Pamini: A framework for assembling mixed-initiative human-robot interaction from generic interaction patterns

Julia Peltason and Britta Wrede
Applied Informatics, Faculty of Technology
Bielefeld University, Germany
jpeltaso, bwrede@techfak.uni-bielefeld.de

Abstract

Dialog modeling in robotics suffers from lack of generalizability, due to the fact that the dialog is heavily influenced by the tasks the robot is able to perform. We introduce interleaving interaction patterns together with a general protocol for task communication which enables us to systematically specify the relationship between dialog structure and task structure. We argue that this approach meets the requirements of advanced dialog modeling on robots and at the same time exhibits a better scalability than existing concepts.

1 Introduction

The need for interaction modeling on robots is widely acknowledged, not only for instructing them but also for enabling them to learn from humans within interaction. Yet, today's robotic systems often do not have a dedicated dialog system but employ simple command-matching techniques (e.g. (Böhme et al., 2003)). Other systems rely on finite-state based dialog managers (e.g. (Bauer et al., 2009)) that couple dialog and task management which hampers maintainability and reuse and does not scale well for more complex interaction scenarios.

On the other hand, concepts for reusable dialog frameworks have been established within the spoken dialog community for traditional information-seeking domains where the system first collects the required parameters, then presents the desired information to the user, e.g. an accommodation or travel information (e.g. (Bohus and Rudnicky, 2009)). However, these concepts are not directly transferable to robotics, due to the situated nature of human-robot interaction. A robot perceives the world and acts on it – and it is supposed to communicate about that. For instance, a robot manipulating objects needs to be able to verbalize its intended actions and give feedback about its internal state, but it also needs to react on feedback comments or corrections uttered by the human tutor. In order to equip a robot with such fine-grained communication skills, tight integration of action execution into interaction is required. In simple terms, while the dialog frameworks mentioned can be referred to as information-oriented, we need rather action-oriented approaches for robotics.

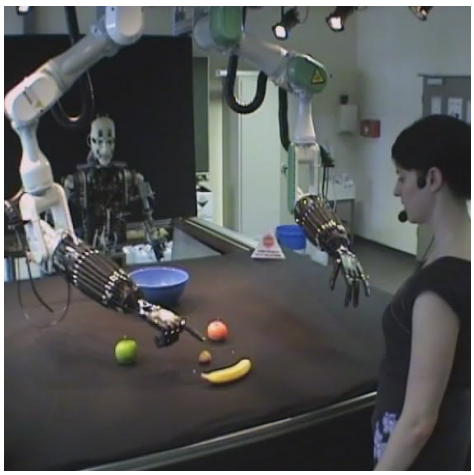

Figure 1: The Curious Robot set-up.

R: What is that?	⟨points⟩
H: This is a lemon.	
R: A melon. OK.	
H: No, I said 'lemon'!	
R: A lemon. OK.	
R: How should I grasp the lemon?	
H: That way.	⟨demonstrates grip⟩
R: Alright, with the power grasp.	
R: I am going to grasp it	⟨starts grasping⟩
R: I start grasping now.	
H: Robot, what objects do you know?	
R: Lemon.	
R: OK, I finished grasping.	⟨stops grasping⟩

Table 1: A typical interaction with the Curious Robot.

2 The Pamini framework

This paper presents the Pamini (Pattern-based mixed-initiative) human-robot interaction framework. Pamini proposes a new approach for dialog

Proceedings of SIGDIAL 2010: the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pages 229–232,
The University of Tokyo, September 24-25, 2010.

modeling on robots that includes (1) a task-state protocol providing a fine-grained interface to the complex domain processing of the robotic system and (2) the concept of generic interaction patterns that support rapid prototyping of human-robot interactions and can be combined in a flexible way.

Previous versions of the Pamini framework have been applied in several mixed-initiative learning scenarios. For example, in the Home-Tour scenario a mobile robot builds up a spatial model of its environment and gradually improves its model by attempting to obtain information from the human (Peltason et al., 2009). In the Curious Robot scenario shown in figure 1, an anthropomorphic robot learns to label and grasp objects, applying a proactive dialog strategy that provides guidance for untrained users (Lütkebohle et al., 2009). A dialog excerpt is shown in table 1.

