

Dual-Mode Model Predictive Motion Cueing

Tim Nicolai

`tim.nicolai@itwm.fraunhofer.de`

Fraunhofer Institute for Industrial Mathematics ITWM

Sebastian Emmerich

Fraunhofer Institute for Industrial Mathematics ITWM

Michael Burger

Fraunhofer Institute for Industrial Mathematics ITWM

Matthias Gerdtz

University of the Bundeswehr Munich

Research Article

Keywords: Driving simulation, Motion cueing, Model predictive control, Stability analysis, Invariant sets, Quadratic programming

Posted Date: March 12th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-8743301/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Dual-Mode Model Predictive Motion Cueing

Tim Nicolai^{1,2*}, Sebastian Emmerich¹, Michael Burger¹,
Matthias Gerdts²

¹Mathematics for Vehicles and Systems, Fraunhofer Institute for
Industrial Mathematics ITWM, Fraunhofer-Platz 1, Kaiserslautern,
67663, Germany.

²Department of Aerospace Engineering, University of the Bundeswehr
Munich, Werner-Heisenberg-Weg 39, Neubiberg, 85579, Germany.

*Corresponding author(s). E-mail(s): tim.nicolai@itwm.fraunhofer.de;
Contributing authors: sebastian.emmerich@itwm.fraunhofer.de;
michael.burger@itwm.fraunhofer.de; matthias.gerdts@unibw.de;

Abstract

Dynamic driving simulators rely on motion cueing algorithms to generate platform trajectories that reproduce accelerations and rotations of simulated vehicles within physical limits. The employment of model predictive control (MPC) has emerged as a promising approach for this task, primarily due to its capability to address constraints explicitly. However, ensuring closed-loop stability poses a fundamental challenge. Conventional stability-enforcing mechanisms, such as terminal equality constraints or terminal costs, limit workspace utilization by inducing premature washout behavior, while extending the prediction horizon conflicts with stringent real-time requirements. This paper presents a dual-mode architecture that decouples stability certification from performance optimization. A safety mode based on linear state feedback defines a maximal positive invariant set that guarantees asymptotic stability and constraint satisfaction. Within this set, a performance mode employs MPC for reference tracking without terminal constraints, maximizing workspace utilization. The approach is formulated as a quadratic program and validated on a robot-based driving simulator. Simulation results confirm guaranteed constraint satisfaction and safe mode switching capability. The architecture reconciles stability certification with performance optimization for MPC-based motion cueing.

Keywords: Driving simulation, Motion cueing, Model predictive control, Stability analysis, Invariant sets, Quadratic programming

1 Introduction

Dynamic driving simulators have become indispensable tools for vehicle development, driver training, and human factors research [1, 2]. These systems integrate real-time vehicle dynamics simulation with physical motion platforms, providing drivers with vestibular, somatosensory, and proprioceptive feedback [3, 4]. The validity of simulator-based studies depends critically on the fidelity with which vehicle motions are reproduced and the resulting level of driver immersion [5, 6]. The central component enabling this motion reproduction is the motion cueing algorithm (MCA), which translates the dynamics of the simulated vehicle into motion commands for the physical platform.

Various motion platform configurations have been developed for driving simulation. The two most common architectures are hexapod-based Stewart platforms, which provide six degrees of freedom through parallel kinematic actuation, and serial kinematic robots [2]. Regardless of the specific platform architecture, simulator workspaces remain fundamentally smaller than required to fully reproduce real vehicle motion [7]. This workspace limitation constitutes the central challenge addressed by the MCA.

1.1 Classical Washout Algorithm

The predominant MCA approach in industrial practice remains the classical washout algorithm (CWA), introduced by Schmidt and Conrad [8] and subsequently refined in numerous studies [9–11]. The CWA employs linear high-pass and low-pass filters to decompose vehicle motion into transient and sustained components. High-frequency transient accelerations are reproduced through direct translational platform displacement, while sustained low-frequency accelerations are rendered via tilt coordination. This technique exploits the gravito-inertial ambiguity of the human vestibular system, wherein the otolith organs cannot distinguish between gravitational and inertial accelerations for small tilt angles and rates [12, 13]. A defining characteristic of the CWA is the washout mechanism, which ensures that the motion platform returns toward its neutral configuration after reproducing transient motions [9].

Despite its widespread adoption, the CWA exhibits several inherent limitations. First, the algorithm lacks state feedback, meaning that actual platform position relative to workspace constraints is not incorporated into the motion generation [6]. Consequently, filter parameters must be tuned conservatively for worst-case maneuvers, resulting in reduced motion magnitude during regular driving [14]. Second, linear filtering introduces phase shifts between vehicle and simulator motion, potentially creating perceptual artifacts [15]. Third, the tuning process involves numerous interdependent parameters, requiring extensive subjective testing and expert knowledge [16].

1.2 Model Predictive Control

The limitations of filter-based algorithms prompted the investigation of optimization-based model predictive control (MPC) algorithms, which emerged as a promising approach [6, 17]. MPC formulates motion cueing as a finite-horizon optimal control problem, computing control inputs by minimizing a cost function that penalizes

tracking errors between vehicle and simulator motion while respecting platform constraints [18]. Using a dynamic model to predict future platform states, MPC can anticipate upcoming maneuvers and workspace limitations, enabling proactive motion shaping [19].

A key advantage of MPC over filter-based methods is the explicit incorporation of platform constraints within the optimization problem [14]. Rather than requiring conservative worst-case tuning, MPC can utilize the full workspace for each individual maneuver while guaranteeing that hardware limits are not exceeded. Studies have demonstrated that MPC-based motion cueing generally achieves higher perceptual quality than classical filter-based approaches [20, 21].

However, the application of MPC introduces practical challenges. The real-time requirement imposes stringent computational constraints, as the optimization problem must be solved within the sampling interval, typically in the order of milliseconds [2]. Moreover, ensuring closed-loop stability poses a fundamental theoretical challenge [22, 23]. Conventional stability-enforcing mechanisms in MPC, such as terminal equality constraints or terminal costs, limit workspace utilization by biasing the optimization toward returning to the neutral position [24, 25].

1.3 Contributions

This paper presents a dual-mode MPC architecture for motion cueing that systematically decouples stability certification from performance optimization. The proposed framework addresses the fundamental tension between guaranteed stability and maximum workspace utilization by introducing two operational modes with distinct objectives. A safety mode employing stabilizing linear state feedback defines a maximal positive invariant set that guarantees asymptotic stability and constraint satisfaction. Within this set, a performance mode employs MPC for reference tracking without terminal constraints, thereby maximizing workspace utilization.

The dual-mode architecture is formulated independently of the motion platform and relies solely on a linear, time-invariant, state-space representation of the platform dynamics. This general structure allows the approach to be applied to various configurations of dynamic motion simulators, including hexapod-based Stewart platforms and serial kinematic robots. This work demonstrates and validates the methodology using a robot-based driving simulator.

The main contributions are summarized as follows:

1. A dual-mode control architecture that provides formal guarantees for recursive feasibility and closed-loop stability without requiring terminal costs, terminal equality constraints, or extended prediction horizons in the MPC formulation.
2. A systematic procedure for constructing the maximal positive invariant set under linear state feedback, establishing a certified safe operating region for motion cueing applications.
3. A quadratic program formulation enabling real-time computation with computational complexity scaling with the control horizon.
4. Application to a robot-based driving simulator demonstrating the integration of tilt coordination through an extended platform model with dedicated virtual tilt channels.

The remainder of this paper is organized as follows. Section 2 presents the mathematical preliminaries, including discrete-time linear time-invariant systems, MPC fundamentals, and stability concepts. Section 3 develops the dual-mode architecture comprising safety mode and performance mode, establishing recursive feasibility and stability guarantees. Section 4 derives the quadratic program formulation for real-time implementation. Section 5 details the application to a robot-based driving simulator, including tilt coordination, constraint specification, and simulation results. Section 6 concludes with a summary and directions for future research.

