Transferable Multi-modal Dialogue Systems for Interactive Entertainment

Oliver Lemon
CSLI, Center for the Study of Language and Information
Stanford University, California, USA
lemon@csli.stanford.edu
www-csli.stanford.edu/~lemon

October 5, 2001

Abstract

We present and demonstrate a dialogue system for multi-modal conversations with autonomous mobile robots, and argue that it could feasibly be used in games where a user controls a variety of NPC[1] 'bots'. Moreover, the 'plug-and-play' nature of the system makes it 'transferable' in the sense of being portable to conversations with robots which have different abilities and vocabularies. This modularization of the dialogue manager is achieved via a level of abstraction over robot 'activities' which allows game developers to use the system with very minimal linguistic knowledge. We illustrate these ideas by demonstrating a dialogue interface to the (simulated) WITAS robot helicopter, or UAV ('Unmanned Aerial Vehicle').[2] We argue that this type of system points the way forward in the development of conversational systems in games.

1 Introduction

Enthusiasm for developing conversational characters in games is not difficult to generate [1, 2], but most of these visions seem to rely on the dream of solving all of the problems of Computational Linguistics. Since such a breakthrough is unlikely to happen anytime soon, we present a more modest proposal, which still allows for complex spoken conversational interactions with a variety of NPCs in games.

One of the main problems in developing spoken dialogue systems for interactive games is that individual dialogue systems have been application-specific, and difficult to transfer to new domains, and thus to new games or to various different characters within a game. Moreover, most of the dialogue systems developed in the past have been for simple "form-filling" interactions which are relatively uninteresting as far as gaming is concerned. We have made some progress in developing a "plug-and-play" multi-modal (i.e. speech and graphical interaction) dialogue system for controlling 'devices' such as mobile robots. We argue that many of the same criteria apply to developing dialogue systems in games as apply to their development for (simulated) mobile robots. The main idea of our system is a level of abstraction over 'activities' of individual devices (in this case robots), where each device has an *activity model* describing the tasks that dialogue can be used to initiate and monitor for that device (e.g. for a robotic helicopter, searching for, and then tracking, a moving vehicle). This level of abstraction enables different dialogue

[1] "Non-player character"

[2] Wallenberg laboratory for research on Information Technology and Autonomous Systems (WITAS) Unmanned Aerial Vehicle, under construction at Linköping University, Sweden. This research was funded under the WITAS Project by the Wallenberg Foundation, Sweden.

systems to be compiled for different 'devices' or robots which the user might interact with in the course of a game. The power of this idea is that one dialogue system can be used to interact with various devices or characters, with minimal linguistic expertise from the developer (in fact, they need only know which lexical items trigger certain activities – for example that "find", "look for", and "search" all trigger a robot *search* activity).

2 Dialogues with robots and NPCs

In the past, many dialogue systems have been built for use in contexts where conversational interactions are predictable and can be scripted, and where the operating environment is static. For example, a dialogue for buying an airline flight can be specified by way of filling in certain parameters (cost, destination, and so on) and a database query, report, and confirmation cycle. This 'travel-planning' paradigm has been the focus of dialogue system research for several years, and seems relatively uninteresting in the context of gaming. In these form-filling cases a dialogue can be modelled by a finite-state transition network for all paths through dialogue states to recognisable completion states.

But consider a conversation with an autonomous mobile robot with perceptions in a environment which is constantly changing. Dialogues with such a device will be very different (see e.g. arguments in [3]) to those in the travel-planning paradigm. There will be no predictable course of events in the dialogues. The robot itself may 'need' to communicate urgently with its operator. There may not be well-defined endpoints to conversations, and relevant objects may appear and disappear from the operating environment. Tasks given to the robot will also need to be specified, ordered, and their execution monitored verbally. The same comments apply when we consider conversations with NPCs in games.

We have argued [4] that the dialogue modelling and management techniques developed under the travel-planning paradigm are not rich enough to support these more complex interaction scenarios, and we have found that different structures and methods need to be adopted. See [5, 4, 6] for technical descriptions of our work in dialogue modelling to support such conversations, and below for a sample of what we have achieved.

2.1 A collaborative UAV game using dialogue

Helicopter operation is a popular and relatively common game type, from simulation-style games (e.g. Novalogic's *Comanche Gold*) to futuristic settings (e.g Psygnosis's bladerunneresque *G-Police*). The WITAS UAV project endeavours to construct a real-life autonomous helicopter with onboard planning and deliberative systems, and vision capabilities (for details see e.g. [7]). This robot helicopter will ultimately be controlled by the dialogue system developed at CSLI, though at the moment we interact with a simulated[3] UAV – just the sort of agent, in fact, that could be used in a game.

