5G Virtual Reality Manipulator Teleoperation using a Mobile Phone

Alexander Werner and William Melek

***Abstract*— This paper presents an approach to teleoperate a manipulator using a mobile phone as a leader device. Using its IMU and camera, the phone estimates its Cartesian pose which is then used to to control the Cartesian pose of the robot's tool. The user receives visual feedback in the form of multi-view video - a point cloud rendered in a virtual reality environment. This enables the user to observe the scene from any position. To increase immersion, the robot's estimate of external forces is relayed using the phone's haptic actuator. Leader and follower are connected through wireless networks such as 5G or Wi-Fi. The paper describes the setup and analyzes its performance.**

I. INTRODUCTION

The principal goal of the presented approach is to provide users with a portable and intuitive way of teleoperating a manipulator. With regards to existing teleoperation interfaces, we present an extreme choice. The interface is already in everyone's pocket - a mobile phone. This guarantees portability but requires working within the limitations of the device.

Most manipulator teleoperation products are on the opposite end of the design space: a premium tabletop teleoperation workstation which includes either a virtual reality (*VR*) headset or an autostereoscopic display for visual feedback combined with a six-degree-of-freedom haptic device equipped with fine-grained position sensing and force feedback abilities. None of these three modalities can be directly replicated on a mobile phone.

As illustrated in Fig. 1, our approach implements visual feedback by rendering a point cloud of the scene containing the robot and the environment. The user can navigate the scene by moving the phone, which remedies the lack of depth perception on the phone display. For control, we estimate the pose of the mobile phone using proprioceptive and exteroceptive sensors: the onboard inertial measurement unit (IMU) and camera. With the activation of a virtual clutch, changes in this pose are replicated 1:1 at the position level by the manipulator in both translation and orientation (SE3) if and as desired.

To relay contact forces to the operator, the built-in haptic actuator is used, but this is hardly equivalent to the force feedback a purpose-designed haptic interface can generate. The actuator is incapable of generating static forces, and only minute dynamic forces in one direction can be generated. We relay static forces with an amplitude modulation scheme driving the haptic actuator. Especially for haptics, a reliable network connection with low latency provided by wireless networks such as 5G is important.

Authors are with the University of Waterloo, Waterloo, Canada.
name.surname@uwaterloo.ca

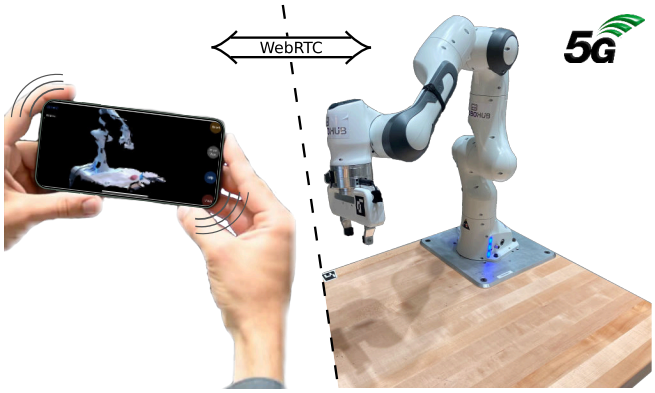

Fig. 1. Experimental setup used for evaluation.

Where can this approach be useful? We think that the best use case is in the temporary control of a manipulator in a scenario where human understanding of the manipulation task can directly be applied. Due to the temporary nature, conventional haptic interfaces are unlikely to be an efficient choice. An example for such a scenario is error recovery for industrial or collaborative manipulators. Our approach allows the operator to intervene remotely using a more intuitive interface than the traditional button-based pendants.

The contributions of this paper are the following. To the authors' knowledge, using a pose obtained from visual-inertial fusion on a mobile phone to directly control a manipulator pose in SE3 is novel. This paper also proposes a novel method to encode transient and static contact forces into haptic actuator feedback. Additionally, this paper demonstrates how to stream typical RGB-D point clouds over wireless networks to a mobile device considering bandwidth and computation limits while still achieving real-time rendering.

The remainder of this paper is structured as follows. After a review of the relevant literature in Sec. II, Sec. III describes the design choices which led to the approach detailed in Sec. IV. Sec. V then describes the experimental setup used to evaluate the approach generating the data presented in Sec. VI. Finally, Sec. VII provides a conclusion.

II. RELATED WORK

The authors of [1] present an approach where only proprioceptive sensors of a mobile device are used to teleoperate a manipulator. In this approach, the orientation of the mobile device is mapped to the angular velocity of the manipulator while the translation velocity is controlled via the touch screen. This approach seems quite intuitive and is well adapted for use as an assistive device for people

with limited mobility in the upper extremities as it does not require to move the device in translation. Teleoperation of a manipulator using a mobile device also has been presented in [2], the approach uses proprioceptive sensors only. It maps the translation and orientation offsets of the phone directly to the manipulator. Both papers focus on control and do not describe any visual or haptic feedback on the mobile device. Shared autonomy-based teleoperation interfaces and interactive manipulation methods can be aided by object recognition as shown in [3]. Other alternative interfaces for teleoperation of a manipulator include using human pose estimation include using RGB-D cameras [4], [5], or using controllers together with a VR headset [6].

For anchored teleoperation setups, research has provided insights into the requirements of *haptics* as part of force-feedback [7]. Feedback using haptic actuators has been explored for teleoperation for some time. [8] presents an approach using haptics-enabled armbands and uses hand-tracking for controlling the robot's pose. The paper also shows that having haptic feedback is preferred by the user's teleoperation setup.

