The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

Research on Sharing of Robotics Skills

Guilherme C. Affonso*1 Kei Okada Inaba Masayuki

The University of Tokyo, JSK Laboratory

Growing number of robots and increase in its capabilities impose new demands on meta robotic systems, capable of performing hardware abstraction and sharing skills between different robots. In this work such systems are studied, in order to make evident what can actually be done now and what are the problems that must be dealt with in the future.

1. Introduction

Holding the advent of deep learning and the third artificial intelligence boom as background [Matsuo 15], one of the keys of current robotics research is to be *Adaptable* and *Universal*. This gives place to the rise of robots as a new and tangible market, with total sales of household and personal service robots forecast to grow 23.5% per year from 2015 to 2020 [Grace Market Data 15]. Such demands feed robotics research back, in a cycle that leads to growth in both overall number and capabilities of robots.

Under such situation, the administration of specific programs build for specific platforms becomes more and more complex, urging for the unification of software into meta robotic systems capable of abstracting hardware and sharing skills between different robots. Although the creation of a system that can both solve different problems and be compatible with several robots is a common goal for robot software architects, as described in [Ferland 15], robotics research nowadays proves to be highly dependent on hardware platform and software implementation, focusing on environment adaptability rather than universality of execution.

In this work the sharing of skills between robots is explored, in order to delimit what can currently be done towards its solving and make clear what kind of problems still need to be overcome.

2. Definition and Classification of Skills

In order to reason about the sharing of robotic skills, it is necessary to first properly define them in clear and concrete terms. In this work we use the definitions proposed by Affonso et al. [Affonso 18b], modified from Jacobsson et al. [Jacobsson 16] and described as follows.

- **Skill**: *The potential for a robot to reproduce a certain task.*
- **Task**: *A set of actions with meaningful outcome.*
- **Action**: *A set of motions which cause an effect on the outside environment.*
- **Motion**: *Movements that alter the robot's physical state.*

Furthermore, classification of skills into joint, action point and objective level, also given at Affonso et al.'s [Affonso 18b] work, are used. These are described as follows.

1. **Joint level**
   At joint level, each motion is described by a vector gathering angle values for all of robot's degrees of freedom. Skills are the enumeration of such motions.
2. **Action point level**
   At action point level, skills are described by the trajectory of the end effector, with emphasis on its points of application. Possible relations with target objects and effects on the outside environments are also considered, being equivalent to the overall procedure for achieving the task in question.
3. **Objective level**
   At objective level, skills are solely described by the final state to be achieved, without further instruction on motions and subprocedures to be taken.

The above definitions are treated as equivalent statements for appearance level, action level and purposive task level imitation of human behavior, defined by Kuniyoshi et al. [Kuniyoshi 07], for being the sharing of skills between two robots comparable to the imitation between them.

3. Sharing Robotic Skills

3.1 Requirements on Skill Sharing

To share a skill between two robots means, in simple statements, that both of these robots will be able to achieve a task that previously only one of them could accomplish. In further analysis, for being tasks characterized by its *meaningful outcome*, or, in other words, objective, we can understand skill sharing as both the sharing of objectives and the ability to achieve them.

Regarding the sharing of objectives, it is easy to conclude that it can be accomplished through the description of skills in objective level. The opposite is also valid, being possible to say that the generation of objective level statements is a

Contact: Guilherme C. Affonso, The University of Tokyo, 113-8656 Tokyo, Bunkyo-ku, Hongo, 7-3-1, tel. 03-5841-7416, affonso@jsk.imi.i.u-tokyo.ac.jp

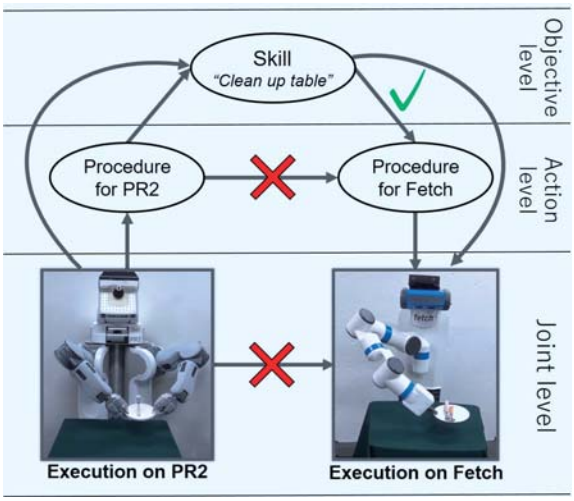

Figure 1: Sharing of robotic skills, as the transition between objective level and task level descriptions.

necessary step for the sharing of objectives and, therefore, skills.

