Geographically distributed real-time co-simulation of electric vehicle

1st Jesus Alfonso
Mechatronics Dept.
Instituto Tecnológico de Aragón
Zaragoza, Spain
ORCID 0000-0002-3694-8130

2nd Jose Manuel Rodriguez
Mechatronics Dept.
Instituto Tecnológico de Aragón
Zaragoza, Spain
ORCID 0000-0001-8080-7696

3nd Carlos Bernad
Mechatronics Dept.
Instituto Tecnológico de Aragón
Zaragoza, Spain
ORCID 0000-0002-4235-0776

4rd Viktar Beliautsou
Automotive Engineering Group
Technische Universität Ilmenau
Ilmenau, Germany
ORCID 0000-0002-2071-1198

5th Valentin Ivanov
Automotive Engineering Group
Technische Universität Ilmenau
Ilmenau, Germany
ORCID 0000-0001-7252-7184

6th Jose Angel Castellanos
Instituto de Investigación en Ingeniería de Aragón
Universidad de Zaragoza
Zaragoza, Spain
ORCID 0000-0001-5977-8720

***Abstract*—The present paper shows the capabilities of a distributed real-time co-simulation environment merging simulation models and testing facilities for developing and verifying electric vehicles. This environment has been developed in the framework of the *XILforEV* project and the presented case is focused on a ride control with a real suspension installed on a test bench in Spain, which uses real-time information from a complete vehicle model in Germany. Given the long distance between both sites, it has been necessary to develop a specific delay compensation algorithm. This algorithm is general enough to be used in other real-time co-simulation frameworks. In the present work, the system architecture including the communication compensation is described and successfully experimentally validated.**

***Index Terms*—co-simulation, real-time, FMU, HIL**

I. Introduction

Electric vehicles are a fundamental part of the new paradigm in mobility and they pose new challenges and opportunities arising both from the electrification itself and the relationship between the different components and elements in the vehicle. The resulting complexity is difficult to handle due to the different technologies, maturity levels, locations and development teams involved in the process. Methodologies like Software-in-the-Loop (SIL), Model-In-the-Loop (MIL), Hardware-In-the-Loop (HIL) or Test-Rig-In-the-Loop (TRIL) [1] contribute to facilitating the process but there are still aspects to support the concurrent and integrated development and testing of the vehicle: on the one hand, to make possible the real-time collaboration between different groups at different locations; and on the other hand, the possibility to check the designs at different development stages, therefore merging virtual and real prototypes. The present work proposes a solution for those points and it is built using the *XILforEV* framework [2] [3], which creates a distributed X-In-Loop environment (XIL) merging SIL, MIL, HIL, and TRIL for electric vehicles. The environment connects test platforms and setups from different physical domains placed at different locations.

With the advances in communications, examples of distributed XIL architectures can be found in the literature. [4] tests a distributed XIL between Germany and China evaluating the effect of a remote driver sending acceleration and brake commands to a vehicle simulator and remarks the oscillatory performance of a speed control due to the delay (around 800 ms) and other system features like model structure and solver. [5] describes an Internet-distributed XIL for a parallel HEV (Hybrid Electric Vehicle). In this case, the slow dynamics of battery, engine, and vehicle are present as HIL and VIL (Vehicle-In-Loop) components respectively, and the fast response of the drivetrain model is simulated. The information is shared through a cloud orchestrator. In this case, the test takes place within a distance of around 20 km which, given the distribution of simulated and HIL/VIL components, assures the stability. [6] describes a XIL architecture between Germany and Austria using the ACOSAR Distributed Co-Simulation Protocol linking simulated components and a small test bed for the vehicle dynamics. In all cases, the effect of the network delays is remarked as a key factor for assuring the performance and stability of the co-simulation. This factor is crucial in XIL environments as part of the nodes are real components that could become damaged during the test. For this reason, the *XILforEV* environment includes delay compensators for handling both network delays and jitter, as also packet losses.

