Production Management and Process Control, Vol. 36, 2022, 1–8
https://doi.org/10.54941/ahfe1001613

Augmented Reality-based Programming of Robot Arms

Vladimir Kutscher, David Bassauer, Thomas Dasbach, and Reiner Anderl

Department of Integrated Design, Technical University of Darmstadt Otto-Berndt-Str. 2, D-64287 Darmstadt, Germany

ABSTRACT

Programming of production systems depends on highly specialized personnel and proprietary development environments. This can hinder the evolution towards flexible production. In this paper we present a concept for an intuitive way of human-machine interaction by utilizing the augmented reality technology. This involves a method of programming a machine using an augmented reality headset. All desired machine operations can be predefined via a virtual user interface and by arbitrary placement of virtual working points. We further show how the referencing of the coordinate systems of the augmented reality environment and the robot can be accomplished. We prototypically apply the concept to a robotic arm with 6 degrees of freedom and employ a Microsoft HoloLens 2 optical head-mounted display to interact with robotic arm in the virtual world.

Keywords: Augmented reality, Robot programming, Robot teaching, Industries 4.0

INTRODUCTION

The industrial trend towards increased product variance and smaller batch sizes requires flexible production systems that can adapt quickly to changing tasks. The ability and success of programming the systems depend on highly specialized personnel with the appropriate expertise in using the programming environments and the involved systems themselves. As a result, the reprogramming of production systems is in many cases a major impediment on the way to flexible production systems in the sense of the future vision Industrie 4.0 (Anderl 2016).

BACKGROUND AND RELATED WORK

As an introduction to the topic, we would like to present the state of the art and the relevant work in the addressed research field. Therefore, we will start with the augmented reality (AR) technology, describe the programming of robotic systems and show which scientific work has already been done in this domain.

AR is a subcategory of mixed reality (Milgram et al. 1999). Mixed reality is the area between the real and completely virtual world. The goal of AR is the perception of the real environment and the objects superimposed

on it and the interaction with the virtual content by means of movements in the real world. Different technical tools exist to immerse in AR and interact with virtual objects. One possibility is the use of head-mounted displays, such as the Microsoft HoloLens 2. The HoloLens 2 in particular supports hand tracking for both hands simultaneously, recognizes a variety of gestures, and offers eye tracking. To create applications for the HoloLens 2, the game engine Unity was used (Unity Technologies 2022). Furthermore, the open-source development kit Mixed Reality Toolkit was utilized, which provides basic components for the development of AR applications (Microsoft 2022).

A robot is an automatically controlled, freely programmable manipulator that can be programmed in three or more axes (DIN EN ISO 10218-1). The focus of this work is on stationary industrial robots. These move along fixed programmable paths, while sensors for spatial orientation are usually not used. The movement of the robot in space is described by the distance traveled, the velocity and acceleration and is referred to as robot kinematics. (Paul 1992). Several coordinate systems are used in robotics at the same time. The transformation between the coordinate systems is done by means of the Denavit-Hartenberg transformation (Sciavicco 2000). In the context of this work, an immutable world coordinate system is referenced, which has its origin in the foot element of the robot. A robot is controlled by means of a corresponding interface, such as a handheld controller, which provides all the functions of robot control. The programming systems depend on the manufacturer of the robot system and are not uniform. Therefore, trained personnel are needed to program for the specific system. For this reason, AR robot programming can provide a more intuitive experience, making it easier to get started with robot teaching (Wassermann et al. 2018). Evlampev and Ostanin have developed as a basis a concept of path finding from a starting point to an end point with obstacle avoidance (Evlampev et al. 2019). For this purpose, they use the spatial mapping function of the HoloLens 2 to create a mesh from the environment and thus detect obstacles. In this way, they extended the concept of Ostanin and Klimchik, which addresses interactive programming of robot systems by means of mixed reality (Ostanin et al. 2018). As a further step, Puljiz et al. developed a method for referencing the HoloLens 2 to the coordinate system of the real robot and then enabled control of the robot by gestures and hand movements (Puljiz et al. 2019). Blankenmeyer et al. developed a method for intuitive robot programming using markers detected with the image registration algorithm Vuforia. Objects identified in this way can be picked up and moved (Blankemeyer et al. 2018). The concept presented below is a complement to the related work described above.

