Latency and sampling compensation in mixed-reality-in-the-loop simulations of production systems

Schnierle, M.; Röck, S.

Production Engineering, Springer-Verlag, 2022

https://doi.org/10.1007/s11740-022-01175-2

Production Engineering
https://doi.org/10.1007/s11740-022-01175-2

ASSEMBLY AND AUTOMATION

Latency and sampling compensation in mixed-reality-in-the-loop simulations of production systems

Marc Schnierle[1] · **Sascha Röck**[1]

Received: 6 September 2022 / Accepted: 24 November 2022

Abstract
X-in-the-Loop Simulation methods (Model-in-the-Loop, Software-in-the-Loop and Hardware-in-the-Loop Simulation) enable the virtual commissioning of production systems in the mechatronic development process by coupling control systems and digital twins. Mixed-Reality-in-the-Loop Simulations (MRiLS) extend this principle with Mixed Reality visualisation technologies to enhance the visual fusion of reality (e.g., real environment and human) and virtuality (digital twins), opening up a simulation loop in the reality-virtuality continuum with novel application potentials e.g., from development, training to maintenance. A major challenge in MRiLS is the positioning error of actuated real-data driven virtual components caused by latency and sampling processes between the industrial control system and the Mixed Reality device which significantly limits the application scope. To reduce this error, the paper proposes a compensation method that synchronises the Mixed Reality device to the stable time base of the control system and integrates a predictive positioning of virtual components. A software-based synchronisation method is presented, which allows the online estimation of the End-to-End latency between the control cycle and the visualisation. For prediction, interpolating and approximating section-wise defined polynomials are analysed. The error reduction by applying the compensation method is shown in a realisation example of a virtual gripper linked to a real robot kinematics.

Keywords X-in-the-loop simulation · Real-time simulation · Virtual commissioning · Mixed-reality-in-the-loop simulation · Augmented reality · Latency and sampling compensation

1 Introduction

The test configurations Model-in-the-Loop, Software-in-the-Loop and Hardware-in-the-Loop Simulation are part of the X-in-the-Loop Simulation (XiLS) method series which enables the virtual commissioning of automated production systems in the mechatronic development process [1]. The approach of XiLS is increasingly adopted in mechanical engineering [2], as it enables the parallelisation of development steps and the early testing of the control system by simulating error situations with the digital twin of the production system.

Besides the great advantages, the test configurations are limited in the visual fusion of reality (real environment or human) and virtuality (digital twin), because they visualise the 3D-geometry models of the digital twins mainly with 2D-projections on conventional computer screens as an exocentric visualisation that neither incorporates user perspective nor takes human stereoscopic vision into account.

The Mixed-Reality-in-the-Loop Simulation (MRiLS) addresses this limitation by extending the XiLS method series with visualisation and interaction methods of Mixed Reality, which introduce a reality-virtuality continuum [3]. As shown in Fig. 1, the MRiLS integrates the real environment and the human (user) into the simulation loop between the control system and the digital twin of the production system. Figure 2 illustrates this concept by an exemplary MRiLS scenario. The shown user interacts with an industrial robot via the real teach pendant of the

Marc Schnierle and Sascha Röck contributed equally to this work.

✉ Marc Schnierle
marc.schnierle@hs-esslingen.de

Sascha Röck
sascha.roeck@hs-esslingen.de

[1] Virtual Automation Lab, Faculty of Mechanical and Systems Engineering, Esslingen University of Applied Sciences, Kanalstraße 33, 73728 Esslingen, Baden-Württemberg, Germany

Published online: 13 December 2022

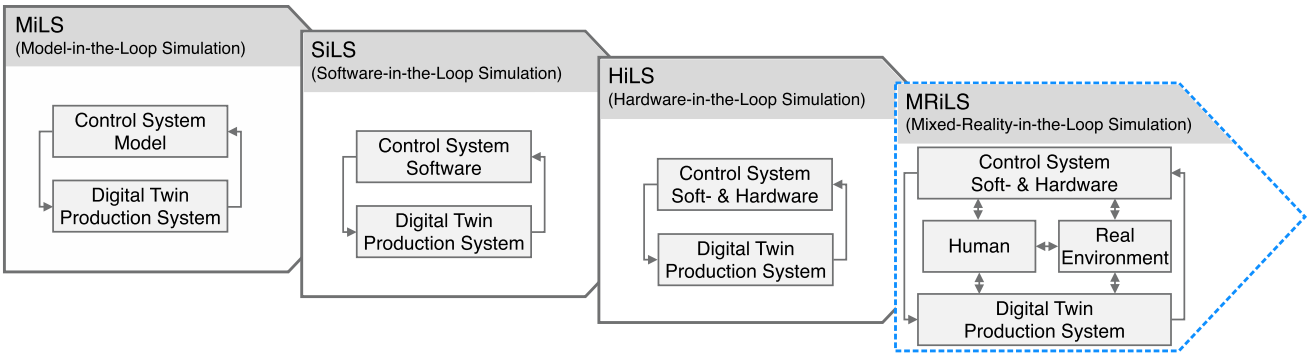

Fig. 1 Extension of the X-in-the-Loop Simulation method series by the Mixed-in-the-Loop Simulation

Fig. 2 Exemplary scenario of Mixed-Reality-in-the-Loop Simulation (virtual objects highlighted in green)

control system and utilises optical see-through Augmented Reality glasses (referred to as MR-device in the following) to view a perspective and true-to-scale visualisation of the Mixed Reality scene. Stationary peripheral components and a gripper linked to the robot kinematics based on position data from the control system are displayed as virtual objects into the reality (the virtual objects in Fig. 2 are highlighted in green for better visibility). The virtual objects are perceived as anchored in the real environment and distances can be sensed by the user through the egocentric (view-dependent) and stereoscopic (separate image per eye) rendering. The resulting fusion of real environment and digital twin combined with the immersion of the user opens up numerous applications with different levels of reality and virtuality, e.g., in development, training or maintenance scenarios. This concept allows the enhancement of Mixed Reality applications in robotics [4] and mechanical engineering [5] with XiLS methods.

For user immersion and applicability of the MRiLS, both the stationary and the moving objects must be positioned correctly in each visualisation cycle of the MR-device. The basis of the positioning process is the continuos adaption of the egocentric visualisation to a change in the user's viewing pose in order to display the updated position and perspective of the digital twin in the MR-device coordinate system. The user-position-dependent transformation matrix between the MR-device and the world coordinate system is determined via tracking methods (e.g., time of flight or markerbased tracking). The error in positioning stationary components (e.g., non-moving peripherals in Fig. 2) has been the subject of various scientific studies (e.g., [6]) and is primarily due to tracking errors, static model errors and reference errors between the coordinate system of the digital twin and the real environment [7], that can be compensated by optimisation of tracking methods or adjustments of calibration and model parameters.

In addition to the stationary objects, the real-data driven objects like the virtual gripper in Fig. 2 are moving in the local coordinate system of the digital twin based on position data from the control system. Through this mechanism positioning errors e arise, because sampling rates and latencies from communcation, networking, data processing and visualisation result in an End-to-End latency L between the control system and the MR-device as shown in Fig. 3. The End-to-End latency L of a Mixed Reality system refers to the time span from the creation of information in the control system to the visualisation on the MR-device [8].

