Remote Robot Control with Human-in-the-Loop Over Long Distances using Digital Twins

Ievgenii A. Tsokalo*, David Kuss*, Ievgen Kharabet*, Frank H.P. Fitzek*†, and Martin Reisslein‡
*Deutsche Telekom Chair of Communication Networks - Technische Universität Dresden, Germany
†Centre for Tactile Internet with Human-in-Loop (CeTI)
‡Electrical, Computer, and Energy Eng.; Arizona State University; Tempe, AZ 85287-57060, USA
Email: {ievgenii.tsokalo|ievgen.kharabet|frank.fitzek}@tu-dresden.de, david.kuss@mailbox.tu-dresden.de, reisslein@asu.edu

***Abstract*—The sharing of skills over the Internet enables professionals to democratize their expertise and skills without exhausting their availability, e.g., through excessive traveling. To enable this Internet of Skills, we present a novel Digital Twin (DT) platform for the remote control of machines with human-in-the-loop. The DT of a remotely controlled machine acts effectively as an interlayer between the operator and the controlled machine, e.g., robot arm. The DT can be optimized for a particular application to interact with the operator with an intuitive low-latency interface and, on other side, to control and monitor the quality of the remote task. Essentially, the human operator controls the DT, while the DT controls the remote robot. This paper introduces the DT framework for the remote control. The human-machine-human control loop is split into Virtual Reality (VR), remote control, and robot control loops. The proposed framework achieves low latency visual feedback and very short system reaction times for unexpected changes with arbitrary distances between operator and robot. Within the DT framework, this paper proposes a robot control algorithm for controlling time-critical robot applications over networks with considerable delays and jitter. The proposed framework has been implemented in a demonstrator with a robot arm and its DT in VR.**

***Index Terms*—Tactile Internet, edge cloud, low latency, remote control, closed-loop control, robot arm.**

I. Introduction

The expertise of doctors, engineers, teachers, and other highly qualified professionals is often not available outside of metropolises or in certain countries. The sharing of skills over the Internet can help to bring expertise to any place in the world. While the current Internet has democratized the access to information for any given time or location, the ongoing Tactile Internet research strives to democratize the access to skills and expertise on a global scale to promote equity for people of different genders, ages, cultural backgrounds, or physical limitations. An essential part of the Internet of Skills will be a system for remote control of machines with a human-in-the-loop, e.g., for teleoperation [1] and for handling emergency situations [2]. It is already possible to control robots for space exploration [3] and over large distances on Earth [4]. Despite existing implementations, the limits on the quality of service of the remote machine and the quality of experience of the operator are not well studied.

There are several remote control approaches, which include the use of an input device, such as a joystick or control pad, touch-free operation via gesture control, and the use of augmented or virtual reality (VR) to facilitate robot control and to make it more intuitive [5]–[7]. We selected VR as a flexible and feature-rich solution with fast visual feedback. VR allows for creating a digital twin (DT) of the controlled object. With the DT, the feedback is not affected by the remote link delay. The earlier idea of a physical robot twin solves the same problem but requires more investment [8]. The VR model also allows for customizing the view and the control interface so as to simplify the remote control. Although the VR model strives to accurately represent the real robot, there are always small differences between the VR model and the real robot that may result in slightly different operator reactions. A method to eliminate the errors caused by the differences between the model and real robot has been presented in [8]. Other steering inaccuracies can be caused by delayed feedback or human mistakes. There are several ways to compensate for steering errors. For example in [1], tactile feedback from the catheter tip inserted into a blood vessel is utilized to reduce the risk of damaging the blood vessel walls. Sensors and fuzzy logic have been used in [9] to track obstacles in the planned path of a mobile robot. It is also possible to account for the lag induced in the control loop by a long distance through the use of mathematical algorithms while operating a space exploration robot [3].

We suppose that an object detection system is available at the working site of the remote robot. The implementation of object detection in an Edge Cloud (EC) allows for achieving different latency and accuracy requirements through dynamic scaling of computing resources. The EC will be implemented by software defined networks (SDN) in combination with network function virtualization (NFV) [10], [11]. Beside accuracy as a quality estimator of remote control, there are other important factors. The human should have a convenient interface with small visual feedback to avoid nausea. The robots often have strict timing requirements that cannot be realized over the Internet. In order to meet the demands of application, human, and robot, the control loop of human and robot should be decoupled.

