Chapter 10
Investigation of the Use of Commercial Robotic Arms for Real-Time Hybrid Substructuring

Arian Kist, David Stadler, Rok Belšak, Vasja Plesec, Timi Karner, Gregor Harih and Daniel Rixen

Abstract Real-Time Hybrid Substructuring (RTHS) allows components of dynamic systems to be tested under realistic boundary conditions from an early stage of development by coupling the prototype of the component in real-time with a co-simulation of the remaining structure of the dynamic system. Real-time synchronization of the interfaces using actuators and sensors is a prerequisite for stable experiments and high-fidelity results. Since these hardware components suffer from imperfect transfer behavior and delays, additional controllers are required in most RTHS setups. This paper presents an RTHS framework specifically designed to use off-the-shelf robotic arms as actuators. A combination of Iterative Leaning Control and Normalized Passivity Control acts as a pure outer-loop controller in the robot's task-space, making it independent of the robot used and avoiding integration into the robot's low-level controllers. We demonstrate the ability of the control framework to provide high-fidelity results in virtual, i.e. fully simulated, RTHS experiments. We also present a first experimental realization of the framework using a KUKA® KR16 robot arm as an actuator. Although a stable RTHS experiment could be performed with this hardware, the fidelity improvement through iterative learning could not be experimentally validated yet.

Keywords Real-time hybrid substructuring · Robotic arm · Iterative learning control · Passivity control · Delay compensation

Introduction

Dynamic Substructuring is a well-established method in the field of structural dynamics [1]. In this approach, the dynamics of an overall system is examined by independently analyzing the dynamic behavior of several substructures that compose the overall system. This can be done both experimentally and numerically. The dynamics of the overall system is then emulated by coupling the individual subcomponents. To do this, the *compatibility* of the displacements and the *equilibrium* of the forces at each of the interfaces between the substructures must be satisfied. This concept is visualized in fig. 1a. The overall system is decomposed into two substructures (A and B). To preserve the overall dynamics when investigating the dynamics of the substructures individually, the compatibility condition requires the interface displacements to be equal, i.e. $\boldsymbol{z}^A = \boldsymbol{z}^B$, and the equilibrium condition requires the resulting interface force to be zero, i.e. $\boldsymbol{F}^A + \boldsymbol{F}^B = \boldsymbol{0}$. Based on these conditions, the dynamics of the whole system can be reconstructed from the individual results of the substructures.

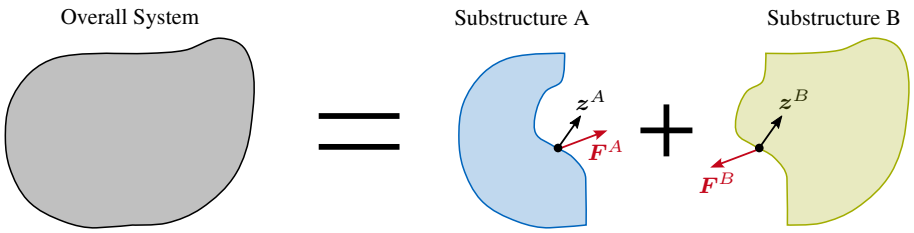
Real-Time Hybrid Substructuring (RTHS) is an efficient and economical testing approach to investigate the dynamic behavior of systems under realistic conditions by coupling a numerical co-simulation to physical components in real-time [2, 3, 4]. The approach therefore makes use of the concept of Dynamic Substructuring and the system under test is

Arian Kist · David Stadler · Daniel Rixen
Chair of Applied Mechanics, TUM School of Engineering and Design, Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Boltzmannstr. 15, 85748 Garching, Germany
e-mail: arian.kist@tum.de; david.stadler@tum.de; rixen@tum.de

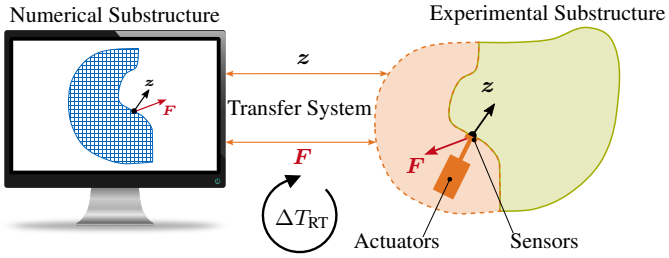
Rok Belšak · Timi Karner
Laboratory for Robotics, Faculty of Mechanical Engineering, University of Maribor, Smetanova Ulica 17, 2000 Maribor, Slovenia
e-mail: rok.belsak@um.si; timi.karner@um.si

Vasja Plesec · Gregor Harih
Laboratory for Integrated Product Development and CAD, Faculty of Mechanical Engineering, University of Maribor, Smetanova Ulica 17, 2000 Maribor, Slovenia
e-mail: vasja.plesec@um.si; gregor.harih@um.si

Walter D'Ambrogio, et al. (eds.), *Dynamic Substructuring & Transfer Path Analysis, Vol. 4*, Conference Proceedings of the Society for Experimental Mechanics Series, https://doi.org/10.13052/97887-438-0149-8_10



(a) Dynamic Substructuring.



(b) Real-Time Hybrid Substructuring.

Fig. 1 Illustration of Real-Time Hybrid Substructuring (b) in the context of Dynamic Substructuring (a). In order to couple the dynamics of substructures A and B to satisfy the dynamics of the overall system, the compatibility of the interface displacements ($\boldsymbol{z}^A = \boldsymbol{z}^B$) and the equilibrium of the interface forces ($\boldsymbol{F}^A + \boldsymbol{F}^B = \boldsymbol{0}$) must hold. In Real-Time Hybrid Substructuring, one substructure is simulated numerically and is therefore referred to as the Numerical Substructure. This is usually the well-defined part of the overall system, i.e. it can be modeled with high fidelity by numerical methods. For the part of the system where simulation is not feasible, a physical component is realized and investigated experimentally. This part is called the Experimental Substructure. To emulate the dynamics of the overall system, both substructures are coupled in real-time (denoted by ΔT_{RT} in the figure). To synchronize the interface motion $\boldsymbol{z}$ and the interface force $\boldsymbol{F}$ and thus to satisfy the compatibility and equilibrium conditions, a Transfer System is used. The Transfer System aims to mimic the interface to the numerical substructure using actuators and sensors.

divided into different substructures. An illustration of the RTHS approach is given in fig. 1b. The specific challenge of RTHS is that it involves both numerically simulated as well as experimentally investigated substructures, i.e. physical components, and that the coupling of these substructures is performed in real-time. A Transfer System aims at satisfying the compatibility and equilibrium condition at the interfaces between the numerical and experimental substructures under real-time conditions. This requires the use of sensors and actuators to measure and apply interface motions and forces. Perfect emulation of the reference system is only possible if these hardware components do not introduce synchronization errors, i.e., violate the compatibility and equilibrium condition. In addition, imperfect interface synchronization can even make the test unstable and damage the test setup. The stability issues of RTHS experiments due to interface synchronization errors and delays are discussed, e.g., in [5, 6, 7, 8]. Therefore, special attention must be paid to the selection of the hardware components involved and control strategies must be developed to mitigate synchronization errors and ensure stability. Over the last decades intensive research has been put in designing appropriate control approaches, among many others, ranging from simple polynomial extrapolation [5] over model based controllers [9] to learning [10, 11] and passivity-based [12, 13] approaches. An overview can be found, e.g., in [14].

