Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 93 (2020) 1218–1223

www.elsevier.com/locate/procedia

53rd CIRP Conference on Manufacturing Systems

An Augmented Reality Framework for Robotic Tool-path Teaching

Sonia Mary Chacko, Armando Granado, Vikram Kapila*

Department of Mechanical and Aerospace Engineering, NYU Tandon School of Engineering, Brooklyn, NY 11201, USA

Abstract

Robot manipulators are widely used in industries to automate myriad operations. To address the demand shifts in industries, automated systems such as robots must flexibly adapt in response. Such dynamics often necessitate end-users to reprogram robots to meet the changing industry needs. This leads to a demand for experienced programmers, familiar with proprietary robot software. As one alternative, an augmented reality (AR) framework can offer a user-friendly solution that permits end-users to reprogram robots without any domain expertise. This paper presents an AR teaching (ART) methodology that allows end-users to program varied manipulators in an intuitive and effortless manner for tool-path teaching. The ART method is contrasted with an alternative kinesthetic teaching method for its performance and user experience. Results show that the ART method yields a convenient and time-efficient teaching method and it is recommended by users over the kinesthetic teaching method.

Keywords: Augmented/mixed reality; human-robot interaction; user interface design; intuitive robot programming

1. Introduction

Robot manipulators are commonly employed in industries to improve productivity by automating processes. Many operations in small and medium-sized industries, as well as small-batch production, require a flexible and rapid programming method for robotics-driven processes to meet the ever changing production requirements. Programming industrial robots to perform high precision tasks is challenging for robot operators. Many task-specific industrial robotics operations such as welding, glue dispensing, painting, material handling, and machining (e.g., cutting, milling, polishing, among others) are time-consuming to teach a robot and require an expert operator for programming the robot. For example, in welding operations, each change of parts to be welded requires the robot to be reprogrammed. The time and effort required to program the robot are key challenges in robotic welding [1]. With the introduction of intuitive and agile programming methods, the time and effort needed to program a robot can be significantly reduced. Moreover, a simple robot teaching approach can make robotics-based industrial tasks accessible to non-expert operators and obviate the need for complex robot programming approaches.

AR approaches have revolutionized the robot programming process by providing intuitive modes for user interaction with robots. An AR interface can provide a real-world view of the robot and spatial information of its workspace, allowing a robot operator to define and visualize the robot pose and trajectories in its work environment. Moreover, various modes of communications such as the use of tangible devices [2]–[4] or gesture-based task definitions [5]–[7] make it possible to devise interactive programming by demonstration approaches that simplify the robot programming task.

In this paper, an intuitive and user-friendly robot teaching method is developed by making use of markerless and marker-based AR technologies. The resulting method incorporates several benefits of existing traditional online and offline teaching methods such as simplicity of user interface (UI), access to the workspace of a real robot, no direct physical contact with the real robot, flexibility to modify the robot and workspace specifications, and 'no-code programming' capabilities. The main contribution of this work is the design and development of a cost-effective smartphone-based AR framework (see Fig.1) to teach tool-path to a robot manipulator by considering factors such as: intuitiveness to learn and operate the system; flexibility to integrate user interface with varied robot manipulators; cost reduction by eliminating the need for any additional specialized hardware; ease of installation in an industry environment with minimum calibration requirement; assurance of user safety by obviating the need for physical contact with the robot; and reduced teaching time compared to traditional methods; among others.

* Corresponding author. Tel.: +1-646-997-3161
E-mail address: vkapila@nyu.edu (Vikram Kapila).

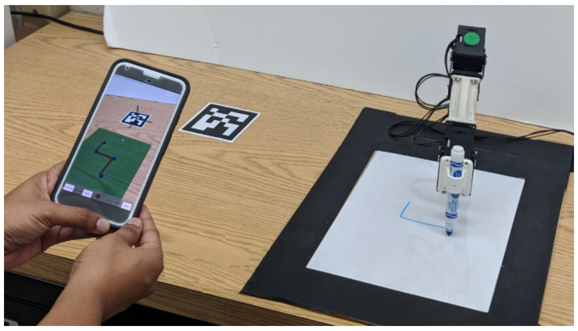


Fig. 1: AR interface for robotic tool-path programming

2. Related Works

In recent years, researchers have proposed alternative and flexible AR-based approaches for programming robot manipulators. One of the pioneering works in this field was reported in [8], which proposed Robot Programming using AR (RPAR) wherein a probe with a single 2D marker is used as an interaction device. Users interact with a scalable virtual robot to specify the tool-path by moving the probe in the virtual robot workspace. In a follow-up paper, [9] proposed RPAR II as an improvement wherein a cube affixed with four markers is used to interact with the virtual robot overlaid on a real robot.

