Adaptive model predictive control for actuation dynamics compensation in real-time hybrid simulation

N. Tsokanas[a], R. Pastorino[b], B. Stojadinović[a]

[a] ETH Zurich, IBK, D-BAUG, Stefano-Franscini Platz 5, 8093 Zurich, Switzerland.
[b] Siemens Industry Software NV, Interleuvenlaan 68, 3001 Leuven, Belgium.

Abstract

Real-time hybrid simulation is an experimental method used to obtain the dynamic response of a system whose components consist of loading-rate-sensitive physical and numerical substructures. The coupling of these substructures is achieved by actuation systems, i.e., an arrangement of motors or actuators, which are responsible for continuously synchronizing the interfaces of the substructures and are commanded in closed-loop control setting. To ensure high fidelity of such hybrid simulations, performing them in real-time is necessary. However, real-time hybrid simulation poses challenges as the inherent dynamics of the actuation system introduce time delays, thus modifying the dynamic response of the investigated system and hence compromising the simulation's fidelity and trust in the obtained response quantities. Therefore, a reference tracking controller is required to adequately compensate for such time delays.

In this study, a novel tracking controller is proposed for dynamics compensation in real-time hybrid simulations. It is based on an adaptive model predictive control approach, a linear time-varying Kalman filter, and a real-time model identification algorithm. Within the latter, auto-regressive exogenous polynomial models are identified in real-time to estimate the changing plant dynamics and used to update the prediction model of the tracking controller. A parametric virtual real-time hybrid simulation case study is used to validate the performance and robustness of the proposed control scheme. Results demonstrate the effectiveness of the proposed controller for real-time

hybrid simulations.

Keywords: real-time hybrid simulation, adaptive model predictive control, Kalman filter, real-time model identification, dynamic response, actuation dynamics compensation.

1. Introduction

Hybrid simulation (HS), sometimes also called hardware-in-the-loop (HiL) or system-in-the-loop testing, is an experimental method used to obtain the dynamic response of a prototype system whose model consists of numerical (NS) and physical substructures (PS). The so-called hybrid model is obtained by the coupling of NS and PS. Such coupling is achieved using an efficient interface data exchange system and an actuation system, i.e., an arrangement of motors or actuators, which are responsible for continuously synchronizing the interfaces of the substructures and are commanded in a closed-loop control setting. HS is based on a step-by-step numerical solution of the equations governing the motion of the hybrid model [1]. In more detail, in each time step of HS, the NS generates a command that needs to be tracked by the PS to preserve synchronization of the substructures' boundary conditions. The dynamic response of the PS is then measured using appropriate sensors and fed back to the NS to establish the state of the hybrid model and advance the HS to the next time step.

HS is a desirable dynamic response testing technique since it combines the versatility and risk-free testing of numerical simulations along with the realism of experimental campaigns. However, ensuring high fidelity of hybrid dynamic response simulations of a loading-rate-sensitive prototype often necessitates performing HS in real-time. Real-time hybrid simulation (RTHS) poses several challenges. One challenge arises from complex NS because of the computational power required to compute the NS response on time, i.e., within the HS time step. For example, to adequately capture the dynamic response of high dimensional and/or nonlinear numerical models, it is common practice to decrease the time step of the NS simulation which increases the demanded computational power. Not completing the computations to establish the state of the NS on time introduces delays and risks distorting the time scale of HS. In such cases, model order reduction of the NS is a very effective countermeasure [2, 3].

Another challenge in RTHS originates from the PS, specifically from the actuation system, as its inherent dynamics introduce time delays that modify the dynamic response of the PS and hence compromise the fidelity of the entire HS. Addressing this challenge is the focus of this paper.

Various control schemes for dynamics compensation in RTHS exist in the literature. A selection of such schemes is presented below. An initial approach has been developed by Horiuchi using a polynomial extrapolation algorithm [4], which was afterwards extended into an adaptive scheme [5, 6]. Compensation for the phase shift of the actuation system, via phase-lead compensators, was also proposed by various authors [7, 8, 9]. Inverse compensation is another technique used in RTHS, in which an inverse model of the actuation system is employed as a feedforward compensator [10, 11]. In order to improve the robustness of RTHS in the presence of experimental uncertainties, adaptive compensation strategies were developed aiming at estimating in real-time the controller parameters [12, 13, 14, 15]. H_∞ loop shaping control schemes were also proposed as an alternative robust approach [16, 17]. Additionally, a feedforward-feedback control scheme based on linear-quadratic regulator (LQR) and Kalman filter was developed for controlling both single and multi-actuation setups [18, 19]. A model-based sliding mode control approach has also been developed for RTHS making use of a reduced plant [20]. Recently, a control scheme based on model predictive control (MPC) [21] was also presented.

As shown in [21], MPC is an effective control strategy for RTHS. A key advantage of MPC is its ability to perform optimization in real-time as well as satisfying constraints of the examined system, e.g., actuation system capacity limitations. However, in MPC the real-time optimization is performed based on predictions of the future dynamic response of the system under control. Therefore, the performance of the optimization depends on the accuracy of the predictions. Classical MPC utilizes a prediction model, e.g., a linear time-invariant (LTI) dynamic model, for these predictions. In case the dynamics of the examined system are highly nonlinear or linear with time-varying parameters, the predictions from a LTI model may not be accurate and thus the performance of MPC will be degraded. In this regard, an *adaptive MPC* is a suitable solution, since the LTI model used to predict the dynamic response of the controlled system is adapted during the simulation in order to capture the changing dynamics of the system under control (the co-called plant). Some of the common practices for model updating in MPC include real-time system identification and successive linearization [22, 23].

This study presents a novel tracking controller for dynamics compensation in RTHS extending the classical MPC of [21] to an adaptive scheme. It is based on the adaptive MPC approach, a linear time-varying Kalman filter, and a real-time model identification algorithm. Within the latter, auto-regressive exogenous (ARX) polynomial models are identified in real-time to estimate the changing plant dynamics and used to update the prediction model of the adaptive MPC. A parametric virtual RTHS case study is used to demonstrate and validate the performance and robustness of the proposed controller.

The paper is organized as follows. Section 2 describes the tracking controller. Section 3 introduces the virtual RTHS case study where the proposed controller is developed, validated and demonstrated. Section 4 discusses the obtained results and Section 5 presents the overall conclusions of this study.

2. The tracking controller

In this section, the individual components of the proposed tracking controller are described. As mentioned above, these correspond to the adaptive MPC, the linear time-varying Kalman filter and a real-time model identification algorithm used to update the prediction model of the MPC. Figure 1 illustrates the block diagram of the controller. The plant corresponds to the system under control. The latter is addressed in more detail in Section 3.