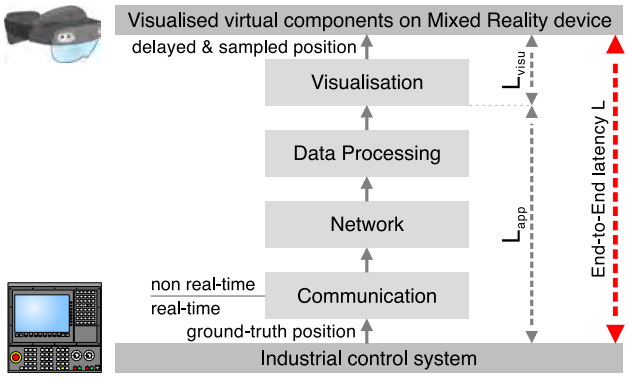

Fig. 3 Latency and sampling chain in a Mixed-Reality-in-the-Loop Simulation

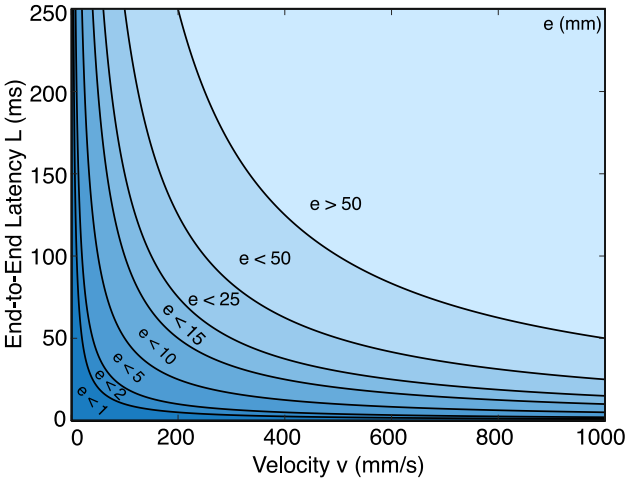

Fig. 4 Positioning error e as a function of constant velocity v and End-to-End latency L

Due to varying latencies in wireless networks and the non-deterministic processing and visualisation steps on the MR-device, L must be assumed to be time-varying.

To emphasise the influence of End-to-End latency L on the positioning error of real-data driven objects, Fig. 4 visualises fields of positioning errors e as function of the constant velocity v of the moving objects and L with the relationship: $e = v \cdot L$ in analogy to the proportionality $\Delta s = v \cdot \Delta t$ of a distance Δs to velocity v and duration Δt. The diagram illustrates that relevant positioning errors occur at common operating velocities up to 1000 mm/s (e.g., maximum end-effector speed of the robot UR 5 from Universal Robots). The effective End-to-End latency L is strongly dependent on the operating conditions such as the control system, the network equipment, the data acquisition protocol and intermediate processing steps. In total, L can reach values in the range of approximately 30–250 ms, due to data acquisition with subscription intervals of 10–100 ms, network and processing latency of 10–100 ms and visualisation latency of 10–50 ms caused by rendering and display pipelines with variable refresh rates (see [9] for a technological explanation of the resulting stutter and delay). The End-to-End latency therefore not only significantly reduces the user experience [10], but also reduces the applicability of MRiLS in an industrial context.

In the industrial Mixed Reality application context, several research works (e.g., robotics [11] or machine tools [12]) integrate data from the fieldbus level in Mixed Reality visualisations, but do not pursue End-to-End latency compensation between control system and MR-device. The presented problem of End-to-End latency can be associated to challenges in telerobotics [13], the control of sampled and latency-affected systems [14], Multiplayer gaming [15], tracking [16] as well as human-computer interactions [17]. In these areas, prediction techniques are applied to reduce the error between the sampled and delayed signal and the ground truth of the control system without taking into account the characteristics of MRiLS. For example, compensating solely the communication latency is not sufficient in MRiLS applications, as a synchronisation between the control cycle in the control system and the visualised object on the MR-device is needed. The presented research therefore is based on approaches from the areas mentioned above and proposes a compensation method suited for MRiLS. The compensation method including End-to-End latency estimation and a predictive positioning of the virtual objects is explained in Sect. 2. Subsequently, Sect. 3 presents a realisation example of a virtual gripper linked to a real robot kinematics.

2 Compensation method

The compensation method for reducing the positioning error e is illustrated in Fig. 5 using an exemplary position profile of a robot axis as ground truth (blue line).

The position $s(t_k)$ generated on the control system is sampled at time t_k (blue squares) and is visualised on the MR-device delayed by the End-to-End latency L_k at time $t_{k,L} = t_k + L_k$ (red points). The resulting error $e(t_{k,L}) = s(t_{k,L}) - s(t_k)$ can be seen in the diagram as difference between the ground truth position $s(t_{k,L})$ and the zero-order hold interpolation of the position $s(t_k)$ visualised on the MR-device (red dashed line). Using the position data $[s(t_0), ..., s(t_k)]$ sampled from the control system and the associated End-to-End latencies $[L_0, ..., L_k]$, a prediction function (green line) can be used to calculate an estimation $s^*(t_{k,L})$ of the position $s(t_{k,L})$ and thus reduce the positioning error $e^*(t_{k,L}) = s(t_{k,L}) - s^*(t_{k,L})$ on the MR-device compared to the uncompensated error $e(t_{k,L})$. Between $t_{k,L}$ and $t_{k+1,L}$ the estimated value can be determined in each visualisation step

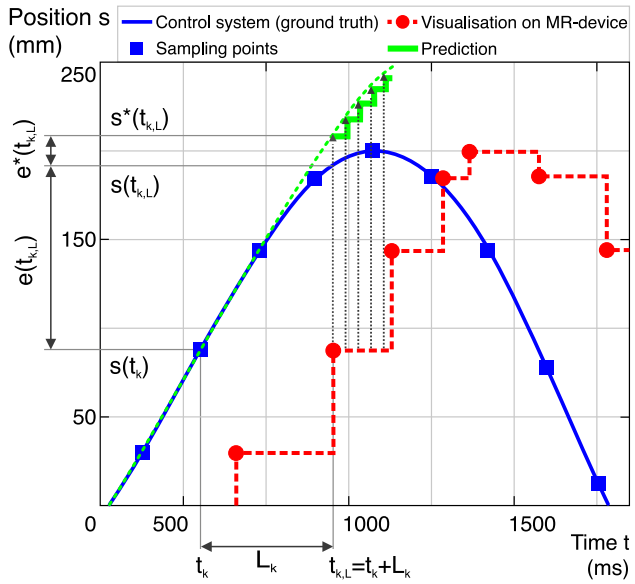

Fig. 5 Principle of the proposed compensation method

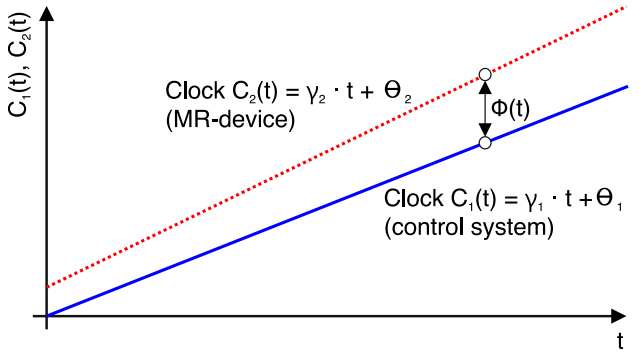

Fig. 6 Linear clock model for control system and MR-device (illustration and nomenclature inspired by [24])

of the MR-device (dotted grey arrows) by evaluating the prediction function.

For the calculation of $s^*(t)$ an online estimation of the End-to-End latency L_k^* and a prediction function are necessary. These two topics will be discussed in the following.

2.1 Online estimation of End-to-End latency L_k^*

The determination of the End-to-End latency of a Mixed Reality system is pursued in various research works with the same methodological principle. The main characteristic is the generation of a test signal, which is acquired by both the MR-device and an accurate reference system (e.g., high speed camera). The time difference between the two signals represents L_k^*. Examples are mechanical generated test signals with pendulum [18] or turntable [19], electrical generated test signals with light emitting diodes (LED) [20] or software-based generated test signals with encoded timestamps [21]. For further description of these methods see [22] and [23].

This principle cannot be directly transferred to MRiLS, because a MRiLS requires the online estimation of L_k^* without additional hardware setup. Therefore this paper proposes to decompose L_k^* into two parts (see Fig. 3):

$$L_k^* = L_{app,k}^* + L_{visu,k}^* \tag{1}$$

- Application latency $L_{app,k}^*$: time span from the creation of information in the control system to the end of processing on the MR-device, before the rendering job (process in the graphics pipeline) is triggered. The estimation of $L_{app,k}^*$ is explained in Sect. 2.1.1.
- Visualisation latency $L_{visu,k}^*$: time span from triggering the rendering job until the virtual content is visible on the display of the MR-device. The estimation of $L_{visu,k}^*$ is explained in Sect. 2.1.2.

