2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

VF Designer: CAD-Guided Virtual Fixtures for Enhanced Robot Teleoperation in Multi-Step Manipulation Tasks

Peter So[†1], Diego Fernández Prado[†2], Yassine Ben Chehida[2] and Eckehard Steinbach[1,2]

***Abstract*— As robots are deployed into new domains, teleoperation with a human-in-the-loop remains an important method for training new skills. Direct control of a remote robot over a network demands high concentration of the human teleoperator and remains challenging due to unavoidable network delays and congestion that can degrade performance. In this work, we demonstrate a shared control solution using well established virtual fixtures (VFs) as a teleoperator assistance tool and introduce a novel VF generation pipeline leveraging mate constraints in available CAD data of the manipulated objects. This approach simplifies the definition of VFs by taking parameters from relationships already defined within the CAD data. Using an industry-sponsored task board and a bilateral leader-follower hand-guided robot scenario, we demonstrate how a set of VFs can be constructed and activated in a series to support teleoperators with a multi-step manipulation task including the pressing of buttons, peg-in-hole with the picking and inserting of a Multimeter Probe Plug, and a novel VF task of opening of a hinged door. We present data from a pilot user study with eight teleoperators and 67 trial attempts with two scenarios (with and without VFs enabled) across three test conditions of round-trip network delays of 0 ms, 100 ms, and 250 ms. We found teleoperators had an increased task success rate, lowered the total travel distance, and overall reduced task execution time, in the best case of 0 ms delay by 26 seconds, or 21%, when VFs were enabled versus when they were not. Performance was maintained or improved at higher network delays.**

I. INTRODUCTION

Teleoperation refers to the manipulation of physical objects from afar which is typically motivated by the delivery of scarce, skilled motion of an expert to remote or hard to access locations, such as in outer space, deep underwater, or inside a body cavity during surgery. Virtual Fixtures (VFs), first introduced by [1], are mathematical constraints which support the teleoperator to successfully complete complex and sensitive tasks by restricting the motion of the remote robot and rendering haptic feedback on the input device.

The bilateral Leader-Follower teleoperation configuration [2] is popular for its intuitive logic, whereby a teleoperator physically manipulates a structurally identical Leader robot as an input device to control a remote Follower robot connected over a network who attempts to replicate the Leader robots movements exactly. Network delays or external forces acting on the remote can cause the two robots to fall out of sync. While VFs are known to increase teleoperation performance, their generation to accomplish complex tasks remains a tedious and iterative process.

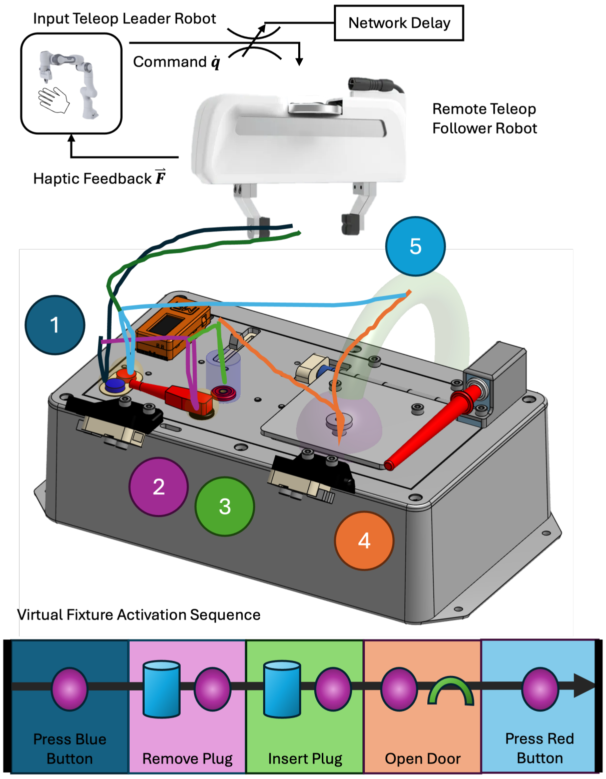

Fig. 1. Sequentially activated, CAD-generated geometric Virtual Fixtures provide haptic feedback to guide a teleoperator to manipulation regions of interest during a multi-step teleoperation procedure on a task board. The colored paths indicate the 5-steps of the Follower robot trajectory with the task-specific virtual fixture primitives shown in the row below.

In this work, we propose a VF generation pipeline of an industrial object with computer aided design "CAD" data to complete a multi-step manipulation task. We examine the teleoperation use cases to press buttons, pick and insert a plug, and open a hinged door using a bilateral leader-follower robot setup under varying levels of network delay conditions. Fig. 1 shows a typical trajectory of the task sequence overlaid on a task board with a series of transparent virtual fixtures. Novice teleoperators use the proposed system to execute the task procedure on a remote robot with and without VFs enabled to support their performance.

