Real-time implementation of yaw rate and sideslip control through individual wheel torques

TRISTANO, Mariagrazia, LENZO, Basilio <http://orcid.org/0000-0002-8520-7953>, XU, Xu <http://orcid.org/0000-0002-9721-9054>, FORRIER, Bart, D'HONDT, Thomas, RISALITI, Enrico and WILHELM, Erik

Real-time implementation of yaw rate and sideslip control through individual wheel torques

Mariagrazia Tristano
Dept. of Engineering and Mathematics
Sheffield Hallam University
Sheffield, UK
mariagrazia.tristano@student.shu.ac.uk
https://orcid.org/0000-0002-6418-5054

Basilio Lenzo
Dept. of Industrial Engineering
University of Padova
Padua, Italy
basilio.lenzo@unipd.it
https://orcid.org/0000-0002-8520-7953

Xu Xu
Dept. of Engineering and Mathematics
Sheffield Hallam University
Sheffield, UK
xu.xu@shu.ac.uk

Bart Forrier
TSVT division
Siemens Digital Industries Software
Leuven, Belgium
bart.forrier@siemens.com

Thomas D'hondt
TSVT division
Siemens Digital Industries Software
Leuven, Belgium
thomas.dhondt@siemens.com

Enrico Risaliti
TSVT division
Siemens Digital Industries Software
Leuven, Belgium
enrico.risaliti@siemens.com

Erik Wilhelm
Head of Research
KYBURZ Switzerland AG
Freienstein, Switzerland
erik.wilhelm@kyburz-switzerland.ch

***Abstract*—Improving vehicle passenger safety is of major importance in modern automotive industry. Within this framework, vehicle stability controllers play a key role, as they actively contribute to maintain vehicle driveability even in potentially dangerous situations. An example of such a controller is Electronic Stability Control (ESC), that brakes individual wheels to generate a direct yaw moment to stabilize the vehicle (e.g. from excessive understeer or oversteer). This paper presents the real-time implementation of a stability controller based on measured (and/or estimated) yaw rate and sideslip angle and on phase-plane related stability criteria. The control strategy is first developed in MATLAB-Simulink environment with a simplified vehicle model. Then, the controller is assessed via software-in-the-loop using a full vehicle model developed in Simcenter Amesim, before implementing it on a real-time platform. Results are promising, endorsing the implementation of hardware-in-the-loop using an Electronic Control Unit.**

***Index Terms*—vehicle dynamics, real-time, control, direct yaw moment, yaw rate, sideslip angle, vehicle stability.**

I. Introduction

Vehicle stability controllers allow to maintain driveability in safety-critical situations, preventing vehicle loss of control (e.g. spinning or drifting). A well-known example of vehicle stability control is the Electronic Stability Program (ESP), also referred to as Electronic Stability Control (ESC). ESC is mandatory in modern passenger cars and it has been shown to significantly contribute to reducing the number of traffic-related fatalities [1].

In [2], van Zanten focuses on the relationship between loss of control and lateral tire forces, proposing an algorithm based on the estimated slip ratio. [3] tackles the issue by acting on individual brake pressures on the wheels in order to prevent excessive lateral forces from causing vehicle rollover. The optimised distribution of wheel torques is at the foundation of [4], where the control strategy to influence the vehicle yawing behaviour is split in a three-level controller. In [5], the benefits of the ESC are combined with those of the active front steering in order to achieve a more efficient tracking of the desired yawing behaviour. In [6], a stability control algorithm operates on an electric vehicle equipped with four in-wheel motors, allowing torque vectoring. A vehicle with two motors (one per axle) is used by Gimondi in [7].

Despite several vehicle yaw control methods existing in the literature, many contributions only propose model-based validations, having limited significance. While deployment on full-scale vehicles might not always be an option, intermediate solutions between model-based and full-scale validation are definitely possible.

