DELIVERABLE D2.2

D2.2: SOCIALLY-ENABLED LINGUISTIC INTERACTION

Amy Isard (UEDIN)

Beneficiaries: UEDIN (lead), HWU, FORTISS, UNIBI
Workpackage: WP2: Natural Language Communication
Description: The general objective of WP2 is to develop components for recognising, understanding, and generating embodied natural language. In this report we describe the natural language processing components included in the final integrated system, which can understand and generate a wide range of utterances and multimodal social behaviours.

Version: Final
Nature: Report (R)
Dissemination level: Public (PU)
Pages: 7
Date: 2014-03-31
1 Introduction

In JAMES, natural language plays an important role in both the input processing and in the selection of behaviours for the robot agent, since natural-language interaction is central to social communication. The general objective of WP2 is to develop components for recognising, understanding, and generating embodied natural language. This report describes the components which we have developed for the final version of the integrated JAMES system (see D7.3).

2 Language Input

2.1 Speech Recognition

We use the Microsoft Kinect and the associated Microsoft Speech API for the speech recognition component of JAMES in both English and German, along with have speech recognition grammars in SRGS format [Hunt and McGlashan, 2004] Details of this format can be found in deliverable D2.1. The output from the speech recognition is an ordered list of hypotheses, each with a confidence value and an angle of input, which are passed to the natural language interpretation module.

3 Constraining Speech Grammars

In our user studies, we have found that the speech recognition still produces errors, particularly when the test subjects are further from the microphones or there is background noise. To attempt to alleviate this problem, we plan to use information from the Planner or Social Skills Execution Module concerning the expected next user response in order to weight the speech grammar and prioritise more likely utterances.

In collaboration with WP4, we have drawn up a list of expected user utterances types as follows:

- **yesno** response to a system question: "yes", "no"
- **drinks** a drink order: "a lemonade please"
- **thanks** response to a drink handover: "thanks"
- **greet** initial greeting: "hello"
- **bye** final leave taking: "bye"
- **lemonade** a particular drink type, in response to system request for clarification: "lemonade"

SRGS grammars allow weights to be attached to grammar items, and we plan to map each utterance type to a subset of the speech recognition grammar, and to dynamically increase the weights to the appropriate utterances, which will hopefully improve recognition accuracy. We plan to test this strategy during the project extension.

3.1 Parsing

Natural language interpretation and generation are carried out using bi-directional OpenCCG grammars [White, 2006], and details of the formats of these grammars can be found in deliverable D2.1. In collaboration with WP3, we designed an ontology of Communicative Acts which are used to pass the results of the parsing to the Social State Processing Module. Along with the act type, we also communicate the confidence value of the original speech string, the string which was parsed, and the original string hypothesis if the parse was for a fragment. If the communication act concerns a drink order, we also pass the list of drinks. If no parses could be found
from the recognition string, we also pass this information on. The set of communicative acts is listed below, with examples.

**greet** "hello"

**yes-no** "yes", "no"

**goodbye** "bye"

**thanks** "thanks"

**no-order** "I don’t want anything", "nothing"

**drink-order** "a coke please", "please could you give me two lemonades"

**confirm-order** "yes a coke"

**correct-order** "no a lemonade"

**drink-list-request** "what drinks do you have"

**bad-asr** no parses could be found.

We attempt to parse each speech hypothesis in turn. If a full parse is found, we add it to a list of Communication Act hypotheses. If no parse is found, we attempt to parse fragments of the hypothesis, based around the nouns from the grammar which refer to drink orders, and the words for "yes" and "no" in the appropriate language. We add the result of the longest parse to the list of Communication Act hypotheses. Once all the speech hypotheses have been processed, we send the list of the communicative acts to the Social State Processing Module.

### 4 Output

#### 4.1 Linguistic Output

JAMESs linguistic behaviour is based on carrying out high-level communicative acts selected by the high-level planner (WP4). The output focuses on domain actions such as requesting and clarifying the users needs, as well as social actions such as managing attention.

In JAMES we are using OpenCCG to generate the robot language output, as described in deliverable D2.1. The basic RST-style format [Isard and Matheson, 2012] of the API for communication between the planner and the output module was described in D2.1, but a number of additions have been made to accommodate the final JAMES scenario.

#### 4.2 RST for language output

Dialogue acts are divided into four types, each of which has one or more subtypes, and in some cases the form of the output may be specified. If no specification is made, the system will choose at random between the various ways to express the content.

- **query**
  - **offer** help: "can I help you?", order: "did you want to order anything?"
  - **offer-more** else: "anything else?", all: "will that be all?"
  - **choose** "would you like blue lemonade or red lemonade?"
  - **clarify** "did you say blue lemonade?"
how-many “how many lemonades?”