Regarding the ability to achieve a certain objective *i.e.* final state, it is vital to observe that, as any machine, robots are only capable of executing sufficiently concrete and small orders. This means that all commands send to the real robot must be ultimately translated into joint level orders before execution. Therefore, the ability to achieve a certain objective is held as being equivalent to the potential to generate joint level orders based on a given final state.

Above reasoning leads to the conclusion that in order to attain general and certain sharing of robotic skills, it is necessary to (i) share objective level descriptions, and (ii) generate joint level commands accordingly.

An interesting point to notice is that, although being impossible to share skills between different robots directly on joint level, for those being extremely hardware dependent, it is possible to obtain functional results when sharing directly on action point level. This happens in cases where both robots have sufficiently similar hardware properties, such as number of limbs and overall reachability, enough to grant that the original procedure remains valid on the new robot. However, these are limited cases, having general and certain sharing of skills relying on, as previously stated, the combination of objective and joint level descriptions.

3.2 Difficulties on Skill Sharing

The main difficulty imposed by the sharing of robotic skills is the necessity for the machine to deal with objective level descriptions, either in its extraction or in its conversion into joint level commands. Differently from humans, who act consciously of the cause and effect of each motion, knowing what they should and should not do in order to achieve certain objectives; robots – and machines, in general – are only able to process concrete and simple orders, which are executed without any concern on its original purposes or potential side effects. This is due to the incapability for machines to comprehend the cause-consequence nexus, widely known as the *Frame Problem*. The *Frame Problem* can also be understood as the difficulty for artificial intelligence to classify phenomena between related and unrelated [Matsuo 15], in more basic levels. Because it is impossible to know the relations of a certain event with other ones, it is impossible to know which phenomena is directly influenced by which event, being therefore impossible to understand the consequences of actions taken by others or themselves.

4. Past Work

In the past years great effort have been made to build up objective level interfaces, in which programming would be resumed to the definition of the desired final state, using syntax resembling natural language or logical programming like statements. Some examples of such task level programming languages can be verified on [Lozano-Perez 77] [Lieberman 77] [Okano 88], holding the merit of (i) intrinsically attaining the sharing of objectives, and (ii) easing the act of programming itself, for providing methods closer to the objective-oriented human behavior. However, even making use of such interfaces, the necessity of assigning appropriate joint level commands still prevails, making the sharing of skills between different hardware unable to be achieved.

When attempting to deal with the Frame Problem, KnowRob [Tenorth 09] can be pointed out as being of notable interest. Knowrob is a knowledge database implemented on Prolog and the Web Ontology Language OWL, in which the analysis of its enormous database allows robots to learn from experience and reason about the environment and effects of its actions. However, although it is possible to correctly assign subprocedures to higher level commands using this method, it is still limited to specific cases.

Extraction of joint level commands from higher descriptions also finds precedents on the inverse kinematics field [Buss 04] [Nakamura 86], which enables conversion of coordinate based action point level commands into joint level orders. Even though inverse kinematics calculation relies on the number of links and its length, being highly hardware dependent, common interfaces capable of building appropriate solvers based on each robot's hardware descriptions are observed [Matsui 90]. Use of such interfaces allows writing programs fully in action point level, being able to share overall procedures between robots. Although not able to attain complete sharing of skills, sharing of procedures can also be effective on platforms with similar hardware. However, since inverse kinematics result depends on initial position, such codes tend to rely on environment conditions, generating unstable outcomes.

5. Joint - Action level Transition

In this work the extraction of action point level procedures from joint level commands is tested. When combined with common interfaces capable of translating coordinate based commands back into joint level orders, as the ones introduced above, this method makes possible to share procedures of any robotic skill. This is done through source code

and run-time data analysis, by (i) determining and translating motion patterns, and (ii) identifying each action and the relations between them. Details of such method are given on [Affonso 18b], which also proposes the use of a visual and natural language-like programming based interface for facilitating the manipulation of extracted procedures, making therefore possible to share skills in a semi-automatic way.

