A Spoken Language Interface to a Mobile Robot *

Johan Bos
Dipartimento di Informaticà
Universita di Roma "La Sapienza"
via Salaria 113 00196 Roma, Italy

Tetsushi Oka
Dept. of Information and Systems Engineering
Fukuoka Institute of Technology
3-30-1 Wajiro-higashi, Fukuoka 811-0295 Japan

Abstract

We describe a spoken dialogue interface to a mobile robot, which a human can direct to specific locations, ask for information about its status, and supply information about its environment. The robot uses an internal map for navigation, and communicates its current orientation and accessible locations to the dialogue system. In this paper, we focus on linguistic and inferential aspects of the human-robot communication process.

1 Introduction

The use of spoken language is arguably the most natural way of establishing a communication channel between a human and a robot. A successful implementation of a talking robot requires a good understanding of many different aspects of conversation, ranging from acoustic signals, the syntactic structure of the language, the meaning associated it, and the underlying goals derived from it. This paper presents an approach to human-robot dialogue understanding. In the framework that we introduce, the meaning of utterances of the dialogue participants and other information of the situation are represented as logical forms of compositional semantics, and logical inferences are drawn to manage the direction of the dialogue from the robot's point of view: when to perform which actions, how to answer a question, and whether to show agreement or disagreement towards the user's contribution.

*This work was conducted at ICCS, School of Informatics, University of Edinburgh.

2 Natural Language Understanding

2.1 Speech Recognition

The robot that we developed is able to communicate with humans in spoken language, rather than typed. In our system, we use Nuance's speaker-independent speech recognition system (`www.nuance.com`), which allows language models to be specified in the GSL (Grammar Specification Language), a form of context free grammars. Rather than coding the grammar-based language model in GSL directly, we use a generic domain-independent, but linguistically motivated grammar as a starting point. The grammar in question is a unification grammar for English, which gets translated into GSL, using the UNIANCE compiler [1]. We did not adopt the standard slot-filling paradigm for semantic interpretation, but used UNIANCE's feature to employ a sophisticated compositional semantics involving λ terms which are passed as the value of a single slot for the recognised sentence. As a result the output of the speech recognition component is a genuine semantic representation, and no further parsing is required before handing it over to the dialogue manager.

2.2 Semantic Interpretation

Discourse Representation Theory (DRT) [5] is used for meaning representation in the system. It is a formal theory of discourse interpretation, covering a wide variety of natural language phenomena including referring anaphoric and deictic expressions, quantified expressions, plural noun phrases, and a wide spectrum of presuppositional expressions [3].

We use Discourse Representation Structures (DRSs) to represent the meaning of the dialogue between user and system. There are computational implementations that provide means to extend existing linguistic grammars with DRS-construction tools that we use to design the semantic interpretation component. There is a standard translation from DRSs to

formulas of first-order logic that behaves linear on the size of the input, which gives us the means to implement logical inference.

DRT was initially designed to deal with texts, so we use an extension of standard DRT that enables us to cope with the semantics of imperatives and interrogatives. We use a modified DRS-language covering these extensions:

Syntax of DRSs

1. If $\{x_1, \ldots, x_n\}$ is a set of discourse referents, and $\{\gamma_1, \ldots, \gamma_m\}$ is a set of DRS-conditions, then the ordered pair $\langle\{x_1, \ldots, x_n\}, \{\gamma_1, \ldots, \gamma_m\}\rangle$ is a DRS;
2. If B_1 and B_2 are DRSs, then so is $(B_1;B_2)$. (merge of DRSs)

In DRS conditions, DRS-action-terms are used to deal with imperatives in dialogue, following [4]. We distinguish between atomic and composite DRS-action-terms:

Syntax of DRS-action-terms:

1. If B is a DRS, then δB is a DRS-action-term;
2. If A_1 and A_2 are DRS-action-terms, then so are $(A_1;A_2)$ (sequence) and $(A_1|A_2)$ (free choice).

Syntax of DRS-conditions:

1. If R is a relation symbol for an n-place predicate and $x_1 \ldots x_n$ are discourse referents then $R(x_1,\ldots,x_n)$ is a DRS-condition;
2. If x_1 and x_2 are discourse referents, then $x_1 = x_2$ is a DRS-condition;
3. If B is a DRS, then $\neg$B, $\Box$B, $\Diamond$B, ?B are DRS-conditions;
4. If B_1 and B_2 are DRSs, then $B_1 \vee B_2$, $B_1 \Rightarrow B_2$, $B_1?B_2$ are DRS-conditions;
5. If x is a discourse referent and B a DRS, then x:B is a DRS-condition;
6. If A is a DRS-action-term, and B a DRS, then [A]B and $\langle$A$\rangle$B are DRS-conditions;
7. If A is a DRS-action-term then !A is a DRS-condition.

