FAIM/ISCA Workshop on Artificial Intelligence for Multimodal Human Robot Interaction
14-15 July 2018, Stockholm, Sweden

A Knowledge Based Approach to User Support for Robot Programming

Elin A. Topp and Jacek Malec
Department of Computer Science, Lund University, Sweden
{Elin_Anna.Topp, Jacek.Malec}@cs.lth.se

Abstract

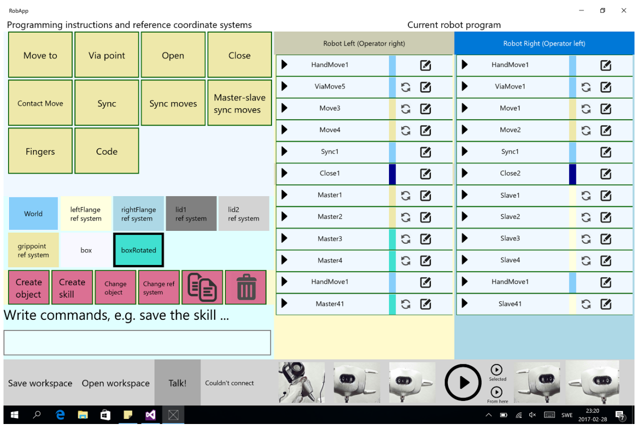
We summarize our successful efforts to support intuitive programming of industrial robotic assembly tasks with a knowledge based approach to the representation of skills. These skills can be specified, re-used, refined and transferred between robots with the help of a multimodal interface combined with kinesthetic teaching. We argue that while it is certainly possible and suitable to have robots acquire skills or skill primitives through various learning methods, it is still crucial to provide explicit knowledge and semantics available to them.

1 Introduction

Over the recent years a new type of inherently safe industrial manipulators intended for direct collaboration with human shop floor workers has entered the market, examples of which are Universal Robots' UR5, ABB's YuMi, Rethink Robotics' Baxter and KUKA's LBR iiwa. These, as well as some traditional manipulators can be set up in a dual-arm configuration (in the case of ABB's YuMi or Rethink Robotics' Baxter this is the default solution) to handle bi-manual tasks, e.g., in product assembly.

A common property of these robot models is the *lead–through* programming, or kinesthetic teaching, mode which enables the user to physically guide the robot manipulator into desired positions instead of using e.g., a joystick [Rodamilans *et al.*, 2016, for example] or to apply approaches to learning from demonstration, [Billard *et al.*, 2016, as an overview]. Still, the robot program has to be created as a sequence of motions and control logic, e.g., by adding instructions one by one using a *teach pendant*, a textual program editor installed on a PC or simplified programming interfaces such as the YuMi Online Windows app.

An approach to making the programming of complex robot instructions more available to non-expert users is the use of high–level instructions to simplify robot programming for industrial tasks, e.g., using natural language [Stenmark and Nugues, 2013] and semantically defined parameterized motion primitives [Felip *et al.*, 2013], often called *skills* [Pedersen *et al.*, 2016, for example]. Such semantic descriptions are desirable because they enable the automatic generation



Current robot program

Robot Left (Operator right)	Robot Right (Operator left)
HandMove1	HandMove1
ViaMove5	ViaMove1
Move3	Move1
Move4	Move2
Sync1	Sync1
Close1	Close2
Master1	Slave1
Master2	Slave2
Master3	Slave3
Master4	Slave4
HandMove1	HandMove1
Master41	Slave41

Figure 1: Our graphical interface

of standard PDDL[1]–descriptions for planning and scheduling of tasks or path planning and generation from virtual models [Perzylo *et al.*, 2016]. However, in most cases these abstractions assume a library of skills or motion primitives that can be represented in executable robot controller code sequences, which can be applied in the high-level task description. Unfortunately, this assumption does not necessarily hold for highly specialized motions and configurations that might be needed in a specific assembly task.