In this contribution, we share our experience in using a commercial robotic arm, a KUKA® KR16 [15] with a KR C4 robot controller [16], as an actuator to synchronize the interface between a Numerical and an Experimental Substructure in an RTHS test setup. In our previous work, see e.g. [17, 18, 19], we used as an actuator a Stewart platform built in house. This Stewart platform allowed direct access to the joint motors, minimizing potential communication delays. It also allowed the design of advanced joint controllers to reduce tracking errors and compensate for actuator dynamics. When considering industrial RTHS applications, the use of application-specific actuators may not always be an economically efficient approach. Since industrial robot arms are well established in many industries, their use as actuators for RTHS may be the more feasible option in many cases. In addition, as shown in [19], the capabilities of our Stewart platform are limited in terms of workspace and performing highly dynamic motions, which could be overcome by using an industrial robot arm. Therefore, we compare the performance of the KUKA® KR16 [15] in a simple RTHS experiment with the performance of our Stewart platform.

Since the user typically does not have access to the low-level control of individual joints on industrial robots, we adapted our learning-based joint-space control scheme [18] to directly adjust the desired task-space trajectory to mitigate synchronization errors. We demonstrate the performance and robustness of this adapted control scheme through simulation, i.e., in a virtual RTHS experiment, by treating the controlled actuator as a black box and inducing different amounts of artificial delay.

RTHS Test Setup

For the investigations of this contribution, we analyze the system shown in fig. 2 using RTHS. It consists of a one-dimensional two-mass oscillator including a contact scenario. The upper mass m_{NUM} is connected to a moving point by a spring with stiffness k_{NUM} and a damper with damping ratio d_{NUM}. The lower mass m_{EXP} is attached to the upper mass by a spring with stiffness k_{EXP}. To excite the system, a cosine trajectory $z_{\mathrm{ex}}(t)$ with frequency f_{ex} is prescribed for the moving point so that the lower mass intermittently contacts the ground. For the investigation using RTHS, the moving point, the upper spring-damper and the upper mass are chosen as the Numerical Substructure, while the lower spring, the lower mass and the ground contact are chosen as the Experimental Substructure. The interface displacement is denoted as $z(t)$. A detailed description of this system has been given in our previous work, e.g.[20, 10, 13, 18], and further description is omitted here. The simplicity of the system allows us to compute a reference solution to assess the fidelity of the RTHS experiments, which is generally not possible for RTHS experiments.

The fig. 3 illustrates the signal flow of the entire RTHS loop executed in real-time. It includes the control scheme to ensure a robust and high-fidelity experiment, which we adapted from our previous work [18] to serve as a pure outer-loop control scheme. The numerical time-integration of the Numerical Substructure provides as output for each hybrid simulation step the interface displacement z (and its corresponding velocity $\dot{z}$). This serves as the desired task-space motion command for the robot to synchronize the interface displacements and satisfy the compatibility condition. The Experimental Substructure is mounted on the interface to the robot's end-effector. In this work, we investigate the use of a commercial robot as an actuator, so we assume that the low-level control of this robot is not accessible and the controlled actuator is treated as a black box. Possible processing and communication delays are accounted for using delay blocks as shown in fig. 3. The robot executes the motion z', $\dot{z}'$ and moves the Experimental Substructure. This motion is not the same as the commanded motion due to the preprocessing delay block (which accounts for delays in the robot for receiving and processing the motion command) and imperfect trajectory tracking of the low-level actuator controller. For our outer-loop control, a measurement of z' is required. Since acquiring this measurement may introduce additional delay compared to the actual motion of the robot, we introduce a postprocessing delay block that accounts for delays due to acquiring, processing, and transmitting the actual task-space position of the robot. The measured task-space motion of the robot is denoted as z'_{meas}, and it is generally assumed that $z \neq z' \neq z'_{\mathrm{meas}}$.

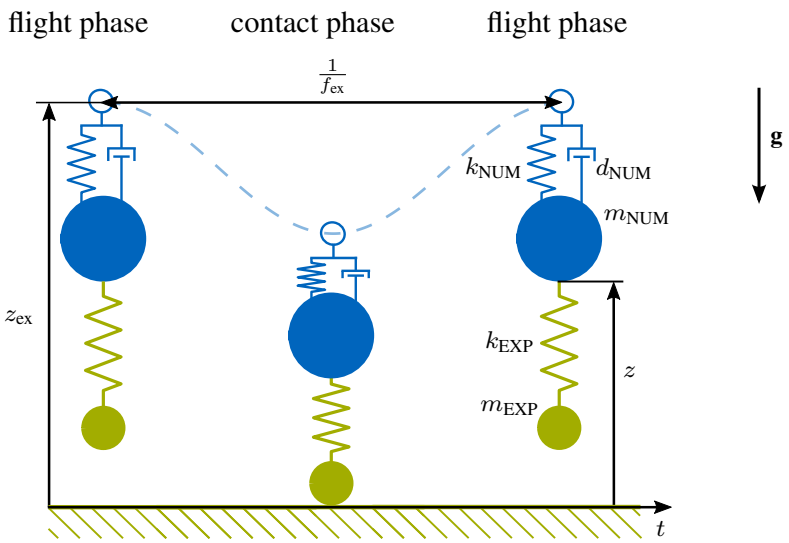


Fig. 2 Overall system whose dynamics are investigated with RTHS in this work. The Numerical Substructure is colored in blue, while the Experimental Substructure is colored in green. Figure adapted from [21].

In order to drive the total measurable tracking error of the robot $e_{\text{track}} = z - z'_{\text{meas}}$ to zero, we implement an Iterative Learning Controller (ILC). In [10, 18] we used the ILC as a feedforward controller within the joint controller of the Stewart platform. In this work, we use the ILC scheme to directly adjust the task-space motion command z to best compensate e_{track}. ILC is used when a motion command is executed several times in a row. It stores and processes the tracking error from the current trial j to improve the tracking error in the next trial $j+1$ by learning to generate a control signal Δ_{ILC} to compensate for the plant dynamics. The updating law for this control signal over the trials is:

$$\Delta_{\text{ILC},j+1} = Q \cdot (L \cdot e_{\text{track},j} + \Delta_{\text{ILC},j}) \tag{1}$$

Note that $e_{\text{track},j}(t)$ and $\Delta_{\text{ILC},j}(t)$ are the full time signals during the whole motion task of trial j, i.e. $t \in \left[0, \frac{1}{f_{\text{ex}}}\right]$. The learning function L is used to process the error signal of trial j. To ensure the robustness and convergence of the ILC, the entire signal is processed with a robustness filter Q. In this work, the learning function is chosen as a P-controller, i.e.: $L = L_{\text{p}}$ and the robustness filter is chosen as a zero-phase lowpass filter with a cutoff frequency $f_{\text{Q,cut}}$. Our implementation of the ILC is a purely data-based approach that does not require modeling of the system. However, the parameters L_{p} and $f_{\text{Q,cut}}$ must be tuned to achieve stable convergence of the learning. A further discussion of our ILC implementation and its convergence analysis is omitted here and the reader is referred to our previous publications [10, 18]. The adjusted task-space motion command for the robot is thus $z_{\text{ad}} = z + \Delta_{\text{ILC}}$.

To enforce the equilibrium condition of the interface forces, the force F resulting from the robot's motion z' is measured at the interface of the Experimental Substructure using a force-torque sensor (FTS). The measured force F', which is generally not equal to the actual interface force due to measurement errors in the FTS, is fed back to the Numerical Substructure. Note that we assume that this signal is not affected by the postprocessing delays in the robot, since we assume to use an additional FTS and do not rely on internal force/torque measurements in the robot.

