Article Identifier
DOI: 10.22032/dbt.58871
URN: urn:nbn:de:gbv:ilm1-2023isc-085:6

60th ILMENAU SCIENTIFIC COLLOQUIUM
Technische Universität Ilmenau, 4 – 8 September 2023
URN: urn:nbn:de:gbv:ilm1-2023isc:1

SELF-ADAPTING MOTION CUEING ALGORITHM BASED ON A KINEMATICS REFERENCE MODEL

Büchner, Florian[1]; Jestädt, Lukas[2, 3]; Ivanov, Valentin[1]; Bachmann, Thomas[1]

[1] TU Ilmenau, Department of Mechanical Engineering, Automotive Engineering Group, 98693 Ilmenau, Germany
[2] Work performed formerly under: TU Ilmenau, Department of Mechanical Engineering, Automotive Engineering Group, 98693 Ilmenau, Germany
[3] Current work relationship: Dr. Ing. h.c. F. Porsche AG, 71287 Weissach, Germany

ABSTRACT

Due to a number of advantages over traditional development methods, the importance of dynamic driving simulators in automotive research and development has grown continuously in recent years. Motion simulation via motion cueing algorithms contributes significantly to the driving experience and provides the driver with valuable information about the current driving dynamics. The adaptation and tuning process of these algorithms can be difficult and time-consuming tasks. It needs to be repeated after changes to the vehicle or driving scenario. This paper discusses and presents an adaptive or rather self-adapting motion cueing algorithm (MCA) concept. The approach is based on the integration of a kinematic reference model to dynamically and adaptively adjust the motion behavior dynamically and adaptively. This concept allows to reduce the parameter tuning effort drastically in long term, since the algorithm can adapt itself to different conditions such as vehicle type, driving situation, or driver behavior. In the following, the proposed algorithm structure is explained and illustrated. The advantages of the proposed MCA are demonstrated by an experimental comparison with a classical algorithm. Thereby it is shown how a self-adaptation of the algorithm can proceed and how to avoid violation of workspace boundaries.

Index Terms - Driving Simulator, Motion Cueing Algorithm

1. INTRODUCTION

With the increasing number of new driver assistance systems, the system complexity of modern vehicles continues to rise. To keep the required effort in terms of testing time within reasonable limits, the share of the simulation and test bench experiments over the whole vehicle testing process growths continuously against the field tests.
With the introduction of dynamic driving simulators, a powerful development tool has been created that can be used for a wide range of automotive applications. The number of such driving simulators equipped with coupled motion systems is constantly increasing and is being used more and more in the various development stages of automotive engineering.
An introduction of different driving simulator systems is given e.g. in [1], [2] or [3]. Usually, in industry and research, a hexapod motion system with six degrees of freedom and attached vehicle mockup is often used, see [4], [5], [6]. The system combines acceptable simulation characteristics and minimal installation space requirements. Such dynamic driving simulators are used e.g. in the field of acceptance evaluation of novel driver assistance systems. Especially

developments of new vehicle safety systems and algorithms can be supported. In this way, new algorithms can be tested at an early stage in combination with a real driver. Driver behavior can be studied this way efficiently, locally and without much effort. In this context, the main challenge in realizing dynamic driving simulators is to achieve correct motion simulation while optimally using the available operating range.
The movements implemented by the motion system are determined on the basis of the current driving state as a result of so-called motion cueing algorithms (MCA). In the process, it must be ensured that no workspace limits are violated and no incorrect motion perceptions, so-called false cues, are provided. MCA optimization is of major interest especially for smaller driving simulators with a limited workspace.
High requirements are demanded of all subsystems of the simulation environment to ensure that the test driver in the driving simulator interacts in a similar way compared to real driving situations. One of the key challenges in this context is to achieve adequate accuracy in the motion simulation of highly dynamic vehicle accelerations and motions. The system actuation to be performed by the motion system is determined on the basis of the current driving state by the MCA. [7] In this process, the MCA needs to keep the platform motions within the limited workspace of the motion system.
To provide an optimal representation of the virtual vehicle state of motion to the driver, a driving simulator would need to provide neutral motion and acceleration behavior. However, this is difficult due to the limited range of motion. Especially when simulating different classes of vehicles or different driving profiles. [5]
Conventional MCA's have to be adapted specifically to the respective scenario in order to make optimal possible usage of the usually very limited workspace. In the following, a modified algorithm approach for automated online adaptation of a conventional MCA is presented so that case-specific adjustments are not necessary. The approach presented and discussed here in the publication is based on the elaborations of [8]. The design is first presented in concept and then verified by experimental tests. In this context a hexapod driving simulator based at Thuringian Innovation Center for Mobility (ThIMo) at Technical University of Ilmenau was used. The simulator consists of a hexapod motion system and a coupled seating mock-up with an active steering wheel and an active pedal unit. The motion system provides the driver with realistic vehicle motion cues based on it's six degrees of freedom.