The key contributions of this work are:

– A method to generate VFs using geometric features defined in the constraints of the CAD data.
– A framework for activating VFs in a guided sequence in a robot teleoperation setup for a multi-step task.

[†] Authors contributed equally

[1]Munich Institute of Robotics and Machine Intelligence (MIRMI) Technical University of Munich, Georg-Brauchle-Ring 60 80992 Munich, Germany
[2]Chair of Media Technology, Technical University of Munich, Arcisstrasse 21, 80333 Munich, Germany
Corresponding Author: peter.so@tum.de

– A pilot user study to investigate the effect of VFs in teleoperation tasks, including opening a hinged door across degraded network conditions.

II. RELATED WORK

Virtual Fixtures as a teleoperation guide, first introduced by Rosenberg in 1993 [1], are well studied and understood to increase teleoperation performance in both speed and accuracy [3]. However, their design and implementation still require individual effort for each task and scenario. The efficient generation of virtual fixtures grounded on physical objects within a dynamic scene is the topic of current research. A popular implementation uses geometric primitives such as spheres, cylinders, and paths with signed direction fields (SDFs) with vector inequalities [4] to guide the Follower robot's motion along a preferred path or region. Other works use VFs to define forbidden planes to prevent tool ingress into sensitive areas [5] or to maintain a tool orientation while following along a path which is especially helpful for cutting tasks [6] in both cases VFs were defined manually. VFs are also used to restrict the teleoperator to a confined space [7] or regulate the stiffness of the Follower robot along a desired pre-determined path [8]. We build on previous work on virtual fixtures where their parameters are learned in simulation with reinforcement learning [9] and with imitation learning [10]. In contrast to previous works which define VFs from point cloud data or simulation, we propose a method to define a sequence of VFs generated from dimensional data extracted from a STEP AP242 file in a multi-step set of precise manipulation tasks. Furthermore, to our knowledge we contribute the first study of using VFs to assist a teleoperator to open a hinge door. [11] shows how assembly constraints can be extracted from STEP AP242 files among other properties to automatically generate robot programs, like assembly planning [12]. The selected manipulation tasks in our study are a subset from the electronic task board used in the Robothon Grand Challenge [13].

Teleoperation over a network is susceptible to packet loss and traffic congestion, and therefore, the challenge is to smoothly compensate for delayed or missing data packets between the Leader and Follower robots as stated in [14]. To mitigate jumps and instabilities between the leader and follower robots, network conditioning is done with Time-Domain Passivity Approach with energy reflection (TDPA-ER) as described in [15].

III. VIRTUAL FIXTURE GENERATION PIPELINE

A Virtual Fixture (VF) is a geometric volume that supports a teleoperator by providing guiding haptic force feedback to assist in the completion of a predefined task. In this work, we implement VFs inspired by [9] and [10], but unlike in these two papers, the presented VFs are generated from proposals based on mate constraint features present in the CAD assembly file of the manipulated object. We propose using a python script with PythonOCC [16] to extract features (vertices, axes) associated with constraint features between individual components in the STEP AP242 file of a CAD assembly following the ISO 10303-21 standard and provides the user an initial proposal of the VF parameters. PythonOCC is a Python wrapper around Open Cascade Technology[1] project which provides a library of 3D geometric tools. The user could then compose a sequence of VFs by selecting from a list of supported geometric primitives with pre-filled parameters and their locations selected from a set of detected CAD constraint features as detailed in Algorithm 1. Using notation from [17], we consider geometry $\mathcal{G} = (T, \mathcal{F}, P)$ with T as a primitive type (limited to spheres, cylinders, and tori), defined with a position in coordinate system $\mathcal{F}$ and parameters P, e.g. radius and length. Fig. 2 shows a cross-section of a VF located with respect to the robot's base frame.

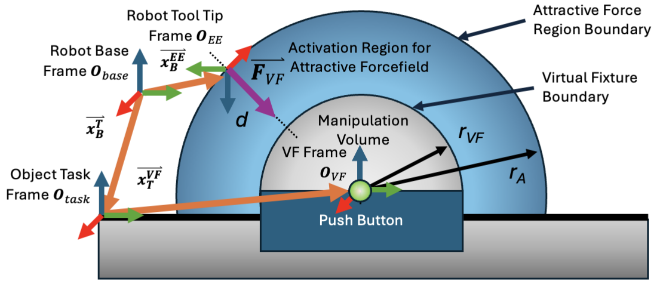

Fig. 2. Cross-section view of a spherical VF defined at the center of a push button located in the Follower robot's workspace. The region where an attractive force pointing towards the center of the sphere is applied to the Leader robot when the distance d between the EE position and the center of the sphere crosses the defined attraction radius r_A and stops when the EE reaches r_{VF}, the inner free manipulation volume of the task.