Similar to [8, 9], a hand-held 3D pointer device to define the waypoints and obstacles in the workspace of an industrial robot manipulator was proposed in [10]. In this case, a stereoscopic infrared motion capture system tracks the 3D pointing device and projects it on the workspace and displays it on a monitor. In yet another approach, [11] used a laser pointer to define the waypoints. A laser range finder captures the waypoints and a camera system records the workspace environment. The laser captured points are fused with the camera images and are displayed on a desktop monitor to visualize the tool trajectory as augmented waypoints. Finally, a virtual tool model is also provided to adjust the orientation and liftoff distance with the computer mouse input. The specially designed hand-held devices proposed by [8]–[10] can limit their application to a constrained research environment. Additionally, the dependency on various tracking equipment and settings used by [10, 11] can be impractical in an actual industry shop-floor.

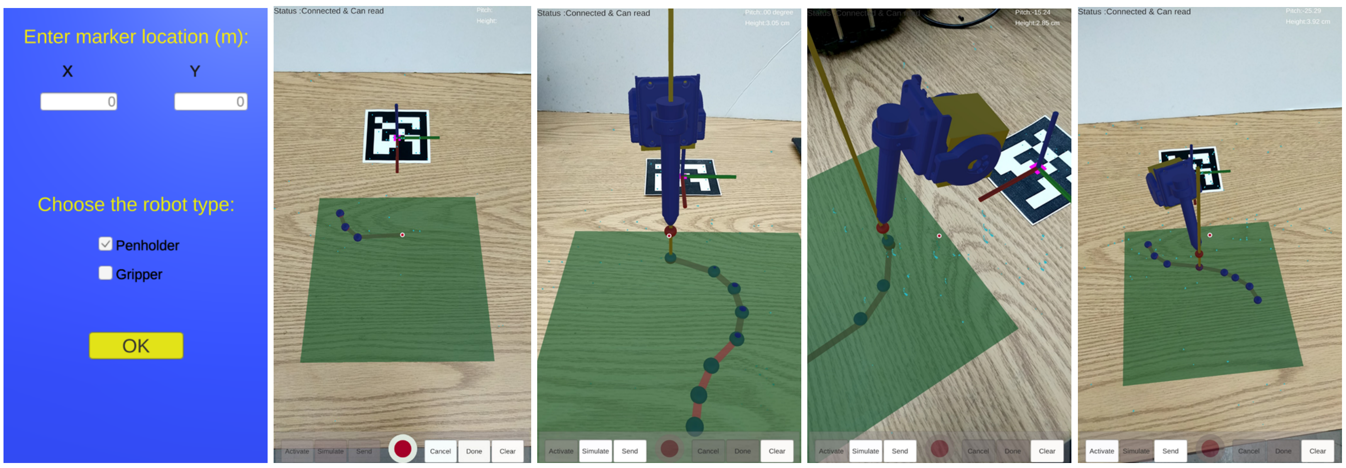
In recent research, mixed-reality head-mounted displays (MR-HMD) (e.g., Microsoft HoloLens) have been considered for AR-based human-robot interactions. In [12], free space or surface trajectories are defined by using an MR-HMD for a seven-DOF robot manipulator. The free space path is set by creating virtual objects at the initial and final anchor points using speech and gesture inputs. The trajectory is auto-generated from the current tool location to the defined two anchor points; the user can modify the path later. In the surface mode, waypoints are specified by creating virtual points at the corresponding locations by gaze and speech inputs. Similarly, in [13], Hololens is used to define the robot motion by adjusting three hologram groups to denote the start, middle, and end points of the end effector (EE). For a complex path, many such groups need to be created to get a required motion path. Likewise, in [14], Hololens is used to visualize a tool-path, which is taught offline, to uncover any issues and to modify it if necessary. The interface also supports advanced users to create new trajectories by inserting manipulable linear or circular segments. Though many recent AR human-robot interaction (HRI) works use MR-HMDs, the high cost of these devices and reported discomforts and cybersickness [15, 16] may affect their industrial acceptance. Another important concern is the potential occlusion of users' vision, which may affect their safety.