2.1.1 Clock synchronisation for estimation of $L_{app,k}^*$

The application latency $L_{app,k}^*$ can be estimated by subtracting the timestamp $C_1(t_k)$ of the control system (included in the transmitted position data packet) from the timestamp $C_2(t_k + L_{app,k})$ of the MR-device when processing the position data:

$$L_{app,k}^* = C_2(t_k + L_{app,k}) - C_1(t_k) \tag{2}$$

This approach requires the clock synchronisation of the control system clock C_1 and the MR-device clock C_2 to ensure an equal time-reference basis. Figure 6 shows the progression of C_1 and C_2 with a linear clock model $C_i = \gamma_i \cdot t + \Theta_i$ over a shared base time t.

The objective is the synchronisation of C_2 to the master clock C_1. The parameters of C_1 as master clock can be assumed to be $\gamma_1 = 1$ and $\Theta_1 = 0$ [24]. The unknown time offset Φ between C_2 and C_1 is given by:

$$\begin{aligned}\Phi(t) &= C_2(t) - C_1(t)\\ &= (\gamma_2 - \gamma_1) \cdot t + \Theta_2 - \Theta_1\\ &= (\gamma_2 - 1) \cdot t + \Theta_2\end{aligned} \tag{3}$$

It should be mentioned at this point that protocols from the field of synchronisation of physical clocks in distributed computer systems like the Network Time Protocol (NTP, RFC 958) and Precision Time Protocol (PTP, IEEE 1588) could in principle be adapted. But in this work it is assumed that the MR-device does not support hardware-based synchronisation. Synchronisation protocols and

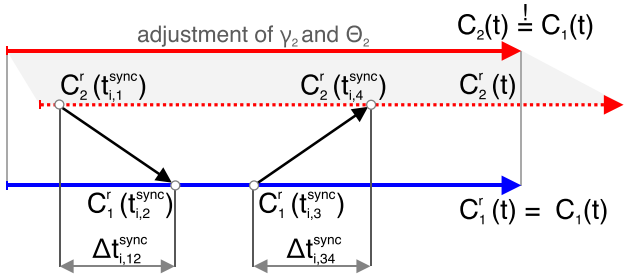

Fig. 7 Synchronisation sequence to determine the time offset Φ^* from measured timestamps C_1^r and C_2^r

implementations with hardware-based timestamps are consequently not suitable for MRiLS, as the method should be independent of the requirements on the network adapters of the MR-devices. Therefore, this paper proposes to perform clock synchronisation in the application layer of the MR-device based on the principle of NTP in further development to [25], which demonstrated the basic concept in wide area Industrial Internet of Things (IIoT) networks. The procedure extends the Cristian's algorithm [26] and is characterised by the continuous exchange of synchronisation messages containing real measured timestamps C_1^r and C_2^r which can be used to estimate the time offset Φ^* for adjusting the parameters γ_2 and Θ_2 on the MR-device. Figure 7 shows one synchronisation sequence i.

With sufficiently symmetrical transmission latencies, the following relation can be assumed:

$$\Delta t_{i,12}^{sync} = \Delta t_{i,34}^{sync} \tag{4}$$

Under the assumption of Eq. 4 the time offset Φ_i^* can be calculated from the aquired timestamps (for derivation of the equation see [27]):

$$\Phi_i^* = \Phi^*(t_{i,4}^{sync}) = \frac{C_2^r(t_{i,1}^{sync}) - C_1^r(t_{i,2}^{sync})}{2} + \frac{C_2^r(t_{i,4}^{sync}) - C_1^r(t_{i,3}^{sync})}{2} \tag{5}$$

According to Eq. 3, two synchronisation sequences would be sufficient to determine γ_2 and Θ_2 with the condition $\Phi^* - \Phi \overset{!}{=} 0$. However, it must be assumed that the transmission latencies vary over the time and the underlying symmetry condition in Eq. 4 is violated. For this reason, multiple synchronisation sequences are executed and an overdetermined system of equations is solved using a least squares approach for parameter estimation of γ_2 and Θ_2:

$$\min \{\Phi^*(t_{i,4}^{sync}) - \Phi(t_{i,4}^{sync})\} =$$
$$\min \sum_{i=1}^{n} \underbrace{\left(\Phi_i^* - ((\gamma_2 - 1) \cdot t_{i,4}^{sync} + \Theta_2)\right)}_{\text{fitting error } \epsilon_i^{sync}}^2 \tag{6}$$

In this minimisation problem measurement points with a large fitting error ϵ_i^{sync} may indicate an outlier. To obtain a more robust result in Eq. 6, this approach is extended with a RANSAC (random sample consensus) algorithm [25]. The RANSAC algorithm tests different hypotheses for γ_2 and Θ_2 against multiple measurements and maximises the number of inliners within a given error bound. Thus outliers can be excluded from the parameter estimation and the synchronisation can be made more insensistive to scattered measurements. To enhance this effect a subsequent moving average filter for γ_2 and Θ_2 was integrated in this algorithm.

Figure 8 shows a measurement example during synchronisation of a Microsoft HoloLens 2 Industrial with a EtherCAT fieldbus participant of a Beckhoff TwinCAT control system. As the synchronisation sequences progress, the parameters γ_2 and Θ_2 of the MR-device clock C_2 converge and the synchronisation error ϵ_i^{sync} is minimised (under the

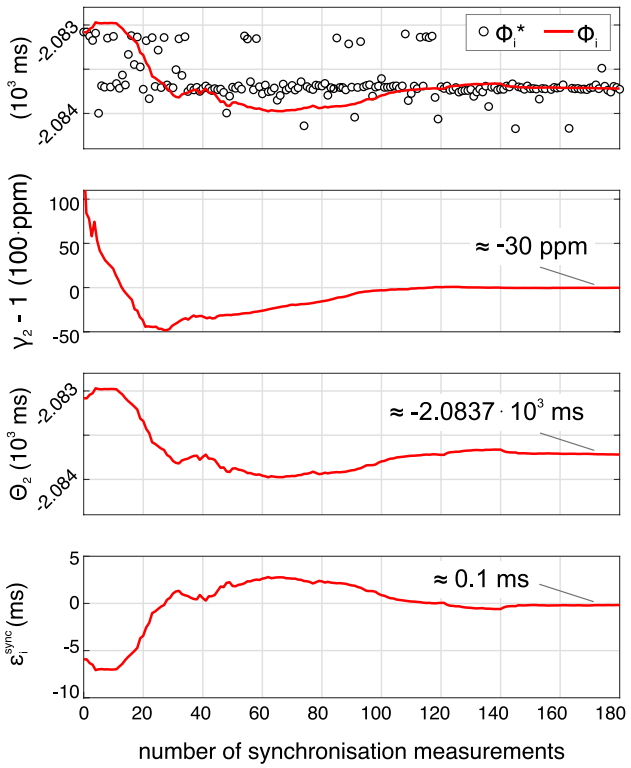

Fig. 8 Software-based synchronisation between a Microsoft HoloLens 2 Industrial and a Beckhoff TwinCAT fieldbus node, RANSAC (100 measurements history, 20 hypothesis samples), moving average filter (20 samples), time span between measurements: 200 ms

test conditions, $\epsilon_i^{sync} < 1$ ms was achieved). The clock skew $\gamma_2 - 1$ is given in parts per million (ppm) and thus describes the drift of C_2 and C_1 due to different clock rates in $\frac{\mu s}{s}$. The large measured values of Φ^* can be explained by the different starting times of the clocks. Furthermore the results showed that the Round-Trip Time (time span from sending the synchronisation message to its completion) $RTT_i = C_2^r(t_{i,4}^{sync}) - C_2^r(t_{i,1}^{sync})$ can be used as quality indicator of the synchronisation sequence. With the presented method, the clock of the MR-device C_2 can be synchronised to the clock C_1 of the control system and $L_{app,k}^*$ can be determined for each incoming data packet with Eq. 2.