This paper develops a Digital Twin (DT) based framework for remote control consisting of three control loops as illustrated in Figs. 2 and 3. In particular, we split the human-machine-human control loop into three loops and define the interfaces between them. In the first loop (the "Digital Twin" loop in Fig. 3), the operator interacts with the DT to define the job for a remote robot. The second loop ("DT2R" loop in Fig. 3) serves for interaction between the DT and the Robot Control Application (RCA, also referred to as Robot Control Algorithm) over the Internet. The RCA controls the job quality and allows for decoupling the time-critical robot loop from the rest of the network. Each millisecond, the RCA creates transition points between the commands from a human. In the third loop ("Robot" loop in Fig. 3), the actual job is done involving the RCA and the robot. This paper examines the limits on the quality of service in a remote control system with a human-in-the-loop and presents results from a prototype

978-1-7281-0962-6/19/$31.00 ©2019 IEEE

implementation. Our "Digital twin" setup in Fig. 3 achieved a 70 ms control loop duration for the visual feedback. Since the images are processed in the EC in the proximity of the operator, this control period duration is 99% of the time independent of the actual distance between the operator and the robot. In the remaining 1% of time, the duration can be as high as 131 ms (as detailed in Sec. IV-D). The latency between the DT and the RCA is brought down to half of the round-trip ping time [12] by minimizing the amount of traffic. This latency should be very low since it is the system reaction time for unexpected changes. The DT can force the RCA to halt or correct the movement of the robot in case of an emergency. The operator is informed about such situations as well and overtakes the handling of the situation after the reaction of the DT.

II. Problem Description

The period of the control loop T_{cc} includes all delays on the path from the human operator through the controlled system and back to the human. The delays caused by human reaction are outside of the scope of this study. T_{cc} influences the reliability of the control application. We define real-time operation as the capability to guarantee a prescribed level of reliability of the control application, even in case of unexpected changes in the controlled system. Unexpected changes are changes that occur against the will of the remote operator and may or may not be caused by the actions of the remote operator. Generally, the reliability can only be guaranteed for a pre-defined set of unexpected changes, for which a required minimum reaction time can be derived. To summarize, the real-time capability is defined through the accuracy of the performed task and the overall minimum reaction time for unexpected changes. The real-time operation creates bounds for the network latency and jitter. We do not claim to design a system to guarantee real-time operation for a specific application. Instead, we demonstrate how to design a system with a small T_{cc} that facilitates typical applications. Besides the real-time requirements, T_{cc} is restricted by human and robot capabilities.

III. System Model

A. Use case

Remotely controlled robots can assist humans in a wide range of use tele-operation cases, including operation in environments that would endanger human lives. We focus on a generic remote control robot system. In particular, we examine a system with which a human operator remotely controls a robot arm to manipulate objects. The operator holds a wireless remote controller and steers a remote robot arm through hand movements. The end-point of the robot arm, i.e., the so-called gripper, should mimic the operator hand movements in 3D space.

We examine a series of three increasingly sophisticated setups, starting from an elementary single control loop setup shown in Fig. 1. The steering signal propagates from the human over the Internet and gateway to the robot. On the remote side, a web camera captures and streams a video of the robot arm via the gateway and Internet back to the human. The human uses the visual feedback to adapt the movements of the robot arm. We consider an elementary task for laboratory conditions: Wooden pieces on the table in front of the robot should be picked up and stacked to build a tower.

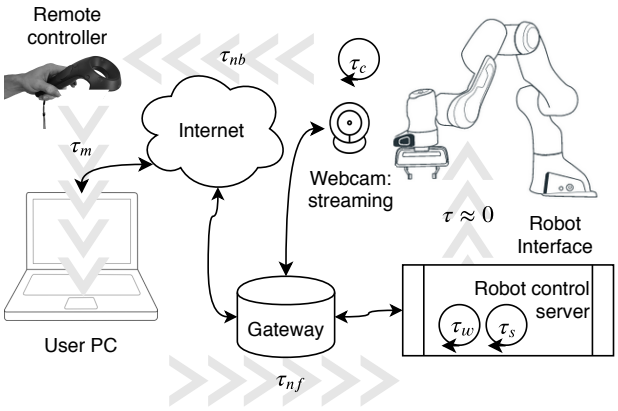

Fig. 1. Elementary single-loop setup: The human operator controls the remote robot based on video streamed from the remote robot site.

The extended setup in Fig. 2 should perform the same task, but with higher accuracy and better quality of operator experience. A similar system was proposed in [13], [14] for space exploration. In our use case, we exploit the knowledge of the environment where the robot is located to ease the work for the operator and to achieve a higher accuracy of the remote control.

The key performance metric in our evaluation is the control loop period T_{cc}. In the setup in Fig. 1, which is analyzed in Sec. III-B, there is only one control loop, which corresponds to the "Main" control loop in Fig. 3. In this Main loop, the operator hand movement is coupled with the robot movement. Such a single-loop setup is highly challenging due to the strict timing requirements of robot arms. The "Main" loop can be divided into the left "Remote control" loop and the right "Robot" loop in Fig. 3 in order to fulfill the timing requirements of the robot, as analyzed in Sec. III-C. We further divide the "Remote control" loop into the "Digital Twin" loop and the DT to Robot (DT2R) loop to reduce the visual feedback latency, as analyzed in Sec. III-D.

