A Hybrid User Interface Combining AR, Desktop, and Mobile Interfaces for Enhanced Industrial Robot Programming

Jan Krieglstein[1,2], Jan Kolberg[1], Aimée Sousa Calepso[2], Werner Kraus[1] and Michael Sedlmair[2]

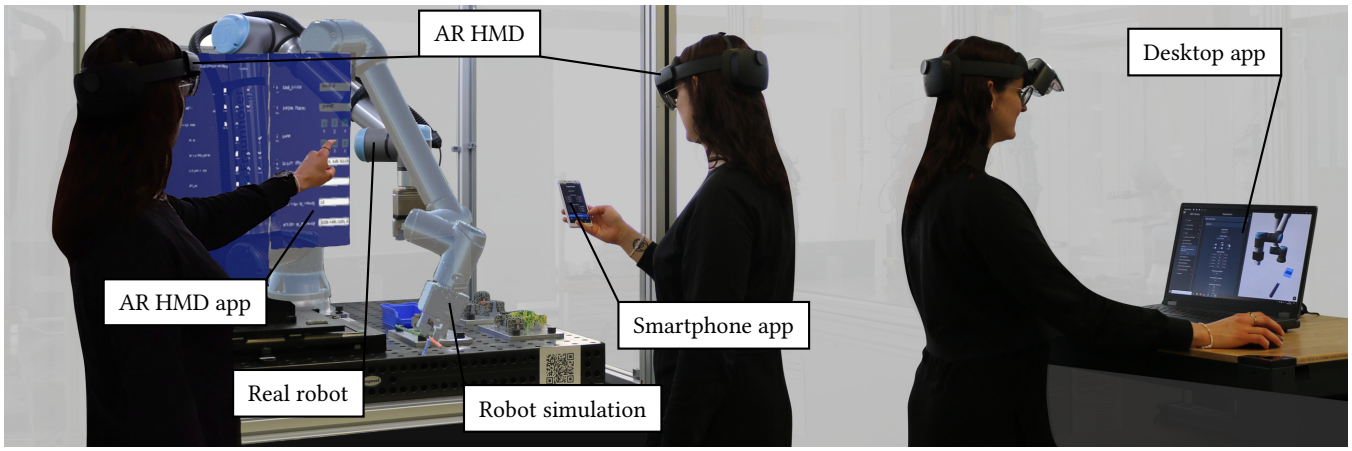

Fig. 1: The proposed hybrid user interface for robot programming comprises a head-mounted display (HMD) app showing Augmented Reality (AR) visualizations and a respective user interface, a smartphone app, and a desktop app. All apps are synchronized and show the same state, so the user can seamlessly switch between them, allowing the user to choose the appropriate device for each sub-task of robot programming.

***Abstract*— Robot programming for complex assembly tasks is challenging and demands expert knowledge. With Augmented Reality (AR), immersive 3D visualization can be placed in the robot's intrinsic coordinate system to support robot programming. However, AR interfaces introduce usability challenges. To address these, we introduce a hybrid user interface (HUI) that combines a 2D desktop, a smartphone, and an AR head-mounted display (HMD) application, enabling operators to choose the most suitable device for each sub-task. The evaluation with an expert user study shows that an HUI can enhance efficiency and user experience by selecting the appropriate device for each sub-task. Generally, the HMD is preferred for tasks involving 3D content, the desktop for creating the program structure and parametrization, and the smartphone for mobile parametrization. However, the device selection depends on individual user characteristics and their familiarity with the devices.**

This work was supported by the Ministry of Economic Affairs of the state Baden-Württemberg in Germany under AI Innovation Center (Learning Systems and Cognitive Robotics) – Grant No. 017-180036; by the Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (BMUV) within the project "Desire4Electronics" - Grant No. 67KI31054A; and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany´s Excellence Strategy – EXC 2120/1 – 390831618.

[1]J. Krieglstein, J. Kolberg and W. Kraus are with the Fraunhofer Institute for Manufacturing Engineering and Automation IPA, 70569 Stuttgart, Germany {jsk, jsk-jk, wek}@ipa.fraunhofer.de

[2]J.Krieglstein, A. Sousa Calepso and M. Sedlmair are with the Visualization Research Center (VISUS), University of Stuttgart, 70569 Stuttgart, Germany aimee.sousacalepso@visus.uni-stuttgart.de, michael.sedlmair@visus.uni-stuttgart.de

I. Introduction

Programming robots, especially for complex assembly tasks, requires expert knowledge and is time-consuming and error-prone. At the same time, the need for more straightforward robot programming is growing, especially for small and middle-sized enterprises [1], due to personnel costs, lack of professionals, quality and traceability requirements, and potentially small lot sizes. One approach to make robot programming more efficient is skill-based programming: Reusable skills, with arbitrary abstraction levels, can be composed and then only require parametrization [2], [3]. However, this process is still difficult for complex tasks and requires deep knowledge of the robot programming language.

In recent years, the development of Augmented Reality (AR) head-mounted displays (HMDs) has grown significantly. AR allows immersive visualization of 3D content within the target environment, which can bridge the gap between the programming environment and the actual programming task. It enables the operator to work mobile and hands-free within the (often complex) robot cell environment. While AR shows advantages in the domain of 3D robot simulation [4], [5], AR user interfaces (UIs) come with other challenges: most people are not used to it, and interactions like, e.g., text entry using a holographic keyboard, are still very challenging and time-consuming compared to a classical user interface [6].