2 Preliminaries

This section delineates the mathematical underpinnings of the proposed dual-mode model predictive motion cueing framework. The focus is on discrete-time linear time-invariant (LTI) systems, which form the basis of the proposed architecture. The principles of MPC are introduced, with an emphasis on stability concepts related to this control strategy. The stability guarantees in linear MPC and the restrictions these guarantees present for motion cueing applications are examined.

2.1 Discrete-Time LTI State-Space Models

Throughout this paper, we consider the state-space representation of deterministic discrete-time LTI systems

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k)\end{aligned}\tag{1}$$

with the system matrices $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times m}$, $\mathbf{C} \in \mathbb{R}^{p \times n}$, $\mathbf{D} \in \mathbb{R}^{p \times m}$, the state vector $\mathbf{x}(k) \in \mathbb{R}^n$, the input vector $\mathbf{u}(k) \in \mathbb{R}^m$, the output vector $\mathbf{y}(k) \in \mathbb{R}^p$, and the time index $k \in \mathbb{N}_0$.

Remark 1 (Feedthrough Elimination) Any direct feedthrough matrix $\mathbf{D}$ in (1) can be eliminated by augmenting the state and employing the input increment $\Delta\mathbf{u}$ according to

$$\tilde{\mathbf{x}}(k) := \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{u}(k) \end{bmatrix} \in \mathbb{R}^{n+m}, \quad \Delta\mathbf{u}(k) := \mathbf{u}(k+1) - \mathbf{u}(k) \in \mathbb{R}^m.$$

The augmented discrete-time state space model is then given by

$$\begin{aligned}\tilde{\mathbf{x}}(k+1) &= \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{O}_{m \times n} & \mathbf{I}_m \end{bmatrix}}_{\tilde{\mathbf{A}}} \tilde{\mathbf{x}}(k) + \underbrace{\begin{bmatrix} \mathbf{O}_{n \times m} \\ \mathbf{I}_m \end{bmatrix}}_{\tilde{\mathbf{B}}} \Delta\mathbf{u}(k), \\ \mathbf{y}(k) &= \underbrace{[\mathbf{C} \ \mathbf{D}]}_{\tilde{\mathbf{C}}} \tilde{\mathbf{x}}(k),\end{aligned}$$

with $\tilde{\mathbf{D}} = \mathbf{O}$.

To model basic simulator dynamics for motion cueing applications, the state vector typically contains platform velocities and positions and the input vector comprises commanded accelerations.

2.2 Model Predictive Control

MPC is an optimization-based control strategy that employs a dynamic model to determine control actions in a predictive manner, see [22, 23] for a comprehensive overview. Using the discrete-time model (1) to predict future behavior, the control task can be formulated as a finite-horizon optimal control problem. Given a prediction horizon $N \in \mathbb{N}$ and the current state $\mathbf{x}_k \in \mathbb{R}^n$, the objective is to compute a control sequence $\{\mathbf{u}(i)\}_{i=k}^{k+N-1}$ that minimizes a specified cost function while respecting system constraints.

2.2.1 Cost Function

The cost function encodes the control objectives, thereby influencing the controlled system dynamics. In the context of motion cueing, the goal is to track a given reference trajectory $\{\mathbf{r}(i)\}_{i=k}^{k+N-1}$ with $\mathbf{r}(i) \in \mathbb{R}^p$ representing the vehicle motion to be replicated by the simulator platform. A quadratic cost function penalizing tracking deviation and control effort is given by

$$J_N = \frac{1}{2} \sum_{i=k}^{k+N-1} ((\mathbf{r}(i) - \mathbf{y}(i))^\top \mathbf{Q} (\mathbf{r}(i) - \mathbf{y}(i)) + \mathbf{u}(i)^\top \mathbf{R} \mathbf{u}(i)), \quad (2)$$

where $\mathbf{Q} \in \mathbb{R}^{p \times p}$ and $\mathbf{R} \in \mathbb{R}^{m \times m}$ are symmetric positive semidefinite weighting matrices. The relative weighting of conflicting objectives is determined by these weighting matrices, in which larger entries in $\mathbf{Q}$ prioritize accurate replication of the vehicular motion, while larger entries in matrix $\mathbf{R}$ promote smoother actuator behavior.

2.2.2 Constraint Handling

Since dynamic motion platforms are inherently limited by actuator capabilities and physical workspace boundaries, both the control input $\mathbf{u}$ and the state $\mathbf{x}$ are restricted to admissible sets $\mathcal{U} \subset \mathbb{R}^m$ and $\mathcal{X} \subset \mathbb{R}^n$, respectively. Commonly, these sets are given as convex polyhedral constraint sets,

$$\mathcal{U} = \{\mathbf{u} \in \mathbb{R}^m \mid \mathbf{F}_\mathbf{u} \mathbf{u} \leq \mathbf{g}_\mathbf{u}\}, \quad \mathcal{X} = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_\mathbf{x} \mathbf{x} \leq \mathbf{g}_\mathbf{x}\}, \quad (3)$$

where $\mathbf{F}_\mathbf{u} \in \mathbb{R}^{n_\mathbf{u} \times m}$, $\mathbf{g}_\mathbf{u} \in \mathbb{R}^{n_\mathbf{u}}$, $\mathbf{F}_\mathbf{x} \in \mathbb{R}^{n_\mathbf{x} \times n}$, and $\mathbf{g}_\mathbf{x} \in \mathbb{R}^{n_\mathbf{x}}$. It is assumed that both sets contain the origin, ensuring that the equilibrium pair $(\mathbf{x}, \mathbf{u}) = (\mathbf{0}, \mathbf{0})$ is admissible.

2.2.3 Receding Horizon Principle

Combining the cost function (2) with the admissible sets (3), the MPC optimization problem at time step k can be written as

$$\begin{aligned}
& \underset{\{\mathbf{u}(i)\}_{i=k}^{k+N-1}}{\text{minimize}} && \frac{1}{2} \sum_{i=k}^{k+N-1} ((\mathbf{r}(i) - \mathbf{y}(i))^\top \mathbf{Q} (\mathbf{r}(i) - \mathbf{y}(i)) + \mathbf{u}(i)^\top \mathbf{R} \mathbf{u}(i)) \\
& \text{subject to} && \mathbf{x}(k) = \mathbf{x}_k, \\
& && \mathbf{x}(i+1) = \mathbf{A}\mathbf{x}(i) + \mathbf{B}\mathbf{u}(i), \\
& && \mathbf{y}(i) = \mathbf{C}\mathbf{x}(i) + \mathbf{D}\mathbf{u}(i), \\
& && \mathbf{u}(i) \in \mathcal{U}, \\
& && \mathbf{x}(i+1) \in \mathcal{X}, \\
& && i = k, k+1, \dots, k+N-1.
\end{aligned} \tag{4}$$

Solving (4) given the current state $\mathbf{x}_k$ and a reference trajectory $\{\mathbf{r}(i)\}_{i=k}^{k+N-1}$ yields an optimal input sequence $\{\mathbf{u}^*(i)\}_{i=k}^{k+N-1}$. After applying the first element $\mathbf{u}^*(k)$, the remaining input sequence is discarded and the prediction horizon is shifted by one step, i.e. $k \leftarrow k+1$. The system state $\mathbf{x}_{k+1}$ is then measured or estimated again, and a new optimization problem is formulated based on the updated information. This procedure, referred to as the *receding horizon principle*, implicitly defines the feedback law

$$\mu_N(k, \mathbf{x}_k) := \mathbf{u}^*(k) \tag{5}$$

yielding the closed-loop system dynamics

$$\begin{aligned}
\mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mu_N(k, \mathbf{x}(k)), \\
\mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mu_N(k, \mathbf{x}(k)), \quad k = 0, 1, 2, \dots,
\end{aligned} \tag{6}$$

with the initial condition $\mathbf{x}(0) = \mathbf{x}_0$.

Algorithm 1 Basic MPC Implementation

Require: Prediction horizon $N \in \mathbb{N}$. Set $k := 0$.