B. Elementary single-loop setup

In the single-loop setup in Fig. 1, the human moves the handle, whose position is tracked in space. The robot copies the movements of the handle. The web camera captures and streams the video of the robot site to the human so as to provide visual feedback. The period T_{cc} of the "Main" control loop (outer loop in Fig. 3) consists of several delays. First, an operator hand movement or the pressing the buttons needs to be translated into the corresponding robot command, incurring the delay τ_m. The command is sent via the Internet in a UDP data stream, incurring a network (forward) delay τ_{nf}. Upon arrival, the command is buffered at the robot until the beginning of the next robot control cycle, incurring the waiting delay τ_w. After the movement command is applied, the current velocity vector of the robot is adjusted. Due to the limited acceleration and jerk, it takes time τ_s to settle on the new velocity vector corresponding to the movement of the human. The robot movement is captured by the web camera that inserts an extra delay τ_c dependent on frames per second rate [15]. The real-time video of the robot is streamed back through the UDP socket. This stream has significantly higher data rate than the stream of commands, resulting in a longer network (backward) delay $\tau_{nb} > \tau_{nf}$. The control loop period can thus be calculated as:

$$T_{cc} = \tau_m + \tau_{nf} + \tau_w + \tau_s + \tau_c + \tau_{nb}. \quad (1)$$

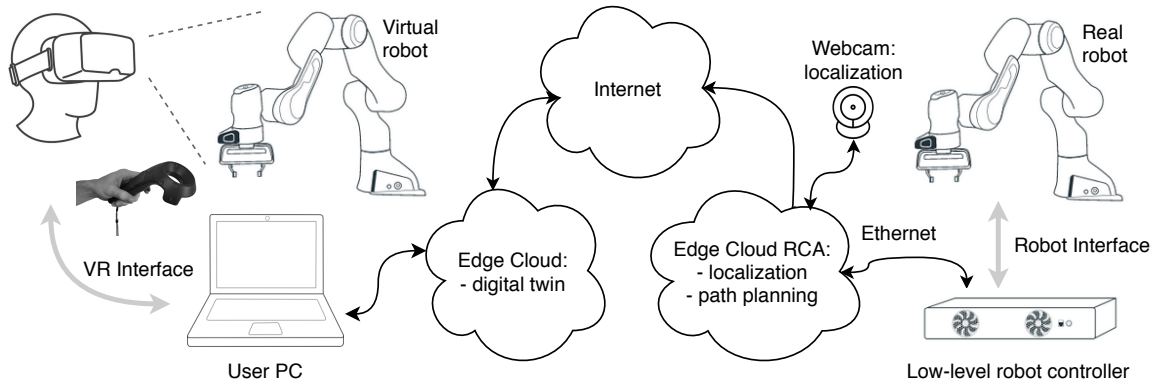

Fig. 2. Digital Twin (DT) setup with Virtual Reality (VR) display of DT computed in Edge Cloud (EC) near the human operator and robot control algorithm (RCA) executed in EC near the robot.

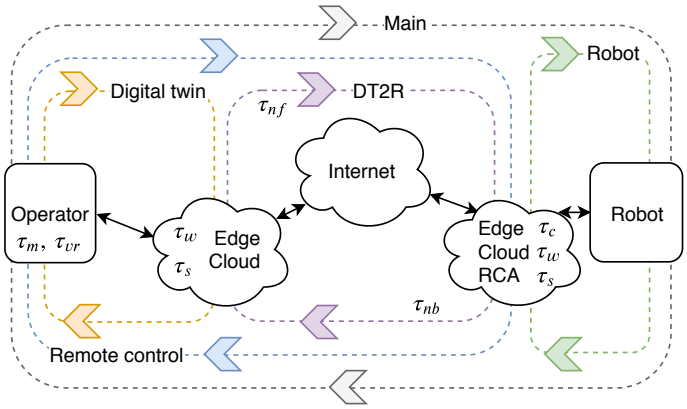

Fig. 3. Illustration of control loops: Single (Main) loop for elementary setup in Fig. 1; Two (Remote control and Robot) loop control; and Three (Digital twin, DT2R, and Robot) loop control for DT setup in Fig. 2.

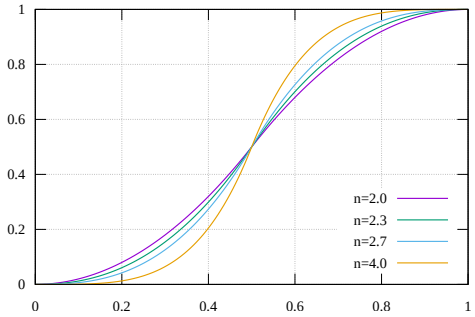

Fig. 4. Robot control increment function Eq. (2) for different shaping parameters n.