2.1. Adaptive model predictive control

In the MPC approach, the current control action is calculated by solving, in each control interval, a finite horizon optimization problem utilizing the current state of the plant as its initial conditions. The outcome of this optimization is an optimal control sequence. The first control action of this sequence is used in the current control interval [24, 25]. Control intervals are sets of consecutive time steps, obtained from a sampling

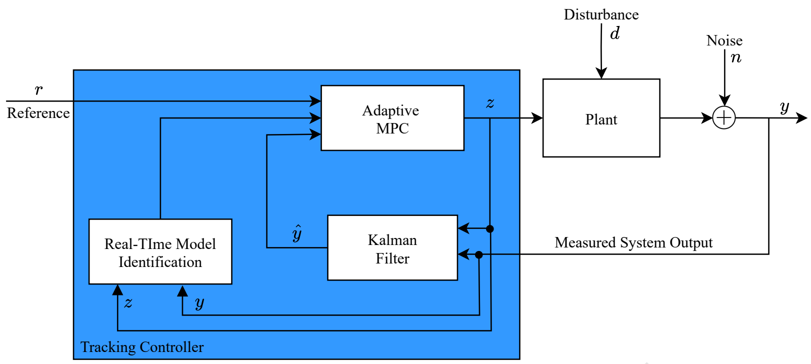

Figure 1. Architecture of the proposed tracking controller.

frequency that the MPC uses internally. Essentially, a MPC solves a finite horizon optimization problem similar to H_∞ and LQR control techniques. A fundamental difference nevertheless lies in the fact that MPC solves it in real-time; namely, at each control interval a new optimization problem in a receding horizon approach is solved. Therefore, the control actions are updated in real-time, while in classical optimal control, e.g., LQR, the optimization is solved offline and thus the control law is defined prior to the simulation and remains constant. Additionally, MPC offers the capability of satisfying hard constraints regarding the inputs, outputs and/or states of the plant while solving the optimization problem in real-time [26].

A MPC encompasses four elements, namely the prediction model, the performance index (or the cost function), the constraints, and a solver used to compute the control actions. The prediction model constitutes the core of the MPC as it is used to predict the future plant outputs based on information available up to the current time instant, e.g., previous and current inputs/outputs, and on the future control actions [27]. As mentioned above, the overall performance of MPC is directly influenced by the responses of the prediction model. Therefore, this model should capture the dynamics of the plant as accurately as possible. Although the MPC updates the control actions in real-time, the controller is still not adaptive in its structure, e.g., the prediction model remains the same. Updating the prediction model in real-time results in the so-called adaptive MPC.

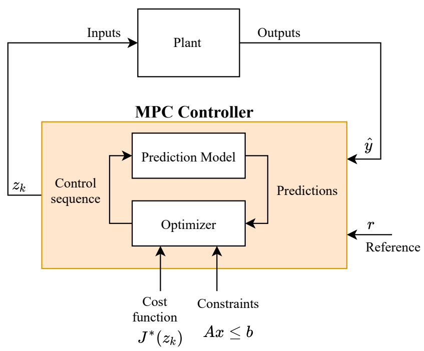

Figure 2. MPC structure of the tracking controller.

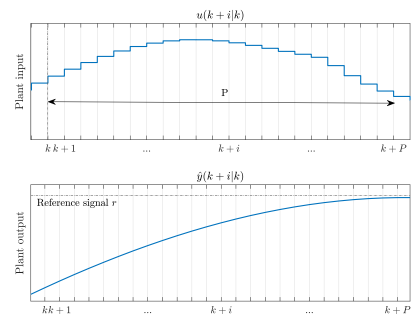

Figure 3. MPC methodology.

Figure 2 displays the structure of the MPC and Figure 3 illustrates its methodology. In more details, the prediction model predicts at each control interval k the next plant outputs $\hat{y}(k+i|k),\ i = 0, \ldots, P$ for every time instant i within a prediction horizon P. The notation $\hat{y}(k+i|k)$ correspond to the output at the time instant $k+i$ evaluated at k. The control sequence $z_k^T = [u(k|k)^T \ldots u(k+i|k)^T \ldots u(k+P-1|k)^T]$ includes the sequence of future control actions $u(k+i|k)$. At each k, a new control sequence is computed optimizing a new cost function expressed by:

$$\mathbf{J}^*(r_k, \hat{y}_k, z_k) = \sum_{j=1}^{n_y} \sum_{i=1}^{P} \left\{ w^{y_j} [r_j(k+i|k) - \hat{y}_j(k+i|k)] \right\}^2 + \sum_{j=1}^{n_u} \sum_{i=0}^{P-1} \left\{ w^{u_j} [u_j(k+i|k) - u_j(k+i-1|k)] \right\}^2, \tag{1}$$

where n_y is the number of plant outputs, n_u the number of plant inputs, $r_j(k+i|k)$ the reference value to be tracked at the i-th time instant step from the j-th plant output, $\hat{y}_j(k+i|k)$ the predicted value of the j-th plant output at the i-th time instant, $u_j(k+i|k)$ the j-th plant input at the i-th time instant, w^{y_j} the tuning weight of the j-th plant output and w^{u_j} the tuning weight of the j-th plant input. The P, w^{y_j} and w^{u_j} parameters of Eq. (1) are computed offline and remain constant in the work presented herein.

In addition, an output disturbance d and a measurement noise model n are used in the proposed tracking controller. They are implemented as additive to the plant outputs, representing potential disturbances and sensor noise that could be present in RTHS. The models of the disturbance d and the noise n are expressed in state-space, as described in Eqs. (2) and (3), respectively:

$$\begin{aligned} x_d(k+1) &= A_d x_d(k) + B_d u_d(k) \\ d(k) &= C_d x_d(k) + D_d u_d(k), \end{aligned} \tag{2}$$

$$\begin{aligned} x_n(k+1) &= A_n x_n(k) + B_n u_n(k) \\ n(k) &= C_n x_n(k) + D_n u_n(k), \end{aligned} \tag{3}$$

where A_d, B_d, C_d, D_d and A_n, B_n, C_n, D_n are matrices associated with the disturbance d and noise n respectively. The inputs to these state-space models, namely u_d for the disturbance and u_n for the noise, are Gaussian variables with zero mean and unit variance.

The control sequence z_k is computed at the beginning of each control interval in the optimizer (see Figure 2). The cost function of Eq. (1) is formulated into a Quadratic Programming (QP) problem [28, 29] and this is what the optimizer eventually solves. The QP formulation admits:

$$\min_x \left(\frac{1}{2} x^T H x + f^T x\right) \tag{4}$$

$$\text{subject to} \quad Ax \leq b, \tag{5}$$

where the $Ax \leq b$ inequality represents the applied constraints, x the solution vector and H its Hessian matrix. The vectors A and b define the constraints, and f is a vector computed using:

$$f = Kx(k|k)^T x(k|k) + Kr(k|k)^T r(k|k) + Ku(k|k-1)^T u(k|k-1), \tag{6}$$

where $r(k|k)$ is the reference signal at the current control interval, $u(k|k-1)$ is the applied control action in the previous control interval and K a weighting factor. For a more comprehensive review of QP, the reader is encouraged to consult [30, 31].

In the proposed tracking controller, the QP problem of Eqs. (4) and (5) is solved employing an active-set solver which applies the KWIK algorithm [32]. The latter is a built-in QP solver from the Model Predictive Toolbox of MATLAB, used in this study to compute the control law sequence [33].

The utilized adaptive MPC algorithm is summarized as follows:

1. **Prediction step:** Considering a discrete time multiple-input-multiple-output (MIMO) system that represents a linearized model of the plant, the prediction model in state-space formulation at the control interval k follows:

$$\begin{aligned} x_p(k+1) &= A_p^k x_p(k) + B_p^k u(k) \\ y_p(k) &= C_p^k x_p(k) + D_p^k u(k), \end{aligned} \tag{7}$$

   where A_p^k, B_p^k, C_p^k and D_p^k are matrices associated with the prediction model at the control interval k, representing the plant dynamics. The disturbance and noise models of Eqs. (2) and (3) respectively, are assumed to be additive to the measured plant output y signal in Figure 1, and hence the latter admits: $y(k) = y_p(k) + d(k) + n(k)$.
2. **Identification step:** At each control interval k, an ARX polynomial model of the plant is identified and used to adapt the prediction model of MPC, i.e., the matrices A_p^k, B_p^k, C_p^k and D_p^k, (see Section 2.3 for a more detail description of the algorithm).
3. **Update step:** At each control interval k, the L and M vectors of the Kalman filter are updated to be consistent with the adapted prediction model (see Section 2.2).
4. **Optimization step:** Using the current state estimates and predicted plant outputs from the Kalman filter (Eqs. (16) and (19)), the MPC solves the optimization problem described by Eq. (1) for the current control interval k:

$$\min_{z_k} \mathbf{J}^*(r_k, \hat{y}_k, z_k) \tag{8}$$

$$\text{subject to} \quad u_{j_{min}} \leq u_j \leq u_{j_{max}}, \tag{9}$$

where the above constraints represent safety limits regarding input signals to the actuation system.