Several methods have been proposed in the literature for handling communication delays. Apart from the standard zero-order-hold approaches [24], there are different methods for predicting and compensating the transmitted value: linear or polynomial predictors [7] and FIR filters [8], whose coefficients are dynamically estimated using the input data; model

The work presented in this article is part of the activities carried out in the European Projects ”Connected and Shared X-in-the-loop Environment for Electric Vehicles Development (XILforEV)” and ”Digitbrain”. These projects have received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreements No. 824333 and No. 952071.
The authors want to acknowledge XILforEV partners for the prototypes and electric vehicle models used for the development of the experiments.

based predictors like [9], [10] with simple second-order models, or [11] using neural networks. The delay compensation is a fundamental part of the *XILforEV* framework and several solutions have been evaluated and tested: in [12] the Package Predictive Delay and Dropout Compensation (*PPDDC*) algorithm is described based on using a ROM (Reduced Order Model) for predicting and compensating the delays in a XIL with a smart suspension bench. [13] uses a prediction approach for compensating the delays but in this case, the estimation is done by using a first order predictor further smoothed by an observer-distributor Kalman filter. The present work shows the performance of combining *PPDDC* with a power limitation based on *NEPCE* (Nearly Energy-Preserving Coupling Element) [14] in the model nodes, combined with linear extrapolators at the output of the test rigs.

The proposed architecture is general enough to be used in different applications, but it is proved in the present work by testing smart ride control strategies [15] for the vertical motion dynamics of the vehicle combining the actuation of traction individual electric motors and suspension.

The current paper is organized as follows. The description of the XILforEV architecture is done in Section II. The performance of the communication network and the delay compensation are described in Section III. Finally, the results of the experimental validation appear in Section IV and the conclusions appear in Section V.

II. Architecture of XIL for ride blending

A. Architecture

The XILforEV architecture for the Ride blending test involves two different cities: Zaragoza in Spain and Ilmenau in Germany with a straight line distance of 1351 km between them (Figure 1).

At the *Technische Universität Ilmenau* there is a complete vehicle model of the all-wheel drive sport utility vehicle. The model has been created using *Simcenter Amesim* software. It includes the advanced vehicle dynamic model and it assembles different submodels created in *Amesim* and *Matlab Simulink*. The model also includes the high level controllers

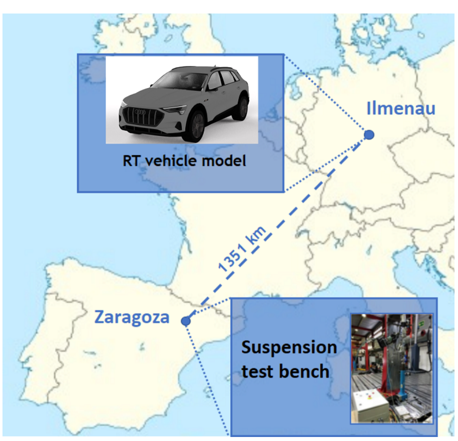

Fig. 1. The two nodes involved in the Ride blending real-time XIL.

that provide commands to in-wheel motors (IWM), brakes and active suspensions. The complete model runs on a dSpace SCALEXIO real-time computer. In a separate building, there are different tests benches: an in-wheel motor (IWM) test bench for measuring the response of an IWM prototype, an hydraulic brake test bench for testing the pad force and a friction brake test bench measuring the torque generated by the brake pads on the brake disk.

At the *Technological Institute of Aragon* in Zaragoza there is a suspension test bench. It consists of a complete automotive active damper installed on a bench that can simulate the stroke of the suspension and measures the force done by the actuator.

The complete vehicle model can use information from the test rigs or the component models. The stroke information which is tracked by the test rig at Zaragoza is calculated in the real-time model at Ilmenau for one of the wheels. The force at the real suspension is fed to the model as a simulation input. The behavior of the suspensions at the other wheels is simulated by the full vehicle model.