The design and deployment of a vehicle control system require an important number of intermediate steps, where hardware is gradually integrated to replace simulation blocks. An overview of the steps to be followed to that end is given by the so-called V-cycle presented in [8], whose original structure is adapted in Fig. (1). Once design requirements are defined, a controller is developed (e.g. in Matlab) and tested on a validated vehicle model. This phase, denoted as "Model-based testing", takes place offline. The subsequent step is "Online testing", in which the individual building blocks are made software-independent and tested as interconnected black boxes, to provide an indication of running time closer to the

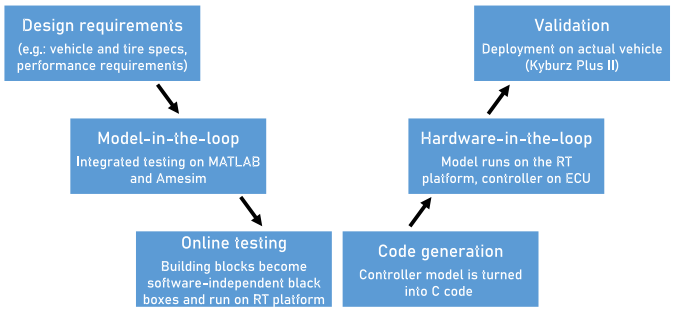


Fig. 1. Development cycle for the full validation of the presented vehicle control system.

one expected for the real-time application. Then, the controller is turned into code to allow "Hardware-in-the-loop testing", featuring a bench that reproduces the complexity of the plant and runs the controller in real time.

This paper presents a yaw rate and sideslip angle stability controller and its current evolution along the aforementioned V-cycle. The final aim will be to implement such controller on the Kyburz PLUS II vehicle ([9]).

Section II presents the control strategy. Section III and IV describe respectively the Model-in-the-loop and the Software-in-the-loop implementations. Results are presented in Section V. Conclusions and future steps are in Section VI.

II. Control strategy

The proposed control framework is shown in Fig. 2 and features three main components:

- **Reference generator.** It makes use of driver inputs and estimated/measured vehicle states to generate a desired reference yaw rate.
- **High-level controller.** It uses the difference between the reference yaw rate and its actual value, provided as a feedback measurement, to compute - by means of, e.g., a Proportional-Integral (PI) controller - an appropriate direct yaw moment action M_z to implement the desired cornering behaviour on the vehicle.
- **Low-level controller.** It allocates torques T_i to the individual wheels to achieve the desired direct yaw moment.

The following subsections dive deeper in the role of the aforementioned components.

A. Reference generator

The handling reference, r_h, is the output of a two-dimensional lookup table: based on the steering wheel angle and the longitudinal velocity values, i.e. δ_SW, v_x respectively, the desired handling behaviour is computed. The handling yaw rate is defined based on a desired cornering response, hence the parameters to be defined are: the linear understeer gradient K_lin, the limit acceleration for the linear range of operation a_y^*, and the maximum achievable acceleration $a_\mathrm{y,max}$. More details can be found in [10], [11].

The stability reference, r_s, is based on the limits of maximum lateral acceleration, defined by Eq. (1), where the lateral acceleration a_y is scaled by a factor $k_\mathrm{s} < 1$ to ensure a sufficient safety margin.

$$r_\mathrm{s} = k_s \frac{a_\mathrm{y}}{v_x} \tag{1}$$

When the driving scenario is deemed safe, the handling reference r_h is the only target, otherwise the stability reference r_s is involved to help the driver regain control of the vehicle.

The stability assessment that regulates the priority of either references comes from the concept of phase-plane stability. Phase portraits are a widely-employed tool in vehicle dynamics (e.g. [12], [13], [14], [15], [16]): they display the changes in time of non-linear systems by illustrating how a given state behaves against another key state, or its rate of change, in the form of so-called trajectories. In the case at hand (Fig. 3) the sideslip angle β is displayed against the yaw rate r. The trajectory behaviour of a number of initial working points within reasonable boundaries is investigated: the points whose evolution settle within certain state boundaries constitute the stability region.