We are interested in how these issues could be addressed

by merging the advantages of the different technologies. To this end, we contribute a novel hybrid user interface for robot programming, consisting of a 2D desktop UI, a smartphone application, and an AR HMD, towards the next generation of industrial robot programming interfaces. The main idea is that users can choose the appropriate device for the respective sub-task and switch seamlessly between the devices. We evaluated the interface in an exploratory user study with six domain experts on typical assembly tasks. The expert study allowed us to collect insights on different device combinations and derive research implications for future AR-assisted robot programming interfaces in an industrial context.

II. Background & Related Work

There are versatile ways to create robot programs for industrial robotic arms, the most common of which are offline programming, learning programs in simulation, programming-by-demonstration, and online programming approaches like the classical teach pendant programming [7]. Although learning- and demonstration-based approaches have been studied for decades, they still struggle with the demands of industrial robot applications, such as reliability and robustness [8]. Thus, manually programming robots is still the go-to solution, which is often more intuitive and time-efficient in industrial environments. Our work focuses on the method of mixed offline/online programming using a skill-based programming framework like pitasc [9] but also applies to other offline/online programming approaches.

A. AR-based user interfaces for robotics

While AR has already been used for more than a decade for visualization during robot programming [10]–[14], advances in technology have made AR HMDs more popular. Devices have become cheaper and more ergonomic, expanding the applications and use cases that can be implemented with them.

Most applications in robotics focus on trajectory planning and visualization. Quintero et al. [15] introduced an AR user interface allowing for trajectory creation and visualization as well as online reprogramming, reporting reduced teaching time and better performance but a higher mental workload compared to kinesthetic teaching. Gadre et al. [4] included gripper actions, as well as a task list where the user can create sequential programs from a set of robot actions and evaluate them on a pick-and-place task. The results showed a lower cognitive workload, higher usability, and higher naturalness with the AR HMD interface. Ostanin et al. [16] introduced spatial markers for trajectory waypoint definition and workspace visualization. Rivera-Pinto et al. [17] used the HMD spatial perception for collision avoidance and compared two trajectory teaching methods. Krieglstein et al. [18] applied a full-featured AR user interface to skill-based robot programming, including frame teaching, program composition, simulation, and execution on the real robot, as well as visualizing skill parameters like velocity- or force-controlled axes situated in 3D space.

While these works showed that AR can beneficially be used for robot programming, they do not provide a concept of how AR can be integrated into existing robot programming workflows, given today's ergonomic and usability issues with AR HMDs [19].

B. Hybrid user interfaces

It has been shown that smartphones and tablets can help users to navigate through HMD AR/VR visualizations since most people are used to smartphones nowadays, and the precise touch resolution and physical touch feedback. Butscher et al. [20] used AR-visualization from an HMD, along with a touch-sensitive tabletop for visualizing and analyzing multidimensional data. They found that touch input is more fluid and precise than a gesture-based system. Surale et al. [21] used a 3D-tracked multi-touch tablet as an input device within an immersive VR environment for solid 3D modelling, finding advantages of the touch-based input and disadvantages in handling and 3D-tracking of the tablet. Similarly, Vock et al. [22] used a smartphone and an AR HMD together with multimodal interaction, namely head gaze and voice input. They found that smartphone control is "reasonable" and "pleasant" due to familiarity with such devices, and it is particularly useful in writing text.

Jansen et al. [23] combined a Virtual Reality (VR) HMD with a desktop view in an analytics tool for automotive user interface evaluation. They found that the desktop 3D view was advantageous for overview tasks, while the VR view showed advantages in detailed passenger movement analysis due to a better perception of spatial distances and highlights the need for a seamless transition between the devices to reduce disorientation. Most recently, Lunding et al. [24] proposed a hybrid user interface for authoring virtual content for AR-supported human-robot collaboration, combining traditional inputs (e.g., desktop computer, mobile touch-screen device) with in-situ inspection with a head-mounted display.

The existing works have shown that hybrid interfaces are actively proposed and discussed since they can address the issues of today's AR HMDs. The possibility of combining novel AR techniques with existing, well-designed and optimized desktop UIs could be a good fit for the challenges in industrial robot programming. To the best of our knowledge, there exists no application of a hybrid user interface for robot programming yet.