Visual feedback in the form of multi-view video streaming has been used in [9] where multiple RGB-D cameras are merged to obtain complete coverage of the scene. This paper uses a VR headset to allow the user to view the point cloud from multiple angles with depth perception. For bandwidth-constraint wireless applications, compression of either point clouds or RGB-D data is of importance, [10] compares various approaches. [11] presents applied point cloud compression and streaming to mobile devices. The Motion Picture Experts Group (MPEG), also involved in *H.264* is developing codecs for streaming point cloud data [12]. Utilizing edge computing to remove the burden of streaming and rendering the point cloud is a popular idea, [13] describes a framework. Most codecs for depth data compression implement lossy compression. Jiffy Codec [14] provides a lossless approach to the compression of depth data coming from LIDAR or RGB-D sensors. It utilizes single instruction multiple data (SIMD) based integer compression.

From a networking perspective, [15] gives a list of connectivity requirements for a teleoperation setup. [16] discusses 5G teleoperation-specific aspects. [17] presents a comparison of networking methods for teleoperation, including 5G.

III. DESIGN

In the quest to implement a user-friendly system for mobile teleoperation of manipulators, we observe three different dominating modalities: control of the manipulator position, visual feedback and feedback about contacts.

Alteration of the manipulator pose: Conventional haptic devices for teleoperation provide position-level continuous input in SE3. Any other approach such as velocity-level interfaces or discrete input (buttons) greatly reduces the intuitiveness. Especially when involving haptic feedback, it is crucial to allow the user to react with reflexes to newly encountered contacts. In our approach, we want to replicate the user experience during kinesthetic teaching or hand guiding of a robot. Just that instead of grasping the tool, the user grasps the phone and moves the tool indirectly. A clutch button has to be pressed and held by the user to actually move the robot. As user observes the robot through the VR environment from a known position, the motions of the phone are applied to the robot in a way that the manipulator shown in the VR environment moves in the same way. Hence, the input of the user is always relative to the current pose of the manipulator, but the mapping takes into account the absolute position of the user in the VR space.

Control of manipulator position and integrated force control: While optimal transparency can only be realized by small time constants in all subsystems, this imposes high requirements on the used devices and the network connecting them. We replace the need for perfect tracking and force feedback with a compliant control and simplified force feedback requiring a less complex leader device. As there is no way to constrain the motion of the operator holding the phone, the operator can push the robot into contact and create excessive contact forces when using a non-compliant controller. In the presented approach, the compliant control approach implements an upper limit to the contact forces. This is also relevant in case the pose estimated by the visual-inertial fusion can occasionally contain outliers. As an added feature, a compliant control approach allows the user to control the interaction force indirectly by pushing in the direction of a contact constraint.

Contact feedback: Information about contacts is necessary for effectively working within a teleoperation environment. Again, our objective is to come as close as possible to 6D force feedback provided by haptic interfaces using the available hardware on a mobile device. As only a single haptic actuator is available in current devices, reproducing arbitrary static forces or force in multiple directions is impossible. Hence, we designed a system which enables the user to sense transients related to the creation of a first contact and additionally sense continuous forces through an amplitude modulation scheme. All different force directions are mapped to a single haptics channel. For high-frequency components of the forces created when establishing a new contact, we relay a transient haptic signal which is scaled to the impulse the robot loses when making this contact. This removes the need for high-frequency sensor-based contact force measurement and allows to rely on estimation of external forces. For continuous contact forces, we relay a continuous haptic signal which is scaled with the magnitude of the norm of the contact forces.

Visual feedback: We opted for a multi-view live stream of the scene. The multi-view video stream is presented in a virtual reality environment which allows the user to reposition the view port by moving the phone. Motions of the user are mapped 1:1 into the virtual environment. Additionally, the design includes a clutch which temporarily disables the movement in the VR space. This allows the user to index through the space similar to indexing (activation of the clutch) for control or teleportation in other VR applications. We opted to capture both the scene and the

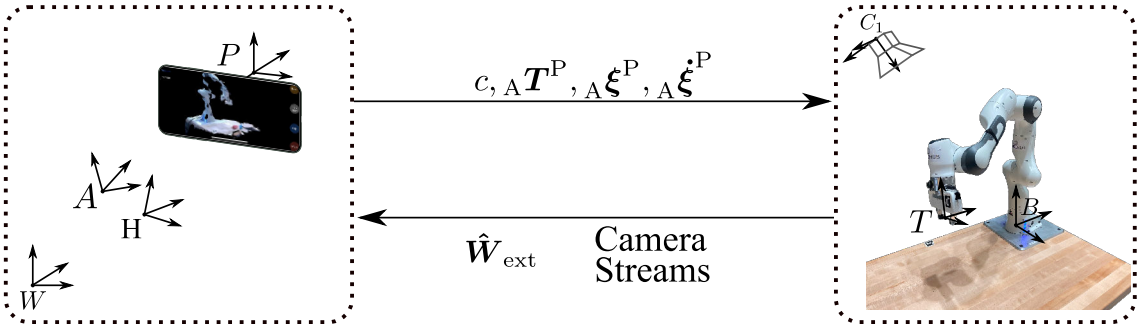

Fig. 2. Leader and follower system and transmitted data streams. Also shown: Relevant frames.

robot with RGB-D cameras and aim to provide 360-degree coverage. In our setup, we implement multi-view video by streaming separate colour and depth data from a number of *RGB-D* cameras to a phone where they are assembled into one point cloud. Extrinsic properties of the cameras w.r.t. the robot are obtained beforehand.

