Towards a Robotic Dialogue System with Learning and Planning Capabilities

Karolina Eliasson
Linkoping University
Department of Computer and Information Science
SE - 581 83 Linköping
karel@ida.liu.se

Abstract

We present a robotic dialogue system built on case-based reasoning. The system is capable of solving references and manage sub-dialogues in a dialogue with an operator in natural language. The approach to handle dialogue acts and physical acts in a unison manner together with the use of plans and sub-plans makes the system very flexible. This flexibility is used for learning purposes where the operator teaches the system a new word and the new knowledge can directly be integrated and used in the old plans. The learning from explanation capability makes the system adaptable to the operator's use of language and the domain it is currently operating in. The implementation of a case-based planner suggested in the paper will further increase the learning and adaptation degree.

1 Introduction

Human-Robot Interaction is a broad and interesting area which deals with the interaction between a human and a physical embodied robot. In our work we have focused on interaction in natural language with an autonomous Unmanned Aerial Vehicle (UAV). In a dialogue with such a robot, the dialogue manager must be able both to interpret the utterance from the operator, receive additional information to be able to react correctly to the utterance, and perform a sequence of actions. To do this, it has to distinguish between *dialogue acts* and *physical acts*. A dialogue act can be to ask a clarifying question to the operator and a physical act can be to perform the high-level command `land`. These acts must be executed in a correct order to solve the problem at hand. If the information is not sufficient for the dialogue manager to complete the task it has to ask for complementary information from the operator. In such a case a discourse model is needed to solve anaphoric references and to manage sub-dialogue.

In case the robot will be used by the same operator, it can be convenient if the robot can adapt to the operator's use of language. In that case a flexible control system is needed which gives the operator the opportunity to explain what a word means and how a task, new to the dialogue system, shall be performed. In this manner, the dialogue system can be adapted not only to the current operator's use of language but also to the different tasks that can be of interest in different flight scenarios. To further increase the usefulness of the dialogue system, we would like it to be able to perform mixed-initiative planning with user interaction and reuse old plans and experiences.

In this paper we will describe a dialogue system which uses case-based reasoning (CBR) to find suitable responses to the utterances from the operator and messages from the robot. CBR is a form of machine learning where the system stores problems and their corresponding solutions in a case base. When a new target case enters the system, it searches the case base for similar cases. When the most similar case is found, its corresponding solution is adapted to the new target case and the new solution is returned. The new target case and its solution is then stored in the case base for future use. See for example [Aamodt, 1994] for an overview. We are also addressing case-based planning (CBP) where a generative planner is used when there are no suitable plans in the case base. The planner should not plan a new solution from scratch but reuse the old plans as much as possible.

CBR provides our dialogue system with a simple and modular design where both the dialogue acts and the commands regarding the physical acts can be represented in a uniform way. New functionality is directly added by writing new cases and storing them in the case base. New domain knowledge similar to existing knowledge can be added to the system in a simple manner. It can directly be used by the system without any additional changes to the case base or the case-bae manager, due to the flexible and adaptable nature of the CBR design. This provides us with the facility of letting the system incorporate new information, such as new words or knowledge about the physical world, into the system. This knowledge can then directly be used by the cases in the case base, hence giving the system mechanisms for updating its own knowledge and increasing its performance. The new information can be obtained from dialogue with an operator. Because phrase matching is necessary both in CBR and in discourse modeling, in the latter to allocate incoming new phrases to the correct dialogue thread, it makes CBR and discourse modeling a suitable combination without producing any additional overhead.

Our dialogue system CEDERIC, Case-base Enabled Dialogue Extension for Robotic Interaction Control, which addresses the above mentioned features is still under construction and this paper will discuss both implemented features and some ideas and work in progress. Most of the parts, such as the CBR framework which uses existing plans, the discourse model, and the learning from explanation dialogues are implemented and tested in a simulator, but the work with the planner is still ongoing.