III. System Design

A. Application Design

As a baseline interface, we use a classical 2D desktop app with a mouse and keyboard interface for the given skill-based robot programming software pitasc [25]. For situated 3D visualization and trajectory definition, as well as mobile and hands-free work inside or outside the cell, we choose an AR HMD app. Interaction with AR UIs is sometimes tricky, particularly when entering text or numbers with the holographic keyboard. Thus, a smartphone app extends the

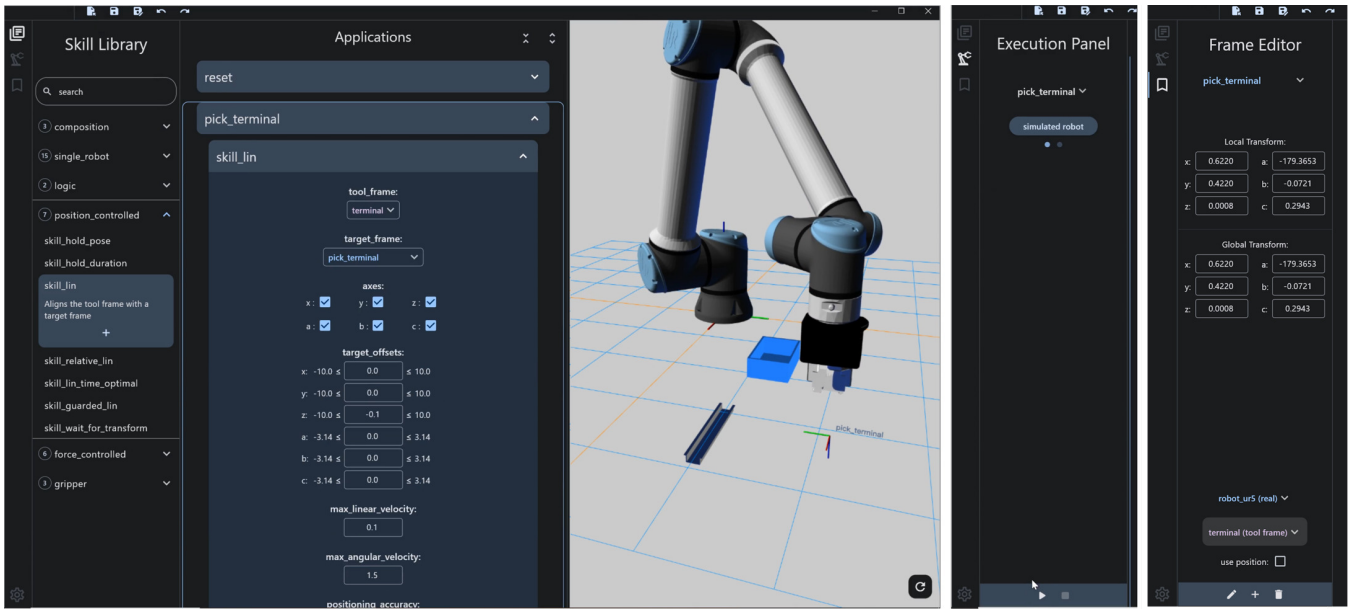

Fig. 3: Desktop application design. In the skill parametrization panel (left), skills can be added from the skill library and parametrized. The 3D scene (center) shows the existing coordinate frames and the robot simulation, which can be started from the execution panel. The frame editor panel (right) allows for frame creation and editing.

AR HMD app for more familiar text and number input, also addressing the mobile aspect of HMDs.

To reduce complexity and learning time, we choose an asynchronous approach, i.e., no simultaneous usage of different devices is required. The applications are always synchronized so all devices are updated on user input. Thereby, users can decide at any time which device they want to use; they are not being forced to one device.

2D Desktop App: The desktop app (see Fig. 3) orients towards commercial state-of-the-art software like ArtiMinds [26] and Intrinsic Flowstate [27]. The main panel is the skill parametrization panel, where the user can compose and parametrize the skills by expanding on click. Adding skills can be performed from the skill library panel, where the user can search and add skills. The frame editor panel allows for frame creation and editing. The execution panel allows for execution on the real robot or in simulation in the 3D scene on the right, containing a 3D view of the virtual robot cell and the digital twin of the robot.

AR HMD App: The AR interface is based on Krieglstein et al. [18]. Like in the desktop app, a panel to compose and add skills from a skill library opens up after clicking on the *add* button. The skill parametrization panel opens when clicking on the respective skill (see Fig. 4). For (rough) frame teaching and validating the robot's kinematic configuration at the taught pose, the UI includes a frame teaching mode, in which the user can drag the end effector of the virtual robot to its desired pose (see Fig. 5). The hardware-depending accuracy of the holograms limits the frames' accuracy, but, depending on the given task, this is still enough, e.g., for pre-positioning and trajectory waypoints, and allows for faster teaching than using the robot teach panel. In the execution

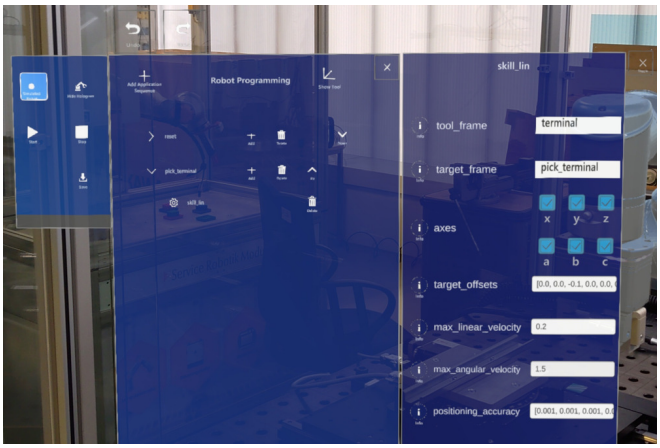

Fig. 4: AR HMD app design. The skill parametrization panel (center) allows for skill composition and skill parametrization (right). The user can start the robot simulation and the real robot in the execution panel (left).

panel, the user can start applications either in simulation or on the real robot. The holographic robot allows for situated in-place visualization and validation in the real robot cell.