2.1.2 Online estimation of $L_{visu,k}^*$ by calibration with visible light communication

Estimating the visualisation latency L_{visu}^* requires the quantification of the time span between triggering the rendering job and displaying the virtual content on the MR-device. To investigate and measure this time span, Visible Light Communication (VLC) is utilised in this work by transfering and further development of the principle shown in [20] and [28]. Figure 9a illustrates the measurement method. The control system operates a LED-matrix that displays the timestamp $C_1(t_{VLC})$ as LED-code in each control cycle. The MR-device synchronises to the time base of the control system by connecting to a fieldbus participant that supports the software-based method presented in Sect. 2.1.1. After convergence of the clock parameters γ_2 and Θ_2, the MR-device visualises its current timestamp $C_2(t_{MR})$ as virtual text element. An external camera system is then used to capture an overlay of the virtually displayed timestamp and the LED-matrix by recording through the MR-device. The image captured by the external camera system contains all the information for determining L_{visu}^* as shown in Fig. 9b and can be automatically evaluated with image processing and optical character recognition. During the rendering and display time, the timestamp $C_1(t_{VLC})$ displayed on the LED-matrix has already progressed compared to $C_2(t_{MR})$. Accordingly, the visualisation latency L_{visu}^* can be expressed by the difference between these timestamps, assuming error-free synchronisation at the application layer, meaning $C_2(t) = C_1(t)$:

$$L_{visu}^* = C_1(t_{VLC}) - C_2(t_{MR}) \tag{7}$$

Although the method is suitable for determining the visualisation latency based on the observed image, this approach is not suitable for online estimation of L_{visu}^* in a MRiLS because an additional camera system is required and the necessary information for Eq. 7 can only be extracted from the image at the end of the rendering pipeline.

It was found that the visualisation latency remains constant up to a certain number of rendered vertices, which corresponds to the load on the rendering pipeline. As a metric of the rendering load, the render time Δt_{render} of the last frame can be obtained in the application layer on the MR-device by measuring the time between triggering the render job and the returning message from the rendering pipeline. To control the load, geometry meshes with adjustable vertex counts are procedurally generated and displayed on the MR-device. Figure 10a, b present two measurements of Δt_{render} and L_{visu}^* with different loads (300.000 and 1.500.000 vertices) on the rendering pipeline for consecutive measurement images of the VLC-method. In case of normal load in Fig. 10a between Δt_{render} and L_{visu}^* a constant relationship can be observed, whereas in case of high load in Fig. 10b this relationship is no longer valid. Figure 11a presents the mean and deviation bars of L_{visu}^* and Δt_{render} in relation to the number of vertices in a range from 0 up to 2 million vertices.

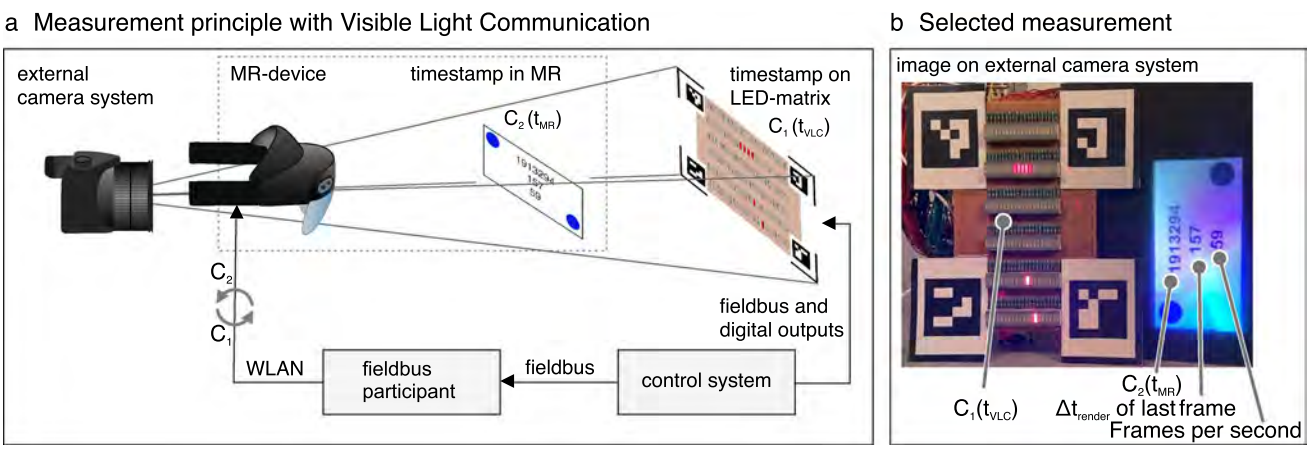

Fig. 9 Measurement method based on Visible Light Communication (VLC)

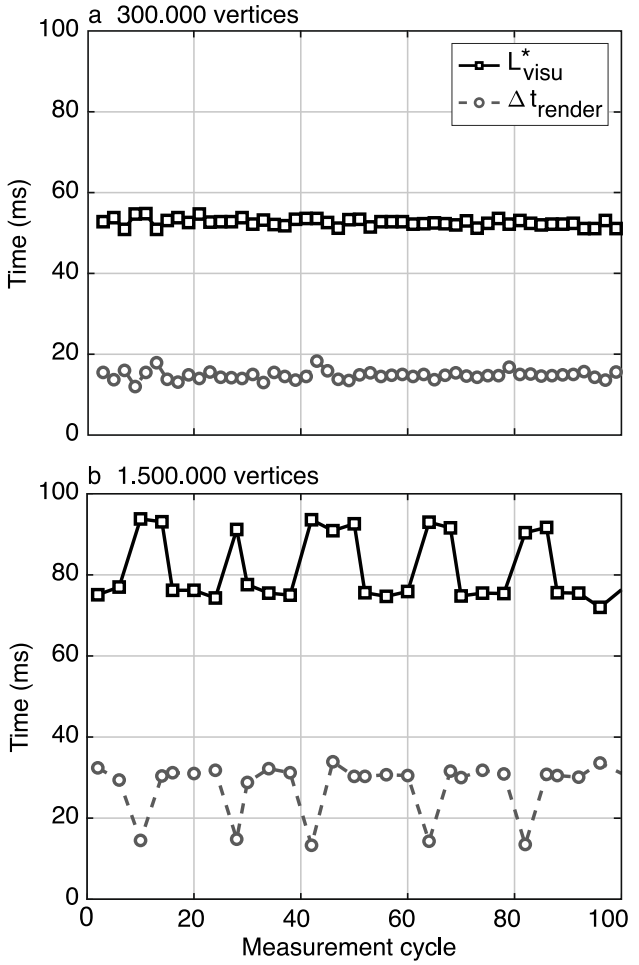

Fig. 10 Time behaviour of Δt_{render} and L^*_{visu} for different count of rendered vertices on a Microsoft HoloLens 2 Industrial with Unity engine