3. Approach

The main objective of this work is to design and develop an intuitive, cost-effective, and generic solution for end-users to program robotic manipulators without needing specialized knowledge. The proposed method uses mobile AR technology to design and prototype a user-friendly interface for interacting with the robot manipulator to teach tool-path, liftoff distance, and orientation of EE. The AR application is developed by using Google's ARCore [17] library on the Unity3D platform. A combination of markerless and marker-based AR technologies are applied to develop the AR framework for defining waypoints and tool orientation. The marker-based AR technology detects and tracks a reference 2D image marker. This marker's pose is used for calibrating the robot's pose in the smartphone's coordinate frame. The markerless AR technology enables the detection of features of the surface plane where the marker and robot workspace are located. This enables a user to create virtual objects at the detected feature points on the plane surface and track their pose relative to the smartphone. The user begins by specifying waypoints for the robot tool on the workspace surface plane. Next, a virtual tool model is created at the starting location of the path following which the height and orientation of the tool are specified. Having established the tool configuration, the user can simulate the virtual tool in the AR environment to verify the motion of robot tool prior to executing the plan with the real robot.

3.1. Virtual Tool-path Creation and Interactions

A tool-path is defined by creating virtual waypoints (displayed as small blue spheres) at desired locations on the surface by clicking a Create button (see UI Design subsection). A red-dot target on the center of the screen helps to select a desired location on the surface. A ray is projected onto the camera's view of the physical world from the red-dot target location on the screen to establish the corresponding 3D location in the physical space where an anchor is instantiated. A virtual waypoint is attached to the anchor, allowing the waypoint to remain anchored to the same location in the AR scene over time. The virtual waypoint locations are stored and tracked by the AR application, along with tracking the reference marker. When a virtual waypoint is created, a floating line is generated from the virtual waypoint and follows the red-dot target as the user moves it. Each new waypoint is connected to the previous one by such a connecting line; this helps the user to get a better visualization of the tool-path while defining the waypoints. Especially for complex paths, such as a curved path, these connecting lines are useful for the user to decide the required number of waypoints and the distance between them to define the path precisely.



(a) (b) (c) (d) (e)

Fig. 2: Screenshots of the AR application: (a) Menu page to select robot type and enter marker coordinates; (b) Creating waypoints; (c) Created virtual tool and adjusting the elevation; (d) Adjusting the tool orientation; and (e) Simulation in the AR scene

The virtual waypoint locations relative to the smartphone are transformed into the robot's coordinate system through a set of coordinate transformations. The location of each waypoint in the smartphone's coordinate frame is first transformed into the marker's coordinate frame, and then it is transformed in the robot's coordinate frame based on the location of the marker with respect to the robot. The underlying AR technique for creating and tracking a virtual object, computations to estimate its location in the robot's frame, and transferring data to the robot system are adapted from our previous work [4].

Before executing a program on a robot, simulations are performed in the AR environment by activating the virtual tool to verify the tool path taught. By default, the virtual tool is oriented normal to the surface plane and placed on the surface level. The tool height and orientation are defined by adjusting the elevation and orientation of the virtual tool. The user can intuitively interact with the virtual tool through touch screen interactions to change its elevation by moving it up or down with a single finger drag interaction. Two-finger twirling interactions adjust the orientation (pitch angle). The tool orientation is restricted to meet the joint constraints of the end-effector. The tool height and orientation values are also displayed on the screen. If the tool height exceeds a safe range, the height display turns red to alert the user.

3.2. UI Design

The AR interface is developed by considering different design aspects such as ease of use, operation flexibility, and ease of integration.