The communication between the real-time simulator and the test benches uses User Datagram Protocol (UDP) based on previous investigations from the XILforEV consortium [16]. The XILforEV approach also uses a virtual private network (VPNs) for connecting the distributed XIL nodes. The VPN uses Internet to communicate the real-time simulator with the test rig in Zaragoza, and the LAN of the *Technische Universität Ilmenau* to communicate with the other test rigs.
The orchestration of the co-simulation is managed by status machines at the test rigs and the numerical simulator with defined states. The numerical simulator sends request messages to the test rigs to advance their status machines and waits for the completion of all required participants before moving to the next state. Once co-simulation starts, the participants exchange messages with time information, including the transmitted test values and the status of the local state machines.

B. Ride Blending controller

The coordinated action of the different actuators in the vehicle chassis by the use of advanced control strategies has a great potential for improving driving performance and comfort. The ride blending controller tested in the *XILforEV* environment involves two groups of actuators: active suspension and in-wheel electric motors. The architecture of the ride blending controller has several layers (Figure 2) to generate the control demands for the actuators: the *reference generator*, the *high level* controller, and the *ride blending controller*.

The *reference generator* sends the reference values for the sprung mass and it calculates the error signal using the estimated vehicle states. To provide maximum comfort and safety for the driver and passengers, the controller keeps minimum body movements and rotations, and therefore the reference values for the vertical acceleration, pitch and roll are zero. The output includes the vertical acceleration error in CoG ($\ddot{z_{cog}}$), the roll angle error ($\dot{\Phi}$) and the pitch angle error ($\dot{\Theta}$).

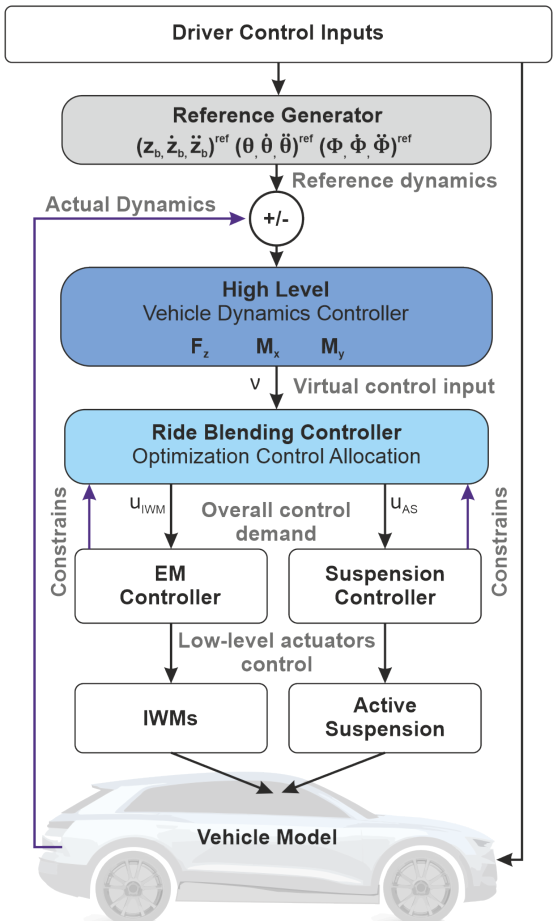

Fig. 2. Architecture of ride blending controller.

The *high level controller* controls the vehicle movement in three dimensions: vertical translation and rotations around x-axis and y-axis. Therefore, three input values are calculated using the reference generator output parameters:

$$\chi = \begin{pmatrix} \ddot{z_{cog}} \cdot m_v \\ \dot{\phi} \cdot I_x \\ \dot{\Theta} \cdot I_y \end{pmatrix}, \tag{1}$$

where m_v is the vehicle mass and I_x, I_y are the pitch and roll moments of inertia.

For calculating the command controls (vertical force $F_{z,dem}$, pitch-axis torque $M_{x,dem}$, roll-axis torque $M_{y,dem}$) there is one PID for each χ dimension:

$$\upsilon_{HL,i} = K_p \chi_i(t) + K_i \int \chi_i(t)\, dt + K_d \frac{\partial \chi_i}{\partial t}, \tag{2}$$

where K_p, K_i and K_d are the proportional, integral and derivative terms of the PID.

