Toward A Knowledge Transfer Framework for Process Abstraction in Manufacturing Robotics

Jacob Huckaby HUCKABY@GATECH.EDU
Center for Robotics and Intelligent Machines, Georgia Institute of Technology, 801 Atlantic Dr, Atlanta, GA 30332 USA

Henrik I. Christensen HIC@CC.GATECH.EDU
Center for Robotics and Intelligent Machines, Georgia Institute of Technology, 801 Atlantic Dr, Atlanta, GA 30332 USA

Abstract

Robust methods for representing, generalizing, and sharing knowledge across different robotic systems and configurations are important in many domains of robotics research and application. In this paper we present a framework for capturing robot capability and process specification to simplify the sharing and reuse of knowledge between robots in manufacturing environments. A SysML model is developed that represents knowledge about system capabilities in the form of simple skills and skill primitives that can be used in different situations or contexts. We present a discussion of the form this model takes and advantages of this type of representation, as well as a demonstration of how the model can be applied to different assembly tasks.

1. Introduction

The popularity of robotics today is growing across all industries. Even in manufacturing, where robots have traditionally been commonplace, new research has aimed at investigating how to insert robotics even more into the manufacturing and assembly process. However, this is still a process that is fraught with difficulty, and there are a number of issues that can arise. One problem that continues to plague the adoption of robotics at all levels of industry is one simple fact: programming robots is hard. It requires deep technical knowledge, it is time consuming, and programming robots is done by specification with respect to the system requirements. Other issues have to do with establishing the system in the first place. When a factory is setting up a robotic or automation system to run in a manufacturing environment, there is normally a lot of care that goes into designing the setup. The system must be well-characterized, the environment must be known and structured and well-defined, and a lot of programming must be done to get the robot to perform its task, usually by a trained robot technician; and all efforts are made to keep the robot in a certain operating state and the environment in its predetermined, structured state, so that the system is calibrated such that it can continue to complete the task. However, there are often changes that need to be made to the system. Robots may need to be upgraded to newer models, or something may change in the manufacturing process or the tooling, or the robot may be required to perform a new task. When something changes like this, a lot of work has to be duplicated to realize this new system. The old system needs to be reconfigured and recalibrated, the environmental structure redefined, and the trained robot technician will need to do all of the task-specific programming again.

The question that must then be asked in this case is if there is a way to take this knowledge that already exists in the system, the knowledge relevant to the task, the environmental constraints, and the robot's own capabilities, and try to capture this knowledge and represent it in a way that makes this process more efficient. How can we not lose all of that knowledge that has been put into the system, and preserve at least some of it to be shared and used at a later time, possibly in another system or context?

For this to happen there needs to be a method for

Proceedings of the 30^{th} International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

modeling the tasks necessary to enable the robot to achieve its programming objective, along with the requirements, constraints, and skills needed to accomplish those tasks. To that end, this paper proposes a taxonomy-based knowledge transfer framework that can capture the knowledge a robot has about tasks and environmental constraints, as well as its own capabilities and configuration. This knowledge is represented in the form of a skill taxonomy that describes the hierarchy of skills that the robot possesses, as well as the requirements and constraints on those skills. The model will be utilized to generalize skill use across manufacturing processes, as well as the creation of a library or collection of high level, commonly used skills. This library of skills will be used to demonstrate that the framework can generalize enough to show abstraction of processes, as shown in two different assembly tasks.

This paper will proceed as follows. We will first address previous work done in the area of knowledge reuse in general robotic systems, including work done in programming by demonstration, industrial manufacturing, and knowledge representation in domestic and service robotics, in section 2. Section 3 will then discuss the proposed assembly taxonomy, along with the different components that compose the taxonomy and the justification for their inclusion. A motivating demonstration involving different assembly tasks will be presented and discussed in section 4, and a brief discussion of the conclusions and future work will follow in section 5.

2. Related Work

The idea of knowledge transfer, or sharing knowledge across systems, is not a new concept. There has been a lot of work over the years that has investigated the most efficient way to accomplish this, though most of this work has focused solely on only one knowledge transfer modality. This section will highlight some important areas of knowledge transfer.