2 CEDERIC

CEDERIC is a dialogue system designed for dialogue with a physical robot, in particular the WITAS autonomous UAV. The WITAS project focuses on the development of an airborne computer system that is able to make rational decisions about the continued operation of the aircraft, based on various sources of knowledge including pre-stored geographical knowledge, knowledge obtained from vision sensors, and knowledge communicated to it by data link [Doherty *et al.*, 2000]. The UAV used in the project is a Yamaha RMAX helicopter which an operator can control by high level voice commands or by written commands. The operator can ask the UAV to perform different tasks and request information from it.

CEDERIC consists of a *parser*, a *case base*, *domain knowledge*, a *discourse module* and a *case-base manager* as shown in Figure 1. The parser parses the sentence given by the operator and generates a parse tree. The parse tree is matched against cases in the case base by the case-base manager. The discourse module is responsible for maintaining a discourse model of the dialogue so far to be able to interpret the operator's sentences in the right context. The discourse model helps the system to interpret references which may refer to sentences earlier in the dialogue. The domain knowledge contains an ontology of the world as the robot knows it and a categorization of the world items. The purpose is twofold. It serves as a world representation which gives CEDERIC knowledge about which buildings there are in the known world, what kind of buildings they are, where they are places, and their attributes such as color and material. It also gives CEDERIC fundamental knowledge about categorization e.g. which items that can be called buildings in the dialogue and which can not.

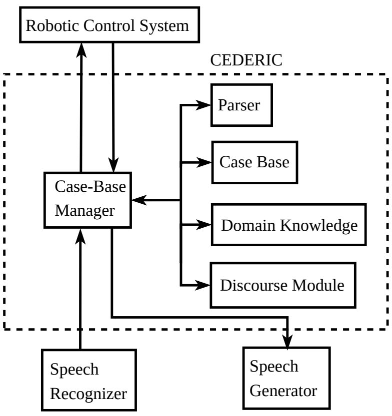

Figure 1: Architecture of CEDERIC.

```
(PHRASE1
 (VERBPHRASE_IMP}
  (VERBPHRASE_IMP7 (FLY FLY) (TO TO)
   (NP_FLY_TO
    (NP_BUILD (DET THE)
     (NOUN_BUILD
      (NO_CAT SUPERMARKET)))))))}
```

Figure 2: Parse tree with an unknown word.

The operator can choose to use either speech or text for the input to the dialogue system. The speech recognizer used is the off-the-shelf product Nuance and the speech generator used is one of the off-the-shelf products Festival or Brightspeech. When learning a new word using speech recognition, one can choose between having a considerably bigger grammar for the speech recognition than the dialogue manager and only consider learning in the dialogue manager, or provide the new word in text form in the learning phase and then compile it into the speech recognition grammar if that can be done at runtime. We have chosen the second approach where the unknown words are provided in text and the learning phase extends the grammar and lexical rules in the parser within CEDERIC. The grammar and the lexical rules are the same as those for the speech recognizer, so the extensions should be easy to integrate with the speech recognition system if it allows runtime updates of the grammars.

Regardless of wether a speech recognizer is used or not, a sentence from the operator arrives to CEDERIC in plain text format. It is parsed, processed in the CBR engine and returns either a new phrase in text format to be sent to the speech generator, or a request to the robotic control system. The robot requests are in a format similar to KQML [Finin *et al.*, 1994]. The response the robot sends back to CEDERIC is also in a KQML-like format. Those messages are sent to CEDERIC on a different channel than the one the operator uses. This is because the response should not be parsed in any way, but be processed directly in the CBR engine.

2.1 Parser

The text string that results from the speech recognizer or is obtained directly from the operator if he or she provides the input in text form, is parsed by a chart parser. The chart parser returns a parse tree of the input sentence. If some of the words are unknown to the parser it provides all possible solutions where the unknown words are first labeled with the possible label and then with the label `no_cat`. Figure 2 shows an example of a parse tree where the word `SUPERMARKET` is unknown. It has been categorized with the possible category `NOUN_BUILD`. When there is an unknown word in the sentence, all the possible parses are collected in a parse tree which is sent to the case-base manager. The case-base manager can then try the different possible matches and see if it finds a suitable match with one of its cases.