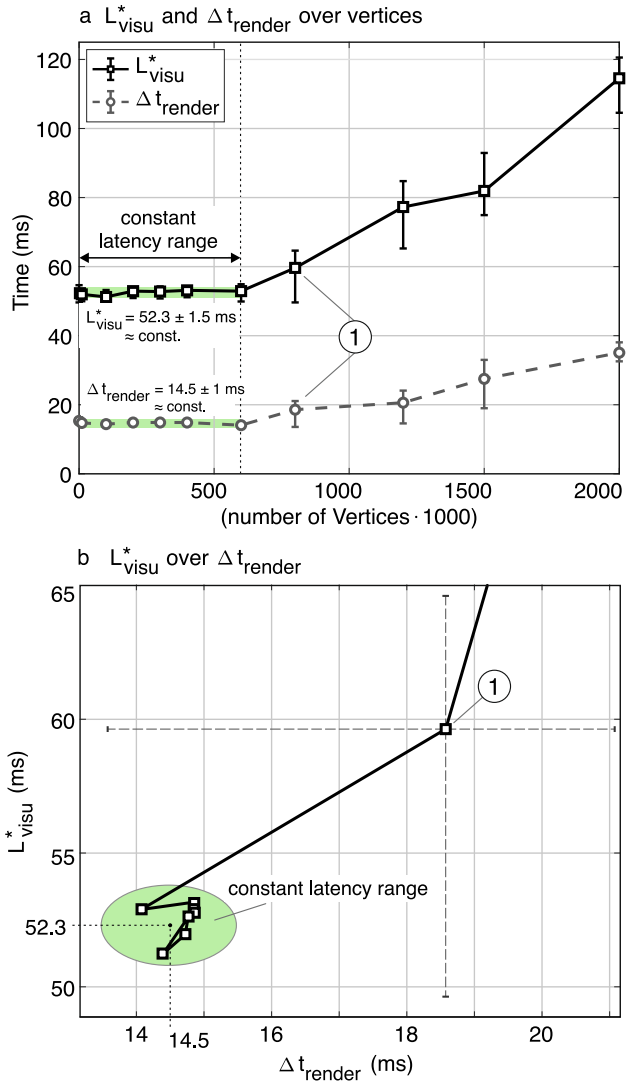

Fig. 11 **a** L^*_{visu} and Δt_{render} dependency of number of rendered vertices and **b** L^*_{visu} over Δt_{render}

Up to a certain vertex count (see constant latency range), both Δt_{render} and L^*_{visu} can be assumed to be approximately constant. In addition to the vertex count, there are further parameters influencing the load on the rendering pipeline, such as the complexity of the scene graph including the number of objects, number of light sources or the complexity of materials (e.g. surface effects like reflection). Depending on the composition of the models, the vertex count limit shown can therefore shift. In order to be independent of the absolute count of vertices, the constant latency range can be described in dependence of L^*_{visu} over Δt_{render} as displayed in Fig. 11b. Render times outside the constant latency range (see point 1) show large variances and are therefore less suitable for compensation. By measuring Δt_{render} in several visualisation cycles on the MR-device, it can be checked whether the current operating point lies within the constant latency range. The L^*_{visu} inside the constant latency range can therefore be stored on the MR-device to be compensated in the prediction method.

This Sect. 2.1 presented a software-based synchronisation method that allows the estimation of $L^*_{app,k}$ by synchronising the MR-device to the stable time base of the control system. Furthermore a solution for estimation of $L^*_{visu,k}$ based on the load of the rendering pipeline was introduced. Both methods combined enable the online estimation of L^*_k according to Eq. 1 and therefore provide the foundation for the prediction method.

2.2 Prediction

After estimating the End-to-End latency L^*_k, the position data of the control system arriving at the MR-device can be used

in a prediction function to obtain a new forecast s^* in each visualisation cycle. For application- and model-independent prediction with high computational efficiency a data-based prediction by using section-wise defined polynomials is proposed in this work. Based on the velocity-, acceleration- and jerk-limited position profile (so-called 7-phase profile) as a widespread interpolation strategy for servo-drives of production systems, linear (polynomial degree $n = 1$), quadratic ($n = 2$) and cubic ($n = 3$) polynomials for prediction of the position signal are applied in the following. Including the h past sampling points $[s(t_k), ..., s(t_{k-h})]$ from the ground truth of the control system, a linear system of equations (Eq. 8) can be set up to calculate the polynomial coefficients $[\beta_0, ..., \beta_n]$ with the given sampling times $[t_k, ..., t_{k-h}]$.

$$\begin{bmatrix} 1 & t_k & t_k^2 & \cdots & t_k^n \\ 1 & t_{k-1} & t_{k-1}^2 & \cdots & t_{k-1}^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_{k-h} & t_{k-h}^2 & \cdots & t_{k-h}^n \end{bmatrix} \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_n \end{bmatrix} = \begin{bmatrix} s(t_k) \\ s(t_{k-1}) \\ \vdots \\ s(t_{k-h}) \end{bmatrix} \tag{8}$$

The linear system of equations is solved in each sampling step to determine the parameters β_i of the section-wise defined polynomials. Within each section the polynomial function is evaluated in the visualisation cycle on the MR-device.

For $h = n$ the equation system of Eq. 8 has a unique solution and the polynomials interpolate the sampling points $[s(t_k), ..., s(t_{k-n})]$.

For $h > n$ the equation system of Eq. 8 is overdetermined and the polynomial parameters are calculated by polynomial regression algorithms like the ordinary least squares approach. In this case, polynomial approximation for $[s(t_k), ..., s(t_{k-h})]$ takes place.

To analyse the parameters influencing the prediction quality of the section-wise defined polynomials, a simulative analysis was carried out with a 7-phase position profile as a test signal (see Fig. 12).

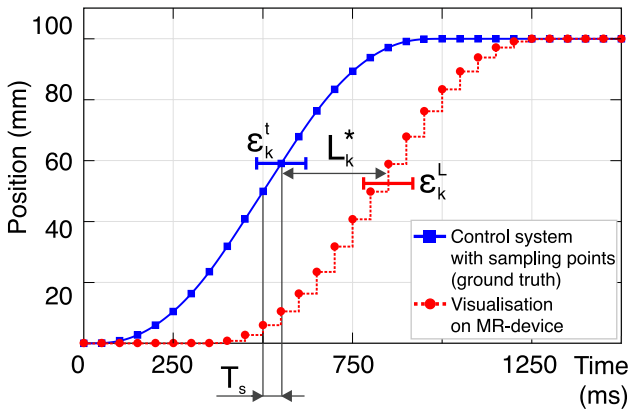

Fig. 12 7-phase position signal for simulative analysis of the polynomial prediction and the influencing parameters

The diagram illustrates the different influencing parameters: the sampling interval T_s, the magnitude of the End-to-End latency L_k^*, the error ϵ_k^L of the estimation L_k^* and the error ϵ_k^t of the assumed sampling time t_k. An error ϵ_k^t occurs, if the timestamp t_k e.g. has to be set outside the real-time basis of the control system or is inaccurate due to dead time effects in the acquisition of sensor data. The influences of T_s, L_k^*, ϵ_k^L, ϵ_k^t on the error e^* for interpolating and approximating polynomials are discussed in the following. For analysing the influence of one specific parameter, the other parameters were set with the boundary conditions $T_s = 50$ ms, $L_k^* = 100$ ms, $\epsilon_k^L = 0$ ms, $\epsilon_k^t = 0$ ms. The visualisation rate was set to 60 frames per second for all simulations.

2.2.1 Interpolating polynomials ($h = n$)

Figure 13 presents the root mean square of the positioning error e^* with interpolating polynomials in dependence of T_s, L_k^*, ϵ_k^L and ϵ_k^t.

It can be seen in Fig. 13a that the quadratic polynomial provides the best prediction in large parts of the value range of sampling interval T_s. Compared to the quadratic polynomial, the cubic polynomial is more sensitive to an increase in T_s. The quadratic and cubic polynomial provide the most accurate prediction for variation of L_k^* and ϵ_k^L as shown in Fig. 13b, c. Figure 13d indicates the sensitivity of the quadratic and cubic polynomials to an error ϵ_k^t as the polynomials react strongly to fluctuations in t_k.

In summary, the quadratic polynomial shows the best simulation results except the influence of ϵ_k^t - hence, the next section presents the simulation results for approximating quadratic polynomicals with $h = 3$ and $h = 4$ compared to the interpolating quadratic polynomial.

2.2.2 Approximating polynomials ($h > n$)

Figure 14 shows the simulation results for quadratic ($n = 2$) approximating polynomials for $h = 3$ and $h = 4$. For better comparison, the interpolating polynomial with $h = 2$ is also drawn into the diagrams.

Figure 14a, b and c depicts the decrease in prediction accuracy in dependence of T_s and L_k^* and ϵ_k^L for the approximating polynomials compared to the interpolating polynomial in the case of exactly known t_k ($\epsilon_k^t = 0$). While the integration of further sampling points negatively influences the prediction in the shown cases, it greatly increases the robustness to ϵ_k^t as recognisable in Fig. 14d.