C. RCA: Enabling control loop decoupling

The used robot arm has 1 kHz control cycle frequency and does not tolerate jitter, nor systematic delay increases. Since the period T_{cc} (and its variations) of the elementary single-loop control is definitely greater than the required 1 ms, the remote steering of the robot arm is facilitated by decoupling the control loop of the robot from the human-robot-human control loop. We split the main loop in Fig. 3 into a robot control loop and a remote control loop, whereby the EC executes the robot control algorithm (RCA). The EC has interfaces to interact with both loops.

We designed the following RCA. During initialization, the algorithm obtains the physical limits of the robot arm, such as velocity, acceleration, jerk, and maximum/minimum translation/rotation of the robot gripper. At runtime, the algorithm obtains target points over the network with latency $\gg$ 1 ms and considerable jitter, as common in today's Internet [12]. The points consist of three translation and three rotation values that specify the gripper vector position in 3D space. These points are used to calculate the gripper transition points, which are applied each millisecond. The algorithm approximates the robot trajectory between the points received from the operator to produce the transition points corresponding to the 1 kHz robot control frequency. Also, the algorithm enforces the physical limits of the robot. The approximation is implemented in a way that relaxes the necessity to synchronize the robot control loop and the remote control loop.

The algorithm treats each translation and rotation as a separate dimension, i.e., there are six dimensions. The calculations below are separately applied for each dimension. The actual robot gripper position p_a is calculated from the initial gripper position p_0 and the position increment D as $p_a = p_0 + D$, whereby both p_0 and D change with time. The position increment D is calculated as $D = \sum_{i \in [0,\dots,I)} d_i \cdot s(\phi_i)$, where i denotes the index of buffered movement commands received from the DT. The distance d_i is calculated based on the latest target point of the remote operator p_t, the actual position p_a, and the current velocity of the robot (Algorithm 1, line 12). We design the robot control based on the principle "only the last target position matters". This principle means that the robot does not need to follow a specific trajectory; instead, the robot strives to reach the last target position p_t as fast as possible. The sum of all distances d_i gives the total distance that the robot will move in a given dimension if p_t does not change. The function $s(\phi_i)$ makes the movement gradual. The phase ϕ_i, $\phi_i \in [0, 1]$, is the equivalent of time. The component $d_i \cdot s(\phi_i)$ describes the correction of D when a new target point is received from the remote operator. When $\phi_i = 1$, the initial position p_0 is incremented by d_i and the movement command with index i is removed from the buffer. In our system, we define $s(\phi_i)$ using a polynomial function and extend it by the shaping parameter n that varies the robot acceleration according to the ternary operator

$$s(\phi_i, n) = (\phi_i < 0.5) \ ? \ y(\phi_i, n) : 1 - y(1 - \phi_i, n), \qquad (2)$$

with $y(\phi_i, n) = 2^{n-1} \cdot \phi_i^n$. Examples for different n are shown in Fig. 4. The selection of n depends on the experimentally observed robot dynamics. We use $n = 2$ for the Franka Emika robot arm.

Upon reception of p_t, Algorithm 1 adjusts the trajectory by selecting one of the following actions for each individual dimension: accelerate, stop as soon as possible, stop at a specified

point, or do nothing. This algorithm uses several definitions. The final stop point of the robot (if no further points from human are received) is calculated as $p_f = p_0 + \sum_{i\in[0,\dots,I)} d_i$. The direction of the movement b is equal to "1" when the robot moves in the positive direction in the given dimension, "-1" for the negative direction, and "0" not defined:

$$b = (p_f > p_a)\ ?\ 1\ :\ ((p_f < p_a)\ ?\ -1\ :\ 0). \tag{3}$$

The minimum stop distance according to the current acceleration and velocity of the robot is:

$$p_m = p_0 + \sum_{i\in[0,\dots,I)} \left[\phi_i > 0.5\ ?\ d_i\ :\ 2\cdot d_i \cdot s(\phi_i)\right], \tag{4}$$

which means that the robot needs at least the same time to decelerate as it has already accelerated.