Connectivity: The objective is to propagate the real-time streams shown in Fig. 2 over a wireless network. Those are the command (pose, velocity, acceleration, clutch status), and feedback consisting of the external forces and video streams. Each stream has its own properties in terms of bandwidth, acceptable latency and reliability. Another design goal was to build a system which works in realistic network environments and reliably establishes a connection in the presence of firewalls or other network obstructions. Teleoperation is a perfect example of a peer-to-peer application and is directly impacted by those. *WebRTC* [18] is a proven design for video streaming with the option to add additional real-time data streams. It uses *UDP* to encapsulate video streams and is able to work around said network obstructions using interactive connection establishment (ICE). Additionally, it implements transport layer security encryption. Alternatives include *ZeroMQ* or *ROS2/DDS* but they provide only a subset of these features.

IV. CONTROL & FEEDBACK

In this section, we describe the control scheme across the different components. This comprises the leader and the follower side, refer to Fig. 2 for an overview. The following frames are defined: W is a world-fixed frame on the leader side, P is a frame attached to the mobile device, B is the robot base frame, and T is the robot tool frame. We proceed with the convention of denoting a transformation which transforms from frame Y to frame X as ${}_{\mathrm{X}}\boldsymbol{T}^{\mathrm{Y}} \in \mathrm{SE}(3)$. The transformation can be defined using the translation ${}_{\mathrm{X}}\boldsymbol{p}^{\mathrm{Y}}$ and Rotation matrix ${}_{\mathrm{X}}\boldsymbol{R}^{\mathrm{Y}}$, and has the associated twist ${}_{\mathrm{X}}\boldsymbol{\xi}^{\mathrm{Y}}$. The meaning of the auxiliary frames A and H is described below, frame C_1 is the optical frame of the first camera. The Boolean clutch state is denoted as c.

A. Visual feedback

For visual feedback, the goal is to present the user with a freely navigable view of the scene, including the robot. On the robot end, we use multiple RGB-D cameras to capture the scene. The camera positions ${}_{\mathrm{B}}\boldsymbol{T}^{\mathrm{C,i}}$ are known w.r.t. the robot's base. We compress the colour image stream from each camera using the video codec *H.264*. For the depth stream compression Jiffy compression [14] is used. After decompression, the known camera intrinsics are used to deproject the depth images to a point cloud. For indexing through the VR world, we introduce the frames A and H. The position of these frames can be changed by the user by holding the "Hold view" button and moving the phone. During holding, the view port will not change. This allows the user to remain at one location and e.g. look at the scene from an opposite viewpoint without physically moving there.

$$ {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{P}} = \begin{cases} {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{H}} & \text{hold view} \\ {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{W}}{}_{\mathrm{W}}\boldsymbol{T}^{\mathrm{P}} & \text{free} \end{cases} \tag{1} $$

$$ {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{W}} = \begin{cases} {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{H}}{}_{\mathrm{P}}\boldsymbol{T}^{\mathrm{W}} & \text{hold view} \\ \text{unchanged} & \text{free} \end{cases} \tag{2} $$

$$ {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{H}} = \begin{cases} \text{unchanged} & \text{hold view} \\ {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{P}} & \text{free} \end{cases} \tag{3} $$

$$ \tag{4} $$

The point cloud of camera i is shown in the VR environment at:

$$ {}_{\mathrm{P}}\boldsymbol{T}^{\mathrm{C}} = {}_{\mathrm{P}}\boldsymbol{T}^{\mathrm{A}}{}_{\mathrm{B}}\boldsymbol{T}^{\mathrm{C,i}} \tag{5} $$

this shows that the frame A in the VR space is equivalent to frame B in the scene. To obtain the ${}_{\mathrm{W}}\boldsymbol{T}^{\mathrm{P}}(t)$, a visual-inertial fusion is used. The description of such is outside the scope of this paper. One possible approach is described in [19]. Initially, ${}_{\mathrm{W}}\boldsymbol{T}^{\mathrm{P}} = \boldsymbol{I}$ with the frame W being gravity-aligned.

B. Command

The visual-inertial fusion additionally estimates the translation velocity $\boldsymbol{v}_{\mathrm{P}}$. We can obtain the rotation velocity $\boldsymbol{\omega}_{\mathrm{P}}$ and the translation acceleration $\boldsymbol{a}_{\mathrm{P}}$ directly from sensor data. To move the manipulator, the user enables the clutch at time t_{c} at the position ${}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{P}}(t_{\mathrm{c}})$. The Cartesian Impedance controller received the following the desired pose:

$$ {}_{\mathrm{B}}\boldsymbol{T}^{\mathrm{T,d}}(t) = {}_{\mathrm{B}}\boldsymbol{T}^{\mathrm{T}}(0){}_{\mathrm{T}}\boldsymbol{T}^{\mathrm{D}}(t) \tag{6} $$

with the offset ${}_{\mathrm{T}}\boldsymbol{T}^{\mathrm{D}}$ from the initial pose of the tool. When the clutch is engaged, ${}_{\mathrm{T}}\boldsymbol{T}^{\mathrm{D}}$ is updated using

$$ {}_{\mathrm{T}}\boldsymbol{T}^{\mathrm{D}}(t) = {}_{\mathrm{T}}\boldsymbol{T}^{\mathrm{D}}(t_{\mathrm{c}}) \begin{bmatrix} \boldsymbol{I} & -{}_{\mathrm{A}}\boldsymbol{p}^{\mathrm{P}}(t_{\mathrm{c}}) \\ \boldsymbol{0} & 1 \end{bmatrix} {}_{\mathrm{A}}\boldsymbol{T}^{\mathrm{P}}(t) \begin{bmatrix} {}_{\mathrm{P}}\boldsymbol{R}^{\mathrm{A}}(t_{\mathrm{c}}) & \boldsymbol{0} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{7} $$

and left constant otherwise.

