SPS2022
A.H.C. Ng et al. (Eds.)

doi:10.3233/ATDE220143

Augmented Reality Approach for a User Interface in a Robotic Production System

Bernard SCHMIDT [a,1], Adrian SÁNCHEZ DE OCÃNA TORROBA [a], Göran GRAHN [a], Ingemar KARLSSON [a] and Amos H.C. NG [a,b]
[a] *School of Engineering Science, University of Skövde, Sweden*
[b] *Division of Industrial Engineering & Management, Uppsala University, Sweden*

Abstract. In a Cyber-physical system, the information flow from the cyber part to the physical part plays a crucial role. This paper presents the work of development and initial testing of an augmented reality approach to provide a user interface for operators that could be a part of a robotic production system. The solution is distributed and includes a communication hub that allows the exchange of data and information between multiple clients e.g. robot controllers, an optimization platform, and visualization devices. The main contributions of the presented work are visualization of optimization results and visualization of information obtained from the robot controller and the integrated communication framework. The paper also presents challenges faced during the development work and opportunities related to the presented approach. The implemented interface uses HoloLens 2 mixed reality device to visualize in real-time information obtained from a robot controller as well as from simulation. Information regarding the placement of work objects and targets or currently executed lines of code can be useful for robotic cell programmers and commissioning teams to validate robot programs and to select more optimal solutions toward sustainable manufacturing. The operator can simulate the execution of the robot program and visualize it by overlying the robot cell with the 3D model of the simulated robot. Moreover, visualization of future robot motion could support human-robot collaboration. Furthermore, the interface allows providing the user with details from multi-objective optimization performed on a digital twin of the robotic cell with the aim to reduce cycle time and energy consumption. It allows visualizing selected scenarios to support decision-making by allowing comparison of proposed solutions and the initial one. The visualization includes cell layout, robot path, cycle time, robot energy consumption. The presented approach is demonstrated in industry-inspired cases and with the use of an industrial ABB robot.

Keywords. robot cell, augmented reality, mixed reality, optimization

1. Introduction

In a Cyber-physical system, the information flow from the cyber part to the physical part plays a crucial role, and the human operator is an essential part of the system. In the report [1] European Commission presents Industry 5.0 that complements the Industry 4.0 paradigm to make the sustainable, resilient and human-centric industry a reality. One way of achieving it is by developing innovative technologies in a human-centric way that could support and empower workers.

[1] Bernard Schmidt, Corresponding author, School of Engineering Science, University of Skövde, PO Box 408, 541 28 Skövde, Sweden, Sweden; E-mail: bernard.schmidt@his.se.

Targeting this transformation, the goal of the presented research is to provide interface and support to various users of robotics applications. The approach is toward intuitive robot programming and configuration with consideration of optimality aspect, including energy efficiency and cell awareness in human-robot collaboration. In the paper, we report the initial work done on the integration of various technology.

In literature, there are several examples of usage of Augmented Reality (AR) and Mixed Reality (MR) in industry. A survey [2] overview AR smart glasses for assembly operators in the manufacturing industry. Honig et al. [3] described the advantages of using MR in robotics, pointing out easy debugging and a reduced gap between simulation and implementation as the main benefits in the context of robotic applications. Lambrecht et al. [4] demonstrated that AR-based robot programming could be more efficient than classic teach-in and offline programming and described the possibilities unveiled by combining the benefits of online and offline programming into an AR system [5]. Makris et al. [6] presented an augmented reality system for operator support in human-robot collaborative assembly. Ostanin and Klimchik [7] introduced an MR robot programming system for industrial robots. Moreover, eye-tracking can make it possible to connect holograms with the user's range view and identify the user's intention. This can be used in Human-Robot Interaction, and [8] presented a new method of interaction with the robot using eye gazing and head gestures combined with deep learning in MR. These possibilities in MR devices result in new evaluation capabilities that robot programmers and operators can use for testing different robotic scenarios in the cell virtually before physical implementation.

AR is also applied to other aspects of manufacturing than providing support for operators, e.g., Karlsson et al. [9] combined augmented reality and simulation-based optimization for decision-maker support in manufacturing.

In the optimization context, several works have been published on the optimization of robotic cells, e.g. [10-11]. However, not much could be found about the visualization of robotics optimization results using augmented/mixed reality.