2.1 The task state protocol

A dialog system for robotics needs to coordinate with a number of components, e.g. for perceptual analysis, motor control or components generating nonverbal feedback. To realize this, we use the concept of *tasks* that can be performed by components. Tasks are described by an execution state and a task specification containing the information required for execution. A protocol specifies task states relevant for coordination and possible transitions between them as shown in figure 2. Task updates, i.e. updates of the task state and possibly the task specification, cause event notifications which are delivered to the participating components whereupon they take an appropriate action.

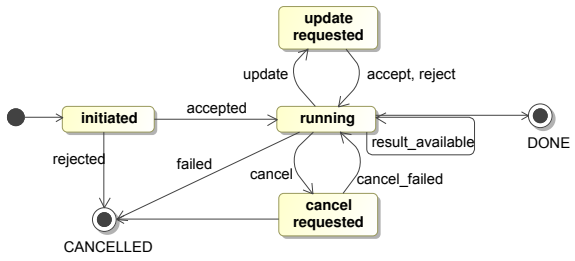

Figure 2: The task life-cycle. A task gets initiated, accepted, may be cancelled or updated, may deliver intermediate results and finally is completed. Alternatively, it can be rejected by the handling component or execution may fail.

Tight integration with action execution A robot performing e.g. a grasp action supervised by the human requires multi-step communication between the dialog system and the arm control as illustrated in figure 3. Generally, with the *accepted* state, the proposed protocol enables the dialog system to provide feedback during slow-going actions indicating the internal system state. Further, with the *update* and *result_available* states, it supports the modification of the task specification during execution and thus gives the robot the ability to react to comments, corrections and commands on-the-fly.

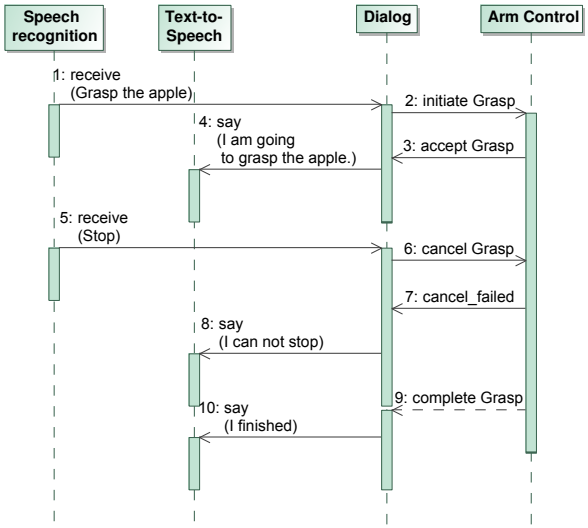

Figure 3: The component communication for a grasp action requested by the human. As the dialog manager (DLG) receives the grasp command, it *initiates* a grasp task which is *accepted* by the arm control. The DLG is notified about the task state update and acknowledges task execution. As the human commands cancelling, the DLG sets the task state *cancel*. Since the arm control fails to cancel the task, it sets the task state *cancel_failed* which the DLG reacts on by expressing an excuse. Finally the task is *completed*, and the DLG acknowledges successful task execution.

Mixed-initiative interaction The Pamini dialog manager offers dialog tasks for other components, e.g. greeting the human, informing the human about anything or conversely requesting information from the human. While *human initiative* is realized whenever input from a speech understanding component is received, *robot initiative* occurs when a system component requests a dialog task to be executed. Situation permitting, the dialog manager will accept the dialog task, go into interaction with the human, and finally complete the dialog task. Thus, it can react to the system's and the human's initiative using the same task-state protocol

Learning within interaction The task state protocol supports robotic learning within interaction by establishing mechanisms for information transfer from the dialog system to the robotic subsystem. Once information is available from the human, Pamini augments the task specification

with the new information and sets the task state *result_available*. Since this transition may be taken multiple times, given information can be corrected. Also, mixed-initiative enables *active learning*, where the learner provokes a situation providing new information instead of waiting until such situation eventually presents itself.