2. MOTION CUEING

Many different types of MCA's have been developed and introduced [7]. However, the most commonly used concept is still the classical washout algorithm [9], which uses a combination of linear filters to determine the target platform displacements from the driving simulation inputs. In the literature, many studies have been done on the classical washout algorithm. There are also many different representations to be found, see: [10], [11], [12] [13] or [5]. The structure of a typical classical washout algorithm, which was used as a basis for research and further developments in this publication, can be found in Figure 1.

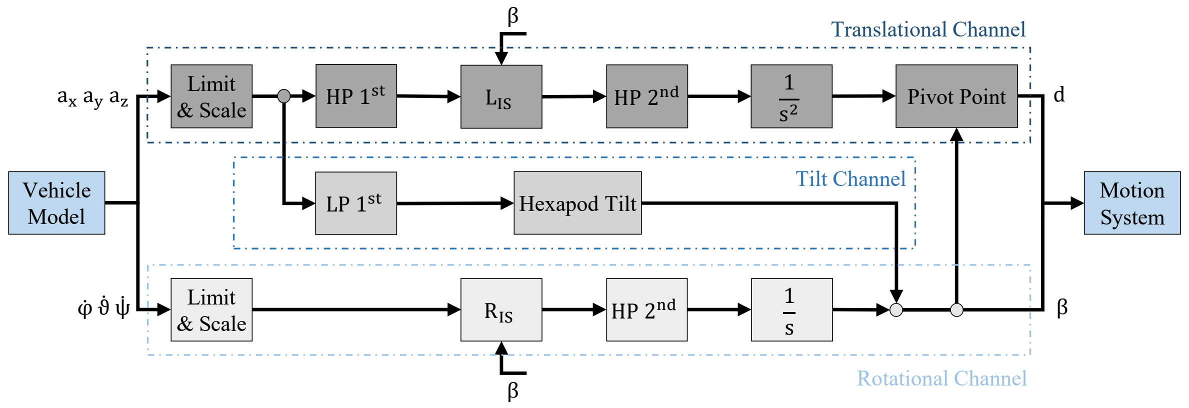


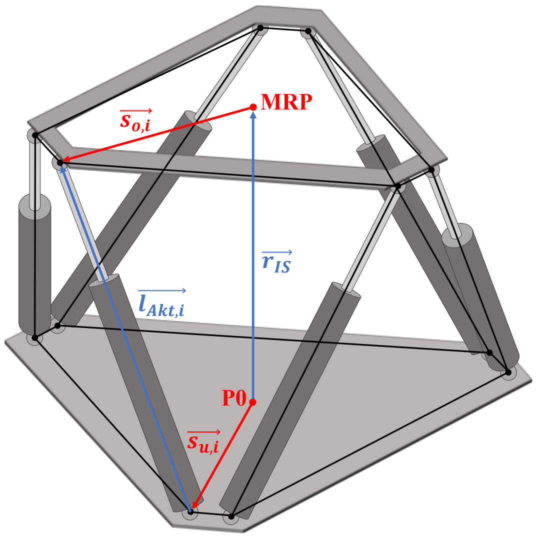
Figure 1: Scheme of a classical washout filter based MCA