To ensure stable RTHS experiments, a Normalized Passivity Controller (NPC) is implemented[1]. The NPC monitors the power flow between the substructures and the Transfer System. If the power at the interface to the Experimental Substructure $P_{\text{EXP}} = \dot{z}'_{\text{meas}} \cdot F'$ [2] is greater than the power at the interface to the Numerical Substructure $P_{\text{NUM}} = \dot{z} \cdot F'_{\text{ad}}$, i.e. $P_{\text{error}} = P_{\text{EXP}} - P_{\text{NUM}} \geq 0$, the experiment is considered active and therefore unstable. In this case, artificial damping force F_{d} is added to the force feedback to restore the passivity of the experiment. It is calculated as follows:

$$F_{\text{d}} = \begin{cases} G_{\text{d}} \cdot \frac{\tilde{P}_{\text{error}}}{|\tilde{P}_{\text{tot}}|} \cdot \dot{z} & \text{if } \tilde{P}_{\text{error}} > 0 \\ 0 & \text{if } \tilde{P}_{\text{error}} \leq 0 \end{cases} \tag{2}$$

Where $P_{\text{tot}} = P_{\text{EXP}} + P_{\text{NUM}}$, i.e. the total power, is used to normalize the error power P_{error} for the calculation of the damping gain. G_{d} is a tuning parameter. Note also that $\tilde{\bullet}$ indicates filtering with a first order lowpass filter with time constant T_{LP} to smooth the signals. Since the NPC already acts on the outer control loop and not within the robot's low-level controllers, we have not modified our implementation of [13, 18] and it is referred to that publication for further information on our NPC scheme. The adapted measured interface force $F'_{\text{ad}} = F' + F_{\text{d}}$ is then finally used in the next time step of the numerical simulation.

Compared to our previous implementation of the scheme, the adapted control scheme presented here has the advantage of acting completely as an outer-loop controller. No interaction nor integration with internal computations in the hardware components, such as low-level actuator controllers, is required. The only requirements for the used robot/actuator are that its desired task-space motion can be commanded by an external device and that a measurement of the actual task-space motion is available. Apart from these requirements, with this control scheme the hardware components for the robot and the FTS are interchangeable. However, if different hardware components are used, the control parameters of the ILC and NPC may need to be adjusted, but a model of either of these components is not required. In addition, it should be noted that the ILC now aims to drive the total delay induced by the robot/actuator used to zero, thus including processing and communication delays and not just improving low-level actuator tracking performance.

[1] In our control approach, ILC aims to improve the test fidelity and NPC ensures stability [18]. Even though a high fidelity RTHS experiment is less likely to become unstable, since ILC reduces the tracking error iteratively, in the first iterations the error is usually still large and additional approaches are needed to ensure stability. In our case, this is addressed by the NPC. On the other hand, NPC alone cannot improve the fidelity of the test, as it only prevents instability. Hence the combination of NPC and ILC.

[2] The actual power at the interface to the Experimental Substructure would require using the actual interface force F. However, in a real experimental setup, only the measured force F' is available, and we use this force to calculate the power at this interface.

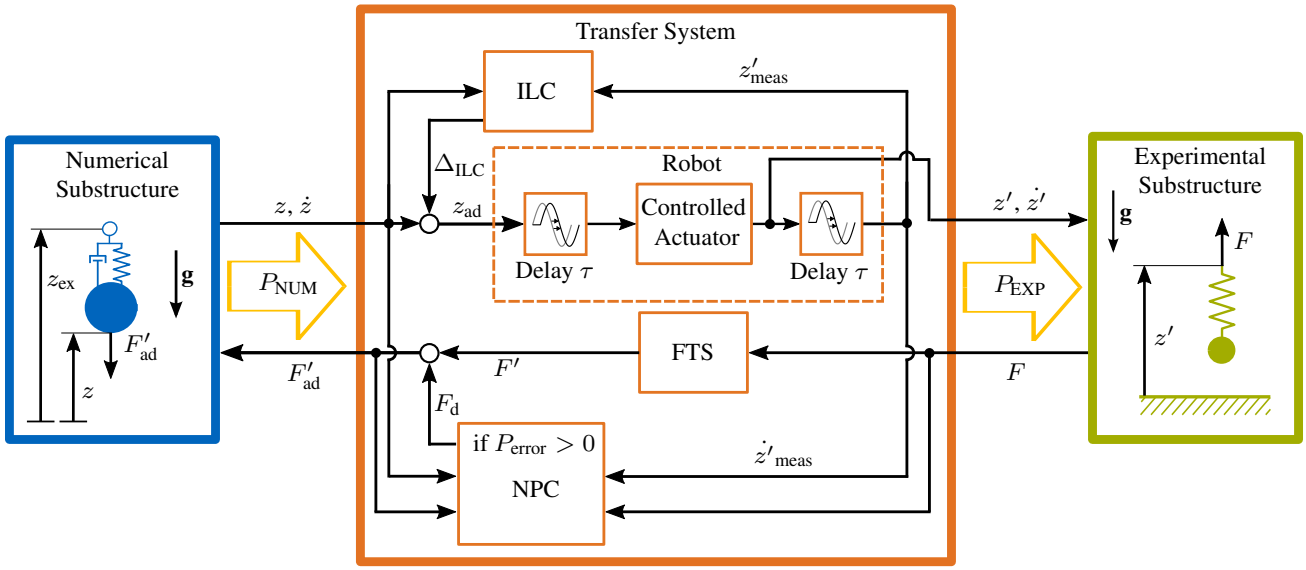


Fig. 3 Signal flow of the whole RTHS loop used in this work. It includes the control scheme using a combination of ILC and NPC to ensure robust and high-fidelity RTHS experiments, which we adapted from our previous work outlined in [18] to act as a pure outer-loop controller. Figure adapted from [21].

Simulative Investigation

To validate the performance of the adapted control scheme presented in the previous section, we performed simulations, namely purely virtual RTHS experiments. Thereby, the Transfer System and the Experimental Substructure are also simulated numerically. The virtual RTHS experiment is implemented in MATLAB®/Simulink® (version R2024a, MathWorks®). We use three different dynamic models for the actuator. To model the dynamics of industrial robotic arms, we use the *Robotic System Toolbox* in MATLAB®. In addition to the available functions for calculating the robot kinematics, we use the *Joint Space Motion Model* block in *Independent Joint Motion* mode with its default parameters to simulate the closed-loop motion of the robot. The algorithms of this toolbox are based on the definition of a *rigidBodyTree* object in MATLAB®. To simulate the KUKA® KR16 robotic arm, we custom-built the *rigidBodyTree* object from CAD data. We also import a *rigidBodyTree* object for the ABB® IRB 120 robot arm, which is already importable from the *Robotic System Toolbox*. For our in-house Stewart platform, a dynamic and kinematic model is available from previous work. The dynamics of its legs have been identified up to 100 Hz and a standard cascaded individual joint controller with velocity feedforward is used for joint control. See [17, 21] for more information on the Stewart platform, its system identification, and the implemented joint controller. To also model the effects of delays, two *Transport Delay* blocks are implemented within the signal flow as shown in fig. 3. These blocks delay a signal by the time delay τ. In our simulation we assume that the delay for the pre- and postprocessing is the same. We also assume ideal transfer behavior of the FTS, i.e. $F' = F$. The control scheme is implemented as described in the previous section.