Key to the idea of knowledge transfer is the concept of knowledge representation. There have been many different types of knowledge representations that have been proposed to try and capture the knowledge in a robot system, and to best represent that knowledge with respect to certain requirements and constraints on the system. In the Programming by Demonstration paradigm, this is accomplished by modeling knowledge about the motion and task. One approach encodes task knowledge as a function of motion. Examples of this type of representation include dynamical systems (Ijspeert et al., 2001) and object-action complexes (Krüger et al., 2011). Lyons, et. al. (Lyons & Arbib, 1989) defines a model for robot computation using port automata. Kosecka, et. al. (Kosecka & Bajcsy, 1993) used a discrete event systems framework to model tasks and behaviors. Other work includes that of Dantam, which takes a grammar-based approach to represent sensorimotor information (Dantam & Stilman, 2011).

Another approach represents important task and system knowledge symbolically, such as using skill trees (Konidaris et al., 2011) or topological task graphs (Abbas & MacDonald, 2011). This symbolic approach to knowledge representation assumes that the system has more inherent knowledge (it knows how the relationships between the symbols and physical instantiations behave), while it allows for the modeling of more high-level concepts than motion-based representations. Work by Kress-Gazit, et. al., (Kress-Gazit et al., 2011), (Finucane et al., 2010) uses linear temporal logic to model task specifications to produce correct robot controllers for different tasks. Some researchers have used this symbolic approach to address the issue of knowledge reuse or knowledge transfer in areas outside of manufacturing, including (Ekvall et al., 2006), (Ekvall & Kragic, 2006), (Nicolescu & Mataric, 2003).

There is also work that has been done in representing specifically manufacturing and assembly objectives, such as in the application of Petri Nets (Rosell, 2004). Another approach is the work of de Mello and Sanderson (Homem de Mello & Sanderson, Apr), which uses AND/OR graphs to enumerate all possible paths through the assembly process to get to the overall objective (e.g. an assembled product.) The paper then proposes to use a graph search algorithm to find an appropriate path through the graph based on specific problem specifications.

Ontologies have been used in many different frameworks as a representational tool for sharing knowledge (primarily semantic or relational knowledge) between systems. Systems like ConceptNet (Liu & Singh, 2004) (which utilizes the MIT Open Mind Common Sense database (Singh et al., 2002)) and CYC (Lenat, 1995) attempt to model human commonsense knowledge as a relational ontology, with a small, predefined set of semantic relationships determining how concepts are tied together. KNOWROB (Tenorth & Beetz, 2009) is a robot-specific framework that bases its knowledge representation on an ontological implementation, here using OWL2 (Hitzler et al., 27 October 2009) as the ontology standard. The ontology is defined around concepts and relations important for robots operating in the real world.

Recent work has also been done by Balakirsky, et. al. (Balakirsky et al., 2012) to define a knowledge representation for industrial robotics. This representation is similarly based on the OWL ontology standard. In this work the authors attempt to provide a method of structuring knowledge in such as way as to be able to reuse it for different problems. This work also proposes a multi-layered knowledge representation. The work in this paper likewise proposes a multi-layered representation, but at a different level of abstraction. It also differs fundamentally from the work of Balakirsky in that this representation is proposed to improve not only modularity in knowledge, but also usability and intuitiveness for users.

The last research area presented in this section is that of actual systems or frameworks that are specifically designed for knowledge transfer. Given a specific knowledge representation, a number of other projects have addressed the problem of representing, storing, and transferring knowledge between various robotic systems.

In the SIARAS (Skill-based Inspection and Assembly of Reconfigurable Automation Systems) project (SIARAS, 2011), a system is developed to assist in the automatic reconfiguration of automation systems. This is done due to the need for light-weight (low overhead) processes to address current manufacturing demands. System components are designed both to represent skills and parameters, as well as the process flow. This is done using a specific ontology. A skill server is designed to aid a human operator to match process requirements with the representations in the database.