Smartphone App: The smartphone app extends the AR UI with the following features: It detects when the user edits a text or number field and pops up the respective text field on the smartphone, allowing for typing with the smartphone keyboard (see Fig. 5). The entered letters are synchronized with the HMD UI, so the user can switch the interface anytime. Furthermore, once a skill is selected on the HMD for parametrization, it is automatically opened on the smartphone, such that the whole parametrization can also

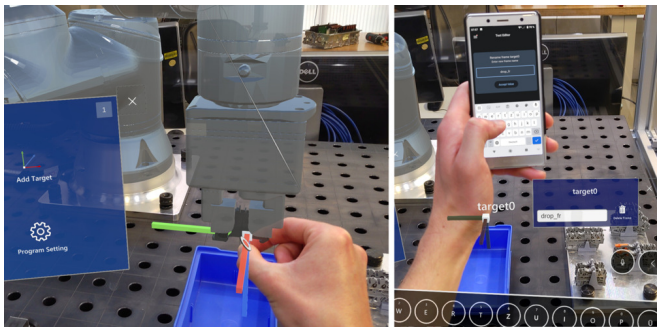

Fig. 5: The frame teaching procedure with the HMD app. The user drags the end effector to the respective spatial position and presses the "Add Target" button (left). The frame can be renamed by clicking on the name (right). Using the smartphone app, the user can avoid the holographic keyboard for faster input. The text input is synchronized between the HMD and the smartphone app. Robot cell and HMD are aligned using a QR marker.

be performed on the smartphone.

B. Architecture & Implementation

To allow for multi-device programming, we propose a server-client architecture for robot programming (see Fig. 6). The server holds the instance of the program and provides an interface via GraphQL on a ROS [28] service. It provides queries for fetching the pitasc applications and the skill library, setting parameter values, adding, moving and removing skills and applications, and undoing/redoing.

The clients fetch the current program and show it to the user. If the user performs changes, i.e., sets a parameter or adds a skill, the client asks the server to change this parameter. If the server accepts the change, all clients are informed about the changed parameter(s) and fetch the changes. Thus, all clients are always up-to-date and show the same program state.

This architecture also enables multi-user applications, which could be particularly interesting for educational purposes, e.g., for robot programming trainings including HMDs with AR technologies, and for shared or collaborative programming approaches.

IV. Evaluation

Equipped with this HUI system, we are now interested in exploring how robot programmers would use the different devices on realistic tasks, and whether their combination in a HUI provides benefits to them. We seek exploration, insights, and qualitative expert feedback for further development and possibly later industrial usage. We want to see which device the users prefer for which task and which challenges the users are facing with the usage of HUIs.

With these goals in mind, we opted to conduct an exploratory study with a small number of expert users, i.e., experienced programmers with the skill-based pitasc software. Such expert studies are recommended by modern human-computer interaction literature for our purpose [29]–[32], as they are optimized for ecological validity and realism [33]. Following this literature, we actively chose not to follow a classical controlled experiment with Null Hypothesis Significance Testing (NHST) [34]. NHST would have come with significant drawbacks in our case: the number of available expert participants is limited, and we thus would have needed to invite novice users as well; tasks would have thus needed to be simplified, resulting in a substantial reduction of ecological validity; quantitative performance measures would be mainly governed by current technical limitations of AR HMDs, disguising the true underlying effects [35]; and, strict hypothesis testing would have come at the cost of reduced possibilities for exploration and insights [36].

A. Study Design

Setup: The experiment is conducted at a safe robot cell with a Universal Robots UR10e robot arm. Fig. 1 shows the setup for the study. We choose a terminal mounting task as a typical assembly task, preferably solved using force-controlled assembly skills. Fig. 7 shows the cell setup for the tasks, i.e., the robot, the top hat rail, the terminal clamp, and the box to throw the terminal into.

Tasks and Procedure: First, the participants familiarize themselves with the devices one after another in a tutorial. They create an application to pick a terminal from the tray, name the application, add skills to it, parametrize them, and teach a frame to which the robot can move after picking. By sequentially introducing the three different devices step-by-step, they learn all critical aspects of the system.

To explore the singular usage of the devices, then, the participants perform a simple placing task in two conditions: once using the 2D desktop application (condition A) and once using AR HMD and smartphone (condition B). The task is to throw the terminal into a box. The order alternates for each participant.

In the final task, all devices can freely be used (condition C). To engage the users in choosing the most appropriate device for programming, this task is more complex compared to the previous stages. The task is the actual mounting of the terminal on the mounting rail, requiring complex assembly skills.

Interview: For collecting qualitative expert feedback, we conducted personal interviews after the last task, asking the following questions:

Q1. Could you imagine using this system in your daily work?
Q2. Do you think having the hybrid setup helped perform the tasks?
Q3. Which interface did you prefer and why?
Q4. What was the hardest part about performing the tasks?
Q5. What do you think can be improved in general?

The answers were evaluated using thematic analysis [37].