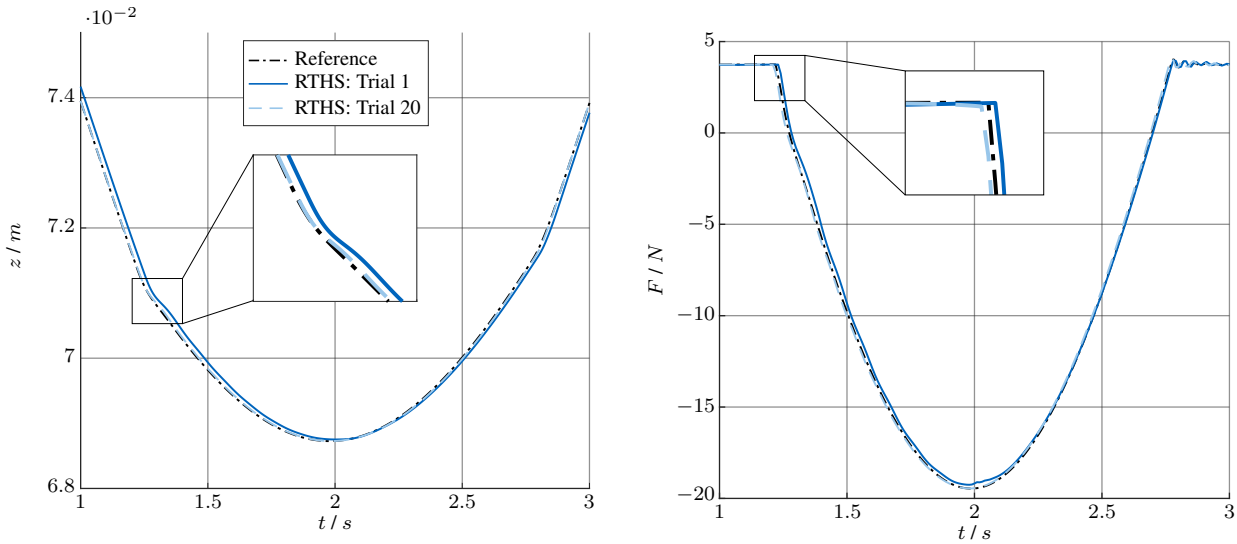
In the virtual RTHS experiment, the Experimental Substructure is also simulated. The simplicity of the system design also allows for easy modeling of the Experimental Substructure as it is a simple one-dimensional oscillator. For a more stable numerical simulation, we added a small damper with damping ratio d_{EXP} in parallel to the spring k_{EXP}. To model ground contact, we activate a ground reaction force on m_{EXP} as soon as it touches the ground. This reaction force is based on a nonlinear spring-damper model as described in [22] (note that we have increased the vertical ground stiffness of this model to $10^6\,\mathrm{kg\,m^{-2}}$ to minimize ground penetration). In table 1 all relevant parameters for the simulation can be found, including the chosen Simulink® solver and the sample time ΔT_{RT}.

Figure 4 shows the result of a virtual RTHS experiment with a delay value[3] of $\tau = 10\,\mathrm{ms}$ and the KUKA® KR16 as actuator without ILC (Trial 1) and after ILC convergence (Trial 20) compared to the reference solution. The reference solution is the actual behavior of the system in fig. 2 and what the RTHS experiment should ideally predict. In fig. 4a the interface displacement z'_{meas} of the virtual RTHS experiment is shown compared to the reference solution z_{ref}. The result

[3]Note that we implement two delays, one for receiving and preprocessing and one for postprocessing and sending, with the same value τ, cf. fig. 3. Thus, the total artificial delay in addition to the robot's trajectory tracking errors is always 2τ.

Table 1 Parameters used in the simulation. For an exact formulation of the initial conditions of the system shown in fig. 2 and the trajectory $z_{ex}(t)$ see [18, 21]. We have also added a small damping to the Experimental Substructure to allow a more stable numerical simulation of it.

variable	value	variable	value
f_{ex}	0.25 Hz	g	$9.81\,\mathrm{m\,s^{-2}}$
m_{NUM}	9.6187 kg	m_{EXP}	0.3813 kg
k_{NUM}	$10\,000\,\mathrm{kg\,s^{-2}}$	d_{NUM}	$500\,\mathrm{kg\,s^{-1}}$
k_{EXP}	$8650\,\mathrm{kg\,s^{-2}}$	d_{EXP}	$10\,\mathrm{kg\,s^{-1}}$
L_p	0.5	$f_{Q,cut}$	4 Hz
G_d	$800\,\mathrm{kg\,s^{-1}}$	T_{LP}	0.01 s
Simulink® Solver	ode3	ΔT_{RT}	0.001 s



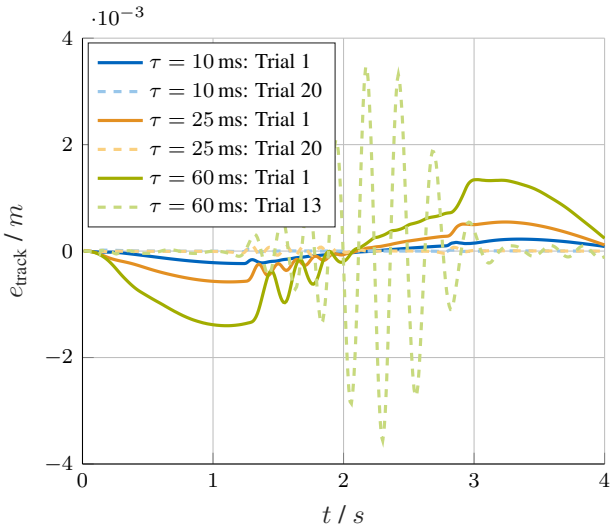
(a) The interface displacement z'_{meas} of the virtual RTHS experiment before and after convergence of the ILC compared to the reference solution z_{ref}.

(b) The interface force F'_{ad} of the virtual RTHS experiment before and after convergence of the ILC compared to the reference solution F_{ref}.

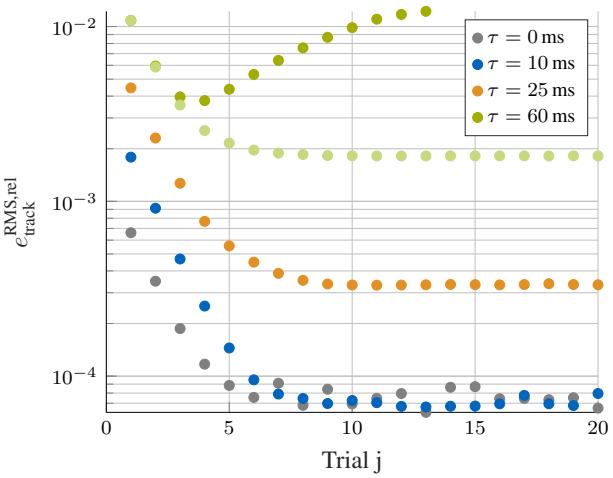
Fig. 4 Result of a virtual RTHS experiment with a delay value of $\tau = 10\,\mathrm{ms}$ and the KUKA® KR16 as actuator without ILC (Trail 1) and after ILC convergence (Trial 20) compared to the reference solution.

clearly shows that our adapted control scheme achieves the desired behavior and proves its applicability, as the RTHS solution after convergence of the ILC is almost identical to the reference solution. Thus, the ILC is able to almost completely mitigate the total error induced by the transfer system on the interface displacement. In fig. 4b the interface force F'_{ad} of the virtual RTHS experiment is shown compared to the reference solution F_{ref}. The results show that the RTHS experiment predicts a slightly earlier contact after convergence of the ILC compared to the reference solution. The reason is that the ILC aims at shifting the motion command for the robot to compensate the total measured error on the interface displacement $e_{track} = z - z'_{meas}$, which includes an additional delay for measuring, postprocessing, and transmitting the actual motion of the robot z'. The Experimental Substructure and the force measurement by the FTS, which is assumed not to be part of the robot, are not affected by this additional delay. Since the ILC takes this delay into account, it shifts the motion command too much from the perspective of the Experimental Substructure, resulting in the early contact compared to the reference solution. This aspect must be taken into account when interpreting the results when the RTHS experiment and the control scheme are implemented as presented. In addition, we mention that even if the NPC was in use, it barely intervened, since the RTHS loop was hardly active[4] during the whole (virtual) experiment (The additional damping force was below 0.2 N during the whole experiment).

[4] In terms of passivity analysis, 'active' refers to the case where the Transfer System induces additional energy into the RTHS loop, which was hardly the case here.



(a) The tracking error e_{track} over time for the different delay values without ILC (Trial 1) and after its convergence (Trial 20).



(b) The relative RMS tracking error $e_{\text{track}}^{\text{RMS,rel}}$ over the trials for the different delay values. The light green values represent the result for $\tau = 60\,\text{ms}$ after adjusting the robustness filter Q, i.e. choosing $f_{\text{Q,cut}} = 2\,\text{Hz}$.