At the first step, motions patterns are identified by (i) segmenting and labeling original code into minimal, unconditional divisions, and (ii) searching for common sets of labels between executions. Each motion is then translated into action point level descriptions by comparing end effector position relative to the robot base, target object center and previous position. Coordinate systems capable of describing the end effector position in unchanged statements are taken to be the application point of the motion, process illustrated on Fig. 2.

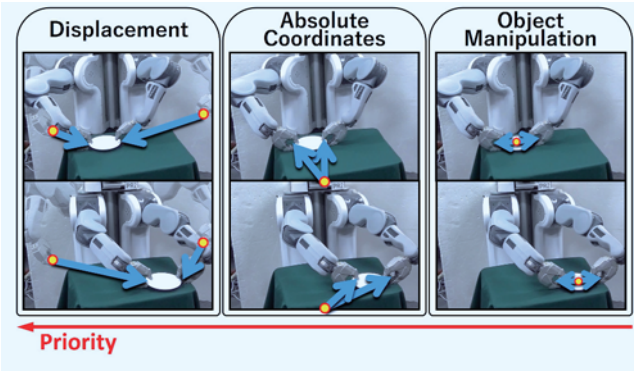

Figure 2: Example of coordinates analysis on motion translation. Here, object centered coordinate system allows unchangeable representation of end effector position, signalizing the application point.

At the second step, the heading label for each motion pattern is analyzed, identifying its trigger conditions and other related motion patterns included on its context. Nested structures containing several motion patterns related to the same conditional clause are interpreted as actions, and bound accordingly. Detailed procedure is given on [Affonso 18a].

6. Experimentation

Experimentation is conducted on the simple domestic task of cleaning up a table after a meal, removing dishes and cutlery left on it. This task is schematically shown at Fig. 3, and is executed a total of seven times for collecting run-time data. Conditions of each trial are given at Table 1.

Without counting the action describing initial pose, five motions patterns and three actions were extracted from given task. Overall procedure, including the relation between such actions and motion patterns, is shown at Fig. 4. Here, actions A, B and C are verified to correspond to original actions of piling-up dishes, collecting cutlery and picking-up (refer to Fig. 3).

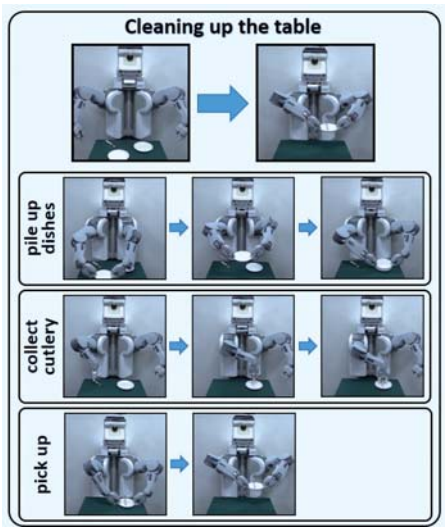

Figure 3: Task of cleaning up table, executed by piling up dishes, collecting cutlery and picking everything up.

Table 1: Conditions of conducted experiments.

trial number	1	2	3	4	5	6	7
number of plates	1	1	1	1	2	2	2
number of cutlery	0	1	1	2	0	0	1
pile plate executions	0	0	0	0	1	1	1
collect cutlery exec.	0	1	1	2	0	0	1
pick up executions	1	1	1	1	1	1	1

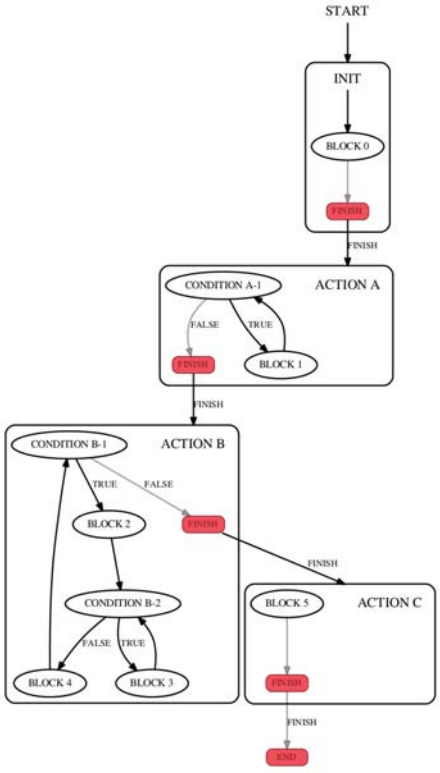

Figure 4: Flow diagram describing task's overall procedure, showing relations between extracted motion patterns (*blocks*) and actions. Conditions are abbreviated.