3.2.1. Ease of Use

Design of AR application is simplified to aid users to operate it without any technical assistance. The robot type and marker location are entered on the menu page (see Fig. 2a). The names of buttons help to anticipate the corresponding action. The center *Create* button creates a waypoint at the selected location of the red-dot target. The *Cancel* button removes most recent waypoint. It helps to modify the path by deleting previous waypoints one-by-one. In an alternative approach, the user can selectively delete specific waypoints for modifying the tool path. The *Done* button finishes waypoints creation. The *Activate* button creates the virtual tool at the first waypoint location. The *Simulate* button simulates the virtual tool motion through the created path. The *Clear* button removes waypoints and the virtual tool. The *Send* button sends virtual tool and waypoint information to the robot system. The buttons are automatically enabled or disabled sequentially to help the user to navigate the buttons to follow the correct sequence of operations (see Fig. 2).

3.2.2. Flexibility of Operation

The ART application is useful for flexibly teaching the tool-path *in-situ* with or without an actual robot by positioning the 2D reference marker at an arbitrary location. For instance, if we want to teach the tool-path without a robot, we can create an AR environment by affixing the marker anywhere (see Fig. 1). In this case, the marker coordinates are zero and it is assumed that during actual task performance the robot will be located at the center of the marker. That is, by default, the marker center is considered as the base coordinate of the robot, and waypoint locations are estimated relative to it in the robot coordinate system. Otherwise, the tool-path can be taught by affixing the marker somewhere near the robot (and by entering the marker coordinates with respect to the robot on the menu page), while ensuring that the marker orientation is aligned with the robot base coordinate frame. A 2D AR workspace is rendered and virtual waypoint locations are calculated based on the entered position of the marker.

3.2.3. Ease of Integration

The AR interface can be designed for any robot manipulator system. The CAD model of the robot manipulator tool is exported to the Unity software environment to get the virtual tool model. The robot specifications, such as the workspace reachability and maximum allowable EE orientations, decide the virtual 2D workspace visualization area and virtual tool interaction limits, respectively. The AR application can be installed in any ARCore compatible Android or iOS smartphone; this eliminates the requirement for additional UI hardware and corresponding cost burden. On the robot system, wireless communication is established with the smartphone for receiving the data from the AR application.

For the ART interface, a robot teaching setup is prepared by affixing a single marker in the robot workspace or anywhere else, as mentioned above. To ensure good performance of markerless technology in detecting feature points and planar surfaces, it is necessary to have a textured surface and sufficient ambient light in the workspace area. Hence, the marker ought to be affixed to a textured surface to create the AR workspace environment for tool path teaching.

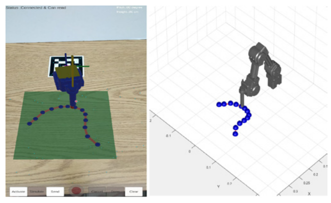
3.3. Robot Integration

The robot manipulator used for this work is an OpenManipulator-X, an open-source robotic manipulator. We tested the manipulator with two different end-effector types – a penholder and a gripper. The penholder is used for attaching a marker-pen for drawing patterns generated with the ART interface. Similarly, for the gripper manipulator, the AR application allows to define path between the start and end points. The determination of end-effector orientation for picking and placing of objects is not in the scope of this work and can be incorporated by implementing our approach from [18].

The ART interface can be adapted for various robotic software platforms. In our prototype, the motors are programmed using Dynamixel SDK libraries on the MATLAB platform. Moreover, an inverse kinematics solver and simulation environment are created using the MATLAB's Robotics System Toolbox. We established a Bluetooth communication between the desktop running MATLAB and the smartphone with the AR application to send waypoint data seamlessly. The inverse kinematics solver generates joint angles corresponding to the tool trajectory based on the received waypoints. The waypoints are simulated and analyzed in Matlab (See Fig 3) before being executed on the real robot.