- 1: Measure the state $\mathbf{x}_k$ at time k .
 - 2: Solve (4) for optimal input sequence $\{\mathbf{u}^*(i)\}_{i=k}^{k+N-1}$. Set $\mu_N(k, \mathbf{x}_k) = \mathbf{u}^*(k)$.
 - 3: Apply $\mu_N(k, \mathbf{x}_k)$ to the system for the time step from k to $k+1$.
 - 4: Set $k \leftarrow k+1$ and go to step 1.
-

2.3 Stability Concepts

Stability is a fundamental property of dynamic systems that ensures predictable and safe behavior over time. In motion cueing applications, stability ensures that the motion platform returns to its neutral position, $\mathbf{x} = \mathbf{0}$, when no reference trajectory,

or the zero reference $\mathbf{r}(i) \equiv \mathbf{0}$, is to be tracked, thus preventing drift and avoiding unsafe excursions.

To analyze the dynamics in the absence of external inputs, the discrete-time state-space model (1) with zero input, i.e., $\mathbf{u}(k) \equiv \mathbf{0}$ is considered. This leads to the autonomous state equation

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k). \quad (7)$$

The state $\mathbf{x}_e = \mathbf{0}$ constitutes an *equilibrium state*, since (7), upon reaching this state, remains there for all subsequent time steps.

Definition 1 (Asymptotic Stability) The equilibrium state $\mathbf{x}_e = \mathbf{0}$ of the autonomous system (7) is referred to as *asymptotically stable* if there exists $\gamma > 0$ such that $\|\mathbf{x}(k)\| \leq \gamma\|\mathbf{x}_0\|$ for all $k \geq 0$ and $\lim_{k \rightarrow \infty} \|\mathbf{x}(k)\| = 0$.

A fundamental theoretical result relates asymptotic stability to the eigenvalue spectrum of the system matrix. Let $\rho(\mathbf{A})$ denote the spectral radius of $\mathbf{A}$, defined as the largest absolute value of its eigenvalues.

Theorem 1 (Schur Stability) Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be the state matrix of the autonomous system (7) with eigenvalues $\lambda_1, \dots, \lambda_n \in \mathbb{C}$. The equilibrium state $\mathbf{x}_e = \mathbf{0}$ is asymptotically stable if and only if all eigenvalues lie strictly inside the unit disk, that is, $\rho(\mathbf{A}) < 1$. $\square$

An alternative characterization of asymptotic stability is given by the Lyapunov theory [26].

Theorem 2 (Lyapunov's Direct Method for Discrete-time Systems [26, 27]) Let an autonomous discrete-time system be given by $\mathbf{x}(k+1) = f(\mathbf{x}(k))$, where $f: D \rightarrow \mathbb{R}^n$ is assumed to be locally Lipschitz continuous on an open set $D \subseteq \mathbb{R}^n$ containing the origin, i.e., $\mathbf{0} \in D$. Suppose there exists a function $V: D \rightarrow \mathbb{R}$ satisfying:

1. V is continuous,
2. $V(\mathbf{0}) = 0$,
3. $V(\mathbf{x}) > 0$ for all $\mathbf{x} \in D \setminus \{\mathbf{0}\}$.

Then, $\mathbf{x}_e = \mathbf{0}$ is asymptotically stable, if $V(\mathbf{x}) - V(f(\mathbf{x})) > 0$ holds for all $\mathbf{x} \in D \setminus \{\mathbf{0}\}$. In this context, V is called a Lyapunov function for the system. $\square$

Remark 2 (Control Lyapunov Function) For a non-autonomous discrete-time system $\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k))$, a function V satisfying the conditions of Theorem 2 is called a *Control-Lyapunov function (CLF)*. [28]

It is important to note that Theorem 1 and Theorem 2 provide equivalent characterizations of asymptotic stability for the autonomous system (7). Both concepts extend to non-autonomous systems, where the objective becomes the synthesis of a stabilizing feedback law for the closed-loop system (6). In particular, *Schur stability* (Theorem 1) underpins the design of a linear state feedback $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$ by

assigning the eigenvalues of $\mathbf{A} - \mathbf{BK}$ to lie inside the unit disk. In contrast, *Lyapunov theory* provides the theoretical foundation for establishing closed-loop stability in MPC through the construction of an appropriate CLF.

2.3.1 Stability of Linear MPC

Since the MPC feedback law (5) is implicitly defined by Algorithm 1, stability must be ensured through the structure of the optimization problem itself. Considering the *washout* case with zero reference, $\mathbf{r}(i) \equiv \mathbf{0}$, the MPC optimization problem (4), including feedthrough elimination (Remark 1) when $\mathbf{D} \neq \mathbf{0}$, may be written as

$$\begin{aligned} & \underset{\{\mathbf{u}(i)\}_{i=k}^{k+N-1}}{\text{minimize}} && \frac{1}{2} \sum_{i=k}^{k+N-1} \left(\mathbf{x}(i)^\top \tilde{\mathbf{Q}} \mathbf{x}(i) + \mathbf{u}(i)^\top \tilde{\mathbf{R}} \mathbf{u}(i) \right) \\ & \text{subject to} && \mathbf{x}(k) = \mathbf{x}_k, \\ & && \mathbf{x}(i+1) = \mathbf{A} \mathbf{x}(i) + \mathbf{B} \mathbf{u}(i), \\ & && \mathbf{u}(i) \in \mathcal{U}, \\ & && \mathbf{x}(i+1) \in \mathcal{X}, \\ & && i = k, k+1, \dots, k+N-1 \end{aligned} \tag{8}$$

The central idea linking linear MPC to Lyapunov theory is to employ the MPC value function as a candidate CLF. Define the cost functional $J_N: \mathbb{R}^n \times (\mathbb{R}^m)^N \rightarrow \mathbb{R}$ and the optimal value function $V_N: \mathcal{X}_N \rightarrow \mathbb{R}$ by

$$\begin{aligned} J_N(\mathbf{x}_k, \{\mathbf{u}(i)\}_{i=k}^{k+N-1}) &:= \frac{1}{2} \sum_{i=k}^{k+N-1} \left(\mathbf{x}(i)^\top \tilde{\mathbf{Q}} \mathbf{x}(i) + \mathbf{u}(i)^\top \tilde{\mathbf{R}} \mathbf{u}(i) \right), \\ V_N(\mathbf{x}_k) &:= \inf_{\{\mathbf{u}(i)\}_{i=k}^{k+N-1} \in \mathcal{U}_N(\mathbf{x}_k)} J_N, \end{aligned}$$

where $\mathcal{U}_N(\mathbf{x}_k)$ denotes the set of admissible input sequences for state $\mathbf{x}_k$, and $\mathcal{X}_N := \{\mathbf{x} \in \mathcal{X} \mid \mathcal{U}_N(\mathbf{x}) \neq \emptyset\}$ is the feasible set. By construction, V_N satisfies the properties of continuity and positive definiteness if $\tilde{\mathbf{Q}}$ and $\tilde{\mathbf{R}}$ are positive definite [29]. However, the Lyapunov decrease condition $V_N(\mathbf{x}_{k+1}) < V_N(\mathbf{x}_k)$ along closed-loop trajectories is not guaranteed by the basic formulation (8).

Three principal approaches ensure that V_N serves as a CLF for the closed-loop system [22, 23]:

1. *Sufficiently long prediction horizon:* $V_N(\mathbf{x}_{k+1}) < V_N(\mathbf{x}_k)$ can be ensured if N exceeds a critical threshold. While theoretical bounds on the required horizon length have been established [23, 30], verifying these conditions in practice may be challenging. However, letting $N \rightarrow \infty$ guarantees the Lyapunov decrease via Bellman's principle of optimality [31, 32].
2. *Terminal equality constraint:* Enforcing the equality constraint $\mathbf{x}(k+N) = \mathbf{0}$ ensures that the shifted input sequence, appended with $\mathbf{u} = \mathbf{0}$, remains feasible, thereby guaranteeing $V_N(\mathbf{x}_{k+1}) < V_N(\mathbf{x}_k)$.