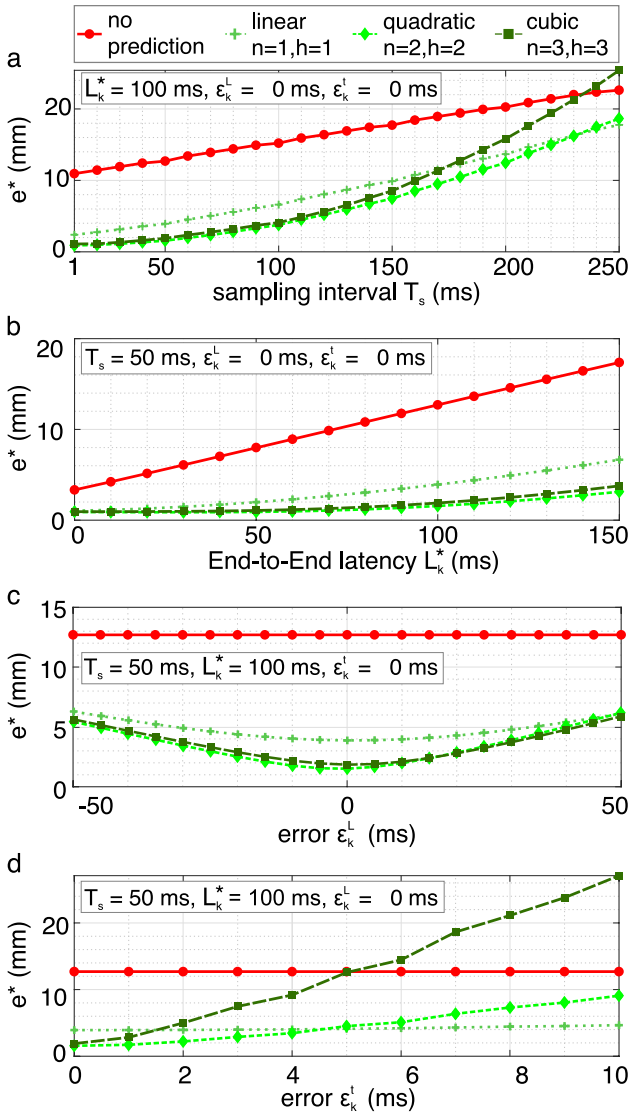

Fig. 13 Simulated root mean square position error e^* with interpolating section-wise polynomial prediction in dependence of T_s, L_k^*, ϵ_k^L and ϵ_k^t

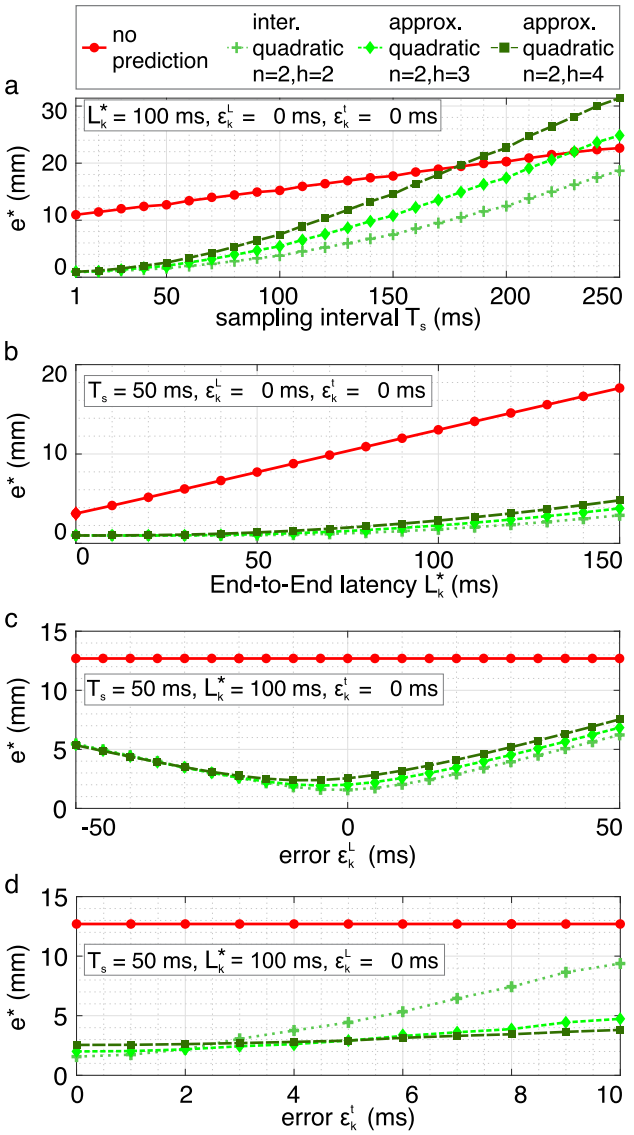

Fig. 14 Simulated root mean square position error e^* with approximating section-wise polynomial prediction in dependence of T_s, L_k^*, ϵ_k^L and ϵ_k^t

In this Sect. 2.2, the prediction with interpolating and approximating polynomials was presented and analysed. The simulative analysis using a 7-phase position profile provided the assessment of the influencing factors T_s, L_k^*, ϵ_k^L, ϵ_k^t and demonstrated that the error e^* is significantly reduced with the presented prediction approach in practically relevant parameter combinations. Depending on boundary conditions for T_s, L_k^*, ϵ_k^L and ϵ_k^t, an application-specific choice between interpolating and approximating polynomials can be made.

3 Realisation and proof of concept

The presented compensation method was implemented and evaluated in a MRiLS, which extends a real robot kinematics with a virtual gripper. The architecture of the realisation example is shown in Fig. 15.

The industrial control system (Beckhoff industrial PC with TwinCAT software) operates the robot kinematics of a KUKA youBot using the real-time EtherCAT fieldbus protocol. To avoid the need of modifications to the control system, an additional fieldbus participant (Beckhoff industrial PC with TwinCAT software) is connected to the fieldbus. For external systems the fieldbus participant provides a synchronisation interface with an ADS (Automation Device

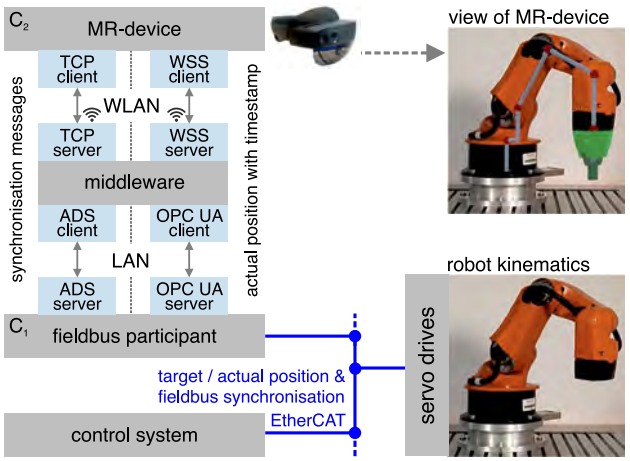

Fig. 15 Architecture of the MRiLS realisation example

Specification) server and an interface for the actual position with an OPC UA (Open Platform Communications Unified Architecture) server. In order to achieve MR-device independent communication and to avoid high communication load in the fieldbus participant due to multiple connected MR-devices, a middleware connects as a client to the ADS and OPC UA server and bundles the communication with intermediate processing in a Digital-Twin-as-a-Service concept [29]. The middleware implements the synchronisation method from Sect. 2.1.1 in a TCP server (Transmission Control Protocol) as a lightweight and efficient interface and offers the position data with a WSS (WebSockets) server over the wireless network to the MR-devices. For the proof of concept, the optical see-through Augmented Reality device Microsoft HoloLens 2 Industrial is used as MR-device. The prediction of the position data including the findings from 2.1.2 and the kinematic forward transformation for positioning the virtual gripper takes place on the MR-device. Since in this realisation example the timestamp t_k is set directly in the control system and therefore $\epsilon_k^t = 0$, the prediction is performed with a quadratic interpolating polynomial ($n = 2, h = 2$) based on the results of the simulative analysis in Sect. 2.2. The spatial superimposition between the base coordinate system of the real robot kinematics and the virtual model is achieved by marker-based tracking combined with time of flight tracking of the MR-device.