4. Illustrative Use Cases

To experiment with the usefulness of the ART method for programming a welding robot, a mock-up welding scene is created and demonstrated with the penholder robot, as shown in Fig. 4a. Consider a part is to be welded to a base-plate by the robot moving along the path of the weld-joint by pointing the marker-pen (assume marker-pen represents a welding tool) towards the weld piece. The waypoints are created in close proximity to the weld-joint, followed by the adjustment of the welding tool inclination and height with the virtual tool. The pen-


(a) OpenManipulator with penholder

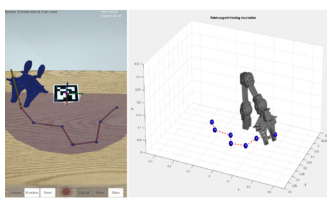
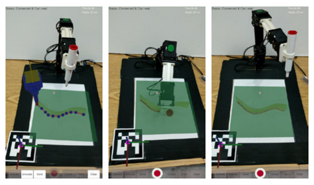

(b) OpenManipulator with gripper

Fig. 3: AR simulation (left) and Maltab simulation (right)


(a)

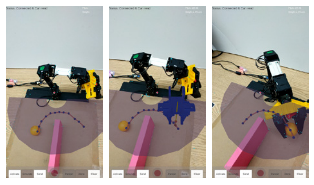

(b)

Fig. 4: (a) Mock-up welding: Weld path is defined and simulated with a virtual tool (1st fig.); Robot moves along the weld path (2nd fig.); Path is drawn (shown as red line) while traversing through the virtual path (3rd fig.). (b) Mock-up material handling: Pick-and-place path is defined (1st fig.); Simulated with virtual tool (2nd fig.); Robot traverses along the path taught (3rd fig.)

holder traverses through each waypoint and draws a path close to the weld-joint. Similarly, the gripper robot is used to perform material handling in its workspace that is partitioned as exterior and interior to an obstacle wall, as shown in Fig. 4b. The robot tool-path is taught by defining virtual waypoints around the obstacle wall. The AR simulation and robot motion are performed along the path taught.

5. User Study Experiments

The ART method is tested with a comparable kinesthetic teaching (hand-guided) method to assess attributes such as intuitiveness, user-friendliness, and task teaching time. The hand-guided (HG) method was selected for comparison due to its intuitiveness and ease of teaching characteristics. The performance of the system is evaluated by instructing the robot to draw two patterns with different complexity levels. The level one teaches a simple straight-line trajectory of a pentagon by creating waypoints at each corner of a given pentagon template. The level two teaches a complex trajectory of a half-circle by creating a sufficient number of approximately equally-spaced waypoints. In this case, the users were free to decide the number of waypoints on a given half-circle template (see http://engineering.nyu.edu/mechatronics/videos/artoolpath.html for a video illustrating the ART method). The experiments are conducted with both interfaces to examine the following hypotheses.

H1:The ART method takes less time to teach tool-path to a manipulator compared to the HG method.

H2: The ART method requires less effort compared to the HG method to teach complex tool-paths.

H3: The ART method is more intuitive and can be learned more easily compared to the HG method.

A total of 22 users participated in the study (females: 5, males: 17, age group: below 18: 27% , 19-24: 41% , 25-34: 32%), all of whom are graduate, undergraduate, or high school students with varying degrees of prior knowledge and/or experience with robotic systems. All participants were able to complete all tasks without failure using both modes of teaching. After each mode of teaching, users reported their perceived workload using NASA raw task load index (NASA-RTLX) survey [19]. To avoid any influence of the order of the experiments on the results, half of the users finished the HG method first, while the remaining users finished the ART method first. Upon completing experiments with both modes of teaching, the users responded to a post-study system usability questionnaire (PSSUQ) (see https://tinyurl.com/surveyPSSUQ), and compared their experiences with the two modes of teaching. The questionnaire is designed to perform a subjective evaluation of the performance, intuitiveness, and user experience of the ART and HG modes of teaching.