The main contributions of the presented work are visualization of optimization results and visualization of information obtained from the robot controller and the integrated communication framework. The paper also presents challenges faced during the development work and opportunities related to the presented approach.

The rest of the paper is structured as follows. Section 2 introduces our concept; Section 3 provides implementation details; Section 4 presents results from case studies; and finally, Section 5 concludes the paper, overviews faced challenges, and highlights our future work.

2. Concept Overview

The presented concept integrates industrial robots, industrial offline programming (OLP) and simulation tool, optimization platform, and mixed reality interface to support robot users. Targeted robot users are operators that work alongside or in collaboration with robots; robot programmers that need to program or re-program the robot; and decision-makers that work with designing and improving robotic cell or robotic production lines, see Figure 1.

Operators can benefit from the visualization of future robot motions and safety zones that might reduce human stress and increase "safety feeling" [6]. Robot programmers and engineers could be provided with visualization of targets, work objects, and safety zones as well as visualization of robot motion within the physical cell e.g. to validate introduced code changes before deploying and testing it on a real robot. Finally, solutions from optimization could be simulated and evaluated to support the decision on the selection of solutions when designing a new or improving an existing robot cell. As simulation-based optimization might use a simplified model of the cell it is important to validate the solution before the final selection is made. In the approach, the decision-maker can send a preliminary selected solution to the automation engineer on the side to assess its feasibility in the physical layout.

An overview of the framework with indicated different scenarios is illustrated in Figure 2. Operators will benefit from scenarios 1 and 2, robot programmers could make use of scenarios 1 to 3, while decision-makers will be involved in scenarios 3 and 4.

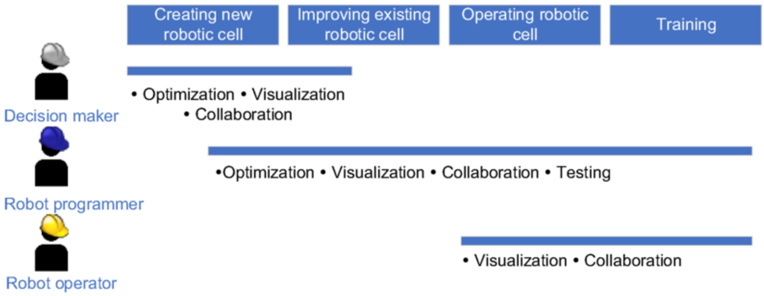

Figure 1. Different users and use cases of presented concept.

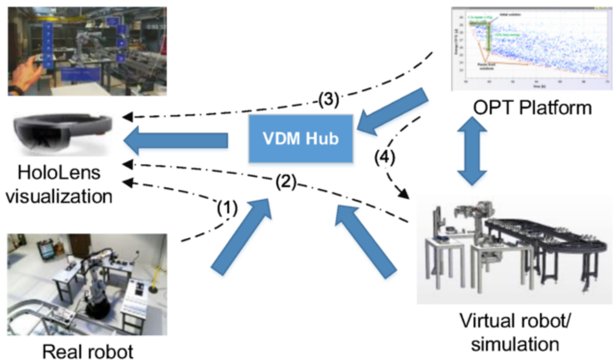

Figure 2. Overview of the framework. Black dash-dotted lines indicate different modes of visualization (1) visualization of real robot execution; (2) visualization of simulation/virtual controller; (3) visualization of optimization results, pre-recorded simulation; (4) triggering simulation/visualization for selected parameters

2.1. Augmented Reality

The Microsoft HoloLens 2 (HL) augmented/mixed reality glasses are utilized in the presented approach. It supports immersion and intuitive interactivity. Users can select and position holograms with the use of direct touch or hand rays for distant holograms. There are different interactivity gestures, such as touch, air-tap, and tap-and-hold. HL can also perform eye-tracking. The possibility of showing a virtual representation on top of the real view and combined with interactivity unblock new opportunities in the industry. One of the application areas is robotics.

The presented work aims to create a simple and visual interface for robotic cell users. This graphical interface could test virtually different automated scenarios in the cell and provide decision-making tools and optimization insights. During robot reconfiguration, engineers can simulate the execution of modified code and visualize the robot overlaid with the cell to validate changes, i.e., placement of the work objects, targets, and collisions of the robot arm with cell layout.