Algorithm 1: Robot Control Algorithm (RCA): Calculation of transition point.

if $[b = 1 \cap\ p_t < p_m \cup b = -1 \cap p_t > p_m]$ **then**
  ; Stop as soon as possible
  $\forall i \in [0,\dots,I):\ \phi_i \leftarrow (\phi_i > 0.5)\ ?\ \phi_i\ :\ 1-\phi_i$
else if $[b = 1 \cap p_t < p_f \cup b = -1 \cap p_t > p_f]$ **then**
  ; Stop at specified point
  $\Delta d \leftarrow p_t - p_t^{\text{previous}}$; correction to distance
  $d_c \leftarrow d_i \cdot s(\phi_i)$; currently traversed distance
  $pr_i \leftarrow \min(1,\ [d_c + \Delta d]/[2\cdot d_i])$; current progress
  $\phi_i^D \leftarrow \min(0.5,\ s^{-1}(pr_i))$; deceleration phase
else if $|p_t - p_f| = 0$ **then**
  ; Do nothing
else
  ; Accelerate
  $I \leftarrow I+1;\ d_{I-1} \leftarrow (p_t - p_f)$
$\phi_i \leftarrow (\phi_i \geq \phi_i^D)\ ?\ 1-\phi_i : \phi$; start deceleration if necessary

Algorithm 1 updates the robot trajectory at any time when receiving a new target point from the human. The delaying of new points does not cause any violation of robot timings. The period of the robot control loop is $T_{cc}^R = \tau_w$. The period of the remote control loop can be calculated with Eq. (1), i.e., $T_{cc}^{RC} = T_{cc}$.

D. Remote Control with DT (RCDT): Three loop control

In order to reduce the visual feedback delay to the human operator, we introduce the Remote Control with the DT (RCDT) setup illustrated in Fig. 2. The human puts on a VR head-mounted display (HMD), which displays the DT of the real robot arm. The DT has the same size, mass, dynamic characteristics, and uses the same library for inverse kinematics (Denavit–Hartenberg transformation) as the real robot. Hence, the movement of the real robot corresponds to the movement of the DT when guided by the human. We also assume knowledge of the environment around the robot. In dynamic environments, the photo-realistic virtual reality can be achieved through photo-geometric imagery [16].

In the RCDT setup, the human controls the DT, while the DT controls the real robot. Since the VR software can be executed in close proximity of the operator, e.g., in the EC, the visual feedback to the human is much faster than in the simplistic one-loop system. We refer to the interconnection between the EC near operator and the EC near the robot as the RCDT interface. The RCDT interface synchronizes the real robot and the DT in the DT2R control loop (Fig. 3). The synchronicity can be ensured within the control loop period T_{cc}^{DT2R}. The commands received by the EC RCA are applied after ensuring the synchronicity. In case the DT and the real robot become asynchronous, the RCDT cancels any new commands from the human that were issued after the moment of loosing synchronicity.

The period of the "Digital twin" control loop is:

$$T_{cc}^{DT} = \tau_m + \tau_w + \tau_s + \tau_{vr}, \tag{5}$$

where τ_{vr} is the time to apply the next position of the DT and to play the corresponding video in the VR display.

The period of the "DT2R" control loop is $T_{cc}^{DT2R} = 2\tau_{nf}$. Since no video stream is sent back in the RCDT setup, the network feedback delay τ_{nf} is substantially reduced (assuming a symmetric Internet link, for simplicity). Basically, T_{cc}^{DT2R} is limited only by the speed of light. If the human and the robot are located closer than 150 km, then $T_{cc}^{DT2R} < 1$ ms.

The RCDT reduces the visual feedback delay T_H experienced by the human operator. As long as the virtual and real worlds remain synchronous, a human experiences only the delay of the VR (case I). In case, the movement of the real robot is not synchronous to the DT, the RCDT can react at the latest after τ_w (case II). In case the real environment changes, the RCDT reacts at the latest after τ_c (case III). T_H for these cases is calculated as:

$$T_H = \begin{cases} T_{cc}^{DT} & \text{(I)};\\ T_{cc}^{DT} + T_{cc}^{DT2R} + \tau_w & \text{(II)};\\ T_{cc}^{DT} + \tau_{nf} + \tau_c & \text{(III)}. \end{cases} \tag{6}$$

E. Enhancing setup with the localization system support

The accuracy of robot operation can be improved with partial automation of the robot control. Different phases of the remote control require different movement accuracy levels. Object interactions require generally high accuracy levels; while other phases can tolerate low accuracy levels. We use a web camera with high definition (HD) resolution and 30 frames/s to detect the position of the wooden pieces in front of the robot. This visual information is used for two purposes. First, the digital models of the wooden pieces are also displayed in front of the DT of the robot arm. Second, they are used to simplify the gripping procedure. Whenever the robot hovers in the area with a certain radius from the middle of the wooden cube and the height of the robot grip above the table is below 15 cm, the robot arm automatically centers above the middle of the wooden cube and only the movement along the vertical axis is allowed (out of six movement dimensions of the robot grip). This enables the precise gripping of the wooden piece. Whenever the human steers the robot higher than 15 cm above the table, then the control over all six dimensions is regained. With such a system design, the user can still select the pieces to be picked. But the precision is no more dependent on the T_{cc}^{DT} or T_{cc}^{DT2R}. Rather, the precision depends only on the localization system and the hardware in the robot control loop. Using multi-thread localization algorithms, the accuracy of the localization system can be improved by scaling the computational resources for the localization service in the EC. The accuracy of such a system is very high and mostly depends on the robot hardware manufacturer rather than on the network design. The accuracy can be impaired in case of unexpected