2.2 Case Design

To be able to reuse parts of a solution, each solution to a sentence is divided into smaller units, each of which performs an isolated part of the solution. These units are called *atomic cases*. For the robot to know how to construct the whole solution out of the atomic cases it needs a plan. Already known and tested plans are stored in *plan cases*. Both plan cases and atomic cases have a *problem part*. This part of the case consists of the name of the case and the problem information that the target case need to have to match the case. The problem information can be either:

- a parse tree
- the result from a previously executed atomic case or plancase
- a response from the robotic control system in KQML-like format.

They also have a *discourse update part* which updates the discourse model of the dialogue with new information about the current dialogue.

The *solution part* of a plan case is a plan. The plan consists of the name of other plan cases and atomic cases. The solution part of an atomic case consists of procedures which return an answer. An atomic case can for example be *get-referenceid* which takes a parse-tree of a sentence like *white church*, checks its domain knowledge for a match and returns every matching reference id to such a building. The next item in the plan could then use this information to produce another step in the plan.

2.3 Case Matching

When a case match is performed, the problem information of the case is matched against the target case. The whole target case does not have to match the problem information completely to be considered a perfect match. It is sufficient that all the problem information occurs in the correct order in the target case. This approach makes the cases more general and does not leave it to the case-base manager to adapt each target case in a suitable way to obtain the new target case match. This is very useful when you want a case to be able to match several similar target cases. An example in our domain is the command `fly to the red hospital`. The parse tree for this sentence looks like this:

```
(PHRASE1
 (VERBPHRASE_IMP
  (VERBPHRASE_IMP7 (FLY FLY) (TO TO)
   (NP_FLY_TO
    (NP_BUILD (DET THE)
     (ADJPHRASE_BUILD (ADJ_COLOR RED))
      (NOUN_BUILD HOSPITAL))))))
```

If the whole parse tree where to be matched, we would need a new case for every possible combination of color and building. Moreover, we would have to write new cases for the similar cases where another attribute instead of color is given, or where no attribute at all is provided in the sentence. This could however be solved with adaptation in the case-base manager before the match but the more similar target cases there are, the more information about adaptation of the target cases has to be hard coded into the case-base manager with no possibility to learn or expand the knowledge by experience. We have taken the decision to let the problem information leave out some of the specific information in the match and act as a general case. The problem information for a case matching the example above can look like this:

```
(PHRASE1
 (VERBPHRASE_IMP
  (VERBPHRASE_IMP7 (FLY FLY) (TO TO)
   (NP_FLY_TO ))))
```

The cases are ranked dependent on how well they match the target case. The ranking includes two parameters:

- How well the target case covers the case.
- How well the case covers the target case.

Both parameters are expressed on a scale from 0 to 100. The first parameter is obviously the most important one and the cases are firstly sorted by it and secondly by the the second one. That is, we want cases for which the information in the target case are enough to fulfill the requirement of the case and if we find several such cases, we prefer cases which match as much of the target case as possible.

2.4 Discourse Module

For a dialogue in natural language to run smoothly, the participants have to know the history of it. If a computer dialogue system will be able to work properly in such a natural dialogue with a human user it has to maintain a discourse model of the dialogue so far to be able to interpret the utterances of the user in the right context. The discourse model helps the system to interpret references to utterances earlier in the dialogue.

The discourse model implemented in CEDERIC is a slightly modified version of the one described in [Eliasson, 2005]. Its design is highly inspired by the discourse model presented in [Pfleger *et al.*, 2003] for the SmartKom project. It is built up of four different objects, that are linked to one another in a hierarchical manner to represent the meaning of the dialogue.

The linguistic objects. These objects are furthest down in the chain of objects and thus most specific on the word level. They contain information of how the nouns in the dialogue where uttered. They could for example have been references by the word `it` or by a noun and a determinant.