To achieve that, the HL needs to be provided with data and information from multiple sources, including robot controllers and simulation software.

2.2. VDM Hub

Virtual manufacturing involves many different techniques and dedicated software. The concept of Virtual Decision-Making (VDM) hub is developed to integrate these software tools and especially simulation-based optimization. The VDM hub makes it possible to send both data and events between the software and the simulation/optimization platform. Not only traditional software can be connected but also visualization hardware like virtual and augmented reality headsets. An example usage is that users can send optimization results to the VDM hub which is distributed to all the nodes that are subscribed to that type of data. The VDM hub can be made accessible over company networks as well as the internet so that users in many locations can interact with each other.

The VDM hub makes it possible for users to interact and cooperate on virtual manufacturing simulation models, both for training and real decision-making cases. In this work, the VDM hub is implemented by an MQTT broker, Eclipse Mosquitto [12].

2.3. Simulation

In the presented approach, to simulate robot cell, ABB RobotStudio (RS) [13] has been selected. It is a state-of-the-art OLP and simulation tool. It makes use of a Virtual Controller (VC) that is an exact copy of the software that runs on a physical robot controller allowing very realistic simulations.

RS provides support for Virtual Reality, however, the AR is currently only supported via export of static models or recorded animations in GLB format the binary version of glTF (GL Transmission Format).

Implemented simulation models are used to provide data for visualization in HL, i.e. joint values for a rendered robot, as well as part of simulation-based optimization, where simulation is used to evaluate objective functions. Simulations are performed for different layouts with modification of path parameters and as output cycle time, energy consumption, and layout area are returned.

2.4. Optimization

Optimization results come from OPT Platform [14]. All evaluations performed during simulation-based multi-objective optimization are stored in internal databases and can be recalled through OPT Browser tool and delivered via VDM Hub. When evaluation results are accompanied with simulation data i.e. robot joint values those could be transmitted to HL and visualized there directly. When joint values are not stored, simulation parameters can be transmitted to RS to repeat simulation and this simulation can be visualized in real-time through HL.

3. Implementation

3.1. Augmented Reality

An AR application was built using the Unity engine. For that purpose, Mixed Reality Toolkit (MRTK) was utilized. It is an open-source project, which shares foundational components, building blocks for interactions, and Graphical User Interface (GUI) controls for building mixed reality experiences in Unity [15].

In Unity, the C# programming language is used for developer scripting. Different C# scripts that set up the kinematics of the virtual model were implemented along with a Unified Robotic Description Format (URDF) importer. The URDF is an XML file format to describe all elements of a robot [16].

The URDF importer was posteriorly substituted for a GLB importer. File in the GLB format stores information such as node hierarchy, cameras, materials, animations, and meshes in binary form resulting in compact file size [17]. Support for GLB files allows importing robot and cell models exported directly from the RS.

In the application different GUI elements from the MRTK were implemented, such as pressable buttons, menus, dialogs, and spline tooltips. Simplicity and clear visibility were prioritized. A rotation script that aligns GUI interfaces with the user's point of view was implemented for this purpose. Different hide and show buttons were also implemented due to the difficulty of showing all the GUI elements in a limited view range.

The virtual representations give more information to the user providing insights that can be helpful for robot programming, debugging, safety and decision-making. The main visualization elements are virtual robot visualization, live command dialogue, joint tooltip, graphs, and robotic targets.

a) The virtual robot can mimic the actual robot path or can visualize an optimized robotic path. The virtual robot anticipating the movement of the actual robot could support safety in a collaborative environment since the predictive movement of the robot could be seen in advance to prevent possible risks. The optimized robotic path could be visualized on the virtual robot as an alternative solution to the robot programmer for decision making.
b) The live command dialogue indicates the current line of code executed by the robot controller. This can allow users to identify different moments on the production cell and connect them to the robot code that could be useful for debugging or verifying introduced changes in the code.
c) The joint information spline tooltip displays the value and speed of the joint. This information panel moves along with the joint.
d) The graphs can display physical and optimized robots data, such as peak power, energy consumption, supporting decision-making, and optimization.
e) The robotic targets can be visualized to understand the robot's path better and predict the robot's subsequent movements.