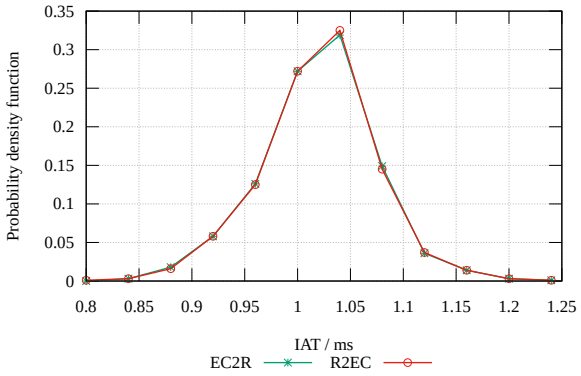

Fig. 5. Probability density function of Inter Arrival Time (IAT) of robot control packets.

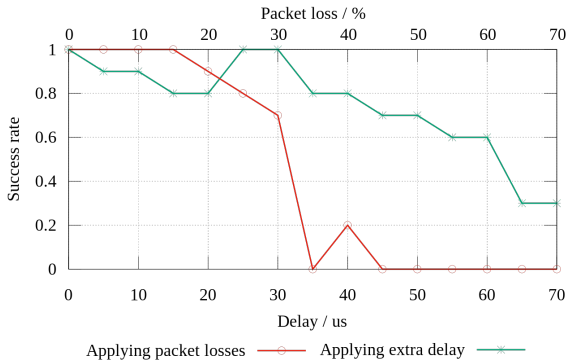

Fig. 6. Success rate of experimental runs as functions of packet loss and delay.

events in the real-time operation. Then, T_{cc}^{DT} and T_{cc}^{DT2R} will also impact the accuracy since the human has to react.

IV. Numerical Evaluation

A. Robot control loop requirements

We used the Franka Emika robot arm with seven joints. The controlling PC is connected to the robot over Ethernet. The PC runs a real-time Linux system. We first studied the timing requirements of the Franka Emika robot. In accordance with the manufacturer specifications [17], the control cycle has a duration of 1 ms that includes the program execution of the controlling PC, the communication, and the robot movement. It is also specified that 20 consecutively lost or dropped packets lead to an error and will cause a movement stop. We verified these specifications by artificially impairing the communication link with the Linux $netem$ network simulation function and tc traffic control. We used $wireshark$ for traffic probing and analysis. First, we measured the Inter-Arrival Time (IAT) of the packets received by the EC from the robot (R2EC direction) and of packets in the EC to robot (EC2R) direction. Fig. 5 shows the IAT probability density function, which is centered around 1 ms. We note that the motion command in each robot control cycle is created with the knowledge of the actual duration of the previous robot control cycle. Therefore, the jitter observed in Fig. 5 can be compensated for during the creation of the next motion command.

In a second measurement, we introduced systematic delays and random packet losses on the EC2R link while the robot performed a simple movement example of the libfranka C++ library. Each time when the robot completed the movement, the experimental run was marked as successful. Fig. 6 shows the success rate for a range of prescribed delays and packet losses. For each scenario (for a particular delay or packet loss), we conducted ten independent runs. Clearly, the robot does not tolerate systematic delay shifts. This is because the delay shifts cannot not be compensated during the creation of the motion commands, because the packets with already created commands are also delayed. Even a five or ten microseconds of extra delay caused 10% of the runs to be unsuccessful. For an added delay of 65 μs, the success rate dropped to 30%. Also, Fig. 6 indicates that packet losses up to 15% can be tolerated.

B. Control loop period in elementary single-loop setup

This section evaluates the period T_{cc} for the simplistic single-loop setup (Fig. 1). We used the VR HTC Vive remote controller, whose position is tracked by two base stations. The position of the remote controller in the 3D space is read out with a period of 20 ms, while the coordinate system translation and the robot command formation incur negligible delays; hence, $\tau_m = 20$ ms. At the CES and CCNC 2019 events in Las Vegas, NV, U.S., we steered a robot arm located in Dresden, Germany [4]. The sending delay of the small command messages (about 100 bytes) was approximately half of the round-trip ping time of 170 ms, i.e., $\tau_{nf} = 85$ ms for the considered Las Vegas, NV, US, to Dresden, Germany, example scenario. The Franka Emika robot arm has a 1 kHz control cycle. The command can be processed and applied to the robot within 1 ms, i.e. $\tau_w \leq 1$ ms. The video from the webcam uses the MJPEG codec with 640×480 pixel resolution and 25 frames/s. With the considered 640×480 pixel video resolution, the operator can reliably distinguish robot movements of about 1 cm. It will take the longest to move 1 cm from the stale state. The maximum translation acceleration of the robot arm is 13 m/s^2 [18]. Thus, we estimate $\tau_s = (0.01\,\text{m} / 13\,\text{m/s}^2)^{0.5} \approx 28$ ms.