AUGMENTED REALITY BASED PROGRAMMING

In the following, a concept for interaction with a robot using an AR headset is explained. The focus here is on teaching robots and their program sequences. The basic idea is to use the AR environment as an interface between the operator and the robot, to plan the actions of the robot arm in the virtual environment by means of so-called working points and to generate a control

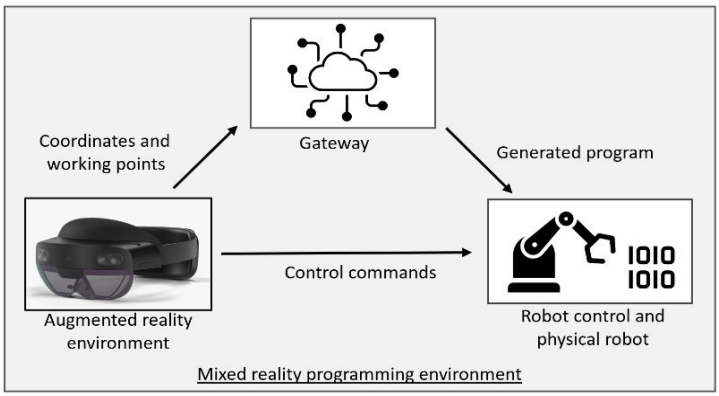

Figure 1: Concept structure of augmented reality-based of robot arms.

program based on them. Figure 1 illustrates the concept architecture. It shows the subsystems used with the respective interfaces used for communication. If the robot system offers a corresponding interface, the AR environment can be directly connected to it in order to directly address the functions of the robot arm, such as an emergency stop or the release function. If the robot system does not offer the possibility of direct communication or if it is not possible to create a control program with the robot's own functions, additional software is required. This software can be in the form of a communication server, which performs the function of a gateway and mediates or translates between the AR and the robot environment. In the presented case, some functions could be addressed directly, while the creation of a control program was done by means of the communication server.

To create the control program, all necessary functions can be displayed in configurable views in the AR environment at appropriate positions to support intuitive operation. Figure 2 shows the main menu for controlling the robot.

Figure 2: Main menu for controlling the robot arm in thaugmented reality.

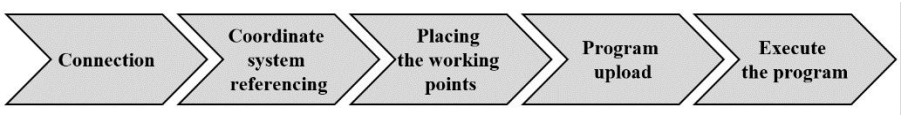

Figure 3: Process overview of generating robot arm programs in the augmented reality.

Starting from the main menu, other menus can be displayed and locked in the AR environment. The creation of the robot program is divided into five main steps, which can be seen in Figure 3.

The process begins with establishing a connection between the systems. The AR headset communicates with the robot system by means of a Wireless Local Area Network (WLAN) using the Transmission Control Protocol (TCP) and the Internet Protocol (IP). This can be used to send direct commands to the robot controller as well as data to the gateway, which is transferred to the robot controller after processing. To enable the robot system to behave in a manner that is accurate in terms of position and orientation, the coordinate systems must be synchronized in the next step. For this purpose, a method was developed according to which one point is identified on each of the three coordinate axes of the robot's base coordinate system to determine the position and orientation. Starting from a home point, the robot moves to the three points one after the other and the operator places a reference point on the respective positions (Figure 4). The three reference points can be used to determine the center of the described triangle. From the three reference points, the center S of the spanned plane (Figure 5) can be calculated. Formula (1) is used for this purpose. Subsequently, the origin *originPoint* can be calculated by means of the formula (2). The distance of the individual reference points from the starting position of the robot arm is considered with the factor a. Finally, the *homePoint* is subtracted to conclude the origin of the robot system and thus to calibrate the two coordinate systems.

Figure 4: Robot operator is placing working points in the augmented reality environment.

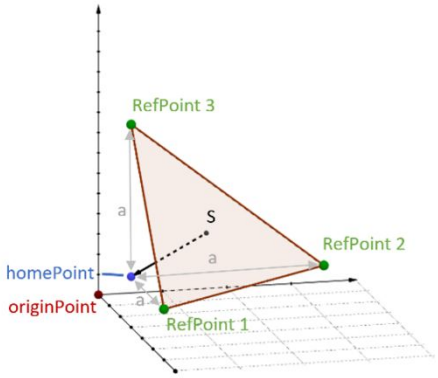

Figure 5: Visualization of the spanned triangle used for the referencing process.