The relatively simple structure of the general classical washout algorithm allows transparent and fast tuning of the filter parameters to the respective application. Since these tuning methods must always consider a worst-case situation, classical washout algorithms often do not provide the necessary level of sensitivity in regular driving situations. Another problem is the fact that the tuning of the algorithm has to be repeated after changes of the scenario like maneuver, vehicle type or driver behavior.
In accordance with this, the first adaptive algorithm structures were developed very early in the research process. As an example, especially the Coordinated-Adaptive-Algorithm can be mentioned, which has already been extensively studied in the literature, cf. [11] and [12].
The basic algorithm design is similar to the classical washout, but the filter blocks have been replaced by adaptive filters. These filter characteristics can be optimized during operation to minimize a predefined cost function [7]
However, solving the cost functions requires a considerable amount of additional computing power. Also the tuning of the adaptive filter parameters becomes more difficult. As shown in [14], an inappropriate choice of parameters can also lead to an unstable behavior of the adaptive algorithm.
This paper presents the results and the adaptive approach proposed in [8] to extend a classical washout. However, the adaptive MCA approach has been slightly modified and tested again using adequate studies. The result is a simple but efficient approach for an extension of the classical washout algorithm, which allows a dynamic or adaptive adjustment to the respective driving scenario and thus a self-adaptive operation.
The proposed approach does not require a cost function and thus has no special computational requirements. The goal was to keep the overall simple system structure and the transparent tuning process so that a easy and efficient tuning based on the existing boundary conditions is possible. Nevertheless, the overall sensitivity and the scenarios usability should be increased compared to the classical approach. Due to the adaptive behavior, a one-time configuration of the algorithm parameters should be sufficient to handle a variety of types of driving scenarios. [8]

3. KINEMATIC REFERENCE MODEL

In order to consider the system limits, it is necessary to implement information about the present system state and the current displacements of the system. Therefore, it has become useful to integrate a kinematics model of the hexapod platform into the adaptive concept. Using the kinematics model, the actual actuator lengths can be calculated from the given translational and rotational displacements of the motion platform. Knowing the current actuator excursions as

well as the correspondent excursion limits for each actuator enables an estimation about the remaining platform workspace in each simulation step. Especially critical situations can be detected at an early stage.
The calculated actuator lengths and velocities are subsequently used to calculate the adaptive adjustments for the algorithm parameters. This should provide a valuable addition, since the various degrees of freedom are coupled in hexapod systems. This means that a given displacement along one degree of freedom may or may not be critical to the system limits depending on the displacements along the other degrees of freedom. Without a model of the system kinematics, it is not possible to consider this factor, which would result in a higher required margin of safety and thus lower accuracy. The kinematics model itself is based on some rather simple geometric considerations, which have already been documented in various scientific publications, see [15] or [7].
Figure 2 shows the kinematic dependencies on a typical six degree of freedom hexapod system required for the calculation of a kinematic model. All required formula symbols can be found in Table 1 in section 6. The length of each actuator leg can be calculated based on $\overrightarrow{s_{o,i}}$ and $\overrightarrow{s_{u,i}}$, which are constant vectors defined by the platform's geometry, in combination with $\overrightarrow{r_{IS}}$, which describes the current translational system displacement. It has to be taken into account, that the orientation of $\overrightarrow{s_{o,i}}$ changes dependant on the rotational displacement of the platform. The obtained formulas result in equation (1) and equation (2).


$\overrightarrow{s_{o,i}}$
MRP
$\overrightarrow{r_{IS}}$
$\overrightarrow{l_{Akt,i}}$
P0
$\overrightarrow{s_{u,i}}$
Figure 2: Length calculation for one of the actuator legs

$$\overrightarrow{l_{Akt,i}} = -\overrightarrow{s_{u,i}} + \overrightarrow{r_{IS}} + \overrightarrow{s_{o,i}} \quad | \quad i = 1 \ldots 6 \tag{1}$$

$$\left|\overrightarrow{l_{Akt,i}}\right| = \sqrt{{l_{Akt,i,x}}^2 + {l_{Akt,i,y}}^2 + {l_{Akt,i,z}}^2} \quad | \quad i = 1 \ldots 6 \tag{2}$$

Since the valid range of values for the actuator lengths used is known, this can be used to evaluate whether the desired state of cylinder excursion can be realized without violating the system limits in each simulation step.