Given the vast amount of recent approaches to semi- or even unsupervised learning of skills or motion primitives [Levine *et al.*, 2016, for example], an obvious way to go seems to have the robotic systems learn even such skill primitives that can later be re-used in new contexts. However, from our experience with complex manipulation and assembly tasks [Stolt *et al.*, 2011; Stenmark *et al.*, 2016], and also based on our observations in a user study [Stenmark *et al.*, 2017a] we identified a number of situations, where a more explicit instruction and explanation of goals and crucial parts of the task execution is desirable, and where users should be supported by the system in making their goals and requirements explicit for the system. Also, we are convinced that for the sake of re-use and refinement of previously acquired skills a grounding in human-comprehensible concepts and descrip-

[1]Planning Domain Definition Language

tions is a crucial property of these skills. In other words, a robot that has learned how to pick up a screw through unsupervised learning needs to understand that what it just did was a "pick-up" operation performed on an object called "screw". Hence, we consider learning of skills from demonstration or through imitation as one way of acquiring certain procedural knowledge relevant to task execution, while still arguing that there needs to be a solid underlying representation for skills, tasks, and objects that can then incorporate such knowledge and combine it with more explicitly taught (programmed or up-front designed) aspects.

In the following, we summarize our recent and ongoing efforts to establish a middle ground between high-level instructing by demonstration (partially based on learning) and explicit programming by combining kinesthetic teaching and a multimodal user interface with parametrized skill representations to simplify robot programming, which allows also non-expert users to specify and semantically annotate their own skills (motion primitives and also more complex program sequences). We use this summary to make a case for this holistic view on providing a complete end-to-end system for the specification, re-use, refinement and correction of robot programs.

2 Skill Based Robot Programming

The knowledge in our system is centered around physical *objects* (e.g., workpieces), *devices* such as manipulators, grippers and sensors, and *robot skills* used to realize assembly processes, which are implemented as robot programs. Parametrized skills form a hierarchy, with *primitives* in its bottom, which is described in a number of skill ontologies in our knowledge base. To specify and parameterize (re-use) skills, several ways of accessing the knowledge base are possible, of which the most recent is the graphical user interface we will refer to in the following, after giving an overview of the ontological representation.

2.1 Robot-agnostic skill ontology in the knowledge base

The skill ontologies available in the knowledge base of the system have grown since at least a decade [Björkelund *et al.*, 2011; Haage *et al.*, 2011; Malec *et al.*, 2013; Stenmark and Malec, 2015; Stenmark *et al.*, 2014]. They are organized in a modular structure introduced in [Jacobsson, 2015], see Figure 2.

The rosetta ontology contains the basic definitions of skills and devices. Configuration ontology describes the necessary parameterization of skills, including their hardware demands (i.e., which devices can possess this skill), pre- and postconditions (useful for planning and consistency checking), frame references (for introducing all necessary coordinate frames a skill may refer to) and actual parameters (like weights, positions, colours, etc.). Coordination in turn relates the behaviour implementing the skill (expressed, e.g., as a finite state machine, or as a piece of native robot code) to the environment, synchronizing it with the incoming events. A schematic structure of a skill is presented in Figure 3.

The device library defines the set of instances for the main rosetta.owl skill ontology. It contains a number of robots

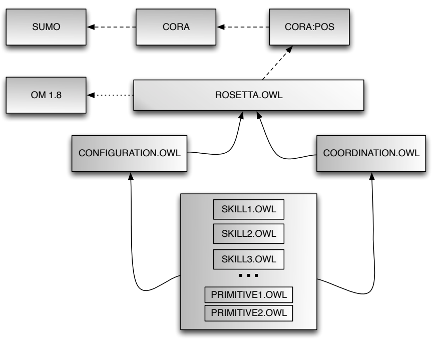


Figure 2: Skill ontologies available in the KIF knowledge base. Dashed arrows denote import of external ontologies.

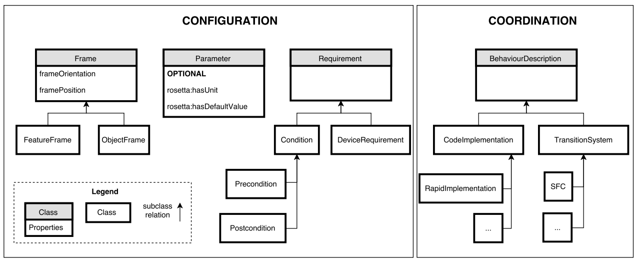


Figure 3: Skill structure.

