Instructing Novice Users on How to Use Tools in DIY Projects

Gregor Behnke[1], Marvin Schiller[1], Matthias Kraus[1], Pascal Bercher[1], Mario Schmautz[1], Michael Dorna[2], Wolfgang Minker[1], Birte Glimm[1], Susanne Biundo[1]
[1] Faculty of Engineering, Computer Science, and Psychology, Ulm University, Germany
[2] Corporate Research Sector, Robert Bosch GmbH, Renningen, Germany
gregor.behnke@uni-ulm.de, marvin.schiller@uni-ulm.de, matthias.kraus@uni-ulm.de, pascal.bercher@uni-ulm.de, mario.schmautz@uni-ulm.de, michael.dorna@de.bosch.com, wolfgang.minker@uni-ulm.de, birte.glimm@uni-ulm.de, susanne.biundo@uni-ulm.de

Abstract

Novice users require assistance when performing handicraft tasks. Adequate instruction ensures task completion and conveys knowledge and abilities required to perform the task. We present an assistant teaching novice users how to operate electronic tools, such as drills, saws, and sanders, in the context of Do-It-Yourself (DIY) home improvement projects. First, the actions that need to be performed for the project are determined by a planner. Second, a dialogue manager capable of natural language interaction presents these actions as instructions to the user. Third, questions on these actions and involved objects are answered by generating appropriate ontology-based explanations.

1 Introduction

Using electronic tools, like drills, saws, or sanders without proper instruction can be difficult and even dangerous for novice users. They may not know how to handle and configure these tools appropriately for the task at hand. This is problematic for people who want to use such tools for small constructions or household repairs – so-called Do-It-Yourself (DIY) projects. Their perceived inability and the apparent involved danger cause novice users to not even attempt these projects themselves. Assistance by appropriate instruction will enable users to perform DIY projects and thus can make them more self-reliant while allowing for joy and excitement during the project. In collaboration with Bosch Corporate Research, we have developed an assistant providing detailed and adaptive instruction for DIY projects. The assistant combines three core capabilities into a practically usable system: planning, ontological reasoning, and dialogue management. We describe the tasks performed by each component and how they are integrated with each other. As a running example, we use the project of building a wooden key rack from scratch.

2 Overall Architecture

Users start their interaction by selecting a project. The planner then generates a plan for the project based on the available tools and materials. The plan is translated into instructions by adding a textual description, an image, and a video of what to do. Finding the appropriate content is done via reasoning over an ontology of factual knowledge about the domain. The ontology is also used to answer factual questions on tools and materials involved in a presented action. In our running example, the user wants to construct a wooden key rack, see Fig. 1, from a single plank. Although looking simple, this construction is hard for novice users, as it involves using a saw and a drill both for drilling and screwing and requires at least 41 individual steps (under best conditions, i.e., all tools and materials are available). Further, there is a wide range of alternative ways to complete the project, e.g., by choosing to connect the boards via screws, nails, pegs, or glue.

Figure 1: A real key rack and its conceptual drawing.

3 Planning

In our assistant, a planner determines the presented instructions, which correspond to a plan – a sequence of actions that, if executed by the user, will complete the DIY project. The planning domain contains descriptions of a multitude of possible actions in a DIY environment like adding and removing batteries, adding/removing drill bits and saw blades, drilling holes, putting screws in them, etc., sufficient to solve a variety of requested projects. The domain also ensures that no unsafe actions are performed (e.g., changing sawing blades while a battery is inserted) thus protecting the user from harm. Planned instructions (instead of hard-coded ones) allow for adapting the assistance to any possible situation – in terms of available tools and materials for the DIY project. They also allow for plan repair, i.e., reacting dynamically to mistakes made by the user while performing tasks by providing altered instructions. Our assistant is based upon a previous one that we have developed for setting up a home theatre [Bercher *et al.*, 2014; 2015; 2017]. Both rely on a hierarchical planning approach [Erol *et al.*, 1996; Ghallab *et al.*, 2016]. Our novel assistant exploits the hierarchy to communicate instructions on different levels of abstraction. Also, most knowledge was still hard-coded in

the previous system, while we now utilise an ontology and its reasoning capabilities for knowledge representation and for conveying information to the user [Behnke *et al.*, 2015b; Schiller *et al.*, 2017]. We use a version of the totSAT planner [Behnke *et al.*, 2018a] that has been extended to handle partially-ordered domains [Behnke *et al.*, 2018b].