Algorithm 1 CAD-to-Virtual Fixtures List Pipeline

1: **Import** CAD assembly as STEP AP242 file (offline / pre-process) and convert to OCC Object.
2: **Extract** geometry and semantics: parts, faces, edges, mate constraints (if present), and coordinate systems.
3: **Detect** geometric features for defining VFs: planes, vertices, axes, face centers.
4: **Map** CAD features to virtual-fixture primitives as a list based on sequence of tasks.
5: **Define** VF parameters as SDFs, constraint Jacobians, or vector fields, using suggested values from [10].
6: **Export** VF definitions as a JSON object list.
7: **Deploy** runtime module that:
   1) Transforms the VF locations to the robot frame,
   2) Evaluates the current position of the robot relative to the currently active VF,
   3) Renders haptic feedback on the input device.

For our study, we define VFs using a combination of spheres, cylinders and a torus for three common manipulation tasks on a single task board object with available CAD data. Fig. 3 illustrates a sequence of screenshots of the process to create the VFs shown as transparent surfaces for each of the

[1]Open Cascade Technology Project (OCCT): https://dev.opencascade.org

tasks on the task board. A list of VFs is constructed by creating a step for each task in the procedure and defining VF(s). The spherical, cylindrical and toroidal VFs have an attractive shell and an inner region of interest where the teleoperator can continue to guide the robot without VF forces. The outer radius defines the boundary where the VF guiding forces are applied and the inner radius defines a neutral manipulation region. In our test sequence Fig. 1, spherical VFs were used to approach regions around the buttons and graspable objects, such as the plug and door handle, while cylindrical VFs guided the removal and insertion of the Multimeter Probe Plug into the receptacle, and the open door task was split into two parts: a spherical VF to approach the door handle to grasp, and a torus VF guided the pulling open of the door along the handle's arc about the hinge. Table. I lists all VFs with their parameters used for the task sequence.

The Follower robot EE is evaluated as a point in the robot workspace where attractive VF forces were applied to the Leader robot when it enters a VF region. The haptic force $\vec{F}_H$ rendered on the Leader robot is calculated as:

$$\tilde{\mathbf{F}}_{\mathbf{H}} = \tilde{\mathbf{F}}_{\mathbf{TDPA-ER}} + \tilde{\mathbf{F}}_{\mathbf{VF}} \tag{1}$$

where $\tilde{\mathbf{F}}_{\mathbf{TDPA-ER}}$ is the TDPA-ER-damped reflected force and $\tilde{\mathbf{F}}_{\mathbf{VF}} = F_{VF} \cdot \hat{\mathbf{r}}_{\mathbf{EE}}^{\mathbf{VF}}$ is the guiding force vector from the activated VF acting towards the center of the fixture.

The rendered forces are applied by comparing the current position of the robot EE to the active VF in the sequence illustrated in Fig. 2.

The equation for the force created by the VF as a function of the distance d between the Follower robot EE and center of the VF is:

$$F_{VF}(d) = \begin{cases} 0, & \text{if } d > r_A \\ C \cdot \left(1 - cos\left(2\pi \frac{r_A - d}{r_A}\right)\right), & \text{if } r_{VF} < d \leq r_A \\ 0, & \text{if } d \leq r_{VF} \end{cases} \tag{2}$$

where C is a constant to define the force magnitude, r_{VF} is the the radius to the VF boundary and r_A is the radius of the VF attractive region. d is the distance from the EE to the VF center for the case of a spherical VF, or to the axis in the case of a cylindrical VF, or to the center path in the case of the torus. In our study, $C = 4$ was selected per our user preference for our setup.

IV. EXPERIMENTAL SETUP

The experimental setup involves two 7-DOF Franka Robotics Panda robots [18] in a bilateral leader-follower teleoperation. The "Leader" robot is directly manipulated by the teleoperator and a command velocity twist is sent over a wired Ethernet network to the an identical "Follower" robot to manipulate the task board in a remote location. Video feeds from two Intel Realsense D435 cameras [19] in the Follower robot's workspace show views of the task board rigidly fixed to the table are synchronized and displayed to the teleoperator, see Fig. 6. The Follower robot is equipped with