3. *Terminal cost and terminal set:* Introducing a terminal penalty $V_f(\mathbf{x}(k+N)) = \mathbf{x}(k+N)^\top \mathbf{P}_f \mathbf{x}(k+N)$ and constraining the terminal state to a control-invariant set, $\mathbf{x}(k+N) \in \mathcal{X}_f \subseteq \mathcal{X}$, ensures that $V_N(\mathbf{x}_{k+1}) < V_N(\mathbf{x}_k)$. This means, $\mathcal{X}_f$ must be positive invariant under a stabilizing feedback, such as $\mu_f(i, \mathbf{x}(i)) = -\mathbf{K}_f \mathbf{x}(i)$, $i \geq k+N$. A common choice is to set $\mathbf{K}_f$ as the unconstrained LQR gain and $\mathbf{P}_f$ as the solution of the discrete-time algebraic Riccati equation. [22]

Each approach ensures that V_N serves as a CLF for the closed-loop system, thereby certifying asymptotic stability of the origin. In particular, terminal costs and terminal constraints have been employed in MPC-based motion cueing [24, 25, 33, 34]. However, these stability-enforcing mechanisms impose limitations for motion cueing applications. Long prediction horizons increase the computational burden of solving equation (4) and may conflict with real-time requirements. Terminal equality constraints, such as $\mathbf{x}(k+N) = \mathbf{0}$, require the platform to reach the neutral position within the prediction horizon, thereby severely restricting workspace utilization. Terminal costs and terminal sets, while less restrictive, bias the optimization toward returning to the neutral position, effectively inducing potentially premature washout behavior and degrading motion fidelity. These observations motivate the decoupling of stability certification and performance optimization. By delegating stability to a dedicated feedback controller, the MPC optimization (4) can be tuned to prioritize performance while allowing maximum workspace utilization.

3 Dual-Mode Predictive Motion Cueing

This section presents a dual-mode control architecture for motion cueing that systematically separates safety guarantees from performance optimization. The approach comprises two operational modes: a safety mode employing stabilizing state feedback and a performance mode utilizing MPC for reference tracking. Throughout this section, it is assumed that a discrete-time LTI state-space representation (1) modeling the motion platform dynamics is available, the pair $(\mathbf{A}, \mathbf{B})$ is reachable, and the full state vector $\mathbf{x}$ is accessible at every time step.

3.1 Safety Mode: Stabilizing State Feedback

The goal of the safety mode is to provide a stabilizing controller that ensures asymptotic convergence to the origin while satisfying all constraints. Since $(\mathbf{A}, \mathbf{B})$ is reachable, a constant feedback matrix $\mathbf{K} \in \mathbb{R}^{m \times n}$ can be designed such that the closed-loop matrix

$$\mathbf{A}_K := \mathbf{A} - \mathbf{B}\mathbf{K}$$

satisfies $\rho(\mathbf{A}_K) < 1$. Under the feedback law $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$, the closed-loop dynamics become

$$\mathbf{x}(k+1) = \mathbf{A}_K \mathbf{x}(k), \tag{9}$$

for which the origin $\mathbf{x} = \mathbf{0}$ is globally asymptotically stable by Theorem 1.

3.1.1 State Feedback Design

The design of $\mathbf{K}$ can be accomplished via *pole placement*, by specifying a set of n desired eigenvalues $\lambda_1, \dots, \lambda_n$ satisfying $|\lambda_i| < 1$. For single-input systems, i.e., $m = 1$, Ackermann's formula [35] yields an explicit construction. Given the monic polynomial

$$p(\lambda) = \prod_{i=1}^n (\lambda - \lambda_i) = \lambda^n + \alpha_{n-1} \lambda^{n-1} + \dots + \alpha_0$$

and the reachability matrix

$$\mathcal{R}_n = [\mathbf{B} \ \mathbf{AB} \ \mathbf{A}^2\mathbf{B} \ \dots \ \mathbf{A}^{n-1}\mathbf{B}] \in \mathbb{R}^{n \times n},$$

the feedback gain placing the eigenvalues of $\mathbf{A}_{\mathbf{K}}$ at the roots of $p(\lambda)$ is given by

$$\mathbf{K} = \mathbf{e}_n^\top \mathcal{R}_n^{-1} p(\mathbf{A}), \quad (10)$$

where $\mathbf{e}_n^\top = [0 \ \dots \ 0 \ 1] \in \mathbb{R}^n$ and $p(\mathbf{A}) = \mathbf{A}^n + \alpha_{n-1} \mathbf{A}^{n-1} + \dots + \alpha_0 \mathbf{I}_n$.

Remark 3 (Eigenvalue Placement Guidelines) To ensure smooth transient behavior and robustness, the closed-loop eigenvalues should be placed with a conservative margin to the unit circle. Negative eigenvalues should be avoided as they induce step-to-step sign alternations, thereby increasing actuator reversals and peak effort.

Remark 4 (State Feedback for Multiple Input Systems) For $m \geq 2$, Ackermann's formula does not directly generalize. In this case, robust assignment methods such as the algorithm by Kautsky, Nichols, and Van Dooren [36] may be employed.

Example 1 (State Feedback for Discrete-Time Double Integrator) Consider a discrete-time double integrator

$$\mathbf{x}(k+1) = \underbrace{\begin{bmatrix} 1 & 0 \\ T_s & 1 \end{bmatrix}}_{\mathbf{A}} \mathbf{x}(k) + \underbrace{\begin{bmatrix} T_s \\ \frac{1}{2} T_s^2 \end{bmatrix}}_{\mathbf{B}} u(k)$$

with the sampling interval fixed at $T_s = 0.012$ s. For a non-oscillatory, critically damped response, the closed-loop poles are placed at $\lambda_1 = \lambda_2 = 0.98$, yielding

$$p(\lambda) = (\lambda - 0.98)^2 = \lambda^2 - 1.96\lambda + 0.9604.$$

The reachability matrix

$$\mathcal{R}_2 = [\mathbf{B} \ \mathbf{AB}] = \begin{bmatrix} 0.012 & 0.012 \\ 0.000072 & 0.000216 \end{bmatrix} \quad (11)$$

satisfies $\text{rank}(\mathcal{R}_2) = 2$, confirming reachability. Applying (10) yields

$$\mathbf{K} = [3.3167 \ 2.7778], \quad \mathbf{A}_{\mathbf{K}} = \mathbf{A} - \mathbf{BK} = \begin{bmatrix} 0.9602 & -0.0333 \\ 0.0118 & 0.9998 \end{bmatrix}. \quad (12)$$

The closed-loop response for the initial condition $\mathbf{x}(0) = [0 \ 0.3]^\top$ illustrating asymptotic stability is shown in Fig. 1.

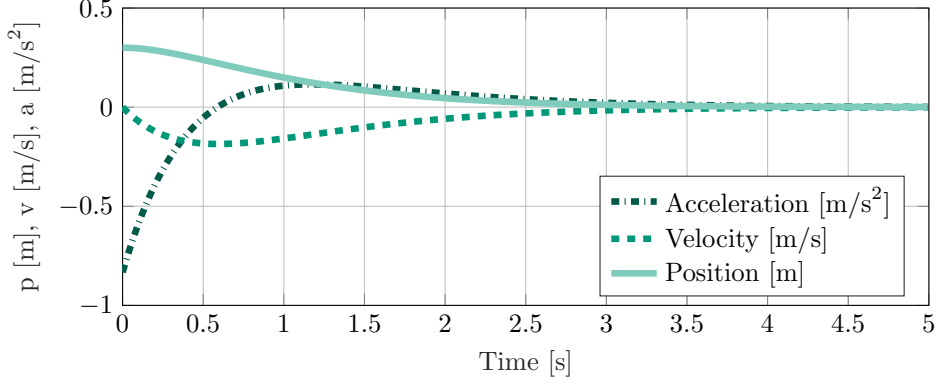


Fig. 1 Closed-loop state trajectory and control input for $v(0) = 0$ m/s, $p(0) = 0.3$ m