C. Manipulator Control

We assume the following rigid-body dynamics of the manipulator with the states $\boldsymbol{q}$ and $\dot{\boldsymbol{q}} \in \mathbb{R}^N$:

$$ \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} + \boldsymbol{J}^{\mathrm{T}}(\boldsymbol{q}) \cdot \boldsymbol{W}_{\mathrm{ext}} \tag{8} $$

with the Mass matrix $\boldsymbol{M}$, Coriolis matrix $\boldsymbol{C}$, gravity terms $\boldsymbol{g}$, Jacobian $\boldsymbol{J}$ associated with the frame T and external wrench $\boldsymbol{W}_{\mathrm{ext}}$ acting on that frame. The actuator torques $\boldsymbol{\tau}$ are computed by a Cartesian impedance control approach with saturated force and torques:

$$ \boldsymbol{\tau} = \boldsymbol{J}^{\mathrm{T}} \left[\boldsymbol{K}(\boldsymbol{e}) \cdot \boldsymbol{e} + \boldsymbol{D}(\boldsymbol{e}) \cdot \dot{\boldsymbol{e}} \right] + \boldsymbol{\tau}_{\mathrm{ffwd}} \tag{9} $$

$$ \boldsymbol{\tau}_{\mathrm{ffwd}} = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{M}(\boldsymbol{q}) \left[\boldsymbol{J}^{\mathrm{T}} \ddot{\boldsymbol{x}}_{\mathrm{d}} + \dot{\boldsymbol{J}}^{\mathrm{T}} \dot{\boldsymbol{x}}_{\mathrm{d}} \right] \tag{10} $$

$$ \boldsymbol{K}(\boldsymbol{e}) = \left[{}^{1}\!/_{\max(0.1, \cosh(\boldsymbol{p}(\boldsymbol{e})))} \right]^2 \cdot \boldsymbol{K}_{\mathrm{nom}} \tag{11} $$

$$ \boldsymbol{p}(\boldsymbol{e}) = \boldsymbol{K}_{\mathrm{nom}} \begin{bmatrix} \mathrm{diag}(F_{\max}) & \boldsymbol{0} \\ \boldsymbol{0} & \mathrm{diag}(\tau_{\max}) \end{bmatrix} \cdot \boldsymbol{e} \tag{12} $$

with the constant diagonal matrix nominal stiffness $\boldsymbol{K}_{\mathrm{nom}}$, current stiffness matrix $\boldsymbol{K}(\boldsymbol{e})$, the damping matrix $\boldsymbol{D}(\boldsymbol{e})$, and feed-forward torques $\boldsymbol{\tau}_{\mathrm{ffwd}}$. The error $\boldsymbol{e}$ is defined as

$$ \boldsymbol{e} = \begin{bmatrix} {}_{\mathrm{B}}\boldsymbol{p}^{\mathrm{T,d}} - {}_{\mathrm{B}}\boldsymbol{p}^{\mathrm{T}} \\ \boldsymbol{r}_{\mathrm{err}}({}_{\mathrm{T}}\boldsymbol{R}^{\mathrm{B}}\, {}_{\mathrm{B}}\boldsymbol{R}^{\mathrm{T,d}}) \end{bmatrix} \tag{13} $$

with the position of the frame T $\boldsymbol{p}_{\mathrm{T}}$, orientation $\boldsymbol{R}_{\mathrm{T}}$, and angular error function $\boldsymbol{r}_{\mathrm{err}}$. $\boldsymbol{D}(\boldsymbol{e})$ derived using double-diagonalization based damping design [20] taking into account the current stiffness $\boldsymbol{K}(\boldsymbol{e})$.

D. Contact Feedback

We use a momentum-based external force observer [21] to obtain an estimate of $\boldsymbol{W}_{\mathrm{ext}}$, $\hat{\boldsymbol{W}}_{\mathrm{ext}}$:

$$ \hat{\boldsymbol{W}}_{\mathrm{ext}} = \boldsymbol{J}^{\mathrm{T}} \hat{\boldsymbol{\tau}}_{\mathrm{ext}} \tag{14} $$

$$ \hat{\boldsymbol{\tau}}_{\mathrm{ext}} = \boldsymbol{K}_{\mathrm{O}} \left[\boldsymbol{M}(\boldsymbol{q})(\dot{\boldsymbol{q}}) - \int \boldsymbol{\tau} - \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{C}^{\mathrm{T}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}_{\mathrm{ext}} \mathrm{d}s \right] \tag{15} $$

with $\boldsymbol{K}_{\mathrm{O}}$ a diagonal matrix with observer gain. For relaying the observed wrench $\boldsymbol{W}_{\mathrm{ext}}$ to the user, we determine the haptic pattern to be played based on the contact state. The contact state is determined with a hysteresis around the 2-norm of the forces $\boldsymbol{F}_{\mathrm{ext}} \in \mathbb{R}^3$.

