2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Robot Programming Through Augmented Trajectories in Augmented Reality

Camilo Perez Quintero[1], Sarah Li[2], Matthew KXJ Pan[1], Wesley P. Chan[1], H.F. Machiel Van der Loos[1] and Elizabeth Croft[3]

***Abstract*— This paper presents a future-focused approach for robot programming based on augmented trajectories. Using a mixed reality head-mounted display (Microsoft Hololens) and a 7-DOF robot arm, we designed an augmented reality (AR) robotic interface with four interactive functions to ease the robot programming task: 1) Trajectory specification. 2) Virtual previews of robot motion. 3) Visualization of robot parameters. 4) Online reprogramming during simulation and execution. We validate our AR-robot teaching interface by comparing it with a kinesthetic teaching interface in two different scenarios as part of a pilot study: creation of contact surface path and free space path. Furthermore, we present an industrial case study that illustrates our AR manufacturing paradigm by interacting with a 7-DOF robot arm to reduce wrinkles during the pleating step of the carbon-fiber-reinforcement-polymer vacuum bagging process in a simulated scenario.**

I. Introduction

Manufacturing is evolving. Online and on-demand purchasing in a global marketplace, at both the consumer and business-to-business level, has forced modern manufacturers to address untraditional concerns such as high product customization, just-in-time production, minimal warehousing, shorter product life cycles, and smaller batch sizes [1]. Furthermore, the introduction of sophisticated manufacturing processes and new materials that require human expertise at different stages of the production have increased the presence of higher skilled workers to production lines. The next generation of robots working in factories is being designed to work and interact in a complementary fashion alongside skilled human workers, completing collaborative tasks that increase overall productivity.

This new paradigm in industrial manufacturing demands more flexible and intuitive ways of programming industrial robots that allows the user to focus on the task and not on the complexity of the robotic system [2], [3], [4]. Augmented Reality (AR) is a promising alternative for the industry; it facilitates interaction by enhancing the physical world with virtual, computer-generated information [5]. Assembly [6], training [7], maintenance [8] and repair [9] have been some of the industrial tasks where researchers have applied AR to accelerate productivity. Merging AR with robotic systems brings new human-robot interaction (HRI) possibilities. An essential component for effective HRI is the quality of the shared space between human and robot [10]. In traditional robot programming, the shared space is limited to setting way-points with a teach pendant and then replaying the trajectory through them. AR offers a new communication channel to enhance the shared space. For instance, Fang *et al.* [11] proposed an AR interface that uses a marker-cube attached to a probe, which allows a user to guide a virtual robot by setting way-points and orientations. The AR scene is visualized through a desktop monitor. The authors pointed out that participants tended to focus on the screen more than on the real scenario. As a result, due to the difficulty of perceiving depth from the screen, it led to improper guidance of the robot.

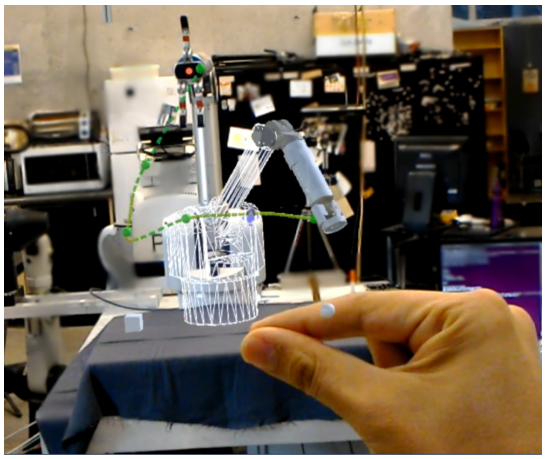

Fig. 1. A user's point of view from within the Hololens: The user creates, modifies, simulates, and executes a pick-and-place trajectory using our AR robotic system. Notice that the holographic 7-DOF robot arm overlays the real robot and permits the user to simulate motions before actual execution. All interaction is performed through speech and gestures.

As an improvement over Fang *et al.*'s system, we aim to explore capabilities such as in-situ task specification, simulation, visualization, and online interaction through the use of AR. In this paper, we propose an AR manufacturing paradigm in which a user can: 1) Specify on the fly a robot trajectory through free space or in contact with a surface, 2) visualize a preview of the robot movement, and 3) monitor and modify robot variables during the simulation or execution mode. We report a pilot study comparing our system with kinesthetic teaching for two task types. Furthermore, we present a case study where our AR system can be applied to a carbon-fiber-reinforcement-polymer manufacturing process, which was formerly a labour-intensive, manual task. Based on human participant experiments, we provide guidelines

[1]Collaborative Advanced Robotics and Intelligent Systems (CARIS) Laboratory, University of British Columbia
[2]Department of Aeronautics and Astronautics, University of Washington
[3]Monash University

978-1-5386-8094-0/18/$31.00 ©2018 IEEE

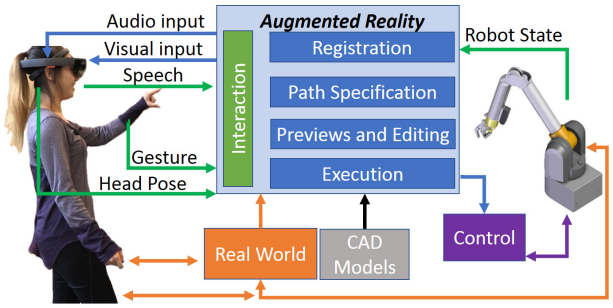

Fig. 2. System Block Diagram

and insights for creating human-robot interactive workspaces through augmenting reality.