Another project working on robot deployment in manufacturing environments in the ROSETTA (RObot control for Skilled ExecuTion of Tasks in natural interaction with humans; based on Autonomy, cumulative knowledge and learning) project (ROSETTA, 2009). This project tries to design industrial robotic systems that are suitable for working around and collaborating with humans in the manufacturing process. One important aspect of their approach is a skill repository.

RoboEarth (Waibel et al., 2011), (Zweigle et al., 2009) is a project aimed at creating a global repository for all knowledge relevant to a robotic systems, including information on environments, object models, action recipes, and semantic information. The architecture is organized in three layers, with the top layer being the global database, which acts as an information server, a second layer containing hardware independent functions such as action recognition and semantic mapping, and a bottom layer consisting of robot-specific implementations. Knowledge representation and processing is handled by the KnowRob system. (Tenorth & Beetz, 2009)

3. System

This section will discuss the knowledge transfer framework as it has been applied to the aerospace robot application domain, or airplane assembly and manufacturing, as well as some important features of the assembly taxonomy, and some overall strengths of using the taxonomy. A demonstration of an implementation of this will follow.

3.1. Modeling Language

We define a taxonomy for airplane assembly actions (see figure 1) which is able to accurately capture all of the capabilities a robot would need to perform the highly complex task of building an airplane.

In this work, the taxonomy takes the form of a SysML (OMG, 2010) model that specifies each of these capabilities. SysML (or Systems Model Language) is a general modeling language for systems and systems engineering, and is defined as an extension of the popular UML modeling language. One clear advantage of using an expressive modeling language like SysML is the ability to leverage tools associated with formal modeling languages. Some of these tools include validation and code generation. Validation, or model checking, can be used to check that the model is well-behaved. This tool can be used to verify that all of the task requirements are met and that system constraints are satisfied long before it is run on hardware, which greatly reduces the overhead associated with testing these potentially large, complex systems. Automatic code generation can be used on the model to significantly simplify the task of generating and maintaining the software implementation of the model. When changes are made to the model, this tool can be used to propagate those changes throughout the system, making it easier to avoid potential conflicts between model and implementation.

3.2. Abstraction & Hierarchy

Using a taxonomy to model complex tasks is a natural choice, as a taxonomy is able to capture the hierarchical nature of task decomposition (figure 2).

Consider how a robot interacts with its environment. Robots interact with the world by working toward some objective, such as constructing an airplane wing. These objectives can be decomposed into a sequence of tasks that must be accomplished to successfully

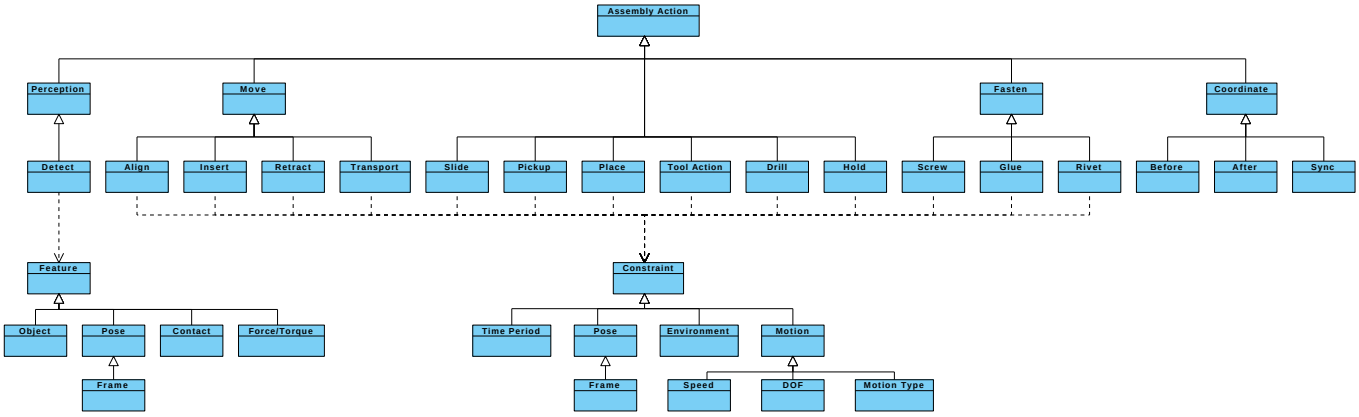

Figure 1. Taxonomy for manufacturing assembly task.

