Contextual Programming of Collaborative Robots

Chien-Ming Huang

Johns Hopkins University, Baltimore, MD 21218, USA
cmhuang@cs.jhu.edu
http://www.cs.jhu.edu/~cmhuang

Abstract. Collaborative robots are envisioned to assist people in an increasing range of domains, from manufacturing to home care; however, due to the variable nature of these fields, such robots will inevitably face unfamiliar situations and unforeseen task requirements, and must be able to interact with users who possess diverse skill sets, backgrounds, and needs. Presently, robust, autonomous solutions for appropriately handling these vast possibilities and uncertainties are unattainable. *End-user robot programming* offers an alternative approach that lets end users provide task specifications and author robot skills to meet their own specific contextual constraints and custom task needs. Contextual information—such as task objects, environmental landmarks, and user preferences—is essential in realizing desirable, flexible, and reliable robot programs. However, most robot programming systems at present do not afford intuitive ways of specifying contextual information. In this paper, we draw on our prior work to illustrate the barriers to end-user robot programming when using a state-of-the-art programming interface. We then present two case studies that explore new approaches to providing a robot system with contextual information about the user, task, and environment, and how these methods can help improve task performance and user experience. We discuss our findings and future directions for building effective end-user programming tools to bring robotic assistance closer to everyday users.

Keywords: Human-Robot Interaction · Collaborative Robotics · End-User Robot Programming · Programming by Demonstration.

1 Introduction

Collaborative robots hold great promise for augmenting human capabilities and providing assistance in various domains—from manufacturing to home care to search and rescue missions. The successful integration of robotic assistance into these types of task domains requires that robots work with people with diverse skill sets and needs, perform a variety of manipulation tasks, and operate in complex human environments that typically involve established physical infrastructures—all while being customizable to meet specific task requirements and contextual constraints. At present, developing a robot that can au-

tonomously and appropriately respond to every possible situation is an intractable problem. *End-user robot programming* offers an alternative approach that empowers end users to harness the full potential of programmable robots and facilitates the integration of collaborative robots into human environments in the short term.

Programming by demonstration (PbD)—also known as learning from demonstration (LfD) [6,7,9,30]—has emerged as a promising framework of end-user robot programming. It explores various abstraction methods, supporting tools, and authoring interfaces for robot programming, aiming to teach robot skills and adapt them to a variety of task configurations (e.g., [4,10,23,37,42]). PbD research has investigated how people may train robots in new skills through kinesthetic teaching [3,14], natural instructions with verbal and non-verbal behaviors [19,21,24,33,34,36], vision-based human demonstrations [10,42], visual programming [5,12,17,26], embodied enactment [27], and teleoperative demonstrations in virtual reality [43] or via motion capture [8,16,31]. While still in its infancy, research on PbD has established its potential in training task concepts (e.g., [22]) and manipulation skills (e.g., [2]). Among the variations of PbD, kinesthetic teaching allows end users to program robots through direct action demonstration, while visual programming offers accessible interfaces to allow users to focus on programming the logic of a robot task. A hybrid use of kinesthetic teaching and visual programming has become a common implementation of robot programming by demonstration in both industry (e.g., Universal Robots, Elephant Robotics) and academia (e.g., [17,26]).

Although this hybrid approach has certainly lowered the barriers to entry for end-user robot programming, most people still have difficulties providing effective and efficient demonstrations. As we illustrate in our informative study [1] (Section 2), everyday users generally lack an accurate mental model of how high degrees-of-freedom robots operate (e.g., what their capabilities and limitations are) and are typically provided with unproductive programming interfaces to begin with. In particular, in most current PbD systems, there is no straightforward way of specifying pertinent contextual information—such as task-relevant objects, environmental constraints, and user preferences—to ensure that the resulting robot programs are correct and productive. We note that contextual information is critically important for collaborative robots, as they have direct access to and influence upon their surrounding environments and users. The necessary consideration of physical context sets PbD apart from end-user programming for typical computer programs [39], which usually do not involve real-time physical interactivity.

In this paper, we argue that straightforward methods of encoding contextual information via PbD are integral to an end user's production of effective and efficient robot programs that meet their task constraints and needs. We first describe an informative study that highlights key barriers to productive end-user robot programming [1]. We then present two case studies that illustrate new modalities for providing contextual information in robot programs and address a number of the previously identified barriers. The first case study focuses on

enabling users to directly specify environmental contexts in situ using gestures and illustration tools (***programming by situated illustration***) [11], while the second study uses egocentric vision to capture user preferences for performing complex manipulative tasks (***programming by first-person demonstration***) [38]. We demonstrate that these modalities improve task performance and user experience for repetitive task specification [11] and dynamic human-robot collaboration [38]. We conclude this paper with a discussion of the resulting implications for designing effective end-user robot programming systems.

2 Barriers to End-User Robot Programming

2.1 Informative Study

To explore potential barriers to programming a collaborative robot, we conducted an informative study in which participants were asked to program a UR-5 robotic arm to complete four common manipulation tasks, including stacking, hanging, and pouring task objects such as blocks, cups, and towels [1] (Fig. 1). Participants were instructed to use kinesthetic teaching to program the robot and were free to use continuous trajectories, waypoints, or a combination of the two during the programming process [2]. After giving their informed consent, each participant was provided with a tutorial on how to use the state-of-the-art programming interface for the UR-5: the PolyScope on the teach pendant that supplements the robot. They were then given an opportunity to practice by programming a pick-and-place task. After completing the practice task, participants completed four manipulation tasks using the programming interface. They could correct the program as many times as they wanted until they were happy with the task result. Regardless of their progress, each participant was stopped fifteen minutes before the hour to complete an open-ended interview and a demographics questionnaire. The study was video-recorded and participants received $10 USD for study completion. Eight participants (6 females and 2 males) were recruited for the study, with ages ranging from 19 to 57 ($M = 31.13, SD = 16.22$). These participants came from various backgrounds and disciplines and had varying degrees of experience in programming and technology.