II. System

Trajectory programming is a core task in industrial robot automation. However, traditional methods are tedious, non-intuitive, and visually non-collocated with the users' intended task. Our work attempts to overcome these limitations through generalizing trajectory specification by providing two in-situ programming modes: free space trajectories and surface trajectories. Figure 3 and 4 illustrate the two modes and show some functionalities of our proposed AR robotic system [1]. In the former mode, a pick and place task is completed. The user wears a Microsoft Hololens and, through speech and gesture input, selects pick and place locations in the physical workspace. Based on these locations, the user is shown in-situ previews of auto-generated trajectories which the robot can take to move from the 'pick' location, to the 'place' location. Furthermore, the user can modify the path dynamically in free space during simulation and execution. In the second example, as shown in Figure 4, a user is asked to erase a path drawn on a white-board. Using head tracking, speech recognition, and gesture detection, the user sets a virtual path on the surface. Then, using a MYO armband [12] acting as a 1-DOF control input, the user moves the robot's end-effector along the path. The end-effector orientation is automatically kept normal to the surface, applying a constant normal force. In the following section, we provide details on how these functionalities were implemented.

A. System pipeline

Our AR robotic system is composed of a back-drivable torque controlled 7-DOF Barrett Whole-Arm Manipulator (WAM) and a Microsoft Hololens - the first untethered mixed reality head-mounted display that allows holographic visualization [13]. Our general robot programming pipeline, shown in Figure 2, consists of six modules: interaction, registration, path specification, preview and editing, execution, and control.

The ***interaction module*** builds on top of the Hololens SDK gesture and speech recognition libraries and provides the user with abilities such as editing of virtual paths through pinch gestures, multiple speech commands to change state during the interaction, and a virtual cursor controlled by the user's head pose that moves over holograms.

The ***registration module*** registers the virtual world to the real world. This can be done through the use of AR markers - i.e., by manually matching a virtual and real reference feature as shown in Figure 3D - or by using the Hololen's spatial scanning capabilities and then placing holograms on the scanned surface at position of choice, e.g., as shown in Figure 3B where the robot hologram is overlaid on top of the real robot, or in Figure 4D where the hologram of the robot is placed on a table next to the real robot.

The ***path specification module*** implements two types of paths: surface path and free space path. In the former, path way-points are set by the user via a virtual cursor that is visualized on the intersection between the surface of the 3D mesh or CAD model and a ray coming from the user's head orientation. After the user localizes the cursor at a point location, a speech command anchors the position to the 3D surface (Figure 4A and B). The trajectory is interpolated from user-defined way-points. In the free space path creation mode, the user defines two anchor points (cube and sphere in Figure 3B) and a path is automatically generated between the points and the robot's end-effector (Figure 3C).

The path calculation is based on the control points $x_1, \ldots, x_n$ using B-splines [14] with degree $k = 3$ of the form:

$$B_i^k(x) = v_i^k(x) B_i^{k-1}(x) + \left(1 - v_{i+1}^k(x)\right) B_{i+1}^{k-1}(x) \qquad (1)$$

Where,

$$v_i^k(x) = \frac{x - x_i}{x_{i+k} - x_i} \quad \text{and} \quad B_i^0(x) = \begin{cases} 1 & \text{if } x_i \le x < x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

Our motivation for using B-splines instead of standard polynomial representation is B-spline's greater flexibility in allowing dynamical path specification - i.e., changing any point in a path requires a change of all coefficients of a polynomial-based spline interpolation whereas only a few basis functions are affected by using a B-spline implementation.

In the ***preview and editing module***, a speech-trigger virtual simulation of the robot's movement is presented to the user. The current joint state of the real robot is used as initial position; then, based on the virtual way-points specified by the user, the trajectory planning calculations are performed and displayed to the user via the Hololens. During the simulation, the path and robot planning can be dynamically changed using a pinch gesture (Figure 3C).

In the ***execution module***, robot motion is triggered by a voice command. The execution of robot motion along the path can be performed autonomously (i.e., robot automatically proceeds through each way-point) or semi-autonomously (i.e., the user controls the robot's progression along the path using a 1-DOF interface, e.g., gamepad axis or MYO band) depending on process requirements and user preference.