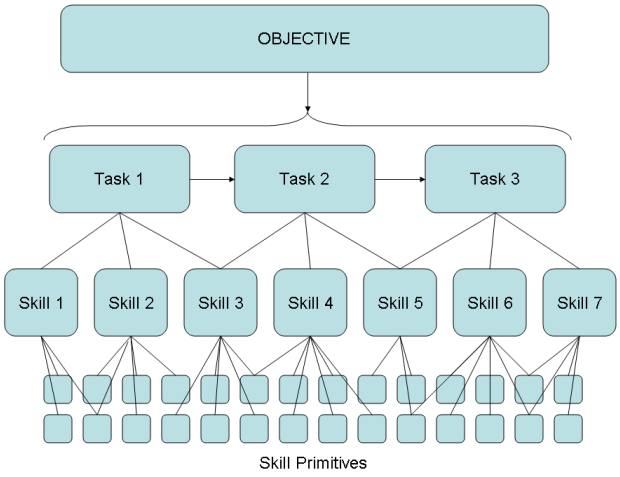

Figure 2. Hierarchy of task decomposition.

achieve the objective, such as attaching two sheets of metal together using a bolt. Tasks can further be broken into skills required to achieve the task, like threading a bolt, and what are called in this work skill primitives, which are simple, robot-specific actions like closing a gripper around an object.

Another aspect to consider is flexibility with respect to the level of abstraction used. The taxonomy model is able to dynamically adjust the level of abstraction depending on the needs of the domain or objective. One example that is highlighted here is the Fasten skill (figure 3.)

3.3. Skill Primitive

Skill primitives refer to the set of basic, atomic actions that a robot is able to perform. Examples include

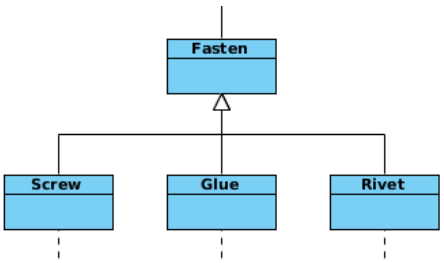

Figure 3. The ***Fasten*** skill can be factored into several different skill primitives; ***Screw***, ***Glue***, and ***Rivet***.

Transport, the basic action of moving from point A to point B, or ***Slide***, the skill to slide across the surface of, say, a wing segment along a specific path or trajectory.

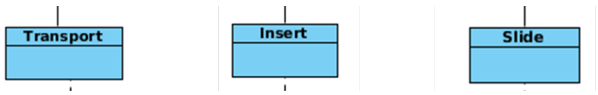

Figure 4. Examples of different skill primitives.

Many of the skill primitives in the assembly taxonomy are straight forward and self-descriptive in their function. For example, the ***Align*** skill primitive is a necessary component that allows a robot to be able to align itself with some feature in the environment, such as an object's pose, relative to some constraints. The ***Insert*** skill primitive is needed to perform any type of insertion task, such as peg-in-hole-type tasks, again relative to certain constraints.

A clear benefit of representing skills in this way is that it allows the representation to be independent of both hardware and implementation. For example, the ***Transport*** skill simply allows the task model to specify that a movement action must be performed,

without worrying about the details of which algorithm must be used to perform that action. In fact any algorithm or combination of algorithms or approaches could be used to perform that movement action through any space, while still satisfying the requirements of ***Transport***.

3.4. Constraints

These skill primitives are defined in terms of the parameters, or constraints, that are placed on them by system requirements for how the action is to be executed. Having a way to specify those requirements and parameters is an essential part of the taxonomy, and is included as shown in figure 5. The types of parameters and constraints depend on the type of action needed.