Apart from the visualization elements, the application contains different menus to interact with the virtual robot. The main features include the possibility of going "live" and visualizing in real-time the robot based on streaming information from the real or virtual robot controller, recording the live data, and storing it on the HL device. It also allows changing to "playback mode," displaying saved information and movements to inspect the visualization further. In addition, it supports receiving data from optimization to improve the decision-making process.

3.2. Integration with Real/Virtual Controller

To provide required data to MR from within RS simulation an RS add-in has been developed in C# using RS SDK 2020.1. It allows obtaining the current status of simulation execution including robot joint position, velocities, and energy and power consumption. Those data are streamed to the VDM hub and could be visualized in HL.

To connect to the real robot controller, as well as virtual controller on a remote machine a stand-alone interface application had been implemented with the use of PC SDK 6.08 that is compatible with Robot Ware installed on the tested ABB IRB2600 robot.

3.3. Interchanging of Data

The data interchanged between RS add-in and HL is performed as messages in JavaScript Object Notation format (JSON), a text-based format for interchanging data. JSON has the advantage of being easy for both humans to read and computers to parse [18]. Figure 3 illustrates how messages are sent from RS add-in through the VDM hub to HL. For sending and receiving messages through the VDM hub, MQTT clients were used. For this purpose, the M2Mqtt library was implemented into RS add-in, and the M2MqttUnity asset was implemented into HL. The M2Mqtt library [19] and the M2MqttUnity asset [20] include foundational components and building blocks for receiving and sending messages to an MQTT broker.

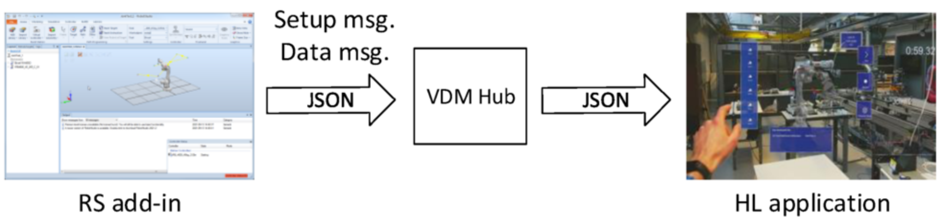

Figure 3. Interchanging information between RS add-in and HL application.

3.3.1. Message Structure

The messages have been designed with the aim of being small. Therefore, properties are omitted when not changed since their previous sent message and are not necessary for identifying the message. Furthermore, two types of messages were defined, setup messages and data messages. Setup messages were used for data that were not regularly transferred, such as metadata and target points. In contrast, data messages were used for data transferred continuously, such as robot joint positions and power consumption. A class structure for these message types was developed in C# and implemented in both RS add-in and HL. Table 1 gives a brief description of the messages' properties with an indication of when a property is message-type specific.

Table 1. Data types used in messages

Property	Data type	Description
Timestamp	DateTime	Message creation date and time.
Robot id	string	String for uniquely identifying a robot.
Message type	string	The type of message being sent, e.g., setup or data.
Target points[1]	List <Target point>	A list with all the target points in the robot path. Each target point includes a vector representing its position and an array representing its rotation.
Property names[1]	Dictionary <string, string>	A dictionary for describing the properties and the text properties sent in the data message
Properties[2]	Dictionary <string, float>	A dictionary for sending numerical values properties
Text properties[2]	Dictionary <string, string>	A dictionary for sending string values properties
Joint angles[2]	Dictionary <string, float>	A dictionary representing the angle values of each joint on the robot

[1] – optional field for setup message
[2] – optional field for data message

3.4. Recording and Replaying

When recording live stream data of a robot's movement, all incoming messages sent to HL are stored in a file in the persistent data path. On HL, the persistent data path can store data between multiple application runs [21]. Once a recording has finished, it can be replayed at once or at a later time. When a recording is replayed, all messages from its file are first loaded into the HL memory and then parsed in the same order and speed as they were created, using the creation date and time property included in each message.

The additional synchronization signal is implemented at the beginning of the robot program to be able to synchronize visualization of multiple pre-recorded visualizations or to synchronize pre-recorded visualization with visualization driven in real-time based on data from the real robot controller.