4. ENHANCED ADAPTIVE MOTION-CUEING ALGORITHM

To achieve an adaptive behavior of the overall algorithm, the kinematics model has to be integrated into the classical washout algorithm [8]. Figure 3 shows the structure of the proposed algorithm. Compared to the explanations from [8], the approach was extended in the context of the publication and can now adaptively optimize not only the translation and tilt path but also the rotation path. The platform kinematics model uses the actual platform displacements along the six degrees of freedom as inputs to calculate the corresponding actuator excursions and actuator velocities. These values are passed on to a high-level calculation unit based on [8], which can be used to adapt various parameters of the classical washout algorithm according to the existing driving situation. This feedback loop gets activated when the behavior of the actuator lengths or actuator velocities indicates that a critical situation is about to occur. The parameters that should be changed by the high-level calculation unit includes the scaling factors on one hand and different filter parameters on the other hand.

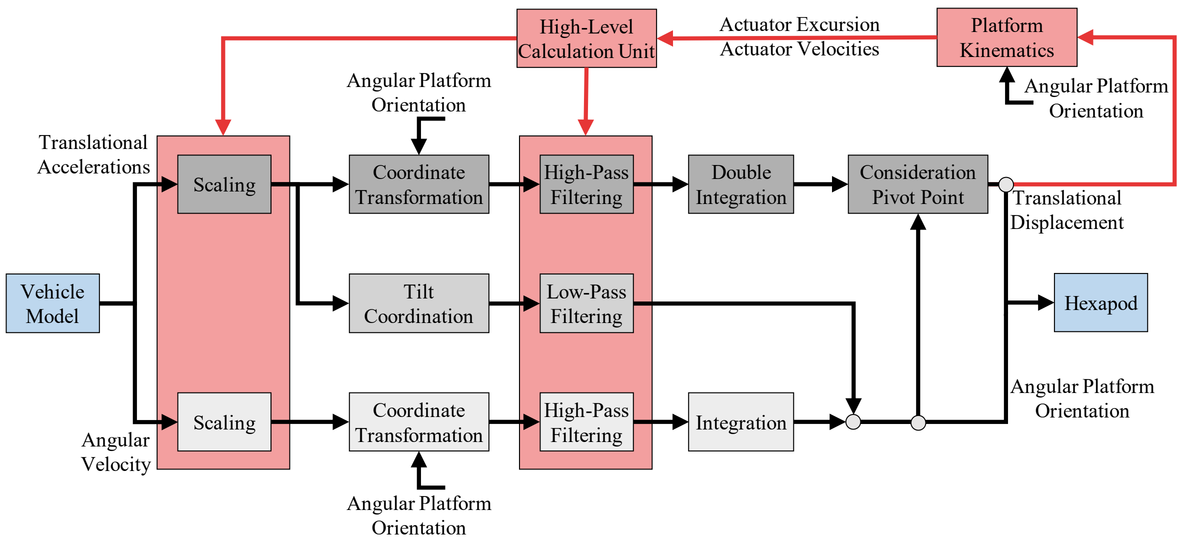

High-Level Calculation Unit
Actuator Excursion
Actuator Velocities
Platform Kinematics
Angular Platform Orientation
Angular Platform Orientation
Translational Accelerations
Scaling
Coordinate Transformation
High-Pass Filtering
Double Integration
Consideration Pivot Point
Translational Displacement
Vehicle Model
Tilt Coordination
Low-Pass Filtering
Hexapod
Angular Platform Orientation
Angular Velocity
Scaling
Coordinate Transformation
High-Pass Filtering
Integration
Angular Platform Orientation

Figure 3: Overall structure of the adaptive MCA extended based on [8]

Previous adaptive approaches use the principle of continuously changing and adjusting algorithm parameters. It could be argued that this helps to achieve high sensitivity without violating the system limits, but it also compromises the proportionality that the driver might feel between different driving maneuvers. For example, a strong acceleration peak may not be significantly different from a weaker one, as the system adapts during the critical situation and then returns to the original parameters when the situation is over.
In this contribution, a different approach is used for the concept of adaptive motion cueing based on [8]. First, the tuning parameters are set to achieve the desired sensitivity in normal and non-critical driving situations. For situations that are evaluated as a risk to the actuator limits, the algorithm adjusts scales and filter parameters in such a way so that the motion system remains within its workspace. Instead of restoring the original values after the critical situation, the calculation unit saves the critical state. The calculated parameters are then saved, even when the critical situation is over. This means not only that the overall sensitivity is reduced after critical situation, but also that proportionality between any driving situations can be ensured even after adaptive downscaling. Thus, the driving simulator can self-adapt to any situation step by step. Based on a preliminary run-in procedure, the necessary system parameters can thus be determined automatically. Afterwards, valid driving tests can be carried out. Necessary optimization times by the developers can be avoided as a result of these procedures.
Detecting the occurrence of a maneuver that could violate the system limitations can still represent a complex task. The coordinated-adaptive algorithm uses a cost function that takes into account the target platform displacements and velocities along the different degrees of freedom. For the concept presented, this task will be accomplished as a result of evaluating the