5. **Control step:** The control sequence $z_k^T = [u(k|k)^T \dots u(k+i|k)^T \dots u(k+P-1|k)^T]$ for the current control interval is computed and its first control action is applied to the plant.
6. Steps 1-5 are repeated till completion of the RTHS.

If the control interval coincides with the sampling frequency of RTHS, then at the next time step of RTHS a new optimization will be performed and a new control sequence will be computed taking into account the new information available, e.g., disturbance, noise, introduced in that time step. When the control interval is larger than the RTHS time step, i.e., includes several time steps, then the first control action of step 5 will be applied as long as the ongoing control interval lasts. Therefore, reducing the time steps in the control interval would result in a more robust MPC. However, the smaller the control interval the more the optimizations that must be performed, thus increasing the computational load. In this regard, a higher dimensional prediction model, that could potentially capture more accurately the plant dynamics, would increase the performance of MPC as the predictions would be more accurate. Nevertheless, a higher dimensional model would require additional computational power. Consequently, taken the aforementioned points into consideration, there exist a trade-off between controller accuracy and its computational performance and simulation model accuracy and its computational performance.

2.2. Linear time-varying Kalman filter

As mentioned above, the performance of the MPC depends on the accuracy of the predicted plant outputs. In order to provide the MPC with low-noise plant outputs, a Kalman filter is used. Furthermore, the Kalman filter provides estimates of the plant outputs when there are none available from the plant sensors. Since the prediction model of the MPC is adapted in real-time, the gain vectors of the Kalman filter also need to be adapted in real-time, resulting in a linear time-varying Kalman filter. Considering Eqs. (7), (2) and (3), the state-space formulation including the plant dynamics, the disturbance and noise model follows:

$$
\begin{aligned}
x_c(k+1) &= A_c^k x_c(k) + B_c^k u_c(k) \\
\hat{y}(k) &= C_c^k x_c(k) + D_c^k u_c(k),
\end{aligned}
\tag{10}
$$

where

$$
\begin{aligned}
A_c^k &= \begin{bmatrix} A_p^k & 0 & 0 \\ 0 & A_d & 0 \\ 0 & 0 & A_n \end{bmatrix}, \quad B_c^k = \begin{bmatrix} B_p^k & 0 & 0 \\ 0 & B_d & 0 \\ 0 & 0 & B_n \end{bmatrix}, \\
C_c^k &= \begin{bmatrix} C_p^k & C_d & C_n \end{bmatrix}, \quad D_c^k = \begin{bmatrix} D_p^k & D_d & D_n \end{bmatrix}, \\
x_c^T &= \begin{bmatrix} x_p^T & x_d^T & x_n^T \end{bmatrix} \text{ and } u_c^T = \begin{bmatrix} u^T & u_d^T & u_n^T \end{bmatrix}.
\end{aligned}
\tag{11}
$$

A_p^k, B_p^k, C_p^k and D_p^k matrices are updated in real-time while A_d, B_d, C_d and D_d and A_n, B_n, C_n and D_n of Eqs. (2) and (3) are considered constant during RTHS. The x_c, u_c vectors contain the states and inputs of the plant p, the disturbance d and noise n, respectively. The weighting coefficients Q, R, N of the Kalman filter are constant during RTHS and are computed from the following expectations:

$$
Q = \mathbb{E}\big[dd^T\big], \quad R = \mathbb{E}\big[nn^T\big], \quad N = \mathbb{E}\big[dn^T\big]. \tag{12}
$$

Figure 1 displays the interconnection of the Kalman filter within the proposed tracking controller. In more detail, at each control interval k, the x_c states are estimated as follows:

1. **Gain computations:** The gain vectors L, M of the Kalman filter are updated to be consistent with the adapted prediction model:

$$
L^k = \Big(A_c^k P^{k|k-1} C_c^{k^T} + N\Big)\Big(C_c^k P^{k|k-1} C_c^{k^T} + R\Big)^{-1}, \tag{13}
$$

$$
M^k = P^{k|k-1} C_c^{k^T} \Big(C_c^k P^{k|k-1} C_c^{k^T} + R\Big)^{-1}, \tag{14}
$$

$$
P^{k+1|k} = A_c^k P^{k|k-1} A_c^{k^T} - \Big(A_c^k P^{k|k-1} C_c^{k^T} + N\Big) L^{k^T} + Q, \tag{15}
$$

where L^k, M^k denote the L, M gain vectors at the control interval k and $P^{k+1|k}$ the state estimate error covariance matrix at $k+1$, calculated with information obtained from the k control interval.

2. **Measurement update step:** The current state estimate $x_c(k|k)$ is adjusted with the latest measurements:

$$x_c(k|k) = x_c(k|k-1) + M^k \left[y(k) - C_c^k x_c(k|k-1) \right], \tag{16}$$

where $x_c(k|k-1)$ denotes the state estimate from the $k-1$ control interval.

3. **Prediction step:** The state for the next, $k+1$, control interval is estimated as:

$$x_c(k+1|k) = A_c^k x_c(k|k-1) + B_{c1}^k u(k) + L^k \left[y(k) - C_c^k x_c(k|k-1) \right], \tag{17}$$

where $u(k)$ the control action used from $(k-1)$ till k, $y(k)$ the plant output measured at k and B_{c1}^k the first column of B_c^k.

Once the current state of the plant is estimated, the controller predicts plant outputs for the entire prediction horizon of the current control interval.

1. For any successive time instant of the prediction horizon, $i = 1, \ldots, P$, the predicted plant states are obtained by:

$$x_c(k+i|k) = A_c^k x_c(k+i-1|k) + B_{c1}^k u(k+i-1|k). \tag{18}$$

2. The predicted plant output for $i = 1, \ldots, P$ is obtained by:

$$\hat{y}(k+i|k) = C_c^k x_c(k+i|k). \tag{19}$$

2.3. Real-time model identification algorithm

At each control interval k, a real-time model identification algorithm is employed and ARX polynomial models [34] of the plant are identified and then used to update the prediction model of the MPC. The formulation of ARX polynomial models is the following:

$$A(q)y(t) = B(q)u(t) + e(t), \tag{20}$$

where q is a time shift operator, $A(q) = 1 + \alpha_1 q^{-1} + \alpha_2 q^{-2} + \cdots + \alpha_{na} q^{-na}$, $B(q) = b_1 + b_2 q^{-1} + b_3 q^{-2} + \cdots + b_{nb} q^{-(nb-1)}$ and $u(t)$, $y(t)$ and $e(t)$ the inputs, outputs and error respectively.