At each time instant, the vehicle working point coordinates (β, r) are estimated and measured (respectively). A unique reference is generated, as a weighted sum of both references by means of a weight factor, ρ (Eq. 2):

$$r_\mathrm{ref} = \rho \cdot r_\mathrm{s} + (1-\rho) \cdot r_\mathrm{h} \tag{2}$$

At each time step, ρ is the result of a smoothed varying weight function depending on two evaluation indexes, one for the vehicle sideslip angle and one for the yaw rate, respectively referred to as I_β and I_r. The indexes, defined in Eq. 3 and Eq. 4, give an indication of the closeness of the working point to the boundaries of the stability region, defined by minimum and maximum values of both the sideslip angle (β_min and β_max in Eq. 3) and the yaw rate (r_min and r_max in Eq. 4). Further indications on how to retrieve such values are provided in [17].

$$I_\beta = 1 - \mathrm{sign}((\beta_\mathrm{max} - \beta)(\beta - \beta_\mathrm{min})) \times \frac{\min(|\beta_\mathrm{max} - \beta|, |\beta - \beta_\mathrm{min}|)}{\frac{\beta_\mathrm{max} - \beta_\mathrm{min}}{2}} \tag{3}$$

$$I_\mathrm{r} = 1 - \mathrm{sign}((\mathrm{r}_\mathrm{max} - \mathrm{r})(\mathrm{r} - \mathrm{r}_\mathrm{min})) \times \frac{\min(|\mathrm{r}_\mathrm{max} - \mathrm{r}|, |\mathrm{r} - \mathrm{r}_\mathrm{min}|)}{\frac{\mathrm{r}_\mathrm{max} - \mathrm{r}_\mathrm{min}}{2}} \tag{4}$$

At each time step the maximum between such two indexes, i.e. the most critical one, is chosen (Eq. 5) and compared to a pre-set threshold value (I_t) to then compute ρ, using Eq. (6).