Based on user proximity and if the task requires contact with the environment, the ***control module*** can operate

[1] Video demonstration -https://youtu.be/amV6P72DwEQ

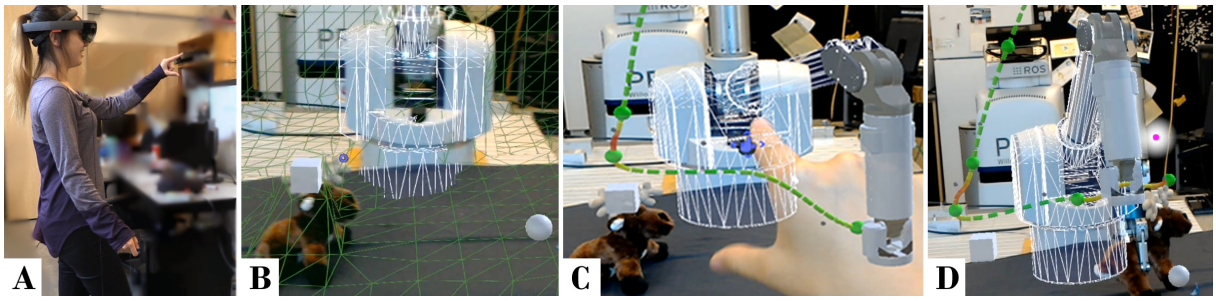

Fig. 3. A pick and place task is completed using our AR Robotics system. (A) An operator wearing the Hololens and gesturing to interact with a 7DOF robot arm. (B) Visualization of, 3D spatial grid, real and virtual robot arm, pick (grey cube) and place (grey sphere) location. (C) Editing visual trajectory by gesturing and then simulate virtual robot through the trajectory. (D) Pick and place execution with virtual and real robot overlapping.

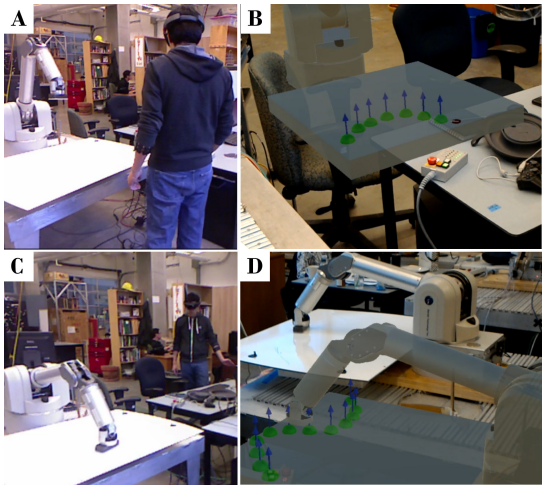

Fig. 4. (A, B) The user set the virtual robot and the CAD model of the dry-erase white board in the side of the real robot and set the path points on the virtual surface. (C, D) The user constrain the end-effector of the robot to the path with an opposite orientation to the surface normals and controls the robot movements inside the path by waving his hand using a MYO device.

in position-controlled or impedance-controlled mode. Our impedance controller is based on previous work [15]. A contact path interaction is shown in Figure 4. The user sets the path way-points by orienting a virtual cursor with his/her head. The user can then anchor the point by way of using a speech command (Figure 4A and B). The corresponding normal is calculated based on the 3D environmental reconstruction or by considering the surface of the CAD model. Then, the user can lock-in the robot to the path, again via speech command. The robot controller utilizes the way-points $\boldsymbol{x}_1,...,\boldsymbol{x}_n$ and their normals (represented as blue arrows in Figure 4B and D) to constrain the robot's end-effector to the path. This is done by minimizing the projection of the distance vector between the robot's end-effector and the closest point on the path onto $\hat{\boldsymbol{s}} = \hat{\boldsymbol{n}} \times \hat{\boldsymbol{t}}$ direction, where $\hat{\boldsymbol{n}}$ is the normal unit vector and $\hat{\boldsymbol{t}}$ is the tangential unit vector calculate from the 3D point path $\hat{\boldsymbol{t}}_i = \boldsymbol{x}_{d_i} - \boldsymbol{x}_{d_{i-1}}$. The force controller is given by:

$$F_s = K_{p_p}\big((\boldsymbol{x} - \boldsymbol{x}_d)\cdot\hat{\boldsymbol{s}}\big) + K_{d_p}(\dot{\boldsymbol{x}}\cdot\hat{\boldsymbol{s}}) \tag{2}$$

where, F_s is the magnitude of the side direction force resulting from the virtual spring (K_{p_p}) and damper (K_{d_p}) system, pulling $\boldsymbol{x}$ towards the closest point on the reference path. The torque command that generates this force is

$$\boldsymbol{\tau}_p = \boldsymbol{J}^T(F_s\,\hat{\boldsymbol{s}}) \tag{3}$$

where $\boldsymbol{J}$ is the task Jacobian matrix of robot.