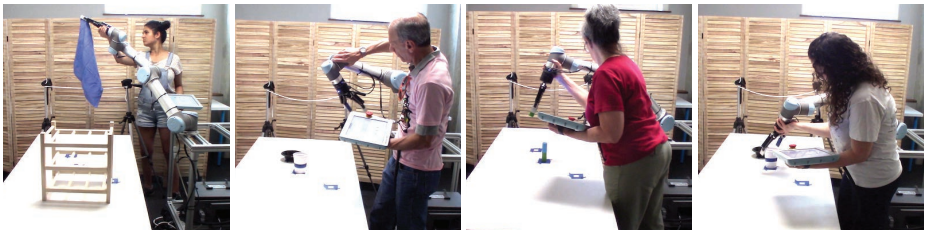

Fig. 1. Our informative study revealed that users face various difficulties in task planning and action demonstration when programming a collaborative robot [1].

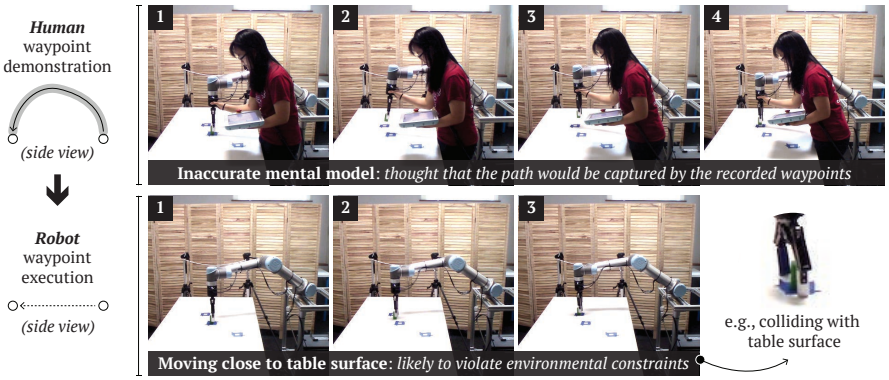

Fig. 2. A common error in robot programming by demonstration. The user mistakenly thought the robot would record trajectories between waypoints, which resulted in task failures and collisions.

2.2 Key Observations and Findings

Overall, we observed a number of user frustrations and challenges in specifying and editing robot programs, and found that participants generally did not have an accurate mental model of the robot's capabilities and limitations; a complete, accurate mental model allows users to learn about and interact with a system more effectively [18,41] and to predict and explain their interactions with that system [25]. In particular, our observations of the demonstration performances and interviews with the participants revealed these common difficulties in action demonstration during robot programming:

- *Trouble moving the robot into a desired configuration.* The unfamiliar kinematics of the robot hindered the participants' ability to provide direct demonstrations of task-relevant actions (Fig. 1).
- *Fatigue during kinesthetic teaching.* Three participants directly mentioned how difficult programming the robot could be when using kinesthetic teaching, while other participants conveyed this indirectly through expression of fatigue while programming or when they showed reluctance in—or even decided to forego—correcting their flawed programs.
- *Demonstrations resulting in suboptimal or unsmoothed action trajectories.* Participant fatigue during kinesthetic teaching resulted in extraneous movements and lag time (where no movement occurred) in most of the participants' programs that used continuous recording.
- *Confusing waypoint specification with path recording.* One participant thought that trajectories between waypoints were recorded and that they represented actual paths that the robot would later trace through (Fig. 2). Such inadequate understanding often led to task failures or unexpected collisions with obstacles.
- *High cognitive demand when tracking programming steps.* Two participants forgot to record a path during one of the tasks, illustrating how a complex

programming process can lead to high cognitive load—which consequently reduces the quality of the resulting user-specified robot programs.
- *Disconnect between specified program logics and contextual landmarks.* Some participants found the tab structure of the PolyScope interface confusing, since it separates robot action selection from robot action parameter specification. One participant suggested that it would be more intuitive for both command selection and program editing to be located in one central hub. Additionally, we observed participants make frequent switches between action demonstration (i.e., kinesthetic teaching) and program specification (i.e., using the visual programming interface); this disconnect may have contributed to the participants' insufficent mental models of their robot programs.

Overall, the user experiences that we observed reveal there is still significant room for improvement in current end-user programming interfaces, especially with respect to increasing usability and maximizing user productivity. Based on our observations and the feedback we received from participants, we identified several design opportunities for end-user programming interfaces; in particular, we sought to enable and streamline the process of providing contextual information in robot programming.

Next, we present two novel frameworks of robot programming: (1) programming by situated illustration—a framework that allows users to directly specify environmental contexts and task actions in situ, thus overcoming the disconnect between program specification and action demonstration and thereby reducing users' cognitive load—and (2) programming by first-person demonstration—a framework that allows users to demonstrate tasks from their own perspectives without worrying about the complex, unfamiliar kinematics of the robot.