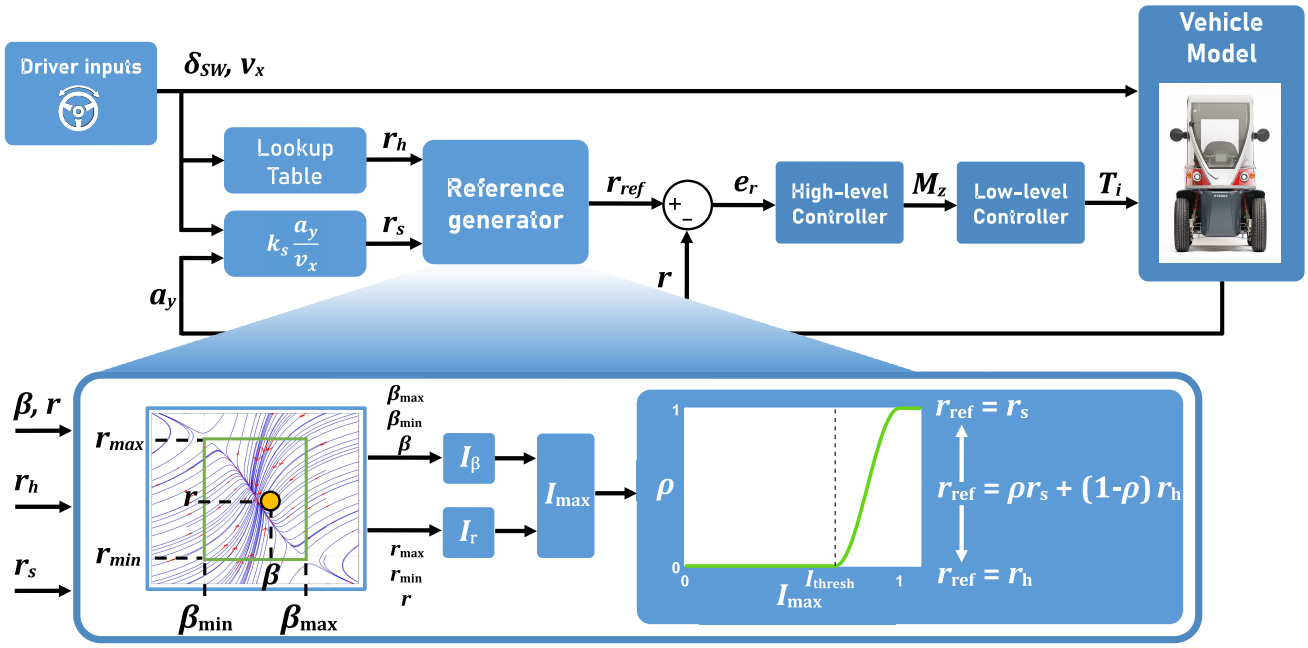


Fig. 2. Control scheme framework.

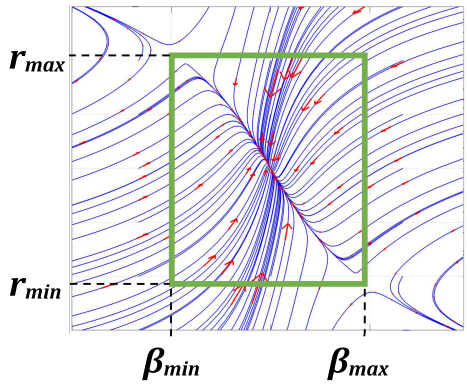


Fig. 3. Generic $\beta-r$ phase portrait featuring orientation vectors for individual trajectories; the parallelogram in bold represents the stability region.

$$I_{\max} = \max(I_\beta, I_\mathrm{r}) \tag{5}$$

$$\rho(I_{\max}) = \begin{cases} 0, & \text{if } 0 \le I_{\max} < I_\mathrm{t} \\ \frac{1}{2}(1 - \cos\left(\pi \frac{I_{\max} - I_\mathrm{t}}{1 - I_\mathrm{t}}\right)), & \text{if } I_\mathrm{t} < I_{\max} \le 1 \end{cases} \tag{6}$$

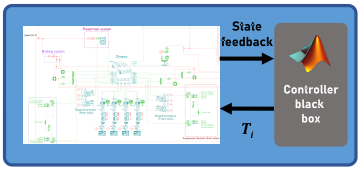
B. High-level controller and low-level controller

The high-level controller is a PI controller with static gains, whose input is the error defined as the difference between the reference yaw rate and the actual yaw rate coming from feedback measurements. The error is multiplied by the coefficient K_p while its integral is multiplied by the coefficient K_i as in Eq. (7), where e indicates the error. Both contributions are then summed to obtain the overall required direct yaw moment M_z, whose role is to adjust the vehicle cornering behaviour according to the selected reference.

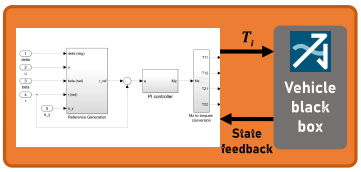
$$M_z = K_p \cdot e + K_i \cdot \int e \, \mathrm{d}t \tag{7}$$

The low-level controller is essentially a torque allocator. In the case-study application (Kyburz PLUS II), only individual braking torques are allowed. So, the torque allocator translates the computed direct yaw moment into an individual braking torque effort to be applied on the wheels.

Intuitively, a positive (anti-clockwise) direct yaw moment can be achieved by assigning more braking torque on the left-side wheels; conversely, a negative (clockwise) direct yaw moment is accomplished by allocating a higher braking torque on the right-side wheels. The torque effort for the left and/or right side is equally split between the front and rear wheel, in order to keep the implementation as simple as possible for faster computation. The torque allocation is hence performed following Eq. (8), where R_w represents the wheel radius, t_v is the vehicle track (front and rear are assumed to be equal) and T_i is the torque on the left or right side, to be then split equally between front and rear wheel.



(a) MATLAB/Simulink environment.



(b) Amesim environment.