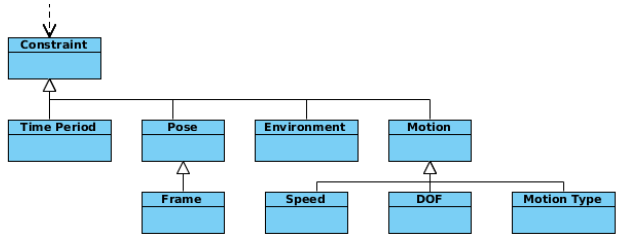

Figure 5. Constraints on skill primitives.

3.5. Skill Library

In the proposed framework, a robot works toward accomplishing the objective through the execution of skill primitives. However, sometimes a robot will come across a series of skill primitives that are frequently performed together repeatedly, such as those required to fetch a part from the parts bin. In these cases, we can take advantage of the hierarchical organization of tasks, and define a high level skill whose parameters satisfy the parameters for each of the component skill primitives. We call the collection of these high level skills available for easy reuse the skill library. An example of this concept is presented below.

Algorithm 1 *

SCREW-INSERT **(X,T)**

1: *DETECT*
2: *TRANSPORT(X)*
3: *ALIGN*
4: *INSERT*
5: *SCREW MOTION(T)*
6: *UNGRASP*
7: *RETRACT*

For the operation of taking an object and screwing it in at the appropriate location, the ***Screw-Insert*** composition is defined. In it the necessary capabilities are present, including moving the part to pose X (the insertion point) , and screwing the part in with parameters T, which is the torque threshold used to determine when the action is complete. Similarly, the ***Pick*** and ***Place*** compositions can also be defined.

3.6. Perception

An indispensable part of the effective use of the taxonomy is the interaction with the perception system. Many different types of perception are required to execute an assembly task, and the taxonomy is able to facilitate this interaction. While the taxonomy discussed is essentially an action taxonomy, or a hierarchy of the skills describing the actions a robot can do, this must also include not just physical actions, but all other actions the robot needs to complete the task, such as detecting features in the environment. This type of representation for perceptual actions is one feature that is missing from many of the systems mentioned in section 2.

The ***Detect*** skill primitive is essentially the interface module into the perception actions. Constraints or parameters into this skill primitive are defined as features to be detected, and include ***Object***, ***Pose***, ***Contact***, and ***Force/Torque***.

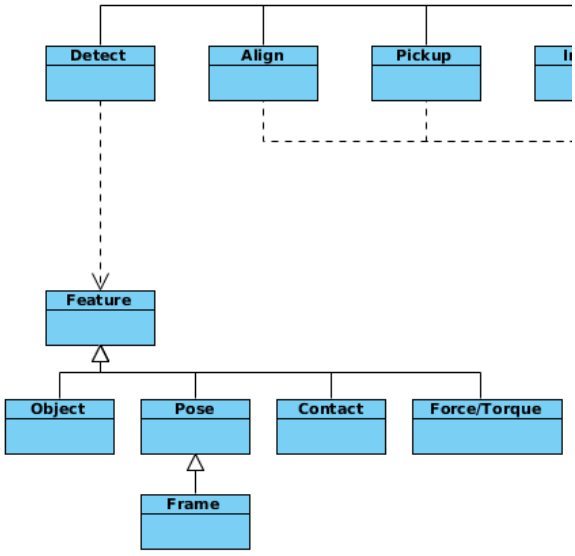

Figure 6. Detect skill.

These feature types handle the various types of perception that is required in assembly. This would cover sensors and algorithms that have to do with pose estimation and object detection, such as cameras and 3D

scanners, and other methods of finding and localizing desired objects for manipulation. Other sensors of interest that are useful in this context include contact sensors and force/torque sensors for detecting contact within the environment and performing complex, accurate manipulation tasks.

4. Demonstration

In this section we discuss a demonstration prepared to show how the taxonomy can be used to model assembly tasks. The first example task used to illustrate this point is the assembly of a model airplane using a Baufix construction kit, as seen in figure 7.