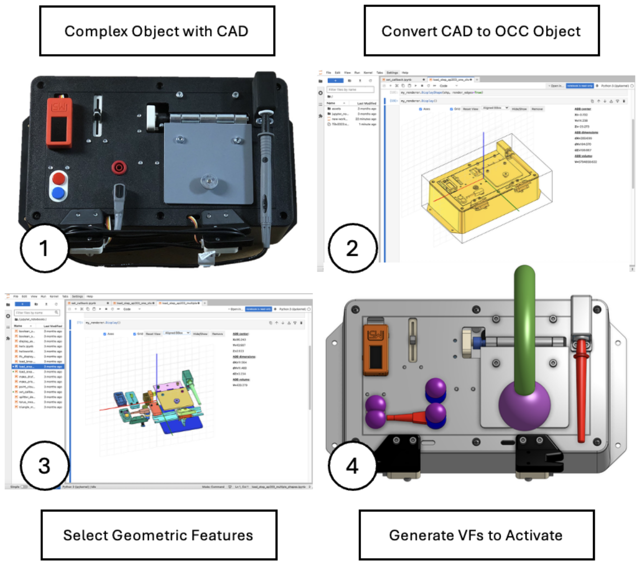

Fig. 3. A series of screenshots illustrates the VF generation pipeline. (1) Shows a pictures of an industrial assembly with CAD data, (2) the portable STEP AP242 file is imported into a Jupyter notebook running PythonOCC to visualize individual parts and their geometry features, (3) the task specific CAD features can be selected and their dimensions saved and used to parameterize the VFs, (4) several VFs can be created and exported as a list to be used in a teleoperation session.

a wrist-mounted Bota Systems Force-Torque Sensor (BFT-SENS-ECAT-M8) "FT-sensor" and a custom-made extension cable to connect the electric 2-finger Franka Robotics gripper to detect the environment forces wrench felt by the Follower robot and render them as haptic feedback to the teleoperator via calculated joint torques on the Leader robot. During the experiments, the two robots are located in a single room divided by a curtain to obscure the teleoperator's view of the Follower robot. The two robots, cameras, and FT-sensor are all connected to a PC run with a launch file written with ROS Noetic. A configuration file specifying the parameters of the set of VFs and network delays per experiment setup was used for convenience. The details of the network configuration are further discussed in Section IV-C.

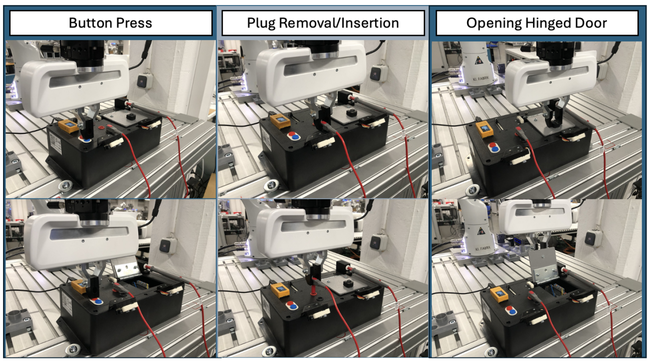

Fig. 4. Close up view of Follower Robot during task execution: Column 1) Press push buttons; Column 2) Pick up and insert Multimeter plug into a probe port; 3) Grasp and pull open the hinged door.

TABLE I

TABLE SHOWING THE SELECTED VF PRIMITIVE AND DEFINITION DEFINED FOR THE SET OF VF_i IN THE MULTI-STEP TASK SEQUENCE

Step	Task	Procedure	Primitive Type T	Physical Task Feature	CAD Feature	VF Definition P
1, 5	Press Button	Approach Button Push along Axis	Sphere Approach	Finger Thickness b Button Radius r_b	Button Axis Button Centroid $\vec{x}_T^{VF}$	$r_{VF_1} = r_b$, $r_{A_1} = r_b + \frac{b}{2}$
2	Remove Plug	Approach Plug & Grasp Pull along Hole Axis	Sphere Grasp Cylinder Remove	Plug Width a Peg Length h Peg Radius r_p Clearance s	Peg Axis Peg Centroid $\vec{x}_T^{VF}$	$r_{VF_2} = r_p$, $r_{A_2} = r_p + \frac{b}{2}$, $r_{A_3} = r_p + \frac{b}{2}$, $l_{VF_3} = h + s$
3	Insert Plug	Approach Hole Push along Axis	Cylinder Insert	Hole Radius r_h Peg Length h	Hole Axis Hole Centroid $\vec{x}_T^{VF}$	$r_{VF_4} = r_h$ $r_{A_4} = r_p + \frac{b}{2}$ $l_{VF_4} = h + s$
4	Open Door	Approach Handle & Grasp Pull along Curved Path	Sphere Grasp Torus Pull	Handle Thickness t_H Handle-Hinge Distance d_H	Hinge Axis Handle Centroid $\vec{x}_T^{VF}$ Major Radius R Minor Radius r_{VF}	$r_{VF_5} = t_H$, $r_{A_5} = t_H + \frac{b}{2}$, $r_{VF_6} = t_H$, $R = d_H$

A. Teleoperator Manipulation Test Sequence

The electronic task board [13] provides industry-relevant manipulation tasks inspired by a Multimeter electrical inspection use case, and therefore, we focused on a subset of the presented tasks following a timed 5-step sequence, shown in Fig. 4 and described below, to evaluate the teleoperator's performance with and without VFs. The test set up is reset to the same starting configuration after each experiment.