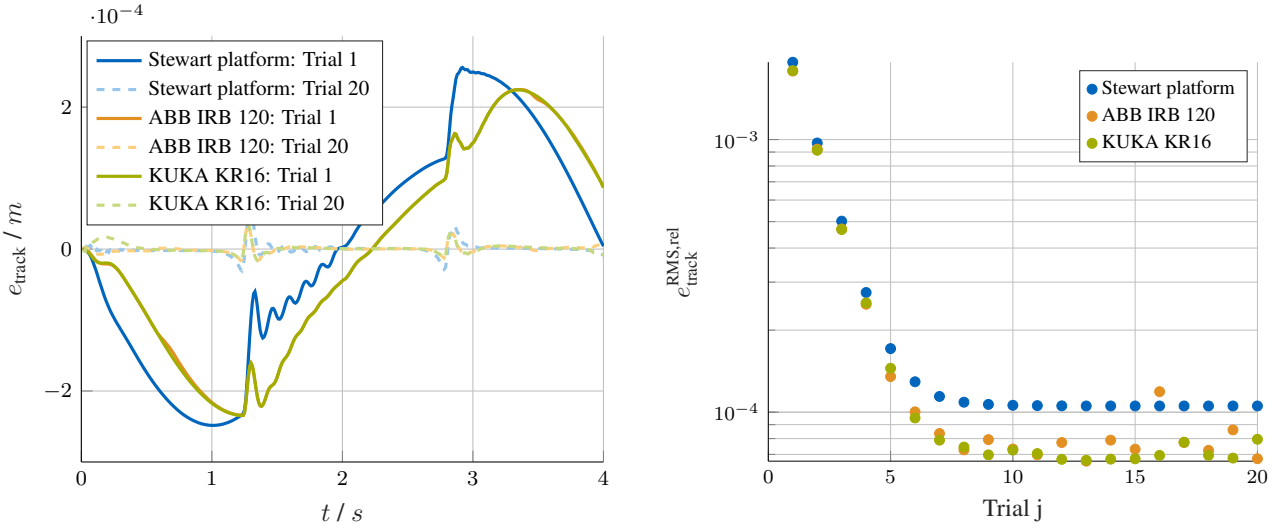
Fig. 5 Simulative investigation of the robustness of our control scheme to different amounts of delay τ for the KUKA® KR16 used as actuator in virtual RTHS experiments.

In fig. 5 the robustness of our controller to different amounts of delay τ in the virtual RTHS experiments is investigated. Again we use the KUKA® KR16 as actuator. We show the tracking error e_{track} over time for the different delay values without ILC (Trial 1) and after its convergence (Trial 20) in fig. 5a, while in fig. 5b we visualize the convergence of the ILC over the trials by calculating the relative root-mean-square (RMS) tracking error $e_{\text{track}}^{\text{RMS,rel}} = \frac{\text{RMS}(e_{\text{track}})}{\text{MAX}(|z|)}$ per trial. It becomes apparent that our control scheme can cope with varying amounts of delay without retuning its parameters, since it significantly reduces the total tracking error and thus compensates for the delays. Note, however, that with a robustness filter Q less than 1 for frequencies above $f_{\text{Q,cut}}$, the residual error after convergence is non-zero. Furthermore, this residual error depends on the transfer behavior of the reference system itself as well as the transfer behavior of the Transfer System. See [18] for the convergence analysis of the ILC in RTHS. This explains why the tracking error is not completely zero after convergence, and its final value also depends on the chosen delay value. In [18] we also show that a convergence criterion must be satisfied for ILC in RTHS. Thus, there is a limit to how much the delay τ can be increased without violating the convergence criterion. This can be seen in fig. 5 for the results for $\tau = 60\,\text{ms}$ as the error diverges over the iterations. However, after retuning the ILC by adjusting the robustness filter Q, e.g. by choosing $f_{\text{Q,cut}} = 2\,\text{Hz}$, stable convergence could be achieved also for $\tau = 60\,\text{ms}$ as shown by the light green values in fig. 5b.

In fig. 6 we show the robustness of our controller to different actuators in the virtual RTHS experiments. The delay is chosen as $\tau = 10\,\text{ms}$ for the simulations with all three actuators. Again, the applicability and robustness of our approach is demonstrated as the tracking error converges similarly for all three actuator types and a significantly lower residual tracking error is achieved after convergence.

Experimental Results

The experimental realization of the RTHS test including the proposed control scheme is visualized in fig. 7. A dSpace® MicroLabBox dS1202 [23] is used to run the numerical time-integration of the Numerical Substructure and the control scheme in real-time. Both are developed in MATLAB®/Simulink® (version R2016b, MathWorks®) on a Host PC and compiled and executed in real-time on the MircoLabBox. The software ControlDesk® (version 6.0, dSpace®) is used on the Host PC to monitor and log relevant variables during real-time execution. The output of the numerical simulation z is adjusted to compensate the measured tracking error of the robot e_{track}. For these first experiments, we do not use a zero-phase low-pass filter as robustness filter Q, but a scalar constant $Q = Q_{\text{Kuka}} \in]0, 1]$ for simplicity. Parallel to the ILC, an additional outer-loop



(a) The tracking error e_{track} over time for the different actuators without ILC (Trial 1) and after convergence (Trial 20).

(b) The relative RMS tracking error $e^{\text{RMS,rel}}_{\text{track}}$ over the trials for the different actuators.

Fig. 6 Simulative investigation of the robustness of our control scheme to different actuators in virtual RTHS experiments with $\tau = 10\,\text{ms}$.

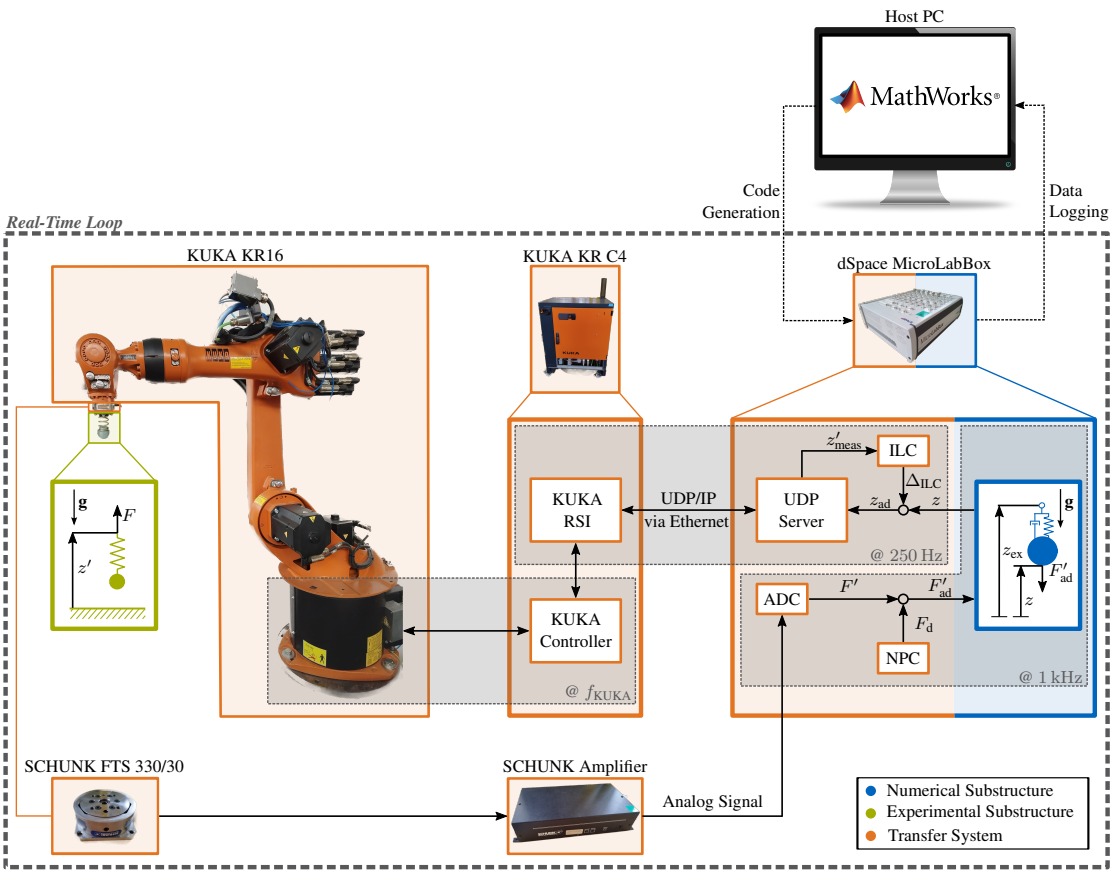


Fig. 7 Illustration of the experimental realization of the RTHS experiment using a KUKA® KR16 as actuator, including all used hardware components, the signal flow and the used cycle rates. Note that the inputs for the NPC are not drawn for the sake of clarity.