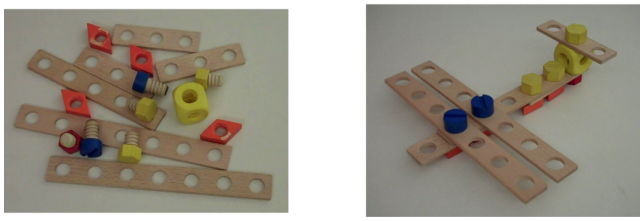

Figure 7. Airplane constructed using toy kit.

The goal is to be able to assemble the model, seen in the right side of the figure, using the component parts shown in the image on the left. The toy parts used to assemble the plane are simple, brightly colored wooden construction parts such as screws, washers, blocks, nuts and boards. The model airplane was built using 17 individual parts.

With this objective in mind, and given the taxonomy of the robot skills available, the next step was the creation of a SysML sequence model, or sequence diagram, of the actual task the robot was to perform. Several SysML sequence diagrams have been prepared to demonstrate a method of using the taxonomy to model the airplane assembly. The sequence diagram is intended to show the organization of an assembly objective into a sequence of actions that are to be performed, and utilizes the skills described in the taxonomy as discrete steps in the sequence.

The diagram (figure 8) shows the model for the assembly of the airplane using a single manipulator and a static fixture designed specifically to aid the robot in this airplane assembly task. This sequence diagram is composed of three main lines; the Robot, the Fixture, and a Part Bin. Each message from the robot to either the fixture or part bin represent an instantiation of a skill primitive (message 1 is a ***Detect***, message 2 is ***Align***, etc.) It is assumed in this simulation that all desired parts in the part bin are visible and accessible. Execution of the assembly sequence is performed

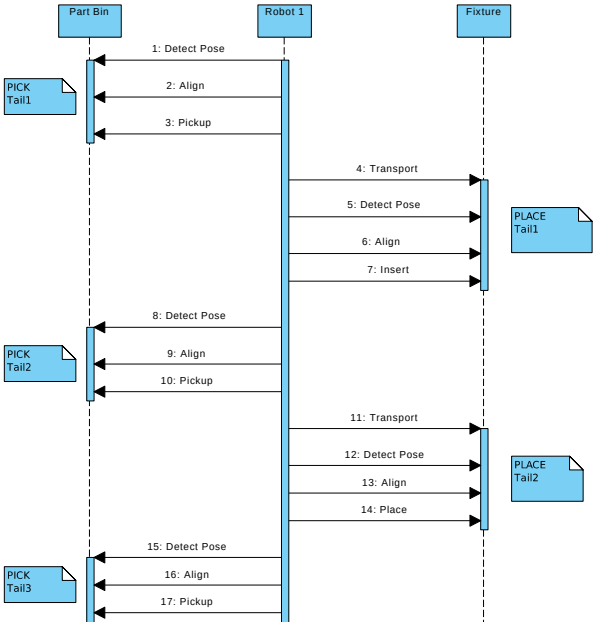

Figure 8. Sequence diagram for airplane assembly.

from top to bottom (as indicated by the numbered messages.) As can be seen on the left of the diagram, small groups of messages (skill primitives) comprise stages (high level skills) in the assembly process. For example, messages 1-3 can be thought to represent the goal of getting part *Tail1* from the part bin, while messages 4-7 place *Tail1* into the fixture. While only a portion of the sequence diagram is shown (the first 17 actions), it took 258 actions, or instantiations of skill primitives, to complete the assembly of the toy airplane.

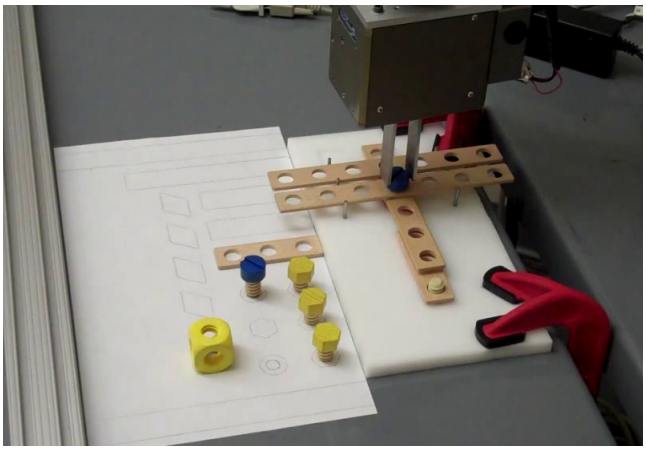

Figure 9. Airplane constructed using toy kit.

