Introducing Novice Operators to Collaborative Robots: A Hands-On Approach for Learning and Training

Andreas Kornmaaler Hansen, Valeria Villani, *Member, IEEE*,
Andrea Pupa, *Member, IEEE*, and Astrid Heidemann Lassen

***Abstract*— Collaborative robots (cobots) have seen widespread adoption in industrial applications over the last decade. Cobots can be placed outside protective cages and are generally regarded as much more intuitive and easy to program compared to larger classical industrial robots. However, despite the cobots' widespread adoption, their collaborative potential and opportunity to aid flexible production processes seem hindered by a lack of training and understanding from shop floor workers. Researchers have focused on technical solutions, which allow novice robot users to more easily train collaborative robots. However, most of this work has yet to leave research labs. Therefore, training methods are needed with the goal of transferring skills and knowledge to shop floor workers about how to program collaborative robots. We identify general basic knowledge and skills that a novice must master to program a collaborative robot. We present how to structure and facilitate cobot training based on cognitive apprenticeship and test the training framework on a total of 20 participants using a UR10e and UR3e robot. We considered two conditions: adaptive and self-regulated training. We found that the facilitation was effective in transferring knowledge and skills to novices, however, found no conclusive difference between the adaptive or self-regulated approach. The results demonstrate that, thanks to the proposed training method, both groups are able to significantly reduce task time, achieving a reduction of 40%, while maintaining the same level of performance in terms of position error.**

***Note to Practitioners*—This paper was motivated by the fact that the adoption of smaller, so-called collaborative robots is increasing within manufacturing but the potential for a single robot to be used flexibly in multiple places of a production seems unfulfilled. If more unskilled workers understood the collaborative robots and received structured training, they would be capable of programming the robots independently. This could change the current landscape of stationary collaborative robots towards more flexible robot use and thereby increase companies' internal overall equipment efficiency and competencies. To this end, we identify general skills and knowledge for programming a collaborative robot, which helps increase the transparency of what novices need to know. We show how such knowledge and skills may be facilitated in a structured training framework, which effectively transfers necessary programming knowledge and skills to novices. This framework may be applied to a wider scope of knowledge and skills as the learner progresses. The skills and knowledge that we identify are general across robot platforms, however, collaborative robot interfaces differ. Therefore, a practical limitation to the approach includes the need for a knowledgeable person on the specific collaborative robot in question in order to create training material in areas specific to that model. However, with our list of identified skills, it provides an easier starting point. We show that relatively few skills and knowledge areas can enhance a novice's programming capability.**

***Index Terms*— Robot programming, collaborative robotics, adaptive training, self-regulated training.**

I. Introduction

IN THE last decades, technological progress has significantly changed the industrial scenario, not only in terms of production processes and results but also from the occupational perspective. With the paradigm of Industry 4.0, formally started in 2011 as a German strategic initiative to elevate their manufacturing industry [1], technologies for advanced automation, interconnectivity, and flexible manufacturing have become accessible to any-sized companies [2].

Among such technologies, the availability of collaborative robots (also called cobots) has introduced intelligent agents that aid human operators, working side by side with them. They are smaller and quicker to set up than industrial robots, which helps to make smaller batch sizes achievable with quicker change-overs in production. Due to a wide range of internal and third-party external safety measures, cobots can be placed outside of safety cages (depending on their material handling) [3]. These capabilities, combined with a lower price tag compared to traditional industrial robots, have made cobots attractive to smaller-sized manufacturers [4]. It is not only the smaller-sized companies who are interested in cobots, as is seen by the general trend. From 2017-2021, global industrial

Manuscript received 26 February 2024; accepted 9 May 2024. This article was recommended for publication by Associate Editor Z. Liu and Editor L. Zhang upon evaluation of the reviewers' comments. *(Corresponding author: Valeria Villani.)*

This work involved human subjects or animals in its research. The authors confirm that all human/animal subject research procedures and protocols are exempt from review board approval.

Andreas Kornmaaler Hansen was with the Center for Industrial Production, Aalborg University, 9220 Aalborg, Denmark, and also with the Industrial Digital Transformation, University College of Northern Denmark, 9200 Aalborg, Denmark. He is now with the Department of Architecture, Design and Media Technology, Aalborg University, 9220 Aalborg, Denmark (e-mail: a.kornmaaler.hansen@outlook.com).

Valeria Villani and Andrea Pupa are with the Department of Sciences and Methods for Engineering (DISMI), University of Modena and Reggio Emilia, 41121 Reggio Emilia, Italy (e-mail: valeria.villani@unimore.it).

Astrid Heidemann Lassen is with the Center for Industrial Production, Aalborg University, 9220 Aalborg, Denmark.

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TASE.2024.3403709.

Digital Object Identifier 10.1109/TASE.2024.3403709

robot installations saw an annual increase of 5%, whereas cobots experienced a 37% increase in installations in the same period [5].

An important reason behind the surge in collaborative robots is that they can safely share spaces, tasks, and goals with humans and complement each other's capabilities: while robots are suited for physically demanding and tedious repetitive tasks, human operators can be put in charge of those tasks requiring advanced reasoning capabilities. Promisingly, human-robot collaboration (HRC) results in a team with advanced capabilities that cannot be found otherwise solely in robots or human operators. As a result, a large body of the literature has been devoted to the study of algorithms and solutions that endow collaborative robots with advanced perception and reasoning capabilities [3]. The goal is to make robots aware of the surrounding environment and humans' intentions in order to engage in a fluent collaboration between peers.

To this end, researchers have explored ways to make interaction modalities intuitive, for example, resorting to vocal interaction [6] and gestures [7]. The ultimate aim of human-robot collaborative strategies is, on the one hand, to enhance human capabilities by adding robot strengths to the workforce and, on the other hand, to leverage the way people collaborate with each other to keep collaborative work natural and intuitive. This vision is also enforced by the next-generation paradigm for the factories of the future, namely Industry 5.0. It complements and extends Industry 4.0, promoting the application of existing technologies beyond efficiency and productivity as the sole goals. In particular, Industry 5.0 places sustainability, flexibility, and worker well-being at the centre of the production process, while technology is used to provide prosperity beyond jobs and growth [8].

Nonetheless, the application of such novel approaches in current industrial scenarios is still quite limited [9]. Small and medium-sized enterprises experience big barriers to digitalisation, such as knowledge on how to navigate increased complexity [10], limited funds [11], and lack of specialised knowledge and training needed across different professional fields [12]. With respect to collaborative robotics, Michaelis et al. pointed out that most uses of cobots in automation are for simple, long-term automation applications, done as start/stop machine-tending or pick-and-place tasks (level 1 of HRC interactions [13]). In these applications, the cobot performs one part of a process, and the human is limited to starting, setting up, and/or ending the process. Collaboration is, hence, limited to pressing a few buttons to start/stop the process, load/unload the robot and basic navigation through the user interface [9]. One of the reasons for this is the lack of proper training of workers who interact with and operate manufacturing equipment. Existing training designed for collaborative robots focuses on building basic skills for trivial operations and is inadequate for applications requiring higher-level collaboration. As a result, it has been reported that low-skill workers are frequently not capable of programming, deploying, or collaborating with cobots [9] and effective training is needed for manufacturing workers [14].

To cover this gap, in this paper, we propose a training approach for unskilled workers operating collaborative robots. Specifically, we target operators who may be highly skilled in their vocational domain, but are new to the use of robots and never interacted with or programmed a collaborative robot before. The current research on human-robot collaboration concerned with advanced reasoning capabilities and more intuitive interfaces have yet to leave the research labs. As a consequence, combining the operators' experience with the advantages inherent to collaborative technologies is often unattainable, in part due to lack of effective and accessible training. Currently, additional resources for robot experts are required to implement changes to work cells, brought on by the increasing need for flexible manufacturing processes. To overcome this, our aim is to provide such workers not only with the basic knowledge required to operate robots, but also to make them develop the capability to configure them, rapidly program, and reprogram them in the presence of changes in the manufacturing process and low-volume tasks. In particular, we investigated what knowledge and skills a novice robot operator needs to master before they can safely interact and program simple robot tasks, and explicitly pinpoint key topics and focus areas. Then, building on such knowledge, we developed a hands-on training framework for robot programming that facilitates operators to practice tasks with increasing difficulty and keep track of their progress. Specifically, by robot programming, we consider the problem of writing the code that allows the robot to perform some tasks, and we focus on programming through proprietary robot language with the proprietary robot user interface.

The paper is organized as follows. Section II we discuss the related work regarding training for cobot programming and different learning approaches. In Sec. III we present the proposed contribution and the associated research questions. Then, in Sec. IV we discuss the elementary skills that are required to learn how to program robots, while in Sec. V the proposed learning framework is presented. The methodology for its experimental validation and the achieved outcomes are presented in Sec. VI and VII, respectively. Finally, Sec. VIII follows with some concluding remarks.

II. Related Work

A. Training and Learning for Cobot Programming

Despite the wide diffusion of cobots nowadays, few offer easily accessible and intuitive learning material beyond hardware specifications and manuals, thus making it a steep learning curve for non-experts. We surveyed 16 cobot manufacturers to investigate whether they offered learning material and what format it was presented in. In the supplementary material, we provide an overview and description of the publicly available learning material from the cobot companies. We focused on interactive online material designed to facilitate the learning process with specific learning outcomes, thereby going beyond structured information such as online encyclopedias, Wikis, videos, and manuals. We found that only five out of the 16 cobot manufacturers offer easily accessible, interactive, and structured learning material on how to properly engage with and program their systems via their web pages. As such, it can be difficult to learn more about cobot programming suitable for novice users. Moreover, existing literature has pointed out that text-based manuals and e-learning approaches are less effective than hands-on training [15], [16], [17], [18]. These works highlight the importance of the learning actions to comprise solid hands-on practical experimentation and promote blended learning programs, where e-learning training is accompanied by practical experimentation that considers real problem-solving [15]. Additionally, they highlight the importance of the presence of instructors acting as facilitators in the demonstration of examples and answering questions.