The *ride blending controller* allocates the υ_{HL} vector demand into four vertical forces F_z^{ij} and four torques T_{IWM}^{ij} at each ii suspension and IWM actuator. It takes into account the efficiency ranges, the actuator limits, the wheel slip and the high level demand. The allocation is done by using an optimization algorithm. In the present work, the cost function was adapted to control two real actuators (one IWM and one suspension) installed on two benches inside the distributed *XILforEV* environment.

III. Analysis of the Network and Delay Compensation

UDP communication is used for data transmission from the real-time simulator and the test rig to the VPN router. Although TCP is a more reliable protocol because it assures the order of the packages and it re-sends the message in case of communication issues, in the present application UDP is considered a better option [16] due to its compatibility with communication hardware, transmission unit size and data transfer rates. The participants in the XILforEV constellation connect through Cisco VR-340 routers, which provide a virtual private network (VPN) between the participants using VPN TLS protocol. It protects the facility from man-in-the-middle attacks or accidental access to the devices of the XILforEV network. The router encrypts the information in TCP packages using TLS protocol, so UDP transmission is not affected by dropouts or race conditions from one router to another.

A. Network delays

The delay in Internet communications is influenced by the transmission speed in the communication medium and the infrastructure. There are experiments with cables that can almost reach light speed in vacuum [17] but the current infrastructure uses silica fiber optic cable with a limited 200.000 km/s speed. The distance between the two nodes tested in the present work is 1.350 km, so a minimal delay of 7 ms is expected in the communications in one direction and 14 ms for round trip time (RTT). The RTT is the time to receive the response and it is usually measured with the PING tool [18]. In the present experiment, the measured average RTT for the longest distance was 40 ms.

Usually, the internet connection between points is negotiated by several internet providers, but in this case, the participants in the experiment are research institutions that connect through the European academic network named GEANT. The use of this network allows maintaining a quality similar to the best results reported in [18] and better than [4].

The delays in the LAN of the *Technische Universität Ilmenau* affect the communication between the real-time controller and the test benches located on the campus. The measured RTT in the LAN is 2 ms.

Delays in the LAN are so small that do not cause stability problems in the co-simulation, but the delay in the communication of the suspension test bench and the real-time simulator produces instability. So delay compensation is applied to the data exchange between them.

B. Coupling algorithm

Figure 3 shows the connection of the test benches to the numerical simulators and the exchange of UDP messages. There is no direct communication between test benches.

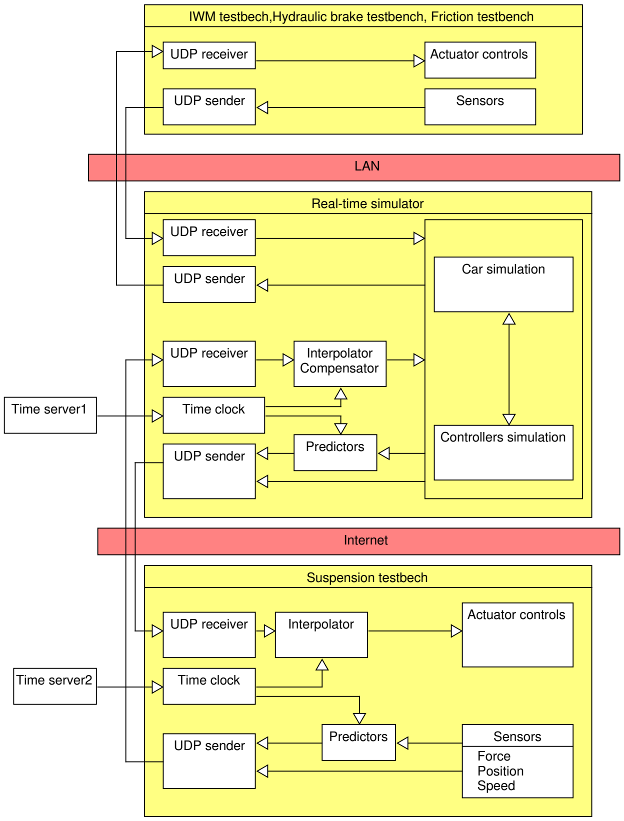

Fig. 3. Delay compensation of suspension testbech.