Task 1: Pressing a Push Button evaluates the teleoperator's ability to maneuver the remote Follower robot to a relatively small area (5mm radius about the button face center) and then press down to activate the button. An electrical circuit confirms the successful completion the task.

Task 2: Grasping and removing a Multimeter Plug involves accurately positioning the remote Follower robot and activating the gripper to grasp a Multimeter probe plug and pull it straight out of its receptacle. An electrical circuit confirms the successful removal of the plug.

Task 3: Plugging in Multimeter Plug into Test Port requires moving the remote Follower robot with the grasped plug above a different plug receptacle and inserting the plug until it is fully seated in the socket. This is an example of the highly studied peg-in-hole manipulation task. An electrical circuit confirms the successful insertion of the plug.

Task 4: Grasping and Opening a Hinged Door challenges the teleoperator to accurately grasp a door knob and then pull it open following an arc in a constrained motion. An angle sensor on the door hinge confirms when the door has been successfully opened.

Task 5: Pressing the Red Button is now a familiar task similar to Task 1 but now the teleoperator has more experience piloting the remote Follower robot. An electrical circuit confirms the successful completion of the task and this action also marks the end of the test sequence.

The following metrics are used to assess individual performance during the pilot user study:

Metric 1: Task Success Rate evaluates how many of the five defined tasks are successfully completed (pass/fail) in a given experiment. Tasks are considered failed when the teleoperator decides to skip a task they consider too difficult or when the Follower robot experiences excessive forces and crashes due to a collision with the environment. Higher the success rate corresponds to better performance.

Metric 2: Task Execution Time is measured from when the experiment coordinator indicates the start and end of a task by visual inspection and presses a key on the keyboard. Teleoperators can decide to skip a task after 60 seconds have elapsed. Smaller task execution time is better.

Metric 3: Total Distance Traveled examines the total Euclidean distance traveled by the Follower robot EE during the test sequence execution. Shorter distances are considered better performance.

B. Virtual Fixture Activation

An experiment begins with a launch script to align the Leader and Follower robots and zero the values of the FT sensor. The Leader robot enters a gravity compensated free drive mode where the teleoperator can guide the robot with their hands to produce their desired motion on the Follower robot. Key mappings on the experiment coordinator's keyboard control the progression of the test. The test time is started when the experiment coordinator presses the test sequence start button, "a" to enable the VFs and "b" to begin without VFs enabled. The experiment coordinator sits next to the teleoperator and resets the task board and robots to the same start conditions for each user. During the test the teleoperator gives verbal commands to the experiment coordinator to open and close the Follower robot's gripper who does so by pressing the "o" or "c" key respectively. The coordinator also confirms the completion of each task by pressing "d" to advance to the next task and automatically record the task execution time and distance traveled.

C. Network Configuration

Communication delays can greatly reduce the stability of a teleoperation system with haptic feedback, especially when interacting with the environment. To minimize the effects of network delays we build on previous work with the state-of-the-art time-domain passivity approach with energy reflection method (TDPA-ER) [15] to minimize position drift of the Follower robot and to best track the haptic force feedback on the Leader robot.

The Leader and Follower robots are connected to a gigabit network switch and controlled by a single computer running a 1kHz control look on a Ubuntu 22.04 real-time kernel. The robots are controlled over the Franka Control Interface (FCI) as documented in [18] and a series of ROS launch files [20] synchronize the starting positions of the Leader and Follower robots, start and synchronize the camera streams, and communicate with FT-sensor. We test teleoperation performance at three levels of network roundtrip latency (0 ms, 100 ms, 250 ms) achieved by adding wait timers to message publishers to study how virtual fixtures can support the teleoperators in carrying out their tasks in scenarios with progressively higher network congestion. This means video frames and haptic feedback arrive at the teleoperator with a delay often leading to hesitation, over-corrections, and unwanted collisions with the environment. A connection diagram for the experimental setup is illustrated in Fig. 5.

V. RESULTS

A pilot user study including eight individuals performing six teleoperation experiments each for scenarios with and without VFs activated was conducted for three network delay test conditions. During the pilot study, participants were allowed to repeat the experiment until completing all tasks. A total of sixty-seven experiments were collected and included in the analysis. Each participant was briefed on the test sequence and allowed to practice moving the Follower robot by hand-guiding the Leader robot for 2-minutes prior to starting their test. Participants were not informed of their test condition to avoid biasing the data. Overall, the collected data indicated higher performance when VFs are activated versus no VFs activated across all three metrics.