Generally speaking, cobots often come with teach pendants with pre-installed interfaces, making it possible for complete beginners to relatively easily start programming simple robot movements and operations. Despite this, the interfaces vary a lot between different cobot manufacturers [19], which makes it necessary to devote some time to learn how they operate. To combat this, Hoyos et al. presented a framework that enables robot programming through an interface independent of cobot manufacturer [20]. They identified robot skills and built a user interface that enables the operator to identify the specific robot skill they want to use, and quickly program the robot. Similarly, Ciontos et al. developed an app for a smartphone, which could program a diverse range of robots using a single interface [21]. This would enable a person to only learn one interface but have the ability to control a wide array of different robot platforms, reducing the complexity and flattening the learning curve if a new robot platform is introduced.

Most of the research for intuitive robot programming goes toward letting robots understand users' intentions, endowing robots with some sort of reasoning and sensing capabilities [3]. In this regard, Wang et al. presented an approach to facilitate the robot to figure out and predict human hand-over intentions to improve task efficiency in human-robot collaboration [22]. Schou et al. decomposed the problem of programming complex tasks by introducing the concept of robot tasks, skills and primitives, which can be understood as a layered structure [23]. Device primitives, such as move or open/close gripper, are simple, primitive operations, but combined together they form a skill (e.g., pick up object). This can further be elevated into tasks that contain multiple of these skills and primitives (e.g., a machine feeding task seen as the combination of: pick up object, go to the machine, place in the machine, etc.). Moreover, they added an interface in which operators could get an overview of available skills and tasks and add new skills by using kinesthetic teaching, during which the operator is guided using text and audio output from the interface [23]. Pedersen et al. continued this work and showed that such robot skills are very intuitive to use for non-expert operators as they use language they are already familiar with from their standard operating procedures [24]. Koch et al. also worked with skill-based programming and looked further into how the user interface from [23] could be improved for novice users. They decided to decrease the complexity of the interface, which meant going from high variation and flexibility of how the tasks are put together to a finite selection of predefined tasks [25]. In general terms, these works investigate how to transfer human high-level reasoning capabilities to a robot in order to program it. This line of approach is well suited in the case of complex programming tasks but does not address the need, in common industrial applications, for novice users to learn how the robot works and master basic programming skills.

Thus, a lot of work has gone into the capabilities and skills of the robot [22], [23], [26], with less attention to the capabilities and skills of the people intended to interact with robot systems. Research within learning factories has produced great results by adhering to a constructivist, problem-based learning approach, where participants get a mixture of hands-on experience and theoretical knowledge [27], [28]. A specific cobot learning framework was suggested by Mayerhofer et al. in a learning factory setup [29]. Their approach aims to adapt the learning to the individual based on their current experience level and interests. They make use of "learning nuggets", where a main learning path is created but with the possibility to veer off into related topics. Depending on individual experience level, one starts at different points on the learning path: for example, if a new robot platform is introduced and the user has prior robot arm experience, they are not presented with the basics of cobots. Balancing the amount of assistance is important, as too frequent feedback has shown detrimental to learning performance [30]. However, while advocating for the accessibility of online training material to be used whenever needed, few details were provided in [29] about the online material and what a novice cobot user needs to learn.

Within surgical robots, an adaptive training framework using virtual reality was suggested by Mariani et al. [31]. They identified some elementary skills needed for robotic surgery and also designed some complex tasks, which consisted of multiple elementary skills. This could be simulated in an offline virtual reality environment with no involvement of real patients, thus reducing risk while increasing learning outcomes. Here the authors showed that adapting the training exercises to the actual performance of the trainees showed an increase in performance and learning compared to a non-adaptive, self-regulated group. The self-regulated group chose their own exercises and did not utilise the training period as efficiently as the adaptive group, thus resulting in lower performance than the adaptive group.

Building upon existing works, our proposed framework focuses on identifying the elementary robot knowledge and skills that are needed for a novice who is learning how to program a cobot and implement them in a training framework. To this end, we were inspired by the principles of cognitive apprenticeship. The robot manufacturer we chose had a good online e-learning course for people new to robot programming. Their e-learning overlaps with how we presented our tasks to the learners. However, the main differences boil down to us demonstrating learning in a physical robot setup and relying on cognitive apprenticeship, which made sure that the tasks varied in both execution, complexity, and the physical environment.

B. Use of Cognitive Apprenticeship as a Learning Approach

In a cognitive apprenticeship learning environment, tasks are chosen to elucidate specific knowledge and techniques, which will be useful in a diverse range of settings that translate into real-world applications [32]. Typically, this entails increasing complexity slowly as the learner advances to allow them to combine elementary skills into more cohesive and meaningful tasks. Thus, tasks should be sequenced in such a way as to support the changing demands of learning [32]. This is also referred to as *scaffolding*, meaning a theoretical structure of support, which slowly gets dismantled as the learner becomes more capable. According to Collins and Kapur, cognitive apprenticeship relies on four key principles: *content, methods, sequencing* and *sociology* [32]. In the following, we have highlighted elements which we used in our work:

- *content*: Domain knowledge about key concepts, procedures, and subject matter facts.
- *methods*: A common method relies on *scaffolding*, which means that learning is structured and supported for the learner. As they progress, the support is gradually dismantled in keeping with their learning development.
- *sequencing*: Increasing complexity and diversity of the tasks presented to the learner. This may be combined with the scaffolding method.
- *sociology*: Situated learning where the learner is taught or works in the context of realistic tasks, e.g., a real collaborative robot instead of a simulated environment.

III. Outline of Proposed Contribution

In this work, we propose a learning framework for cobot programming, which covers the elementary robot knowledge and skills that are needed for a novice who is learning how to program a cobot. The learning framework is inspired by the principles of cognitive apprenticeship [32], where operators can combine theoretical learning with hands-on practice, thus being guided to become self-sufficient in using robots. Specifically, we analyse the importance of an adaptive approach to training, where the training and exercises are more closely tailored to the individual's level like the approaches in [29] and [31] suggest.

To elaborate on these considerations, we introduce two research questions:

RQ1 *What are the elementary robot knowledge and skills that a non-expert operator should master to program a collaborative robot?*

Operators need a baseline of knowledge and understanding of a robot before they will be able to understand and create robot programs. This holds true irrespective of the modality to program and interact with the robot. To answer this research question, as discussed in Sec. IV, we surveyed experts in collaborative robotics and automation to define the set of essential knowledge and skills needed to program and operate a collaborative robot.

Then, building on the outcome of **RQ1**, we focus on how to transfer such knowledge to workers.

RQ2 *How could such elementary knowledge and skills be taught to make non-expert operators self-sufficient in the use of collaborative robots?*

We used the general elementary skills identified with **RQ1** to create a training curriculum to provide hands-on teaching support in a physical environment. The framework used to structure the curriculum, discussed in Sec. V, was inspired by the concept of cognitive apprenticeship [32] drawing on the four key principles of *content, methods, sequencing* and *sociology* introduced in Sec. II-B. The rationale behind it was to expose the learner to the elementary knowledge derived from **RQ1** (*content*), and let them gradually develop the needed skills. The framework should provide guidance (i.e. providing a learning scaffold) in the execution of related tasks with increasing complexity (*methods* and *sequencing*), and let learners practice in a tangible, hands-on situated learning environment (*sociology*) [32]. Moreover, we investigated how the approach to learning may have an effect on the learning outcome when being introduced to physical robots, as shown by Mariani et al. [31]. However, Mariani et al. used a virtual reality simulation for robot-assisted surgery, which focused mostly on the proprioception and motor skills of the learner. In contrast, we demonstrate how elementary knowledge and skills needed to program a collaborative robot could be introduced in a physical training environment, where the interaction is cyber-physical as the commands received in the digital interface culminate into physical manipulations of the environment.

To this end, the designed training and learning framework was administered with two different conditions: i) an adaptive approach; and ii) a self-regulated approach. While the findings of both [29] and [31] support that an adaptive approach is preferable, this also entails more complexity e.g., the addition of a ranking or scoring system to assess the performance of the learners. Therefore it is interesting to investigate if there are any notable differences between an adaptive approach and a self-regulated approach in physical human-robot training scenarios.

IV. Elementary Skills for Robot Programming

In this section, we discuss the methodology implemented to answer **RQ1** (Sec. IV-A) and the resulting set of elementary skills and knowledge for robot programming (Sec. IV-B).

A. Methodology

1) Description of the Questionnaire: A questionnaire was created to identify the elementary skills needed for novice robot users to safely and effectively program a cobot, in response to **RQ1**. It was administered to experts within the field of collaborative robots and automation. The form listed skills and knowledge areas that the authors heuristically perceived as being essential to elementary cobot programming and generally applicable for simple operations in a manufacturing environment (e.g., a pick-and-place task). To this end, the form was separated into two fields:

1) knowledge about cobots, meant as knowledge the operator needs to know before programming the cobot;

2) cobot programming skills, meant as actual hands-on skills that the operator needs the capability to perform.