$$S = \frac{1}{3} \begin{pmatrix} Ref1.x + Ref2.x + Ref3.x \\ Ref1.y + Ref2.y + Ref3.y \\ Ref1.z + Ref2.z + Ref3.z \end{pmatrix} \tag{1}$$

$$originPoint = S + \frac{a}{\sqrt{3}} \bullet normalVector - homePoint \tag{2}$$

From the resulting three reference points, the HoloLens 2 calculates the position of the *homePoint* predefined by the physical robot. Finally, the orientation of the coordinate system must be adjusted. For this purpose, a command is used in the HoloLens 2 that allows to rotate the calculated origin. The orientation of the Z-axis is done automatically, while a second axis (X or Y) must be adjusted manually in the orientation to get a unique coordinate system in position and orientation. With referencing using three points instead of one, the accuracy of the calibration is improved. Finally, the determined offset between the *homePoint* and the *originPoint* of the robot system allows the conversion between the coordinate systems. The conversion also considers whether the two coordinate systems differ in terms of left or right orientation. Once referencing is complete, the next step in the process chain follows. To create a program the user can mark with a hand gesture points within the working area. These points are used as support points for the trajectory (Figure 6). At each point, further commands, e.g. regarding the end effector can be placed. This information is represented via a color code (Figure 7). After the machining has been planned, a control program is generated as part of the *Program Upload*. For this purpose, the coordinates of the working points are read out and transferred to the gateway. This transforms the program description into the format of the robot arm and forwards it to the robot arm. In the last step, the control program is started by means of *Run Program*. With the concept presented here, it is possible to interact directly with the robot controller as well as to create control programs using the gateway, which are then used in the automation process.

Figure 6: Robot operator is placing working points in the AR environment.

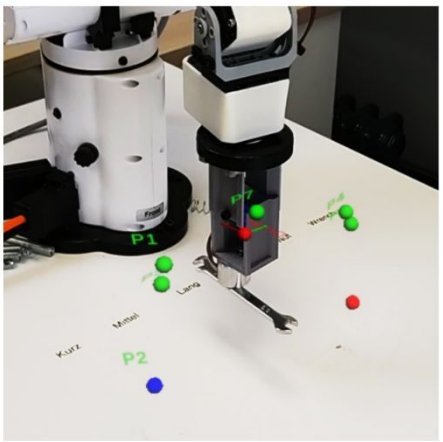

Figure 7: Robot arm picking up a spanner on blue point and moving to red point on the right.

VALIDATION AND CONCLUSION

A prototype implementation was carried out to validate the presented concept. For this purpose, the robot system Mover6 from Commonplace Robotics was utilized (Commonplace Robotics 2022). This robot is equipped with an electromagnet as an end effector. The robot has a dead weight of 3.5kg and can handle loads of up to 400g. The connection to the robot controller is established via a Universal Serial Interface (USB). The robot control is implemented as the Windows application software CPRog. The control programs of the CPRog are based on Extensible Markup Language (XML) files. In addition, the robot controller has a Commonplace Robotics Interface (CRI) Ethernet interface. The prototypical gateway, implemented in the Python programming language, is connected to the CRI interface. Communication between the gateway and the AR environment is in JavaScript Object Notation (JSON) format. The programming of the AR environment

for the Microsoft HoloLens 2 was done in the Unity game engine and using the Microsoft Mixed Reality Toolkit.

The validation of the concept and the implementation showed that programming a robotic system by specifying working points in the AR environment is a promising approach to redefine and enhance the intuitiveness of human interaction with machines by means of the novel AR technology. In the implemented use case, the robot was programmed to move a magnetic object in a handling process. For this purpose, the electromagnet is switched on and off as the end effector of the robot arm in the AR environment. In principle, the implemented concept can also be used to realize other handling processes such as separating, joining, coating, scanning, etc. Compared to established processes such as hand-guided teaching, the concept offers the advantage that it can also be used with existing systems that do not support intuitive programming.

However, the implementation carried out also shows limitations. In particular, the accuracy of the positioned working points is not sufficient for most industrial processes. This depends on several factors of the employed system. The detection of hand movements is not sufficient when measured against industrial accuracies. This depends, among other things, strongly on the lighting conditions of the working area and the viewing angle of the operator. In addition, the position of the working points is recalculated with each frame, so that especially with increased movements of the AR headset, the working points can drift away from their positions and have to be readjusted. In addition, depth perception is more difficult in the AR display and the working points must be viewed from different positions in order to place them as accurately as possible. These factors not only lead to inaccuracies in the positioning of the working points, but also influence the coordinate system referencing and thus affect the overall accuracy of the method. Thus, the presented method has to be further developed in order to be used in industrial practice.