The event of creating a contact is relayed to the user by playing a haptic pattern with the intensity i_{impulse} which replicates the impulse the robot lost when creating the contact, expressed as the impulse lost in a certain time window Δ approximated by:

$$ i_{\mathrm{impulse}} = i_{\mathrm{impulse,min}} + \left| \boldsymbol{S}\boldsymbol{J}\boldsymbol{M}\boldsymbol{J}^{T}\boldsymbol{J} \left[\dot{\boldsymbol{q}}(t) - \dot{\boldsymbol{q}}(t - \Delta) \right] \right|_2 \tag{16} $$

with the minimum intensity $i_{\mathrm{impulse,min}}$ and the selection matrix $\boldsymbol{S}$ which removes the impulse associated with the angular velocities.

After the contact has been created, the $\boldsymbol{F}_{\mathrm{ext}}$ is mapped to a different cyclic haptic pattern for which the intensity i_{cyclic} scales with the norm of the external forces:

$$ i_{\mathrm{cyclic}} = i_{\mathrm{cyclic,min}} + \left[|\boldsymbol{F}_{\mathrm{ext}}|_2 - F_{\mathrm{ext,thres}} \right]. \tag{17} $$

with the minimum intensity i_{cyclic} and the force threshold for contact activation $F_{\mathrm{ext,thres}}$. External torques are ignored in the current implementation.

V. EXPERIMENTAL SETUP

This section will go over the different components of the setup, with notes about implementation details and other insights.

For the leader system: we selected an *Apple IPhone* [22] because of the low-latency and high-fidelity haptic actuator. For the Wi-Fi tests an Iphone 11 was used, while an Iphone 14 was used for the 5G tests. While the application can be realized as a web app using the *WebXR* API, there is no access to the haptic actuator from within the browser sandbox. Hence, we implemented a native application. This additionally enables leveraging single-instruction-multiple-data (*SIMD*) parallelism for the compression of the depth data.

For the follower system: we implemented the control described in Sec. IV for a *Franka-Emika* collaborative robot [23], leveraging the provided *FCI* interface to control the robot. Our application provides $\boldsymbol{\tau}$ to the robot. The robot is controlled by a small form factor computer with an *Intel i7-8700T* CPU and a *NVidia Quadro P620* graphics card which also handles all remaining computational tasks.

Camera streams and compression: Four *Intel Realsense D435* cameras are configured to 848×480 resolution at 30fps. The colour streams are compressed using the *NVENC* [24] hardware accelerated *H.264* video codec using the graphics card. On the phone, the streams are decompressed using the *VideoToolbox* API, leveraging hardware acceleration. All colour stream compression is handled by *ffmpeg*. The depth streams are compressed using the Jiffy codec [14] which uses the *SIMD* CPU features for compression and decompression. We reimplemented the Jiffy codec in C++ for portability reasons and changed the integer compression library to *TurboPFor* [25] to support ARM platforms. We opted for 30fps even though the cameras we are able to stream at 60fps to avoid saturating the phone CPU.

Camera extrinsics: ${}_{\mathrm{B}}\boldsymbol{T}^{\mathrm{C,i}}$ is estimated first using hand-to-eye calibration. For cameras with significant overlap in point clouds, the extrinsic properties are refined using the iterative closest point algorithm. This allows for short setup time and flexible camera placement without a requirement for overlapping camera views, e.g. in typical extrinsic calibration approaches.

Impedance Controller: stiffness is set to $400^{\mathrm{N}}/_{\mathrm{m}}$ in translation, and $40^{\mathrm{Nm}}/_{\mathrm{rad}}$ in rotation using critical damping. The leader provides desired position ${}_{\mathrm{B}}\boldsymbol{T}^{\mathrm{T}}$, desired velocity ${}_{\mathrm{B}}\boldsymbol{\xi}^{\mathrm{T}}$ and desired acceleration ${}_{\mathrm{B}}\dot{\boldsymbol{\xi}}^{\mathrm{T}}$ used in the controller (12). The control approach is completed by a Nullspace controller regularizing the robot configuration. We found that the compliant control approach utilizing feed-forward terms as much as possible also reduces the sensitivity to transient errors in the estimated pose ${}_{\mathrm{W}}\boldsymbol{T}^{\mathrm{P}}$ which would otherwise generate significant jumps or at least noise on the robot side. In the setup only ${}_{\mathrm{W}}\boldsymbol{T}^{\mathrm{P}}$ is affected by those errors, as ${}_{\mathrm{B}}\boldsymbol{\xi}^{\mathrm{T}}$ and ${}_{\mathrm{B}}\dot{\boldsymbol{\xi}}^{\mathrm{T}}$ are directly derived from sensor data.

External force and impulse observer: We use the external force observer data provided through the *FCI* interface. We found that in our tabletop setup, the estimate of external force can be inaccurate when the robot is in motion. We are using a $10\mathrm{N}$ hysteresis activation threshold $F_{\mathrm{ext,thres}}$ before starting to render contact forces using the haptic actuator. This avoids signalling non-existent contacts to the user.

Haptic patterns: We have chosen a pattern for contact creation which feels like a sharp spike. For relaying a continuous contact force to the user, we chose a smoother continuously repeating pattern. We scale the patterns to cover a minimum and maximum static contact force between $10N$ and $40N$.