2.2 Interaction patterns

In an interaction, dialog acts are not unrelated events, but form coherent sequences. For example, a question is usually followed by an answer, and a request is typically either accepted or rejected. Influenced by the concepts of adjacency pairs (Schegloff and Sacks, 1973), conversation policies (Winograd, 1986) and software design patterns, we propose the concept of *interaction patterns* that describe recurring dialog structures on a high level. Interaction patterns can be formalized as transducer augmented with internal state actions, consisting of

- a set of human dialog acts H and a set of robot dialog acts R, e.g. *H.request* or *R.assert*;
- a set of incoming task events T, e.g. *accepted* or *failed*;
- a set of states S representing the interaction state;
- a set of actions A the dialog manager performs, e.g. initiating or updating a task or reset interaction;
- an input alphabet $\Sigma \subset (H \cup T)$;
- an output alphabet $\Lambda \subset R$;
- a transition function $T : S \times \Sigma^* \longrightarrow S \times A^* \times \Lambda^*$.

By admitting task events as input and internal actions that perform task initiation and update, the dialog level is linked with the domain level. The patterns have been implemented as statecharts (Harel, 1987), an extended form of finite state machines, which provides both an executable model and an understandable graphical representation as shown in figure 5. For instance, the *cancellable*

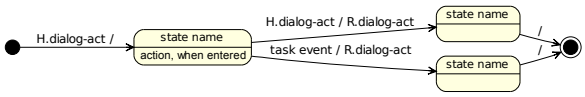

Figure 5: Interaction patterns are represented as transducer that takes as input human dialog acts and task events and produces robot dialog acts as output.

action request pattern shown in figure 4 describes an action request initiated by the human that can be cancelled during execution. The normal course of events is that the human requests the action to be executed, the dialog manager initiates the domain task, the responsible system component accepts execution so that the dialog manager will assert execution. Finally, the task is completed and the robot acknowledges. In contrast, the *correctable information request* pattern is initiated by the human. Here, on receiving the respective dialog task request, the dialog manager will ask for the desired information and accept the dialog task. Once the human provides the answer, the robot will repeat it as implicit confirmation that can be corrected if necessary. Table 2 lists all patterns that have been identified so far.

Initiated by user	Initiated by robot
Cancellable action request	Self-initiated cancellable action
Simple action request	Self-initiated simple action
Information request	Correctable information request
Interaction opening	Simple information request
Interaction closing	Clarification
Interaction restart	
System reset	

Table 2: Available interaction patterns.

Pattern configuration The patterns themselves do not determine what kind of task is to be executed or what kind of information to obtain exactly. These specifics are defined by the configuration associated with each pattern, and a concrete scenario is realized by configuring a set of patterns using a domain-specific language and registering them with the dialog manager.

In detail, it needs to be specified for the human's dialog acts what kind of (possibly multimodal) input is interpreted as a given dialog act which is done by formulating conditions over the input. For the robot's dialog acts, their surface form needs to be specified. Up to now, speech output and pointing gestures are implemented as output modalities and can be combined. Moreover, also the task communication needs to be configured, i.e. the task specification itself as well as possible task specification updates. In addition, the developer can define context variables and use them to parameterize the robot's dialog acts and in task specification updates. This is how e.g. for the robot's information request the answer is transferred from the human to the responsible system component.

Interleaving patterns during interaction During interaction, the registered patterns are employed in a flexible way by admitting patterns to be interrupted by other patterns and possibly resumed later which leads to interleaving patterns. By default, simpler patterns are permitted to be nested within temporally extended patterns. For example, it seems reasonable to permit monitoring questions uttered by the human to be embedded in the robot's slow-going grasp execution as shown