Participants: Six participants (one female and five male) took part in the study, with an average age of $M = 29.5$ years ($SD = 2.9$). All participants were experts in robot programming (self-rated experience of $M = 4.2$ ($SD = 0.7$) on a scale of 1 (low) to 5 (high)) and had experience with the

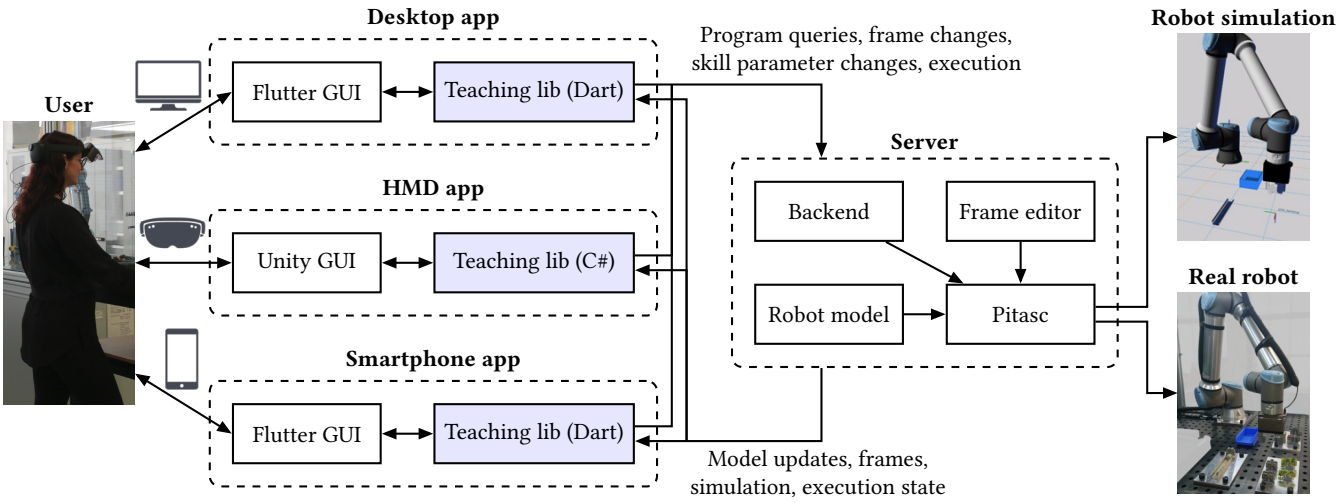

Fig. 6: Proposed software architecture. The user interacts with the GUI component of the respective device app. The Teaching lib component processes the user input and sends it to the server. On changes, the server, respectively the changed components, send updates to the clients, which are received by the Teaching lib component and provided to the user through the respective GUIs.

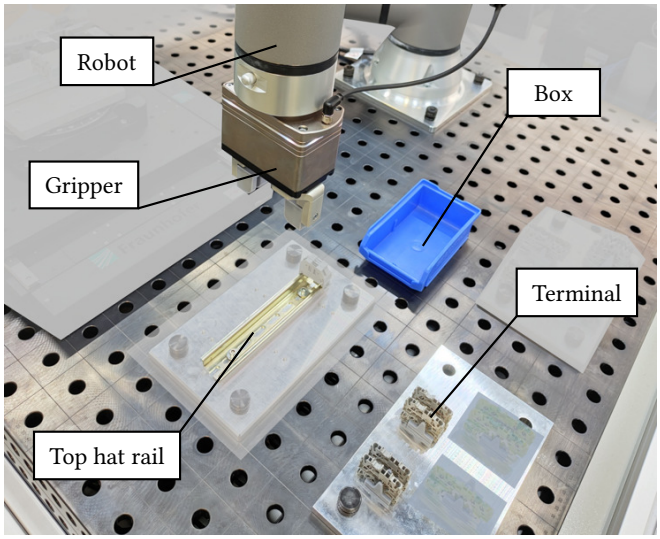

Fig. 7: The cell layout for the assembly task. The terminal clamp has to be picked up and thrown into the box (simple task). In the complex task, it has to be mounted on the top hat rail.

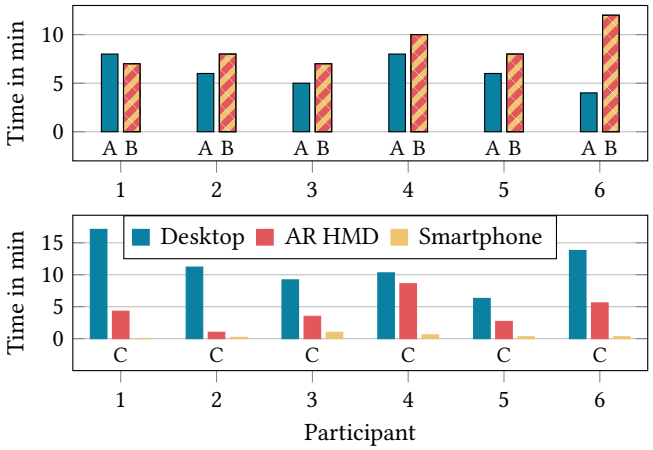

Fig. 8: Absolute device usage for all conditions for each participant.

pitasc system (self-rated experience of $M = 3.2$ ($SD = 1.5$)). The self-rated AR experience was $M = 2.0$ ($SD = 0.9$).