OUTLOOK

The introduced concept is to be assigned to an early prototype phase. As the validation has shown, some aspects still need to be improved. In the following, possibilities for improving the programming in the AR environment are shown. On the one hand, the implementation can be evolved by implementing additional functions. For specific parameters such as movement speed, default values are assumed for this concept, but these can also be made customizable via the AR environment if the concept is further developed. In addition, a point-to-point control system was implemented as a prototype, which can be extended to other control types. Technologically, more accurate sensor technology can continue to be adopted to improve accuracy. For example, a Light Amplification by Stimulated Emission of Radiation Detection and Ranging (LIDAR) sensor can be used to improve the spatial mapping function. In addition, special position styli already exist (Holo-Light 2020), which, in combination with additional sensors in the AR headset, can significantly improve accuracy. Referencing can be further improved with the

help of an image registration algorithm, such as that offered by the Vuforia Engine (Olbort et al. 2020). Thereby, it is possible to load models of a physical robot into the AR environment and to perform coordinate referencing by means of additional geometry information (Puljiz et al. 2019). In order to transfer this concept more easily to other robot application a standardized interface should be introduced. According to the current status, the presented concept has to be adapted to the respective interfaces for each robot of a different manufacturer, which prevents extensive use in the industry. In this context, the Open Platform Communications Unified Architecture (OPC UA) technology needs to be pointed out, which have already defined standardized information models for robotics (OPC Foundation 2022).

ACKNOWLEDGMENT

This work was funded by the Hessian LOEWE initiative within the Software-Factory 4.0 project.

REFERENCES

Anderl, R. (2016). Industrie 4.0 - Digital Transformation in Product Engineering and Production, 21st International Seminar on High Technology.

Blankemeyer, S.; Wiemann, R.; Posniak, L.; Pregizer, C.; Raatz, A. (2018). Intuitive Robot Programming Using Augmented Reality. In: Procedia CIRP.

Commonplace Robotics (2022). Mover 6 Robotarm. Available at https://shop.cpr-robots.com/?product=mover-6-robotarm&lang=en, accessed 04.02.2022.

Evlampev, A.; Ostanin, M. (2019). Obstacle avoidance for robotic manipulator using Mixed reality glasses. In: DCNAIR, Russia.

Holo-Light (2020). Stylus XR - AR Hardware for High Precision Tracking. Available at https://holo-light.com/products/stylus-xr/, accessed 08.02.2022.

Microsoft (2022). MRTK-Unity Developer Documentation - Mixed Reality Toolkit. Available at https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/mrtk-getting-started, accessed 02.02.2022.

Milgram, P.; Colquhoun, H. (1999). A Taxonomy of Real and Virtual World Display Integration. In: Mixed Reality. Springer Berlin Heidelberg.

Olbort, J.; Herden, H.; Kutscher, V.; Anderl, R. (2020). Mixed Reality for Visualization of Operating Data and Semantic Self-Descriptions of Machines using OPC UA. In: Advances in Manufacturing, Production Management and Process Control. Springer International Publishing.

OPC Foundation (2022). OPC Unified Architecture for Robotics. Available at https://opcfoundation.org/developer-tools/specifications-opc-ua-information-models/opc-unified-architecture-for-robotics/, accessed 08.02.2022.

Ostanin, M.; Klimchik, A. (2018). Interactive Robot Programing Using Mixed Reality. In: IFAC-PapersOnLine, 51.

Paul, R. P. (1992). Robot manipulators. Mathematics, programming, and control; the computer control of robot manipulators. Cambridge, Mass.: MIT Pr.

Puljiz, D.; Stöhr, E.; Riesterer, K. S.; Hein, B.; Kröger, T. (2019). General Hand Guidance Framework using Microsoft HoloLens.

DIN EN ISO 10218-1. (2021). Robotics - Safety requirements.

Sciavicco, L. (2000). Modelling and Control of Robot Manipulators. Springer.

Unity Technologies (2022). Unity - Manual: Unity User Manual 2020.3. Available at https://docs.unity3d.com/Manual/index.html, accessed 02.02.2022.

Wassermann, J.; Vick, A.; Krüger, J. (2018). Intuitive robot programming through environment perception, augmented reality simulation and automated program verification. In: Procedia CIRP, 76.