The discourse objects. These objects contain the different nouns together with their attributes mentioned in the dialogue. A discourse object can also be composite. An enumeration of several objects can be seen as a discourse object representing the enumeration as such, and this object contains the enumerated objects as its children. This gives CEDERIC the opportunity to understand references referring to the order of the enumerations, e.g. `the first one`. The discourse objects have a link to the corresponding linguistic object.

The dialogue objects. These objects group those sentences having the same direct goal, with their respective associated information. The sentence `fly to the hospital` gives for example, when it is executed, a dialogue object grouping the sentences `fly to the hospital`, `ok` and `I am at the hospital now`. If any sub-dialogues come up, they will be saved in a new dialogue object with their direct goal to clarify some matter in the dialogue. Dialogue objects contain information about the topic of the dialogue, which discourse objects were created due to the utterances, and which future utterances this dialogue object expects before considering the dialogue or the sub-dialogue to be completed. These expectations on future dialogue are saved in a modified *initiative-response (IR) unit* [Ahrenberg *et al.*, 1991]. Unlike the original IR-units described by Ahrenberg, IR-units in our context can contain more than two subelements. This is necessary because they shall also be able to represent the response from the robot when the system sends a request. The `fly to the hospital` example above shows such an example.

The global focus space. The various objects in the dialogue layer which belongs to the same dialogue, including sub-dialogues, are grouped together in a top object called the global focus space. It contains information about the main topic of the dialogue and about which dialogue objects belong to it. Each global focus space also keeps track of the discourse object last mentioned, to be able to resolve references such as `it`. This is known as the *local focus stack*. The last mentioned discourse object is said to be in focus.

To keep track of the current dialogue in focus, CEDERIC saves the different global focus spaces in a stack called the *global focus stack*. The global focus space on top of the stack is said to be the one in focus. If every IR-unit belonging to a global focus space is closed, that is, has received all its subelements, the global focus space is marked as closed and removed from the stack. Several dialogues can be open and ongoing at the same time and are thus members of the stack but only one dialogue can be in focus at the same time.

When a new sentence is recognized and matched with a case in the case base, the discourse update part of the case is executed. This information creates new objects in the different layers and links them together to reflect the identified dialogue.

2.5 Case-Base Manager

The case-base manager is the main engine responsible for the data flow in the dialogue system. It matches the target case with the cases in the case base as described in section 2.3, selects the first case in the ranked case list, and evaluates the first item in the selected case's plan. During the execution of the plan it stores the plan history. If one of the plan items in the current plan does not match the input from a previously executed plan item, the case-base manager selects the next case in the ranked case list and checks if the history equals the first part of that case's plan. If so, the case-base manager can switch to that plan and continue with it. In this way, CEDERIC can cope with information not known from start. This is important when the dialogue system has a sub-dialogue with the operator to clarify some information. With the operator in the loop, the data gets unpredictable, because CEDERIC can not know which answer the operator will provide in advance and hence the plan chosen from the beginning can be found to be wrong.

The case-base manager is also responsible for keeping track of the different ongoing dialogues. When a new sentence arrives from the operator, it could be one of the following cases:

- The operator continues the current dialogue, possibly by the start of a sub-dialogue.
- The operator returns to an older non-completed dialogue with is not presently in focus.
- The operator starts a new dialogue, possibly without ending the recent dialogue properly.

It is important to recognize which of these three cases it is, to be able to provide the correct discourse for the evaluation of the sentence. The case-base manager starts by checking if there are any ongoing plans and in that case the case-base manager tries to go on with it, possibly by changing to another plan as described above. If it does not give any result, the manager tests to execute one of the saved older non-completed plans with its corresponding discourse. If that fails as well it starts a new dialogue with an empty discourse model.