Phone pose: We make use of the *ARKit* API provided by the phone manufacturer to estimate the phone pose ${}_{\mathrm{W}}\boldsymbol{T}^{\mathrm{P}}$. See [26] for a comparison of the pose estimation performance. The user motions are not scaled, i.e. the user motion is replicated by the robot 1:1. To obtain ${}_{\mathrm{W}}\boldsymbol{\xi}^{\mathrm{P}}$ we directly use the output of the *CoreMotion* API which implements a filter which relies solely on data from proprioceptive sensors. For ${}_{\mathrm{W}}\dot{\boldsymbol{\xi}}^{\mathrm{P}}$ the translation part is also provided by *CoreMotion* and we use a differentiating filter to obtain the angular acceleration. The update rates provided by the *ARKit* and *CoreMotion* API are $60\mathrm{Hz}$ and $100\mathrm{Hz}$ respectively. There are no additional filters implemented in the complete setup. Suitable environments are required for the phone pose estimation to work reliably. Covering or blinding the camera will lead to drift thus making the point cloud rendering drift out of the user's view. Also, the command will be affected by this drift and this can lead to unintended motions.

Network Connection: For initial testing, we used a WiFi 4 access point which connects the phone to the robot. Alternatively, we connect to the Rogers *5G* non-standalone (NSA) test network available on campus using a *Quectel RM520N-GL 5G* modem. To implement the *WebRTC* communication between leader and follower we use *libdatachannel* [27]. Typically the used networks can exhibit a *stochastic packet loss* of below 1% and have non-zero latency and jitter. Hence robustness of the application under these conditions is important. Both real-time data streams for the phone pose ${}_{\mathrm{W}}\boldsymbol{T}^{\mathrm{P}}$ and its derivatives and the estimated external force $\boldsymbol{W}_{\mathrm{ext}}$ are fairly resilient to stochastic packet drops. However, video streams are typically more affected because for the computation of a frame, all fragments have to arrive. *H.264* is designed to be robust to a certain packet loss once an intra-coded frame is successfully received. The Jiffy Codec does not have these properties. Any dropped packet will make it impossible to allow computation of a frame until the next intra-coded frame arrives. The application imprints different traffic classes in the differentiated services field of the IP packets, this allows network equipment to prioritize contact feedback streams over video streams. To reduce network congestion during high changes in the point cloud and thus increase traffic we impose an upper limit on video streams while prioritizing the other feedback traffic. For the case of temporary loss of communication, i.e. drop of multiple packets in the control or force feedback streams for $100\mathrm{ms}$, the application is designed to disengage the clutch.

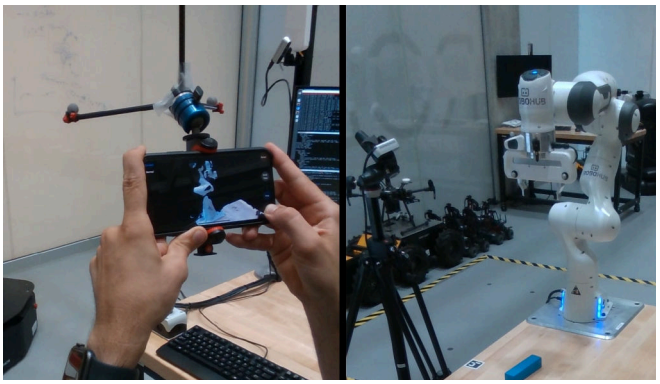

Fig. 3. Evaluation setup: Manipulator and phone equipped with tracking markers.

VI. PERFORMANCE EVALUATION

For the performance evaluation, we created a local setup with near-optimal wireless connectivity. We connected the phone to a wireless access point directly with minimal other traffic on this network. In this setup, we observed an application-level round trip delay between leader and follower of a minimum of $5\mathrm{ms}$, average of $20\mathrm{ms}$, and $60\mathrm{ms}$ maximum delay with minimal packet loss and reordering for Wi-Fi. For 5G NSA we measured $40\mathrm{ms}$, $66\mathrm{ms}$, and $84\mathrm{ms}$ respectively. For the final submission of the paper, this setup will completely move to 5G. The delay was measured using a separate *WebRTC* data channel, thus being affected by all user space, kernel, hardware and network aspects. As a qualitative result, we observed more consistent control over the manipulator position using 5G. We attribute this to the heavily utilized shared 5Ghz spectrum used by Wi-Fi at the test location.

For visual rendering, we observe an average delay created by compression, communication and decompression of the colour images $50\mathrm{ms}$ to $80\mathrm{ms}$ in the Wi-Fi scenario. A frame rate of $30\mathrm{fps}$ is transmitted and rendered without a significant number of frames being dropped. To evaluate tracking performance without unnecessary artifacts we equipped both leader and follower with markers for an external tracking system. This allows us to acquire both the pose of the phone and the robot's tool frame with $1kHz$ and no relative latency.