B. Results

All participants completed all tasks successfully. The average time for condition A (desktop) was $M = 6.2$ minutes ($SD = 1.6$), and for condition B (HMD and smartphone) $M = 8.7$ minutes ($SD = 2.0$). For condition C, the complex task where all devices could be used, the average time was $M = 18.7$ minutes ($SD = 5.9$). The average relative device usage in condition C was $M = 70\%$ ($SD = 13$) desktop app, $M = 27\%$ ($SD = 12$) AR HMD app, and $M = 3\%$ ($SD = 2.5$) the smartphone app. Fig. 8 shows the absolute device usage for all conditions. Interestingly, both the absolute and relative device times vary considerably between the participants, even for similarly rated robot programming and pitasc experience (Participants P4, P5, P6).

Regarding usability in daily work (Q1), the participants stated that they can imagine using the system in a more sophisticated and less prototypic state, since they see benefits compared to a singular interface (*"I think it was a good additional degree of freedom to do things with the tool I consider to be the best fit."* (P6)); nevertheless, P1 and P2 reported concerns about the practical applicability compared to traditional programming methods.

Participants were primarily positive regarding the usefulness of the HUI for completing the task (Q2, 5x yes, 1x no (P2)). Regarding the preferred interface, all participants

prefer the desktop application as a singular interface, mentioning efficiency (P1, P3, P4, P5), pleasantness (P2, P3, P5) and convenience (P1, P2, P6) as reasons.

Concerning the most suitable device for a particular subtask (Q3), all participants agreed that the HMD was best suited to teach new (rough) frame positions; desktop or smartphone were considered more helpful for adjusting the orientation (P1, P6). Additionally, the HMD was stated to be great for visualizing the movement of the simulated robot in the real world (P1, P4). Most participants (P2, P4, P5, P6) explicitly mentioned that the desktop application was best suited for creating the application structure. Other explicitly mentioned benefits were the overview (P3), the interaction (P2), and the exact value input (P1, P2). The smartphone application was disclosed to work well for acceptable tuning frame positions (P1, P2, P6). Its functionality was controversial and ranged from *"useless"* (P1, P3, P4) to *"I could imagine creating whole application structures with it"* (P5). P3 and P4 stated that the smartphone application would be very helpful if the desktop application were unavailable or not so easy to reach, e.g., within a bigger robot cell.

Regarding Q4 and Q5, some participants reported usability issues with the HMD (*"inconvenient"* (P2, P3), *"difficult handling of virtual objects"* (P5), *"the virtual keyboard is a mess"* (P3)). P1 stated that after some learning time, it became more straightforward to use. P2 also mentioned that *"Somebody who is more socialized with a mouse and keyboard will probably prefer the desktop application. Someone who grew up with smartphones and tablets will probably prefer the smartphone application. Finally, someone more into AR and VR will probably prefer the HMD application"*. P6 reported issues with reflections on the smartphone when looking at it through the HMD.

V. Discussion

The results, on the one hand, indicate that AR is helpful when it handles actual 3D content. Participants mentioned the frame teaching in AR as smooth and beneficial since it allows for defining the frames directly at their pose inside the real cell, being much more natural than on a desktop interface and faster than jogging the real robot to the desired pose. Furthermore, the situated robot simulation was considered helpful for visualization of the robot program since the robot program is shown in-place inside the cell, making the simulation very close to as if the real robot was moving, which aligns with the findings of Diehl et al. [5]. On the other hand, some participants reported usability issues with the HMD, as they avoided the text and number input on the HMD with Microsoft's "Mixed Reality Keyboard", which is a commonly known issue [38], [39] that probably decreases with technical progress and proliferation of HMDs. It is also well-known that the rating of HMDs varies with individual user characteristics [40], [41], which presumably caused the varying device usage proportions (see Fig. 8). These individual preferences also apply to the smartphone, whose usefulness was highly controversial. As P3 and P4 stated, it could be more useful in applications requiring a high level of mobility, which also applies to the HMD.

For tasks like creating the program structure and parametrization, both tasks without any 3D content, all participants preferred the desktop environment. This might relate to their experience with existing sophisticated UIs, and it shows that keeping the strengths of existing interfaces can also reduce reservations about new interfaces. All in all, five out of six participants found the HUI beneficial over a singular user interface, which shows that also for robotics, hybrid user interfaces are a valuable chance to incorporate HMDs into today's workplaces.

Limitations: Hybrid user interfaces require some kind of server-client architecture, which mostly does not exist in today's robot programming software, complicating the transition to a hybrid user interface. The recent browser-driven Intrinsic Flowstate software [27] is an exception, indicating that changes might be expected in the future. Furthermore, from a usability perspective, the design space of HMD applications has to be explored further. As two participants stated, a consistent design of the desktop application and the HMD app would help to switch between the devices more seamlessly. It would also help to increase familiarity with the HMD app, which has to be considered for future commercial applications, as well as the support of different devices (cross-device) [42], [43].