6. Results and Discussion

6.1. System Performance

Task teaching time and accuracy test are considered to evaluate the performance of the ART method. Task teaching time taken by users for the two modes of robot teaching is recorded and compared (see Table 1). The task teaching time is the time taken by a user to define the waypoints from the start to end of the path. It includes only the total time taken to teach the robot, not the simulation or execution time. For the straight-line path (pentagon), the users took an average of 38.86 secs and 21.91 secs with the HG and ART methods, respectively. Compared to the straight-line-path experiments, the difference in task teaching time was more significant for the curved path (half-circle) with an average of 55.68 secs and 24.09 secs for the HG and ART methods, respectively. Two paired *t*-tests result shows statically significant differences in teaching time for users to teach with the two methods for the pentagon path (($t(21) = 9.27, p < 0.01$)) as well as the curved path (($t(21) = 6.84, p < 0.01$)). The HG method is more time consuming since the user needs to physically move and position the robot end-effector to each waypoint and save the resulting pose using a desktop GUI. Alternatively, in the ART method, when the user creates a virtual waypoint at a desired location in the workspace, it automatically gets stored in the AR application. Additionally, the path is relatively easy to complete with virtual waypoints from a smartphone screen compared to physically holding and moving the robot tool through the same path. These findings support the first hypothesis (H1).

Table 1: Task teaching time of ART and hand-guided modes

	Pentagon path				Curved path			
Method	Average time (s)	SD	Min (s)	Max (s)	Average time (s)	SD	Min (s)	Max (s)
HG	38.86	8.97	20	55	55.68	22.3	30	132
ART	21.91	4.31	14	31	24.09	5.75	12	37

The accuracy of the ART method to generate virtual waypoints is evaluated by creating a planar path consisting of four waypoints ($\hat{P}_i, i = 1, \ldots, 4$). The waypoints are marked on the workspace of the robot as the ground truth (desired waypoints) and used as guide by the user to create the virtual waypoints. The tool-path teaching experiment is conducted 50 times by a single user to maintain consistency in operator error. The generated virtual waypoints are measured and simulated in MATLAB. The accuracy test of virtual waypoints was performed in accordance with ISO 9283 standard [20]. The accuracy of virtual waypoint created at a desired position is calculated by $\text{Acc}_i = \sqrt{(\mu_{Xi} - \hat{X}_i)^2 + (\mu_{Yi} - \hat{Y}_i)^2}$ where $\hat{P}_i = \left(\hat{X}_i, \hat{Y}_i\right)$ is the desired waypoint, $P_i = (X_i, Y_i)$ is the actual user-created waypoint, and (μ_{Xi}, μ_{Yi}) represents the mean values corresponding to (X_i, Y_i). The accuracy test shows that the ART method yields an average position accuracy of ≈ 0.004 m (see Table 2). The position error is mainly contributed by operator error in selecting desired points on the workspace. The standard deviation (SD) is computed by taking the Euclidean distance from the centroid of the measured waypoint. Since the HG method generates joint poses, it can't be compared with the ART method for its accuracy in generating virtual waypoints.

Table 2: Accuracy test results

	P_1	P_2	P_3	P_4
Ideal (m)	$(0.154, 0)$	$(0.21, 0.04)$	$(0.235, 0.002)$	$(0.183, -0.038)$
Mean (m)	$(0.155, 0.0)$	$(0.209, 0.0395)$	$(0.236, 0.0016)$	$(0.184, -0.0382)$
SD	0.0032	0.0038	0.0039	0.0038
Accuracy (m)	0.00365	0.00387	0.00411	0.00406

6.2. System Usability

The system usability is evaluated from NASA-RTLX and PSSUQ results. The NASA-RTLX reports the average perceived workload of the two modes of teaching (see Fig. 5). Users reported a slightly higher mental workload for the ART since they need to look closely through the screen to precisely create the waypoints. However, with the ART method, users experienced less physical load compared to the HG method. For the HG method, a user needs to hold the tool and move it physically, along with saving the pose on the desktop to record it. Since users need to switch the focus between the workspace and desktop, it results in higher frustration and effort in the HG method *vs.* the ART method. The average overall workload index reported by the ART and HG methods of teaching are 26.17 (SD= 9.02) and 32.59 (SD= 16.83), respectively. The results of a paired *t*-test show statistically significant difference in the average overall workload of the two methods, (with $t(21) = 2.423, p < 0.05$), indicating that the users perceive relatively lower workload with the ART method than the HG method. These results support the second hypothesis (H2).

