User Needs and Design Opportunities in End-User Robot Programming

Gopika Ajaykumar
gopika@cs.jhu.edu
Johns Hopkins University
Baltimore, MD 21218, USA

Chien-Ming Huang
cmhuang@cs.jhu.edu
Johns Hopkins University
Baltimore, MD 21218, USA

Abstract

We report on a user study that sought to understand how users program robot tasks by direct demonstration and what problems they encounter when using a state-of-the-art robot programming interface to create and edit robot programs. We discuss how our findings translate to design opportunities in end-user robot programming.

Keywords

End-User Robot Programming; Programming by Demonstration

ACM Reference Format:
Gopika Ajaykumar and Chien-Ming Huang. 2020. User Needs and Design Opportunities in End-User Robot Programming. In *Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction (HRI '20 Companion), March 23–26, 2020, Cambridge, United Kingdom.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3371382.3378300

1 Introduction

The increasing emergence of assistive and collaborative robots in everyday human environments, such as schools, hospitals, and homes, has the potential to greatly increase quality of life by enabling robotic assistance with daily living tasks, tedious errands, and difficult or unfilled jobs. For robots to assist to their fullest abilities in these capacities, they must be capable of adapting to changing demands, environments, and scenarios. However, developing a robot that can autonomously respond appropriately to every possible situation is currently an intractable problem. *End-user robot programming* offers an alternative approach that facilitates the integration of assistive and collaborative robots into human environments in the short term.

Research in end-user robot programming seeks to enable end-users, including those without a technical background, to easily program a robot to perform custom tasks with contextual constraints [5]. One of the most common robot programming paradigms that has emerged is *Programming by Demonstration* (PbD), also known as *Learning from Demonstration* (LfD) or imitation learning [4, 6, 7]. For the PbD method of robot programming, end-users can demonstrate or teach a robot skills and the application of skills to different task configurations (e.g., [2, 8, 16, 26]). Several different modalities for providing these demonstrations have been explored, such as kinesthetic teaching [1, 11, 18], natural language [14, 15, 17, 22–24], video-based demonstrations [8, 25, 26], visual programming interfaces [3, 10, 13, 19], situated specification [9, 21], and teleoperative demonstrations through motion capture [12, 20] and virtual reality [27]. Among these methods, kinesthetic teaching enables end-users to program a robot by physically guiding it through a task, whereas visual programming allows a user to specify a robot program graphically. Today, PbD is often implemented as a hybrid of kinesthetic teaching and visual programming. Below we report on a user study that sought to examine the ease of use and learnability of one such composite interface, *Universal Robots*' Polyscope (Figure 1).

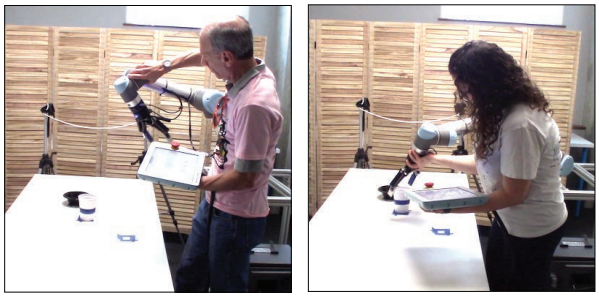

Figure 1: Users face various difficulties when using a state-of-the-art interface to program a collaborative robot. This report describes these difficulties and discusses design opportunities for future robot programming interfaces.

2 User Experiences in End-User Robot Programming

To explore user experiences in programming a collaborative robot, we designed four common manipulation tasks, which involved stacking, hanging, and pouring task objects such as blocks, cups, and towels, and asked participants to complete the tasks with a UR-5 6-DOF robotic arm. Participants were instructed to use kinesthetic teaching to program the robot and were free to use either continuous trajectories, waypoints, or a combination of both during the programming process.

Following informed consent, the participant was provided with a tutorial on how to use the state-of-the-art programming interface for the UR-5, the Polyscope interface on the teach pendant device that comes with the robot. They were then asked to program a practice pick-and-place task. After completing the practice task, the participants completed four tasks using the programming interface. For the purposes of the study, the participant could correct the program as many times as they wanted until they were happy with the task result. The participant was stopped, regardless of their progress, fifteen minutes before the hour to complete an open-ended

HRI '20 Companion, March 23–26, 2020, Cambridge, United Kingdom

ACM ISBN 978-1-4503-7057-8/20/03.
https://doi.org/10.1145/3371382.3378300

interview and demographics questionnaire. The study was video recorded, and participants received $10 USD for study completion.