P-controller in the robot's task-space is used to further drive the tracking error to zero (not visualized in fig. 7). So the motion command for the robot is: $z_{\text{ad}} = z + \Delta_{\text{ILC}} + K_{\text{p,Kuka}}\,(z - z'_{\text{meas}})$. As robot we use a KUKA® KR16 with a KR C4 robot controller. To move the robot based on an external motion command, the software KUKA.RobotSensorInterface® (RSI) [24] is used. RSI runs on the KUKA® KR C4 controller and handles communication with external devices.

In this work, we use UDP/IP communication via Ethernet. To send the motion command z_{ad} from the MicroLabBox to the KR C4 controller, we implemented a UPD server in Simulink® based on [25] and utilizing the RTI Ethernet Blockset [26] provided by dSpace®. The UDP server is compiled with the rest of the software and executed on the MicroLabBox. The RSI, which acts as the UDP client in our structure, initiates the communication by sending the current measured position of the robot's end effector z'_{meas}. As soon as the UPD server on the MicroLabBox receives a data packet, it processes it, calculates the current motion command z_{ad} and responds by sending this command back to the UPD client on the KR C4. The RSI processes the received data packet, forwards it to the internal KUKA® controller as the desired end-effector position and waits until the next communication cycle starts before sending the next data packet. Before using it in the RTHS setup, we validated the UDP communication in a simple benchmark test similar to [27]. In this test, a 3D sine trajectory in the robot's task-space is commanded to the robot from the MicroLabBox via the UDP communication and recorded for 90 seconds. During this time, no packet loss was detected and we estimated the total delay to be $32\,\text{ms}$. This delay includes delays for receiving and preprocessing the motion command, trajectory tracking errors, and delays for measuring, postprocessing, and transmitting the actual motion of the robot[5].

The actual motion of the KR16 robot z' is transferred to the Experimental Substructure, which consists of a compression spring and a 3D-printed half-sphere attached to the robot's end effector. A table is used as a ground to which the half-sphere intermittently makes contact. A SCHUNK® FTS 330/30 [28] is placed between the compression spring and the robot end effector to measure the interface force. The amplified force signal is fed to one of the analog inputs of the MicroLabBox where, after analog-to-digital conversion (ADC) and adaptation by the NPC, it is used as input for the numerical time-integration in the Numerical Substructure.

As shown, in fig. 7, different cycle rates are used in the experimental setup. The major reason is that the smallest possible cycle time of the RSI is $4\,\text{ms}$. Thus, the time between two data packets sent by the UPD client is $4\,\text{ms}$ and consequently a motion command can only be sent with this cycle time. Therefore, we run the ILC and the UDP server on the MicroLabBox with a cycle rate of $250\,\text{Hz}$, while the rest of the software on the MicroLabBox, namely the sampling of the force measurement, the NPC, and the numerical time-integration of the Numerical Substructure, is run with a cycle rate of $1\,\text{kHz}$. So every fourth time step of the time-integration is used to generate the motion command for the robots. This procedure is done to be able to use a small time step for the numerical time-integration in order to obtain a stable and high fidelity numerical simulation. Note that the NPC requires the current measured velocity of the robot's end effector $\dot{z}'_{\text{meas}}$, which is only available at the rate of the UDP communication. To overcome this, $\dot{z}'_{\text{meas}}$ is held constant for three time steps until the next measurement is available. It is also assumed that the robot's internal motion control operates at a much faster cycle rate than the RSI, which is indicated by f_{KUKA} in fig. 7.

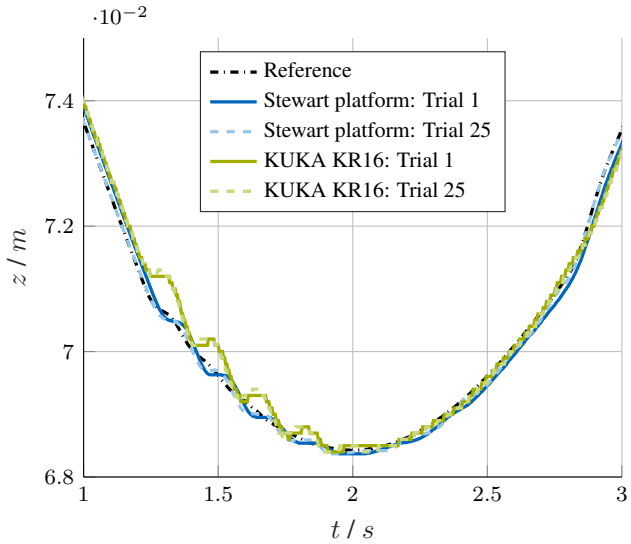
In the following, we compare the results of a first RTHS experiment using the setup from fig. 7 with the performance of the setup from our previous work using the Stewart platform as actuator, which is described in detail in [18, 21]. To be able to compare the results, we also used the SCHUNK ® FTS 330/30 for the measurements with the Stewart platform presented here. However, for the Stewart platform measurements, we used the ILC integrated in the joint controllers as described in [18] and not the version adapted for the commercial robots. It should be noted that in the setup for the Stewart platform also the low-level control is implemented on the MicroLabBox and a desired motor current is commanded to the current controlled joint motors via the analog outputs of the MicroLabBox. Therefore, in this setup no serial communication with additional hardware was required keeping all delays except trajectory tracking errors to a minimum. Table 2 summarizes the important parameters for the experimental investigations.

An RTHS experiment using the KUKA® KR16 robot as actuator was successfully performed with the presented setup. In fig. 8 the results are shown without the ILC (Trial 1) and after 25 ILC trials. They are compared with the reference solution and with the results of an experiment using the Stewart platform as actuator. Although the interface quantities could be roughly replicated using the KUKA setup, the deviation from the reference solution is significantly larger compared to the Stewart platform setup. This is especially true for the period $t \in [1.2\,\text{s}, 2\,\text{s}]$, where not only a delayed trajectory but also additional unwanted oscillations are visible for the measured interface displacement z'_{meas} in fig. 8a. Such unwanted oscillations are typically observed when the RTHS experiment tends to become unstable due to an interface synchronization error. Since the NPC was in operation, the instability and further growth of unwanted oscillations was prevented. Note that the RTHS experiment could not be performed in the presented KUKA setup without the use of NPC due to stability problems, i.e., rapidly growing unwanted oscillations during the contact phase. Apart from that, it can be observed in fig. 8b