VI. Conclusion

We explored how a hybrid user interface consisting of a desktop application, an HMD user interface, and a smartphone app can benefit robot programming for complex tasks. We proposed a software architecture based on a server-client pattern and designed and implemented the first HUI for industrial robot programming. In an exploratory expert user study, we evaluated the HUI on typical and realistic assembly tasks. The insights from domain experts indicate that for sub-tasks like creating the program structure and basic parametrization, a classical desktop app is still preferred, while an HMD is beneficial for sub-tasks which require spatial and situated visualization and interaction. Therefore, the combination of the devices within a HUI can capitalize on the strengths of each technology. However, the acceptance and usefulness of each device is user-specific. Giving users the chance to choose the device that fits their needs most for each sub-task has the potential to increase user experience and programming efficiency, and reduce programming errors.

Future work should further explore the aspect of frame teaching, incorporating computer vision techniques to detect features like planes, edges, and holes and offer possibilities to, e.g., align a frame with the detected geometry. Increasing the tracking accuracy by, e.g., using the existing model of the robot and the cell would increase the utility of the results. Furthermore, the situated visualization can be further examined to visualize different possible trajectories, combined with reachability and singularity analysis. Addressing these challenges could pave the way for AR-assisted robot programming in an industrial context.

References

[1] M. Teulieres, J. Tilley, L. Bolz, P. M. Ludwid-Dehm, and S. Wägner, "Industrial robotics: Insights into the sector's future growth dynamics," McKinsey Company, Ed, 2021.

[2] L. Halt, F. Nägele, P. Tenbrock, and A. Pott, "Intuitive constraint-based robot programming for robotic assembly tasks," in *IEEE International Conf. Robotics and Automation*, 2018, pp. 520–526.

[3] M. R. Pedersen, L. Nalpantidis, R. S. Andersen, C. Schou, S. Bøgh, V. Krüger, and O. Madsen, "Robot skills for manufacturing: From concept to industrial deployment," *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 282–291, 2016.

[4] S. Y. Gadre, E. Rosen, G. Chien, E. Phillips, S. Tellex, and G. Konidaris, "End-user robot programming using mixed reality," in *IEEE International Conf. Robotics and Automation*, 2019, pp. 2707–2713.

[5] M. Diehl, A. Plopski, H. Kato, and K. Ramirez-Amaro, "Augmented reality interface to verify robot learning," in *IEEE International Conf. Robot and Human Interactive Communication*, 2020, pp. 378–383.

[6] J. L. Derby, C. T. Rarick, and B. S. Chaparro, "Text input performance with a mixed reality head-mounted display (hmd)," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 63, no. 1, 2019, pp. 1476–1480.

[7] O. Heimann and J. Guhl, "Industrial robot programming methods: A scoping review," in *IEEE International Conf. Emerging Technologies and Factory Automation*, vol. 1, 2020, pp. 696–703.

[8] L. M. Sanneman, C. K. Fourie, and J. A. Shah, "The state of industrial robotics: Emerging technologies, challenges, and key research directions," *Found. Trends Robotics*, vol. 8, pp. 225–306, 2020.

[9] F. Nägele, L. Halt, P. Tenbrock, and A. Pott, "A prototype-based skill model for specifying robotic assembly tasks," in *IEEE International Conf. Robotics and Automation*, 2018, pp. 558–565.

[10] B. Akan, A. Ameri, B. Çürüklü, and L. Asplund, "Intuitive industrial robot programming through incremental multimodal language and augmented reality," in *IEEE International Conf. Robotics and Automation*, 2011, pp. 3934–3939.

[11] A. Gaschler, M. Springer, M. Rickert, and A. Knoll, "Intuitive robot tasks with augmented reality and virtual obstacles," in *IEEE International Conf. Robotics and Automation*, 2014, pp. 6026–6031.

[12] S. Ong, A. Yew, N. Thanigaivel, and A. Nee, "Augmented reality-assisted robot programming system for industrial applications," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101820, 2020.

[13] D. Ni, A. Yew, S. Ong, and A. Nee, "Haptic and visual augmented reality interface for programming welding robots," *Advances in Manufacturing*, vol. 5, pp. 191–198, 2017.

[14] J. Lambrecht and J. Krüger, "Spatial programming for industrial robots based on gestures and augmented reality," in *IEEE/RSJ International Conf. Intelligent Robots and Systems*, 2012, pp. 466–472.

[15] C. P. Quintero, S. Li, M. K. Pan, W. P. Chan, H. M. Van der Loos, and E. Croft, "Robot programming through augmented trajectories in augmented reality," in *IEEE/RSJ International Conf. Intelligent Robots and Systems*, 2018, pp. 1838–1844.

[16] M. Ostanin, S. Mikhel, A. Evlampiev, V. Skvortsova, and A. Klimchik, "Human-robot interaction for robotic manipulator programming in mixed reality," in *IEEE International Conf. Robotics and Automation*, 2020, pp. 2805–2811.

[17] A. Rivera-Pinto, J. Kildal, and E. Lazkano, "Toward programming a collaborative robot by interacting with its digital twin in a mixed reality environment," *International Journal of Human–Computer Interaction*, pp. 1–13, 2023.

[18] J. Krieglstein, G. Held, B. A. Bálint, F. Nägele, and W. Kraus, "Skill-based robot programming in mixed reality with ad-hoc validation using a force-enabled digital twin," in *IEEE International Conf. Robotics and Automation*, 2023, pp. 11 612–11 618.