Clauses 1 and 2 deals with basic DRS-conditions as in standard DRT [5]. Clause 3 introduces negation, the modal operators, and yes-no questions. Clause 4 covers disjunction, conditionals, and wh-questions. Hybrid DRS-conditions, formed by combining a discourse referent and a DRS, are introduced in clause 5. Clause 6 describes necessary and possible effects of actions, and clause 7 introduces DRS-conditions that describe actions that are commanded.

2.3 Inference

The DRS-language used for representing the semantic content is useful for dealing with various linguistic phenomena, but there are no efficient inference engines available that work directly on DRSs. Since there are a couple of efficient theorem provers and model builders for first-order logic available, we translate the DRSs to first-order formulas for inference tasks.

The core of the translation to first-order covering our extended DRS-language is based on the "standard translation" from DRT to first-order logic [5]. Basically, it introduces existential quantifiers for discourse referents, unless discourse referents are declared in the universe of the antecedent DRSs of an implication, in which case they will undergo universal quantification. Sets of DRS-conditions are translated into a conjunction of first-order formulas. To deal with the modal operators in our extended DRS-language, we use the ideas of the relational translation for modal logic to first-order formulas [6]. This means that all basic DRS-conditions of arity n are translated into first-order relations of arity $n+1$, where the additional argument plays the role of a possible world (a state). DRS-action-terms are translated into three-place relations where the first argument denotes the current state, the second argument describes the actions, and the third argument denotes the resulting state. The full translation is given in [7]. We illustrate the translation with an example.

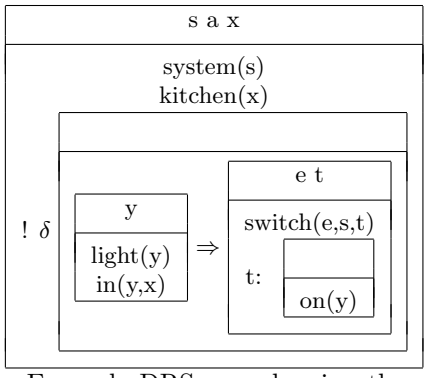

Figure 1: Example DRS paraphrasing the utterance "Switch every light in the kitchen on".

Consider the DRS for *Switch every light in the kitchen on!* in Figure 1. This example will get the following first-order translation:

$$\exists w \exists s \exists a \exists x \text{ (possible\_world(w) } \wedge \text{ system(w,s)} \wedge \text{ kitchen(w,x) } \wedge \exists v \exists a \text{ (action(w,a,v) } \wedge \forall y$$

$$(\text{light(a,y)} \wedge \text{in(a,y,x)} \rightarrow \exists e \exists t (\text{switch(w,e,s,t)} \wedge \text{on(t,y)}))))$$

We make use of two kinds of inference engines: theorem provers, and model builders. A theorem prover, when given a first-order formula, tries to decide whether it is true in all possible models. A model builder, on the other hand, tries to construct a possible model for the input formula. Figure 2 describes this idea in the form of an algorithm. Step 1 illustrates the fact that due to potential ambiguities appearing in the utterances conveyed by the user, semantic interpretation normally result in a set of DRSs. In Step 2, we make interpretation of the dialogue sensitive to context (see Section 2.4). We start with the empty set of interpretations (Step 3) and attempt to find consistent interpretations for each of the DRSs (Step 4). The theorem prover and model builder work in a complementary way. As soon as a proof is found the model builder does not need to further attempt to find a model because it will never succeed in doing so. However, if a model is found, the theorem prover can stop its attempt to find a proof, because it will never be able to do so. Finally, the system is instructed on how to react on the set of interpretations found (Step 5 and 6).

As the algorithm implies, the models generated by the model builder play a crucial role in interpretation. Figure 3 shows the model generated for the DRS of Figure 1. There are two interesting things about models. First, the flat representations of these models (they contain no recursion, in contrast to DRSs) ensure that they are extremely easy to process— all quantification and boolean structures are explicit in models. Second, the models produced by the model builder are minimal models, and contain no irrelevant information.

We use the model to deduce what actions need to be performed by the robot, or to answer questions posed by the user. For instance, in the model in Figure 3 a desired action for the robot is to supply power to the lights `d6` and `d7`.