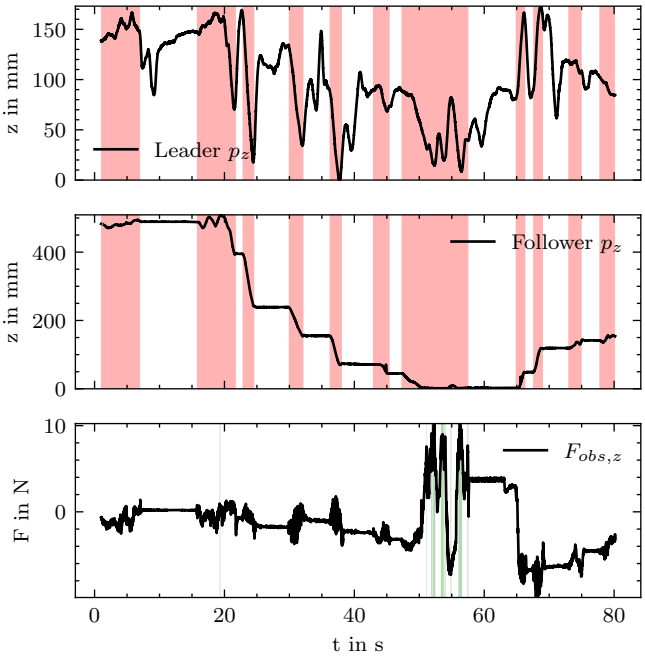

Fig. 4. Example user session captured together with the accompanying video. The top plot shows the position of the leader device in z direction, the middle plot the follower position. The bottom plot shows the estimated external forces. In the upper two plots the shaded areas are periods when the clutch is activated, in the lower plot the shaded areas denote an active contact.

At steady state, we observe a traffic from leader to follower of $0.4^{\text{MBit}/\text{s}}$ and $65^{\text{MBit}/\text{s}}$ in the other direction. Bandwidth requirements of Jiffy codec depth stream heavily depend on depth noise and associated filter setup. In general, the point cloud quality is the bottleneck for the user effectiveness in manipulation tasks as adjacent objects are easily merged. Visual feedback could significantly be improved with better sensor data.

In Fig. 4 a typical user session is shown. The shown data was recorded during the session submitted as the accompanying video. The user is making use of indexing to navigate the robot's workspace as can be seen by the cycles the clutch status is changed. The motion of the user is contained in a relatively small comfort zone (about $150mm$ in z). This is suitable for a user who does not use upper body motions as can be seen in the video. On the robot side, the z workspace is $500mm$. During the session, the user moves the robot into contact and receives haptic feedback about this. In the bottom diagram of the figure, it can be seen that the user is taking note of the created contact and does not further increase the contact force.

Fig. 5 shows tracking of leader and follower for a dynamic downwards motion. The data is recorded by the external tracking system which captures the pose of phone and tool, avoiding any misalignment of samples in time.

VII. CONCLUSION

We described the design and implementation of a teleoperation scheme using a mobile phone as a leader device.

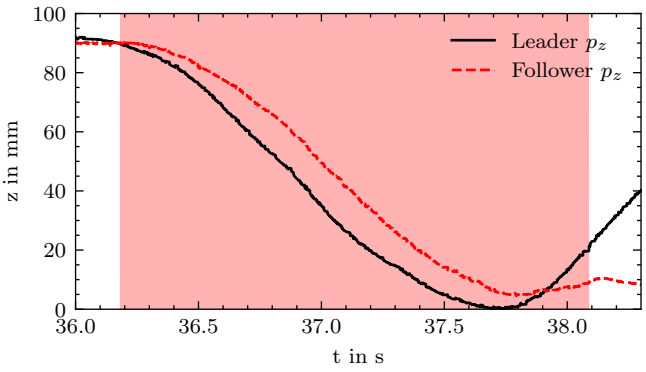

Fig. 5. Tracking performance during a dynamic motion as recorded by the external tracking system. The user holds the clutch engaged during the area shaded in red.

When the user moves the phone through space, the manipulator replicates the motions directly. The phone pose is estimated using the built-in proprioceptive and exteroceptive sensors IMU and camera and transmitted over wireless networks using *WebRTC*. To realize an intuitive interface we implemented a compliant control scheme which uses the estimated pose of the phone to control the manipulator pose. This scheme implements an upper limit to the static forces generated to eliminate potential damage to the environment or the robot. For contact feedback, we map the estimated external force on the manipulator to the haptic actuator on the phone. For visual feedback, we stream multiple *RGB-D* cameras using hardware-accelerated compression schemes to the phone where they are rendered into a point cloud. This enables the operator to freely navigate the scene and overcomes the limitations of teleoperation interfaces with only single camera view and also remedies the problem of missing depth perception on the phone display.

While some aspects of the mobile-phone-based teleoperation interface are clearly inferior to commercially available desk-mounted solutions for teleoperation of manipulators, the focus here is to enable teleoperation with a readily available low-cost leader device. From any place, at any time, connecting to a robot which could also be mobile. The strength of this approach lies in scenarios where teleoperation is used infrequently, e.g. to resolve situations which are not correctly handled or not covered by an autonomous application. Alternatively, this can be used as part of a shared-autonomy scheme for interactive manipulation.

To our surprise, we found that the visual feedback rather than the pose estimation is the bottleneck in this application.

Future Work: We observed significant development in the area of point cloud compression algorithms for real-time multi-view video streaming. Together with better and more sensors, we hope to address the main bottleneck of this approach.

To further improve immersion, we want to augment the displayed point cloud with an estimate of the contact point locations and contact force direction indicators.

REFERENCES

[1] L. Wu, R. Alqasemi, and R. Dubey, "Development of smartphone-based human-robot interfaces for individuals with disabilities," *Robotics and Automation Letters (RA-L)*, vol. 5, no. 4, pp. 5835–5841, 2020.

[2] C. Parga, X. Li, and W. Yu, "Tele-manipulation of robot arm with smartphone," in *6th International Symposium on Resilient Control Systems (ISRCS)*. IEEE, 2013, pp. 60–65.

[3] D. Kent, C. Saldanha, and S. Chernova, "A Comparison of Remote Robot Teleoperation Interfaces for General Object Manipulation," in *International Conference on Human-Robot Interaction (HRI)*. ACM/IEEE, 2017, pp. 371–379.