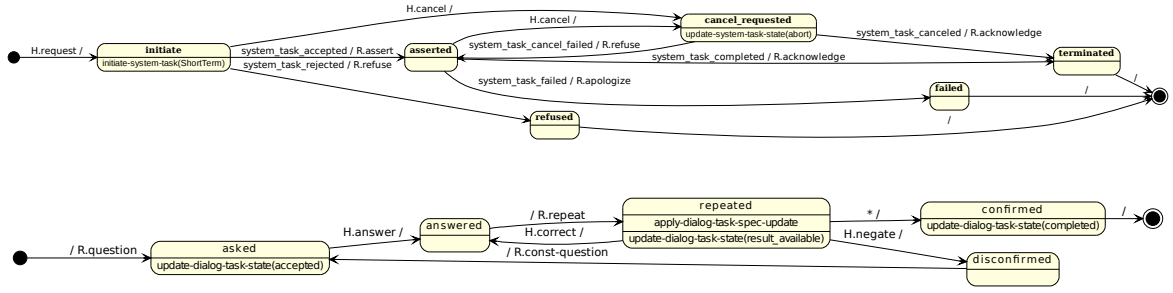

Figure 4: Two example interaction patterns. *Cancellable action request*: an action request which is initiated by the human and can be cancelled during execution. *Correctable information request*: an information request with implicit confirmation initiated by the robot where information can be corrected later if necessary.

in table 1 which equips the robot with multitasking capabilities. Interleaving is realized by organizing active patterns on a stack. Whenever an input is received, the dialog manager attempts to interpret it in the context provided by the topmost pattern. If it fails, the lower and inactive patterns are tried.

3 Discussion and Conclusion

The presented approach to dialog modeling on robots relies on the concept of interaction patterns that constitute configurable (and thus reusable) building blocks of interaction. A fine-grained task state protocol links dialog and domain level. With interleaving patterns, flexible dialog modeling is achieved that goes beyond current state-of-the-art dialog modeling on robots. Further, by encapsulating both the subtleties of dialog management and the complexity of component integration, the proposed interaction patterns support rapid prototyping of human-robot interaction scenarios.

The evaluation of the approach needs to examine framework usability, framework functionality and usability of the resulting dialogs. With respect to framework usability, we already showed that developers unfamiliar with the framework were able to build a simple interaction scenario within one hour (Peltason and Wrede, 2010). With respect to framework functionality, we demonstrated that the robot's mixed-initiative interaction capabilities enable human and robot in the Home-Tour scenario to jointly build up a common representation of their environment and even compensate for classification errors (Peltason et al., 2009). As to dialog usability, a video study indicates that the Curious Robot's proactive dialog strategy guides unexperienced users (Lütkebohle et al., 2009). Further, given a dialog system architecture that supports rapid prototyping, comparative studies become possible. Therefore, we currently prepare a study to compare the curiosity strategy with a user-directed strategy that provides more freedom but also more uncertainty to the user. Last but not least, we will evaluate the patterns themselves and pattern interleavability. Are users likely to interrupt a robot's action by asking questions or even giving new commands? Also, are there other kinds of interaction patterns that occur in a real interaction but are not captured yet?

References

A. Bauer, D. Wollherr, and M. Buss. 2009. Information retrieval system for human-robot communication asking for directions. In *International Conference on Robotics and Automation*.

H.-J. Böhme, T. Wilhelm, J. Key, C. Schauer, C. Schröter, H.-M. Groß, and T. Hempel. 2003. An approach to multimodal human-machine interaction for intelligent service robots. *Robotics and Autonomous Systems*, 44(1).

D. Bohus and A. I. Rudnicky. 2009. The ravenclaw dialog management framework: Architecture and systems. *Computer Speech & Language*, 23(3):332–361.

D. Harel. 1987. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8:231–274.

I. Lütkebohle, J. Peltason, L. Schillingmann, C. Elbrechter, B. Wrede, S. Wachsmuth, and R. Haschke. 2009. The curious robot - structuring interactive robot learning. In *International Conference on Robotics and Automation*.

J. Peltason and B. Wrede. 2010. Modeling human-robot interaction based on generic interaction patterns. In *AAAI Technical Report: Dialog with Robots*. submitted.

J. Peltason, F. Siepmann, T. Spexard, B. Wrede, M. Hanheide, and E. Topp. 2009. Mixed-initiative in human augmented mapping. In *International Conference on Robotics and Automation*.

E. A. Schegloff and H. Sacks. 1973. Opening up closings. *Semiotica*, 8(4):289–327.

T. Winograd. 1986. A language/action perspective on the design of cooperative work. In *Conference on Computer-supported cooperative work*.