2.6 Learning from Explanation

CEDERIC can ask the operator for guidance if the parsed sentence contains the category `no_cat`. As described in section 2.1, the parser generates all possible parses with both the `no_cat` category and the possible category. To be able to catch those `no_cat` parses, CEDERIC needs to have cases that match both the sentence where the `no_cat` was found and the `no_cat` itself. By providing CEDERIC with such cases, it can ask the operator questions that makes him or her explain how CEDERIC should react. We take the parse tree in Figure 2 as an example. It has been generated from the sentence `Fly to the supermarket` where the word `supermarket` is not in the parser's vocabulary. The case base contains a case where the problem part is the parse tree in Figure 2, but with the word `supermarket` omitted (obviously, because we want the case to match all unknown words!). This case is fully covered by the target parse tree which characterizes a good match and its plan is executed, which, depending on the answers from the operator, can result in the dialogue in Figure 3. After the execution of this dialogue, the vocabulary of the parser is extended to also contain the word `supermarket` which is a `building`. The newly gained information such as the color and the category of the supermarket will be saved in the domain knowledge as well. The next time the operator wants the robot to fly to the supermarket, the sentence will be correctly parsed and the case that matches such a sentence will provide a correct solution.

The dialogue shown above where the operator guides the UAV to an unknown building can be used with any word that the operator wants to use as an identifier for a particular building. This particular plan case is however not general enough to handle other types of words, which are not buildings. They can be handled analogously by using other plan cases which

Operator:	Fly to the supermarket.
CEDERIC:	I do not know what a supermarket is. Is it a building?
Operator:	Yes.
CEDERIC:	Can you give me a reference object near the supermarket so I can start by flying to it?
Operator:	It is near the red hospital.
CEDERIC:	Ok, I will start by flying to the hospital and then I will ask for more guidance.
CEDERIC:	I am at the hospital now.
CEDERIC:	In which direction shall I fly to look for the. supermarket and what characteristics does it have?
Operator:	Fly eastwards and look for a blue building.
CEDERIC:	I start fly and look for the supermarket. I will stop when I see it.
CEDERIC:	I have found the supermarket now and hover over it.

Figure 3: An example of a teaching situation between the operator and CEDERIC.

describes the sub-dialogue suitable to learn CEDERIC how to react properly on them.

The main approach is generally applicable and a similar solution has been tested for teaching CEDERIC new verbs, which are synonyms for already known commands. The sentence `Go to the red hospital`, where the word `go` is unknown to the parser and hence categorized with the category `no_cat` and the potential category `fly`, is matched with a case whose response is to ask the operator if the words `fly` and `go` are synonyms. If the operator answers yes, CEDERIC will process the command as a fly command and put the new word `go` in the vocabulary. It also creates a new case which changes the word `go` to `fly` and calls the plan case for fly commands.

It should also be possible to learn CEDERIC new composite commands in a similar fashion, if the building blocks are already known. This is however not tested yet.

2.7 Planner

Case-based Planning (CBP) is a CBR field where the solution to a problem is stored as a plan. When a new problem enters the system, the case base is searched for a case whose plan can easily be adapted to the new problem. The found plan may partly be used while the remaining plan items are taken from another case or by using a generative planner. A survey of CBP and CBP systems can be found in e.g. [Munoz-Avila *et al.*, 1998] and [Spalazzi, 2001]. This planning technique can be very useful in CEDERIC for solving new, unseen composite problems. CEDERIC does already handle plans and it can switch between plans if the current one turns out to be wrong, but with the use of CBP the system can also combine several different plans into a composite one. It can also solve problems similar to those which have a solution by using the known solution plans and exchange the non suiting plan items to other plan items suggested by the generative planner. An example of the use of CBP is when CEDERIC knows how to solve the problem `Fly to the hospital` where the solution is to first look up the position of the hospital in the domain knowledge and then send a request to the robotic control system to fly to that position. We assume it also knows how to ask the robotic control system for a position of a car. If the operator gives the command `Fly to the parked car`, CEDERIC can create a plan which first asks the robotic control system for a position of the car and then use this position as input to the second part of the plan for flying to the hospital.

CBP can also be useful for CEDERIC to understand implicit information and to extend a plan when appropriate. One example is if the operator tells CEDERIC to fly somewhere but the UAV has not taken off yet. Then CEDERIC can understand the implicit command to take off and then perform the fly command. To make the planning more reliable, the dialogue system can report the plan to the operator and ask for confirmation before executing it.