The use of test rigs coupled to real-time simulation includes the following sources of error usually not considered in pure numerical simulations: the quantization error introduced by the analog-to-digital converters and the electrical noise in the sensors, the tracking error in the benches and the limited bandwidth of the test rig actuators. This is the reason why the UDP messages sent by the test rigs not only contain the value of the signals but also the state of the actuators. For example, the real-time simulator sent position/speed commands to the actuators that compress the suspension in the test bench. The suspension test bench sends the value of force measured in the suspension, but also the measured position and speed.

Noise problems can be mitigated using low pass filters and averaging the signals but their use must be reduced to a minimum because they also add delay to the signals and can lower the precision and stability of the co-simulation.

Moreover, apart from the previous issues, the use of geographically distributed real-time co-simulation adds delay and dropout in the communications between the test rigs, and also the shift of the clock times as sources of error.

We use a Package Predictive Delay and Dropout Compensation (PPDDC) described in [12] to cope with the problem of variable delays and dropouts in the transmission line. Each simulation unit makes an array of predictions for an array of potential delays in the transmission between simulation units. This array of predictions is sent from the source simulation unit to the consumer simulation unit. The consumer simulation unit receives the information and uses it to generate the values for its inputs using interpolation between the prediction set.

1) Time clock and synchronization of local clocks: The synchronization of the local clocks of the test rigs and the real-time simulator is done by using the timestamp of near network time servers. In the initial stages of the simulation, a UTM time is defined as simulation time 0. In the co-simulation start-up phase, each *XILforEV* node uses Network Time Protocol (NTP) to get a UTM timestamp from the NTP server and the timestamp of the local clock to calculate the offset between the local clock time and the simulation time. Each test rig and the real-time simulator computes the simulation time as the timestamp obtained by the local clock corrected by the offset.

The drift of the clock frequency is due to the drift of the frequency of the quartz oscillator and it is usually measured in part per million (ppm). In the present work, the step time used for the simulation is 1ms and the duration of the longest test is 30s so a precision of 33.3 ppm is necessary to avoid additional synchronization during the test. The precision of the local clock of the suspension testbench measured in an experiment is 5 ppm, so the deviation is 1ms every 200 seconds. Longer tests will require re-synchronization of the local clock during the test.

2) Predictors: Several predictors have been proposed to couple the simulation units and are potential candidates for the predictors embedded in the PPDDC. For example, linear and polynomial extrapolation [19], filters [20] or models [21], [22]. These predictors use the information exchanged between the simulation units to compute the ahead values.

In this paper, we have used a predictor for the numerical simulation unit (car simulation in Figure 3) that uses the knowledge of the physics of the modeled system and the internal state values computed by the simulation unit to improve the quality of the predictions. This way, the predictor obtains the acceleration, speed and position of the suspension from the simulation model. The acceleration is filtered and limited to reduce the bandwidth and to avoid high frequency noise. The obtained values are used with a linear predictors to get the predicted positions and speeds for an horizon of 15 ms, 20 ms, 25 ms and 40 ms. The filters and predictors are tuned using signals obtained from simulation (Model-In-Loop).

The information in a co-simulation unit at each time step is encapsulated in a unique UDP message containing: the timestamp in simulation time, information about the state machine, the current values of the unit output and the array of predictions with its corresponding future timestamp.

3) Interpolator compensator: The objective of the interpolator-compensator is to provide the best estimation of the input in the co-simulation unit for the time assigned by the Time-clock block, that is to say, the real co-simulation time. This block uses interpolation between the array of predictions that arrives to the UDP receiver from the predicted values.

The block of the real-time simulator also includes a low pass filter to block high frequency noises of the force sensor and a power based compensation similar to that described in [14].