Eight participants (6 females) were recruited for the study, with ages ranging from 19 to 57 ($M = 31.13, SD = 16.22$). Participants came from various backgrounds and disciplines and had varying degrees of experience in programming and technology.

2.1 Observations and Findings

Below, we summarize key observations and findings from our study, highlighting user frustrations and challenges in specifying and editing robot programs.

Challenges with Programming Interface. The majority of our observations from the user study and interview related to problems users found with the programming interface. In particular, we found that participants had a poor mental model of their programs, especially those that involved waypoints. This problem led to participants being unpleasantly surprised by the results of their programs. For example, one participant thought that trajectories between waypoints were recorded and represented actual paths that the robot would later trace through. Such inadequate understanding often led to a task failure or unexpected collision with obstacles.

Nevertheless, participants preferred waypoints to recording continuous trajectories, as they found that using the path recording command built in Polyscope required a high cognitive load since they had to keep track of several steps to complete the recording. Two participants forgot to record the path during one of the tasks, which required keeping track of several task steps and may have made keeping track of all the recording steps even more difficult.

Participants also thought the programming interface could be visually and spatially improved. Some participants found the tab structure of the Polyscope interface confusing, especially towards the beginning of the study, since it separates robot action selection from robot action parameter specification. One participant suggested that it would be more intuitive for command selection and editing to be located in one central location. Similarly, a few participants found that there was irrelevant or redundant information on the screen. One participant suggested that the use of color could help with visual organization, especially considering that the current interface is largely monochrome.

Challenges with Kinesthetic Teaching. Three participants directly mentioned how difficult programming the robot using kinesthetic teaching can be during the interview, while some participants conveyed this indirectly when they expressed tiredness while programming the demonstration or when they showed reluctance in, or even decided to forego, correcting their incorrect programs. In addition, participant fatigue during kinesthetic teaching resulted in extraneous movements and lag time, where no movement occurred, in most of the participants' programs using continuous recording.

Challenges with Program Editing. We also observed aspects related to how the process of making corrections to a program could be improved. Two participants mentioned that it was hard to correct the program without having any visualization of the demonstration to refer to. They said that it would have been useful to have a 3-D visualization of the entire path during corrections. Furthermore, several participants wanted to check their work in the process of programming, or even wanted to view the result of the program before editing, but there was no easy way to do this with the Polyscope interface. Similarly, one participant mentioned that it would have been useful to have a feature that could easily bring the robot to a specific point in the program. In a similar vein, participants commented that there is no easy way to undo actions by restoring the state of the program before adding a command using the current interface. Without these capabilities, users were often forced to restart the demonstration from the beginning to make a correction, which they found difficult and frustrating.

3 Design Opportunities in End-User Robot Programming

Based on the observations and feedback we received from participants, we identified several design opportunities for end-user programming interfaces.

Program Visualization and Interface Organization. End-user programming interfaces should have some form of 3-D visual representation of robot programs to help users form and preserve appropriate mental models of their demonstrations, especially those involving waypoints, since waypoints can be confusing without visualization of their locations and surrounding context. Effective end-user programming interfaces should also minimize prerequisite and requisite steps for programming by including simple "one-step" programming actions and commands to minimize cognitive load for users. Interfaces should minimize use of tabs and keep similar actions and commands coherently grouped together. They should organize information in a manner such that items irrelevant to the task do not contribute to visual clutter. Color could be a useful tool for structuring information in interfaces, though it should not be over-relied upon since not all populations are able to see all colors.

Editing and Debugging Capabilities. Visualization is not only critical to the process of specifying robot programs but also essential to effective editing and debugging. In addition to visualizing the full robot program, end-user robot programming interfaces should have easy-to-use replay capabilities to visualize contextualized portions of the robot program. Moreover, since undoing actions is a critical part of editing and correcting, interfaces should have easy- and quick-to-use undo capabilities. These capabilities will directly enhance programming efficiency, allowing users to make changes without starting over with a new demonstration.

Overall, the user experiences that we observed demonstrate that there still remains room for improvement in current end-user programming interfaces with respect to increasing usability and maximizing user productivity. In addition to drawing inspiration from user needs, the design opportunities we describe in this report build off of existing software programming interfaces (e.g., undo, breakpoints) and animation authoring tools (e.g., simulated preview, trajectory illustration and editing), suggesting that designers can draw tools from various sources to create user-centered end-user robot programming toolboxes.

Acknowledgments

This work is partially supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1746891 and the Nursing/Engineering joint fellowship from Johns Hopkins University.

References

[1] Baris Akgun, Maya Cakmak, Jae Wook Yoo, and Andrea Lockerd Thomaz. 2012. Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*. ACM, 391–398.