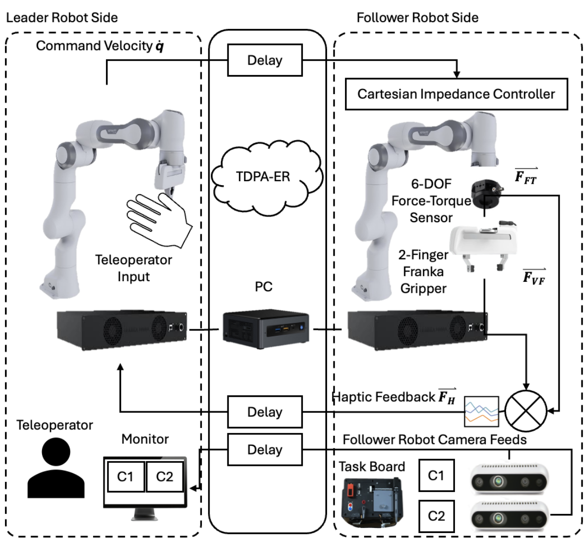

Fig. 5. Network configuration for a single PC controlling the leader-follower robot setup using Time-Domain Passivity Approach with Energy Reflection (TDPA-ER) [15]. The command delay to the Leader robot and the force feedback delay of the Follower robot are adjusted equally to achieve the desired round trip network delay which is synchronized with the frame rate of the camera streams.

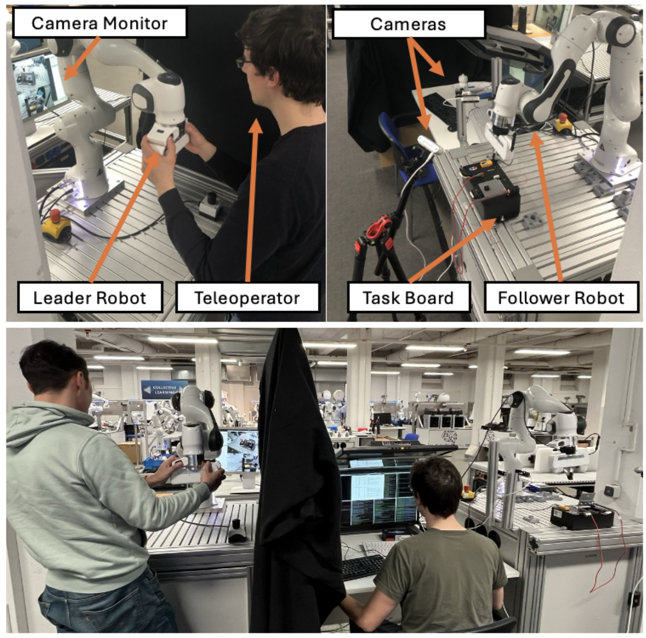

Fig. 6. Left: Leader robot with teleoperator; Right: Follower robot with task board. Bottom: View of both the leader-follower robot setups with the Experiment Coordinator's workstation in between.

The mean success rates across all participant trials for the tasks [1,2,3,4,5] (without VFs : with VFs) were [100%:100%, 100%:100%, 78%:83%, 67%:83%, and 100%:100%] respectively. All participants completed all tasks in the test sequence, however, some participants made mistakes or chose to skip tasks which were recorded as failed attempts. Mistakes occurred during the plug insertion, usually due to a poorly located grasp on the plug or losing grip on the door handle while opening. Some operators encountered non-recoverable robot crashes during the plug insertion trying to drive into the plug receptacle. This was more prominent at higher network delays and when VFs were not activated. In these cases, the experiment was stopped, reset and the teleoperator allowed to repeat their attempt.

Considering the task execution time, the increased network delay corresponded with a increased mean time required to complete the full test sequence as expected. Similarly, the data shown in Fig. 7 confirm that VFs, when activated, reduce the mean task execution time for each network delay condition. Interestingly, the mean task execution time for the worst network delay of 250ms with VFs activated was faster than without network delay (0ms) and no VFs activated, further supporting the case for VFs.

The total distance traveled by the Follower robot during the full test sequence is shown in Fig. 8. While the reduction in total distance traveled metric does not have a significant improvement with VFs versus without VFs, there is a small but measurable decrease when VFs are activated.

VI. DISCUSSION

Across the five task steps in the test sequence, participants stated the most difficult task was the plug insertion. This task was difficult because the teleoperator could grasp the plug in several different locations and angles which would then

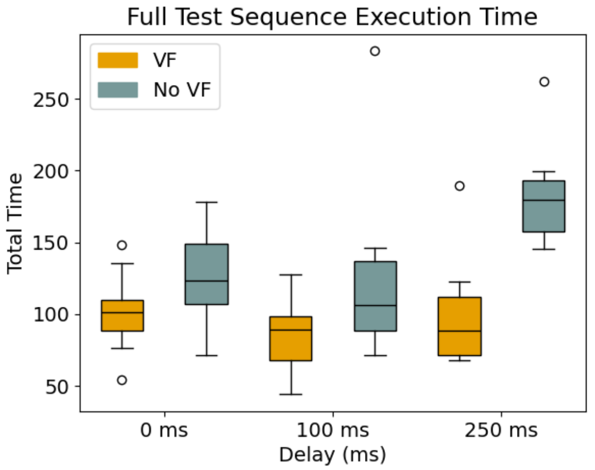

Fig. 7. Box plots comparing the mean total execution time and standard deviation for teleoperation with and without VFs enabled with increasing network delay test conditions. Notably, when VFs are enabled the mean execution time is maintained while performance degrades with higher network delays when VFs are not enabled.