4. Case Study

The application's functionality is to visualize optimization results and information obtained from the robot controller and the integrated communication framework. This is demonstrated in two case studies. The first case presents a visualization of optimization results, while the second presents a visualization of information from the robot controller.

4.1. Visualization of Optimization Results

The optimization was performed on a model, see Figure 4, of a real-world industrial robotic cell. In the robotic cell orchestrated by one mater PLC (Siemens CPU 315-2PN/DP), an ABB IRB4600 robot with IRC5 controller, running RobotWare 5.15, was responsible for material handling in a process consisting of three operations performed by two CNC machines, operations OP1 (OKUMA LVT300 CNC lathe) and OP2 (OKUMA MB-46VAE machining center), and a dedicated marking station OP3. Parts are transported to and from the cell with input and output conveyors. In the optimization, cell layout, targets, and motion parameters have been modified with the goal to optimize cycle time and energy efficiency. The optimization was performed in OPT Framework [14] with nondominated sorting genetic algorithm II (NSGA-II) [22].

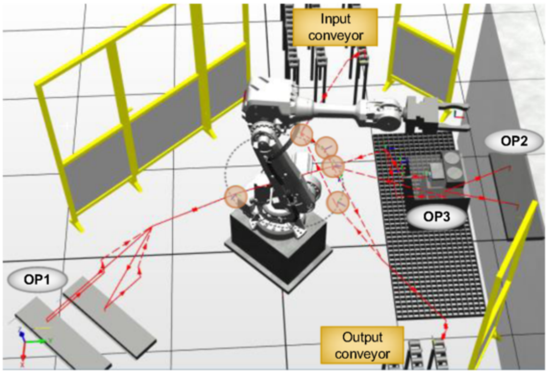

Figure 4. Model of the cell used in the case study

After the simulation-based multi-objective optimization is performed, the decision-maker is presented with a whole range of possible solutions that are placed on an approximation of the Pareto front. In the presented case, the objective space is 4-dimensional and includes cycle time, peak power, energy consumption, and cell area. Figure 5, depict simulation results projected into two objectives space (i.e. time and energy consumption) as presented in OPT Browser (visualization and analysis tool from the OPT framework), at the bottom part of the figure, and visualization of a cell layout for selected solutions obtained from RS, in the top row.

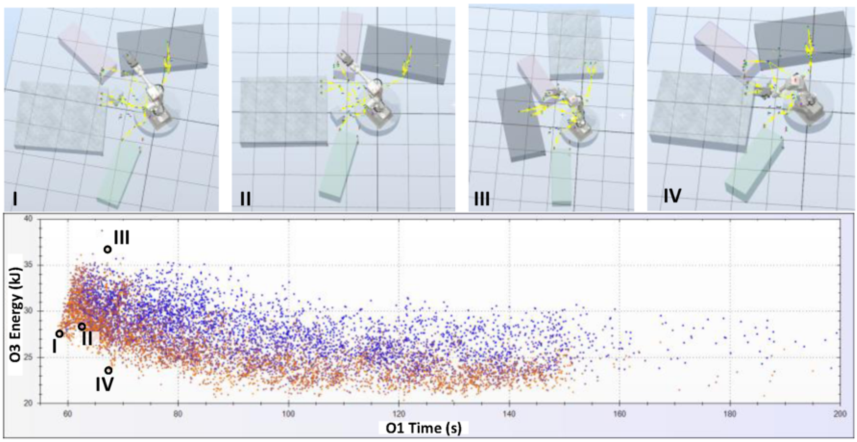

Figure 5. Visualization of the layout of different solutions obtained in RS based on simulation results selected from OPT Browser plot.

Figure 6 presents a visualization in HL of one of the solutions obtained during the optimization process. In this snapshot from the HL video stream, there is a visible part of the screen with OPT Browser, where the selection of a solution to visualize was made and the screen with simulation performed in RS. In this case, RS received simulation configuration through VDM Hub while HL received pre-recorded simulation values. In HL users get a notification which optimization evaluation is currently displayed, see Figure 7.