[2] Jacopo Aleotti and Stefano Caselli. 2006. Robust trajectory learning and approximation for robot programming by demonstration. *Robotics and Autonomous Systems* 54, 5 (2006), 409–413.

[3] Sonya Alexandrova, Zachary Tatlock, and Maya Cakmak. 2015. RoboFlow: A flow-based visual programming language for mobile manipulation tasks. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5537–5544.

[4] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. 2009. A survey of robot learning from demonstration. *Robotics and autonomous systems* 57, 5 (2009), 469–483.

[5] Geoffrey Biggs and Bruce MacDonald. 2003. A survey of robot programming systems. In *Proceedings of the Australasian conference on robotics and automation*. 1–3.

[6] Aude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. 2008. Robot programming by demonstration. *Springer handbook of robotics* (2008), 1371–1394.

[7] Sonia Chernova and Andrea L Thomaz. 2014. Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 8, 3 (2014), 1–121.

[8] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. 2017. One-shot visual imitation learning via meta-learning. *arXiv preprint arXiv:1709.04905* (2017).

[9] Yuxiang Gao and Chien-Ming Huang. 2019. PATI: a projection-based augmented table-top interface for robot programming. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. ACM, 345–355.

[10] Dylan F Glas, Takayuki Kanda, and Hiroshi Ishiguro. 2016. Human-robot interaction design using Interaction Composer eight years of lessons learned. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 303–310.

[11] Micha Hersch, Florent Guenter, Sylvain Calinon, and Aude Billard. 2008. Dynamical system modulation for robot learning via kinesthetic demonstrations. *IEEE Transactions on Robotics* 24, 6 (2008), 1463–1467.

[12] Kaijen Hsiao and Tomas Lozano-Perez. 2006. Imitation learning of whole-body grasps. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5657–5662.

[13] Justin Huang and Maya Cakmak. 2017. Code3: A system for end-to-end programming of mobile manipulator robots for novices and experts. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*. IEEE, 453–462.

[14] Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. 2010. Toward understanding natural language directions. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*. IEEE Press, 259–266.

[15] Stanislao Lauria, Guido Bugmann, Theocharis Kyriacou, and Ewan Klein. 2002. Mobile robot programming using natural language. *Robotics and Autonomous Systems* 38, 3-4 (2002), 171–181.

[16] Alex X Lee, Henry Lu, Abhishek Gupta, Sergey Levine, and Pieter Abbeel. 2015. Learning force-based manipulation of deformable objects from multiple demonstrations. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 177–184.

[17] Cynthia Matuszek, Liefeng Bo, Luke Zettlemoyer, and Dieter Fox. 2014. Learning from unscripted deictic gesture and language for human-robot interactions. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

[18] Carl Mueller, Jeff Venicx, and Bradley Hayes. 2018. Robust Robot Learning from Demonstration and Skill Repair Using Conceptual Constraints. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 6029–6036.

[19] Chris Paxton, Andrew Hundt, Felix Jonathan, Kelleher Guerin, and Gregory D Hager. 2017. CoSTAR: Instructing collaborative robots with behavior trees and vision. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 564–571.

[20] Stefan Schaal, Auke Ijspeert, and Aude Billard. 2003. Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 358, 1431 (2003), 537–547.

[21] Yasaman S Sefidgar, Prerna Agarwal, and Maya Cakmak. 2017. Situated tangible robot programming. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*. IEEE, 473–482.

[22] Lanbo She, Yu Cheng, Joyce Y Chai, Yunyi Jia, Shaohua Yang, and Ning Xi. 2014. Teaching robots new actions through natural language instructions. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 868–873.

[23] Lanbo She, Shaohua Yang, Yu Cheng, Yunyi Jia, Joyce Chai, and Ning Xi. 2014. Back to the blocks world: Learning new actions through situated human-robot dialogue. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. 89–97.

[24] Maj Stenmark and Pierre Nugues. 2013. Natural language programming of industrial robots.. In *ISR*. Citeseer, 1–5.

[25] Yeping Wang, Gopika Ajaykumar, and Chien-Ming Huang. 2020. See What I See: Enabling User-Centric Robotic Assistance Using First-Person Demonstrationss. In *Proceedings of the 15th annual ACM/IEEE international conference on Human-Robot Interaction*. ACM.

[26] Tianhe Yu, Chelsea Finn, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. 2018. One-shot imitation from observing humans via domain-adaptive meta-learning. *arXiv preprint arXiv:1802.01557* (2018).

[27] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. 2018. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1–8.