III. Experiments

We conducted a pilot study of the AR robotic system with ten users. Here, we studied user interactions based on two programming modes: a surface path definition task and a free space path definition task. In each scenario, participants were asked to complete the task using our proposed AR-robotic interface as described in section II, and a kinesthetic teaching interface that consists on a gamepad for storing way-points. In the latter experimental condition, users had to generate each path way-point sequentially by physically positioning and posing the 7-DOF back drivable WAM in gravity compensation mode, before recording the way-point by a gamepad button press (see Figure 5B).

A. Participant Task

Before the start of the experiment session, each participant was provided with basic training on how to operate each interface. In the kinesthetic teaching case, the participant was instructed on how to physically position the robot safely in gravity compensation mode, save way-points using the gamepad, and execute the stored trajectory. In the AR-robotic interface, the participant was instructed on how to specify paths with the AR robotic system and how to move the robot along the specified path.

1) Surface Contact Task: In this task, the end effector of the WAM arm was fitted with a marker eraser. The task consists of erasing two different lines - a straight line and a sinusoidal line - drawn on a dry-erase white board positioned horizontally within the WAM's workspace (see Figures 5A and 6). A grid pattern drawn on the whiteboard using permanent marker assisted with reproducing the lines for each experimental session. During execution, the completion of the erasing task was evaluated by on the percentage of grid boxes that still contained unerased lines following the execution of the robot's trajectory.

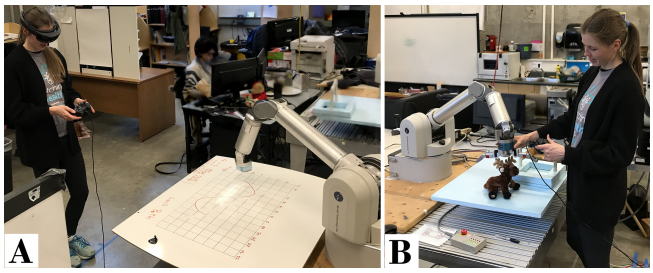

Fig. 5. A) Path specification on a surface setup. B) Path specification on free space setup.

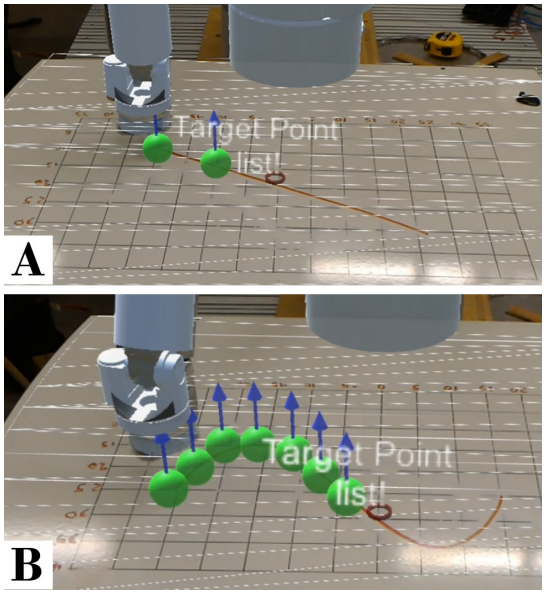

Fig. 6. Participant specifying line (A) and sin (B) shape through the AR-robotic interface.

2) Free Space Task: In this task, a different 7-DOF Barrett WAM arm fitted with a Barrett hand was used. The task consists of programming the robot to pick up an object (a plush animal toy) from one location within the robot's workspace, and place it at another location (in a box) within the workspace. An obstacle is strategically placed to prevent the robot from using a linear trajectory from pick to placement. Thus, the user is required to develop a path that avoids this obstacle and place the object inside the box as shown in Figure 5B. With the kinesthetic teaching method, participants were instructed on how to operate the opening and closing of the Barrett hand using the gamepad, in addition to learning how to store way-points. For the case where the participant uses the AR-robotic interface, they were trained with to edit the auto-generated trajectory path via the pinch-and-release hand gesture.

In both scenarios, measurements including robot pose, joint trajectories, task completion time and task completeness were recorded. Additionally, participants were asked to fill out a NASA Task Load Index survey [16] after each task was completed.

All ten users who participated in this experiment had normal or corrected vision. The experiment took no longer than 60 minutes per participant. The sequence of the tasks and the robot programming approaches were randomized across participants.

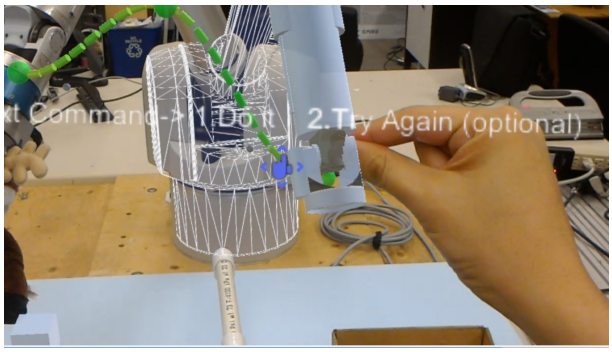

Fig. 7. The participant edits the path to avoid the PVC hurdle and executes a preview to make sure the path is safe.