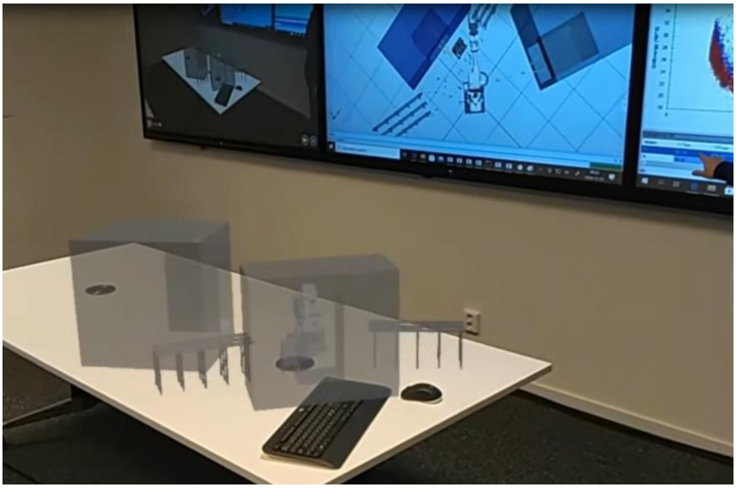

Figure 6. Sending results from optimization to HL.

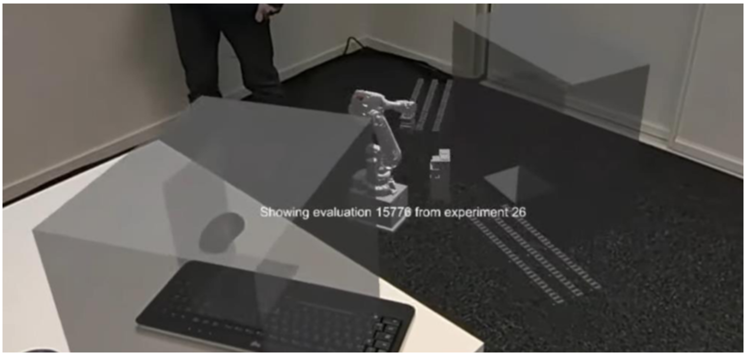

Figure 7. Visualization of selected solution for robotic cell with IRB4600 robot through HL

4.2. Visualization from the Real Controller

In the second case, information was obtained from the real controller with the use of implemented interface. The mixed reality application was tested with an industrial ABB IRB 2600 robot that can be found at Assar Industrial Innovation Arena.

The developed visualization can be seen in the screen grabbed from the HL presented in Figure 8. The main visualization elements are virtual robot visualization, live command dialogue, joint tooltip, graphs, and robotic targets.

a) The virtual robot mimics the actual robot.
b) The live command dialogue indicates the currently executed line of the robot controller code.
c) The joint information spline tooltip displays the joint's position and velocity. This information panel moves along with the joint.
d) The graph displays a trend of the second joint position.
e) The robotic targets are visualized with the use of frames that also indicate orientation.

All these GUI elements can be hidden or displayed depending on the aim of the mixed reality user.

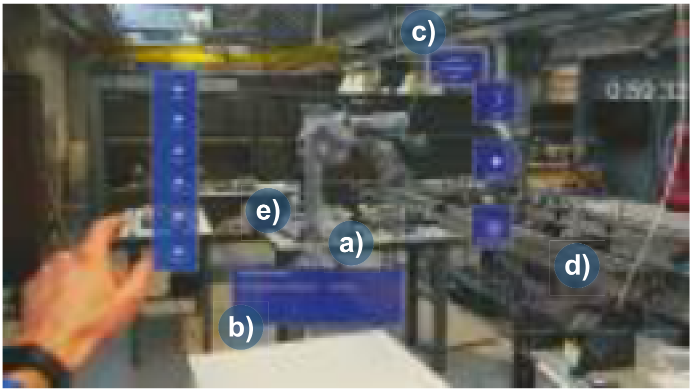

Figure 8. Robotic application for HoloLens 2. Main elements include a) virtual robot visualization, b) live command dialogue, c) joint tooltip, d) graphs and e) robotic targets

5. Conclusion and Future Work

In the paper, we presented a visualization of optimization results and visualization of information obtained from the robot controller and the integrated communication framework. The integrated communication framework includes a VDM hub that allows exchange between HL, optimization platform, robotic OLP, and simulation software as well as a real robot controller. In the case study, optimization of the robotic cell was performed with respect to cycle time, energy consumption, and layout, and selected solutions were transmitted to HL for assessment.

