Improving productivity and worker conditions in assembly
Part 2: rapid deployment of learnable robot skills

Cristian Vergara[1], Greet Van de Perre[2], Ilias El Makrini[2], Bram B. Van Acker[3,4], Jelle Saldien[3], Liliane Pintelon[5], Peter Chemweno[5], Raphaël Weuts[5], Karen Moons[5], Sofie Burggraeve[6], Bram Vanderborght[2], Erwin Aertbeliën[1] and Wilm Decré[1]

***Abstract*— Collaborative robots (cobots) have a strong potential to improve both productivity as well as the working conditions of assembly operators by assisting in their tasks and by decreasing their physical and cognitive stress. The use of cobots in factories however introduces multiple challenges: how should the overall assembly architecture look like? How to allocate specific (sub)tasks to the operator or the cobot? How to program and deploy the cobot? How to make changes to the robot program?**

In this paper dilogy, we briefly highlight our recent contributions to this field. In part I we presented our collaborative architecture for human-robot assembly tasks and discussed the working principles of our task allocation framework, based upon agent capabilities and ergonomic measurements. In this second part we focus on our programming by demonstration approach targeted at expediting the deployment of learnable robot skills.

I. Introduction

Large parts of the world are confronted with an aging workforce and an increasing number of people outside the working-age[1]. With age, the physical capabilities gradually decrease, leading to a higher risk of musculoskeletal deficiencies for physically demanding jobs. In addition, the cognitive capacities such as working memory, patience and flexibility may deteriorate with age, which negatively influences the operator's performance [1], [2]. Especially for this population, the use of cobots can be interesting to decrease both the cognitive and the physical load. While humans have their strength in problem-solving and dexterity, robots complement by their ability of carrying heavy loads and performing repetitive and precise tasks. Therefore, by working together, the robot can assist to lower the physical work load [3]. Additionally, cobots can reduce the operator's cognitive load to obtain a higher quality and less error-prone production [4], which is especially interesting for manufacturing in low quantities and high variability.

The cost associated with the time and expertise needed for programming robots often forms a substantial share of the total costs of an automation project. We expect that this share can even increase because manufacturers are increasingly demanding flexibility (smaller lot sizes, more product varieties), requiring more advanced programming (including sensor-based robot control) and fast reconfigurations to other scenarios. Furthermore, several new players emerge on the collaborative robot market, next to the well-established industrial robotic companies (KUKA with the LBR iiwa, ABB with the YuMi, ...), leading to pricing pressure on cobot hardware. As a result, the share of the programming cost in the total project cost further increases.

Robot programming approaches, with programming considered in a broad sense, that can reduce the programming/application deployment time are hence a key element in boosting the pickup of cobots by industry.

Imitation learning has this potential to facilitate the deployment of robot applications, both in time as well as in technical expertise level required from the robot *programmer*. The goal is to develop robot applications that are quickly deployable into new situations and that are robust against variations in the environments, including humans in the same workspace as the robot. Our approach can be understood in terms of three (out of the many) stakeholders involved in a robotic application:

1) The **application developer** develops a generalized robotic application that can be deployed in different scenarios. He is mainly an application expert, but is able to program the application while deferring certain aspects to the deployer.
2) The **(application) deployer** deploys the robotic application in a specific scenario. This can be done fast and without much training because, instead of using traditional robot programming, he can teach the robot in the current scenario and environment by demonstrating the motions to the robot system. He uses a graphical user interface (GUI) to specify some of the application parameters (e.g., an insertion force).
3) The **operator** interacts on a daily basis with the robot in a natural way and performs human operations in the same work space. He has the possibility to modify application parameters through the GUI.