Fig. 4. Black-box co-simulation approach.

$$T_i = \begin{cases} 2\dfrac{M_z R_w}{t_v}, & \text{i = left} \quad \text{if} \quad M_z \geq 0 \\ -2\dfrac{M_z R_w}{t_v}, & \text{i = right} \quad \text{if} \quad M_z < 0 \\ 0 \text{ otherwise} \end{cases} \tag{8}$$

III. Model-in-the-loop: the co-simulation environment

The control strategy presented so far is implemented through a single-track vehicle model in MATLAB-Simulink. To ensure a reliable assessment, a co-simulation is put in place so that the actual vehicle model affected by the controller action is a 15 degree-of-freedom model defined on Simcenter Amesim. The number of degrees of freedom corresponds to achieving a high level of detail in the simulation environment including aerodynamics information and dependency on the road condition and suspension phenomena among others. A set of virtual sensors allows to measure any quantity of interest.

Simcenter Amesim can communicate with MATLAB-Simulink, and viceversa, through co-simulation interfaces. The interfaces are boxes featuring a number of inputs and outputs: they can be thought of as black-box reproductions of the vehicle model in the controller environment and conversely, of the controller model in the vehicle environment. For example, in the vehicle environment, the Amesim co-simulation block will receive as input the vehicle model outputs and provide the vehicle model with the controller outputs. The data exchange in Amesim and MATLAB-Simulink environment is summarised respectively in Fig. 4(a) and 4(b), where the respective inputs and outputs are listed as well.

The individual braking torques provided by the controller black-box in Amesim are summed to the ones already generated within Amesim, hence the braking action does not override the regular braking actions performed by the driver.

The terminology "co-simulation" implies that both software packages are run concurrently on a unique simulation, so the simulation run parameters need to match. The run parameters are summarised in Table I: a fixed-step solver is chosen (hence the need for a time step) featuring an order 2 Runge-Kutta integration algorithm.

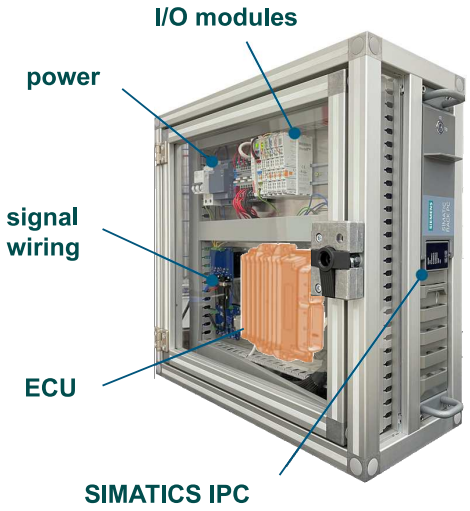


Fig. 5. Real-time test environment.

TABLE I
Run parameters for co-simulation.

Parameter	Description	Value	Unit
t_s	Simulation start time	0	s
t_f	Simulation end time	7	s
Δt	Integration time step	1	ms

IV. Online testing: the real-time platform

After the offline Model-in-the-loop validation is completed, gradual hardware integration in the framework is required. The first introduced component is a Real-Time (RT) platform, displayed in Fig. 5. The RT platform is capable of running the entire framework online by itself, which entails the possibility of an additional degree of middle-ground testing while transitioning from the offline domain to the real-time domain. More specifically, the RT platform uses source code generated in Simulink and requires no algebraic loops.

Figure 5 features an overall description of the platform components. More specifically:

- **I/O modules** take care of not only receiving torques from the controller but also inputs from the driver model, while being connected to the CAN bus and to the sensors that will provide state feedback in the final testing stages.
- **SIMATICS IPC** is a high-performance computer, constituting the computational core of the RT machine and guaranteeing the real-time functionalities of the entire hardware platform.

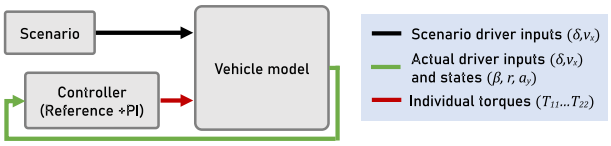


Fig. 6. Overall control scheme in FMI standard.