3.1.2 Constructing the Maximal Positive Invariant Set

Beyond stability considerations, the feedback law $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$ also induces a certified safe operating region in the form of a positive invariant set. For the construction of such sets, input admissibility is mapped into the state space. To simplify notation, the resulting combined admissible set is defined as

$$\mathcal{X}_{\mathbf{K}} := \left\{ \mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_{\mathbf{x}} \mathbf{x} \leq \mathbf{g}_{\mathbf{x}}, \mathbf{F}_{\mathbf{u}}(-\mathbf{K}) \mathbf{x} \leq \mathbf{g}_{\mathbf{u}} \right\} = \left\{ \mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_{\mathbf{K}} \mathbf{x} \leq \mathbf{g}_{\mathbf{K}} \right\}, \quad (13)$$

with

$$\mathbf{F}_{\mathbf{K}} := \begin{bmatrix} \mathbf{F}_{\mathbf{x}} \\ \mathbf{F}_{\mathbf{u}}(-\mathbf{K}) \end{bmatrix}, \quad \mathbf{g}_{\mathbf{K}} := \begin{bmatrix} \mathbf{g}_{\mathbf{x}} \\ \mathbf{g}_{\mathbf{u}} \end{bmatrix}.$$

The construction of the safe operating region within $\mathcal{X}_{\mathbf{K}}$ relies on the following definitions.

Definition 2 (Positive Invariant Set) A set $\mathcal{O}_{\mathbf{K}} \subseteq \mathbb{R}^n$ is positive invariant for (9) if

$$\mathbf{x}(0) \in \mathcal{O}_{\mathbf{K}} \implies \mathbf{x}(k) \in \mathcal{O}_{\mathbf{K}} \text{ for all } k \in \mathbb{N}.$$

It should be noted that the stability condition $\rho(\mathbf{A}_{\mathbf{K}}) < 1$, together with the fact that $\mathcal{X}_{\mathbf{K}}$ is a polyhedral set containing the origin, guarantees the existence of a polyhedral positive invariant set for (9) [37]. Of particular interest is the largest such set.

Definition 3 (Maximal Positive Invariant Set) The set $\mathcal{O}_{\mathbf{K}}^{\infty} \subseteq \mathcal{X}_{\mathbf{K}}$ is the maximal positive invariant (MPI) set for (9) if $\mathcal{O}_{\mathbf{K}}^{\infty}$ is positive invariant and contains all other positive invariant sets within $\mathcal{X}_{\mathbf{K}}$.

A key characteristic of $\mathcal{O}_{\mathbf{K}}^{\infty}$ is providing a formal safety certificate: any state $\mathbf{x}_0 \in \mathcal{O}_{\mathbf{K}}^{\infty}$ ensures that the closed-loop trajectory under $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$ satisfies $\mathbf{x}(k) \in \mathcal{O}_{\mathbf{K}}^{\infty}$

for all $k \in \mathbb{N}$. Following [38], the polyhedral MPI set may be constructed iteratively using the *precursor set*

$$\text{Pre}(\mathcal{X}_{\mathbf{K}}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_{\mathbf{K}} \mathbf{A}_{\mathbf{K}} \mathbf{x} \leq \mathbf{g}_{\mathbf{K}}\} \cap \mathcal{X}_{\mathbf{K}}.$$

Accordingly, the MPI set is the largest fixed point of the precursor operator. Starting from $\mathcal{X}_{\mathbf{K}}$, repeated application according to

$$\mathcal{O}_{\mathbf{K}}^0 = \mathcal{X}_{\mathbf{K}}, \quad \mathcal{O}_{\mathbf{K}}^{i+1} = \text{Pre}(\mathcal{O}_{\mathbf{K}}^i), \quad i = 0, 1, 2, \dots$$

contracts the set until a fixed point is reached. The sequence $\{\mathcal{O}_{\mathbf{K}}^i\}$ is nested and nonincreasing. If $\mathcal{O}_{\mathbf{K}}^{i+1} = \mathcal{O}_{\mathbf{K}}^i$ for some i , then $\mathcal{O}_{\mathbf{K}}^\infty = \mathcal{O}_{\mathbf{K}}^i$. As $\rho(\mathbf{A}_{\mathbf{K}}) < 1$ and $\mathcal{X}_{\mathbf{K}}$ is a polyhedron containing the origin, this procedure terminates in finitely many steps yielding the polyhedral MPI set $\mathcal{O}_{\mathbf{K}}^\infty = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_{\mathbf{K}}^\infty \mathbf{x} \leq \mathbf{g}_{\mathbf{K}}^\infty\}$ [38]. Algorithm 2 implements this iteration.

Algorithm 2 Computation of the Polyhedral Maximal Positive Invariant Set $\mathcal{O}_\infty$

Require: Closed-loop matrix $\mathbf{A}_{\mathbf{K}}$, constraint set $\mathcal{X}_{\mathbf{K}} = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_{\mathbf{K}} \mathbf{x} \leq \mathbf{g}_{\mathbf{K}}\}$

Ensure: Polyhedral MPI set $\mathcal{O}_{\mathbf{K}}^\infty$

- 1: $\mathcal{O}_{\mathbf{K}}^0 \leftarrow \mathcal{X}_{\mathbf{K}}, i \leftarrow 0$
 - 2: **repeat**
 - 3: $\mathcal{O}_{\mathbf{K}}^{i+1} \leftarrow \text{Pre}(\mathcal{O}_{\mathbf{K}}^i)$
 - 4: $i \leftarrow i + 1$
 - 5: **until** $\mathcal{O}_{\mathbf{K}}^{i+1} = \mathcal{O}_{\mathbf{K}}^i$ up to a prescribed tolerance
 - 6: $\mathcal{O}_{\mathbf{K}}^\infty \leftarrow \mathcal{O}_{\mathbf{K}}^i$
-

Example 2 (MPI Set Computation for Single Axis) We consider the discrete-time double integrator with state feedback from Example 1. The constraint sets are defined according to the physical limits of longitudinal motion in the x-direction of the RODOS[®] simulator platform:

$$a \in [-6, 6] \text{ m/s}^2, \quad v \in [-1.5, 1.5] \text{ m/s}, \quad p \in [-0.55, 0.3] \text{ m}. \quad (14)$$

These limits correspond to polyhedral constraint sets

$$\mathcal{U} = \{u \in \mathbb{R} \mid \mathbf{F}_{\mathbf{u}} u \leq \mathbf{g}_{\mathbf{u}}\}, \quad \mathcal{X} = \{\mathbf{x} \in \mathbb{R}^2 \mid \mathbf{F}_{\mathbf{x}} \mathbf{x} \leq \mathbf{g}_{\mathbf{x}}\}, \quad (15)$$

with

$$\mathbf{F}_{\mathbf{x}} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{g}_{\mathbf{x}} = \begin{bmatrix} 1.5 \\ 0.55 \\ 1.5 \\ 0.3 \end{bmatrix}, \quad \mathbf{F}_{\mathbf{u}} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \quad \mathbf{g}_{\mathbf{u}} = \begin{bmatrix} 6 \\ 6 \end{bmatrix}. \quad (16)$$

With $\mathbf{K}$ from (12), the combined admissible set

$$\mathcal{X}_{\mathbf{K}} = \left\{ \mathbf{x} \in \mathbb{R}^2 \mid \begin{bmatrix} \mathbf{F}_{\mathbf{x}} \\ \mathbf{F}_{\mathbf{u}}(-\mathbf{K}) \end{bmatrix} \mathbf{x} \leq \begin{bmatrix} \mathbf{g}_{\mathbf{x}} \\ \mathbf{g}_{\mathbf{u}} \end{bmatrix} \right\}$$

is obtained. Algorithm 2 with initial set $\mathcal{O}_{\mathbf{K}}^0 = \mathcal{X}_{\mathbf{K}}$ converged after 53 iterations to the MPI set $\mathcal{O}_{\mathbf{K}}^\infty = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_{\mathbf{K}}^\infty \mathbf{x} \leq \mathbf{g}_{\mathbf{K}}^\infty\}$, where $\mathbf{F}_{\mathbf{K}}^\infty \in \mathbb{R}^{90 \times 2}$ and $\mathbf{g}_{\mathbf{K}}^\infty \in \mathbb{R}^{90}$. This set is illustrated

in Fig. 2. By construction, every $\mathbf{x}(0) \in \mathcal{O}_{\mathbf{K}}^\infty$ ensures that the closed-loop trajectory governed by (9) satisfies all acceleration, velocity, and position constraints for all $k \in \mathbb{N}$.