3 Programming by Situated Illustration

Common PbD systems used in industrial settings (e.g., the PolyScope interface) involve little perception of the environment and rely primarily upon replaying demonstrated action trajectories or tracing through recorded key waypoints. This limited perception constrains the generalizability and scalability of PbD; however, perception of task-relevant objects and the environment usually requires specialized processes for training objects for visual recognition and robot manipulation (e.g., [17]), which consequently creates additional technical barriers for end users. To address these limitations, we explore the framework of *programming by situated illustration*, which seeks to facilitate programming robots to operate within flexible environments, in which objects and environmental landmarks are not predefined; this type of environment lies in direct contrast to automation scenarios with predefined procedures and task configurations.

Situated tangible robot programming [32] is an example of this framework; it allows users to specify task-relevant objects and provide annotation in the environment through the use of tangible blocks. However, tangible programming does not support 3D specification (e.g., labeling the height of an object), and

Fig. 3. PATI supports the augmented and situated specification of robot tasks, allowing users to teach their robots new tasks by directly referencing and interacting with the environment via virtual tools (e.g., shape tools) and common touch screen gestures, such as pinch, tap, and drag-and-drop. This figure illustrates a sequence of specifications for a simple pick-and-place task.

its expressiveness is limited by the number of blocks used. Below, we describe an alternative implementation of situated programming enabled by projection-based augmentation.[1]

3.1 System Prototype

We developed a Projection-based Augmented Table-top Interface (PATI) for robot programming[2] (Fig. 3)—a prototype system designed to streamline the processes of visual tracking, object referencing, and task specification in robot

[1] Please refer to [11] for a detailed description of our implementation and its evaluation.

[2] This project is available at https://intuitivecomputing.github.io/PATI.

programming by enabling users to directly reference and annotate the environment through gestural illustration. The PATI system allows users to program a robot manipulator directly on a tabletop surface through an intuitive tangible interface. Our current implementation involves the use of a UR5 robot manipulator, a top-down projector, and a Kinect2 RGB-D camera mounted on the ceiling. Aside from the hardware setup, the PATI system consists of four software modules: *Visual Perception*, *Tangible User Interface (TUI)*, *Program Synthesis*, and *Robot Control*. The modules of Visual Perception, Program Synthesis, and Robot Control are implemented in the Robot Operating System (ROS) [29], while the TUI is implemented in Unity. The Visual Perception module detects and tracks objects in the environment—as well as a user's touch input—through the use of depth data and RGB color images from the Kinect2 camera. User input is then sent to the TUI module, in which different types of gestures are recognized. Once the user completes their program specification via the TUI, the Program Synthesis module translates the user's task-level specifications into a set of robot commands, including gripper actions and the waypoint specifications of intended trajectories. The commands are then forwarded to the Robot Control module, which subsequently generates corresponding motion plans for robot execution.

3.2 Empirical Evaluation

Below, we summarize a user study conducted to assess the effectiveness of PATI. Our evaluation focused on simple manipulation tasks (such as pick-and-place) that are fundamental to a variety of common functions which robots have been envisioned to assist people in performing. Our central hypotheses were that the PATI system would help participants achieve greater *task performance* and that it would offer higher *usability* to participants than the built-in PbD method for the UR5. We designed a within-participants study with two experimental conditions. In the control condition, participants used the UR5 programming interface (PolyScope), representing a state-of-the-art method for robot programming in industry. In the experimental condition, participants used our PATI system. We counterbalanced the order of the conditions presented to the participants. The study took approximately one hour to complete and participants were compensated with $10 USD.

We used a combination of objective and subjective measures to assess task performance and usability of the PATI and PolyScope interfaces. For task performance, we measured *task time* (how long it took to complete the task), *number of task mistakes*, and *number of questions asked*. For usability, we measured *practice time* (time users needed to be familiar with and comfortable using a new interface) and *task load* (an adapted version of the NASA TLX [13]).

We recruited 17 participants (9 females and 8 males) on a college campus for this study. They were aged 23.48 years on average ($SD = 4.81$) and reported having familiarity with robots ($M = 4.00$, $SD = 1.97$) and programming ($M = 4.18$, $SD = 1.88$) on 1-to-7 rating scales, with 7 being the most familiar. The participants also had a variety of educational backgrounds, including engineering, psychology, writing, and medicine.