The model identification is based on the recursive infinite-history algorithm [35] following:

$$\theta(t) = \theta(t-1) + K(t)\big(y(t) - y_{\text{pred}}(t|\theta)\big), \tag{21}$$

where

$$\theta(t) = \begin{bmatrix} \alpha_1(t) & \alpha_2(t) & \dots & \alpha_{na}(t) & b_1(t) & \dots & b_{bn}(t) \end{bmatrix}, \tag{22}$$

corresponds to the identified parameters at time t, $y(t)$ the measured plant output at t, $y_{\text{pred}}(t|\theta) = \psi^T(t)\theta(t-1)$ the prediction of $y(t)$ accounting for measurements up to $t-1$ and $K(t) = Q(t)\psi(t)$, where $\psi(t)$ is the gradient of $y_{\text{pred}}(t|\theta)$ and $Q(t)$ admits:

$$Q(t) = \frac{P(t-1)}{1 + \psi^T(t)P(t-1)\psi(t)}, \tag{23}$$

where

$$P(t) = P(t-1) + 1 - \frac{P(t-1)\psi(t)\psi^T(t)P(t-1)}{1 + \psi^T(t)P(t-1)\psi(t)}. \tag{24}$$

$P(t)$ represents the covariance of parameter identification error with $P(0) = 1$.

At each control interval, the parameters $\theta(t)$ are identified and the $A(q)$, $B(q)$ polynomial are converted to state-space formulation resulting in the A_p^k, B_p^k, C_p^k and D_p^k matrices used in Eq. (7).

As mentioned earlier, a Kalman filter is used in the control scheme for state estimation and the recursive infinite-history algorithm for real-time parameter identification of the prediction model. As displayed in Figure 1, these are implemented as two separate blocks, in a decentralized approach. This approach offers the flexibility of the independent design of the state and the parameter estimation algorithms. Consequently, a variety of combinations of different real-time model identification algorithms can easily be tested. A single joint algorithm, a centralized approach, featuring both state and parameter estimation could also be implemented, but this is not examined in this study.

3. Case study

To validate the performance and robustness of the proposed AMPC tracking controller, a parametric case study is utilized. To facilitate the design and testing of the controller, the hybrid model is simulated virtually in the sense that all of its substructures are simulated numerically and thus virtual PS (vPS) are employed instead of physical specimens. In this section, the utilized performance metrics are defined first, then the problem formulation of the case study is introduced and, finally, the design properties of the tracking controller are addressed.

3.1. Performance metrics

To assess the performance of the controller, three metrics are defined. These are:

1. Tracking time-delay, defined by the correlation between the reference signal r and plant output y:

$$J_1 = \left(\arg\max_{k} \left(\mathrm{Corr}(r(i), y(i-k)) \right) \right) f_{\mathrm{RTHS}} \quad \text{[msec]}, \tag{25}$$

   where $f_{\mathrm{RTHS}} = 10$ kHz is the sampling frequency of RTHS.

2. Normalized Root Mean Square Error (NRMSE) of the tracking error, denoted as:

$$J_2 = \sqrt{\frac{\sum_{i=1}^{N} \left[y(i) - r(i) \right]^2}{\sum_{i=1}^{N} \left[r(i) \right]^2}} \times 100 \quad [\%]. \tag{26}$$

3. Peak Tracking Error (PTE), denoted as:

$$J_3 = \frac{\max |y(i) - r(i)|}{\max |r(i)|} \times 100 \quad [\%]. \tag{27}$$

Above, J_1 represents the maximum cross-correlation between the reference and the measured plant output. It is used to quantify how these two signals differ in time. Specifically, cross-correlation represents the number of time steps that the measured plant output should be shifted to match the reference. For $J_1 > 0$ the measured plant output has a delay with respect to the reference, i.e., there is a tracking delay, while for $J_1 < 0$ the measured plant output is leading the reference, i.e., overcompensation is occurring. Performance measure J_2 establishes the level of how quantitatively different the reference and measured plant output are, considering the entire simulation period. Performance measure J_3 represents the maximum tracking error. The objective of the tracking controller is to maintain J_1, J_2 and J_3 as close to zero as possible [36].

3.2. *Problem formulation*

The case study prototype is a motorcycle. The hybrid model of the motorcycle is made of four NS and one vPS. The NS are: i) the engine, ii) the motorcycle body dynamics, iii) the rear wheel braking system and iv) the front wheel braking system. The vPS is the electrically continuously variable transmission (eCVT) of the motorcycle.

The eCVT vPS corresponds to a MIMO model with two sets of one input / one output. The first set is connected to the motorcycle engine NS and the second one to the motorcycle body NS. The latter connection corresponds to the transmission output shaft of the motorcycle. The engine NS simulates the dynamics of the combustion engine. It is represented by a multi-input-single-output (MISO) model, with its inputs being the throttle percentage thr and the angular velocity of the engine ω_{en} and its single output being the torque of the engine τ_{en}. The motorcycle body NS represents the body or chassis dynamics of the motorcycle together with the dynamics of the wheels, tires and suspensions along with the road profile and the environmental driving conditions. It is represented by a MIMO model with 3 sets of one input / one output. The first set is connected to the eCVT vPS with the torque τ_{vd} as input and the angular velocity ω_{vd} of the transmission output shaft as output. The second and third sets are linked to the rear and front wheel braking system NS respectively. Both braking systems are MISO models. The rear wheel braking system NS inputs are the angular velocity of the rear wheel ω_{rw} and the applied force on the brake pedal $F_{br_{rw}}$ while its single output is the

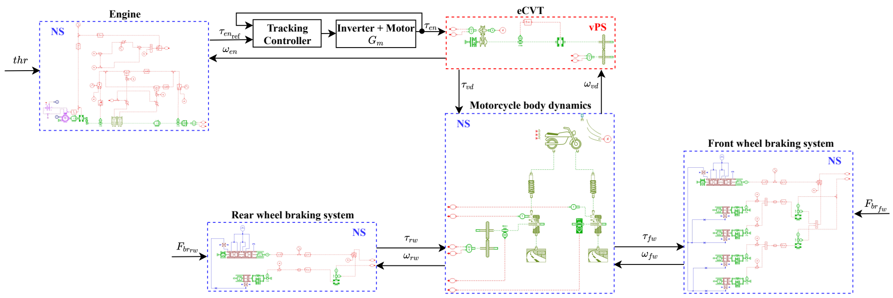

Figure 4. Hybrid model block diagram.

braking torque of the rear wheel τ_{rw}. Respectively, the front wheel braking system NS inputs are the angular velocity of the front wheel ω_{fw} and the applied force on the brake lever $F_{br_{fw}}$ and its single output is the braking torque in the front wheel τ_{fw}. Figure 4 illustrates the interconnections between NS and the vPS, while Figure 5 depicts the real eCVT PS that would be utilized in a non-virtual RTHS. To develop the substructures of the hybrid model, the multi-physics simulation software *Simcenter Amesim* was used. The report of [37] offers a thorough description of the utilized substructures along with the equations governing their motion. To interconnect and co-simulate all substructures as well as the RTHS algorithm in real-time, the *Simcenter real-time platform* was used.

The forces applied on the brake pedal (to activate the rear wheel brakes) and the brake lever (to activate the front wheel brakes) are considered equal and expressed in Newton, $F_{br_{rw}} = F_{br_{fw}}$, and thr, $F_{br_{rw}}$ as defined in Eqs. (28) and (29) respectively. The maximum applied throttle is 0.5 (50%), which corresponds to half-open throttle. In Figure 6 the applied driving scenario of the case study is shown. The road profile, i.e., height of the ground, is expressed by the $h(x)$ sinusoidal signal and follows Eq. (30), where x is the current motorcycle position in meters. The ambient wind velocity is considered to be zero. The simulation duration of the case study is 45 sec. A fourth-order Runge–Kutta (RK4) method with a fixed time step of 0.1 msec is used as the numerical integration scheme in the conducted RTHS of the dynamic response of the motorcycle virtual hybrid model during the driving scenario. Figure 7 displays some indicative dynamic responses of the virtual hybrid model of the motorcycle, excited

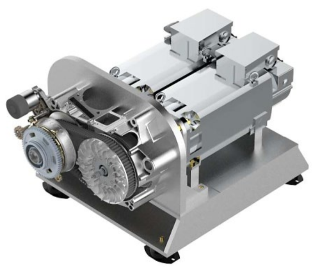

Figure 5. The eCVT test bench.