Together with dialogue features already provided by CEDERIC, CBP can be used not only to perform planning by itself but also to implement mixed-initiative planning with user interaction. The system can ask the operator questions and the planning can be performed in cooperation with the operator. This will increase the usefulness of the system and the system will learn from it and can reuse the plans in the future. CBP can in particular be useful for developing the learning from explanation features described in section 2.6. further, where the system and the operator can plan a new learning sequence.

It has been argued that the CBP technique, where already stored plans are adapted to a new problem, does not add anything considering time complexity compared to generating new plans from scratch [Nebel and Koehler, 1995]. In our case however, low time complexity is not the most important reason for reusing old plans. When dealing with real world problems, several factors can affect the result and all factors may not be included in the problem formulation. Therefore, it is safer to use an already proven plan to a large extent than to generate a new one.

3 Implementation and Testing

The case base manager and the parser are implemented in lisp. The case base, domain knowledge and discourse model are implemented in The Knowledge Machine [Clark and Porter, 2004]. The planning with a generative planner and the use of several case plans to achieve a plan to a new target problem is still work in progress.

CEDERIC has been tested connected to a simulated UAV situated in a simulated environment. The simulated UAV can perform high level actions such as flying to a building with a certain unique identity, flying in a certain direction, take off, land, ascend and descend. It reports the results of an action and also reports if it observes any buildings as it flies. It is able to answer status questions such as the current altitude, velocity and heading, etc [Andersson, 2005]. The interface between CEDERIC and the simulator is very similar to the interface offered by the actual WITAS UAV, in order to make it straightforward to use CEDERIC during actual flights. The simulation of UAV movements and environment only goes to

Operator:	Fly to the school.
CEDERIC:	I have two schools to choose between. Which one do you mean?
Operator:	Which can I choose between?
CEDERIC:	You can choose between the one on Harborroad and the one on Mainstreet.
Operator:	The one on Harborroad.
CEDERIC:	Ok.
CEDERIC:	I am at the school now.

Figure 4: A dialogue example.

the level of detail that is required for this interface, and does not go down to the level of exact flight dynamics.

Due to the work in progress status of CEDERIC, it has not yet been rigorously tested in a formal manner. However, it has been tested with various types of dialogue, from simple robotic control commands such as `take off` and `land` to complex dialogue with sub-dialogues and learning from explanation. Figure 3 gives an example of an implemented and tested learning from explanation dialogue. In the tests, the first sentence has been provided in text because the speech recognizer does not yet recognize the new word, but the rest of the dialogue has been tested using a speech recognizer. The answers are all produced by a speech generator. Figure 4 gives an example of a dialogue with several sub-dialogues. It has been tested using speech in and speech out.

The sub-dialogue for finding a unique object is not tightly connected to a specific dialogue but can be used as a sub-plan in every dialogue where the system needs a unique reference identification to the object the operator refers to. Other sub-dialogues can easily be implemented in a similar fashion and used whenever it may be useful.

4 Related Work

Conversational CBR (CCBR) is an area within CBR where the user usually wishes to query a database with items, e.g. searching for a computer to buy on an e-commerce site. Aha et. al. provide an overview of CCBR in [Aha *et al.*, 2001]. The e-commerce system ExpertClerk is described in [Shimazu, 2002]. Case Advisor is a generic CCBR system which allows an organization to efficiently author and retrieve solutions from a knowledge database to solve customer problem. It is a commercial tool but some of its features is described in e.g. [Racine and Yang, 1997]. CCBR differs from our work in several essential ways. CEDERIC is capable of learning from experience and saves new cases in the case base for further use, which CCBR systems do not. It is also capable of learning from explanation whereas CCBR systems are not. Another big difference is that the sentences in CEDERIC can be saved in cases of their own and do not have to be coupled to a physical action, i.e. the questions are separated from the items in the case base.