The delay for sending the visual feedback to the human is influenced by queuing on all intermediate network devices. Since the video stream transfers much more data than the stream of commands from the human, the video stream is more affected by queuing. The measured delay was about 10 times greater than τ_{nf}, i.e., $\tau_{nb} = 800$ ms. The control loop duration is thus $T_{cc} = 20 + 85 + 1 + 28 + 800 = 934$ ms. With the maximum translation velocity $v = 1.7$ m/s, the length of the erroneous movement d is up to $1.7 \cdot 0.934\,\text{m} = 1.59\,\text{m}$! This tremendous inaccuracy will be reduced by slowing down the robot. Yet still, with a 100 times lower speed, the accuracy will be up to around 1.6 cm. Clearly, the elementary one-loop setup is not usable for any 5G application.

C. DT control loop

The control loop period T_{cc}^{DT} in the three-loop control is strongly influenced by the employed hardware. We employed a PC with CPU Intel® Core™ i7-8750H 2.20GHz, GPU GeForce GTX 1060 (6 GB dedicated RAM), and 16ĠB RAM for the EC computing the VR. The VR application employed Unity 2018.3.0f2 with the SteamVR Plugin ver. 2.2.0.

The component τ_{vr} of T_{cc}^{DT} addresses the capability of the hardware to display the VR video. The most demanding part is usually the graphics rendering. Fig. 7 shows a sample screen shot of the VR environment. Fig. 8 shows the cumulative distribution function of the VR frame processing time, basically τ_{vr}, for computing the VR environment display. Clearly, the warehouse object, which consists of numerous polygons, requires the most computational power. For increasing the achievable VR frame rate, the overall quality of the displayed

Fig. 7. Screen shot of VR display of DT.

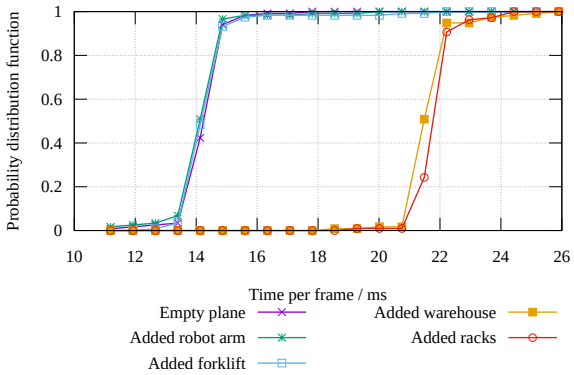

Fig. 8. Cumulative distribution function of VR processing (computing) time in EC.

geometries should be reduced (drawn with only a moderate number of polygons) wherever possible to achieve short VR frame periods. The VR control loop period is evaluated from Eq. (5) as $T_{cc}^{DT} = 20 + 1 + 28 + 21 = 70$ ms. A 5G connected remote controller could possibly decrease the τ_m component of T_{cc}^{DT}.

D. Visual feedback delay

For a distance of 150 km, corresponding to a $\tau_p = 1$ ms between the two ECs in Fig. 2, and 25 frames/s ($\tau_c = 40$ ms) of the object detection webcamThe delay T_H experienced by the human operator evaluated from Eq. (6) as 70 ms for case (I), $70 + 2 + 1 = 72$ ms for case (II), and $70 + 1 + 40 = 131$ ms for case (III). In our demonstrator, $T_H = 70$ ms for 99% of the time.

V. Conclusion

We have derived the key requirements of a remote robot control 5G use case, whereby we have closely considered the accuracy and real-time reaction time requirements. We have introduced a Remote Control with Digital Twin (RCDT) system consisting of three control loops to achieve the accuracy and real-time requirements. The RCTD system allows humans to remotely control robots over long distances. The accurate real-time remote robot control with a human-in-the-loop is made possible through the separation of control cycles. The robot normally has the most strict timing requirements that cannot be fulfilled by the remote control link. The proposed RCTD system decouples the timing requirements of the robot and the human.

The timely feedback for the human improves the work accuracy and prevents nausea for humans. In the proposed setup, the loop with the visual feedback is separated from the remote communication link using virtual reality. The delay experienced by the human operator (70 ms with our hardware) is mainly (for 99% of the time in our evaluations) independent of the network between the operator and the robot.