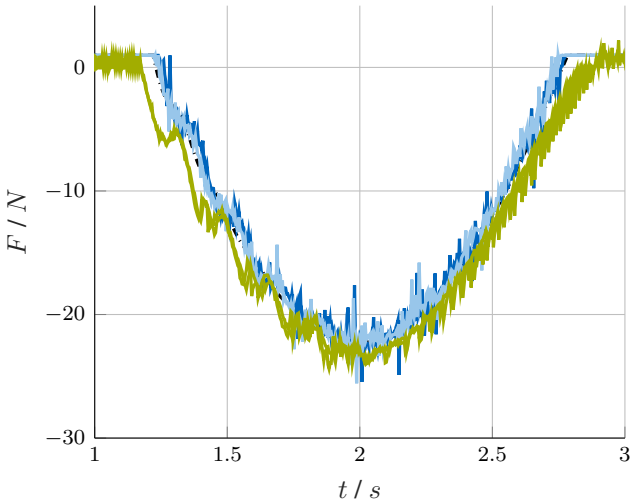
[5] We assume that delays due to processing on the MicroLabBox are well below $1\,\text{ms}$ and thus do not affect the results

Table 2 Parameters used for the experimental investigations. For an exact formulation of the initial conditions of the system shown fig. 2 and the trajectory $z_{ex}(t)$ refer to [18, 21]. Note that there are slight adaptions compared to table 1.

variable	value	variable	value
f_{ex}	0.25 Hz	g	$9.81\,\mathrm{m\,s^{-2}}$
m_{NUM}	9.6187 kg	m_{EXP}	0.0995 kg
k_{NUM}	$10\,000\,\mathrm{kg\,s^{-2}}$	d_{NUM}	$100\,\mathrm{kg\,s^{-1}}$
k_{EXP}	$8650\,\mathrm{kg\,s^{-2}}$	$K_{p,Kuka}$	0.25
$L_{p,Kuka}$	0.04	Q_{Kuka}	0.85
$L_{p,Stewart}$	25	$f_{Q,cut,Stewart}$	4 Hz
G_d	$800\,\mathrm{kg\,s^{-1}}$	T_{LP}	0.01 s
Simulink® Solver	ode8	ΔT_{RT}	0.001 s



(a) The measured interface displacement z'_{meas} of the RTHS experiments compared to the reference solution z_{ref}.



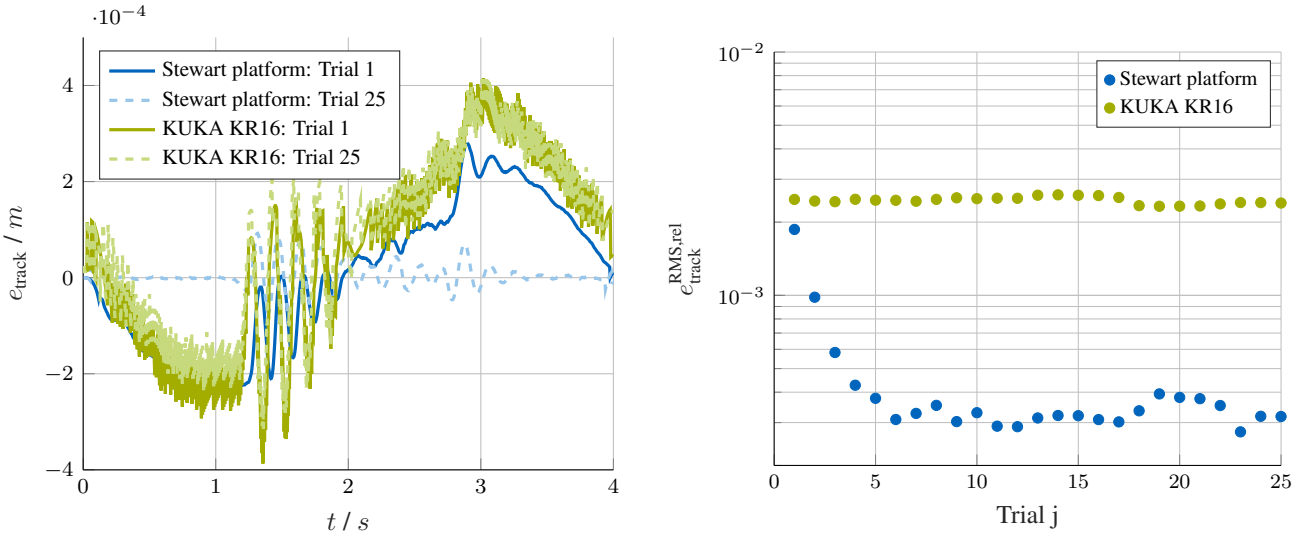
(b) The measured interface displacement F'_{ad} of the RTHS experiments compared to the reference solution F_{ref}.

Fig. 8 Result of an RTHS experiment using the KUKA® KR16 robot as an actuator without ILC (Trial 1) and after 25 ILC trials. The results are compared with the reference solution and with the results of an experiment using the Stewart platform as an actuator.

that the RTHS experiment with the KUKA setup predicts early and late contact. However, this could be due to a slightly wrong initial condition when configuring the starting positions of the KR16 robotic arm, which requires further investigation in future work.

In fig. 9 the tracking performance is analyzed. In fig. 9a the measured tracking error $e_{track} = z - z'_{meas}$ is visualized over time without ILC (Trial 1) and after 25 ILC trials. In general, the magnitude of the tracking errors without ILC is similar for both setups. Again, the unwanted oscillations in the period $t \in [1.2\,\mathrm{s}, 2\,\mathrm{s}]$ are visible, which are larger for the KUKA setup than for the Stewart platform setup. Additionally, in the second half of the experiment with $t \in [2\,\mathrm{s}, 4\,\mathrm{s}]$, where the KR16 robotic arm has to work against gravity, the tracking error is larger than when using the Stewart platform setup. While the ILC converged well for the Stewart platform setup and significantly reduced the tracking error (and the deviation from the reference as shown in fig. 8), no ILC parameter combination could be found for the KUKA setup in this early stage study that then resulted in a converging ILC behavior that significantly reduced the tracking error. This can be seen in fig. 9b as there is almost no change in the relative RMS tracking error over the ILC trials. Thus, although a robust performance was demonstrated in the virtual RTHS tests in the previous section, we could not demonstrate the performance of our control scheme in the experimental setup with the KUKA® KR16.

Future work is therefore required for further investigation. First, a more sophisticated tuning of the control parameters should be performed, since, for example, the NPC parameters were not changed and their influence on the results was not investigated. As described above, $\dot{z}'_{meas}$, which is used in the NPC to compute the power at the interface to the Experimental



(a) The tracking error e_{track} over time for the different experimental setups without ILC (Trial 1) and after 25 ILC trials.

(b) The relative RMS tracking error $e_{\text{track}}^{\text{RMS,rel}}$ over the trials for the different experimental setups.

Fig. 9 Comparison of the tracking performance of the RTHS setup with the KUKA® KR16 as actuator with an experiment using the Stewart platform as an actuator.

Substructure, is held constant over three time steps, while all other inputs to the NPC can change during these time steps. It needs to be investigated if this affects the performance of the NPC. For the ILC, it should be investigated whether using a zero-phase lowpass filter for Q instead of the constant could improve the convergence of the ILC. Note that such a zero-phase lowpass filter was used in the simulations of the previous section as well as for the Stewart platform setup. In addition, later simulations suggested that the additional task-space P controller used in the KUKA setup impeded ILC performance. Therefore, removing this controller may also improve the experimental results. Finally, the general performance of the ILC in the UDP communication framework needs further verification. Although we have shown in the previous section that the ILC can cope with large amounts of delay, the implementation on the hardware introduces further uncertainties, e.g. the processes on the MircoLabBox and on the KR C4 are not actively synchronized. In addition, the reliability of the internal processing on the KR C4 is unclear, which could lead to varying delays that could affect the performance of the ILC.

Conclusion

In this paper, we presented a framework to perform robust RTHS experiments using commercial robotic arms with high fidelity by combining ILC with NPC. The ILC aims to improve experiment fidelity by using iterative learning over multiple trials to mitigate compatibility errors in interface displacements, while the NPC ensures experiment stability by inducing artificial damping in case of instabilities. Both controllers act as pure outer-loop controllers in the task-space of the robotic arm, which makes the approach independent of the robot used, since no integration within the low-level controllers is required. In addition, our approach does not require any system modeling, allowing for easy implementation in different types of test setups.