Figure 16 presents a measured position profil of the first robot axis (blue line), the sampled and delayed position signal (red line) and the predicted signal (green line). The position value describes the distance on the circular path that the end effector passes during the rotation of the first robot axis. The ground truth was recorded directly on the control system and the values of the delayed and sampled signal and the predicted signal were measured using the VLC method as described in Sect. 2.1.2. The diagram in Fig. 16 includes three selected points in time A, B and C, for which the corresponding robot poses visualised on the MR-device with and without compensation are presented in Fig. 17. The error reduction with compensation is clearly visible to the user of the MR-device. The diagram shows that especially non-linear profil phases are challenging for the prediction method. Overall the proposed compensation enables the execution of MRiLS in lower error fields, because it reduces the effective latency without having to decrease the automation system's velocity or optimise the surrounding infrastructure. The visual improvement is as well supported by Fig. 18, which shows the uncompensated and compensated path in the error field diagram for $t < 1500$ ms. In the scenario presented, the total End-to-End latency L_k^* could be

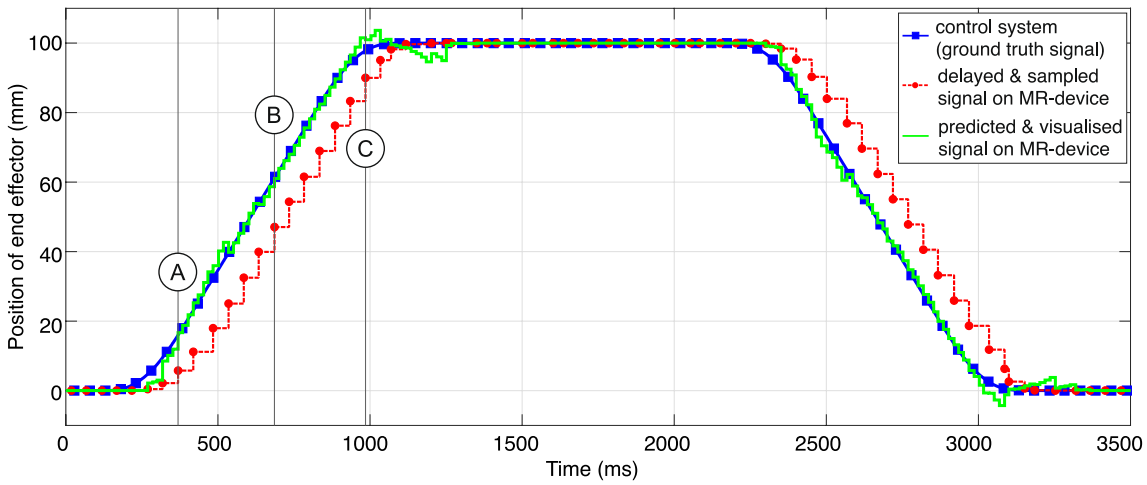

Fig. 16 Position profile of the end effector of the Kuka youBot with ground truth signal of the control system and uncompensated and compensated signal on the MR-device

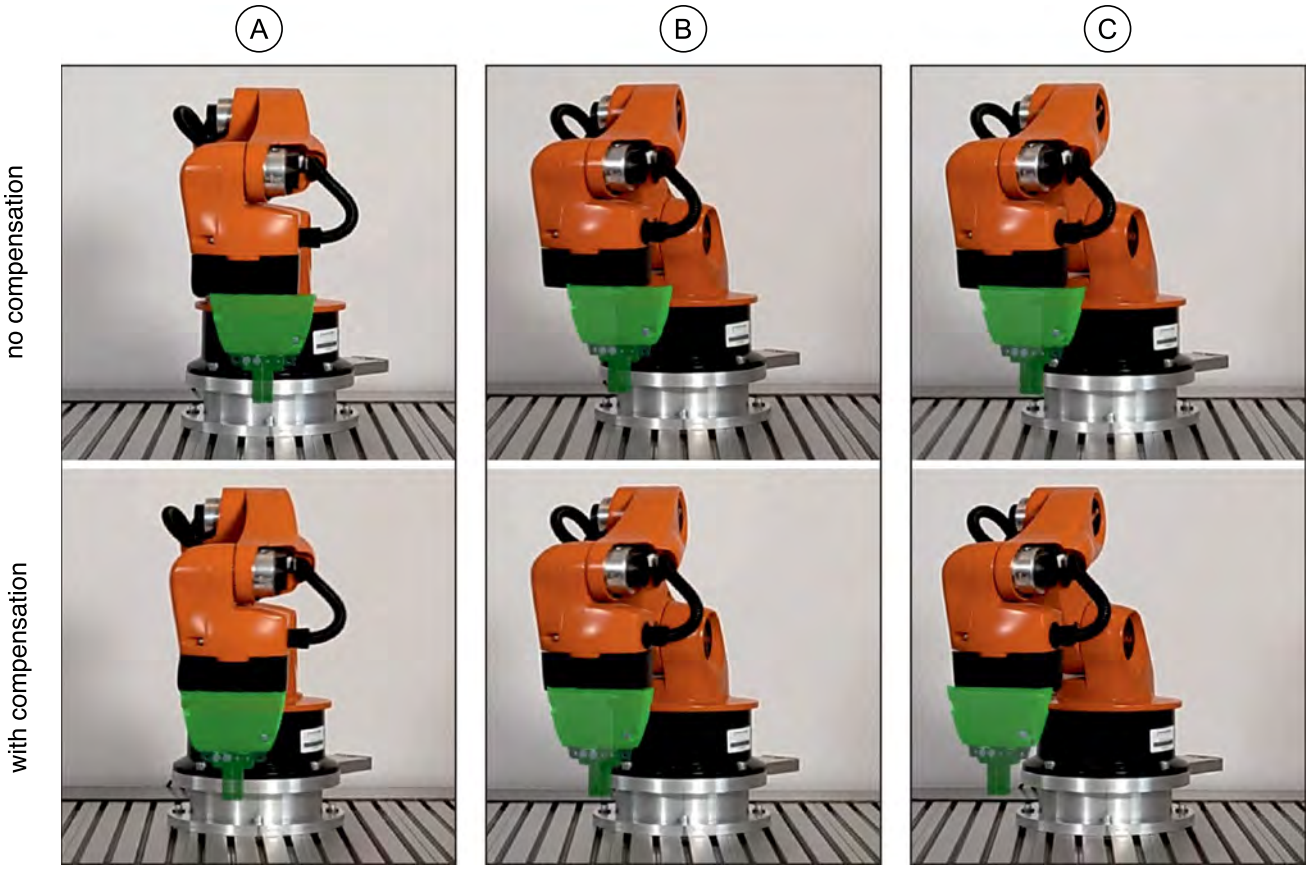

Fig. 17 Comparison of MRiLS scenario with and without compensation (captured with Microsoft HoloLens 2 Industrial)

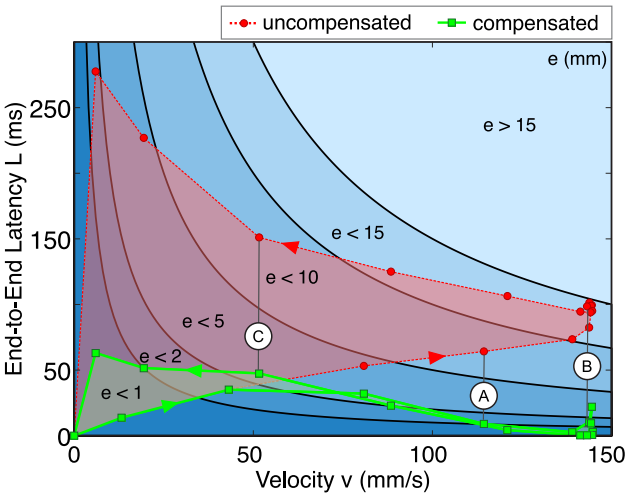

Fig. 18 Uncompensated and compensated MRiLS operating points

determined with an error of $\epsilon_k^L < 4$ ms and the positioning error could be reduced by approximately 80% ($\frac{e^*}{e} \approx 20\%$). The compensation method can be transferred to further optical see-through MR-devices as well as control systems and therefore to industrial MRiLS scenarios.