The usability survey compared the preferences of users with questions about intuitiveness, ease of use, confidence, and level of comfort for the two modes of robot teaching. Most of the users reported that the ART method was intuitive and they learned to use the interface without much assistance. Supported by the survey results of Q1– user-friendliness of ART compared to HG (82% agreed or strongly agreed) and Q2 – more time to learn ART compared to HG (68% disagreed or strongly disagreed), the third hypothesis (H3) is proved. Overall, a majority of the users (86% agreed or strongly agreed) recommended the ART method over the HG method for robot tool-path teaching (see Q10 in Fig. 6).

7. Conclusion and Future Work

In this paper, we presented a mobile AR framework for intuitive tool-path teaching for robot manipulators. The main contribution of this work is the design and implementation of a low cost, generic, user-friendly robot teaching method using mobile AR technology. The choice of a smartphone as the AR device reduces the need for specialized hardware or expensive HMDs. ART enables the robot programming with or without an actual robot system by affixing a single marker within the robot environment or at an arbitrary location. The ART can be easily integrated into any advanced robotic platform by setting a commu-

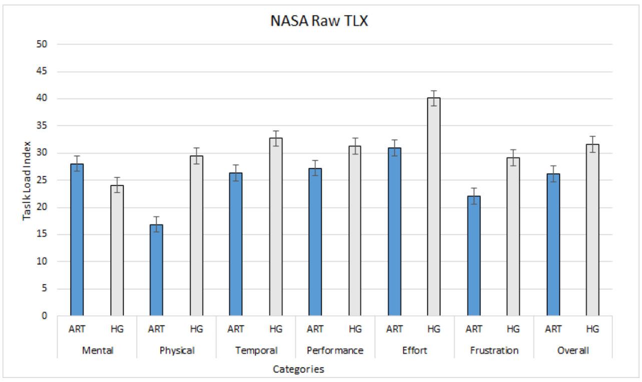


Fig. 5: Workloads perceived in HG and AR

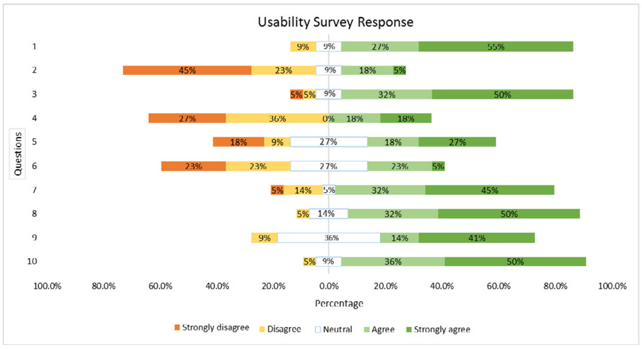


Fig. 6: Usability survey response

nication channel to receive the data sent by the AR application. The simplified UI design enables users to define any tool-path through smartphone touch screen interactions; this avoids physical handling of the real robot. Moreover, it enables users to create any complex tool-path flexibly without having to program the robot or acquiring any specialized robot operating skills.

A comparative study with an existing HG method shows that the ART is beneficial, considering its ease of use, teaching time, user preference, and safety aspects. An average virtual waypoint teaching accuracy of ≈ 0.004 m was obtained with the ART method, which may be further improved by reducing the operator error. At present, the ART method supports creating waypoints on a surface level since the ARCore supports the creation of virtual objects on a detected plane. Moreover, ART supports adjusting the tool-height and orientation once at the beginning of the tool-path, then it keeps the same throughout the path. Though this limits the motion of the manipulator on a level plane, it is beneficial for teaching tool-path for applications such as welding, laser cutting, painting, sorting tasks, etc. Future research will investigate the possibility of defining virtual points in a 3D space to teach spatial path planning.

Acknowledgements

Work supported in part by the National Science Foundation under DRK-12 grant DRL-1417769, ITEST grant DRL-1614085, and RET Site grant EEC-1542286, and NY Space Grant Consortium grant 76156-10488. The authors thank Drs. S. Gosh and P. Sabouri for their helpful suggestions, and the mechatronics, controls, and robotics lab personnel for their support in conducting experiments.