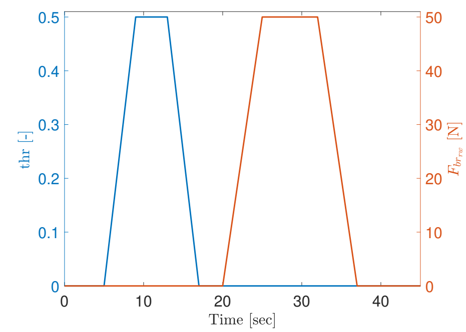

Figure 6. Case study driving scenario.

under the driving scenario on the given road profile and in the given wind conditions.

$$thr(t) = \begin{cases} 0.125t & , 5 \le t < 9 \\ 0.5 & , 9 \le t \le 13 \\ -0.125t & , 13 < t \le 17 \\ 0 & , \text{elsewhere} \end{cases} \tag{28}$$

$$F_{br_{rw}}(t) = \begin{cases} 10t & , 20 \le t < 25 \\ 50 & , 25 \le t \le 32 \\ -10t & , 32 < t \le 37 \\ 0 & , \text{elsewhere} \end{cases} \tag{29}$$

$$h(x) = \begin{cases} 0 & , 0 \le x \le 2 \\ 0.02\Big(\cos\left(\frac{2\pi x}{3}\right) - 1\Big) & , \text{elsewhere} \end{cases} \tag{30}$$

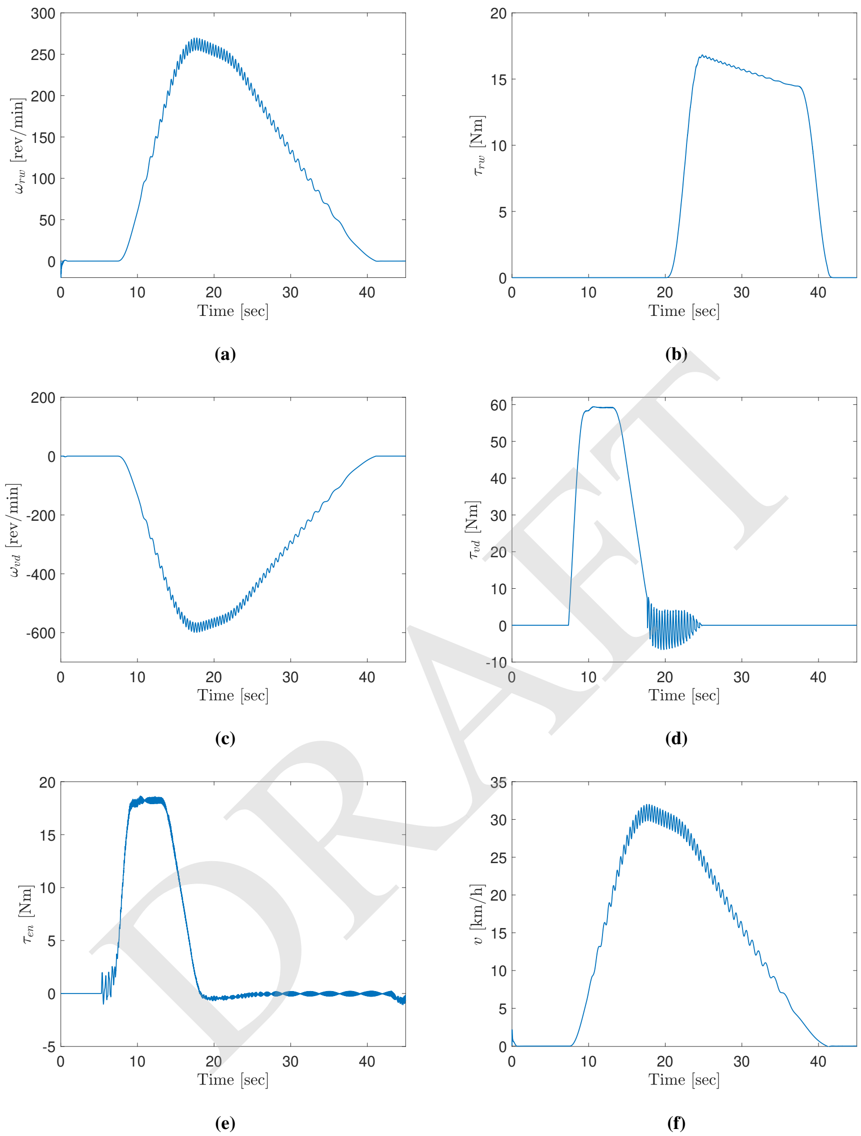

Figure 7. Indicative hybrid model responses: **(a)** angular velocity of the rear wheel, **(b)** braking torque of the rear wheel, **(c)** angular velocity of the transmission output shaft, **(d)** torque of the transmission output shaft, **(e)** torque of the engine and **(f)** motorcycle velocity.

Because the eCVT vPS (the motorcycle transmission) has two inputs, τ_{en} and ω_{vd}, two actuation systems would be required if the RTHS would be conducted physically;

hence two individual tracking controllers would be needed. However, without loss of generality and for the sake of simplicity, one actuation system/tracking controller is utilized as illustrated in Figure 4. It is applied between the eCVT vPS and the Engine NS as the interconnection between the two has the highest frequency content thus requiring the compensation for the RTHS to be as good as possible.

The plant of the case study corresponds to the eCVT vPS actuation system, i.e., the inverter and the electric motor of Figure 4. Therefore, the plant output y of Figure 1 corresponds to the torque of the engine τ_{en} while the reference r signal to $\tau_{en_{\mathrm{ref}}}$. Measurements were conducted on the corresponding test bench at the testing facilities of Siemens Industry Software for the system identification of the inverter and the electric motor. This actuation system was identified as a second-order transfer function. Such system is represented in continuous time as follows:

$$G_m = \frac{b_1 s + b_0}{a_2 s^2 + a_1 s + a_0}, \tag{31}$$

where

$$b_1 = 0,\ b_0 = a_0 = 5657,\ a_2 = 1 \text{ and } a_1 = 78.62. \tag{32}$$

It can also be expressed as $G_m = \omega_n^2/(s^2 + 2\zeta\omega_n s + \omega_n^2)$, with ω_n and ζ its natural frequency and damping, respectively.

In a study by Silva et al. [36], coefficients of variation of up to 5% and 7% of the plant's natural frequency and damping, respectively, were identified after conducting several HS runs. These were mainly due to physical changes of the experimental setup after repeated testing. Therefore, in order to simulate potential variations of the plant G_m used in this study, its natural frequency ω_n and damping ζ were selected to vary. To be conservative and expose the tracking controller to more severe conditions, ω_n and ζ vary with a coefficient of variation of 15% and 17% respectively, during the simulation, every 0.25 seconds of the RTHS. The variations are implemented by changing the values of b_0, a_0. Particularly, $b_0, a_0 \sim \mathcal{N}(5657, 1640.53)$ and b_0 equals a_0. Therefore, the goal of the real-time model identification algorithm, presented in Section 2.3 is to capture the applied variations of G_m and to correctly identify its changing dynamics and provide the MPC with the updated prediction model. Figures 9a-d illustrate in blue color the applied variations of G_m over the simulation time. The initial values of G_m are those of Eq. (32).