The algorithm itself is implemented on top of the information-state approach to dialogue modelling [8], in which an agent's information state is updated on the basis of observed dialogue moves, leading to the selection of a new dialogue move to be performed by the agent. This is realised by a set of update rules, linking preconditions to effects. The dialogue manager repeatedly computes the effects of those update rules whose preconditions are satisfied by the current information state. Preconditions are expressed in terms of current values in the information state, while the ef-

1. Construct a (finite) set of DRSs $\mathcal{B}$ for a new utterance with respect to the previous DRS B_{old};
2. Construct a DRS C representing situational knowledge;
3. Initialise the set of interpretations $\mathcal{I}$ to $\emptyset$;
4. For each $B_i \in \mathcal{B}$:
   (a) Translate the compound DRS $(B_i; C)$ to the first-order representation ϕ_i;
   (b) Compute appropriate background knowledge stated as the first-order theory θ_i for ϕ_i;
   (c) Attempt to build a model for $(\theta_i \wedge \phi_i)$ by simultaneously performing:
      i. Give $(\theta_i \wedge \phi_i)$ to a model builder, possibly resulting in a model M_i; (for consistent interpretations).
      ii. Give $\neg(\theta_i \wedge \phi_i)$ to a theorem prover, possible resulting in a proof (for inconsistent interpretations);
   (d) If a proof is found cancel $4(c)i$. If a model is found add $\langle B_i, M_i \rangle$ to $\mathcal{I}$ and cancel $4(c)ii$;
5. If $\mathcal{I} = \emptyset$ perform a misunderstanding act and quit. Else perform an understanding act and select a preferred interpretation $\langle B_p, M_p \rangle$ from $\mathcal{I}$;
6. Use the information in M_p to decide whether to perform any actions. Replace B_{old} by B_p.

Figure 2: Update algorithm for inference-based dialogue management

```
D={d1,d2,d3,d4,d5,d6,d7,d8}
F(possible_world)={d1,d2,d3}
F(system)={(d1,d4),(d2,d4),(d3,d4)}
F(kitchen)={(d1,d5),(d2,d5),(d3,d5)}
F(action)={(d1,d2,d3)}
F(light)={(d1,d6),(d2,d6),(d3,d6),(d1,d7),
          (d2,d7),(d3,d7)}
F(in)={(d1,d6,d5),(d2,d6,d5),(d3,d6,d5),
       (d1,d7,d5),(d2,d7,d5),(d3,d7,d5)}
F(poweron={(d2,d6),(d2,d7)}
F(off)={(d1,d6),(d1,d7)}
F(on)={(d3,d6),(d3,d7)}
```

Figure 3: Example model

fects will change these values. The 26 update rules in our current system deal with establishing contact with the user, initiating clarification dialogues (when the recognition confidence score is below a certain threshold), answering questions, acknowledging requests and confirming or denying statements.

2.4 Background Knowledge

The interpretation algorithm in Figure 2 integrates background knowledge in two ways: in Step 2 situational knowledge is combined with the DRS; and in Step 4b other background knowledge is computed. These other sources of background knowledge can be divided in ontological knowledge, world knowledge and knowledge linking language to robot primitives.

Ontological knowledge comprises the various relationships between concepts that appear in the application domain. The two main relations that are expressed are subsumption and disjointness. For instance, we need to express that in all possible worlds a kitchen is a region, and that in all possible worlds a kitchen is not a corridor. Both can easily be coded in first-order logic:

$\forall w \forall x(\text{kitchen}(w,x) \rightarrow \text{region}(w,x))$
$\forall w \forall x(\text{kitchen}(w,x) \rightarrow \neg \text{corridor}(w,x))$

World knowledge subsumes all generalisations relevant to the application domain. The so-called 'frame axioms' belong to this class, and rules expressing physical laws. The frame axioms state properties of objects when certain actions are performed. For instance, if a robot moves from one region to another, its position will be different, but the positions of all other objects normally remain unaltered. Again, this can be relatively easily coded in first-order logic.

Knowledge linking natural language to robot primitives can be seen as a set of meaning postulates, rules that map non-logical symbols expressing the meaning of actions to symbols understood by the robot. Put differently, these rules establish the interface between the instruction language (i.e., English) and the hardware of the robot. An example of such a rule is "to switch on a device" and its translation into a DRS (Figure 1), which is mapped to the action `poweron`, as can be seen in the model generated for it (Figure 3).