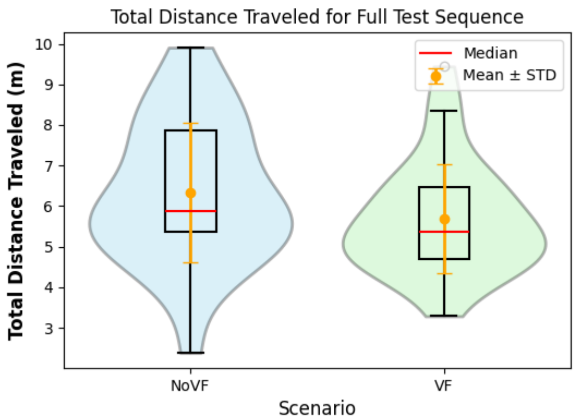

Fig. 8. Violin plot of the mean and distribution of the total distance traveled by the Follower robot end effector for the manipulation task sequence comparing the effect of using VFs versus not using VFs. Notice the shorter distance traveled when VFs are enabled.

affect how they would approach the insertion of the plug into the test port. The plug has a long barrel which could rotate in the robot's grasp and change the new tool tip when feeling around for the port. This potential issue was most present in the higher network delay conditions. While the plug grasp VF guided teleoperators to a the same point, they could still maneuver the Follower robot to different grasp points along the plug with enough force making the following insertion VF less helpful. Further research could look into modulating the VF guiding force based on their influence on future actions.

The VFs significantly supported the participant's test sequence execution time, especially in the presence of network delays. Pressing the push buttons was relatively easy for participants without VFs, but their execution speed increased when the VFs were activated. The button press task was considered to have the lowest risk of failure as the task did not involve any moving parts, where the plug could rotate during the insertion task, and the door knob could slip out of the robot's fingers during the door opening task. While it was anticipated that the door opening task would be the most difficult task, participants completed the task with a higher speed and higher success rate when the door opening VF was activated over the non-activated case.

A. Limitations & Future Work

The VF generation pipeline with ISO-10303 STEP AP242 files is promising as they are designed to be the interchangeable format for CAD data, however, our approach has some limitations. Currently, we only support detection of cylindrical surfaces to define axes and surface center points. The order and selection of the VF primitives still require human input who understands the goals of the manipulation task. Path, parallel, tangent and other constraints are considered the scope of future work. Furthermore, since we consider the robot EE as a point geometry we cannot provide robot tool orientation guidance with our current implementation. The task board CAD assembly file used in this study was designed in Onshape[2] and consists of several individual parts fitting together with mate constraints. STEP AP242 files from other CAD software providers may yield different results.

Future work includes testing the VF generation pipeline with exports of other object assemblies from other popular CAD software including Fusion360 and OpenSCAD.

VII. CONCLUSION

In this work we proposed a method for generating virtual fixtures from CAD data as a shared control assistance tool to perform precise manipulation in a leader-follower teleoperation setup. A test sequence of three different manipulation tasks was used in a pilot user study to compare the effects of virtual fixtures across two test scenarios with and without VFs for three network conditions with delays of 0 ms, 100 ms, and 250 ms. We presented results showing teleoperation manipulation performance degrades with increased network delays without VFs and when VFs are enabled teleoperation manipulation performance increases. Specifically, we found teleoperators had the best performance in test scenarios with the greatest delay with VFs enabled versus the test scenario without delays and without VFs. Teleoperators felt the benefits of VFs are more pronounced and would trust the guiding force more in higher network delay scenarios. Teleoperators anecdotally reported feeling more confident in executing manipulation tasks when the VFs were activated. In closing, we believe VFs present a clear benefit for teleoperation and we welcome feedback on our generation pipeline to make them more accessible to the robotics community.

ACKNOWLEDGMENT

This work was supported by the European Union's Horizon 2020 research and innovation program as part of the project euROBIN under grant no. 101070596. We gratefully acknowledge the funding of robo.innovate by the Bavarian State Ministry for Economic Affairs, Regional Development and Energy (StMWi) and like to thank the initiative "Gründerland Bayern" for the continuous support.

[2]Onshape: http://onshape.com/

References

[1] L. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *Proceedings of IEEE Virtual Reality Annual International Symposium*, 1993, pp. 76–82.