signals generated exclusively by the kinematics model. The relevant values are the actuator excursions as well as the actuator velocities. For example, an actuator can be excursed almost to its maximum stroke length, but if the piston velocity towards the limit is small, the situation is not critical. By the same way, a high piston velocity is also not considered critical if the excursion is small at the relevant moment. To account for these considerations, a 3D map was created in [8] that assigns each platform actuator a specific adaptive scale factor depending on the current excursion and velocity. The parameters were further tuned and adjusted as part of the work. The actuator that produces the smallest scaling is to be considered as the critical and therefor relevant one. Figure 4 shows how such a 3D map could be configured. The adaptive scaling factor calculated in this way is then applied to the input scaling of the classical washout algorithm. The same procedure is also implemented for the adaptive scaling of the filter parameters. For the algorithm to work as desired, a one-time precise tuning and adaptation of this map to the specific driving simulator is necessary.

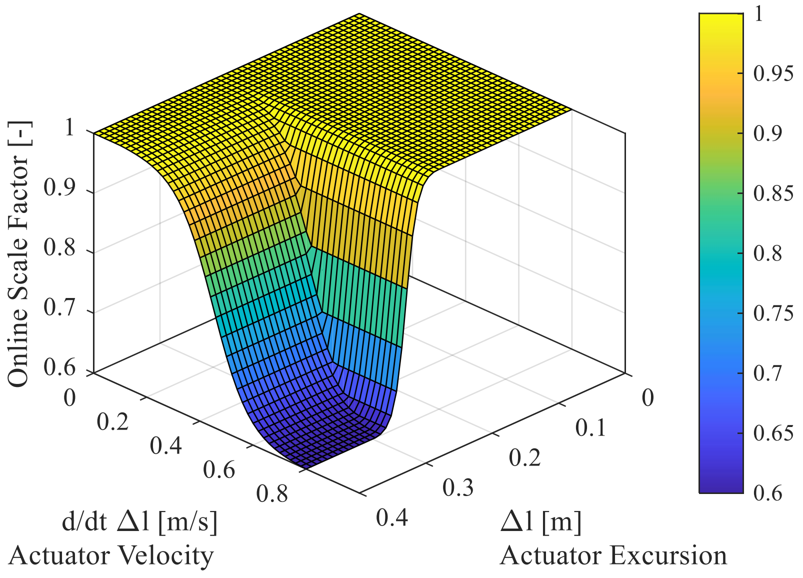


Figure 4: Adaptive scale factor calculation based on actuator excursion and velocity

Numerous driving tests were carried out in the context of this publication on the dynamic driving simulator to validate the functionality. In the following, the section of a racetrack maneuver with a driver model is considered as an example.

Figure 5 shows the variation over time of the critical actuator of the platform during the maneuver. This involved driving through curves with fast load changes and strong adjustment braking. The maneuver was first performed without the adaptive downscaling and then repeated again with activated proposed high level scaling mechanism. In both runs, the maneuver was performed by an identically parameterized driver model to ensure comparability. In agreement with Figure 4, the black lines in the upper diagram in Figure 5 illustrate the corridor where downscaling is triggered. The solid line corresponds to the actuator limit. A violation of this limit results in a significant jerk and consequently in significant false cues for the driver. During the time period presented, the adaptive scaling was triggered several times. A violation of the system limits could be avoided in this way, while the actuator without the mechanism ran into its limit buffers after 17 seconds. Avoiding the actuator limits not only prevents incorrect motion indications, it also protects the servo cylinders and thus increases the lifetime of the overall system.