Figure 9 shows an image of the robot physically executing the sequence diagram modeled in figure 8. It was programmed using just the simple skill primitives

necessary to complete the task, along with the appropriate parameterizations.

Figure 10. Vehicle constructed using toy kit.

Figure 10 shows the assembly of a second example, in this case a type of vehicle model. In this case, high-level skill compositions from the skill library (which are composed of the individual skill primitives) were used to build the vehicle model. All assembly was done using the ***Pick***, ***Place***, and the ***Screw-Insert*** skills. This second example demonstrates the utility of using high-level skills to simplify the programming problem, as well as the flexibilty of the skill representation in using the same knowledge to model different tasks.

For actual execution on the robot, a few assumptions were made for the demonstration. For example, the assumption was made that the pose estimation could be reliable done, both of the individual parts as well as the locations on the fixture, and so predetermined poses were used to simplify the setup. Using this assumption, and the parameterized skill primitive taxonomy model, the robot was successfully able to assembly both the toy airplane and the vehicle models.

5. CONCLUSION

In this paper we have presented a taxonomy for assembly tasks in the domain of manufacturing and industrial robotics. We have proposed this taxonomy as a robust and flexible method for modeling assembly task descriptions that are generalizable across multiple manufacturing tasks in various configurations. This can be a relatively simple and efficient method for simplifying robot programming and reusing knowledge across these robotic systems, easing the transition to new systems and system configurations, as well as reducing time and financial overhead.

Current and future work with this taxonomy includes a demonstration for how well this approach generalizes across hardware platforms. Work will also focus on how it aids in the general problem of knowledge transfer with humans in the loop, including programming by demonstration and other methods.

References

Abbas, T. and MacDonald, B.A. Generalizing topological task graphs from multiple symbolic demonstrations in programming by demonstration (pbd) processes. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 3816 –3821, may 2011. doi: 10.1109/ICRA.2011.5979848.

Balakirsky, Stephen, Kootbally, Zeid, Schlenoff, Craig, Kramer, Thomas, and Gupta, Satyandra. An industrial robotic knowledge representation for kit building applications. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, 2012.

Dantam, N. and Stilman, M. The motion grammar: Linguistic perception, planning, and control. In *Robotics: Science and Systems*, 2011.

Ekvall, S. and Kragic, D. Learning task models from multiple human demonstrations. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pp. 358 –363, sept. 2006. doi: 10.1109/ROMAN.2006.314460.

Ekvall, S., Aarno, D., and Kragic, D. Task learning using graphical programming and human demonstrations. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pp. 398 –403, sept. 2006. doi: 10.1109/ROMAN.2006.314466.

Finucane, C., Jing, Gangyuan, and Kress-Gazit, H. Ltlmop: Experimenting with language, temporal logic and robot control. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 1988 –1993, oct. 2010. doi: 10.1109/IROS.2010.5650371.

Hitzler, Pascal, Krötzsch, Markus, Parsia, Bijan, Patel-Schneider, Peter F., and Rudolph, Sebastian (eds.). *OWL 2 Web Ontology Language: Primer*. W3C Recommendation, 27 October 2009. Available at http://www.w3.org/TR/owl2-primer/.

Homem de Mello, L.S. and Sanderson, A.C. And/or graph representation of assembly plans. *Robotics and Automation, IEEE Transactions on*, 6(2):188–199, Apr. ISSN 1042-296X. doi: 10.1109/70.54734.