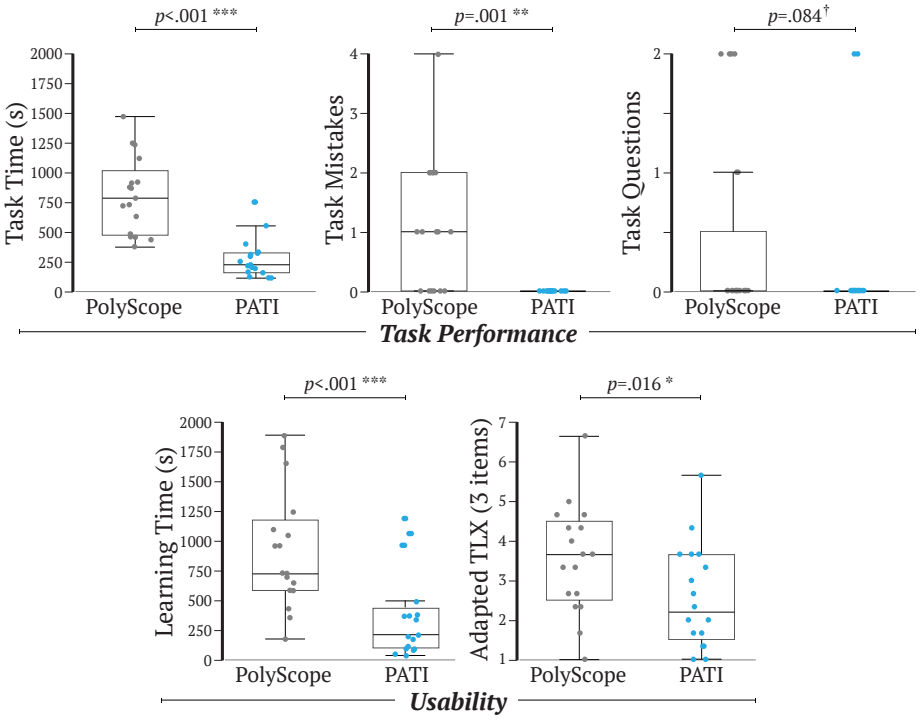

Fig. 4. Box and whisker plots of our data on the objective measures of task performance and the subjective measures of usability.

3.3 Main Findings

Our main findings are summarized in Figure 4. We used non-parametric Wilcoxon signed-rank tests in accordance with the distribution of the analyzed data. With regard to task performance, we found that participants needed significantly less time to complete the experimental task ($Z = 3.62, p < .001$) when using the PATI system ($M = 276.28, SD = 167.94$) than they did when using the PolyScope interface ($M = 808.37, SD = 324.02$); the participants did not make any mistakes when using PATI ($M = 0.00, SD = 0.00$) while they made more than one mistake on average when using PolyScope ($M = 1.12, SD = 1.11$), $Z = 3.23, p = .001$; and the participants asked the experimenter marginally fewer questions ($Z = 1.73, p = .084$) during the task when using PATI ($M = 0.12, SD = 0.49$) than they did when using PolyScope ($M = 0.35, SD = 0.70$).

With regard to usability, we found that participants needed significantly less time before they were familiar with the programming interface and felt ready for the experimental task sooner ($Z = 3.34, p < .001$) when using PATI ($M = 384.21, SD = 366.08$) than they did when using the PolyScope interface ($M = 919.83, SD = 495.77$); we also found that participants experienced less

task load ($Z = 2.41, p = .016$) when using our system ($M = 2.16, SD = 1.31$) than they did in the control condition ($M = 3.55, SD = 1.38$).

Overall, our results indicate the potential of *programming by situated illustration*; in particular, participants were able to learn how to use the system in a shorter period of time and achieved greater task efficiency with less task load when using PATI than they did when using a state-of-the-art PbD system.

4 Programming by First-Person Demonstration

While kinesthetic teaching allows users to directly provide action demonstrations by maneuvering a robot, our informative study revealed that most people have difficulty providing desirable demonstrations due to their unfamiliarity with robot kinematics, which are admittedly unnatural to human users. Arguably, the most natural method of human task demonstration is simply performing the task. Prior research on robot *learning by watching* has explored how robots may learn new tasks by observing how humans perform the tasks either in situ [20]

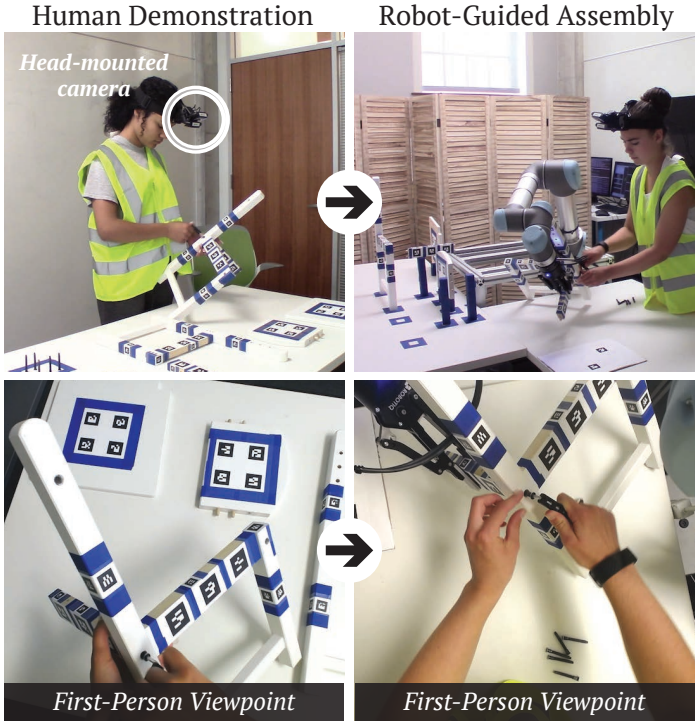

Fig. 5. We explore how first-person demonstrations capture natural behavioral preferences for task performance and how they can be utilized to enable user-centric robotic assistance in human-robot collaborative assembly tasks.

or through videos [40]. Contrary to prior research, the framework of *programming by first-person demonstration* explores methods of enabling robots to learn new skills from a human demonstrator's own first-person perspective (Fig. 5). We note that this type of first-person demonstration is different from showing a task to a learner via a teaching process, which requires consideration and estimation of the learner's perspective during the teaching of the task; the unique aspect of our approach is allowing a robot learner to channel directly into a human teacher's perspective. First-person vision provides a dynamic viewpoint representing the region of attention and offers the human teacher's perspective on task performance (e.g., how the hands interact with objects of interest). We argue that first-person demonstrations naturally capture human preferences for task performance and encapsulate unique, task-relevant contextual information critical to training robot manipulation skills. Below, we describe our exploration of first-person demonstration as an intuitive method of robot programming.[3]