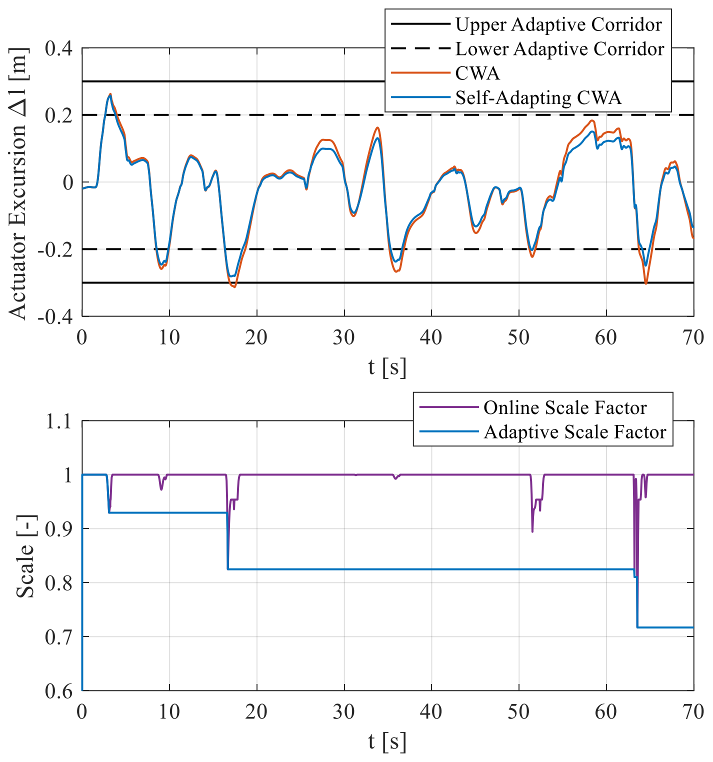


Figure 5: Comparison of the critical actuator excursion between activated and deactivated high level scaling mechanism during an exemplary dynamic driving situation

5. SUMMARY AND OUTLOOK

The objective of the MCA presented in this paper was to combine the advantages of two established concepts (Classical Washout Algorithm & Coordinated Adaptive Algorithm). This aim was achieved in a high degree. The developed structure from [8] has been modified in the context of this publication and tested again on the basis of numerous experimental studies. It keeps the simplicity and transparency of the classical washout, while additional components now allow for adaptive behavior. Adjustments to the algorithm parameters can thus be made quickly, if necessary, and with direct and transparent influence on the system behavior. Furthermore, with the implemented high-level computing unit, a self-adaptive behavior of the driving simulator can be achieved by storing the computed adaptive scaling factors in critical situations and applying them to the entire maneuver in the further course. Based on a run-in procedure, the driving simulator can self-adapt to any driving scenario and to any vehicle model. Experiments have shown that the presented system structure needs to be parameterized and tuned only once to handle all possible scenarios from the driving simulation without violating the system boundaries. This includes changes in vehicle type, road course, or driver behavior. A driving simulator using the presented algorithm structure could thus be used on various events for different purposes without having to change the system parameters. Conventional motion cueing concepts, in contrast, require either time-consuming adjustment times or re-tuning of the algorithm after changes in the driving scenario. Furthermore, the proposed adaptations offer the advantage of a low computational cost, since no more cost functions have to be solved. Accordingly, however, the 3D map of the adaptive high-level calculation unit must be determined in advance once only for the specific driving simulator and the given system limits.

6. LIST OF SYMBOLS

Table 1: List of symbols

Symbol	Unit	Description
i	-	Actuator number
$\overrightarrow{l_{Akt,i}}$	m	Vector between lower and upper actuator joint point
MRP	-	Motion Reference Point of the platform
$\overrightarrow{r_{IS}}$	m	Vector between P0 and MRP
$P0$	-	Projection of the MRP onto the lower platform plane in neutral position
$\overrightarrow{s_{o,i}}$	m	Vector between MRP and upper actuator joint point
$\overrightarrow{s_{u,i}}$	m	Vector between P0 and lower actuator joint point
x_P	m	Longitudinal platform displacement
y_P	m	Lateral platform displacement
z_P	m	Vertical platform displacement