Ijspeert, A.J., Nakanishi, J., Shibata, T., and Schaal, S. Nonlinear dynamical systems for imitation with humanoid robots. In *Proceedings of the IEEE International Conference on Humanoid Robots*, pp. 219–226. 2001. URL http://birg.epfl.ch/page27911.html.

Konidaris, George, Kuindersma, Scott, Grupen, Roderic, and Barto, Andrew. Robot learning from demonstration by constructing skill trees. *The International Journal of Robotics Research*, 2011. doi: 10.1177/0278364911428653. URL http://ijr.sagepub.com/content/early/2011/12/02/0278364911428653.abstract.

Kosecka, Jana and Bajcsy, Ruzena. Discrete event systems for autonomous mobile agents. *Robotics and Autonomous Systems*, 12:187–198, 1993.

Kress-Gazit, H., Wongpiromsarn, T., and Topcu, U. Correct, reactive, high-level robot control. *Robotics Automation Magazine, IEEE*, 18(3):65 –74, sept. 2011. ISSN 1070-9932. doi: 10.1109/MRA.2011.942116.

Krüger, N., Geib, C., Piater, J., Petrick, R., Steedman, M., Wörgötter, F., Ude, A., Asfour, T., Kraft, D., Omrčen, D., Agostini, A., and Dillmann, R. Object-Action Complexes: Grounded Abstractions of Sensory-motor Processes. *Robotics & Autonomous Systems*, 59(10):740–757, 10 2011. doi: 10.1016/j.robot.2011.05.009. URL https://iis.uibk.ac.at/public/papers/Kruger-2011-RAS.pdf.

Lenat, Douglas B. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

Liu, Hugo and Singh, Push. Conceptnet: A practical commonsense reasoning toolkit. *BT Technology Journal*, 22:211–226, 2004.

Lyons, D.M. and Arbib, M.A. A formal model of computation for sensory-based robotics. *Robotics and Automation, IEEE Transactions on*, 5(3):280 –293, jun 1989. ISSN 1042-296X. doi: 10.1109/70.34764.

Nicolescu, Monica N. and Mataric, Maja J. Natural methods for robot task learning: Instructive demonstrations, generalization and practice. In *In Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pp. 241–248, 2003.

OMG. OMG Systems Modeling Language (OMG SysML) Version 1.2 Specification, 2010.

Rosell, J. Assembly and task planning using petri nets: a survey. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 218(8):987–994, 2004.

ROSETTA. The ROSETTA project page. http://fp7rosetta.org, 2009. Accessed 10 March 2012.

SIARAS. Final Report - SIARAS (Skill-based Inspection and Assembly for Reconfigurable Automation Systems). http://cordis.europa.eu/search/index.cfm?fuseaction=lib.document&DOC_LANG_ID=EN&DOC_ID=121979181&q=, 2011.

Singh, Push, Lin, Thomas, Mueller, Erik T, Lim, Grace, Perkins, Travell, and Zhu, Wan Li. Open mind common sense: Knowledge acquisition from the general public. In *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE*, pp. 1223–1237. Springer, 2002.

Tenorth, Moritz and Beetz, Michael. KnowRob — Knowledge Processing for Autonomous Personal Robots. In *IEEE/RSJ International Conference on Intelligent RObots and Systems.*, pp. 4261–4266, 2009.

Waibel, Markus, Beetz, Michael, D'Andrea, Raffaello, Janssen, Rob, Tenorth, Moritz, Civera, Javier, Elfring, Jos, Gálvez-López, Dorian, Häussermann, Kai, Montiel, J.M.M., Perzylo, Alexander, Schießle, Björn, Zweigle, Oliver, and van de Molengraft, René. RoboEarth - A World Wide Web for Robots. *Robotics & Automation Magazine*, 18(2), 2011. Accepted for publication.

Zweigle, Oliver, van de Molengraft, René, d'Andrea, Raffaello, and Häussermann, Kai. Roboearth: connecting robots worldwide. In *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, ICIS '09, pp. 184–191, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-710-3. doi: http://doi.acm.org/10.1145/1655925.1655958. URL http://doi.acm.org/10.1145/1655925.1655958.