7. Conclusion

In this work the sharing of robotic skills was dealt with, analyzing its current situation and problems that must be dealt with.

From past work, we can conclude that now it is possible to:

1. Share objectives between robots, by *e.g.* using task level programming languages.
2. Translate coordinate based action point level descriptions into joint level orders, by *e.g.* using inverse kinematics solvers.

In order to attain the complete sharing of skills among different robots, however, it is also necessary to be capable of executing shared objectives, by assigning joint level orders appropriately. This involves dealing with reasoning on higher level objectives, being required to overcome the Frame Problem – difficulty for machines to understand the cause-consequence nexus – at some extent.

We also note that, although not being able to accomplish sharing in general and certain ways, the extraction of action point level procedures can be helpful, for being able to both give functional results on similar hardware and attain certain degree of hardware abstraction. When willing to share a certain skill between robots, even if incomplete this methods can save a lot of time and human work.

References

[Affonso 18a] Affonso, G. C., Okada, K., and Inaba, M.: Detection of Motion Patterns and Transition Conditions for Automatic Flow Diagram Generation of Robotic Tasks, in *International Conference on Intelligent Autonomous Systems (IAS)* (submitted on 2018)

[Affonso 18b] Affonso, G. C., Okada, K., and Inaba, M.: Extraction and Modularization of Procedures Towards Sharing of Robotic Skills, in *International Conference on Intelligent Robots and Systems (IROS)* (submitted on 2018)

[Buss 04] Buss, S. R.: Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods, *IEEE Journal of Robotics and Automation*, Vol. 17, No. 1-19, p. 16 (2004)

[Ferland 15] Ferland, F., Cruz-Maya, A., and Tapus, A.: Adapting an hybrid behavior-based architecture with episodic memory to different humanoid robots, in *24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 797–802 (2015)

[Grace Market Data 15] Grace Market Data, : Global Domestic Service Robots Market Size And Trends To 2020 (August 2015)

[Jacobsson 16] Jacobsson, L., Malec, J., and Nilsson, K.: Modularization of skill ontologies for industrial robots, in *Proceedings of 47st International Symposium on Robotics(ISR)*, pp. 1–6 (2016)

[Kuniyoshi 07] Kuniyoshi, Y.: Adaptive and emergent imitation as the fundamental of humanoid intelligence, *Journal of the Robotics Society of Japan*, Vol. 25, No. 5, pp. 671–677 (2007)

[Lieberman 77] Lieberman, L. I. and Wesley, M. A.: AUTOPASS: an automatic programming system for computer controlled mechanical assembly, *IBM Journal of Research and Development*, Vol. 21, No. 4, pp. 321–333 (1977)

[Lozano-Perez 77] Lozano-Perez, T. and Winston, P. H.: LAMA: A Language for Automatic Mechanical Assembly., in *IJCAI*, pp. 710–716 (1977)

[Matsui 90] Matsui, T. and Inaba, M.: Euslisp: An object-based implementation of lisp, *Journal of Information Processing*, Vol. 13, No. 3, pp. 327–338 (1990)

[Matsuo 15] Matsuo, Y.: 人工知能は人間を超えるか ディープラーニングの先にあるもの [Will Artificial Intelligence overcome humans? What lies beyond Deep Learning., KADOKAWA (2015)

[Nakamura 86] Nakamura, Y. and Hanafusa, H.: Inverse kinematic solutions with singularity robustness for robot manipulator control, *Journal of dynamic systems, measurement, and control*, Vol. 108, No. 3, pp. 163–171 (1986)

[Okano 88] Okano, A., Matsubara, H., and Inoue, H.: Design and implementation of a task-oriented robot language, *Advanced Robotics*, Vol. 3, pp. 177–191 (1988)

[Tenorth 09] Tenorth, M. and Beetz, M.: KnowRob – knowledge processing for autonomous personal robots, in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 4261–4266 (2009)