The experts were asked to look through the list and check the boxes they found relevant, and leave boxes unchecked in the fields that seemed either irrelevant, too specific, or not generally applicable across cobot platforms. They were invited to add additional skills that a novice operator is expected to master, which we may have overlooked. They had the option to add multiple topics, if necessary. In the end, they could provide additional comments or clarifications on their answers to iteratively identify the right elementary skills and knowledge. When an expert added a point, the form was iteratively updated and the expert additions were visible to the next expert.

Our questionnaire approach was reminiscent of the dissensus Delphi study approach introduced by Steinert [33]. While the Delphi approach is a standard technique to gather and appraise expert opinions, the dissensus Delphi approach aims to maximise the range of expert opinions via iterative online inputs. With reference to the dissensus Delphi approach [33], we took a less grounded approach, where we presented skills and knowledge heuristically deemed necessary for novice users but with the option for the experts to expand or dispute the proposed skills and knowledge areas iteratively.

2) Respondents to the Questionnaire: The questionnaire was administered to experts within the field of collaborative robots. A total of 19 experts (14 from academia, 5 from industry) contributed to the questionnaire from five different organisations. From academia, we targeted research fellows, PhD. students, assistant professors, associate professors and full professors who specialise in industrial manipulators, mobile robots, physical human-robot interaction in natural settings, interfaces for collaborative- and industrial robots, machine learning for cobots, cobots for surgical purposes, and dynamic motion planning. From industry, four respondents worked for a renowned collaborative robot company and one within a private research and technology institute that collaborates closely with industrial partners to solve their cobot implementation challenges. Their key areas included innovation, research and development, competence development, learning environments, operational management, artificial intelligence and learning cobots, cobot implementation, and research and business development for cobots.

B. Expert's Feedback From the Questionnaire

Table I reports the respondents' feedback on the required elementary skills. Specifically, the count in the right column in the table shows how many respondents considered each item relevant with respect to **RQ1**.

Regarding general knowledge about cobots, the experts considered being informed about the emergency stop and how to turn the system on and off as essential. Moreover, knowledge about different motion types was considered important by the greatest majority of respondents. Conversely, the less important items were deemed robot joint names, speed and acceleration and interfaces for external safety equipment. The latter might be considered as an advanced skill, relevant more to design than the use of robotic systems, and thus not that important with respect to **RQ1**.

TABLE I

Questionnaire About Elementary Skills for Robot Programming (N = 19)

	Item	Count
Knowledge about cobots	Emergency Stop	19
	Turn system on/off	17
	Motion types	11
	Tool Center Point (TCP)	11
	Degrees of Freedom (DoF)	10
	Payload and center of gravity	9
	Safety standards	9
	Collision avoidance	8
	Jogging the robot	7
	Speed and acceleration	7
	Kinaesthetic teaching	5
	Interfaces to external equipment, safety precautions	5
	Robot joint names	2
Cobot programming skills	Set waypoints	18
	Activate/deactivate tool (e.g. open/close gripper)	17
	Adjust speed and acceleration	12
	Structure a program (e.g. sequence of operations)	12
	Mount tool	11
	Adhere to safety standards	9
	Configure payload and center of gravity	9
	Use wait commands	9
	Optimize trajectories	2

Regarding specific cobot programming skills, experts agreed on the importance of setting waypoints and operating tools. Indeed, these represent the minimum set of skills needed to program a robot task, assuming that the system was set up by an expert. All the other skills were considered relevant by nearly half of the respondents: they represent some more advanced skills that go beyond basic robot operation. Accordingly, techniques for optimising trajectories were considered not relevant by the greatest majority of the experts. Such optimisation techniques are particularly valuable for users who already possess an understanding of how cobots operate and require a comprehensive knowledge of the task. As a result, these techniques can be acquired in the subsequent learning phase.

Building upon the experts' feedback, the outcome of the questionnaire was then leveraged in a training framework, which can be used generally across cobot platforms to teach novice operators and make sure they gain the necessary skills to program simple operations within manufacturing.

V. Training and Learning Framework for Robot Programming

In this section, we discuss how the experts' feedback on the questionnaire presented in Sec. IV was used to design a training and learning framework for shop floor workers novice to cobots.

A. Methodology

With reference to **RQ2**, the aim of the framework is to teach the elementary robot knowledge and skills that are

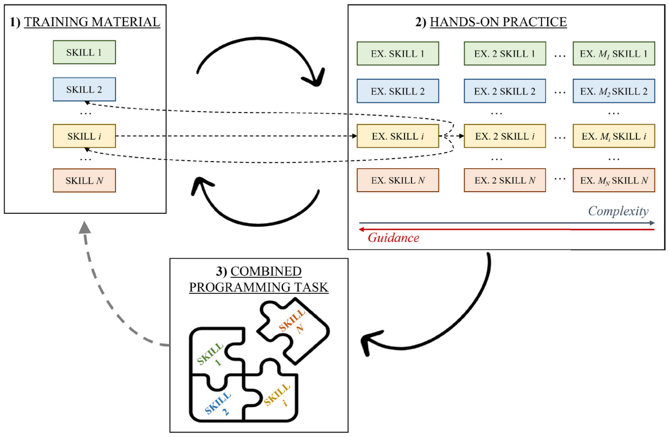


Fig. 1. Organization of the proposed training and learning framework in three parts. *1)* Training material is available for each elementary skill for robot programming. *2)* Hands-on practice material consists of guided examples and exercises with different levels of guidance and complexity for each learnt skill. *3)* A combined programming task is introduced to let learners practice on combining and applying the learnt skills. Solid line arrows show that the first two parts are meant to be visited in parallel, while the third one is meant to be visited after completing the others. The learner can revisit the training material if knowledge is still lacking while performing the combined programming task. The dashed arrows refer to the adaptive logic described in Sec. V-A1.

required to non-expert operators to be self-sufficient in the use of cobots. By non-expert operators, we refer to those subjects who have poor or no knowledge of collaborative robots, despite possibly having high vocational skills in their work domain. The framework is intended to teach them how to use a cobot in an already set up collaborative cell, program simple tasks, or reprogram existing tasks following changes to manufacturing processes.

As discussed in Sec. IV-B, the experts' feedback to the questionnaire was used to define curriculum content. The framework was organised into three main parts, which are depicted in Fig. 1.

First, a description is provided for each elementary skill or piece of knowledge to be used as an introduction to the skill and reference material when practising throughout the curriculum. This is represented in the left upper corner of Fig. 1, where, for the sake of generality, we refer to a number N of elementary skills. The training material for each skill is augmented with images, schemes, and short videos to demonstrate the related skill. The training material briefly presents a cobot, introducing its axes, the rotations that axes are capable of, and the teach pendant. Then, as suggested by the experts in Sec. IV-B, the emergency stop is introduced, explaining when it is needed and how to activate it. The following set of instructions refers to the possible ways to move the robot. To this end, details about kinesthetic teaching, jogging and moving the robot through waypoints, and how to set them are provided (this could for example represent SKILL 1, 2, and 3 in Fig. 1). Available motion types are presented, together with their features and their recommended use. Then, instructions on setting a gripper and changing payload are given for pick and place tasks. Finally, brief instructions are given for slightly more advanced commands, such as setting input/output for tools and sensors and defining the tool center point (TCP).

Second, hands-on practice material is provided for each elementary skill or piece of knowledge. This is represented in the right upper corner of Fig. 1, where a set of M_i guided examples are represented for the skill i (we consider the general case where $M_i \neq M_j$, with $i, j = 1, \ldots N$ and $i \neq j$). Guided examples have progressively increased complexity and gradually reduced guidance by the framework, as suggested by the cognitive apprenticeship approach. Specifically, for each skill, we provide scaffold training in the form of pre-programmed tasks requiring either minimal input from the learner or slightly more advanced reasoning capabilities. Examples with the lowest level of complexity require the learner to enter limited input in an already set robot program structure, reminiscent of a fill-in-the-blanks type of task. A task description is provided in text, and a video recording of the robot correctly executing the program is shown for every task. As complexity increases, the learner is expected to apply the learnt skill in a non-structured example. Finally, each example in this part of the framework is specific for a single skill in order to help the learner consolidate it and build on previously learnt skills, as the learning scaffold is slowly reduced and complexity increases.

Third, a combined programming task is included in the framework, which integrates and combines all the skills learnt separately in the first two parts. This is meant to be visited at the end of the training programme after completing the other two parts. The first two parts are meant to be executed in parallel, practising each skill after having been introduced to the robot and occasionally revisiting the training material if something is not clear (further details in this regard are provided in Sec. V-A1.). The final combined programming task was used as an evaluation and practical demonstration of learnt skills in order for the learner to demonstrate the ability to integrate multiple skills needed to solve a programming task in a cohesive manner.

A key aspect of the proposed framework is how learners navigate it. Since it is composed of different parts, it becomes important that learners do not get lost and the effectiveness of the training programme is not reduced. To this end, we considered two ways of introducing the same material to the learner: i) an adaptive training approach, and ii) a self-regulated training approach.