4.1 System Prototype

We developed a perception system capable of parsing a first-person demonstration into a First-person Experience-based Assembly Tree (FEAsT) model (Fig. 6), which hierarchically stores task and contextual information that a collaborative robot can then use to assist a human user. A FEAsT model encodes (1) the sequential procedure of the task, (2) spatial configurations symbolizing how different task objects fit together, and (3) first-person viewpoints corresponding to the various task actions (i.e., connecting and screwing). A unique aspect of the FEAsT model is that it stores key task viewpoints from the original human demonstration; this additional degree of contextual information captures which spatial configuration a human finds most natural to view a part or set of parts from during an assembly task; we argue that this viewpoint information is key to enabling user-centric robotic assistance. In addition to the perception system, we developed an autonomous robot system that can utilize a FEAsT model to assist users in a collaborative task. Our implementation was grounded in the context of assembling a kit-furniture chair.

We first conducted a data collection study to obtain natural first-person demonstrations of assembling an IKEA children's chair. This study involved 12 participants (7 females, 5 males), aged 18 to 46 ($M = 28.58$, $SD = 10.55$), with various educational backgrounds, including engineering, education, finance, and neuroscience. On average, the assembly task demonstration length was 137.96 seconds ($SD = 26.52\,s$). Overall, participants were able to complete the assembly task and their resulting task demonstrations were of sufficient quality to enable a collaborative robot to provide user-centric assistance. We chose one of these demonstrations to populate the FEAsT model that was used in our evaluation study, described below.

[3] Please refer to [38] for a detailed description of our implementation and its evaluation.

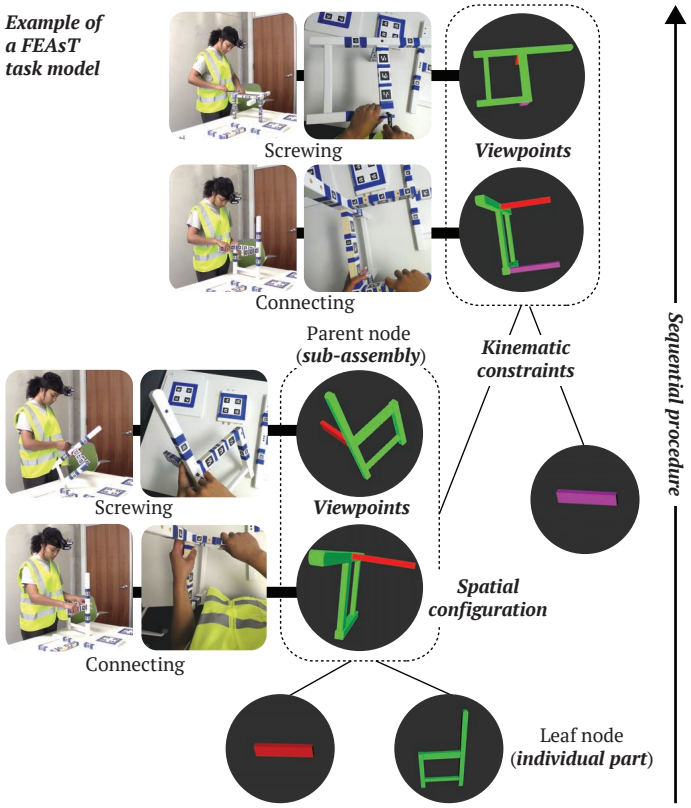

Fig. 6. The hierarchical structure of a FEAsT: leaf nodes are *parts*; parent nodes are *sub-assemblies*; edges are *kinematic constraints* between a parent node and its child nodes; and sub-assembly nodes contain relevant *first-person viewpoints.*

4.2 Empirical Evaluation

To explore the importance of egocentric views of task performance in authoring effective collaborative robot programs, our evaluation focused on assessing how user-centric robotic assistance may influence a user's behavior during a collaboration and affect their perception of their robot partner. We hypothesized that robotic assembly assistance based on a first-person demonstration would lead to better user experience than robotic assistance that did not consider first-person viewpoints.

Our user evaluation followed a within-participants design, with each participant working with the robot in either a *standard assistance* or *user-centric assistance* condition. We counterbalanced the order in which the conditions were presented to participants. In the *standard* condition, the robot fetched task parts and presented them to the participants matching the viewpoint shown on the cover page of the official IKEA assembly manual. In the *user-centric* condition,

the robot fetched parts and presented them to users matching the first-person viewpoint extracted from one of our previously collected human demonstrations.

To assess participants' performance and user experience, we measured the number of times the participants used their non-dominant hands or switched between hands when performing screwing actions. We also measured the number of times participants dropped the screwdriver during the task *(tool drops)*. In addition, we measured participants' *perceived teamwork*, or how cooperative and how good of a teammate participants perceived the robot to be, and *perceived consideration*, or how considerate the robot was of the user's actions, tasks, and comfort during the interaction.

Our data analysis included a total of 20 participants (10 females, 10 males), aged 18 to 65 ($M = 27.70$, $SD = 13.86$), who had a variety of educational backgrounds, including engineering, applied math, biology, music, and international studies. Nineteen out of twenty participants reported themselves as right-handed, with the remaining participant being left-handed. Each participant was compensated with $5 USD for their participation in the study, which lasted approximately 30 minutes.