Situational knowledge, finally, comprises the information of a specific situation in time. In the application domain of mobile robots, this typically consists of the regions and objects that are accessible to the robot, and the current positions of movable objects (including the robot itself). We do not code this knowledge directly into first-order knowledge, but rather use DRSs. This allows us to combine the situational knowledge with the information from the dialogue, enabling to interpret actions directly in the current situation.

3 Mobile Robot Control

3.1 Control Architecture

The control system is a looping algorithm reading sensory input and writing motor output at regular intervals. The sensory input comprises the readings of sonars, infrared sensors, bumpers and odometry. The sonars and infrared sensors are used for detecting occupied space and obstacles, and the bumpers notify if the robot hit an obstacle or wall. The odometry component measures the position and orientation of the robot. The motor output comprises translational and rotational velocity of the robot, and pan-tilt-zoom information for the camera unit.

We use units called "behaviours" to implement robot control. A behaviour has its own internal memory and uses the sensory input and motor output of the control system. While executing, it computes the motor output based on the sensory input and internal memory and updates the memory at regular intervals. It terminates when certain conditions are fulfilled. For example, a behaviour to move forward repeatedly sends velocity commands in a forward direction and terminates when an obstacle is detected or the robot has reached a specified location.

Behaviours are described as object-oriented classes (Figure 4). An instance of a behaviour class can be created using different parameters such as speed, direction, duration, or goal.

The robot needs to be able to deal with multiple goals. For instance, the robot may receive a new verbal command from the user when it is already undertaking an action. In that case, the robot must suspend the current behaviour and create and execute another. In our control system, we use a stack of behaviour instances to pursue multiple goals given by the user. The control system creates instances of behaviour classes, pushes them into the stack and deletes them when they terminate.

The behaviour that is executed is the one on the top of the stack. Any new command suspends the current executing behaviour and starts a new behaviour. When the new behaviour terminates, it is popped of the stack and the previous one resumes execution. The use of a stack simplifies the way the dialogue system sends commands to the control system. It doesn't need

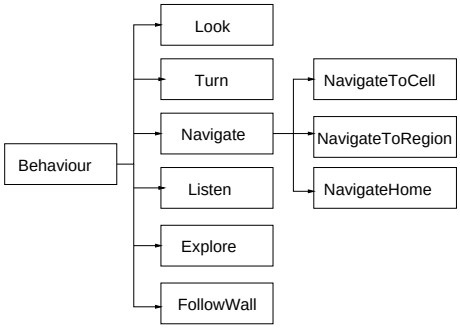

Figure 4: Part of behaviour class hierarchy

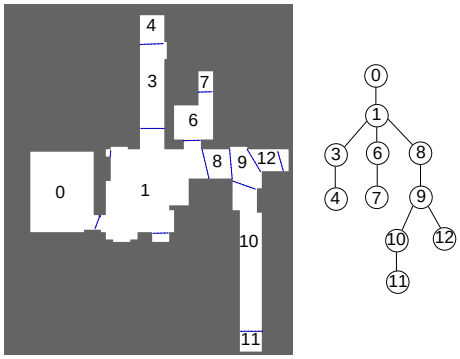

Figure 5: Representation of the environment

send commands to explicitly suspend, resume, terminate or remove behaviours to achieve multiple goals.

There are robot primitives creating simple behaviours such as *go(Distance, Speed)*, *turn(Angle, Speed)* and *look(Pan,Tilt)*, as well as commands introducing complex behaviours like *follow_wall(Distance)* and *move_to_region(N)*. In addition, sequences of actions such as *seq(zoom(100), look(0, 20), turn(90, 20))* can be commanded, in which case three behaviour instances are created and pushed into the stack.

3.2 Internal Map and Situational Knowledge

An internal map of the environment is used to plan a path to the target location and navigate the robot there following the path and avoiding obstacles. There are three layers in the map. The geometrical layer uses an occupancy grid to represent occupied and free space in the environment. The topological layer is automatically constructed from the occupancy grid by subdividing the free space into distinct topological regions corresponding to rooms or parts of the corridor (Figure 5). This is possible by creating a Generalised Voronoi Diagram [9, 10, 11].

The numbers in the geometrical map shown in Figure 5 are identifiers of topological regions which can be seen as nodes of an undirected graph. A further layer of representation connects the map of the navigation system with a vocabulary of semantic symbols used by the dialogue system. This is done by associating semantic descriptions to regions (Table 1). These descriptions can be arbitrarily complex. For instance, the DRS $\lambda p.(\langle [x, y], [office(x), of(x, y), tim(y)] \rangle; p(x))$ could be used to denote Tim's office.