[2] J. Singh, A. R. Srinivasan, G. Neumann, and A. Kucukyilmaz, "Haptic-guided teleoperation of a 7-dof collaborative robot arm with an identical twin master," *EEE Trans. Haptics*, vol. 13, no. 1, p. 246–252, Jan. 2020. [Online]. Available: https://doi.org/10.1109/TOH.2020.2971485

[3] S. A. Bowyer, B. L. Davies, and F. Rodriguez y Baena, "Active constraints/virtual fixtures: A survey," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 138–157, 2014.

[4] H. Ishida, J. A. Barragan, A. Munawar, Z. Li, A. Ding, P. Kazanzides, D. Trakimas, F. X. Creighton, and R. H. Taylor, "Improving surgical situational awareness with signed distance field: A pilot study in virtual reality," *arXiv preprint arXiv:2303.01733*, 2023.

[5] P. Marayong, M. Li, A. Okamura, and G. Hager, "Spatial motion constraints: theory and demonstrations for robot guidance using virtual fixtures," in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 2, 2003, pp. 1954–1959 vol.2.

[6] H. Ishida, M. M. Marinho, K. Harada, J. Gao, and M. Mitsuishi, "Virtual-fixtures for robotic-assisted bi-manual cutting using vector-field inequalities," in *2020 IEEE/SICE International Symposium on System Integration (SII)*, 2020, pp. 395–400. [Online]. Available: https://doi.org/10.1109/SII46433.2020.9026172

[7] B. Khademian and K. Hashtrudi-Zaad, "Shared control architectures for haptic training: Performance and coupled stability analysis," *Int. J. Rob. Res.*, vol. 30, no. 13, p. 1627–1642, Nov. 2011. [Online]. Available: https://doi.org/10.1177/0278364910397559

[8] T. Tsumugiwa, R. Yokogawa, and K. Hara, "Variable impedance control with virtual stiffness for human-robot cooperative task (human-robot cooperative peg-in-hole task)," in *Proceedings of the 41st SICE Annual Conference*, 2002, pp. 2329–2334, sICE Annual Conference (Osaka, 2002). No DOI listed in public indexes; see SICE/J-STAGE / CiNii records. [Online]. Available: https://cir.nii.ac.jp/crid/1570572700873025792

[9] D. F. Prado and E. Steinbach, "Estimating virtual fixture parameters in digital twin environments for robot manipulation tasks using reinforcement learning," in *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, 2023, pp. 1–6.

[10] D. F. Prado, K. Larintzakis, J. Irsperger, and E. Steinbach, "Accelerating virtual fixture estimation for robot manipulation using reinforcement learning and human demonstrations," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, 2024, pp. 2013–2018.

[11] S. K. Mohammed, M. H. Arbo, and L. Tingelstad, "Leveraging model based definition and step ap242 in task specification for robotic assembly," *Procedia CIRP*, vol. 97, pp. 92–97, 2021, 8th CIRP Conference of Assembly Technology and Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827120314281

[12] Y. Pane, M. H. Arbo, E. Aertbeliën, and W. Decré, "A system architecture for cad-based robotic assembly with sensor-based skills," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1237–1249, 2020.

[13] P. So, A. Sarabakha, F. Wu, U. Culha, F. J. Abu-Dakka, and S. Haddadin, "Digital robot judge: Building a task-centric performance database of real-world manipulation with electronic task boards," *IEEE Robotics & Automation Magazine*, pp. 2–14, 2023.

[14] F. Abi-Farraj, C. Pacchierotti, O. Arenz, G. Neumann, and P. R. Giordano, "A haptic shared-control architecture for guided multi-target robotic grasping," *IEEE transactions on haptics*, vol. 13, no. 2, p. 270–285, 2020.

[15] X. Xu, M. Panzirsch, Q. Liu, and E. Steinbach, "Integrating haptic data reduction with energy reflection-based passivity control for time-delayed teleoperation," in *2020 IEEE Haptics Symposium (HAPTICS)*, 2020, pp. 109–114.

[16] T. Paviot, "pythonocc," Dec. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7471333

[17] V. Pruks and J.-H. Ryu, "Method for generating real-time interactive virtual fixture for shared teleoperation in unknown environments," *The International Journal of Robotics Research*, vol. 41, no. 9-10, pp. 925–951, 2022. [Online]. Available: https://doi.org/10.1177/02783649221102980

[18] S. Haddadin, "The franka emika robot: A standard platform in robotics research [survey]," *IEEE Robotics & Automation Magazine*, vol. 31, no. 4, pp. 136–148, 2024.

[19] Intel. (2023) Ros wrapper for intel realsense cameras. [Online]. Available: https://github.com/IntelRealSense/realsense-ros

[20] Stanford Artificial Intelligence Laboratory et al., "Robotic operating system." [Online]. Available: https://www.ros.org