1) Adaptive Training: The adaptive training approach builds upon the results achieved in [29] and [31], where the progression through the curriculum is adapted to the learner's learning rate and competence development. In other words, we propose that the learner is shown which tasks to perform and told whether to perform them again, study training material, or wait for the next task to be presented. Adaptation of curriculum progression is thus based on the learner's performance in the hands-on practice, meaning the learner performs a certain task, and their performance is then assessed. This is used to decide which exercise to perform next, whether it is needed to revisit relevant training material, or whether the learner is ready to move to the next skill. The possible outcomes for the adaptive logic are depicted with dashed lines in Fig. 1. According to the figure, once the training material for a skill ("SKILL i" in the figure) is visited, the learner is presented with the

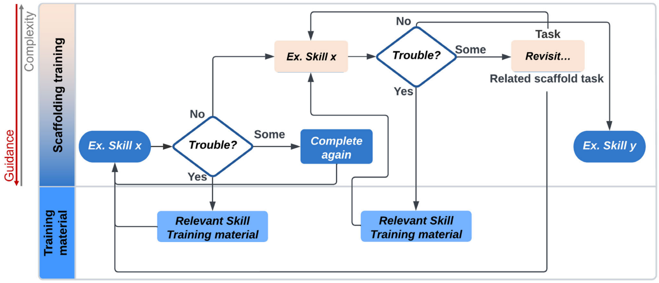
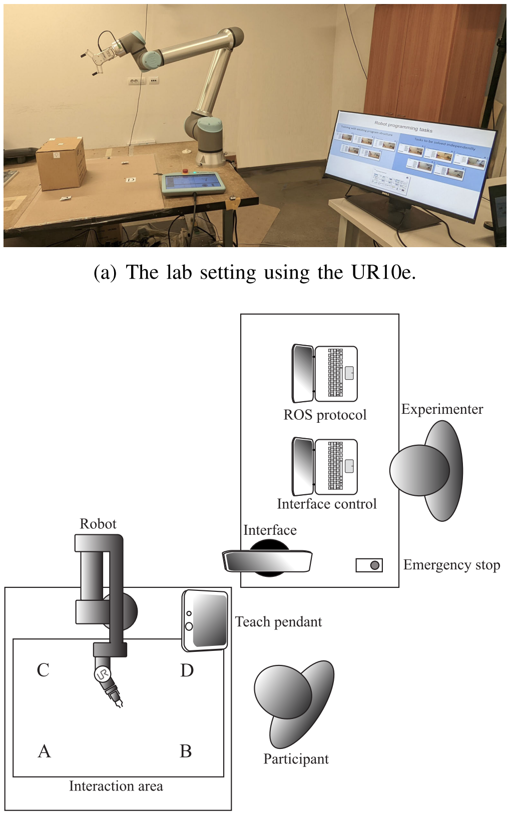


Fig. 2. The logic followed by the WoZ operator (experimenter).

hands-on practice material for that skill. Depending on how the learner performed with a guided example ("EX. SKILL *i*" in the figure), they can be presented with the following example (with increased complexity and reduced guidance) or led back to the training material for the current skill or any previous skill ("SKILL 2" in the figure), if they showed poor confidence with it.

In our implementation, the adaptive logic was designed according to the flow chart reported in Fig. 2 and was implemented with the Wizard of Oz (WoZ) method [34], [35], i.e., the experimenter controlled the user interface by selecting the next task and presenting prompt messages to the participant based on their performance (e.g. prompting them to complete the task again). The WoZ method relies on slight deception to make it appear, in this case, as if the robot and interface detects the user's performance and then autonomously decides the next steps. The WoZ method has proven effective within human-robot interaction studies to quickly obtain convincing results regarding functionalities and interaction modalities not yet fully developed as part of an iterative design cycle [35], [36] – i.e., in this case, the adaptive logic. Prior to the experiments, the flowchart of how to present the information to the participant (Fig. 2) was rehearsed by the WoZ operator in order to provide a uniform experience between subjects. Fig. 2 depicts the simplest flow, where the subject starts with a scaffold task (*Ex. Skill x*). If they experience trouble (*'Yes'*), they are shown training material relevant to the task, before they are prompted to redo *Ex. Skill x*. If they understand the task but the execution was poor (*'Some'* trouble), they are prompted to redo the task to practice their skill. If *'No'* trouble occurred, they are led to a similar task with increased complexity and so on. The flowchart repeats where the subject advances to the next skill (*Ex. Skill y*).[1]

2) Self-Regulated Training: In the self-regulated training, the learner can independently choose which tasks they want to perform and how many times, or seek out relevant training material by navigating the interface. Thus, the learning progress is put into the hands of the learner in a more exploratory manner compared to the adaptive approach. In this setting, the learner has complete control of the user interface displaying the curriculum and can freely navigate through it.[2]



(a) The lab setting using the UR10e.

(b) Layout of the experimental setup.

Fig. 3. The experimental setup.

VI. Experimental Validation

A. Experimental Setup

The framework was demonstrated in two independent setups, located in Italy and Denmark, using two different robots: a UR10e and UR3e, respectively. Both collaborative robots had 6 degrees of freedom and were equipped with an RG2 OnRobot gripper. To monitor and record the robot positions, the ROS Melodic framework with the official Universal Roots ROS drivers were used.[3] The experimental setup is shown in Fig. 3. The learner stands in front of the robot, the teach pendant, and the user interface displaying the training material and tasks. The experimenter is located on their side but outside their field of view governing the controlling devices for the experiment.

B. Implementation of the Proposed Framework

A prototype of the curriculum was implemented as a slideshow with Microsoft Powerpoint.[4] An example of the interface is depicted in Fig. 4. The figure shows the main navigation page for the hands-on practice, where the learner can practice on the single learnt skills: examples on the left offer guidance (i.e., *scaffolding*), as a program structure is already set up for each skill. Exercises on the right require more processing by the learner, as they are only shown a description and a video with no provided program structure.

[1] Adaptive interface video demo

[2] Self-regulated touch interface video demo

[3] https://github.com/UniversalRobots/Universal_Robots_ROS_Driver

[4] https://www.microsoft.com/en-ww/microsoft-365/powerpoint

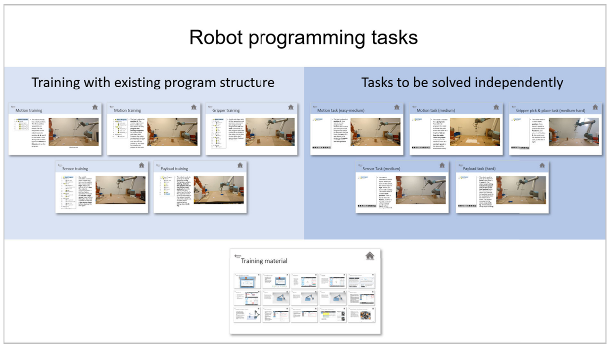


Fig. 4. The main navigation page for the guided examples.

The lower part of the page hosts the link to the training material to allow navigation back and revisit some previous content. Fig. 5 shows a couple of examples with different levels of complexity for practising the use of the gripper.

C. Enrolled Subjects

The inclusion criteria used for the study required participants to possess no, or very low, previous experience with the use of cobots and little knowledge of the basics of robotics. In total, 20 participants were enrolled: following a between-subjects study design, 10 interacted with the UR10e and 10 interacted with the UR3e. In each case, five were randomly assigned to the self-regulated approach and five were randomly assigned to the adaptive approach. Thus, the same approach was followed in both lab settings. The participants (4 females, 15 males, 1 preferred not to say, age range: 21-36 years, mean age: 26.5 years) were volunteer students enrolled in programs related to manufacturing (namely, programs for automation technician, mechatronics engineer and former electricians) and all of them were completely naive to cobot programming.

A participant information sheet was emailed to the participants prior to participation and a physical copy was provided when they showed up. The sheet explained the purpose of the study as well as what was expected of them. The participants all signed an informed consent form agreeing to participate in the study and acknowledged that they understood they were free to withdraw from the study at any time before simple demographic information was collected. All the data were analysed and reported confidentially.

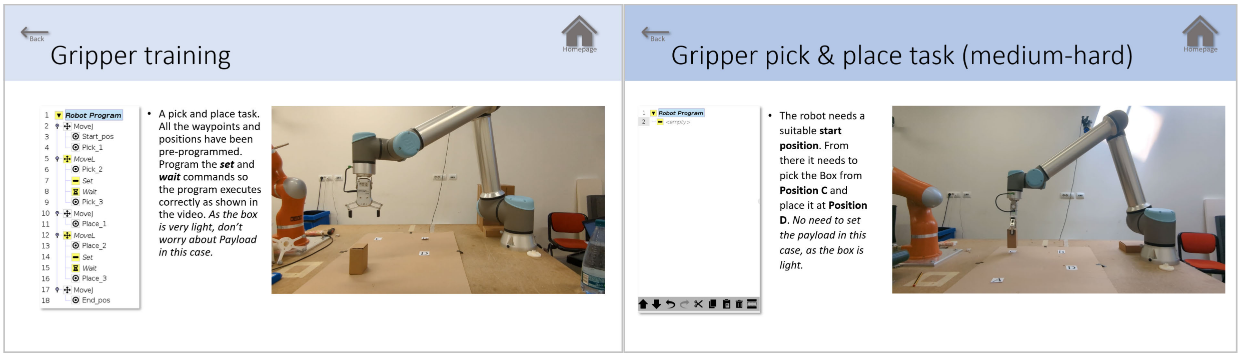
D. Experimental Protocol

The considered experimental protocol is depicted in Fig. 6. It comprised six steps, three of which are the core parts of the proposed learning framework, as in Fig. 1, and the other three were introduced to quantify the subject's improvement and gather their feedback. In particular, the steps are detailed hereafter.