(including ABB IRB 2400 and ABB YuMi used in experiments described in this paper), their effectors, sensors and tool changers, enabling generic skill definition and transfer between different robot installations.

2.2 Interface for Iconic Programming

We have recently proposed a GUI prototype that allows also non-expert users to specify their own skills by combining kinesthetic teaching with iconic progamming. These skills can then be re-used and adapted on a higher level of abstraction into more complex task descriptions [Stenmark *et al.*, 2017a; Stenmark *et al.*, 2018; Stenmark *et al.*, 2017b], also allowing for certain operations to be specified relative to (adaptable) object reference frames, and more recently also supporting synchronized motions for dual-arm operations [Stenmark *et al.*, 2017c]. In our experimental setup, objects are created as abstract entities with a unique name, a type, and positions (called *object frames*) describing places on the object that may be used later as origins of relative reference coordinate systems. Robot motions can then be specified using those reference coordinate frames and the actual positions can be updated, e.g., using cameras like the YuMi wrist camera or our recently developed hint system for RGB-D data based identification and localization of objects in the robot's workspace [Ganslandt and Svensson, 2018]. Every robot manipulator has a specific reference frame attached to the wrist, the *flange*. The gripper is attached to the flange and may have in turn one or more reference points, e.g., the tips of the fingers. This means, that when using a dual-arm robot, motions of one arm may be also specified in the other arm's flange reference system. The low-level implementation in this setup does not allow arbitrary reference points along the robot arms, thus, contact situations where the so-called elbow or lower arm are used need to be specified using joint angles.

The re-usable robot programs, i.e., robot skills, have two components. The first is a high-level step-by-step instruction of how to achieve a goal. The skills can be nested into hierarchical (compound) skills and the lowest semantically described step comprises atomic actions called *primitives*. The primitives must have a mapping to some executable code on the robot system, which is the second component of the skill. Our current implementation generates ABB RAPID, which is the native code executing on an ABB robot controller. As this executable code associated with specific motion primitives in the interface and in the skill description can be made available for different robot systems, it is possible to transfer a robot program from one robot system to another, as long as a mapping of motion primitives to executable code is possible. Such a robot skill for, e.g., a pick-and-place operation for a wooden block would consist simply of a sequence of moves to specific positions (the positions being the parameters of the motion primitive) that can be annotated by the user with arbitrary names (start pick, open gripper, approach, close gripper, lift, start place, approach, open gripper, retract) and then combined into named skills (e.g., *pick-block-skill*, comprising the first five operations and *place-skill*, comprising the later four ones). These annotations allow later both robot system and user to refer to complete skills or specific parts of those by name, so that high-level instruction becomes possible (e.g., by typing or even saying "Pick up the block"), as the instructions can (in this example) be directly related to a skill that has suitable name tags in its annotations. In cases several skills are suitable, these are (for the moment) simply listed to the user, but we plan to incorporate more underlying reasoning to better support users in their choice.

2.3 User Support for Specific Tasks

Our interface, shown in Figure 1 and the underlying representations for skills and objects support the user already in parameterizing, i.e., updating and adapting, previously programmed skills [Stenmark *et al.*, 2017a], as well as annotating skills with explicit (device) requirements to allow for later re-use also through other users than the original programmer [Stenmark *et al.*, 2018]. The interface allows for graphical (iconic) programming as well as typed input (natural language) and it can also be connected to a complete language processing pipeline, so that spoken input can be processed [Stenmark and Nugues, 2013], thus supporting multimodal input channels. Also, the specification of synchronized operations, i.e., dual-arm robot programs, is possible, even their modification is supported [Stenmark *et al.*, 2017c]. As of now, however, the user is required to explicitly specify new reference coordinates in case a dual-arm operation is to be transformed, e.g., in a rotating or mirroring transformation of a task as illustrated in Figure 4. In a skill re-use scenario, the user is supported regarding the explicit requirements of the skill, e.g., if a skill has object references to an object "block", the user is warned about this fact if the current workspace of the interface does not contain a respective object when the skill is imported from the knowledge base. Other, more complex lines of reasoning regarding pre- and post conditions of skills made explicitly in the interface and regarding geometric relationships are subject to current work.