[19] W. Fang, L. Chen, T. Zhang, C. Chen, Z. Teng, and L. Wang, "Head-mounted display augmented reality in manufacturing: A systematic review," *Robotics and Computer-Integrated Manufacturing*, vol. 83, p. 102567, 2023.

[20] S. Butscher, S. Hubenschmid, J. Müller, J. Fuchs, and H. Reiterer, "Clusters, trends, and outliers: How immersive technologies can facilitate the collaborative analysis of multidimensional data," in *Proc. CHI Conf. Human Factors in Computing Systems*, 2018, pp. 1–12.

[21] H. B. Surale, A. Gupta, M. Hancock, and D. Vogel, "Tabletinvr: Exploring the design space for using a multi-touch tablet in virtual reality," in *Proc. CHI Conf. Human Factors in Computing Systems*, 2019, pp. 1–13.

[22] K. Vock, S. Hubenschmid, J. Zagermann, S. Butscher, and H. Reiterer, "Idiar: Augmented reality dashboards to supervise mobile intervention studies," in *Proc. ACM Mensch und Computer*, 2021, pp. 248–259.

[23] P. Jansen, J. Britten, A. Häusele, T. Segschneider, M. Colley, and E. Rukzio, "Autovis: Enabling mixed-immersive analysis of automotive user interface interaction studies," in *Proc. CHI Conf. Human Factors in Computing Systems*, 2023, pp. 1–23.

[24] R. Lunding, S. Hubenschmid, and T. Feuchtner, "Proposing a hybrid authoring interface for AR-supported human-robot collaboration," in *International Workshop on Virtual, Augmented, and Mixed-Reality for Human-Robot Interactions*, 2024.

[25] Fraunhofer IPA. (2024) pitasc modular system. [Online]. Available: https://www.pitasc.fraunhofer.de/en.html

[26] ArtiMinds Robotics GmbH. (2024) Robot automation: Robotic engineering and low-code programming software ArtiMinds. [Online]. Available: https://www.artiminds.com

[27] Intrinsic Innovation GmbH. (2024) Intrinsic flowstate. [Online]. Available: https://www.intrinsic.ai/flowstate

[28] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *Proc. IEEE International Conf. Robotics and Automation, Workshop on Open Source Robotics*, 2009.

[29] S. Greenberg and B. Buxton, "Usability evaluation considered harmful (some of the time)," in *Proc. SIGCHI Conf. Human Factors in Computing Systems*, 2008, pp. 111–120.

[30] P. Dragicevic, *Fair Statistical Communication in HCI*. Springer International Publishing, 2016, pp. 291–330.

[31] A. Cockburn, P. Dragicevic, L. Besançon, and C. Gutwin, "Threats of a replication crisis in empirical computer science," *Commun. ACM*, vol. 63, no. 8, p. 70–79, 2020.

[32] J. Lazar, J. H. Feng, and H. Hochheiser, *Research methods in human-computer interaction*. Morgan Kaufmann, 2017.

[33] J. E. McGrath, "Methodology matters: Doing research in the behavioral and social sciences," in *Readings in human–computer interaction*. Elsevier, 1995, pp. 152–169.

[34] A. Field and G. Hole, *How to design and report experiments*. SAGE Publications Ltd, 2002.

[35] A. S. Calepso, P. Fleck, D. Schmalstieg, and M. Sedlmair, "Exploring augmented reality for situated analytics with many movable physical referents," in *Proc. ACM Symp. Virtual Reality Software and Technology*, 2023, pp. 1–12.

[36] G. Cumming, *Understanding the new statistics: Effect sizes, confidence intervals, and meta-analysis*. Routledge, 2013.

[37] V. Braun and V. Clarke, *Thematic analysis.* American Psychological Association, 2012.

[38] M. Speicher, A. M. Feit, P. Ziegler, and A. Krüger, "Selection-based text entry in virtual reality," in *Proc. CHI Conf. Human Factors in Computing Systems*, 2018, pp. 1–13.

[39] M. Schenkluhn, C. Peukert, A. Greif-Winzrieth, and C. Weinhardt, "Does one keyboard fit all? comparison and evaluation of device-free augmented reality keyboard designs," in *Proc. ACM Symp. Virtual Reality Software and Technology*, 2023, pp. 1–11.

[40] P. Kortum and F. L. Oswald, "The impact of personality on the subjective assessment of usability," *International Journal of Human–Computer Interaction*, vol. 34, no. 2, pp. 177–186, 2018.

[41] I. B. Lima and W. Hwang, "Effects of heuristic type, user interaction level, and evaluator's characteristics on usability metrics of augmented reality (ar) user interfaces," *International Journal of Human–Computer Interaction*, vol. 40, no. 10, pp. 2604–2621, 2024.

[42] M. Nebeling, T. Mintsi, M. Husmann, and M. Norrie, "Interactive development of cross-device user interfaces," in *Proc. SIGCHI Conf. Human Factors in Computing Systems*, 2014, pp. 2793–2802.

[43] M. Speicher, B. D. Hall, A. Yu, B. Zhang, H. Zhang, J. Nebeling, and M. Nebeling, "Xd-ar: Challenges and opportunities in cross-device augmented reality application development," *Proc. ACM Human-Computer Interaction*, vol. 2, no. EICS, pp. 1–24, 2018.