IV. Results analysis

The goal of this section is to verify the performance of the distributed co-simulation using the delay compensation algorithm described in the previous section. To do that, a reference scenario has been run in MIL (Model-In-Loop) without delays as reference, and it is compared with the experimental results obtained in the distributed co-simulated *XILforEV* environment. Initial co-simulation without delay compensation was unstable and made it necessary its implementation.

a) Validation of the predictors in straight braking manoeuvre: First of all, the behavior of the predictor of Fig. 4 is evaluated during a straight braking maneuver, where the vehicle accelerates up to a reference speed. Once the longitudinal movement is constant, the driver actuates the brake pedal at a constant position and the vehicle begins to slow down until it finally stops. The evaluation has been made in virtual conditions, simply calculating and comparing the prediction values.

To check the behaviour of the predictor, the normalized prediction error (NPE) is used. This indicator is defined as the accumulative absolute difference between the prediction at instant K (X_{predK}) and the real value (X), normalized by the accumulative error if no compensation were done:

$$NPE = \frac{\sum_{i=1}^{n} |(X_{predK}(i) - X(i+K)|}{\sum_{i=1}^{n} |(X(i) - X(i+K)|}. \tag{3}$$

Table Ishows the NPE for the speed and position of the suspension for different prediction values. As it can be observed, the errors are higher with the differentiated variables (speed compared to position) and increase with longer prediction times.

TABLE I
NPE value for different predictions

Prediction time [ms]	NPE position	NPE speed
15	0.031	0.50
20	0.03	0.46
30	0.05	0.72
40	0.06	0.74

b) Co-Simulation performance during a double lane change: The performance of the complete co-simulation can be better observed in a complex maneuver like the double lane change. In this case, once the vehicle reaches a predefined velocity, the driver rapidly changes the road lane, and afterwards returns to the previous lane. This situation corresponds to the vehicle avoiding an unexpected obstacle on the road. This evaluation has been made in the complete distributed *XILforEV* environment.

For evaluating the results, the Normalized-Root-Mean-Square error (NRMSE) is used according to the definition below:

$$NRMSE = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(X(i) - X_{XIL}(i))^2}}{X_{max} - X_{min}}, \tag{4}$$

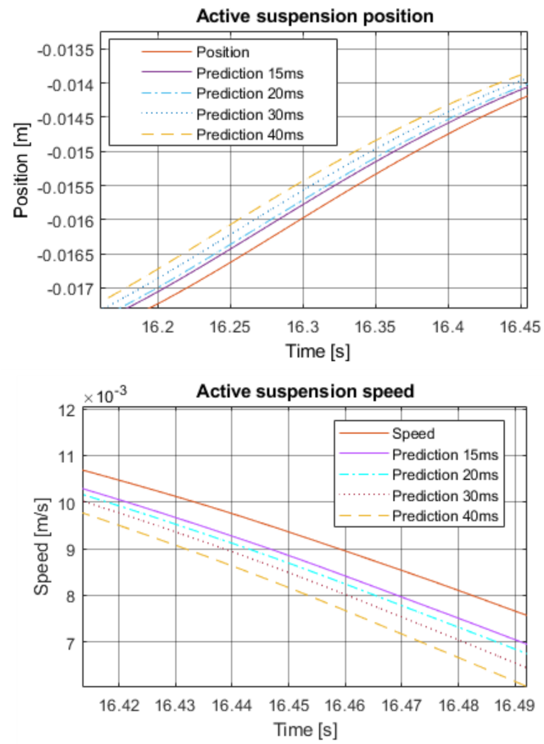

Fig. 4. Predictions used by PPDDC during straight line brake.

where X_{XIL} is the measured value, and $X_{max} - X_{min}$ is the maximum variation range of the variable.

In the tests, the values of NRMSE have been lower than 0.15 for all the relevant variables, therefore validating the precision of the results of the XILforEV framework.