1) Introduction to the training material. In this part, subjects were presented with the whole training material to build their domain knowledge.
2) Baseline measurement task. The aim of this step was to assess the subject's baseline programming skills, before practising with the proposed framework. The task consisted of guiding the robot to four different points, as shown in Fig. 7(a). TCP robot pose was recorded by the experimenter for each point. The points were marked on two sides of a box placed in front of the robot, as shown in Fig. 3(a).
3) Hands-on practice. In this step, subjects were given a fixed time slot to practice the learnt skills through scaffolding training. Duration was set to 70 minutes for the UR10e, and 60 minutes for the UR3e: participants needed more time with UR10e due to the robot being larger and heavier to manoeuvre. During this step, the participant may revisit the training material, as illustrated by the dotted arrows in Fig. 6 and as introduced in Sec. V-A. The A, B, C, and D markers shown in Fig. 3(b) could be re-organised by the experimenter when a task was replayed. This meant that the same programming task contained some variation in order to test the learner's understanding of the programming skill in different physical positions. For every task, the experimenter loaded the right program on the teach pendant (either empty or the scaffolding program in question) and set up the interaction area according to the task.
4) Intermediate measurement task. This was the same as the former baseline measurement task. Similar TCP poses were recorded to be compared to those from baseline measurement in order to assess a subject's individual improvement.
5) Combined programming task. Following the rationale in Sec. V-A, a set of tasks was designed to combine the previously learnt skills into a coherent program in the form of a synthesis of elementary skills. The task is shown in Fig. 3(b). This step served to show whether the learner is capable of putting together multiple elementary skills into actionable programming without access to training material.
6) Exit interview. Participants were asked about their experience with the robot and how the information was portrayed to them.

With reference to Fig. 7(b), the final combined programming task consisted of programming the robot to pick a bottle from a box (subtask 1 in the figure) and place it on the table (2). Then, the robot had to pick another small box (3), wait for a digital input (4) before placing the box in a specified area (5). Finally, the robot had to linearly move along the sides of the box (6, red line). The digital input was connected to a manually activated switch, simulating input from a sensor, while the last sub-task was meant to simulate a welding or gluing task. Indeed, during a welding or gluing task the robot end-effector has to follow exactly a path, usually linear, with a custom tool that performs the operation.

The protocol was organised through the same steps for the self-regulated and adaptive groups, with the only difference being in the hands-on practice. Specifically, for the self-regulated training, the interface was presented on a touch screen, and navigation among the different parts of the framework and the scaffold training was allowed between all the



Gripper training

```
Robot Program
  MoveJ
    Start_pos
    Pick_1
  MoveL
    Pick_2
    Set
    Wait
    Pick_3
  MoveJ
    Place_1
  MoveL
    Place_2
    Set
    Wait
    Place_3
  MoveJ
    End_pos
```

- A pick and place task. All the waypoints and positions have been pre-programmed. Program the ***set*** and ***wait*** commands so the program executes correctly as shown in the video. *As the box is very light, don't worry about Payload in this case.*

Gripper pick & place task (medium-hard)

```
Robot Program
  <empty>
```

- The robot needs a suitable **start position**. From there it needs to pick the Box from **Position C** and place it at **Position D**. *No need to set the payload in this case, as the box is light.*

Fig. 5. Difference between hands-on practice material: scaffolding training with existing program structure (left) and examples to be solved independently (right). The images are stills from a looped video showcasing correct task execution.

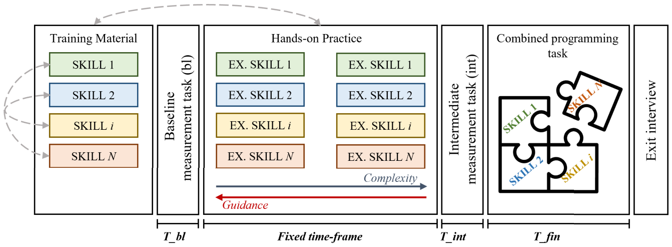
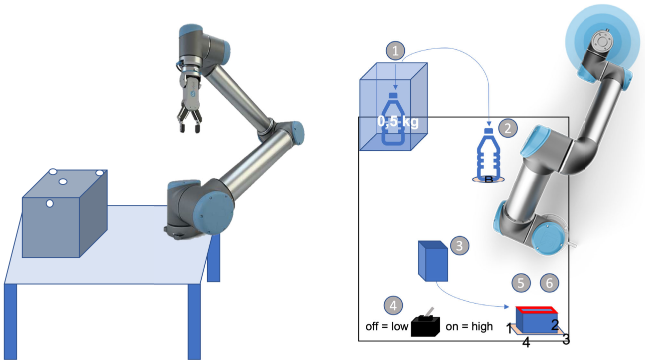


Fig. 6. Experimental protocol: introduction to the robot and the available training material; baseline measurement task; the hands-on practice with scaffold training with a fixed time-frame to practise (UR10e = 70 minutes, UR3e = 60 minutes); intermediate measurement task; a final combined programming task; a semi-structured exit interview.



(a) Baseline and intermediate measurement task.

(b) Final programming task.

Fig. 7. Organization of the baseline and intermediate measurement task, and the final programming task.

pages and controlled by the learner. For the adaptive training, the learner had no access to the touch interface but rather saw the monitor displaying the interface, and navigation was controlled by the experimenter acting as a WoZ operator, following the logic in Fig. 2. Each participant was handed a stylus pen for the teach pendant as UR10e and UR3e were not always responsive to touch input.

E. Performance Metrics

To assess the impact of the proposed framework on learning how to program a robot, we considered different quantitative and qualitative metrics. Subject performance was assessed during baseline measurement, intermediate measurement and final combined programming tasks. Specifically, during baseline and intermediate measurement, time-on-task was measured, and the TCP pose for each point was recorded. Time and TCP poses were used to assess any improvement in terms of time and accuracy in the intermediate measurement with respect to the baseline. During the final combined programming task, the time to accomplish the whole task was recorded, and TCP poses in the four vertices of the box in the gluing task were recorded. Additionally, the experimenter took note of any errors that occurred during the entire programming task. All the robot data were collected using the official UR ROS Driver,[5] which provides complete access to the current joint configuration. For measuring task time, a stopwatch was manually employed, as the introduced error is negligible.

Feedback from test participants was gathered in a short exit interview, consisting of four questions. These asked whether *i)* they found the training material adequate to program the robot (*Was the training material adequate to program the robot?*), *ii)* there were any points during the experiment where there was missing information (*Did you encounter any points during the experiment where you were missing information?*), *iii)* they would have preferred to look up their own training material (for the adaptive group) (*Would you prefer to look up your own training material?* or have relevant training material presented (for the self-regulated group) (*Would you prefer to have relevant training material presented to you, while programming the robot, opposed to looking it up yourself?*), and, finally, *iv)* if they had any comments on the way the information was presented on the monitor (*Do you have any thoughts on how you experienced the way the information was presented to you on the monitor?*). Additionally, a think aloud approach, which consists in keeping track of the user's thoughts while using the system, was used to report unsolicited feedback [37].

VII. Outcomes of Experimental Validation

A. Subjective Feedback

At the end of the experimental protocol, test participants provided, in general, positive feedback about the training material and facilitation. In reply to the first two questions of the exit interview, they all reported the training material adequate

[5] https://github.com/UniversalRobots/Universal_Robots_ROS_Driver

and did not report a lack of important information. Two of them mentioned that there was a lot of information presented to them in the beginning, which felt a little overwhelming at first. Interestingly, in reply to the third question, two of the subjects in the self-regulated training reported that they navigated the different tasks in almost random order and would prefer some guidance in navigating through the interface. In reply to the same questions, all the participants in the adaptive group reported that they were satisfied with guided navigation since they trusted the system and the experimenter. No relevant comments were provided in reply to the fourth question.

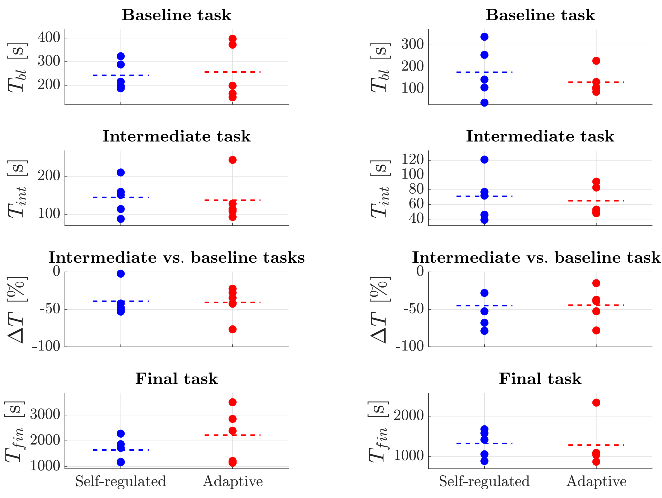
B. Common Novice Mistakes

During the entire experimental protocol, the experimenter observed and took notes of participants' behaviour, comments, and mistakes. Moreover, we inspected the programs generated in the final combined task and held them up against a baseline program created by the experimenter, which served as the correct way of solving the more complex programming tasks. As a result, it was possible to identify some commonly occurring mistakes for novice users who are just learning to program collaborative robots.

However, because there does not exist a finite solution for solving the tasks, but many different ways and approaches could be used to solve the tasks, we focused on major mistakes and discarded minor deviations such as using more waypoints than needed or different angles of approach.

In both robot setups (UR10e and UR3e), regardless of condition (adaptive or self-regulated training), some participants made more mistakes compared to others. There were, however, common recurring mistakes such as neglecting a wait command after activating the gripper, troubles with correct payload, correct use of motion types and neglecting intermediate waypoints in pick and place tasks for safer object handling. Additionally, the experimenter had to activate the emergency stop to avoid collisions with either the environment or the robot itself. These mistakes are summarised hereafter.