The planner determines a suitable abstraction with at most seven abstract tasks (cf. human short-term memory). In our key rack example, the abstract plan consists of four steps: sawing a plank into two boards, connecting the boards, attaching hooks to the back, and adding hooks to the tray. We use this abstract plan for two purposes. First, we show the progress that has already been made to provide immediate feedback and motivation. Second, abstract steps are presented first, which is more appropriate for experienced users since they can abstract away from details like how to use specific tools. If desired, they can be expanded to more detailed plans.

4 Knowledge Management

An instruction, solely based on an action sequence and its abstraction ("How is something done?") is not adequate, as novice users require further information about tools and materials ("How can I identify this drill?"). To provide coherent assistance, the knowledge used by the planner and that conveyed in explanations must be in sync [Behnke *et al.*, 2015b; 2015a]. We achieve this by separating procedural and factual knowledge. Procedural knowledge is stored in the planning model, while factual knowledge is stored in a separate ontology. When one of the system's components needs information, it is passed on in an appropriate format. For example, when a hole is drilled, the planner has to check whether the bit in the drill and the drill's settings (e.g., suitable rotation speed per type of wood) are appropriate. We consider allowed configurations not as procedural, but as factual knowledge. This allows to treat the current state uniformly: it is entirely stored in the ontology. These facts are formulated in OWL 2 using class assertions [Schiller *et al.*, 2017]. If the user requests instructions for a project, the ontology manager translates the facts into PDDL and provides them to the planner. For instance, the ontology manager provides the following facts, pertaining to one option of drilling in softwood:

(materialType config1 Softwood) (rotarySpeed config1 1800)
(drillBitType config1 MetalDrillingBit)

References to texts, images, and videos containing instructions for carrying out the individual actions are also stored in the ontology. Instantiated actions are characterised by their parameters, e.g., a particular drill selected for drilling. For any combination of parameters, the best fitting instructional material needs to be retrieved (e.g., an image showing the specific type of drill with a specific attachment, if possible). For this, we use classification in the ontology: A taxonomy represents classes of actions, which are characterised by properties such as "uses a drill of type A". For an instantiated action, an individual is created in the ontology together with properties representing the action's parameters (e.g., "drill-1 is used"). We use HermiT [Glimm *et al.*, 2014] to find the most specific class of actions of which the current action is an instance.

Figure 2: Instructions for inserting a metal drill bit, the green box contains the description of a metal drill bit requested by the user.

5 Dialogue Management

Our dialogue manager (DM) handles the communication between a user on the one hand and planner and ontology modules on the other. At start-up, the DM asks the user which DIY project he wants to undertake and requests a suitable plan from the planner. The plan is presented as a sequence of task slides enriched with information retrieved by the ontology (textual task description, image, video) as depicted in Fig. 2 (the user interface is in German). From now on, we follow a user-driven interaction strategy. For semantic analysis of user input, we use Microsoft's LUIS [Williams *et al.*, 2015] and use separate language understanding models for analysing planning-, ontology-, and dialogue-related requests. Depending on which model returns the semantic representation with highest confidence, the DM forwards a transformed representation to the respective component in a rule-based manner [Kraus *et al.*, 2018]. On user request, the system provides further information and explanations on used objects like materials and tools. Such requests are handled by the ontology manager, which retrieves facts (object's uses, specifications and appearance) related to the object from the knowledge base and verbalises them. For example, in Fig. 2 the user is instructed to use a 3 mm metal drill bit for predrilling a hole for a 4 mm screw, as determined by the planner using the available background knowledge from the ontology.

6 Conclusion & Future Work

Our system shows how users of different skill levels can be introduced into a complex technical domain through assistance based on the interplay between planning, reasoning, and dialogue. A planned demonstrator will allow a user to request changes to the generated plan, handled via Linear Temporal Logic [Pnueli, 1977] goals. User tests with the system are currently underway as future work.

Acknowledgments

This work is done within the technology transfer project "Do it yourself, but not alone: Companion-Technology for DIY support" funded by the German Research Foundation (DFG). The industrial project partner is the Corporate Research Sector of the Robert Bosch GmbH.