4.3 Main Findings

Fig. 7 summarizes our results based on a one-way repeated-measures analysis of variance (ANOVA), in which the type of assistance—either standard or user-centric—was set as a fixed effect, and the participant was set as a random effect.

Our results revealed that participants were more likely to use their non-dominant hand or switch hands during the assembly task ($F(1, 38) = 37.79, p < .001, \eta_p^2 = 0.499$) when working with the robot providing standard assistance ($M = 2.65, SD = 0.81$) than when working with the robot providing user-centric assistance ($M = 0.75, SD = 1.12$). We observed that the fixed—and sometimes awkward—presentation of the chair parts in the standard condition prompted participants to engage in unproductive hand use. In contrast, participants were more often able to use their dominant hand to carry out task actions when provided with user-centric assistance. Moreover, we observed that participants tended to drop the screwdriver more frequently ($F(1, 38) = 3.73, p = .061, \eta_p^2 = 0.089$) when provided with standard assistance ($M = 0.65, SD = 0.81$) than when provided with user-centric assistance ($M = 0.25, SD = 0.44$). While we were unable to identify a direct relationship between non-dominant hand use and tool drops, we speculate that these two behaviors are possibly associated and might have influenced each other.

Participants reported significantly greater perceived teamwork ($F(1, 38) = 4.88, p = .033, \eta_p^2 = 0.114$) with the user-centric robot ($M = 6.09, SD = 0.69$) than with the standard robot ($M = 5.42, SD = 1.15$). They also perceived the user-centric robot ($M = 5.83, SD = 0.80$) to be more considerate (F(1,38) = 8.27, $p = .007, \eta_p^2 = 0.179$) in terms of accommodating how they preferred to perform the task than compared to the standard robot ($M = 4.86, SD = 1.28$); for example, P15 commented, *"[The user-centric robot] could actually feel when I was uncomfortable—like I couldn't actually put the pieces together in a really*

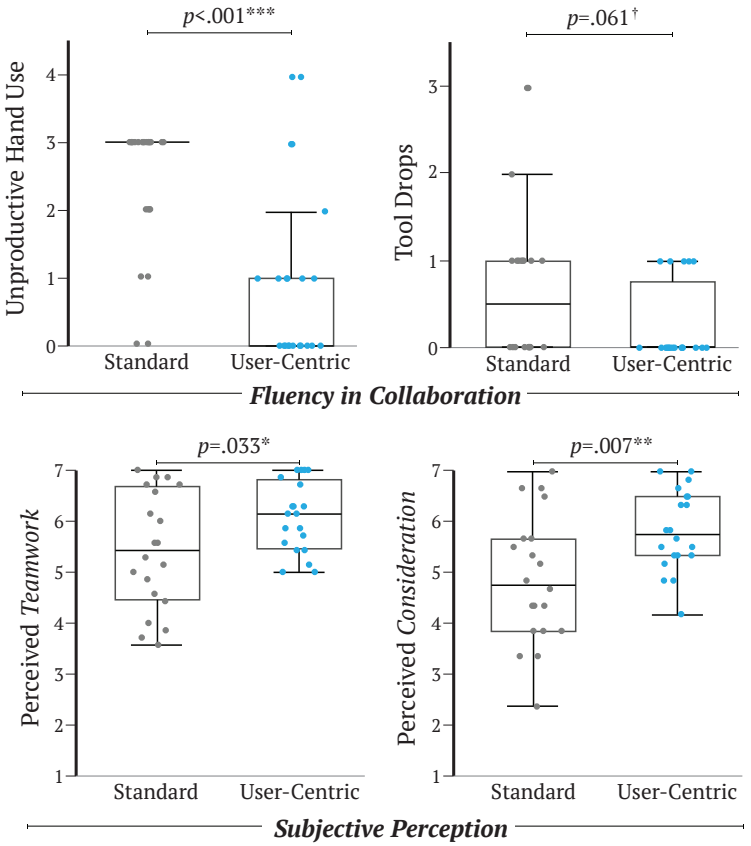

Fig. 7. Box and whisker plots of our data on the behavioral measures of unproductive behavior and the subjective measures of user experience.

comfortable way, so he just kind of helped me do that, and it was really nice because I felt like we were working together." Additionally, participants agreed that the user-centric robotic assistance allowed them to align pieces and screw them together more easily than in the standard assistance condition, as described by P2: *"The [user-centric] robot did a slight, nice movement into the position where I was kind of facing it, so it was easier for me to screw."*

Overall, our results indicate the potential of *programming by first-person demonstration*; in particular, we demonstrate how first-person demonstration may be used to generate user-centric assistance in human-robot collaborative assembly. Our results show that, when working with the robot offering user-centric assistance, participants felt that the robot was a more considerate collaborator and were less likely to engage in unproductive behavior.

5 Discussion

Most robot programming systems in industrial settings have little contextual awareness of their surroundings and users, and simply replay human demonstrations (e.g., action trajectories). While effective in certain settings and applications that require repetitive task performance, this limited contextual awareness restricts the desirability, flexibility, and reliability of the resulting specified robot programs. We present two new modalities of authoring demonstrations that explicitly and implicitly represent contextual information regarding task objects, environmental landmarks, and user preferences. In particular, we explored a new way to streamline the specification of robot actions and environmental contexts (***programming by situated illustration***) and a new method of capturing subtle user preferences for task actions via the egocentric perspective (***programming by first-person demonstration***). Programming by situated illustration improves upon the current hybrid implementation of PbD systems, while programming by first-person demonstration allows for additional user context (e.g., how a user would like to perform a task), which is critically important in human-robot collaboration. Through empirical evaluations with naive participants, we illustrated how these two modalities may enhance task performance and user experience.