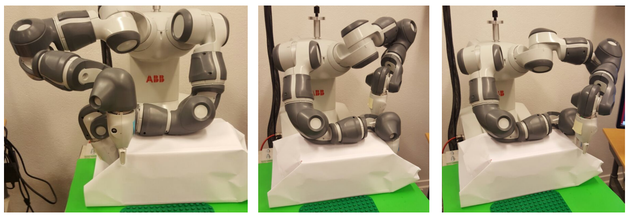

(a) YuMi packing a parcel (b) Rotated (c) Mirrored

Figure 4: Original, rotated and mirrored positions

3 Discussion

From experiences both in experimental evaluation with experts and from a user study with non-expert users, we allow us to assume that our middle-ground approach to iconic programming is suitable for the instruction of industrial robots, specifically regarding emerging tasks in assembly processes. We are aware that (unsupervised) machine learning approaches can be exploited to provide robotic systems with a "vocabulary" of executable action sequences, but argue that these action sequences need to be grounded in their respective context and in user-comprehensible semantics if they are to be (re-)used in new contexts or task descriptions than those they were originally designed for. We also assume, that it is of great value for the user to be able to access the exact granularity of descriptions, annotations and requirements for a certain skill or task description that is required for the task and / or desired by the user. E.g., in the GiftWrapping scenario illustrated in Figure 4a it is important to be able to make the skill description robot arm dependent, as the elbow of the arm is actually used to fixate the box while folding the paper on the side. While it is certainly helpful for the user to abstract away all details of the motion description, it is still important to allow the user to annotate this particular skill with the requirement of a 7DOF-arm, so that this information can be made explicit for other users. We see a challenge from the HRI perspective imposed onto the AI-community to provide knowledge (skill) representations allowing for this type of flexibility in granularity. At the same time, the challenge for the HRI community is to provide (further) actual and realistic scenarios and issues to incorporate into representations, reasoning as well as learning mechanisms and systems, to ultimately be able to provide end-to-end systems that can be instructed, maintained and simply used in collaborative settings on the shop-floor, not only in large plants with a robotic expert just around the corner, but also in a small company where the user can be seen as end-user, operator and programmer all at once.

Acknowledgment

The research leading to these results has received funding from the European Community's Framework Programme Horizon 2020 under grant agreement No 644938 SARAFun.

References

[Billard *et al.*, 2016] Aude Billard, Sylvain Calinon, and Ruediger Dillmann. Learning from humans. In *Springer Handbook of Robotics, Chapter 74*, pages 1995–2014. Springer, 2016.

[Björkelund *et al.*, 2011] A. Björkelund, L. Edström, M. Haage, J. Malec, K. Nilsson, P. Nugues, S. Gestegård Robertz, D. Störkle, A. Blomdell, R. Johansson, M. Linderoth, A. Nilsson, A. Robertsson, A. Stolt, and H. Bruyninckx. On the integration of skilled robot motions for productivity in manufacturing. In *Proc. IEEE International Symposium on Assembly and Manufacturing*, Tampere, Finland, 2011.

[Felip *et al.*, 2013] J. Felip, J. Laaksonen, A. Morales, and V. Kyrki. Manipulation primitives: A paradigm for abstraction and execution of grasping and manipulation tasks. *Robot. Auton. Syst.*, 61(3):283–296, March 2013.

[Ganslandt and Svensson, 2018] Alexander Ganslandt and Andreas Svensson. 6DOF Object Recognition and Positioning for Robotics using Next Best View Heuristics, 2018. MSc Thesis, Dept of Computer Science, Lund University, http://fileadmin.cs.lth.se/ai/psfiles/GranslandtSvensson.pdf.

[Haage *et al.*, 2011] M. Haage, J. Malec, A. Nilsson, K. Nilsson, and S. Nowaczyk. Declarative-knowledge-based reconfiguration of automation systems using a blackboard architecture. In A. Kofod-Petersen, F. Heintz, and H. Langseth, editors, *Proc. 11th Scandinavian Conference on Artificial Intelligence*, pages 163–172. IOS Press, 2011.