Furthermore, to assess the robustness of the proposed tracking controller under various hybrid model configurations, eleven parameters of the motorcycle virtual hybrid model are selected to vary. Table 1 provides an overview of these parameters. A uniform distribution is assigned to each chosen parameter. Their mean values and standard deviations are identified from [38, 39, 40] to reflect a range of possible parameter variations of the corresponding motorcycle components. The nominal parameter values for this case study are their mean values. Using the Latin hypercube sampling (LHS) method [41], 200 samples are obtained from each parameter and hence 200 hybrid model evaluations are performed. In contrast with a_0, b_0 parameters of G_m that are varying in real-time, the parameters of Table 1 remain constant during each RTHS run, but their values are changed in every successive RTHS. For the scope of the presented study, 200 sample points are considered sufficient to adequately capture the underlying probability space and thus to expose the tracking controller to wide parameter variations. The stability of the hybrid model was not affected by the variation of the plant dynamics and Table's 1 parameters. This can be appreciated from the results presented in Table 2 and Figure 8 of Section 4, as in each case the hybrid model response was stable.

3.3. Tracking controller design properties

The initial matrices of the prediction model (A_p^0, B_p^0, C_p^0 and D_p^0 of Eq. (7)) coincide with the dynamics of the plant model of Eq. (31) and follow:

$$
\begin{aligned}
A_p^0 &= \begin{bmatrix} 0.96859 & -0.0348 \\ 0.02519 & 0.9996 \end{bmatrix}, B_p^0 = \begin{bmatrix} 0.0031 \\ 0 \end{bmatrix}, \\
C_p^0 &= \begin{bmatrix} 0 & 11.0488 \end{bmatrix}, D_p^0 = \begin{bmatrix} 0 \end{bmatrix}.
\end{aligned} \tag{33}
$$

The disturbance d and noise n models, described in Eqs. (2) and (3), respectively, are additive to the plant output and admit:

Parameter	Description	Distribution	Mean value	Standard deviation	Units
K_{rt}	Vertical stiffness rear tire	Uniform	58570	11714	N/m
Z_{rt}	Vertical damping rear tire	Uniform	11650	3495	Ns/m
K_{ft}	Vertical stiffness front tire	Uniform	25000	5000	N/m
Z_{ft}	Vertical damping front tire	Uniform	2134	640.2	Ns/m
K_{rs}	Stiffness rear suspension	Uniform	125000	25000	N/m
Z_{rs}	Damping rear suspension	Uniform	10000	3000	Ns/m
K_{fs}	Stiffness front suspension	Uniform	19000	3800	N/m
Z_{fs}	Damping front suspension	Uniform	1250	375	Ns/m
M	Motorcycle mass	Uniform	300	6	Kg
J	Engine moment of inertia	Uniform	0.0115	0.0023	Kgm^2
μ	Engine coefficient of viscous friction	Uniform	0.001	0.00005	Nm/(rev/min)

Table 1. Motorcycle virtual hybrid model parameters with their assigned statistical distributions.

$$
\begin{aligned}
x_d(k+1) &= x_d(k) + 0.0004u_d(k) \\
d(k) &= x_d(k),
\end{aligned} \tag{34}
$$

$$
n(k) = u_n(k), \tag{35}
$$

with $A_d = 1, B_d = 0.0004, C_d = 1$ and $D_d = 0$, while $A_n = 0, B_n = 0, C_n = 0$ and $D_n = 1$. Recall that the inputs to the disturbance and noise models, i.e., u_d and u_n, respectively, follow the standard Gaussian distribution, namely $u_d, u_n \sim \mathcal{N}(0, 1)$. According to Eqs. (10) and (11), the state-space formulation of the plant dynamics for $k = 0$, including the disturbance and noise model follows:

$$
\begin{aligned}
A_c &= \begin{bmatrix} 0.96859 & -0.0348 & 0 & 0 \\ 0.02519 & 0.9996 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, B_c = \begin{bmatrix} 0.0031 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0.0004 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\
C_c &= \begin{bmatrix} 0 & 11.0488 & 1 & 0 \end{bmatrix}, D_c = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix},
\end{aligned} \tag{36}
$$

The initial Kalman filter gain vectors admit:

$$
\begin{aligned}
L^0 &= \begin{bmatrix} 0.157 & 0.917 & 0.398 \end{bmatrix}^T 1\text{e}{-3}, \\
M^0 &= \begin{bmatrix} 0.195 & 0.912 & 0.398 \end{bmatrix}^T 1\text{e}{-3}.
\end{aligned} \tag{37}
$$

The tuning weights of Eq. (1) are selected to be $w^y = 5.204$ and $w^u = 0.096$, while n_u, n_y equal to 1 as the plant is a single-input-single-output (SISO) model. Each control interval is obtained with a sampling frequency of 2.5 kHz and the prediction horizon is $P = 10$. The constraints of Eq. (9) represent safety limits regarding the plant input and admit:

$$
-200 \leq u \leq 200 \quad [\text{Nm}]. \tag{38}
$$

4. Results and discussions

Table 2 presents the performance metrics, J_1, J_2 and J_3, of the proposed tracking controller for the aforementioned simulation and Figure 8 displays their normalized histograms.

	Nominal	Stochastic	
		Mean Values	Standard Deviation
J_1 [msec]	2.5	2.5	0
J_2 [%]	0.32	0.34	0.01
J_3 [%]	2.79	2.82	0.21

Table 2. Performance metrics results.

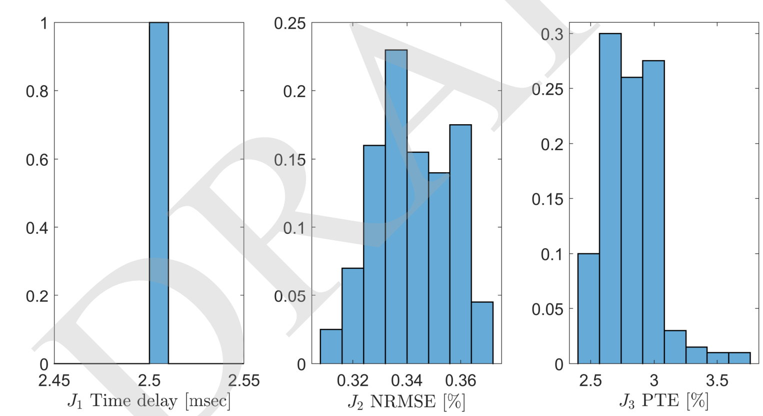

Figure 8. Normalized histograms of J_1, J_2 and J_3, obtained from 200 virtual RTHS evaluations.

The nominal results of Table 2 refer to the case when the mean values of the parameters in Table 1 were used: hence they correspond to a single deterministic RTHS. The stochastic results refer to the outcomes from the 200 RTHS evaluations. In both deterministic (nominal) and stochastic cases, the parameters a_0, b_0 of G_m vary as displayed in Figure 9.