IV. Results and Analysis

A. Task Time and Completion

We performed paired samples t-tests between teaching times (e.g., time to teach the robot the desired motion path) and task completion percentages obtained from the AR-robotic interface and the kinesthetic teaching. Here, we report the results of these tests as well as the effect sizes in terms of Cohen's d [2].

We found significant differences in mean teaching times between the AR robotic interface and kinesthetic teaching for the pick and place [$t(9)$=2.379, p=.041, d=.752], line erase [$t(9)$=3.786, p=.004, d=1.197], and sine curve erase [$t(9)$=2.839, p=.019, d=.898] tasks. In all of these tasks, the average time taken by users to teach the robot the path was significantly less for the AR robotic system (see Figure 9). These differences in teaching times were observed to be the result of the types of physical motions required to train the robot: in the AR robotic interface, programming the robot's path is done by the user by simply changing the orientation of their head and saying a speech command, thus allowing the user to quickly set multiple way-points in a short amount of time. In the kinesthetic teaching task, users are required to physically move and pose the robot at each way-point which is time-consuming and sometimes awkward given that the WAM has smooth surfaces that are difficult for humans to grasp and manipulate.

In terms of mean task completion percentages, we only found a significant difference in the sine curve erase task [$t(9)$=2.478, p=.035, d=.793], with users achieving a 100% completion rate with the AR robotic system as shown in Figure 8. It appeared that this difference arose from the number of way-points placed by participants using each interface: users tended to place significantly more way-points using the AR robotic system compared to the kinesthetic teaching method simply because it was physically easier and

[2]Cohen suggests that d=0.2 be considered a 'small' effect size, 0.5 a 'medium' effect size, and 0.8 a 'large' effect size [17]

faster to do so. As the robot end effector had to be moved to each way-point during kinesthetic teaching, participants tended to keep the number Of generated way-points to a minimum which also minimized the required physical effort. However, by setting more way-points by using the AR robotic system, the auto-generated path more closely follows the sinusoidal curve, thus resulting in higher task completion percentages.

Although no significant difference was seen for the pick and place task, we found that the task completion was less for the AR-robotic interface, as shown in Figure 8. The majority of reported errors that caused task incompleteness was due to the object hitting the PVC hurdle, which occurs if the user doesn't edit the auto-generated path high enough to prevent the object from hitting the obstacle as it is being delivered to the place location (see Figure 7). One way to overcome this issue is to render a virtual representation of the object during the motion preview, as it's being moved.

B. NASA TLX Results

From participant-reported subjective data obtained from the NASA TLX, we found statistically significant differences in the mental and physical workload between interfaces, as shown in Figures 10 and 11.

For both free-space pick and place [$t(9)$=4.341, p=.002, d=1.373] and surface-based erasing [$t(9)$=2.369, p=.042, d=.749] tasks, kinesthetic teaching appears to requires a lower mental load compared to the AR-Robotic interface. This difference most likely arises due to the slightly increased complexity in setting up way-points and executing the path in the AR-Robotic interface - users must remember a number of specific voice commands which must be executed in sequence (e.g., 'set way-point', 'lock path' etc.), whereas in the kinesthetic teaching method, a participant only requires to only press two buttons to set way-points and execute the trajectory. After observing participants in the pilot study, We believe that the mental load of the AR-robotic interface can be reduced by, e.g., providing a menu in the AR interface with the list of commands.

In terms of physical workload, we find that the kinesthetic teaching interface requires significantly increased physical effort compared to the AR-robotic interface for both free-space pick and place [$t(9)$=3.643, p=.005, d=1.153] and surface-based erasing [$t(9)$=3.344, p=.009, d=1.057] tasks. This result is not surprising as the kinesthetic teaching method require users to physically place the robot at each way-point to program the path the robot is to take, whereas only a simple redirection of gaze and voice command is needed for the AR-robotic interface.