[1] Robotics Research Group, PMA Division, KU Leuven, Belgium, Core Lab Flanders Make, www.mech.kuleuven.be/robotics, Corresponding author: `cristian.vergara@kuleuven.be`

[2] Robotics and Multibody Mechanics Research Group, Vrije Universiteit Brussel, Belgium, Core Lab Flanders Make, www.brubotics.eu

[3] Department of Industrial Systems and Product Design, Faculty of Engineering and Architecture, Ghent University, Belgium

[4] Department of Personnel Management, Work and Organizational Psychology, Faculty of Psychology and Educational Sciences, Ghent University, Belgium

[5] Centre for Industrial Management, KU Leuven, Belgium

[6] CodesignS, Core Lab Flanders Make, Belgium

[1]United Nations: `http://www.un.org/en/development/desa/population/publications/pdf/ageing/WPA2017_Report.pdf`

II. Constraint-based programming

The application developer specifies the generalized robot application using *expressiongraph-based task specification language* eTaSL [6], [11] due to its versatility and intuitive approach to constraint-based task specification.

eTaSL is built upon expression graphs which contain geometric entities (e.g., trajectories, orientations, positions, robot kinematics) and allows mathematical operations between them. Robotic behaviors can be specified throughout the use of (possibly conflicting) constraints ruled by priorities and weights. Besides the robotic joints and time variables, extra degrees of freedom in the robotic application can be specified using *feature variables*, e.g., specify a motion along a trajectory in function of the path coordinate.

This framework brings several advantages to robotic applications:

1) high composability of robotic behaviors; aspects related to robot, environment, task can be specified and reused separately. Examples are motion along trajectory, compliance with the environment, collision avoidance, joint limits;
2) sensor-based interactions that are tightly integrated with robot control (localization by vision, force control, collision avoidance from proximity sensors (see skin in fig. 1b);
3) advanced robot behavior while still having interactions with the user.

III. Imitation Learning

An imitation learning approach is implemented such that we can use demonstrations to capture how the trajectories should vary under the influence of a dynamic environment. A set of basis functions is learned from demonstrations to represent trajectories and their variations. To this end, a probabilistic principle component analysis (PPCA) is used in a similar way as [7], [8]. The demonstrations are performed using *kinesthetic teaching*: the robot follows human motion at the end effector using force constraints (see fig. 1a). An advantage of our constraint-based approach is that the kinesthetic teaching can be performed while some of the application constraints are still active (such as collision constraints, joint limits, etc.)

Rapid deployment can be facilitated by keeping the number of demonstrations low using *incremental learning*: an initial learned model is gradually refined by introducing new demonstrations. The constraint-based approach allows us to use the information of previous demonstrations during the kinesthetic teaching, such that the deployer feels stiffer behavior (away from previous model work space) along sections with a lower variability in the previous demonstrations.

In this way demonstrations are facilitated and only the required number of demonstrations is performed.

IV. Use case inspired by an industrial application

The proposed framework is tested in a use case inspired by an industrial application. A rack with five solenoids and a hub, in which the solenoids will be assembled, are placed inside a workstation. The locations and orientations of the solenoids and the corresponding hub holes vary during the day-to-day operation of the application. The location and orientation are sensed by a camera system. In the approach and retract motions during the picking and insertion operations, collisions should be avoided between the solenoids and the environment (see fig. 1c), while still reaching the variable poses of the solenoids and hub holes. To this end, the deployer demonstrates four initial trajectories (see fig 1a). Extra demonstration are performed using our incremental learning approach to extend the work space of the learned model until satisfactory motion is achieved. During the execution of the application, the trajectories are obtained by constraining the end point of the predicted trajectory to targets extracted by the camera. A force of 100 N must be exerted to insert the solenoid in the hub (see fig 1c). To this end, a damped force control strategy similar to [9] is specified using the constraint-based methodology. After the insertion operation, the operator intervenes in the workstation to fix the solenoid by placing two screws while the robot picks the next solenoid. Velocity constraints based on the proximity signals sensed by an artificial skin [10] are specified to avoid collisions [5]. This allows the operator to interrupt or drive backwards the motion of the end effector such that he can finish his task (see fig 1b). The learned motion model is used for all of the five solenoids.

V. Conclusions and future work

In this paper, we described our imitation learning approach integrated in our constraint-based robot programming framework. As future work, we are focusing on developing methods to further reduce the number of required demonstrations, as this will directly lead to shorter programming times. We plan to realize this by further exploiting the composability property of constraint-based programming, allowing us to make an optimal trade off between modeling knowledge and learning through demonstrations.

Acknowledgment

All authors gratefully acknowledge the financial support of Flanders Make through project YVES_SBO.

