdialogues. The action stack contains the actions that the system still needs to execute, but the stack also creates those actions, when it is provided with the user's dialogue acts after the parameters have been parsed and references were resolved. Actions are specified in templates and are part of the action stack.

When the action stack receives a user dialogue act, it will try to find an action template with matching forward tag and argument. It creates actions based on the action names in the template and it extracts the action arguments from the user's dialogue act. The new actions are put on top of the stack and when it is the system's turn, it will take an action and execute it. If a dialogue act is performed within the action, the reference resolver's dialogue model should be updated and the dialogue act should be added to the history. Like a user dialogue act, a system act consists of a forward tag and backward tag. It does not have an argument. Instead it contains the action within which the act was performed.

5 Conclusions

We presented a generic dialogue model for spoken multimodal interaction. One of our applications, discussed here, is a navigation agent that helps users to find their way in a virtual environment. The system uses Dutch speech recognition and synthesis models. References to objects in the environment can be made by user or system. The resolution algorithm is a multimodal version of the well-known Lappin and Leass's algorithm. Backward and forward-looking tags according to the DAMSL scheme are used to model the dialogue structure. Until now we used our system to implement a navigation agent in a virtual environment and, using the speech architecture module, a tutor agent. In progress is an application where we integrate speech and haptics in a virtual nurse education environment.

References

D. Hofs, R. od Akker & A. Nijholt. A Generic Architecture and Dialogue Model for Multimodal Interaction. *Nordic MUMIN* Workshop, Denmark, 2003.

S. Lappin & H. Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535-561, 1994.

J. van Luin, A. Nijholt & R. op den Akker. Natural Language Navigation Support in Virtual Reality. *ICAV3D* Workshop Greece, 2001, 263-266.