- **Electronic Control Unit (ECU)** runs the reference generator and both the high-level and low-level controllers in real-time. The device is flashed using the source code generated by Simulink. At this stage, the ECU is kept idle but will be used in the subsequent phase of Hardware-in-the-loop.

The RT machine is not software-specific: any model can run on it. However for that to happen, the simulation model has to be translated to a specific standard, so that it can be run concurrently with other pieces of software as if they were interfaced black boxes. The aforementioned standard is called Functional Mock-up Interface (FMI [18]), a free standard that defines the connected building blocks (called Functional Mock-up Units, in short FMUs) of a Model-in-the-loop framework into a number of containers and interfaces, using a combination of XML files, binaries and C code zipped into a single file. As a consequence, the software unit becomes a black box featuring a number of inputs and outputs, connected in turn to other black box units.

The value of employing the FMI standard is the ability to take any piece of software and being able to run it on a real-time target platform (the RT machine in the case at hand), independently of the software the unit was created in: this makes the testing universal, rather than platform-dependent.

Once converted to comply with the FMI standard, the overall scheme for the control structure looks like Fig. 6, where the signals are labelled on the right side and the three macroblocks are reduced to black boxes having inputs and outputs.

V. Results

To observe the vehicle behaviour when its stability is challenged, a lane change manoeuvre is performed: i) in a mild scenario, a sinusoidal steering input is applied at a constant speed of 25 m/s; ii) in a challenging scenario, a sinusoidal steering input is applied while the vehicle is traveling at 33 m/s and is experiencing a 3 m/s^2 longitudinal acceleration. The steering wheel amplitude is of 50 degrees and its frequency is 0.5 Hz. Figure 7 shows the speed and steering angle time histories.

Results for the mild scenario are provided in terms of yaw rate and sideslip angle, respectively in Fig. 8 - 9, where the performance of the uncontrolled vehicle is compared to the controlled one. Figure 8 features the reference yaw rate as well, giving a visual indication on how close are the expected and actual yaw rate behaviour.

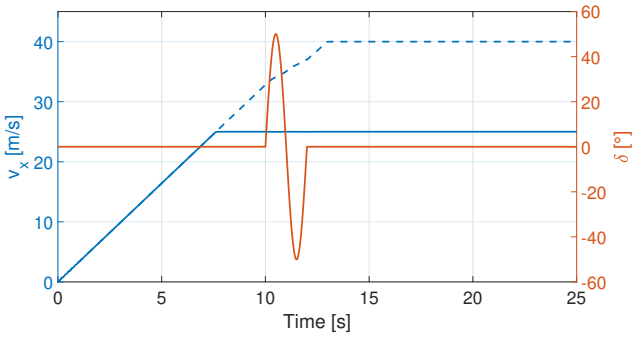


Fig. 7. Single lane change manoeuvre inputs for both presented scenarios.

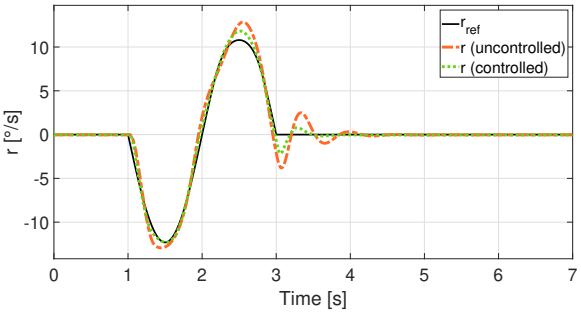


Fig. 8. Yaw rate comparison for the mild scenario.

An indication on the mild nature of the manoeuvre is provided by Fig. 10, where the evaluation indexes I_β and I_r are shown alongside the weight factor ρ: the evaluation indexes never exceed the threshold values (red dashed lines) which never prompts ρ to change from 0. This indicates a safe scenario.