HICUP [Aha *et al.*, 2001] is a CCBR system with integrated planning capabilities developed by Aha et al. HICUP is used to plan noncombatant evacuation operations (NEOs). HICUP uses published military doctrines as well as information from previously performed NEOs to guide the search. As in CCBR, the user gets simple questions to narrow the similar case set. HICUP does not save its own solutions and thus does not learn from experience more than using the information from previously performed NEOs which is manually fed into the system, in contrast to CEDERIC which saves the solutions for further use. HICUP also differs from CEDERIC in the use of language in the system. HICUP does not provide the user with a rich dialogue and does not learn from explanation.

Within the WITAS project, several dialogue systems with various capabilities have been developed. The first WITAS Dialogue System [Lemon *et al.*, 2001] was a system for multi-threaded robot dialogue using spoken I/O. The DOSAR-1 system [Sandewall *et al.*, 2003] was a new implementation using another architecture and a logical base. This system has been extended into the current OPAS system [Sandewall *et al.*, 2005]. Our work takes a rather different approach than their systems due to the use of CBR, the integration of learning capabilities using adaptation of the cases and learning from explanation. Our system also addresses the issue of planning and reuse of plans in dialogue and robotic actions.

Some work has been done concerning robotic dialogue and learning from explanation. Asoh et al. [Asoh *et al.*, 1997] have developed a robot called Jijo-2, which is able to create a map over its surroundings through conversation with a human teacher. The human teacher can give the robot a description of how the surroundings look like and how to solve wayfinding problems and the robot can ask the teacher questions. Theobalt et al. have created a robot similar to Jijo-2, which uses domain knowledge in the learning process [Theobalt *et al.*, 2002]. Carl [Lopes, 2002] is a robot which has some capabilities of learning from explanation. The operator can tell Carl facts that are stored in the memory of the robot. This information can be used later on in the dialogue. None of these robots use CBP for their dialogue and task planning as CEDERIC does and they can not reuse old plans unless told to learn them in advance as in learning from explanation.

5 Conclusions and Future Work

We propose a robotic dialogue system with learning capabilities which make it adaptable with respect to the operator and the problem domain. The system, CEDERIC, is built using CBR techniques and includes a *parser*, a *case base*, *domain knowledge*, a *discourse module* and a *case-base manager*. It is capable of having a dialogue including sub-dialogue such as clarifying questions and can handle references. The dialogue acts and the commands regarded the physical acts are treated in a uniform manner and the dialogue acts contributes to the solution of a problem as much as the physical acts do. The acts are saved in the case base and can be reused at any time. The case base also includes plan cases whose solution is a plan consisting of other sub-plans or atomic actions. The operator can adapt the system by serving as a teacher. The systems learns from explanation and when a new word is learnt, it can easily be used in different contexts. This is done by the flexible plan architecture where composite acts are modeled as plans. Several plans can match the initial problem definition and if it turns out that the selected plan does not match the actual course of events, the system switches to another more suiting plan.

CEDERIC is tested connected to a simulated UAV and it is able to send and receive messages both from the operator and the robotic control system and to perform a dialogue in natural language. More exhaustive tests will be performed when the system reaches a mature status.

CEDERIC is work in progress and an integrated planner module built on CBP techniques is currently developed. The planner module will provide CEDERIC with advanced learning functions and abilities to understand implicit knowledge that the operator does not provide. The planner performs mixed initiative case-based planning with user interaction where the system is able to reuse parts of plans from several cases in the case base to solve a new planning problem. Using the dialogue features, the operator can guide the search and teach the system new information during a planning action.

We expect CEDERIC to be a full fledged dialogue and planning system which can work in cooperation with the operator in a safe and secure way. When it is mature enough, we expect to test the system connected to the physical helicopter and to demonstrate the system in actual flight.

Acknowledgement

This research work was funded by the Knut and Alice Wallenberg Foundation, Sweden.

References

[Aamodt, 1994] Agnar Aamodt. Case-based reasoning; foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.

[Aha *et al.*, 2001] David W. Aha, Leonard A. Breslow, and Hector Munoz-Avila. Conversational case-based reasoning. *Applied Intelligence*, 14(1):9–32, 2001.