The first game-type dialogue interaction we have developed concerns the notion of a "joint-activity" between the UAV and a human operator. In this scenario, the UAV's vision system is not good enough to determine whether a particular vehicle is the one sought-after, or else only the human player has the authority to determine this, so that human and robot must *collaborate* (using the notion of a "joint-activity") to find and track a vehicle. The following dialogue (see figure 1) shows how a typical interaction works. (The system runs in real time on a laptop computer under Windows 2000).

Note here that generated messages from the system involve anaphoric expressions wherever possible (more on generation below) and that the user is able to make explicit queries about the robot's activites (both current and future).

[3]Our UAV simulator is logic-based, using KIF statements under JTP (the Java Theorem Prover)

Figure 1: A demonstration of the working dialogue system

Multi-modal Utterances	**Dialogue Moves**
Operator (O): *Our job is to look for a red car*	Raise joint goal
UAV (U): *Ok. I am looking for one.*	Confirm activity
O: *Fly here [+click on map] please*	Deictic command
U: *Okay. I will fly to waypoint one*	Confirm activity, introduce referent
U: *Now taking off and flying there.*	Report activity
O: *Stop that. Go to the tower instead.*	Command, Revision
U: *I have cancelled flying to waypoint one. Which tower do you mean?*	Report activity status. Initiate disambiguation subdialogue
O: *Show me the buildings*	GUI Command
U: *Here you are* [displays buildings on map].	Backchannel
O: *What are you doing?*	Activity query
U: *I am searching for a red car*	Answer
U: *Which tower shall I fly to ?*	Raise pending question
O: *The black one*	Answer
U: *Okay. Now flying to the black tower and looking for a red car*	Confirm activities
O: *What will you do next?*	Activities query
U: *I have nothing planned.*	Answer
U: *I see a red car on main street* [display on map, show video images], *Is this the right car?*	Report relevant events, Joint-Activity query
O: *Maybe. Zoom in*	Answer, Command
U: *Okay* [displays zoomed image]	Confirmation
O: *Yes, that's the right car*	Confirm complete activity
U: *Okay. I am following it .*	Report initiated activity
U: *Warning, my fuel is low.*	Warning status report

3 Dialogue System Architecture

Our dialogue system uses a standard hub architecture[4] (see Figure 2) with various "off the shelf" components:

- Speech Recogniser: a wrapper to a Nuance 7.0.4 speech recognition server[5] using a language model compiled directly from our Gemini grammar for robot interaction dialogues.
- NL ("natural language"): a wrapper to the SRI Gemini parser and generator [8] using our robot interaction grammar.
- TTS ("text-to-speech"): a wrapper to the Festival 1.4.3 speech synthesiser[6]

4 A 'Plug-and-Play' Dialogue System

As discussed above, the central idea in our dialogue system is an abstraction over "activities" of the robot or NPC with which a conversation takes place. In brief, an *activity model* for a robot or NPC

[4]OAA2: the Open Agent Architecture
[5]Nuance: *www.nuance.com*
[6]Edinburgh University, Centre for Speech Technology Research

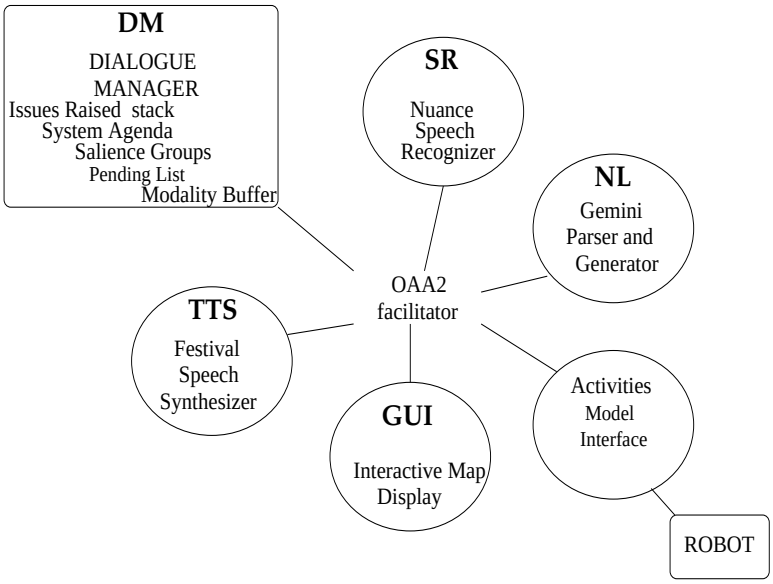

Figure 2: Dialogue system architecture

is a declarative script which describes the hierarchical decomposition of tasks involved in a high-level (linguistically specified) activity. For example the activity model for *go* involves the ideas that;

- the *go* task uses the resource of the UAV itself
- if the UAV is on the ground (precondition checking), it must spawn a *take-off* sub-activity
- a noun-phrase NP is needed as argument for *go*
- if completed, the *go* activity's postcondition is that the UAV is *at* the referent of NP (UAV state update)

Thus the activity script specifies how high-level activities may spawn sequences of subactivities and under what conditions this happens. An ongoing dialogue with a device uses its activity script to build an "activity tree" (AT) which is very similar to a hierarchical task network (HTN). Note that multiple activites may be simultaneously operative, as long as they do not use the same resources.