As mentioned above, the performance of the tracking controller is assessed by how close to zero the performance metrics are. From Table 2 and Figure 8, it can be

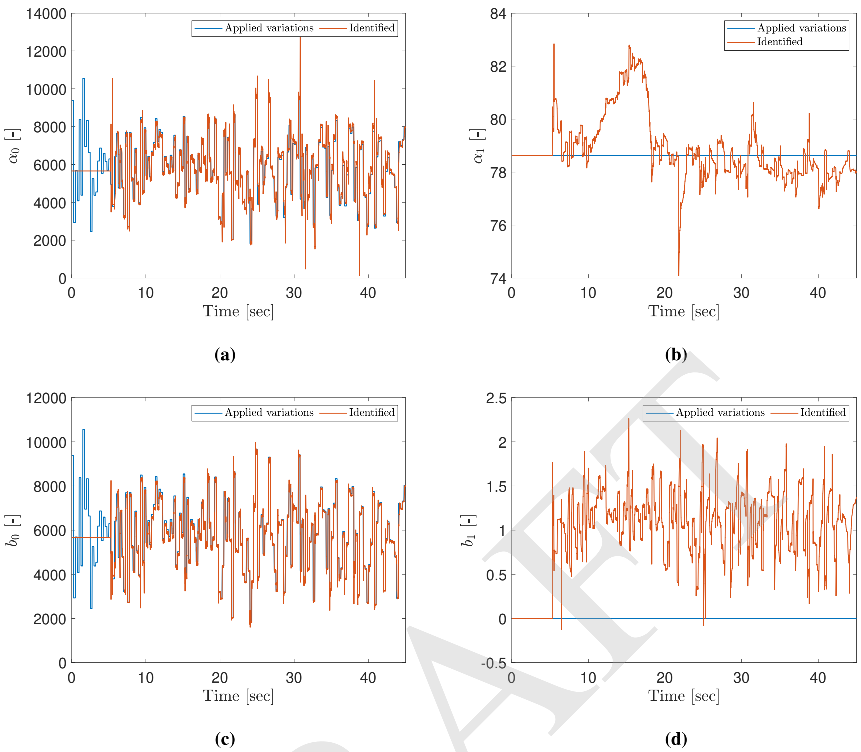

Figure 9. Applied variations in the parameters of the plant and the identified values from the real-time model identification algorithm: **(a)** a_0, **(b)** a_1, **(c)** b_0 and **(d)** b_1.

appreciated that J_1, J_2 and J_3 are quite small and hence the proposed control scheme can adequately regulate the desired plant output even under the presence of hybrid model parameter and plant dynamics variations, proving as well its robustness.

Figure 9 also presents in red color the parameters of the plant dynamics that the real-time model identification algorithm estimates at each control interval. As stated in Eq. (28), for the first 5 seconds of the simulation there is no input (zero values) to the hybrid model and hence the output is also zero. As a result, the identification algorithm returns zero values. It can be observed that after the first 5 seconds, the algorithm starts to respond. Recall that there were no applied variations for a_1 and b_1. However, the algorithm fails to accurately identify these two parameters (Figures 9b,d). Nevertheless, their deviation is relatively small. Regarding a_0 and b_0, the predictions are satisfactory

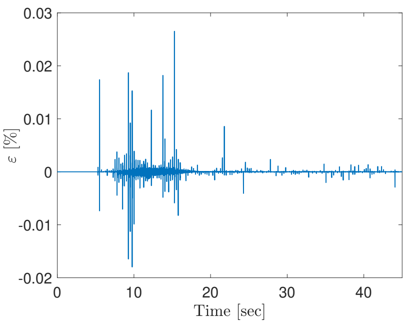

Figure 10. Plant system identification error.

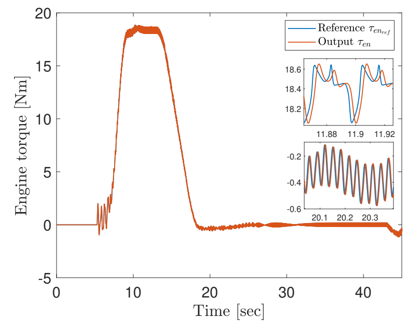

Figure 11. Torque responses of the engine NS: reference $\tau_{en_{ref}}$ and measured τ_{en}.

as shown from Figures 9a,c. Figure 10 displays the overall identification error of the real-time model identification algorithm, defined as:

$$\varepsilon(t) = \frac{y(t) - y_{\text{pred}}(t)}{\max|y(t)|} \times 100 \quad [\%]. \tag{39}$$

From Figure 10, it can be stated that the overall identification error is quite small and thus the parameter identifications are acceptable. It should be noted that the performance of the real-time model identification algorithm is crucial for the overall performance of the adaptive MPC. The model that the algorithm identifies is used internally in MPC as its prediction model. Hence, faulty system identification of the plant could yield large tracking errors.

Figure 11 displays, for the nominal RTHS case, the reference values of the plant, namely the output of the engine NS $\tau_{en_{ref}}$, and the plant output, namely the input to the vPS, τ_{en}. An ideal tracking controller should be able to compensate for the actuation system dynamics so that $\tau_{en_{ref}}$ and τ_{en} are as close as possible. As shown from Figure 11, the tracking error between the two signals is very small indeed.

Based on the above results, the proposed tracking controller can provide the desired performance and at the same time be robust to hybrid model and actuation system variations, as well as to the introduced disturbances and measurement noise. Additionally, using the proposed tracking controller, the tracking errors and time delays that are introduced due to the inherent dynamics of the actuation system used, are satisfactorily compensated, enabling thus RTHS outcomes of high fidelity. As a result, the proposed control scheme is demonstrated to be suitable for RTHS.

5. Conclusions

In this study, a novel tracking controller for dynamics compensation in real-time hybrid simulation was proposed. Although real-time hybrid simulation is an effective testing technique, it is challenging as the inherent dynamics of the actuation system introduce time delays that require compensation. Therefore, a good tracking controller is an element of the physical specimen testing setup needed to obtain high fidelity real-time hybrid simulation. The proposed control scheme is an extension of the model predictive control presented in [21] to include an adaptive behavior. It is based on adaptive model predictive control, a linear time-varying Kalman filter, and a real-time model identification algorithm. For the latter, ARX polynomial models are identified in real-time to estimate the changing plant dynamics and are used to update the prediction model of the model predictive controller. A parametric virtual real-time hybrid simulation case study is used to validate the performance and robustness of the proposed control scheme. In particular, the plant dynamics were varying during the simulation in the presence of disturbance and measurement noise, additive to its output. Additionally, parameters of the hybrid model were chosen and assigned with random values via prescribed probability distributions. A total of 200 samples were generated and 200 hybrid simulation evaluations were performed to assess whether the proposed control scheme is robust and preserves the desired performance. Results indicate that the controller can guarantee small tracking errors under uncertainties that may be present during the hybrid simulation. As a result, the effectiveness of the proposed control methodology was demonstrated proving that the adaptive model predictive control is a suitable approach for real-time hybrid simulation. Future work aims to deploy the proposed tracking controller in a test bench as well as to examine how multiple model predictive controllers could coordinate in real-time, controlling multiple individual physical substructures of a hybrid model.

6. Data Availability Statement

All data and code that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 764547. The sole responsibility of this publication lies with the author(s). The European Union is not responsible for any use that may be made of the information contained herein.

References

[1] A. H. Schellenberg, S. A. Mahin, G. L. Fenves, Advanced Implementation of Hybrid Simulation, Tech. Rep. PEER 2009/104, Pacific Earthquake Engineering Research Center, University of California, Berkeley (2009).

[2] G. Miraglia, M. Petrovic, G. Abbiati, N. Mojsilovic, B. Stojadinovic, A model-order reduction framework for hybrid simulation based on component-mode synthesis, Earthquake Engineering & Structural Dynamics 49 (8) (2020) 737–753.

[3] N. Tsokanas, T. Simpson, R. Pastorino, E. Chatzi, B. Stojadinovic, Model order reduction for real-time hybrid simulation: Comparing polynomial chaos expansion and neural network methods (Aug 2021). doi:10.31224/osf.io/h2bnm.