[Ahrenberg *et al.*, 1991] Lars Ahrenberg, Arne Jönsson, and Nils Dahlbäck. Discourse representation and discourse management for a natural language dialogue system. Technical report, Institutionen för Datavetenskap, Universitetet och Tekniska Högskolan Linköping, 1991.

[Andersson, 2005] Peter J. Andersson. Hazard: A framework towards connecting artificial intelligence and robotics. In *Proceedings of the Workshop on Representation, Reasoning and Learning in Computer Games*, 2005.

[Asoh *et al.*, 1997] Hideki Asoh, Satoru Hayamizu, Isao Hara, Yoichi Motomura, Shotaro Akaho, and Toshihiro Matsui. Socially embedded learning of the office-conversant mobil robot Jijo-2. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1997.

[Clark and Porter, 2004] Peter Clark and Bruce Porter. *KM - The Knowledge Machine 2.0: Users Manual*, 2004.

[Doherty *et al.*, 2000] Patrick Doherty, Gösta Granlund, Krzysztof Kuchinski, Erik Sandewall, Klas Nordberg, Erik Skarman, and Johan Wiklund. The witas unmanned aerial vehicle project. In *Proceedings of the 12th European Conference on Artificial Intelligence*, 2000.

[Eliasson, 2005] Karolina Eliasson. Integrating a discourse model with a learning case-based reasoning system. In *Proceedings of DIALOR-05*, 2005.

[Finin *et al.*, 1994] T. Finin, R. Fritzson, D. McKay, and R. McEntire. KQML as an Agent Communication Language. In N. Adam, B. Bhargava, and Y. Yesha, editors, *Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94)*, pages 456–463, Gaithersburg, MD, USA, 1994. ACM Press.

[Lemon *et al.*, 2001] Oliver Lemon, Anne Bracy, Alexander Gruenstein, and Stanley Peters. The WITAS multi-modal dialogue system. In *Proceedings of EuroSpeech*, 2001.

[Lopes, 2002] L. Seabra Lopes. Carl: from situated activity to language level interaction and learning. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, 2002.

[Munoz-Avila *et al.*, 1998] Hector Munoz-Avila, Ralph Bergmann Manuela Veluso, and Erica Melis. Case-based reaoning applied to planning tasks. *Case-Based Reasoning Technology from Foundations to Applications*, 1998.

[Nebel and Koehler, 1995] Bernhard Nebel and Jana Koehler. Plan reuse versus plan generation: a theoretical and empirical analysis. *Artif. Intell.*, 76(1-2):427–454, 1995.

[Pfleger *et al.*, 2003] Norbert Pfleger, Jan Alexandersson, and Tilman Becker. A robust and generic discourse model for multimodal dialogue. In *Workshop Notes of the IJCAI-03 Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, 2003.

[Racine and Yang, 1997] Kristi Racine and Qiang Yang. Maintaining unstructured case base. In *Proceddings of the Second International Conference on CBR*, 1997.

[Sandewall *et al.*, 2003] Erik Sandewall, Patrick Doherty, Oliver Lemon, and Stanley Peters. Words at the right time: Real-time dialogues with the witas unmanned aerial vehicle. In *Proceedings of the 26th Annual German Conference in AI*, 2003.

[Sandewall *et al.*, 2005] Erik Sandewall, Hannes Lindblom, and Björn Husberg. Integration of live video in a system for natural language dialog with a robot. In *Proceedings of DIALOR-05*, 2005.

[Shimazu, 2002] Hideo Shimazu. ExpertClerk: A conversational case-based reasoning tool for developing salesclerk agents in e-commerce webshops. *Artif. Intell. Rev.*, 18(3-4):223–244, 2002.

[Spalazzi, 2001] Luca Spalazzi. A survey on case-based planning. *Artificial Intelligence Review*, 16(1):3–36, 2001.

[Theobalt *et al.*, 2002] Christian Theobalt, Johan Bos, Tim Chapman, Arturo Espinosa-Romero, Mark Fraser, Gillian Hayes, Ewan Klein, Tetsushi Oka, and Richard Reeve. Talking to godot: Dialogue with a mobile robot. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and System*, 2002.