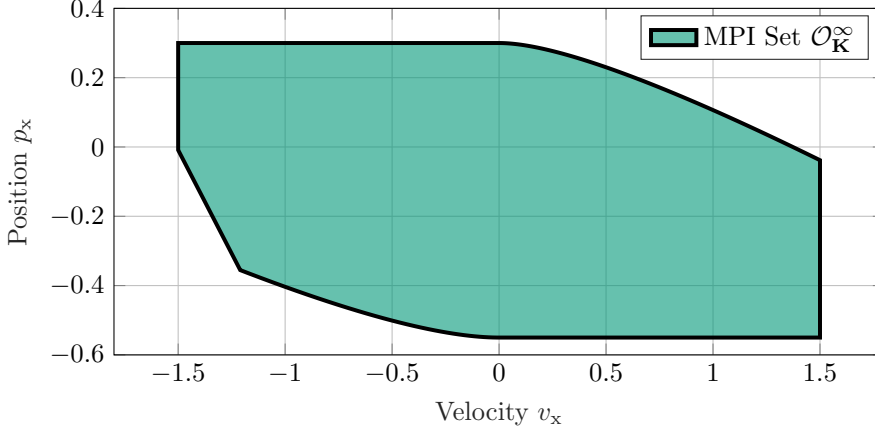


Fig. 2 MPI set for longitudinal movement of RODOS®

The safety mode provides asymptotic stability and formal constraint satisfaction guarantees through the feedback law $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$ and the MPI set $\mathcal{O}_{\mathbf{K}}^\infty$. However, this mode does not account for reference tracking or performance optimization. In motion cueing applications, the primary objective is to replicate vehicle motion through platform movement. To address this requirement, another mode is introduced that prioritizes workspace utilization while leveraging the safety guarantees established by $\mathcal{O}_{\mathbf{K}}^\infty$.

3.2 Performance Mode: Linear MPC for Reference Tracking

The goal of the performance mode is to maximize motion cueing fidelity without the need for stability-enforcing constraints or penalties. By delegating asymptotic stability to the safety mode, the MPC optimization can be designed exclusively for reference tracking and workspace utilization.

3.2.1 Optimization Problem

Given a previewed reference trajectory $\{\mathbf{r}(i)\}_{i=k}^{k+N_p-1}$ representing the vehicle motion to be reproduced by the motion platform, the performance-mode optimization problem

is formulated as

$$\begin{aligned} \underset{\{\mathbf{u}(i)\}_{i=k}^{k+N_c-1}}{\text{minimize}} \quad & \frac{1}{2} \left(\sum_{i=k}^{k+N_p-1} (\mathbf{r}(i) - \mathbf{y}(i))^\top \mathbf{Q} (\mathbf{r}(i) - \mathbf{y}(i)) + \sum_{i=k}^{k+N_c-1} \mathbf{u}(i)^\top \mathbf{R} \mathbf{u}(i) \right) \end{aligned} \quad (17a)$$

$$\begin{aligned} \text{subject to} \quad & \mathbf{x}(k) = \mathbf{x}_k, \\ & \mathbf{x}(i+1) = \mathbf{A}\mathbf{x}(i) + \mathbf{B}\mathbf{u}(i), \\ & \mathbf{y}(i) = \mathbf{C}\mathbf{x}(i) + \mathbf{D}\mathbf{u}(i), \\ & \mathbf{x}(i+1) \in \mathcal{O}_{\mathbf{K}}^\infty = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{F}_{\mathbf{K}}^\infty \mathbf{x} \leq \mathbf{g}_{\mathbf{K}}^\infty\}, \end{aligned} \quad (17b)$$

$$\begin{aligned} \mathbf{u}(i) \begin{cases} \in \mathcal{U} = \{\mathbf{u} \in \mathbb{R}^m \mid \mathbf{F}_{\mathbf{u}} \mathbf{u} \leq \mathbf{g}_{\mathbf{u}}\}, & i < k + N_c, \\ = -\mathbf{K} \mathbf{x}(i), & i \geq k + N_c, \end{cases} \end{aligned} \quad (17c)$$

$$i = k, k+1, \dots, k+N_p-1,$$

with the prediction horizon N_p and the control horizon $N_c \leq N_p$. This temporal separation exploits preview information for anticipatory motion shaping while maintaining computational tractability, since the number of decision variables scales with N_c rather than N_p . The constraint (17b) requires that all predicted states remain within the MPI set $\mathcal{O}_{\mathbf{K}}^\infty$, constructed under the state feedback $\mathbf{u} = -\mathbf{K}\mathbf{x}$. This allows instantaneous safe switching: at any time step k , the safety mode $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$ can be activated without risking infeasibility or constraint violation. The separation between free optimization and feedback-governed prediction is encoded in (17c). The control inputs are treated as decision variables constrained to the admissible set $\mathcal{U}$ for $i < k + N_c$, whereas for $i \geq k + N_c$, inputs are fixed according to the stabilizing feedback law. This feedback tail closure ensures that the predicted trajectory ultimately converges toward the origin, thereby maintaining membership in $\mathcal{O}_{\mathbf{K}}^\infty$. Solving (17) for $\{\mathbf{u}_d^*(i)\}_{i=k}^{k+N_c-1}$ and applying the first element $\mathbf{u}_d^*(k)$, implicitly defines the dual-mode feedback law

$$\mu_d(k, \mathbf{x}_k) := \mathbf{u}_d^*(k). \quad (18)$$

3.2.2 Recursive Feasibility and Stability Guarantees

For practical MPC implementations, it is essential to guarantee that the optimization problem (17) remains solvable at every sampling instant throughout closed-loop operation. This property, referred to as *recursive feasibility*, prevents the control algorithm from encountering infeasibility during runtime, which would result in undefined control actions. The constraint structure of the performance mode, combined with the invariance properties established for the safety mode, provides a mechanism for guaranteeing recursive feasibility.

Theorem 3 (Recursive Feasibility) *Consider the dual-mode framework with performance-mode optimization (17), feedback law (18), and the MPI set $\mathcal{O}_{\mathbf{K}}^\infty$ constructed under the safety-mode state feedback $\mathbf{u} = -\mathbf{K}\mathbf{x}$. If $\mathbf{x}_k \in \mathcal{O}_{\mathbf{K}}^\infty$, then the optimization problem (17) is feasible at time k and the successor state satisfies $\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mu_d(k, \mathbf{x}_k) \in \mathcal{O}_{\mathbf{K}}^\infty$.*

Proof Assume $\mathbf{x}_k \in \mathcal{O}_{\mathbf{K}}^\infty$. We construct a candidate input sequence $\{\hat{\mathbf{u}}(i)\}_{i=k}^{k+N_c-1}$ by applying the safety-mode feedback according to $\hat{\mathbf{u}}(i) = -\mathbf{K}\hat{\mathbf{x}}(i)$ for $i \in \{k, \dots, k+N_c-1\}$, where $\hat{\mathbf{x}}(k) = \mathbf{x}_k$ and $\hat{\mathbf{x}}(i+1) = (\mathbf{A} - \mathbf{BK})\hat{\mathbf{x}}(i) = \mathbf{A}_{\mathbf{K}}\hat{\mathbf{x}}(i)$. Since $\mathcal{O}_{\mathbf{K}}^\infty \subseteq \mathcal{X}_{\mathbf{K}}$ is positive invariant under $\mathbf{A}_{\mathbf{K}}$ per construction, it follows that $\hat{\mathbf{x}}(i) \in \mathcal{O}_{\mathbf{K}}^\infty$ for all $i \geq k$. By definition of $\mathcal{X}_{\mathbf{K}}$ in (13), this implies $\hat{\mathbf{u}}(i) \in \mathcal{U}$, satisfying (17c). Hence, the candidate sequence is feasible.