• inform

  list “we have cola, blue lemonade and red lemonade”

  not-understand please-repeat: “sorry, could you repeat that please’?” what-say: “what did you say?”

  prepare-hand-over “I will get your drinks now”

  hand-over “here is your lemonade”

  confirm “okay, two lemonades and a coke”

• reply

  positive-reply “yes”

  negative-reply “no”

• social

  greet “hello”

  bye “goodbye”

  thanks “thanks”

  acknowledge-thanks “you’re welcome”

  acknowledge “okay”

  wait “I’ll be with you in a minute”

  acknowledge-wait “thanks for waiting”

4.3 Multimodal Presentation Planning

In addition to speech, the robot system expresses itself through facial expressions, gaze behaviour, and actions of the robot manipulators. The presentation planner coordinates the output across the various multi-modal channels to ensure that it is coordinated both temporally and spatially, and passes on the appropriate gesture specifications to the robot modules. Details can be found in deliverable D2.1.

References


A Included Papers

The following paper is included in this deliverable.


Abstract: Many traditional dialogue systems use simple predicates to send information between a Dialogue Manager and a Natural Language Generation system. We propose a flexible RST-style interface to allow for more complex structures and multimodal output, and we place the first stage of content planning under the control of the dialogue management system with access to a system-wide information state.
Rhetorical Structure for Natural Language Generation in Dialogue

Amy Isard and Colin Matheson
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh, United Kingdom
{amy.isard, colin.matheson}@ed.ac.uk

Abstract

Many traditional dialogue systems use simple predicates to send information between a Dialogue Manager and a Natural Language Generation system. We propose a flexible RST-style interface to allow for more complex structures and multimodal output, and we place the first stage of content planning under the control of the dialogue management system with access to a system-wide information state.

1 Introduction

In many dialogue systems, a Dialogue Manager (DM) sits at the centre of the interaction, taking user inputs and sending an output specification to a Natural Language Generation (NLG) system. In multimodal systems, there may be a Dialogue and Interaction Manager, and the outputs may go via an intermediate stage where the different modalities are synchronised, and in other cases a planner may take over the duties of dialogue management, but there is usually a need to store the history and to specify the language output. Where the system has a physical component, facial and hand gestures may also be specified.

We propose a flexible interface between the DM and the output modalities which can be used in a variety of human-machine dialogue domains. It is based on RST structures (Mann and Thompson, 1988), and is related to work by (Stent et al., 2004) who used a similar approach in MATCH, their text-based restaurant recommendation system. However in MATCH, the DM sends high-level goals to the text planner, whereas in our systems, the DM performs part of the content selection task (in some cases by communicating with a domain planner and/or knowledge representation module), and sends a structured representation to the NLG system. The potential content to express can come from a wide range of sources, including the dialogue history, the domain knowledge or the task plan. For example, the task plan may describe sequences of actions which need to be carried out in constructing objects (Foster and Matheson, 2008).

This work can be seen in the broader context of attempting to integrate language processing, dialogue management, and NLG more closely.

2 Use Cases

To date, we have used our RST representation in three working systems in varied domains:

- the JAST system (Foster and Matheson, 2008; Foster et al., 2009), which allows a human to collaborate with a robot in building simple wooden toys
- the JAMES system (Petrick and Foster, 2012; Petrick et al., 2012), a robot bartender
- the Beetle II system (Dzikovska et al., 2011), a tutorial dialogue system for basic electricity and electronics

The top level rhetorical structures which we use are the following:

enablement where one situation or action is necessary (but not always sufficient) for another situation to action to occur, e.g. “to build a tower, insert the green bolt through the red cube and screw it into the blue cube” (JAST).

elaboration where one piece of content adds further information about an object which has already been mentioned e.g. “the battery in circuit 5 is in a closed path which does not contain a bulb” (Beetle II).
enablement
build
join
insert screw
tower green bolt
red cube
green bolt blue cube

Figure 1: Graph representation of enablement and join relations

<output>
<objects>
  <obj id="o1" type="bolt" color="green"/>
  <obj id="o2" type="cube" color="red"/>
  <obj id="o3" type="cube" color="blue"/>
  <obj id="o4" type="tower"/>
</objects>
<rst>
<relation type="enablement">
  <pred action="build" result="o4"/>
  <relation type="join">
    <pred action="insert">
      <obj idref="o1"/>
      <obj idref="o2"/>
    </pred>
    <pred action="screw">
      <obj idref="o1"/>
      <obj idref="o3"/>
    </pred>
  </relation>
</relation>
</rst>
<actions>
  <action type="handover">
    <obj idref="o1"/>
  </action>
</actions>
</output>

Figure 2: Multimodal RST XML for enablement relation

definition where one piece of information defines another e.g. “it means that the battery is damaged” (Beetle II).

join which signifies a simple aggregation of two pieces of content e.g. “hello, what would you like to drink” (JAMES).

A graph of an enablement relation from JAST is shown in figure 1, and a possible surface realisation for this is “to build a tower, insert the green bolt through the red cube and screw it into the blue cube”. The multimodal XML representation of the RST is shown in figure 2; as well as giving the content to be spoken, this specifies that the robot should hand over object o1 (the green bolt) to the user.

3 Conclusions

We designed a flexible interface for communications between the Dialogue Manager and the Natural Language Generation components in a dialogue system. We have used the interface in a number of different systems, and shown that it encourages the integration of multimodal output modalities. The systems we have described all use rule-based Dialogue Management or Planning, but the RST could also be used in a statistical dialogue system as long as the NLG component is grammar-based.

References