- **Wait command:** If a wait command was neglected when activating a gripper or misplaced, it was noted as an error. This happened for 8 participants out of 20.
- **Motion type:** While not critical to task completion, a non-ideal use of motion types was noted as an error. This happened to 11 participants.
- **Payload:** If setting payload was neglected or incorrect, this was noted as an error. This happened for 6 participants.
- **Intermediate waypoint:** If intermediate waypoints in pick and place tasks were neglected, it counted as an error (despite not always resulting in collision). This happened for 4 participants.
- **Task related:** If the experimenter had to intervene verbally or objects were placed imprecisely, this was counted as a task-related error. This happened for almost all the participants, specifically 13 out of 20.
- **Emergency stop:** The experimenter had an external emergency stop button on hand and noted down each time it was activated preemptively to avoid minor collisions and protect the equipment. This happened for 8 participants, in some cases more than once per participant.



(a) Setup with UR10e robot.

(b) Setup with UR3e robot.

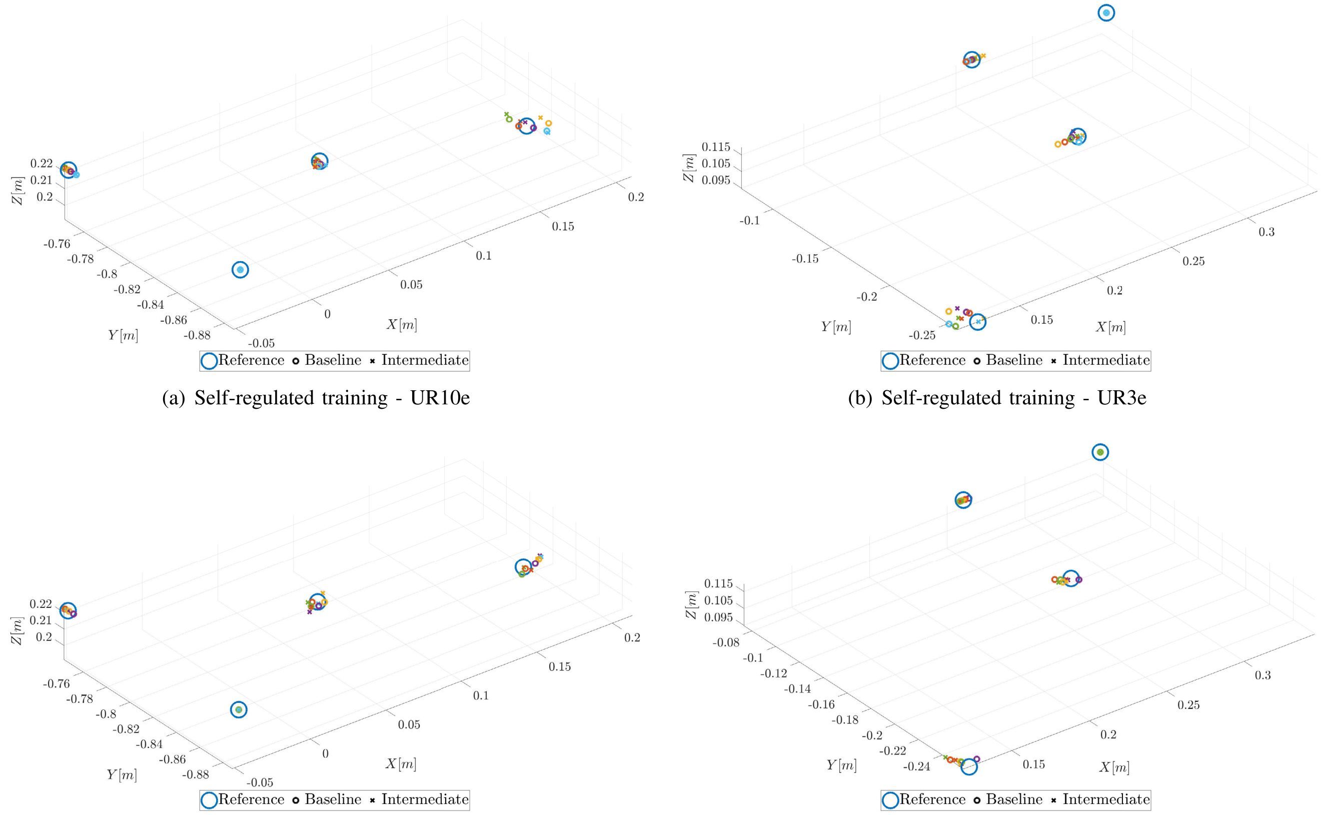
Fig. 8. Time required to complete the different phases of the experimental sessions in the two considered setups (blue: self-regulated; red: adaptive). The boxplots show a significant reduction of time exploiting the proposed training method.

A few additional comments are worthy to be reported. A very common observation for pick and place tasks was that the participants typically did not grip the object to program a precise place position. Instead, they tried to eyeball the right distance from the table with an empty gripper, which very often resulted in imprecise placement or collision with the table. Moreover, it was also often observed that the participants brought the robot in configurations close to singularity, thus accidentally triggering a protective stop. When close to maximum arm extension, the low-level controller is not capable of computing the Cartesian input to move the robot in the desired position [38] and automatically stops the robot to avoid dangerous movements. In our experimental evaluation, this was observed more with the smaller UR3e robot due to the smaller reach available.

C. Comparison Between Self-Regulated and Adaptive Training

Performance comparison between adaptive and self-regulated training was carried out in a post processing phase, considering the time required to complete the experimental protocol and the accuracy of the programmed tasks.

Fig. 8 shows time measurements for the two learning conditions. Specifically, in all the panels, we compared self-regulated (blue, left) and adaptive training (red, right), considering the two setups (with UR10e robot on the left and with UR3e robot on the right). Circles represent the time measured for each test participant (five participants per condition, per setup), while dashed lines denote mean values. From the top, the panels refer to: *i)* time required to accomplish the initial baseline measurement task (T_{bl}), *ii)* the intermediate measurement task (T_{int}), which was the same as the baseline task but happened after the hands-on



(a) Self-regulated training - UR10e

(b) Self-regulated training - UR3e

(c) Adaptive training - UR10e

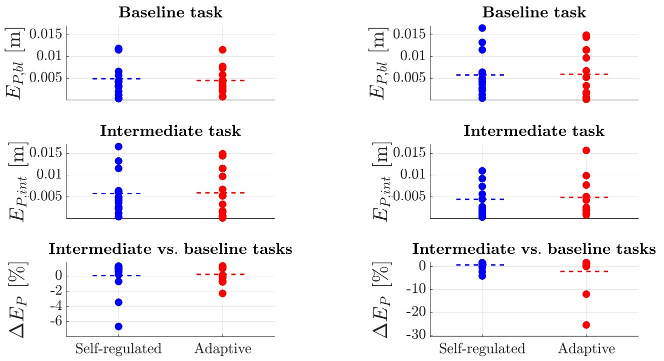
(d) Adaptive training - UR3e

Fig. 9. Position of the end-effector recorded during the baseline task and the intermediate task for both training with the UR10e setup (left) and the UR3e setup (right). The plots show that the users have been able to achieve good performances in positioning the robot.

practice phase, *iii)* the improvement, for each test participant, between the intermediate and baseline measurement tasks with respect to T_{bl}, defined as $\Delta T = \frac{T_{int}-T_{bl}}{T_{bl}} \cdot 100$, and *iv)* the final combined programming task depicted in Fig. 7(b) (T_{fin}). Overall, the figure shows that no relevant differences in terms of required time were measured between the two conditions of self-regulated and adaptive training. Nevertheless, the third panel of both setups, namely ΔT, shows that a notable improvement was achieved with the hands-on practice, thus showing the effectiveness of the proposed training approach. Considering the two setups together, on average, the relative improvement was of 42.0% for the self-regulated condition and 42.5% for the adaptive approach.

Regarding the accuracy of programmed tasks, we measured how accurately users could program the robot to move to the points in Fig. 7(a) and the edges of the box in Fig. 7(b) (step 6, red line).

First, with respect to Fig. 7(a), we assessed any improvement in the intermediate task with respect to the baseline, comparing robot positions in the four points in the intermediate and baseline tasks. Specifically, Fig. 9 shows the points reached by the two groups during both the baseline and the intermediate task for both scenarios, w.r.t. the reference position recorded by the experimenter. In the figure, the big blue circle markers represent the reference points, while the other markers represent the points reached by each user in the baseline task, coloured small circles, and in the intermediate task, coloured small crosses. The same colour refers to the same user.



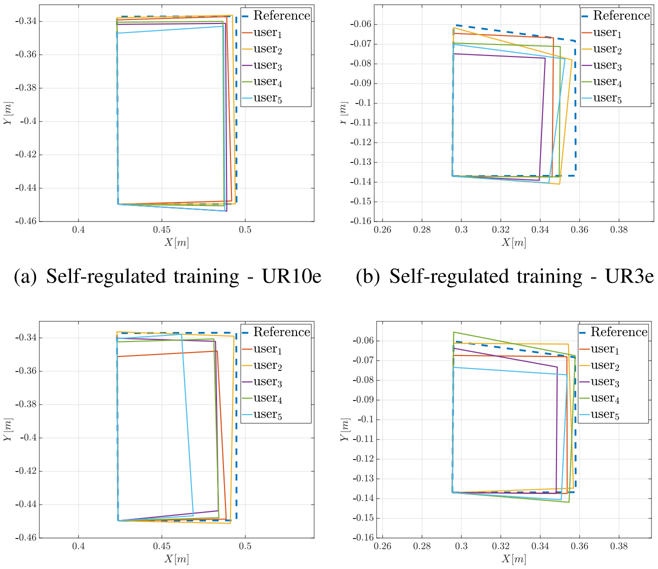
(a) Setup with UR10e robot.

(b) Setup with UR3e robot.

Fig. 10. Position error in the different phases of the experimental sessions for the two considered setups (blue: self-regulated; red: adaptive). The boxplots show that the position error achieved by both groups is always low, without significant difference between the baseline and the intermediate task.