For the successor state, constraint (17b) enforces $\mathbf{x}^*(k+1) \in \mathcal{O}_{\mathbf{K}}^\infty$ for any optimal trajectory. Since $\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_d^*(k) = \mathbf{x}^*(k+1)$, membership $\mathbf{x}_{k+1} \in \mathcal{O}_{\mathbf{K}}^\infty$ follows directly. $\square$

An immediate consequence of Theorem 3 is that initialization within the MPI set guarantees feasibility for all future time steps: if $\mathbf{x}_0 \in \mathcal{O}_{\mathbf{K}}^\infty$, then the dual-mode MPC procedure according to Algorithm 1 with the feedback law (18) is well-defined for all $k \in \mathbb{N}_0$, and the closed-loop state trajectory satisfies $\mathbf{x}_k \in \mathcal{O}_{\mathbf{K}}^\infty$ for all $k \in \mathbb{N}_0$.

The separation of stability and performance objectives represents a key advantage over conventional MPC formulations. By leveraging the precomputed MPI set, the performance mode focuses exclusively on tracking fidelity while inheriting stability guarantees from the safety controller. This decoupling enables tuning of $\mathbf{Q}$ and $\mathbf{R}$ in (17a) for perceptual quality without compromising closed-loop stability. Furthermore, the control horizon N_c and prediction horizon N_p can be selected based on computational budget and available preview information rather than stability requirements, since asymptotic convergence is externally certified. In the following section, we derive the numerical implementation of the performance-mode optimization problem as a quadratic program suitable for real-time computation.

4 Numerical Implementation

The performance-mode optimization problem (17) must be solved at every discrete time step within the sampling interval T_s , which for modern driving simulators typically lies in the order of milliseconds [2]. This real-time constraint necessitates computational methods with deterministic execution times. Since the prediction model is linear, the cost function is quadratic, and all constraints are polyhedral, the MPC problem can be reformulated as a convex quadratic program (QP) of the form

$$\begin{aligned} & \underset{\mathbf{z} \in \mathbb{R}^{n_z}}{\text{minimize}} && \frac{1}{2} \mathbf{z}^\top \mathbf{P} \mathbf{z} + \mathbf{q}^\top \mathbf{z} \\ & \text{subject to} && \mathbf{G} \mathbf{z} \leq \mathbf{h}, \end{aligned}$$

where $\mathbf{P} \in \mathbb{R}^{n_z \times n_z}$ is symmetric and positive definite, $\mathbf{q} \in \mathbb{R}^{n_z}$, $\mathbf{G} \in \mathbb{R}^{n_{iq} \times n_z}$, and $\mathbf{h} \in \mathbb{R}^{n_{iq}}$. In this section, we derive the QP formulation by eliminating the predicted state and output trajectory through recursive substitution of the state equation.

4.1 Rewriting the Cost Function

For ease of notation, the stacked control sequence $\bar{\mathbf{u}} \in \mathbb{R}^{N_c m}$, state vector $\bar{\mathbf{x}} \in \mathbb{R}^{N_p n}$, output vector $\bar{\mathbf{y}} \in \mathbb{R}^{N_p p}$, and reference vector $\bar{\mathbf{r}} \in \mathbb{R}^{N_p p}$ are introduced as

$$\bar{\mathbf{x}} = [\mathbf{x}(k+1)^\top \ \mathbf{x}(k+2)^\top \ \dots \ \mathbf{x}(k+N_p)^\top]^\top,$$

$$\begin{aligned}\bar{\mathbf{u}} &= [\mathbf{u}(k)^\top \ \mathbf{u}(k+1)^\top \ \cdots \ \mathbf{u}(k+N_c-1)^\top]^\top, \\ \bar{\mathbf{y}} &= [\mathbf{y}(k)^\top \ \mathbf{y}(k+1)^\top \ \cdots \ \mathbf{y}(k+N_p-1)^\top]^\top, \\ \bar{\mathbf{r}} &= [\mathbf{r}(k)^\top \ \mathbf{r}(k+1)^\top \ \cdots \ \mathbf{r}(k+N_p-1)^\top]^\top.\end{aligned}$$

To express the predicted trajectories in dependence of $\mathbf{x}_k$ and $\bar{\mathbf{u}}$, recursive application of the state space equations (1) yields

$$\mathbf{y}(k+i) = \mathbf{C}\mathbf{A}^i\mathbf{x}_k + \sum_{j=0}^{i-1} \mathbf{C}\mathbf{A}^{i-j-1}\mathbf{B}\mathbf{u}(k+j) + \mathbf{D}\mathbf{u}(k+i), \quad i = 0, 1, \dots, N_c - 1.$$

For $i \geq N_c$, the control inputs are fixed by the safety-mode feedback $\mathbf{u}(k+i) = -\mathbf{K}\mathbf{x}(k+i)$. Defining $\mathbf{C}_\mathbf{K} := \mathbf{C} - \mathbf{D}\mathbf{K}$ leads to

$$\mathbf{y}(k+i) = \mathbf{C}_\mathbf{K}\mathbf{A}_\mathbf{K}^{i-N_c} \left(\mathbf{A}^{N_c}\mathbf{x}_k + \sum_{j=0}^{N_c-1} \mathbf{A}^{N_c-1-j}\mathbf{B}\mathbf{u}(k+j) \right), \quad i = N_c, \dots, N_p - 1.$$

Hence, a stacked state space representation can be written as

$$\bar{\mathbf{x}} = \bar{\mathbf{A}}\mathbf{x}_k + \bar{\mathbf{B}}\bar{\mathbf{u}}, \quad \bar{\mathbf{y}} = \bar{\mathbf{C}}\mathbf{x}_k + \bar{\mathbf{D}}\bar{\mathbf{u}}, \quad (19)$$

where

$$\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \vdots \\ \mathbf{A}^{N_c} \\ \mathbf{A}_\mathbf{K}\mathbf{A}^{N_c} \\ \vdots \\ \mathbf{A}_\mathbf{K}^{N_p-N_c}\mathbf{A}^{N_c} \end{bmatrix}, \quad \bar{\mathbf{B}} = \begin{bmatrix} \mathbf{B} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{AB} & \mathbf{B} & \cdots & \mathbf{O} \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{A}^{N_c-1}\mathbf{B} & \cdots & \mathbf{AB} & \mathbf{B} \\ \mathbf{A}_\mathbf{K}\mathbf{A}^{N_c-1}\mathbf{B} & \cdots & \mathbf{A}_\mathbf{K}\mathbf{AB} & \mathbf{A}_\mathbf{K}\mathbf{B} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{A}_\mathbf{K}^{N_p-N_c}\mathbf{A}^{N_c-1}\mathbf{B} & \cdots & \mathbf{A}_\mathbf{K}^{N_p-N_c}\mathbf{AB} & \mathbf{A}_\mathbf{K}^{N_p-N_c}\mathbf{B} \end{bmatrix},$$

and

$$\bar{\mathbf{C}} = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \vdots \\ \mathbf{CA}^{N_c-1} \\ \mathbf{C}_\mathbf{K}\mathbf{A}^{N_c} \\ \vdots \\ \mathbf{C}_\mathbf{K}\mathbf{A}_\mathbf{K}^{N_p-N_c-1}\mathbf{A}^{N_c} \end{bmatrix}, \quad \bar{\mathbf{D}} = \begin{bmatrix} \mathbf{D} & \cdots & \mathbf{O} \\ \mathbf{CB} & \ddots & \vdots \\ \vdots & \ddots & \mathbf{O} \\ \mathbf{CA}^{N_c-2}\mathbf{B} & \cdots & \mathbf{D} \\ \mathbf{C}_\mathbf{K}\mathbf{A}^{N_c-1}\mathbf{B} & \cdots & \mathbf{C}_\mathbf{K}\mathbf{B} \\ \vdots & \ddots & \vdots \\ \mathbf{C}_\mathbf{K}\mathbf{A}_\mathbf{K}^{N_p-N_c-1}\mathbf{A}^{N_c-1}\mathbf{B} & \cdots & \mathbf{C}_\mathbf{K}\mathbf{A}_\mathbf{K}^{N_p-N_c-1}\mathbf{B} \end{bmatrix}.$$