The different visualization elements such as robot targets, command lines, joint information can be helpful for robot programmers and operators. Nevertheless, the possibility of making decisions, doing changes to the robot program, and directly implementing it into the cell using the same HL device will bring the full potential of this technology.

5.1. Challenges

The challenges faced are how to create mixed-reality user interfaces. There are currently very few specific protocols that help guide the design of MR interfaces. Different established guidelines in augmented reality user interfaces and virtual reality best practices can be used as a reference.

One of the challenges of HL development was experiencing performance issues. CPU performances include app logic, including processing input, animations, and physics. On the other hand, the Graphics Processing Unit (GPU) handles the application's graphics transforming 3D into pixels. CPU problems were experienced because of the C# scripts implemented into the Unity project. GPU low performances were also experienced, which can be dramatically affected by using shaders not prepared for mixed reality and 3D models with high complexity and polygons [23].

Some of the performance issues could be overcome with the utilization of remote rendering [24], where an external PC can stream remotely rendered content to HL while receiving input stream from HL.

5.2. Future Work

The future work involves implementing different features that facilitate the deployment in a factory environment. Spatial anchoring was tested for this aim, see Figure 9. Spatial anchors enable a user to find nearby objects to anchor them along with the virtual counterparts and share these anchors with multiple devices and users. This is useful to anchor the virtual robot to the location of the physical robot in the factory and does not require the placement of additional markers. This anchor could be shared with different potential users in the factory, such as robot programmers and operators. However, the accuracy performance needs to be further investigated.

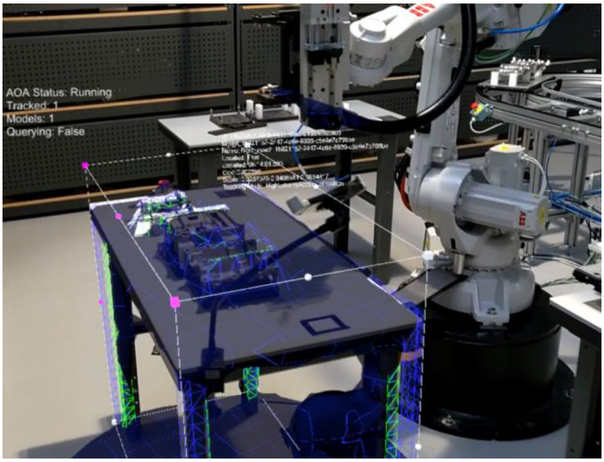

Figure 9. Spatial anchoring in the robot cell.

The main future work remaining is to implement more tools that enable a comparison of real robot current solutions and optimized scenarios, consequently improving the decision-making and creating more optimal robotic paths in real-time. Also, adding functionality for replaying several recordings simultaneously in HL can give better support for comparing different simulations and improving decision-making.

In those future steps user evaluations are planned to assess the concept and support further development of the proposed augmented reality-based interface in a human-centric way.

Acknowledgments

This work was partially financed by the Knowledge Foundation (KKS), Sweden, through the Virtual Factory with Knowledge-Driven Optimization profile (2018-2026), and ACCURATE project (2021-2025).