As for the challenging scenario, results reported in Figure 11 and 12 show the behaviour of the yaw rate and sideslip angle, respectively: while the uncontrolled vehicle clearly shows instability and loss of control, the controlled vehicle is capable of safely completing the manoeuvre. Further proof of the instability is given by the stability index of the yaw rate (Fig. 13), exceeding the allowed threshold and prompting the variable weight factor ρ to increase and prioritise the stability

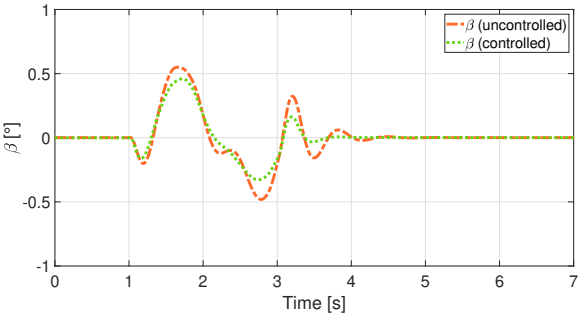


Fig. 9. Sideslip angle comparison for the mild scenario.

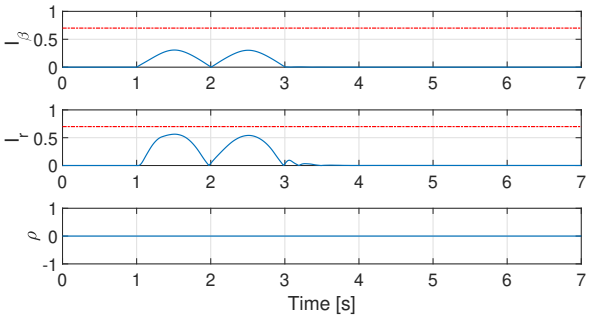


Fig. 10. Stability indexes and varying weight factor for the mild scenario.

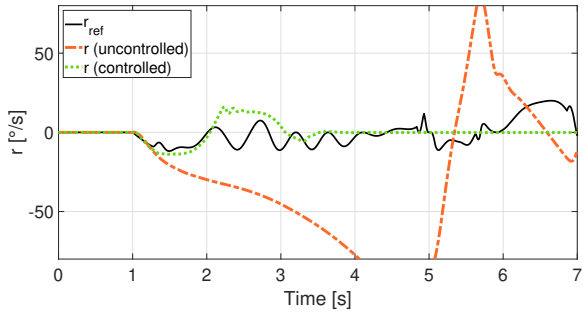


Fig. 11. Yaw rate comparison for the challenging scenario.

reference.

VI. Conclusions

This paper dealt with the implementation of a vehicle stability control through a direct yaw moment calculated based on current yaw rate and sideslip angle. A co-simulation involving MATLAB-Simulink and Simcenter Amesim was developed, to ensure a reliable assessment of the developed control strategy, before setting up a validation on a real-time platform. Immediate future steps include connecting the RT machine to an Electronic Control Unit (ECU), able to run the reference generator along with the high-level and low-level controllers to perform Hardware-in-the-loop testing.

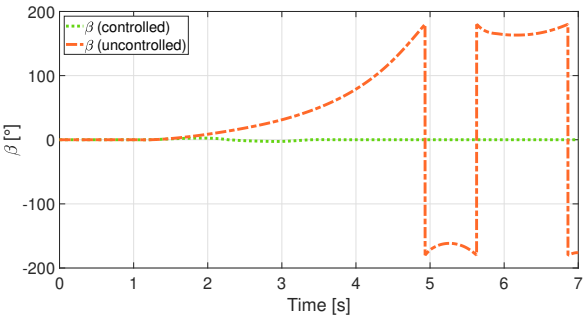


Fig. 12. Sideslip angle comparison for the challenging scenario.

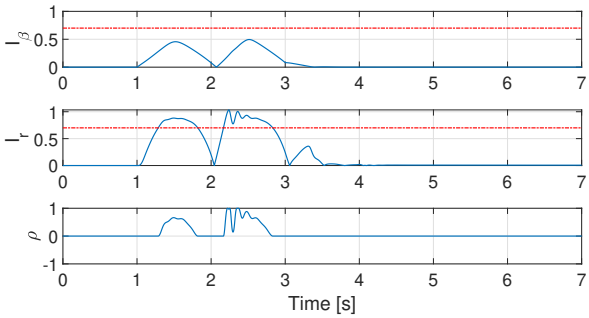


Fig. 13. Stability indexes and varying weight factor for the challenging scenario.