When observing both of these results, it appears that a trade-off between physical and mental workload exists between the two interface types tested - i.e., the AR robotic interface requires slightly more mental effort but much less physical effort to train a robot, and vice versa for the kinesthetic method. Depending on the application, this trade-off may offer a clear advantage of one interface over another. For example, when working with a large or heavy robot,

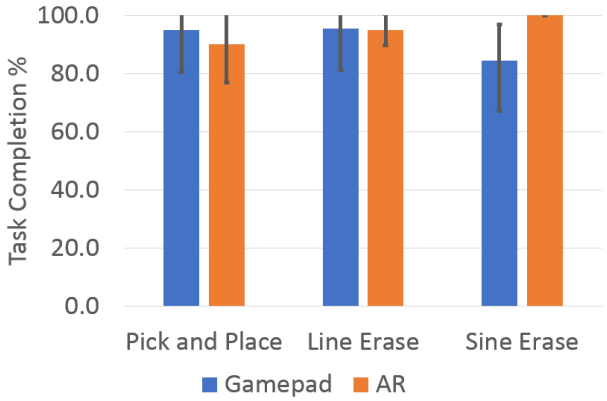

Fig. 8. Pick and place task completion based on avoiding the PVC obstacle and putting the object inside the box. Erase shape time completion based on the number of sections not clean during the erasing task. Error bars represent 95% CIs.

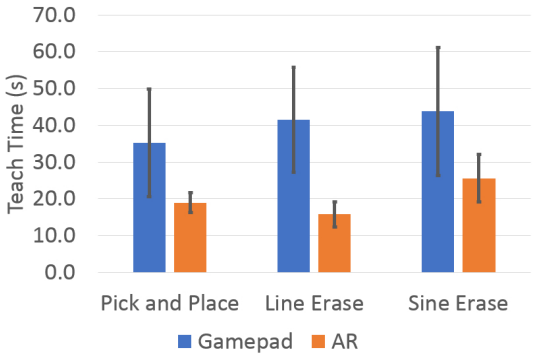

Fig. 9. Time taken to teach the trajectory using kinesthetic teaching and saving way-points using a gamepad controller vs. Time take to teach the trajectory with the AR-robotic interface. Error bars represent 95% CIs.

using the AR-robotic method may offer users a way to quickly prototype trajectories, which would not be possible with kinesthetic teaching.

V. Case Study

A prime example of a complex manufacturing task which may benefit from the AR-robotic interface is the production of carbon-fiber-reinforced-polymer (CFRP) components for the aerospace industry. A commonly used method of producing CFRP parts is through vacuum bagging: the process consists of laying several carbon fiber cloths on a mold (see Figure 12A) before applying a plastic film covering. Subsequently, a vacuum is created between the film and mold to allow resin to diffuse into the sheets of carbon fiber. However, in order for the resin to be properly diffused, the plastic film overlaying the carbon fiber must be pleated to reduce the occurrence of air pockets. Without the film being pleated, manufacturing defects may occur rendering the CFRP component useless. Normally, the pleating process requires experienced technicians to manually manipulate the film, using highly dexterous motions to remove wrinkling and provide proper pleating of the flexible membranes during manufacturing. This task is labor-intensive, requiring an "online expert", and has been

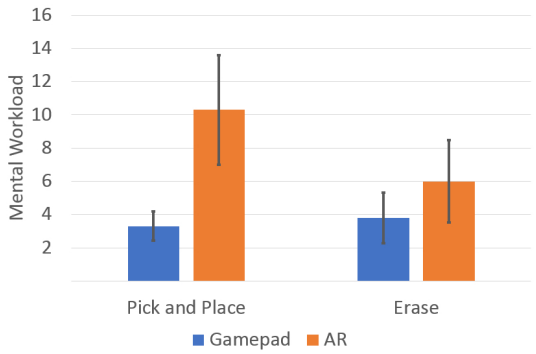

Fig. 10. Mean NASA TLX results on mental workload. Error bars represent 95% CIs.

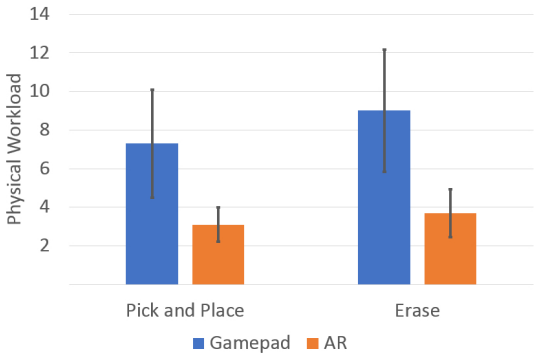

Fig. 11. Mean NASA TLX results on physical workload. Error bars represent 95% CIs.

too complicated to automate completely.