References

[1] EC, Industry 5.0, Towards a sustainable, human-centric and resilient European industry. EU Publications, 2021 Jan [cited 2021 Oct 25]. Available from: https://doi.org/10.2777/308407
[2] Danielsson O, Holm M, Syberfeldt A. Augmented reality smart glasses in industrial assembly: Current status and future challenges. Journal of Industrial Information Integration, 2020;20:100175.
[3] Honig W, Milanes C, Scaria L, Phan T, Bolas M, Ayanian N.(2015). Mixed reality for robotics. IEEE International Conference on Intelligent Robots and Systems, 2015 Dec:5382–5387.
[4] Lambrecht J, Krüger J. Spatial programming for industrial robots: efficient, effective and user-optimised through natural communication and augmented reality. Advanced Materials Research, 2014;1018:39–46.
[5] Lambrecht J, Kästner L, Guhl J, Krüger J. Towards commissioning, resilience and added value of Augmented Reality in robotics: Overcoming technical obstacles to industrial applicability. Robot. Comput.-Integr. Manuf. 2021;71:102178,.
[6] Makris S, Karagiannis P, Koukas S, Matthaiakis AS. Augmented reality system for operator support in human–robot collaborative assembly. CIRP Annals. 2016;65(1):61-64.
[7] Ostanin M, Klimchik A. Interactive Robot Programing Using Mixed Reality, IFAC-PapersOnLine, 2018;51(22):50-55.
[8] Park KB, Choi SH, Lee JY, Ghasemi Y, Mohammed M, Jeong H. Hands-free human-robot interaction using multimodal gestures and deep learning in wearable mixed reality. IEEE Access, 2021;9:55448-55464.
[9] Karlsson I, Bernedixen J, Ng AHC, Pehrsson L. Combining augmented reality and simulation-based optimization for decision support in manufacturing. In Proceedings of the 2017 Winter Simulation Conference; 2017 Dec 3-6; Las Vegas, NV; 2017, pp. 3988-3999.
[10] Pastras G, Fysikopoulos A, Chryssolouris G. A theoretical investigation on the potentialenergy savings by optimization of the robotic motion profiles, Robot. Comput.-Integr. Manuf. 58 (2019) 55-68.
[11] Hovgard M, Lennartson B, Bengtsson K. Simulation Based Energy Optimization of Robot Stations by Motion Parameter Tuning. in Proc. IEEE CASE, Vancouver, BC, Canada, 2019, pp. 456-461.
[12] Eclipse Foundation, Eclipse Mosquito, [cited 2021 Oct 25]. Available from: https://mosquitto.org
[13] ABB. Robot Studio The world's most used offline programming tool for robotics, [cited 2021 Oct 25]. Available from: https://new.abb.com/products/robotics/robotstudio.
[14] M. Andersson, A bilevel approach to parameter tuning of optimization algorithms using evolutionary computing: understanding optimization algorithms through optimization. Ph.D. dissertation, Univ. Skövde, Sweden, 2018.
[15] Semple K, Kurtis, Aharkey K, Wang MS, Ferrone H, Park Y, RogPodge, Coulter D, Juchen MS, MRTK2.7 What is the Mixed Reality Toolkit, 2021 Jul 1 [cited 2021 Oct 25]. Available from: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/mrtk-getting-started
[16] XML Robot Description Format (URDF), Local Digital Libratry II, 2021 Nov 19 [cited 2021 Oct 25]. Available from: http://library.isr.ist.utl.pt/docs/roswiki/urdf(2f)XML.html
[17] GLB, File Format Developer guide [cited 2021 Oct 25]. Available from: https://docs.fileformat.com/3d/glb/
[18] Ecma International, The JSON data interchange syntax, ECMA-404, 2nd edition, 2017 Dec[cited 2021 Oct 25]. Available from: https://www.ecma-international.org/publications-and-standards/standards/ecma-404/
[19] Patierno P, MQTT Client Library and Broker for .Net platform, [cited 2021 Oct 25]. Available from: https://m2mqtt.wordpress.com
[20] Viganò GP, Jasche F, Pogolski C. M2MQTT for Unity, GitHub, 2018 Jul 24[cited 2021 Oct 25]. Available from: https://github.com/gpvigano/M2MqttUnity
[21] Meijers A. HoloLens and Unity tip #004 – Storing and retrieving data on HoloLens device storage , 2018 Aug 6[cited 2021 Oct 25]. Available from: http://www.appzinside.com/2018/08/06/hololens-and-unity-tip-004-storing-and-retrieving-data-on-hololens-device-storage/
[22] Deb K, Pratap A, Agarwal S, Meyarivan T.A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (202) 182-197. https://doi.org/10.1109/4235.996017
[23] Ferrone H, Tieti V. Understanding performance for mixed reality, 2021 Oct 19 [cited 2021 Oct 25]. Available from: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/understanding-performance-for-mixed-reality
[24] Bagar F, Tieto V. Writing a Holographic Remoting remote app using the HolographicSpace API, 2021 Oct 19 [cited 2021 Oct 25]. Available from: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/native/holographic-remoting-create-remote-wmr