Dialogue management itself is carried out using a set of abstract dialogue move classes which are domain independent (e.g. command, activity-query, wh-question, barge-in, revision, ...). Any ongoing dialogue constructs a Dialogue Move Tree (DMT) representing the state of the conversation, whose nodes are instances of the dialogue move classes, and are linked to nodes on the Activity Tree where appropriate.

An obvious issue is that of how general our grammar and vocabulary (used for speech recognition, parsing, and generation) for robot interaction is. Currently we have no clear answer to this, although it would be relatively easy to add new verbs triggering activities of different robots. In the future we aim to explore "plug-and-play grammars" so that each robot or NPC may upload its own specific vocabulary and grammar to the dialogue system when needed (see recent developments in [9]).

4.1 Activities, "Aboutness", and NL Generation

One of the main criticisms of Sega's *Seaman*, which was a commercially released game involving a spoken dialogue system, was that its responses were often not relevant to or "about" what the player is talking

about. In our constrained dialogue settings, the notion of "current-activities" defines a filter on generated messages such that the robot converses about its activities and their progress. This was one also one of the main points raised in [2] – that dialogues must be relevant to tasks that the player cares about. In our system, the notion of activity ensures that dialogue is always in the service of joint activities between the player and an robot or NPC.

As well as this "relevance" property, we have developed a generation algorithm which is "echoic" (mirrors speech forms used by the player), "variable" (involves a random element in order to make produced speech more diverse), and uses anaphoric and deictic expressions where appropriate. A full paper will discuss this algorithm in detail.

5 Conclusion

We present a multi-modal dialogue system for interaction with robots which we argue is generalizable and transferable to dialogues with 'bots' or NPCs in interactive games. We show how a level of abstraction over 'activities' in dialogue management can be used to make modest but practical dialogue systems, feasible for development in interactive games.

References

[1] Steven Poole, *Trigger Happy: the inner life of videogames*, Fourth Estate, 2000.

[2] Bill DeSmedt and Don Loritz, "Can we talk?," in *Artificial Intelligence in Computer Games*. 1999, AAAI Press.

[3] Renee Elio and Afsaneh Haddadi, "On abstract task models and conversation policies," in *Workshop on Specifying and Implementing Conversation Policies, Autonomous Agents'99*, Seattle, 1999.

[4] Oliver Lemon, Anne Bracy, Alexander Gruenstein, and Stanley Peters, "Information states in a multi-modal dialogue system for human-robot conversation," in *5th Workshop on Formal Semantics and Pragmatics of Dialogue (Bi-Dialog 2001)*, Peter Kühnlein, Hans Reiser, and Henk Zeevat, Eds., 2001, pp. 57 – 67.

[5] Oliver Lemon, Anne Bracy, Alexander Gruenstein, and Stanley Peters, "The WITAS Multi-Modal Dialogue System I," in *Proceedings of 7th European Conference on Speech Communication and Technology (Eurospeech' 01)*, Aalborg, 2001.

[6] Oliver Lemon, Anne Bracy, Alexander Gruenstein, and Stanley Peters, "A multi-modal dialogue system for human-robot conversation," in *Proceedings of North American Association for Computational Linguistics (NAACL 2001)*, 2001.

[7] Patrick Doherty, Gösta Granlund, Krzystof Kuchcinski, Erik Sandewall, Klas Nordberg, Erik Skarman, and Johan Wiklund, "The WITAS unmanned aerial vehicle project," in *European Conference on Artificial Intelligence (ECAI 2000)*, 2000.

[8] John Dowding, Jean Mark Gawron, Doug Appelt, John Bear, Lynn Cherny, Robert Moore, and Douglas Moran, "GEMINI: a natural language system for spoken-language understanding," in *Proc. 31st Annual Meeting of the ACL*, 1993.

[9] Ian Lewin, Manny Rayner, Genevieve Gorrell, and Johan Boye, "Plug and play speech understanding," in *Proceeedings of SIGdial 2001*, 2001.