Starting from the points collected in the experiments, it is possible to perform a quantitative evaluation on the performance of each user. It is worth underlining that to achieve a more realistic comparison and evaluation, these three post-processing operations were exploited:

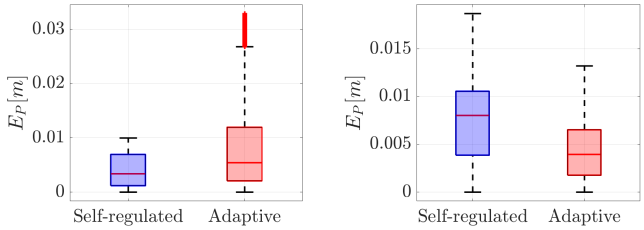
1) The starting point was set coincident with the nominal one for all the users. This is necessary to avoid the



(a) Self-regulated training - UR10e (b) Self-regulated training - UR3e

(c) Adaptive training - UR10e (d) Adaptive training - UR3e

Fig. 11. Paths recorded during the gluing task for both training in the UR10e setup (left) and UR3e setup (right).



(a) Position Error - UR10e (b) Position Error - UR3e

Fig. 12. Boxplots of the position error during the gluing task.

accumulation of errors due to a wrong placement of the box. In fact, we are only interested in how the users perform in programming the robot and not how they performed in manually placing the box.

2) The direction that goes from the first point to the second one is set equal for all the users. As before, this is necessary to compensate for the orientation errors in the placement phase.
3) The Z component has been compensated. This is due to the fact that each user prefers to put the gripper to a different distance w.r.t. box, i.e. someone prefers to touch the box while others prefer to stay at a higher position to avoid contact.

Starting from all the points collected, it is possible to compute the error E_P that each user made in performing the task, i.e. the Euclidean distance w.r.t. the reference. This is represented in Fig. 10. Specifically, as done for time data analysis in Fig. 8, we compared self-regulated (blue, left) and adaptive training (red, right) in both setups. Circles represent the Euclidean distance measured for each test participant, while dashed lines denote mean values. From the top, the panels refer to: *i)* error achieved during the baseline task ($E_{P,bl}$), *ii)* the intermediate task ($E_{P,int}$) *iii)* the improvement, for each test participant, between the intermediate and baseline measurement tasks ($\Delta P_{int} = \frac{E_{P,int} - E_{P,bl}}{E_{P,bl}} \cdot 100$). The figure shows that for the UR10e setup, the adaptive training performed slightly better, while the contrary happened for the UR3e setup. Regarding the difference between before and after the training, on average we did not register notable variations in the performances. The only exception is the adaptive training with the UR3e setup.

Lastly, Fig. 11 shows the overall path computed during the gluing task required in the combined programming task (Fig. 7(b)), w.r.t. the reference one. Since the gluing task requires a linear trajectory of the end-effector, in this phase all the intermediate points have been recorded. It is possible to note that all paths have the lower left corner coincident and that the direction of the left side is equal. This is due to the fact that, as before, we wanted to analyse only the quality of the path performed with the end-effector, without considering the box placing error. In fact, if the box is placed in a wrong position or wrong orientation, it is still possible to achieve a perfect gluing. Thus, in a post-processing phase, the paths have been aligned at the starting point. It is worth noting that the box used in the UR3e setup was not perfectly rectangular, which is why the left side appears skewed.

Starting from the paths, it is possible to compute the error that each user made in the gluing task. As opposed to the time measurements, we have a lot more data points, i.e. the entire path for each user, and thus the error can be meaningfully represented through boxplots, illustrated in Fig. 12. Specifically, for the experiment with the UR10e, the self-regulated training performed slightly better, while for the UR3e it was the opposite.

VIII. Discussion and Concluding Remarks

In this paper, we highlighted a list of necessary *knowledge* and *skills* for letting novice operators be self-sufficient in the use of collaborative robots (**RQ1** - see Table I). We then presented and tested an approach for cobot programming training. The approach focuses on industrial operators who are novices to collaborative robotics and aims to teach them essential elementary knowledge about cobots and their programming basics. The suggested training approach was inspired by a hands-on problem-based approach, relying on cognitive apprenticeship principles (**RQ2**).

Overall, the proposed approach shows promise and is able to adjust to different robots and settings. It shows that relatively little training in a few selected key focus areas can produce great advances in cobot programming capability for novice users in a short time span. All participants demonstrated individual improvement and were capable of programming the robot by the end of the experiments. However, based on our sample of 20 participants, it was inconclusive whether an adaptive approach as suggested by Mariani et al. [31] and Mayrhofer et al. [29] performed better compared to a self-regulated training approach. A deeper investigation of more appropriate adaptive training approaches was out of the scope of this paper. However, it would be useful to extend previous results achieved in the literature to the use of collaborative robotic arms in industrial scenarios.

We also identified some commonly occurring novice mistakes, which highlights specific areas, where increased attention may be necessary when facilitating training for novice cobot users. Many of these mistakes may have been made due to the fact that, during the learning phase, the users become more confident and underestimate the importance of a certain skill, e.g., not adding a pause when operating the gripper could cause a collision. Due to the lack of experience, this could be perceived by the user as a waste of time. An example of a dangerous situation that may occur by not adhering precisely to a skill procedure would be an interesting addition to future training material. This could give the user a more detailed view of the importance of the single skill. One user commented that they would be more precise in their programming if they knew the program was to be run in a real production environment. This underlines the importance of a situated learning environment closely resembling real-world applications [32].

The results achieved in our experimental validation relied mainly on an interface with text-based information, which may not always be the best suited modality to introduce material for learners. Others have compared multiple modalities such as augmented reality projections onto the collaborative robot workspace, which scored high on task performance and user preference compared to text printouts or monitors [39], [40]. Robots endowed with speech capabilities to guide users also show promise when applied in structured learning environments [6]. It is worthwhile to investigate not only adaptive learning environments but also different modalities to facilitate the learning framework for training.

Learning how to operate a robot arm is only a small part of the issues faced when pursuing automation. A robot arm, like a cobot, is often a mere part of a larger automation puzzle with increased complexity for every piece in the network. Just to mention a few common examples, the cobot often needs a tool mounted with additional sensors, which may send signals to activate a conveyor, control external safety devices, or receive signals from other connected machines in the production flow. It all has to fit together in cohesion for automation to work. Learning about the robot may be a first step but the real challenge is how to convey and navigate the complexity of entire automation solutions to the existing workforce. Notwithstanding, small to medium-sized enterprises may use the proposed learning approach as a way to garner interest, engage, and support novice workers in their initial encounter with collaborative robots. Many participants commented on how it was fun to program the robot, which may provide the right incentive to learn more about how the technology works and could be adapted to suit their day-to-day production needs. This also suggests that the difficulty level was just right to achieve a flow-state, where the learner is engaged in tasks that are just challenging enough to keep them going, yet, not easy enough that they feel bored or disengaged [41].

An 'Operator 4.0' typology has already been proposed by [42] , which outlines the type of capabilities workers within Industry 4.0 should be able to do when working in production environments rich with data, virtual reality, exoskeletons, increased information and communication flow, biometric sensors, and robots. Future research may look into unfolding the operator typology specifically for operators expected to work closely with robots. Such a typology could help identify multiple responsibility areas and the extent of the competences needed in those. For example, a shop floor worker may only need the competences to start/stop the robot to refill materials or edit established waypoints, whereas specialised knowledge surrounding safety standards and communication protocols is needed to initially set up the robot on the factory floor to ensure safe operation. Arriving at such a typology may aid in more focused training suitable to the knowledge and skill level required of that person's responsibility area. Future work could also look into more longitudinal effects of the training program or explore more diverse programming tasks, e.g., using cobots from different manufacturers or adding safety-related training. Being safe around robots is still a major topic, and detailed open source safety information for human-robot collaboration (HRC) already exists, such as the COVR Toolkit.[6] Our proposed framework could be used in combination with the COVR toolkit to provide a structured approach to learning for the type of robot operator that is being targeted.

Finally, an extension of this work is ongoing, where the cognitive load of subjects related to the proposed framework is measured. Several physiological data (heart rate, eye activity, electrodermal activity, and brain activity) have been recorded together with user feedback using the NASA-TLX questionnaire while subjects were learning through the proposed framework and performing different programming tasks with varying difficulty [43]. The aim of that study is to collect further insights about the users' perception of the proposed framework and its usability. The collected dataset, named, SenseCobot, is publicly available[7] for further research on this topic.

References

[1] H. Kagermann, W. Wahlster, J. Helbig, and A. Hellinger, *Securing the Future of German Manufacturing Industry: Recommendations for Implementing the Strategic Initiative Industrie 4.0*. Frankfurt, Germany: Forschungsunion, 2013.

[2] L. D. Xu, E. L. Xu, and L. Li, "Industry 4.0: State of the art and future trends," *Int. J. Prod. Res.*, vol. 56, no. 8, pp. 2941–2962, Apr.18, 2018.

[3] V. Villani, F. Pini, F. Leali, and C. Secchi, "Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications," *Mechatronics*, vol. 55, pp. 248–266, Nov. 2018.

[4] A. Weiss, A.-K. Wortmeier, and B. Kubicek, "Cobots in Industry 4.0: A roadmap for future practice studies on human–robot collaboration," *IEEE Trans. Human-Mach. Syst.*, vol. 51, no. 4, pp. 335–345, Aug. 2021.

[5] *World Robotics 2022: Industrial Robots. Statistics, Market Analysis, Forecasts, and Case Studies*, VDMA Services GmbH, IFR, Frankfurt, Germany, 2022.