In virtual RTHS experiments, we demonstrated the ability of the approach to significantly reduce interface synchronization errors to provide high-fidelity RTHS results. We also showed its robustness to different amounts of delay in the Transfer System and different actuator models.

We successfully implemented an RTHS experiment using a KUKA® KR16 as an actuator and integrated our control framework. The ability of the NPC to provide stable RTHS experiments could be verified, because without the use of the NPC, the designed experiment suffered from strongly growing unwanted oscillations. The performance of the ILC, on the other hand, has not yet been experimentally demonstrated. However, the presented experimental results are preliminary and further investigations are planned.

References

1. de Klerk, D., Rixen, D. J., and Voormeeren, S. N. "General Framework for Dynamic Substructuring: History, Review and Classification of Techniques". *AIAA Journal* 46.5 (2008), pp. 1169-1181. DOI: 10.2514/1.33274.
2. Nakata, N., Dyke, S. J., Zhang, J., Mosqueda, G., Shao, X., Mahmoud, H., Head, M. H., Bletzinger, M. E., Marshall, G. A., Ou, G., and Song, C. *Hybrid Simulation Primer and Dictionary.* Tech. rep. George E. Brown, Jr. Network for Earthquake Engineering Simulation, 2014.
3. Blakeborough, A., Williams, M. S., Darby, A. P., and Williams, D. M. "The development of real-time substructure testing". *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 359.1786 (Sept. 2001). Ed. by Chapman, S. J., pp. 1869-1891. DOI: 10.1098/rsta.2001.0877.
4. Williams, M. S. "Real-time hybrid testing in structural dynamics". *5th Australasian Congress on Applied Mechanics, ACAM 2007.* 2007.
5. Horiuchi, T., Inoue, M., Konno, T., and Namita, Y. "Real-time hybrid experimental system with actuator delay compensation and its application to a piping system with energy absorber". *Earthquake Engineering & Structural Dynamics* 28.10 (1999), pp. 1121-1141. DOI: 10.1002/(SICI)1096-9845(199910)28:10⟨1121::AID-EQE858⟩3.0.CO;2-O.
6. Darby, A. P., Williams, M. S., and Blakeborough, A. "Stability and Delay Compensation for Real-Time Substructure Testing". *Journal of Engineering Mechanics* 128.12 (Dec. 2002), pp. 1276-1284. DOI: 10.1061/(ASCE)07339399(2002)128:12(1276).
7. Botelho, R. M. and Christenson, R. E. "Robust Stability and Performance Analysis for Multi-actuator Real-Time Hybrid Substructuring". *Dynamics of Coupled Structures, Volume 4.* Ed. by Allen, M., Mayes, R. L., and Rixen, D. J. Cham: Springer International Publishing, 2015, pp. 1-7. DOI: 10.1007/978-3-319-15209-7_1.
8. Bartl, A., Insam, C., and Rixen, D. J. "Stability Issues in Hardware-in-the-Loop Tests of Flexible Components". *PAMM* 18.1 (2018). DOI: 10.1002/pamm.201800361.
9. Carrion, J. E. and Spencer, B. F. "Model-based Strategies for Real-time Hybrid Testing". *Newmark Structural Engineering Laboratory Report Series 006* (Dec. 2007). URL: http://hdl.handle.net/2142/3629.
10. Insam, C., Kist, A., and Rixen, D. J. "High Fidelity Real-Time Hybrid Substructure Testing Using Iterative Learning Control". *ISR 2020; 52th International Symposium on Robotics.* Dec. 2020.
11. Hochrainer, M. J. and Puhwein, A. M. "Investigation of Nonlinear Dynamic Phenomena Applying Real-Time Hybrid Simulation". *Nonlinear Structures and Systems, Volume 1.* Ed. by Kerschen, G., Brake, M. R. W., and Renson, L. Cham: Springer International Publishing, 2020, pp. 125-131. DOI: 10.1007/978-3-030-12391-8_16.
12. Hashan Peiris, L. D., Plummer, A. R., and Du Bois, J. L. "Passivity Control in Real-Time Hybrid Testing". *2018 UKACC 12th International Conference on Control (CONTROL).* Sept. 2018, pp. 317-322. dOI: 10.1109/CONTROL.2018.8516814.
13. Insam, C., Peiris, L. D. H., and Rixen, D. J. "Normalized passivity control for hardware-in-the-loop with contact". *International Journal of Dynamics and Control* 9.4 (Mar. 2021), pp. 1471-1477. DOI: 10.1007/s40435-021-00790-8.
14. Palacio-Betancur, A. and Gutierrez Soto, M. "Recent Advances in Computational Methodologies for Real-Time Hybrid Simulation of Engineering Structures". *Archives of Computational Methods in Engineering* 30.3 (Apr. 2023), pp. 16371662. DOI: 10.1007/s11831-022-09848-y.
15. KUKA Roboter GmbH. *Technical Data KR 16.* Augsburg, 2005.
16. KUKA Roboter GmbH. *KR C4 Betriebsanleitung.* Augsburg, 2018.
17. Insam, C., Göldeli, M., Klotz, T., and Rixen, D. J. "Comparison of Feedforward Control Schemes for Real-Time Hybrid Substructuring (RTHS)". *Dynamic Substructures, Volume 4.* Ed. by Linderholt, A., Allen, M., and D'Ambrogio, W. Conference Proceedings of the Society for Experimental Mechanics Series. Cham: Springer International Publishing, 2021, pp. 1-14. DOI: 10.1007/978-3-030-47630-4_1.
18. Insam, C., Kist, A., Schwalm, H., and Rixen, D. J. "Robust and high fidelity real-time hybrid substructuring". *Mechanical Systems and Signal Processing* 157 (Aug. 2021), p. 107720. DOI: 10.1016/j.ymssp.2021.107720.
19. Insam, C., Ballat, L.-M., Lorenz, F., and Rixen, D. J. "Hardware-in-the-Loop Test of a Prosthetic Foot". *Applied Sciences* 11.20 (Oct. 2021), p. 9492 . DOI: 10.3390/app11209492.
20. Insam, C., Bartl, A., and Rixen, D. J. "A Step Towards Testing of Foot Prostheses Using Real-Time Substructuring (RTS)". *Special Topics in Structural Dynamics & Experimental Techniques, Volume 5.* Ed. by Dervilis, N. Conference Proceedings of the Society for Experimental Mechanics Series. Cham: Springer International Publishing, 2020, pp. 1-9. DOI: 10.1007/978-3-030-12243-0_1.
21. Insam, C. "Fundamental Methods for Real-Time Hybrid Substructuring with Contact: Enabling Testing of Prosthetic Feet". PhD thesis. Technical University of Munich, 2022.
22. Geyer, H. and Herr, H. "A Muscle-Reflex Model That Encodes Principles of Legged Mechanics Produces Human Walking Dynamics and Muscle Activities". *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 18.3 (June 2010), pp. 263-273. DOI: 10.1109/tnsre.2010.2047592.
23. dSpace GmbH. *MicroLabBox Product Brochure.* 2020.
24. KUKA Roboter GmbH. *KUKA.RobotSensorInterface 3.2 Documentation - Version: KST RSI 3.2 V1.* Augsburg, 2013.
25. Leland, J. *kukaslxctrl. GitHub repository.* 2017. URL: https://github.com/mitmedialab/kukaslxctrl(visitedon03/12/2024).
26. dSPACE GmbH. *RTI Ethernet Blockset. Reference for RTI Ethernet Blockset 1.2.1.* 2016.
27. Lisowski, W. and Miekina, L. "Exemplary application of a virtual sensor with a manipulating robot". *The 18th Conference "Mechatronic Design Workshops".* May 2018.
28. Fritz Schunk GmbH. *Montage- und Bedienungsanleitung für 6 Achsen Kraft-/Momentensensor Type FTS.* 1992.