References

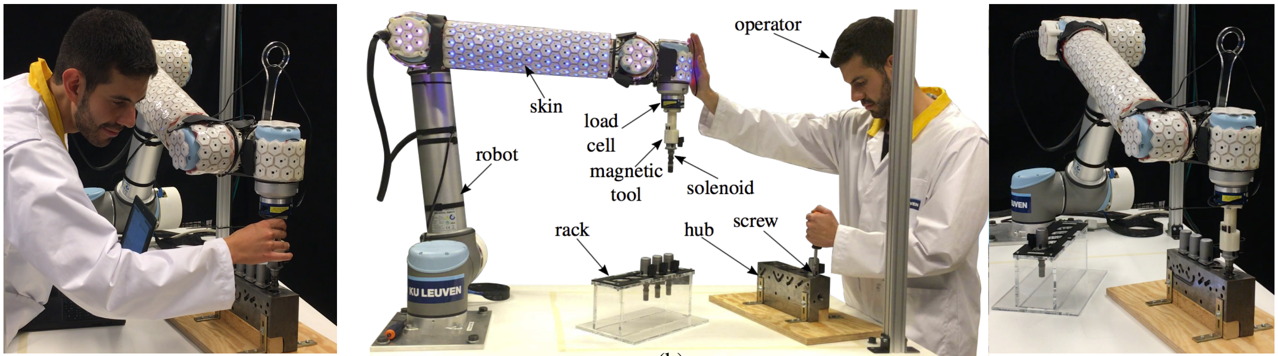
[1] R. W. Morrell and D. C. Park, “The effects of age, illustrations, and task variables on the performance of procedural assembly tasks.” *Psychology and aging*, vol. 8, no. 3, p. 389, 1993.

[2] M. S. Young, K. A. Brookhuis, C. D. Wickens, and P. A. Hancock, “State of science: mental workload in ergonomics,” *Ergonomics*, vol. 58, no. 1, pp. 1–17, 2015.

[3] K. N. Kaipa, C. Morato, J. Liu, and S. K. Gupta, “Human-robot collaboration for bin-picking tasks to support low-volume assemblies,” in *Human-Robot Collaboration for Industrial Manufacturing Workshop, held at Robotics: Science and Systems Conference (RSS 2014)*, 2014.

[4] A. Bauer, D. Wollherr, and M. Buss, “Human–robot collaboration: a survey,” *International Journal of Humanoid Robotics*, vol. 5, no. 01, pp. 47–66, 2008.

[5] C. Vergara, G. Borghesan, E. Aertbeliën, and J. De Schutter, “Incorporating artificial skin signals in the constraint-based reactive control of human-robot collaborative manipulation tasks,” in *2018 3rd International Conference on Advanced Robotics and Mechatronics (ICARM)*, Singapore, 2018, p. Forthcoming.



(a) (b) (c)

Fig. 1: Human-robot collaborative workstation solenoid assembly use case. (a) the application deployer demonstrates approaching trajectories; (b) the operator interrupts the robot motion at runtime while accomplishing his task [5]; (c) the robot performs the insertion operation.

[6] E. Aertbeliën and J. De Schutter, "eTaSL/eTC: A constraint-based task specification language and robot controller using expression graphs," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Chicago, IL, USA: IROS2014, 2014, pp. 1540–1546.

[7] E. Aertbeliën and J. De Schutter, "Learning a predictive model of human gait for the control of a lower-limb exoskeleton," in *5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*, 2014, pp. 520–525.

[8] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society (Series B)*, vol. 61, no. 3, pp. 611–622, 1999.

[9] J. Baeten, H. Bruyninckx, and J. De Schutter, "Integrated vision/force robotics servoing in the task frame formalism," *The International Journal of Robotics Research*, vol. 22, no. 10, pp. 941–954, 2003.

[10] P. Mittendorfer, Eiichi, and Cheng, "3d surface reconstruction for robotic body parts with artificial skins." Vilamoura, Portugal: IROS2012, 2012, pp. 4505–4510.

[11] E. Aertbeliën,"The eTaSL task specification language,"`https://people.mech.kuleuven.be/~eaertbel/etasl`, last visited July 2018.