References

[Behnke *et al.*, 2015a] Gregor Behnke, Pascal Bercher, Susanne Biundo, Birte Glimm, Denis Ponomaryov, and Marvin Schiller. Integrating ontologies and planning for cognitive systems. In *Proceedings of the 28th International Workshop on Description Logics (DL 2015)*. CEUR Workshop Proceedings, 2015.

[Behnke *et al.*, 2015b] Gregor Behnke, Denis Ponomaryov, Marvin Schiller, Pascal Bercher, Florian Nothdurft, Birte Glimm, and Susanne Biundo. Coherence across components in cognitive systems – one ontology to rule them all. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, pages 1442–1449. AAAI Press, 2015.

[Behnke *et al.*, 2018a] Gregor Behnke, Daniel Höller, and Susanne Biundo. totSAT – Totally-ordered hierarchical planning through SAT. In *Proceedings of the 32th AAAI Conference on AI (AAAI 2018)*, pages 6110–6118. AAAI Press, 2018.

[Behnke *et al.*, 2018b] Gregor Behnke, Daniel Höller, and Susanne Biundo. Tracking branches in trees – A propositional encoding for solving partially-ordered HTN planning problems. In *Proceedings of the First ICAPS Workshop on Hierarchical Planning*, 2018.

[Bercher *et al.*, 2014] Pascal Bercher, Susanne Biundo, Thomas Geier, Thilo Hörnle, Florian Nothdurft, Felix Richter, and Bernd Schattenberg. Plan, repair, execute, explain - how planning helps to assemble your home theater. In *Proceedings of the 24th International Conference on Automated Planning and Scheduling (ICAPS 2014)*, pages 386–394. AAAI Press, 2014.

[Bercher *et al.*, 2015] Pascal Bercher, Felix Richter, Thilo Hörnle, Thomas Geier, Daniel Höller, Gregor Behnke, Florian Nothdurft, Frank Honold, Wolfgang Minker, Michael Weber, and Susanne Biundo. A planning-based assistance system for setting up a home theater. In *Proceedings of the 29th AAAI Conference on AI (AAAI 2015)*, pages 4264–4265. AAAI Press, 2015.

[Bercher *et al.*, 2017] Pascal Bercher, Felix Richter, Thilo Hörnle, Thomas Geier, Daniel Höller, Gregor Behnke, Florian Nielsen, Frank Honold, Felix Schüssel, Stephan Reuter, Wolfgang Minker, Michael Weber, Klaus Dietmayer, and Susanne Biundo. *Advanced User Assistance for Setting Up a Home Theater*, chapter 24, pages 485–491. Cognitive Technologies. Springer, 2017.

[Erol *et al.*, 1996] Kutluhan Erol, James Hendler, and Dana Nau. Complexity results for HTN planning. *Annals of Mathematics and AI*, 18(1):69–93, 1996.

[Ghallab *et al.*, 2016] Malik Ghallab, Dana S. Nau, and Paolo Traverso. *Automated Planning and Acting*. Cambridge University Press, 2016.

[Glimm *et al.*, 2014] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: An OWL 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[Kraus *et al.*, 2018] Matthias Kraus, Gregor Behnke, Pascal Bercher, Marvin Schiller, Susanne Biundo, Birte Glimm, and Wolfgang Minker. A multimodal dialogue framework for cloud-based companion systems. In *Proceedings of the 10th International Workshop on Spoken Dialog Systems Technology (IWSDS 2018)*, 2018.

[Pnueli, 1977] Amir Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (SFCS 1977)*, pages 46–57. IEEE, 1977.

[Schiller *et al.*, 2017] Marvin Schiller, Gregor Behnke, Mario Schmautz, Pascal Bercher, Matthias Kraus, Michael Dorna, Wolfgang Minker, Birte Glimm, and Susanne Biundo. A paradigm for coupling procedural and conceptual knowledge in companion systems. In *Proceedings of the 2nd International Conference on Companion Technology (ICCT 2017)*. IEEE, 2017.

[Williams *et al.*, 2015] Jason Williams, Eslam Kamal, Mokhtar Ashour, Hani Amr, Jessica Miller, and Geoff Zweig. Fast and easy language understanding for dialog systems with Microsoft Language Understanding Intelligent service (LUIS). In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL 2015)*, pages 159–161. Association for Computational Linguistics, 2015.