[4] Horiuchi T., Development of a realtime hybrid experimental system with actuator delay compensation, in: Proc. 11th World Conference on Earthquake Engineering, 1996.

[5] M. Wallace, D. Wagg, S. Neild, An adaptive polynomial based forward prediction algorithm for multi-actuator real-time dynamic substructuring, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 461 (2064) (2005) 3807–3826, publisher: Royal Society.

[6] M. I. Wallace, J. Sieber, S. A. Neild, D. J. Wagg, B. Krauskopf, Stability analysis of real-time dynamic substructuring using delay differential equation models, Earthquake Engineering & Structural Dynamics 34 (15) (2005) 1817–1832.

[7] J. Zhao, C. French, C. Shield, T. Posbergh, Considerations for the development of real-time dynamic testing using servo-hydraulic actuation, Earthquake Engineering & Structural Dynamics 32 (11) (2003) 1773–1794.

[8] P. J. Gawthrop, M. I. Wallace, S. A. Neild, D. J. Wagg, Robust real-time substructuring techniques for under-damped systems, Structural Control and Health Monitoring 14 (4) (2007) 591–608.

[9] R.-Y. Jung, P. Benson Shing, E. Stauffer, B. Thoen, Performance of a real-time pseudodynamic test system considering nonlinear structural response, Earthquake Engineering & Structural Dynamics 36 (12) (2007) 1785–1809.

[10] P. A. Bonnet, C. N. Lim, M. S. Williams, A. Blakeborough, S. A. Neild, D. P. Stoten, C. A. Taylor, Real-time hybrid experiments with Newmark integration, MCSmd outer-loop control and multi-tasking strategies, Earthquake Engineering & Structural Dynamics 36 (1) (2007) 119–141.

[11] C. Chen, J. M. Ricles, Analysis of actuator delay compensation methods for real-time testing, Engineering Structures 31 (11) (2009) 2643–2655.

[12] Y. Chae, K. Kazemibidokhti, J. M. Ricles, Adaptive time series compensator for delay compensation of servo-hydraulic actuator systems for real-time hybrid simulation, Earthquake Engineering & Structural Dynamics 42 (11) (2013) 1697–1715.

[13] P.-C. Chen, C.-M. Chang, B. F. Spencer, K.-C. Tsai, Adaptive model-based tracking control for real-time hybrid simulation, Bulletin of Earthquake Engineering 13 (6) (2015) 1633–1653.

[14] A. Najafi, B. F. Spencer, Adaptive model reference control method for real-time hybrid simulation, Mechanical Systems and Signal Processing 132 (2019) 183–193.

[15] T. Simpson, V. K. Dertimanis, E. N. Chatzi, Towards Data-Driven Real-Time Hybrid Simulation: Adaptive Modeling of Control Plants, Frontiers in Built Environment 6, publisher: Frontiers (2020).

[16] G. Ou, A. I. Ozdagli, S. J. Dyke, B. Wu, Robust integrated actuator control: experimental verification and real-time hybrid-simulation implementation, Earthquake Engineering & Structural Dynamics 44 (3) (2015) 441–460.

[17] X. Ning, Z. Wang, H. Zhou, B. Wu, Y. Ding, B. Xu, Robust actuator dynamics compensation method for real-time hybrid simulation, Mechanical Systems and Signal Processing 131 (2019) 49–70.

[18] B. Phillips, B. Spencer Jr., Model-based multiactuator control for real-time hybrid simulation, Journal of Engineering Mechanics 139 (2) (2013) 219–228.

[19] B. Phillips, B. Spencer, Model-based feedforward-feedback actuator control for real-time hybrid simulation, Journal of Structural Engineering 139 (7) (2013) 1205–1214.

[20] H. Li, A. Maghareh, H. Montoya, J. W. Condori Uribe, S. J. Dyke, Z. Xu, Sliding mode control design for the benchmark problem in real-time hybrid simulation, Mechanical Systems and Signal Processing 151 (2021) 107364.

[21] N. Tsokanas, D. Wagg, B. Stojadinović, Robust Model Predictive Control for Dynamics Compensation in Real-Time Hybrid Simulation, Frontiers in Built Environment 6 (2020).

[22] B. R. Maner, F. J. Doyle, B. A. Ogunnaike, R. K. Pearson, A nonlinear model predictive control scheme using second order Volterra models, in: Proceedings of 1994 American Control Conference - ACC '94, Vol. 3, 1994, pp. 3253–3257.

[23] E. Kaiser, J. N. Kutz, S. L. Brunton, Sparse identification of nonlinear dynamics for model predictive control in the low-data limit, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 474 (2219) (2018) 20180335.

[24] D. Q. Mayne, J. B. Rawlings, C. V. Rao, P. O. M. Scokaert, Constrained model predictive control: Stability and optimality, Automatica 36 (6) (2000) 789–814.

[25] J. A. Rossiter, Model-based predictive control: a practical approach, Control series, CRC Press, Boca Raton, 2003.

[26] E. Zafiriou, Robust model predictive control of processes with hard constraints, Computers & Chemical Engineering 14 (4-5) (1990) 359–371.

[27] E. F. Camacho, C. Bordons, Model Predictive control, Advanced Textbooks in Control and Signal Processing, Springer London, London, 2007.

[28] P. Tøndel, T. A. Johansen, A. Bemporad, An algorithm for multi-parametric quadratic programming and explicit MPC solutions, Automatica 39 (3) (2003) 489–497.

[29] F. Delbos, J. C. Gilbert, Global linear convergence of an augmented Lagrangian algorithm for solving convex quadratic optimization problems, Tech. Rep. 00071556, INRIA (2003).

[30] G. Bitsoris, Positively invariant polyhedral sets of discrete-time linear systems, International Journal of Control 47 (6) (1988) 1713–1726.

[31] S. P. Boyd, L. Vandenberghe, Convex optimization, Cambridge University Press, Cambridge, UK ; New York, 2004.

[32] C. Schmid, L. T. Biegler, Quadratic programming methods for reduced hessian SQP, Computers & Chemical Engineering 18 (9) (1994) 817–832.

[33] A. Bemporad, N. L. Ricker, M. Morari, Model Predictive Control Toolbox: Getting Started Guide, The Math Works, 2020.

[34] H. Akaike, Fitting autoregressive models for prediction, Annals of the Institute of Statistical Mathematics 21 (1) (1969) 243–247.

[35] L. Ljung, System identification theory for the user, 2nd Edition, Prentice Hall information and system sciences series, Prentice Hall, Upper Saddle River, N.J, 1999.

[36] C. E. Silva, D. Gomez, A. Maghareh, S. J. Dyke, B. F. Spencer, Benchmark control problem for real-time hybrid simulation, Mechanical Systems and Signal Processing 135 (2020) 106381.

[37] S. M. Pinheiro, Motorcycle modeling for eCVT-in-the-loop real-time hybrid testing, Master's thesis, University of Porto, Porto, Portugal (2020).

[38] T. Kimishima, T. Nakamura, T. Suzuki, The Effects on Motorcycle Behavior of the Moment of Inertia of the Crankshaft, SAE Transactions 106 (1997) 1993–2003.

[39] M. Tanelli, Modelling, simulation and control of two-wheeled vehicles, John Wiley & Sons, 2014.

[40] R. Sharp, S. Evangelou, D. Limebeer, Advances in the Modelling of Motorcycle Dynamics, Multibody System Dynamics 12 (3) (2004) 251–283.

[41] M. D. Mckay, R. J. Beckman, W. J. Conover, A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output From a Computer Code, Technometrics 42 (1) (2000) 55–61.