References

[1] Webb, C. N. Estimating lives saved by electronic stability control, 2011-2015 (No. DOT HS 812 391) (2017).

[2] Van Zanten, A. Bosch ESP systems: 5 years of experience. *SAE Transactions*. pp. 428-436 (2000)

[3] Lu, J., Messih, D., Salib, A. & Harmison, D. An enhancement to an electronic stability control system to include a rollover control function. *Sae Transactions*. pp. 303-313 (2007)

[4] Chen, B. & Kuo, C. Electronic stability control for electric vehicle with four in-wheel motors. *International Journal Of Automotive Technology*. **15**, 573-580 (2014)

[5] Guo, J., Chu, L., Liu, H., Shang, M. & Fang, Y. Integrated control of active front steering and electronic stability program. *2010 2nd International Conference On Advanced Computer Control*. **4** pp. 449-453 (2010)

[6] Zhai, L., Sun, T. & Wang, J. Electronic stability control based on motor driving and braking torque distribution for a four in-wheel motor drive electric vehicle. *IEEE Transactions On Vehicular Technology*. **65**, 4726-4739 (2016)

[7] Gimondi, A., Corno, M. & Savaresi, S. A Yaw Rate Based Stability Control for Under-Actuated Vehicles. *2020 IEEE 23rd International Conference On Intelligent Transportation Systems (ITSC)*. pp. 1-6 (2020)

[8] Forsberg, K. & Mooz, H. The relationship of system engineering to the project cycle. *INCOSE International Symposium*. **1**, 57-65 (1991)

[9] https://kyburz-switzerland.ch/en/passenger_vehicles/plus

[10] De Novellis, L., Sorniotti, A. & Gruber, P. Driving modes for designing the cornering response of fully electric vehicles with multiple motors. *Mechanical Systems And Signal Processing*. **64** pp. 1-15 (2015)

[11] Mangia, A., Lenzo, B., & Sabbioni, E. (2021). An integrated torque-vectoring control framework for electric vehicles featuring multiple handling and energy-efficiency modes selectable by the driver. Meccanica, 56(5), 991-1010.

[12] Zhang, L., Ding, H., Guo, K., Zhang, J., Pan, W., & Jiang, Z. (2019). Cooperative chassis control system of electric vehicles for agility and stability improvements. IET Intelligent Transport Systems, 13(1), 134–140.

[13] Ono, E., Hosoe, S., Tuan, H. D., & Doi, S. (1998). Bifurcation in vehicle dynamics and robust front wheel steering control. IEEE Transactions on Control Systems Technology, 6(3), 412–420.

[14] Farroni, F., Russo, M., Russo, R., Terzo, M., & Timpone, F. (2013). A combined use of phase plane and handling diagram method to study the influence of tyre and vehicle characteristics on stability. Vehicle System Dynamics, 51(8), 1265-1285.

[15] Hao, Z., Xian-sheng, L., Shu-ming, S., Hong-fei, L., Rachel, G., & Li, L. (2011). Phase plane analysis for vehicle handling and stability. International Journal of Computational Intelligence Systems, 4(6), 1179–1186.

[16] Bobier-Tiu, C. G., Beal, C. E., Kegelman, J. C., Hindiyeh, R. Y., & Gerdes, J. C. (2019). Vehicle control synthesis using phase portraits of planar dynamics. Vehicle System Dynamics, 57(9), 1318–1337.

[17] Guo, N., Zhang, X., Zou, Y., Lenzo, B., Du, G. & Zhang, T. A supervisory control strategy of distributed drive electric vehicles for coordinating handling, lateral stability, and energy efficiency. *IEEE Transactions On Transportation Electrification*. **7**, 2488-2504 (2021)

[18] https://fmi-standard.org/