Acknowledgment

Funded by the German Research Foundation (DFG, Deutsche Forschungsgemeinschaft) as part of Germany's Excellence Strategy – EXC 2050/1 – Project ID 390696704 – Cluster of Excellence "Centre for Tactile Internet with Human-in-the-Loop" (CeTI) of Technische Universität Dresden. Additionally, this work has been supported by the German Federal Ministry for Education and Research (FastRobotics, FKZ: 03ZZ0528E and TacNet, FKZ: 16KIS0736), and Ericsson GmbH. The authors alone are responsible for the content of the paper.

References

[1] T. Wang, D. Zhang, and L. Da, "Remote-controlled vascular interventional surgery robot," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 6, pp. 194–201, June 2010.

[2] H. Yoshinada, K. Kurashiki, D. Kondo, K. Nagatani *et al.*, *Dual-Arm Construction Robot with Remote-Control Function*, ser. Disaster Robotics, S. Tadokoro, Ed. Cham: Springer, 21 January 2019, vol. 128.

[3] F. Kulakov, B. Sokolov, A. Shalyto, and G. Alferov, "Robot master slave and supervisory control with large time delays of control signals and feedback," *Appl. Mathem. Sciences*, vol. 10, no. 36, pp. 1783–1796, 2016.

[4] I. A. Tsokalo, H. Wu, G. T. Nguyen, H. Salah, and F. H. Fitzek, "Mobile edge cloud for robot control services in industry automation," in *Proc. IEEE CCNC*, Las Vegas, NV, Jan. 2019.

[5] T. Grzejszczak, M. Mikulski, T. Szkodny, and K. Jedrasiak, "Gesture based robot control," in *Proc. ICCVG*, 2012, pp. 407–413.

[6] S. Hashimoto, A. Ishida, M. Inami, and T. Igarashi, "Touchme: An augmented reality based remote robot manipulation," in *Proc. ICAT*, Osaka Univ., Japan, November 28-30 2011.

[7] A. Kelly, N. Chan, H. Herman, D. Huber *et al.*, "Real-time photorealistic virtualized reality interface for remote mobile robot control," *The International Journal of Robotics Research*, vol. 30, pp. 384–404, 2011.

[8] F. Kulakov, G. Alferov, and P. Efimova, "Methods of remote control over space robots," in *Proc. Int. Conf. on Mech.*, St. Petersburg, Russia, 2015, pp. 1–6.

[9] H.-J. Yeo and M.-H. Sung, "Fuzzy control for the obstacle avoidance of remote control mobile robot," *Journal of the Institute of Electronics Engineers of Korea SC*, vol. 48, pp. 47–54, 25 Jan. 2011.

[10] P. J. Braun, S. Pandi, R. Schmoll, and F. H. P. Fitzek, "On the study and deployment of mobile edge cloud for tactile Internet using a 5G gaming application," in *Proc. IEEE CCNC*, Jan. 2017, pp. 154–159.

[11] Z. Xiang, F. Gabriel, E. Urbano, G. T. Nguyen, M. Reisslein, and F. H. Fitzek, "Reducing latency in virtual machines: Enabling tactile internet for human-machine co-working," *IEEE JSAC*, vol. 37, no. 5, pp. 1098–1116, 2019.

[12] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury, "Ultra-low latency (ULL) networks: The IEEE TSN and IETF DetNet standards and related 5G ULL research," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 1, pp. 88–145, 2018.

[13] J. Funda and R. P. Paul, "Efficient control of a robotic system for time-delayed environments," in *Proc. Int. Conf. on Adv. Robotics Robots in Unstructured Environments*, June 1991, pp. 219–224 vol.1.

[14] X. Xu, B. Cizmeci, C. Schuwerk, and E. Steinbach, "Model-mediated teleoperation: Toward stable and transparent teleoperation systems," *IEEE Access*, vol. 4, pp. 425–449, 2016.

[15] C. Bachhuber, E. Steinbach, M. Freundl, and M. Reisslein, "On the minimization of glass-to-glass and glass-to-algorithm delay in video communication," *IEEE TMM*, vol. 20, no. 1, pp. 238–252, 2017.

[16] A. Kelly, E. Capstick, D. Huber, H. Herman, P. Rander, and R. Warner, "Real-time photorealistic virtualized reality interface for remote mobile robot control," in *Robotics Research*, C. Pradalier, R. Siegwart, and G. Hirzinger, Eds. Berlin, Heidelberg: Springer, 2011, pp. 211–226.

[17] "Requirements on communication interface for panda research robot," Last visited: May 2019, https://frankaemika.github.io/docs/requirements.html.

[18] "Limits on movement dynamics of panda research robot," Last visited: May 2019, https://frankaemika.github.io/docs/controlparameters.html.