Our goal is to use AR robotic interaction to assist (not replace) the expert during this process. To this end, we have prototyped a section of a pressure bulkhead mold (Figure 12B) including the structural reinforcements. Additionally, we have fitted our 7-DOF WAM arm with a round pad that guarantees appropriate friction with the plastic film material. The AR system was designed to allow an operator to plan surface paths that will move wrinkles in the film to the main pleats. To be able to specify such a path using our AR interface, we first needed to find a virtual surface representation of the working object. Our system currently provides two modalities for the integration of physical objects into the holographic representation. The first approach leverages a spatial mapping approach; however, the plastic film is challenging for the Hololens depth sensor, resulting in a poor 3D environmental reconstruction. A second approach consists of registering a CAD model of the physical object to the physical environment. Figure 13 shows an example of what users see through the Hololens when a CAD model of the sectional mold is inputted and registered into the system. This translucent, wire-frame type of visualization is designed to allow the user to see details of the plastic film wrinkles. Thus, based on the observation of those wrinkles, the operator can specify a path to reduce them directly using the AR system. The normal directions on the selected path

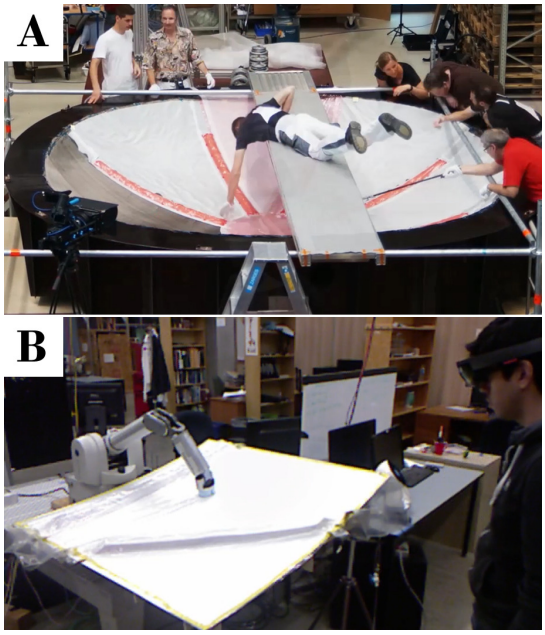

Fig. 12. (A) Expert working on the manufacturing of a CFRP Pressure bulkhead (image courtesy of DLR). (B) Prototype of a section of the bulkhead mold.

are calculated using geometric information from the CAD model and utilized to orient the end-effector during trajectory execution. The robot motion can be set automatically or controlled by the user through a 1-DOF input device. In our implementation, a MYO armband is used.

A. Experiments and Discussion

We proceeded to conduct a pilot study of the system with one participant. At the beginning of the experiment, the participant was instructed on the system's functionalities. Then, he was allowed to interact with the system for 15 minutes on a flat surface. Next, the section mold was placed close to the robot arm and registered with the Hololens (Figure 13B). The user was asked to move all the visible wrinkles to two main pleats located near the structural reinforcements. After completing the task, the plastic film was reset to reintroduce wrinkles for the next test. The participant repeated the test six times. Within all of the six iterations, the participant was able to remove the majority of the wrinkles inside the robot workspace. Two examples of strategies used during the test are shown in Figure 13A and B. In the former; the participant opts for specifying multiple paths perpendicular to the pleats. In the latter, a strategy which uses a single path that tries to move all the wrinkles to the main pleats is shown. The second strategy appeared to reduce task completion time over the first strategy at the expense of degradation of wrinkle-removing performance. After the experiment, the participant was asked to complete a subjective questionnaire.

After running the experiment and receiving the user's qualitative feedback, we learned a few lessons, including:

- The current implementation only allows setting waypoints one by one on the surface. It may be helpful to

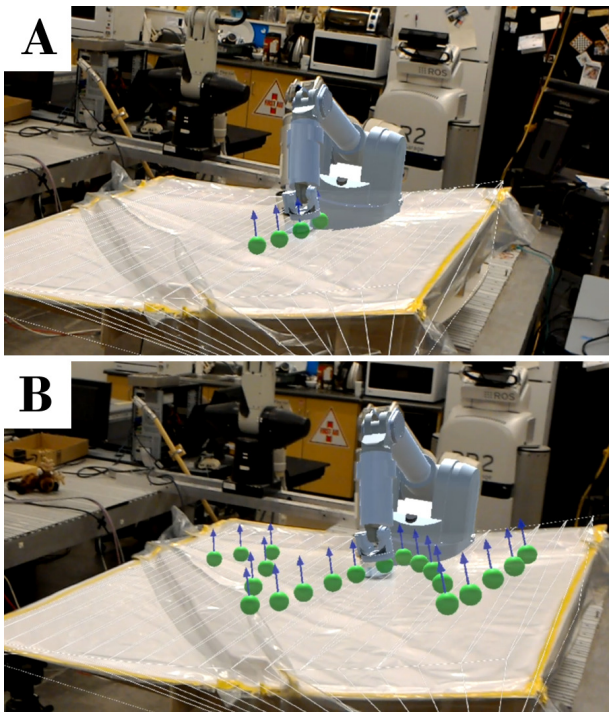

Fig. 13. Two different strategies to reduce wrinkles in the plastic film. The virtual robot and the mold are overlaid on the real scenario. The end-effector is constrained to the path and its orientation is keep normal to the model surface. The user move the robot through a 1-DOF input device.

explore and implement a faster way of setting multiple locations by recording the cursor trajectory.