4 Conclusion

The paper presented a compensation method to reduce positioning errors of real-data driven virtual components in Mixed-Reality-in-the-Loop Simulations. The method synchronises the MR-device and the industrial control system with a software-based approach by estimating the End-to-End latency without necessary additional hardware. The core of the method is clock synchronisation of the MR-device to the stable time base of the control system to determine the application latency as well as the estimation of the visualisation latency of the rendering pipeline. Following the estimation of the End-to-End latency, section-wise defined interpolating and approximating polynomials were introduced as data-based methods for position prediction and simulativly analysed. The overall method was successfully evaluated on a realisation example by reduction of the occurring positioning error of a virtual gripper linked to a real robot kinematics. The compensation method and the

results of the realisation example can be applied to numerous industrial applications of MRiLS.

In further scientific work, data-based prediction methods will be investigated that have extended adaptation capabilities (e.g., Gaussian Process Regression) and the applicability of the compensation method on additional Mixed Reality devices will be studied.

Acknowledgements The authors would like to thank the Federal Ministry of Education and Research (BMBF) for supporting this work by funding of the project MRiLS (16SV8348) and the Baden-Wuerttemberg Ministry of Science, Research and the Arts for funding of the cooperative doctoral programme PROMISE 4.0.

Author contributions All authors contributed to the conception and design of this work. The first draft of the manuscript was written by Marc Schnierle and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. VDI/VDE. Virtual commissioning-part 1: model types and glossary. Guideline VDI/VDE 3963 (2016)
2. VDMA, e.V. Virtual Commissioning Guideline-Recommendations for action for economic entry (2020)
3. Milgram P, Takemura H, Utsumi A, Kishino F (1995) Augmented reality: a class of displays on the reality-virtuality continuum. Telemanipulator Telepresence Technologies 2351:282–292. https://doi.org/10.1117/12.197321
4. Suzuki R, Karim A, Xia T, Hedayati H, Marquardt N (2022) Augmented reality and robotics: a survey and taxonomy for AR-enhanced human-robot interaction and robotic. Interfaces. https://doi.org/10.1145/3491102.3517719
5. VDMA e.V (2018) VDMA IT-Report 2018 to 2020. Survey
6. Bauer M (2007) Tracking errors in augmented reality. Ph.D. thesis, Technical University Munich, Germany. http://mediatum2.ub.tum.de/doc/618255/document.pdf
7. Pentenrieder K, Bade C, Doil F, Meier P (2007) Augmented reality-based factory planning-an application tailored to industrial needs. test 31–42 . https://doi.org/10.1109/ISMAR.2007.4538822
8. Schlegel M (2011) Zeitkalibrierung in Augmented-reality-Anwendungen. Ph.D. thesis, Technical University Munich. https://nbn-resolving.org/urn:nbn:de:bvb:91-diss-20121219-1078613-1-4
9. Hussain SA, Parfitt S (2014) White paper-DISPLAYPORT ADAPTIVE-SYNC. https://www.vesa.org/wp-content/uploads/2014/07/VESA-Adaptive-Sync-Whitepaper-140620.pdf
10. Caserman P, Martinussen M, Göbel S (2019) Effects of end-to-end latency on user experience and performance in immersive virtual reality applications. Robotcup 57–69. https://doi.org/10.1007/978-3-030-34644-7_5
11. Malý I, Sedláček D, Leitão P (2016) Augmented reality experiments with industrial robot in industry 4.0 environment. 2016 IEEE 14th International Conference on Industrial Informatics (INDIN) 176–181. https://doi.org/10.1109/INDIN.2016.7819154
12. Liu C, Cao S, Tse W, Xu X (2017) Augmented reality-assisted intelligent window for cyber-physical machine tools. J Manuf Sys 44:280–286. https://doi.org/10.1016/j.jmsy.2017.04.008
13. Farajiparvar P, Ying H, Pandya A (2020) A brief survey of telerobotic time delay mitigation. Front Robotics AI 7 https://doi.org/10.3389/frobt.2020.578805
14. Nilsson J (1998) Real-time control systems with delays. PhD Thesis, Department of Automatic Control, Lund Institute of Technology, Lund. https://lucris.lub.lu.se/ws/portalfiles/portal/4419523/8840255.pdf
15. Bernier YW (2003) Latency compensating methods in client/server in-game protocol design and optimization. Protocol Design and Optimization, Valve
16. Azuma RT (1995) Predictive tracking for augmented reality. Ph.D. thesis, UNC-Chapel Hill. https://ronaldazuma.com/papers/dissertation.pdf
17. Ushirobira R, Efimov D, Casiez G, Roussel N, Perruquetti W (2016) A forecasting algorithm for latency compensation in indirect human-computer interactions. 2016 European Control Conference (ECC) 1081–1086. https://doi.org/10.1109/ECC.2016.7810433
18. Steed A (2008) A simple method for estimating the latency of interactive, real-time graphics simulations 123–129. https://doi.org/10.1145/1450579.1450606
19. Swindells C, Dill JC, Booth KS (2000) System lag tests for augmented and virtual environments. In: Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology 161–170. https://doi.org/10.1145/354401.354444
20. Billeter M, Röthlin G, Wezel J, Iwai D, Grundhöfer A (2016) A LED-based IR/RGB end-to-end latency measurement device. IEEE Int Symposium Mixed Augmented Reality (ISMAR-Adjunct) 184–188. https://doi.org/10.1109/ISMAR-Adjunct.2016.0072
21. Sielhorst T, Sa W, Kamen A, Sauer F, Navab N (2007) Measurement of absolute latency for video see through augmented reality. 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR 215–220. https://doi.org/10.1109/ISMAR.2007.4538850
22. Stauffert J-P, Niebling F, Latoschik ME (2020) Latency and cybersickness: impact, causes, and measures: a review. Fronti Virtual Reality. https://doi.org/10.3389/frvir.2020.582204
23. Feldstein IT, Ellis SR (2021) A simple video-based technique for measuring latency in virtual reality or teleoperation. IEEE Trans Visualization Computer Graphics 27(9):3611–3625. https://doi.org/10.1109/TVCG.2020.2980527

24. Puttnies H, Schweissguth E, Timmermann D, Schacht J (2019) Clock synchronization using linear programming, multicasts, and temperature compensation. 2019 IEEE Global Communications Conference (GLOBECOM) 1–6. https://doi.org/10.1109/GLOBECOM38437.2019.9013260
25. Gore R. N, Lisova E, Åkerberg J, Björkman M (2021) CoSiWiNeT: a clock synchronization algorithm for wide area IIoT network. Appl Sci 11 (24). https://www.mdpi.com/2076-3417/11/24/11985. https://doi.org/10.3390/app112411985
26. Cristian F (1989) Probabilistic clock synchronization. Distrib Computing 3. https://doi.org/10.1007/BF01784024
27. Gore RN (2021) Investigating software-based clock synchronization for industrial networks. Master's thesis, Mälardalen University
28. Masson L, Cao F, Viard C, Guichard F (2014) Device and algorithms for camera timing evaluation. Image Quality Syst Performance XI SPIE Proceed. 9016. https://doi.org/10.1117/12.2042161
29. Schnierle M, Röck S (2018) Platform for the Mixed-Reality-in-the-Loop-Simulation-A contribution to Mixed-Reality-in-the-Loop-Simulation as an extension of the X-in-the-Loop-Methods. wt Werkstattstechnik online. https://doi.org/10.37544/1436-4980-2018-09-59