References

[1] T. Majeed, M.A. Wahid, and F. Ali, “Applications of Robotics in Welding,” *Int. J. of Emerging Research in Management and Technology*, vol.7, pp. 30–36, 2018.
[2] S.M Abbas, S. Hassan, and J. Yun. “Augmented Reality based Teaching Pendant for Industrial Robot,” *Int. Conf. on Control, Automation and Systems*, pp. 2210-2213, 2012.
[3] J.A. Frank, M. Moorhead, and V. Kapila, “Mobile Mixed-Reality Interfaces that Enhance Human-Robot Interaction in Shared Spaces,” *Frontiers in AI and Robotics*, 4, Art. 20, 2017.
[4] S.M. Chacko and V. Kapila, “An Augmented Reality Interface for Human-Robot Interaction in Unconstrained Environments,” *IEEE Int. Conf. on Intelligent Robots and Systems* (*IROS*), pp. 3222–3228, 2019.
[5] J. Lambrecht and J. Krüger, “Spatial Programming for Industrial Robots Based on Gestures and Augmented Reality,” *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (*IROS*), pp. 466-472, 2012.
[6] S. Blankemeyer et al., “Intuitive Robot Programming Using Augmented Reality,” *Proc. CIRP Conf. on Assembly Technologies and Systems* (*CATS*), pp. 1–6, 2018.
[7] D. Krupke et al., “Comparison of Multimodal Heading and Pointing Gestures for Co-Located Mixed Reality Human-Robot Interaction,” *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (*IROS*), pp. 1-9, 2018.
[8] J. Chong et al., “Robot Programming Using Augmented Reality: An Interactive Method for Planning Collision-free Paths,” *Robotics and Computer-Integrated Manufacturing* , 25(3), pp. 689–701, 2009.
[9] H.C. Fang, S.K. Ong, and A.Y.C Nee, “Robot Programming Using Augmented Reality,” *Int. Conf. on CyberWorlds*, pp. 13–20, 2009.
[10] A. Gaschler et al., “Intuitive Robot Tasks with Augmented Reality and Virtual Obstacles,” *IEEE Int. Conf. on Robotics and Automation* (*ICRA*), pp. 6026–6031, 2014.
[11] C. L. Ng et al., “Intuitive Robot Tool Path Teaching Using Laser and Camera in Augmented Reality Environment,” *IEEE Int. Conf. on Control, Automation, Robotics and Vision*, (*ICARCV*), pp. 114–119, 2010.
[12] C. P. Quintero et al., “Robot Programming through Augmented Trajectories in Augmented Reality,” *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (*IROS*), pp. 1838–1844, 2018.
[13] S. Y. Gadre et al., “End-User Robot Programming Using Mixed Reality,” *IEEE Int. Conf. on Robotics and Automation* (*ICRA*), 2019.
[14] J. Neves, D. Serrario, and J.N. Pires, “Application of Mixed Reality in Robot Manipulator Programming,” *Industrial Robot*, 45, pp. 784–793, 2018.
[15] A. Vovk et al., “Simulator Sickness in Augmented Reality Training Using the Microsoft HoloLens,” *Proc. CHI Conf. on Human Factors in Computing Systems*, pp. 1–9, 2018.
[16] B.M Silva and P. Fernando, “Early Prediction of Cybersickness in Virtual, Augmented and Mixed Reality Applications: A Review,” *IEEE International Conf. for Convergence in Technology* (*I2CT*), 2019.
[17] Google, ARCore, https://developers.google.com/ar/reference/.
[18] S.M. Chacko and V. Kapila, “Augmented Reality as a Medium for Human-Robot Collaborative Tasks,” *Proc. IEEE Int. Symp. Robot and Human Interactive Communication* (*RO-MAN*), 2019.
[19] S.G. Hart, “NASA-Task Load Index (NASA-TLX); 20 Years Later,” *Pro. of the Human Factors and Ergonomics Society Annual Meeting*, 50 (9), pp. 904–908, 2006.
[20] ISO 9283, *Manipulating Industrial Robots. Performance Criteria and Related Test Methods*, International Organization for Standardization, 1998.