- The calibration between the holograms and the real world stays accurate if there is not a significant displacement of the user, e.g., by moving behind the robot, the hologram looks offset about 3-5cm. In this experiment, we only used the 3D reconstruction performed by the depth sensor as the anchor method for the holograms. Potential improvements in virtual-to-real space calibration can be achieved by using AR markers and fusing them with the calibration data coming from Hololens' spatial reconstruction.
- The force applied to the surface is constant. Adding a safe way to interactively change the normal force to the surface may improve the quality of the task.
- Although the plastic wrinkle configuration is hard to predict, executing predefined movements before starting the interactive wrinkle reduction may reduce working time.

VI. Conclusions

In this paper, we presented an AR robotic system for trajectory interaction, and through it, we exemplified two robot programming modalities - free space trajectory, and contact surface trajectory. We validated our system functionalities by comparing it to kinesthetic teaching. We found that it takes less teaching time for users using our proposed AR robotic interface than doing the kinesthetic teaching. We also reported that when specifying a more complicated path (e.g., sine curve) our AR robotic interface presents a better performance than kinesthetic teaching, due solely because it is physically easier and faster to do. In the subjective evaluation, we found that the AR robotics interface has a lower workload than the kinesthetic teaching. However, it presents a higher mental workload. This trade-off can be beneficial when working with a large or heavy robot. To test the functionality of our system in a real scenario, we completed a labor-intensive step in CFRP manufacturing. Potential directions of improvement are: introducing force visualization and control during robot programming, more agile path specification methods, and minimizing registration error after prolonged user position displacements.

References

[1] A. Aryania, B. Daniel, T. Thomessen, and G. Sziebig, "New trends in industrial robot controller user interfaces," in *Cognitive Infocommunications (CogInfoCom), 2012 IEEE 3rd International Conference on*. IEEE, 2012, pp. 365–369.

[2] K. R. Guerin, C. Lea, C. Paxton, and G. D. Hager, "A framework for end-user instruction of a robot assistant for manufacturing," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 6167–6174.

[3] C. Heyer, "Human-robot interaction and future industrial robotics applications," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 4749–4754.

[4] T. Brogårdh, "Present and future robot control development- an industrial perspective," *Annual Reviews in Control*, vol. 31, no. 1, pp. 69–79, 2007.

[5] R. T. Azuma, "fa survey of augmented reality," *Presence: Teleoperators and virtual environments*, vol. 6, no. 4, pp. 355–385, 1997.

[6] X. Wang, S. Ong, and A. Nee, "A comprehensive survey of augmented reality assembly research," *Advances in Manufacturing*, vol. 4, no. 1, pp. 1–22, 2016.

[7] S. Webel, U. Bockholt, T. Engelke, N. Gavish, M. Olbrich, and C. Preusche, "An augmented reality training platform for assembly and maintenance skills," *Robotics and Autonomous Systems*, vol. 61, no. 4, pp. 398–403, 2013.

[8] T. Engelke, J. Keil, P. Rojtberg, F. Wientapper, M. Schmitt, and U. Bockholt, "Content first: a concept for industrial augmented reality maintenance applications using mobile devices," in *Proceedings of the 6th ACM Multimedia Systems Conference*. ACM, 2015, pp. 105–111.

[9] S. Henderson and S. Feiner, "Exploring the benefits of augmented reality documentation for maintenance and repair," *IEEE transactions on visualization and computer graphics*, vol. 17, no. 10, pp. 1355–1368, 2011.

[10] C. Breazeal, A. Edsinger, P. Fitzpatrick, and B. Scassellati, "Active vision for sociable robots," *IEEE Transactions on systems, man, and cybernetics-part A: Systems and Humans*, vol. 31, no. 5, pp. 443–453, 2001.

[11] H. Fang, S. Ong, and A. Nee, "Novel ar-based interface for human-robot interaction and visualization," *Advances in Manufacturing*, vol. 2, no. 4, pp. 275–288, 2014.

[12] S. Rawat, S. Vats, and P. Kumar, "Evaluating and exploring the myo armband," in *System Modeling & Advancement in Research Trends (SMART), International Conference*. IEEE, 2016, pp. 115–120.

[13] B. C. Kress and W. J. Cummings, "11-1: Invited paper: Towards the ultimate mixed reality experience: Hololens display architecture choices," in *SID Symposium Digest of Technical Papers*, vol. 48, no. 1. Wiley Online Library, 2017, pp. 127–131.

[14] T. H. Michael, "Scientific computing: an introductory survey," *The McGraw-540 Hill Companies Inc.: New York, NY, USA*, 2002.

[15] C. P. Quintero, M. Dehghan, O. Ramirez, M. H. Ang, and M. Jagersand, "Flexible virtual fixture interface for path specification in telemanipulation," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 5363–5368.

[16] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Advances in psychology*. Elsevier, 1988, vol. 52, pp. 139–183.

[17] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*. Routledge, 1988.