REFERENCES

[1] E. Blana, “A Survey of Driving Research Simulators Around the World,” Working Paper, Institute of Transport Studies, University of Leeds, Leeds, UK Working Paper 481, 1996. Accessed: Apr. 6 2023. [Online]. Available: https://eprints.whiterose.ac.uk/2110/1/ITS170_WP481_uploadable.pdf

[2] N. Mohajer, H. Abdi, K. Nelson, and S. Nahavandi, “Vehicle motion simulators, a key step towards road vehicle dynamics improvement,” *Vehicle System Dynamics*, vol. 53, no. 8, pp. 1204–1226, 2015, doi: 10.1080/00423114.2015.1039551.

[3] L. Bruck, B. Haycock, and A. Emadi, “A Review of Driving Simulation Technology and Applications,” *IEEE Open Journal of Vehicular Technology*, vol. 2, pp. 1–16, 2021, doi: 10.1109/OJVT.2020.3036582.

[4] B. D. C. Augusto and R. J. L. Loureiro, “Motion cueing in the Chalmers driving simulator: A model predictive control approach,” Master of Science Thesis, Department of Signals and Systems, Chalmers University of Technology, 2009. Accessed: Apr. 4 2023. [Online]. Available: https://publications.lib.chalmers.se/records/fulltext/98871.pdf

[5] W. Brems, *Querdynamische Eigenschaftsbewertung in einem Fahrsimulator*. Wiesbaden: Springer Fachmedien Wiesbaden, 2018.

[6] Y. R. Khusro, Y. Zheng, M. Grottoli, and B. Shyrokau, “MPC-Based Motion-Cueing Algorithm for a 6-DOF Driving Simulator with Actuator Constraints,” *Vehicles*, vol. 2, no. 4, pp. 625–647, 2020, doi: 10.3390/vehicles2040036.

[7] M. Fischer, “Motion-Cueing-Algorithmen für eine realitätsnahe Bewegungssimulation,” in *Berichte aus dem DLR-Institut für Verkehrssystemtechnik: Band 5*, Braunschweig: Deutsches Zentrum für Luft- und Raumfahrt, 2009.

[8] L. Jestädt, “Inbetriebnahme eines dynamischen Fahrsimulators zur Untersuchung des Bewegungsverhaltens bei Einsatz eines eigenen Motion-Cueing-Algorithmus,” Masterarbeit, Fakultät für Maschinenbau. Fachgebiet Kraftfahrzeugtechnik, TU Ilmenau, Ilmenau, 2019.

[9] F. Colombet, M. Dagdelen, G. Reymond, C. Pere, F. Merienne, and A. Kemeny, “Motion cueing: What is the impact on the driver’s behavior,” in *Proceedings of the Driving Simulation Conference 2008: DSC 2008*, 2008, pp. 171–181.
[10] B. Conrad and S. F. Schmidt, “Motion drive signals for piloted flight simulators,” *NASA Contractor Report NASA-CR-1601*.
[11] L. D. Reid and M. A. Nahon, “Flight simulation motion-base drive algorithms: part 1. developing and testing equations,” UTIAS Report, No. 296, 1985.
[12] M. A. Nahon and L. D. Reid, “Simulator motion-drive algorithms - A designer's perspective,” *Journal of Guidance, Control, and Dynamics*, vol. 13, no. 2, pp. 356–362, 1990, doi: 10.2514/3.20557.
[13] A. H. J. Jamson, “Motion cueing in driving simulators for research applications,” Dissertation, Institute for Transport Studies, University of Leeds, Leeds, UK, 2010.
[14] J. Kirdeikis, “Evaluation of nonlinear motion-drive algorithms for flight simulators,” UTIAS Technical Note, No. 272, 1989.
[15] R. J. Telban, W. Wu, F. M. Cardullo, and J. A. Houck, “Motion Cueing Algorithm Development: Initial Investigation and Redesign of the Algorithms,” Contractor Report NASA/CR-2000-209863, 2000. Accessed: Apr. 6 2023. [Online]. Available: https://ntrs.nasa.gov/api/citations/20000041705/downloads/20000041705.pdf

CONTACTS

Florian Büchner

email: florian.buechner@tu-ilmenau.de
ORCID: https://orcid.org/0000-0002-5839-1193