This research points to future directions in enriching the contextual programming of collaborative robots. First, techniques of computer vision and natural language processing should be leveraged to capture the rich contexts embedded in interactive demonstrations and the physical world. Our case studies, along with other prior research (e.g., [17,26]), illustrate how vision-augmented programming can aid end users in efficiently providing program specifications and generating flexible robot programs; for instance, the system may recognize a user's handedness and adjust the parameters of a demonstration accordingly. Second, as programming media mature and supporting tools (e.g., visual sensing) become more widely available, the scale and complexity of robot programs will grow exponentially. Furthermore, the situated interactivity that physically embodied robots afford adds an additional layer of complexity to their programs; the increasing complexity of robot programs necessitates that program outputs accurately reflect program specifications (e.g., [28]). Assured robot programs will help end users build trust in and adopt emergent robotic technologies over time. Third, as analogous to common metrics in human-robot interaction (e.g., [15,35]), in order to make productive progress in advancing end-user robot programming, we need to develop a common set of evaluation metrics for robot programming. Prior research has used various measures of efficiency in programming and debugging (e.g., [5]), task load (e.g., [11]), program accuracy, and custom scales of user experience (e.g., [38]); however, it is less clear how to quantify the generalizability, quality, and expressiveness of a program. Reliable, agreed-upon scales to probe different aspects of user experience will also be instrumental in catalyzing the advancement of end-user robot programming.

In addition to the research directions discussed above, future research on end-user robot programming should involve domain-specific end users, such as

production workers and caregivers, all throughout the stages of design, development, and evaluation, following the principles of user-centered design. Working closely with real-life domain users will help develop more effective systems for potential users in those fields and ensure that the resulting systems can be more easily integrated into existing workflows and adopted by the intended users. In a similar vein, we must deepen our understanding of domain contexts and incorporate them when designing domain-specific programming tools. For instance, an ethnographic study on how production workers currently use state-of-the-art programming interfaces would provide invaluable insight into the challenges and barriers they face, contextualized in real-world settings and constraints.

6 Conclusion

We envision that robotic assistance will positively transform the future of work, care, and living, and believe that empowering end users to be able to intuitively author robot programs will improve the accessibility of robotic assistance while better meeting domain users' needs and constraints. Through empirical studies, we have determined that there still remains much room for improvement in current end-user programming interfaces with respect to increasing usability and maximizing user productivity, and that contextual information is key in creating effective task specifications and user-friendly robotic assistance. Advancing the contextual programming of collaborative robots will require further research on human-computer interaction, artificial intelligence, robotics, formal methods, and other related fields. Together, these efforts will help democratize robotic assistance for use by everyday people.

Acknowledgements

I would like to thank Gopika Ajaykumar, Yuxiang Gao, and Yeping Wang for helping with system implementation and user evaluation, and Jaimie Patterson for proofreading this paper. This work is partially supported by the National Science Foundation award #1840088.

References

1. Ajaykumar, G., Huang, C.M.: User needs and design opportunities in end-user robot programming. In: Proceedings of the 15th International Conference on Human-Robot Interaction (HRI), Late-Breaking Report. ACM (2020)
2. Akgun, B., Cakmak, M., Jiang, K., Thomaz, A.L.: Keyframe-based learning from demonstration. International Journal of Social Robotics **4**(4), 343–355 (2012)
3. Akgun, B., Cakmak, M., Yoo, J.W., Thomaz, A.L.: 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. pp. 391–398. ACM (2012)

4. Aleotti, J., Caselli, S.: Robust trajectory learning and approximation for robot programming by demonstration. Robotics and Autonomous Systems **54**(5), 409–413 (2006)
5. Alexandrova, S., Tatlock, Z., Cakmak, M.: Roboflow: A flow-based visual programming language for mobile manipulation tasks. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on. pp. 5537–5544. IEEE (2015)
6. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robotics and autonomous systems **57**(5), 469–483 (2009)
7. Billard, A., Calinon, S., Dillmann, R., Schaal, S.: Robot programming by demonstration. In: Springer handbook of robotics, pp. 1371–1394. Springer (2008)
8. Calinon, S., Billard, A.G.: What is the teachers role in robot programming by demonstration?: Toward benchmarks for improved learning. Interaction Studies **8**(3), 441–464 (2007)
9. Chernova, S., Thomaz, A.L.: Robot learning from human teachers. Synthesis Lectures on Artificial Intelligence and Machine Learning **8**(3), 1–121 (2014)
10. Finn, C., Yu, T., Zhang, T., Abbeel, P., Levine, S.: One-shot visual imitation learning via meta-learning. arXiv preprint arXiv:1709.04905 (2017)
11. Gao, Y., Huang, C.M.: Pati: a projection-based augmented table-top interface for robot programming. In: Proceedings of the 24th International Conference on Intelligent User Interfaces. pp. 345–355. ACM (2019)
12. Glas, D.F., Kanda, T., Ishiguro, H.: Human-robot interaction design using interaction composer eight years of lessons learned. In: Human-Robot Interaction (HRI), 2016 11th ACM/IEEE International Conference on. pp. 303–310. IEEE (2016)
13. Hart, S.G.: Nasa-task load index (nasa-tlx); 20 years later. In: Proceedings of the human factors and ergonomics society annual meeting. vol. 50, pp. 904–908. Sage Publications Sage CA: Los Angeles, CA (2006)
14. Hersch, M., Guenter, F., Calinon, S., Billard, A.: Dynamical system modulation for robot learning via kinesthetic demonstrations. IEEE Transactions on Robotics **24**(6), 1463–1467 (2008)
15. Hoffman, G.: Evaluating fluency in human–robot collaboration. IEEE Transactions on Human-Machine Systems **49**(3), 209–218 (2019)
16. Hsiao, K., Lozano-Perez, T.: Imitation learning of whole-body grasps. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5657–5662. IEEE (2006)
17. Huang, J., Cakmak, M.: Code3: A system for end-to-end programming of mobile manipulator robots for novices and experts. In: Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction. pp. 453–462. ACM (2017)
18. Kieras, D.E., Bovair, S.: The role of a mental model in learning to operate a device. Cognitive Science **8**(3), 255–273 (1984)
19. Kollar, T., Tellex, S., Roy, D., Roy, N.: Toward understanding natural language directions. In: Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction. pp. 259–266. IEEE Press (2010)
20. Kuniyoshi, Y., Inaba, M., Inoue, H.: Learning by watching: Extracting reusable task knowledge from visual observation of human performance. IEEE transactions on robotics and automation **10**(6), 799–822 (1994)
21. Lauria, S., Bugmann, G., Kyriacou, T., Klein, E.: Mobile robot programming using natural language. Robotics and Autonomous Systems **38**(3-4), 171–181 (2002)
22. Lázaro-Gredilla, M., Lin, D., Guntupalli, J.S., George, D.: Beyond imitation: Zero-shot task transfer on robots by learning concepts as cognitive programs. arXiv preprint arXiv:1812.02788 (2018)