[6] C. Li, A. K. Hansen, D. Chrysostomou, S. Bøgh, and O. Madsen, "Bringing a natural language-enabled virtual assistant to industrial mobile robots for learning, training and assistance of manufacturing tasks," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Jan. 2022, pp. 238–243.

[7] I. El Makrini et al., "Working with walt: How a cobot was developed and inserted on an auto assembly line," *IEEE Robot. Autom. Mag.*, vol. 25, no. 2, pp. 51–58, Jun. 2018.

[8] M. Breque, L. De Nul, and A. Petridis, *Industry 5.0: Towards a Sustainable, Human-Centric and Resilient European Industry*. New Delhi, India: Publications Office, 2021.

[6] https://www.safearoundrobots.com/ - Accessed 2023-08-14

[7] https://doi.org/10.5281/zenodo.8363762

[9] J. E. Michaelis, A. Siebert-Evenstone, D. W. Shaffer, and B. Mutlu, "Collaborative or simply uncaged? Understanding human-cobot interactions in automation," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, Apr. 2020, pp. 1–12.

[10] M. Cugno, R. Castagnoli, and G. Büchi, "Openness to Industry 4.0 and performance: The impact of barriers and incentives," *Technolog. Forecasting Social Change*, vol. 168, Jul. 2021, Art. no. 120756.

[11] D. Horváth and R. Z. Szabó, "Driving forces and barriers of Industry 4.0: Do multinational and small and medium-sized companies have equal opportunities?" *Technolog. Forecasting Social Change*, vol. 146, pp. 119–132, Sep. 2019.

[12] A. Raj, G. Dwivedi, A. Sharma, A. B. Lopes de Sousa Jabbour, and S. Rajak, "Barriers to the adoption of industry 4.0 technologies in the manufacturing sector: An inter-country comparative perspective," *Int. J. Prod. Econ.*, vol. 224, Art. no. 107546, Jun. 2020, doi: 10.5281/zenodo.8363762. [Online]. Available: https://www.safearoundrobots.com/

[13] L. G. Christiernin, "How to describe interaction with a collaborative robot," in *Proc. Companion ACM/IEEE Int. Conf. Hum.-Robot Interact.*, Mar. 2017, pp. 93–94.

[14] A. M. Djuric, R. J. Urbanic, and J. L. Rickli, "A framework for collaborative robot (CoBot) integration in advanced manufacturing systems," *SAE Int. J. Mater. Manuf.*, vol. 9, no. 2, pp. 457–464, Apr. 2016.

[15] A. Oliveira, H. Assumpcao, J. Queiroz, L. Piardi, J. Parra, and P. Leitao, "Hands-on learning modules for upskilling in industry 4.0 technologies," in *Proc. IEEE 5th Int. Conf. Ind. Cyber-Phys. Syst. (ICPS)*, Coventry, U.K., May 2022, pp. 1–6.

[16] A. Djuric, J. Rickli, V. Jovanovic, and D. Foster, "Hands-on learning environment and educational curriculum on collaborative robotics," in *Proc. ASEE Annu. Conf. Expo.*, Jun. 2017, pp. 1–15.

[17] C. A. S. Geraldes et al., "Co-design process for upskilling the workforce in the factories of the future," in *Proc. IECON 47th Annu. Conf. IEEE Ind. Electron. Soc.*, Oct. 2021, pp. 1–6.

[18] E. Baumgartner, "Combining simulations and hands-on learning in robotic programming courses," in *A Retrospective of Teaching, Technology, and Teacher Education During the COVID-19 Pandemic*. AACE-Association for the Advancement of Computing in Education, 2022, p. 203.

[19] C. Schmidbauer, T. Komenda, and S. Schlund, "Teaching cobots in learning factories—User and usability-driven implications," *Proc. Manuf.*, vol. 45, pp. 398–404, Jan. 2020.

[20] J. Hoyos, A. B. Junaid, M. R. Afzal, A. Tirmizi, and P. Leconte, "Skill-based easy programming interface for industrial applications," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Jan. 2022, pp. 210–217.

[21] A. Ciontos, I.-M. Sarivan, and C. Schou, "Universal industrial interface - mobile," *Proc. Manuf.*, vol. 38, pp. 391–399, Jan. 2019.

[22] W. Wang, R. Li, Y. Chen, Y. Sun, and Y. Jia, "Predicting human intentions in human–robot hand-over tasks through multimodal learning," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 2339–2353, Jul. 2022.

[23] C. Schou, J. S. Damgaard, S. Bøgh, and O. Madsen, "Human-robot interface for instructing industrial tasks using kinesthetic teaching," in *Proc. IEEE ISR*, Oct. 2013, pp. 1–6.

[24] M. R. Pedersen et al., "Robot skills for manufacturing: From concept to industrial deployment," *Robot. Computer-Integrated Manuf.*, vol. 37, pp. 282–291, Feb. 2016.

[25] P. J. Koch et al., "A skill-based robot co-worker for industrial maintenance tasks," *Proc. Manuf.*, vol. 11, pp. 83–90, Jan. 2017.

[26] H. Wu, W. Yan, Z. Xu, T. Cheng, and X. Zhou, "A framework of robot skill learning from complex and long-horizon tasks," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 3628–3638, Oct. 2022.

[27] E. Abele et al., "Learning factories for future oriented research and education in manufacturing," *CIRP Ann.*, vol. 66, no. 2, pp. 803–826, 2017.

[28] D. G. H. Sorensen, A. H. Lassen, M. S. S. Larsen, and A. K. Hansen, "Learning factories for learning and experimentation on Industry 4.0 in SMEs," in *The Future of Smart Production for SMEs: A Methodological and Practical Approach Towards Digitalization in SMEs*, O. Madsen, U. Berger, C. Møller, A. Heidemann Lassen, B. Vejrum Waehrens, and C. Schou, Eds. Cham, Switzerland: Springer, 2023, pp. 431–440.

[29] W. Mayrhofer, S. Nixdorf, C. Fischer, T. Zigart, C. Schmidbauer, and S. Schlund, "Learning nuggets for cobot education: A conceptual framework, implementation, and evaluation of adaptive learning content," *SSRN Electron. J.*, 2021, doi: 10.2139/ssrn.3868713.

[30] R. A. Schmidt, "Frequent augmented feedback can degrade learning: Evidence and interpretations," in *Tutorials in Motor Neuroscience*, J. Requin and G. E. Stelmach, Eds. Dordrecht, The Netherlands: Springer, 1991, pp. 59–75.

[31] A. Mariani, E. Pellegrini, and E. De Momi, "Skill-oriented and performance-driven adaptive curricula for training in robot-assisted surgery using simulators: A feasibility study," *IEEE Trans. Biomed. Eng.*, vol. 68, no. 2, pp. 685–694, Feb. 2021.

[32] A. Collins and M. Kapur, "Cognitive apprenticeship," in *The Cambridge Handbook of the Learning Sciences* (Cambridge Handbooks in Psychology), 2nd ed., R. K. Sawyer, Ed. Cambridge, U.K.: Cambridge Univ. Press, 2014, pp. 109–127.

[33] M. Steinert, "A dissensus based online delphi approach: An explorative research tool," *Technolog. Forecasting Social Change*, vol. 76, no. 3, pp. 291–300, Mar. 2009.

[34] J. F. Kelley, "An iterative design methodology for user-friendly natural language office information applications," *ACM Trans. Inf. Syst.*, vol. 2, no. 1, pp. 26–41, Jan. 1984.

[35] L. Riek, "Wizard of OZ studies in HRI: A systematic review and new reporting guidelines," *J. Hum.-Robot Interact.*, vol. 1, no. 1, pp. 119–136, Jul. 2012.

[36] C. Li, D. Chrysostomou, X. Zhang, and H. Yang, "IRWoZ: Constructing an industrial robot wizard-of-OZ dialoguing dataset," *IEEE Access*, vol. 11, pp. 28236–28251, 2023.

[37] J. Nielsen, *Usability Engineering* (Interactive Technologies). San Francisco, CA, USA: Elsevier, 1994, ch. 6.8 Thinking Aloud, pp. 195–200.

[38] B. Siciliano, O. Khatib, and T. Kröger, *Springer Handbook of Robotics*, vol. 200. Cham, Switzerland: Springer, 2008.

[39] R. S. Andersen, O. Madsen, T. B. Moeslund, and H. B. Amor, "Projecting robot intentions into human environments," in *Proc. 25th IEEE Int. Symp. Robot Hum. Interact. Commun. (RO-MAN)*, Aug. 2016, pp. 294–301.

[40] W. P. Chan, G. Hanks, M. Sakr, T. Zuo, H. F. Machiel Van der Loos, and E. Croft, "An augmented reality human-robot physical collaboration interface design for shared, large-scale, labour-intensive manufacturing tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 11308–11313.

[41] S. Engeser and F. Rheinberg, "Flow, performance and moderators of challenge-skill balance," *Motivat. Emotion*, vol. 32, no. 3, pp. 158–172, Sep. 2008.

[42] D. Romero et al., "Towards an operator 4.0 typology: A human-centric perspective on the fourth industrial revolution technologies," in *Proc. CIE 46th Int. Conf. Comput. Ind. Eng.*, Apr. 2017, pp. 1–11.

[43] S. Borghi et al., "Unlocking human-robot dynamics: Introducing SenseCobot, a novel multimodal dataset on Industry 4.0," in *Proc. ACM/IEEE Int. Conf. Hum.-Robot Interact.* Peru. New York, NY, USA: Association for Computing Machinery, Mar. 2024, pp. 880–884.