[Jacobsson, 2015] Ludwig Jacobsson. A Module-Based Skill Ontology for Industrial Robots, 2015. MSc Thesis, Dept of Computer Science, Lund University.

[Levine *et al.*, 2016] Sergey Levine, Chelsea Finn, Trevor Darrell, and Peter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17:1–40, 2016.

[Malec *et al.*, 2013] Jacek Malec, Klas Nilsson, and Herman Bruyninckx. Describing assembly tasks in declarative way. In *Proc. IEEE ICRA 2013 Workshop on Semantics, Identification and Control of Robot-Human-Environment Interaction*, 2013.

[Pedersen *et al.*, 2016] Mikkel Rath Pedersen, Lazaros Nalpantidis, Rasmus Skovgaard Andersen, Casper Schou, Simon Bøgh, Volker Krüger, and Ole Madsen. Robot skills for manufacturing: From concept to industrial deployment. *Robotics and Computer-Integrated Manufacturing*, 37:282 – 291, 2016.

[Perzylo *et al.*, 2016] Alexander Perzylo, Nikhil Somani, Stefan Profanter, Ingmar Kessler, Markus Rickert, and Alois Knoll. Intuitive instruction of industrial robots: Semantic process descriptions for small lot production. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Republic of Korea, October 2016.

[Rodamilans *et al.*, 2016] Guilherme Boulhosa Rodamilans, Emília Villani, Luís Gonzaga Trabasso, Wesley Rodrigues de Oliveira, and Ricardo Suterio. A comparison of industrial robots interface: force guidance system and teach pendant operation. *Industrial Robot: An International Journal*, 43(5):552–562, 2016.

[Stenmark and Malec, 2015] Maj Stenmark and Jacek Malec. Knowledge-based instruction of manipulation tasks for industrial robotics. *Robotics and Computer-Integrated Manufacturing*, 33:56 – 67, 2015.

[Stenmark and Nugues, 2013] Maj Stenmark and Pierre Nugues. Natural language programming of industrial robots. In *Robotics (ISR), 2013 44th International Symposium on*, pages 1–5. IEEE, 2013.

[Stenmark *et al.*, 2014] Maj Stenmark, Jacek Malec, and Andreas Stolt. From high-level task descriptions to executable robot code. In *Intelligent Systems' 2014*, pages 189–202. Springer, 2014.

[Stenmark *et al.*, 2016] Maj Stenmark, Andreas Stolt, Elin A. Topp, Mathias Haage, Anders Robertsson, Klas Nilsson, and Rolf Johansson. The GiftWrapper: Programming a Dual-Arm Robot With Lead-through. In *ICRA Workshop on Human-Robot Interfaces for Enhanced Physical Interactions*, 2016.

[Stenmark *et al.*, 2017a] Maj Stenmark, Mathias Haage, and Elin A. Topp. Simplified Programming of Re-usable Skills on a Safe Industrial Robot – Prototype and Evaluation. In *Proceedings of the IEEE/ACM Conference on Human-Robot Interaction (HRI)*, 2017.

[Stenmark *et al.*, 2017b] Maj Stenmark, Mathias Haage, Elin A. Topp, and Jacek Malec. Making robotic sense of incomplete human instructions in high-level programming for industrial robotic assembly. In *In Proceedings of the AAAI-17 Workshop on Human Machine Collaborative Learning*, 2017.

[Stenmark *et al.*, 2017c] Maj Stenmark, Elin A. Topp, Mathias Haage, and Jacek Malec. Knowledge for synchronized dual-arm robot programming. In *In AAAI Fall Symposium Series, Technical Report, AAAI Digital Publications*, 2017.

[Stenmark *et al.*, 2018] Maj Stenmark, Mathias Haage, Elin A. Topp, and Jacek Malec. Supporting Semantic Capture during Kinesthetic Teaching of Collaborative Industrial Robots. *Int'l Journal of Semantic Computing*, 12(1), 2018.

[Stolt *et al.*, 2011] Andreas Stolt, Magnus Linderoth, Anders Robertsson, and Rolf Johansson. Force controlled assembly of emergency stop button. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3751–3756. IEEE, 2011.