Figure 5 shows the acceleration and the rotational speeds of the car body during the double lane change maneuver. The figure shows the results in the test rigs of the *XILforEV* framework with a pure numerical simulation (MIL), where the test rigs were replaced by models without delay in the communications. The figures show a good correspondence between the simulated and the co-simulated in the distributed *XILforEV* framework. The obtained results show the capability of the proposed algorithm stabilize the co-simulation in presence of the communication delays.

V. Conclusion

The present work describes the co-simulation results for a ride blending controller in the *XILforEV* framework. The tested scenario involves the interaction between traction motors (IWM), active suspension and brakes. The *XILforEV* framework includes test benches connected to a real-time simulator by both LAN and Internet with two distributed co-simulation nodes in Germany and Spain. Due to the distance between them, the effect of the communication delay is not negligible and a compensation is required. This is done by implementing a PPDDC combined with a correction of transferred power in the coupling. The experimental results

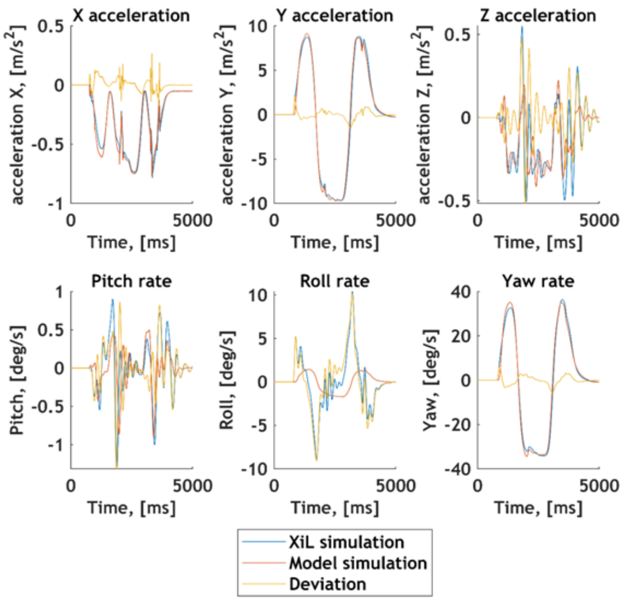

Fig. 5. Comparison of MIL, XIL.

show that the proposed co-simulation architecture is robust in front of the communication delays and can run the tested scenarios with similar results to those obtained with a detailed simulation model without delay. Thanks to its implementation, it was possible to overcome the instability of the XIL co-simulation caused by the delays. The developed compensation algorithm is general enough to be used in other applications. Further activities are focused on using adaptive predictive models in the delay compensator so that it takes into account variations in the system.

References

[1] K.Augsburg, S.Gramstat, R.Horn, V.Ivanov, H.Sachse, B.Shyrokau, ”Investigation of brake control using test rig-in-the-loop technique,” SAE Tech. Paper Ser. 2011.

[2] https://xil.cloud/ (last access 05.01.2022).

[3] V.Ivanov, K.Augsburg, C.Bernad, M.Dhaens, M.Dutré, S.Gramstat, P.Magnin, V.Schreiber, U.Skrt, N.Van Kelecom, ”Connected and Shared X-in-the-Loop Technologies for Electric Vehicle Design,” World Electr. Veh. J. 10, 83 (2019).

[4] W.Niu,K.Song, T.Zhang, ”Analysis of geographically distributed vehicle powertrain system validation platform based on X-in-the-Loop theory,” SAE Tech. Paper Ser. 2017.

[5] Y.Zhang, S.Lu, Y.Yang, Q.Guo, ”Internet-distributed vehicle-in-the-loop simulation for HEVs,” IEEE Trans.Veh. Technol. 2018, 67, 3729–3739.

[6] P.Baumann, M.Krammer, M.Driussi, L.Mikelsons, J.Zehetner, W.Mair, D.Schramm, ”Using the distributed co-simulation protocol for a mixed real-virtual prototype,” IEEE International Conference on Mechatronics, 2019.

[7] R.Liu, M.Mohanpurkar, M.Panwar, R.Hovsapian, A.Srivastava, S.Suryanarayanan, ”Geographically distributed real-time digital simulations using linear prediction,” International Journal of Electrical Power & 415 Energy Systems 84 (2017).