[4] A. Handa, K. Van Wyk, W. Yang, J. Liang, Y.-W. Chao, Q. Wan, S. Birchfield, N. Ratliff, and D. Fox, "Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9164–9170.

[5] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox, "AnyTeleop: A General Vision-Based Dexterous Robot Arm-Hand Teleoperation System," in *2023 Robotics: Science and Systems (RSS)*, 2023.

[6] R. Hetrick, N. Amerson, B. Kim, E. Rosen, E. J. de Visser, and E. Phillips, "Comparing virtual reality interfaces for the teleoperation of robots," in *Systems and Information Engineering Design Symposium (SIEDS)*. IEEE, 2020.

[7] J. G. Wildenbeest, D. A. Abbink, C. J. Heemskerk, F. C. van der Helm, and H. Boessenkool, "The Impact of Haptic Feedback Quality on the Performance of Teleoperated Assembly Tasks," *Transactions on Haptics*, vol. 6, no. 2, pp. 242–252, 2013.

[8] J. Bimbo, C. Pacchierotti, M. Aggravi, N. Tsagarakis, and D. Prattichizzo, "Teleoperation in cluttered environments using wearable haptic feedback," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2017, pp. 3401–3408.

[9] D. Wei, B. Huang, and Q. Li, "Multi-view merging for robot teleoperation with virtual reality," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 4, pp. 8537–8544, 2021.

[10] F. Pereira, A. Dricot, J. Ascenso, and C. Brites, "Point cloud coding: A privileged view driven by a classification taxonomy," *Signal Processing: Image Communication*, vol. 85, p. 115862, 2020.

[11] J. Hu, A. Shaikh, A. Bahremand, and R. LiKamWa, "Characterizing Real-Time Dense Point Cloud Capture and Streaming on Mobile Devices," in *3rd ACM Workshop on Hot Topics in Video Analytics and Intelligent Edges*, ser. HotEdgeVideo '21. New York, NY, USA: Association for Computing Machinery, 2021.

[12] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai, "An overview of ongoing point cloud compression standardization activities: video-based (V-PCC) and geometry-based (G-PCC)," *APSIPA Transactions on Signal and Information Processing*, vol. 9, no. 1, 2020.

[13] P. Qian, V. S. H. Huynh, N. Wang, S. Anmulwar, D. Mi, and R. R. Tafazolli, "Remote Production for Live Holographic Teleportation Applications in 5G Networks," *IEEE Transactions on Broadcasting*, vol. 68, no. 2, pp. 451–463, 2022.

[14] Jeff Ford and Jordan Ford, "Lossless SIMD Compression of LiDAR Range and Attribute Scan Sequences," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

[15] F. Hu, Y. Deng, H. Zhou, T. H. Jung, C.-B. Chae, and A. H. Aghvami, "A Vision of an XR-Aided Teleoperation System toward 5G/B5G," *Communications Magazine*, vol. 59, no. 1, pp. 34–40, 2021.

[16] Y. Qiao, Q. Zheng, Y. Lin, Y. Fang, Y. Xu, and T. Zhao, "Haptic Communication: Toward 5G Tactile Internet," in *Cross Strait Radio Science & Wireless Technology Conference (CSRSWTC)*, 2020, pp. 1–3.

[17] X. Chen, L. Johannsmeier, H. Sadeghian, E. Shahriari, M. Danneberg, A. Nicklas, F. Wu, G. Fettweis, and S. Haddadin, "On the Communication Channel in Bilateral Teleoperation: An Experimental Study for Ethernet, WiFi, LTE and 5G," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2022, pp. 7712–7719.

[18] "WebRTC: Real-Time Communication in Browsers," https://www.w3.org/TR/webrtc/, 2023, [Online; accessed 2023-08-21].

[19] A. I. Mourikis and S. I. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," in *International Conference on Robotics and Automation (ICRA)*, 2007, pp. 3565–3572.

[20] A. Albu-Schaffer, C. Ott, U. Frese, and G. Hirzinger, "Cartesian impedance control of redundant robots: Recent results with the DLR light weight arms," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2003.

[21] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot Collisions: A Survey on Detection, Isolation, and Identification," *IEEE Transactions on Robotics (TRO)*, vol. 33, no. 6, pp. 1292–1312, 2017.

[22] "IPhone 11 - Technical Specifications," https://www.apple.com/by/iphone-11/specs/, 2023, [Online; accessed 2023-08-21].

[23] "Franka-Emika Robot Instruction Handbook," https://download.franka.de/documents/100010_Product%20Manual%20Franka%20Emika%20Robot_10.21_EN.pdf, 2023, [Online; accessed 2023-08-21].

[24] "NVIDIA Video Codec SDK," https://developer.nvidia.com/video-codec-sdk, 2023, [Online; accessed 2023-08-21].

[25] *powturbo*, "TurboPFor-Integer-Compression," https://github.com/powturbo/TurboPFor-Integer-Compression.git, 2023.

[26] P. Kim, J. Kim, M. Song, Y. Lee, M. Jung, and H.-G. Kim, "A Benchmark Comparison of Four Off-the-Shelf Proprietary Visual–Inertial Odometry Systems," *Sensors*, 2022.

[27] "libdatachannel - C/C++ WebRTC network library," https://github.com/paullouisageneau/libdatachannel, 2023, [Online; accessed 2023-08-21].