Table 1: Semantic labels

Region	**Description**	**Semantic label**
0	the office	λp.(x | OFFICE(x) ;p(x))
1	the hallway	λp.(x | HALLWAY(x) ;p(x))
3, 4, 8-12	the corridor	λp. x | CORRIDOR(x) ;p(x))
6	the rest room	λp.(x | REST_ROOM(x) ;p(x))

4 Implementation and Evaluation

We implemented a talking mobile robot system, Godot, integrating a dialogue interface and a control system on an RWI Magellan Pro mobile robot platform with an on-board PC and a laptop computer on the top (Figure 6).

Godot is a complete end-to-end system for human-robot communication, where users can start a new dialogue simply by addressing the robot in spoken language, with the robot responding in real time. Although we have not yet reached the stage of carrying out formal usability studies, we have tested Godot when visitors to our department unfamiliar with Godot have controlled its movements and its camera with success. Another evaluation took place over two days in the Scottish museum, where museum visitors could address the Godot in a simulated house. This was a different environment from Godot's usual one (the basement of our department) but our modular approach allowed us to quickly adapt it—in fact we only needed to change the internal map. In these interactions with new users we collected the

spoken data and used that to improve the system, by extending the vocabulary and grammar constructions. We have published videos of Godot in action at `http://www.ltg.ed.ac.uk/godot/`.

Figure 6: Godot, the mobile robot

5 Conclusions

We have developed an effective interface between natural language semantics and the robot control layer, thus enabling users to refer to locations in a natural way, rather than resorting to expressions like *go to grid cell 45-66* or *you are in region 12*. The framework is practically domain-independent: a change of environment would only require a change of the internal map and possibly a new lexicon.

Our robot employs an inference-based approach to dialogue understanding, with the aim to find a *consistent* semantic representation capturing the meaning of the dialogue. If a consistent interpretation cannot be found, we have got a signal that something is going wrong in communication. Such a situation might arise from disagreement or misunderstanding between dialogue participants. Inference also contributes to ambiguity resolution (for instance the resolution of pronouns and other anaphoric expressions) and helps finding preferred interpretations. Moreover, with the help of *model building*, inference can be used for performing actions and answering questions, too. The resulting dialogue system is formulated on an abstract level, and is easily portable to robots in new domains or with different applications.

Future work will address the interpretation of vague expressions (*the end of the corridor*), metonymic expressions (*go to the door*, where an artifact is interpreted as a location), together with commands which require Godot to reason and talk about its current activities (*continue going to the kitchen*).

Acknowledgements

This work was supported by IBL for mobile robots (EPSRC GR/M90160) and the EU Project Magicster (IST 1999-29078). We thank Nuance for permission to use their software and tools.

References

[1] Johan Bos, "Compilation of Unification Grammars with Compositional Semantics to Speech Recognition Packages," *COLING 2002. Proceedings of the 19th International Conference on Computational Linguistics*, 106–112, 2002. 1982.

[2] Hans Kamp, "A Theory of Truth and Semantic Representation," *Formal Methods in the Study of Language*, 277–322, 1982.

[3] A. Rob Van der Sandt, "Presupposition Projection as Anaphora Resolution," *Journal of Semantics*, vol.9 333-377, 1992.

[4] Alex Lascarides, "Imperatives in Dialogue," *Proceedings of the 5th International Workshop on Formal Semantics and Pragmatics of Dialogue (BI-DIALOG)*, 1–16, 2001.

[5] H. Kamp and U. Reyle, *From Discourse to Logic; An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*, Kluwer, 1993.

[6] Robert C Moore, "Reasoning about Knowledge and Action," *SRI International Technical Report*, 1980.

[7] Johan Bos, "Implementing the Binding and Accommodation Theory for Anaphora Resolution and Presupposition Projection," *Computational Linguistics*, vol.29, 2, 179–210, 2003.

[8] David Traum et al, "A model of dialogue moves and information state revision," *Trindi Technical Report D2.1,* 1999.

[9] Jean Claude Latombe, "Robot Motion Planning," *Kluwer Academic Publishers*, 1991.

[10] Sebastian Thrun, "Learning Maps for Indoor Mobile Robots," *Artificial Intelligence*, vol.99(1), 21–71, 1998.

[11] Christian Theobalt, "Navigation on a Mobile Robot," *Master's Thesis, University of Edinburgh*, 2000.