[8] G.Stettinger, M.Benedikt, M.Tranninger, M.Horn, J.Zehetner, ”Recursive FIR-Filter design for fault-tolerant real-time co-simulation,” 25th Mediterranean Conference on Control and Automation (MED) (2017).

[9] W.Chen, M.Klomp, S.Ran, ”Real-time Co-simulation Method Study for Vehicle Steering and Chassis System,” IFAC-PapersOnLine 51 (9) (2018).

[10] G.Stettinger, M.Horn, M.Benedikt, J.Zehetner, ”Model-based coupling approach for non-iterative real-time co-simulation,” European Control Conference (ECC)(2014).

[11] D. López-Echevarría and M. E. Magaña, ”Neural network and observer-based compensation of networked control systems with packet dropout,” *IFAC Proceedings Volumes*, vol.46, no.29 (2013).

[12] J. Alfonso et al., ”Distributed Simulation and Testing for the Design of a Smart Suspension”, SAE Intl. J CAV, vol. 3, no. 2, (2020).

[13] V.Schreiber, K.Augsburg, V.Ivanov, H.Fujimoto, ”Novel Developing Environment for Automated and Electrified Vehicles using Remote and Distributed X-in-the-Loop Technique,” 2020 IEEE Vehicle Power and Propulsion Conference (VPPC)(2020).

[14] M. Benedikt, D. Watzenig, J. Zehetner, and A. Hofer, ”Nepce-a nearly energy-preserving coupling element for weak-coupled problems and co-simulations,” *COUPLED V: proceedings of the V International Conference on Computational Methods for Coupled Problems in Science and Engineering:*, CIMNE (2013).

[15] L.Hott et al. ”Ride Blending Control for AWD Electric Vehicle with In-Wheel Motors and Electromagnetic Suspension,” 2020 IEEE Vehicle Power and Propulsion Conference (VPPC)(2020).

[16] V.Schreiber, V.Ivanov, K.Augsburg, M.Noack, B.Shyrokau, C.Sandu, P.S.Els, ”Shared and distributed X-in-the-loop tests for automotive systems: Feasibility study,” IEEE Access 2018, 6, 4017–4026.

[17] F.Poletti et al., ”Towards high-capacity fibre-optic communications at the speed of light in vacuum”, Nature Photon, vol. 7, no. 4, (2013).

[18] O.Krajsa and L.Fojtova, ”RTT measurement and its dependence on the real geographical distance”, 34th International Conference on Telecommunications and Signal Processing (TSP)(2011).

[19] R.Liu, M.Mohanpurkar, M.Panwar, R.Hovsapian, A.Srivastava, and S.Suryanarayanan, ”Geographically distributed real-time digital simulations using linear prediction”, International Journal of Electrical Power & Energy Systems, vol. 84, (2017).

[20] G.Stettinger, M.Benedikt, M.Tranninger, M.Horn, J.Zehetner, ”Recursive FIR-Filter design for fault-tolerant real-time co-simulation”, 25th Mediterranean Conference on Control and Automation (MED), (2017).

[21] W.Chen, M.Klomp, S.Ran, ”Real-time Co-simulation Method Study for Vehicle Steering and Chassis System”, IFAC-PapersOnLine, vol. 51, no. 9 (2018).

[22] G.Stettinger, M.Horn, M.Benedikt, J.Zehetner, ”Model-based coupling approach for non-iterative real-time co-simulation”, European Control Conference (ECC), (2014).

[23] C.Lehne, V.Ivanov, K.Augsburg, F.Büchner, V.Schreiber, J.Herman, J.Pašič, B. Zavrl, ”Distributed Local X-in-the-Loop Environment – A Tool for Electric Vehicle Systems Design,” The 5th International Electric Vehicle Technology Conference (2021).

[24] G.Stettinger, M.Benedikt, N.Thek, J.Zehetner, ”On the difficulties of real-time co-simulation,” International Conference on Computational Methods for Coupled Problems in Science and Engineering, (2013).