23. Lee, A.X., Lu, H., Gupta, A., Levine, S., Abbeel, P.: Learning force-based manipulation of deformable objects from multiple demonstrations. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). pp. 177–184. IEEE (2015)
24. Matuszek, C., Bo, L., Zettlemoyer, L., Fox, D.: Learning from unscripted deictic gesture and language for human-robot interactions. In: AAAI. pp. 2556–2563 (2014)
25. Norman, D.A.: Some observations on mental models. In: Mental models, pp. 15–22. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc (1983)
26. Paxton, C., Hundt, A., Jonathan, F., Guerin, K., Hager, G.D.: Costar: Instructing collaborative robots with behavior trees and vision. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 564–571. IEEE (2017)
27. Porfirio, D., Fisher, E., Sauppé, A., Albarghouthi, A., Mutlu, B.: Bodystorming human-robot interactions. In: Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology. pp. 479–491 (2019)
28. Porfirio, D., Sauppé, A., Albarghouthi, A., Mutlu, B.: Authoring and verifying human-robot interactions. In: The 31st Annual ACM Symposium on User Interface Software and Technology. pp. 75–86. ACM (2018)
29. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: Ros: an open-source robot operating system. In: ICRA workshop on open source software. vol. 3, p. 5. Kobe, Japan (2009)
30. Ravichandar, H., Polydoros, A.S., Chernova, S., Billard, A.: Recent advances in robot learning from demonstration. Annual Review of Control, Robotics, and Autonomous Systems **3** (2020)
31. Schaal, S., Ijspeert, A., Billard, A.: Computational approaches to motor learning by imitation. Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences **358**(1431), 537–547 (2003)
32. Sefidgar, Y.S., Agarwal, P., Cakmak, M.: Situated tangible robot programming. In: Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction. pp. 473–482. ACM (2017)
33. She, L., Cheng, Y., Chai, J.Y., Jia, Y., Yang, S., Xi, N.: Teaching robots new actions through natural language instructions. In: Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on. pp. 868–873. IEEE (2014)
34. She, L., Yang, S., Cheng, Y., Jia, Y., Chai, J., Xi, N.: 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). pp. 89–97 (2014)
35. Steinfeld, A., Fong, T., Kaber, D., Lewis, M., Scholtz, J., Schultz, A., Goodrich, M.: Common metrics for human-robot interaction. In: Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction. pp. 33–40 (2006)
36. Stenmark, M., Nugues, P.: Natural language programming of industrial robots. In: ISR. pp. 1–5. Citeseer (2013)
37. Sylvain, C.: Robot programming by demonstration: A probabilistic approach (2009)
38. Wang, Y., Ajaykumar, G., Huang, C.M.: See what i see: Enabling user-centric robotic assistance using first-person demonstrations. In: Proceedings of the 15th International Conference on Human-Robot Interaction (HRI). ACM (2020)
39. Weintrop, D.: Block-based programming in computer science education. Commun. ACM **62**(8), 2225 (Jul 2019). https://doi.org/10.1145/3341221

40. Yang, Y., Li, Y., Fermüller, C., Aloimonos, Y.: Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web. In: AAAI. pp. 3686–3693 (2015)
41. Young, R.M.: The machine inside the machine: Users' models of pocket calculators. International Journal of Man-Machine Studies **15**(1), 51–85 (1981)
42. Yu, T., Finn, C., Xie, A., Dasari, S., Zhang, T., Abbeel, P., Levine, S.: One-shot imitation from observing humans via domain-adaptive meta-learning. arXiv preprint arXiv:1802.01557 (2018)
43. Zhang, T., McCarthy, Z., Jow, O., Lee, D., Goldberg, K., Abbeel, P.: Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. 2018 IEEE International Conference on on Robotics and Automation (ICRA) (2018)