Substituting (19) into the cost function of (17), introducing block-diagonal weighting matrices $\bar{\mathbf{Q}} = \text{diag}(\mathbf{Q}, \dots, \mathbf{Q}) \in \mathbb{R}^{N_p p \times N_p p}$ and $\bar{\mathbf{R}} = \text{diag}(\mathbf{R}, \dots, \mathbf{R}) \in \mathbb{R}^{N_c m \times N_c m}$, and omitting constant terms, the objective takes the standard QP form

$$J = \frac{1}{2} \bar{\mathbf{u}}^\top \bar{\mathbf{P}} \bar{\mathbf{u}} + \bar{\mathbf{q}}^\top \bar{\mathbf{u}}, \quad (20)$$

with

$$\bar{\mathbf{P}} := \bar{\mathbf{D}}^\top \bar{\mathbf{Q}} \bar{\mathbf{D}} + \bar{\mathbf{R}}, \quad \bar{\mathbf{q}} := \bar{\mathbf{D}}^\top \bar{\mathbf{Q}} (\bar{\mathbf{C}} \mathbf{x}_k - \bar{\mathbf{r}}).$$

The matrix $\bar{\mathbf{P}}$ is symmetric and positive definite if $\bar{\mathbf{Q}} \succeq \mathbf{O}$ and $\bar{\mathbf{R}} \succ \mathbf{O}$, ensuring strict convexity and uniqueness of the optimal solution.

4.2 Constraint Reformulation

For input admissibility $\mathbf{u}(k+i) \in \mathcal{U}$ with $i \in \{0, \dots, N_c - 1\}$, the constraints are stacked as

$$\bar{\mathbf{F}}_{\mathbf{u}} \bar{\mathbf{u}} \leq \bar{\mathbf{g}}_{\mathbf{u}}, \quad (21)$$

where $\bar{\mathbf{F}}_{\mathbf{u}} = \text{diag}(\mathbf{F}_{\mathbf{u}}, \dots, \mathbf{F}_{\mathbf{u}}) \in \mathbb{R}^{N_c n_{\mathbf{u}} \times N_c m}$ and $\bar{\mathbf{g}}_{\mathbf{u}} = [\mathbf{g}_{\mathbf{u}}^\top, \dots, \mathbf{g}_{\mathbf{u}}^\top]^\top \in \mathbb{R}^{N_c n_{\mathbf{u}}}$. State constraints $\mathbf{x}(i+1) \in \mathcal{O}_{\mathbf{K}}^\infty$ require explicit enforcement only through the control horizon, since positive invariance of $\mathcal{O}_{\mathbf{K}}^\infty$ under $\mathbf{u}(i) = -\mathbf{K}\mathbf{x}(i)$ ensures automatic satisfaction for $i \geq N_c$. Hence, for $i \in \{0, \dots, N_c - 1\}$ the state constraints can be written, using (19), as

$$\bar{\mathbf{F}}_\infty \bar{\mathbf{B}}_{N_c} \bar{\mathbf{u}} \leq \bar{\mathbf{g}}_\infty - \bar{\mathbf{F}}_\infty \bar{\mathbf{A}}_{N_c} \mathbf{x}_k. \quad (22)$$

Here, $\bar{\mathbf{A}}_{N_c} \in \mathbb{R}^{N_c n \times n}$ and $\bar{\mathbf{B}}_{N_c} \in \mathbb{R}^{N_c n \times N_c m}$ denote the first N_c block rows of the prediction matrices $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$, respectively. Furthermore, $\bar{\mathbf{F}}_\infty = \text{diag}(\mathbf{F}_{\mathbf{K}}^\infty, \dots, \mathbf{F}_{\mathbf{K}}^\infty) \in \mathbb{R}^{N_c n_\infty \times N_c n}$ and $\bar{\mathbf{g}}_\infty = [\mathbf{g}_{\mathbf{K}}^{\infty\top}, \dots, \mathbf{g}_{\mathbf{K}}^{\infty\top}]^\top \in \mathbb{R}^{N_c n_\infty}$, where n_∞ denotes the number of inequalities defining the MPI set $\mathcal{O}_{\mathbf{K}}^\infty$.

Combining (20), (21) and (22), the performance-mode optimization problem (17) is reformulated as

$$\begin{aligned} & \underset{\bar{\mathbf{u}} \in \mathbb{R}^{N_c m}}{\text{minimize}} && \frac{1}{2} \bar{\mathbf{u}}^\top \bar{\mathbf{P}} \bar{\mathbf{u}} + \bar{\mathbf{q}}^\top \bar{\mathbf{u}} \\ & \text{subject to} && \bar{\mathbf{G}} \bar{\mathbf{u}} \leq \bar{\mathbf{h}}, \end{aligned} \quad (23)$$

with

$$\bar{\mathbf{G}} = \begin{bmatrix} \bar{\mathbf{F}}_{\mathbf{u}} \\ \bar{\mathbf{F}}_\infty \bar{\mathbf{B}}_{N_c} \end{bmatrix} \in \mathbb{R}^{n_{\text{iq}} \times N_c m}, \quad \bar{\mathbf{h}} = \begin{bmatrix} \bar{\mathbf{g}}_{\mathbf{u}} \\ \bar{\mathbf{g}}_\infty - \bar{\mathbf{F}}_\infty \bar{\mathbf{A}}_{N_c} \mathbf{x}_k \end{bmatrix} \in \mathbb{R}^{n_{\text{iq}}},$$

where $n_{\text{iq}} = N_c(n_{\mathbf{u}} + n_\infty)$ denotes the total number of inequality constraints.

Remark 5 (Problem Dimensions) The QP formulation (23) has $N_c m$ decision variables and $N_c(n_{\mathbf{u}} + n_\infty)$ inequality constraints. Both quantities scale with the control horizon N_c rather than the prediction horizon N_p . This enables extended prediction horizons for preview exploitation in motion cueing applications without increasing the optimization problem size.

Remark 6 (Online Computation) At each time step k , the Hessian $\bar{\mathbf{P}}$ and the constraint matrix $\bar{\mathbf{G}}$ remain constant since they depend only on the system matrices and horizons. Only the gradient $\bar{\mathbf{q}}$ and the right-hand side $\bar{\mathbf{h}}$ require updating based on the reference $\bar{\mathbf{r}}$ and the current state $\mathbf{x}_k$. This structure permits efficient warm-starting strategies.

4.3 QP Solver Selection

The standard QP formulation (23) can be solved using a variety of numerical optimization algorithms. In the context of real-time MPC, active-set methods are particularly attractive, as they naturally support warm starting. By initializing the optimization at each sampling instant with the solution obtained at the previous time step, only a small number of iterations is typically required to reach convergence [39].

Since (23) contains only inequality constraints, it is compatible with active-set solvers such as qpOASES [40] and DAQP [41]. It is worth mentioning that the suitability of qpOASES for MPC-based motion cueing applications has been successfully demonstrated in previous work [18, 24]. This proven applicability in the motion cueing domain, coupled with its robust warm-starting capabilities, motivates the choice of qpOASES for the implementation on our robot-based simulator platform.

5 Application to a Robot-Based Driving Simulator

This section considers the application of the proposed dual-mode model predictive motion cueing framework to the robot-based driving and operation simulator RODOS[®] at the Fraunhofer ITWM. We first demonstrate how the established concept of tilt coordination, which enables the rendering of sustained accelerations, can be integrated in the proposed framework. Subsequently, an extended 8-DOF platform model incorporating dedicated tilt channels is derived and discretized. The constraint parameterization, reference generation strategy, and washout mechanism are discussed. A signal flow diagram illustrating the motion cueing architecture is provided in Fig. 3.

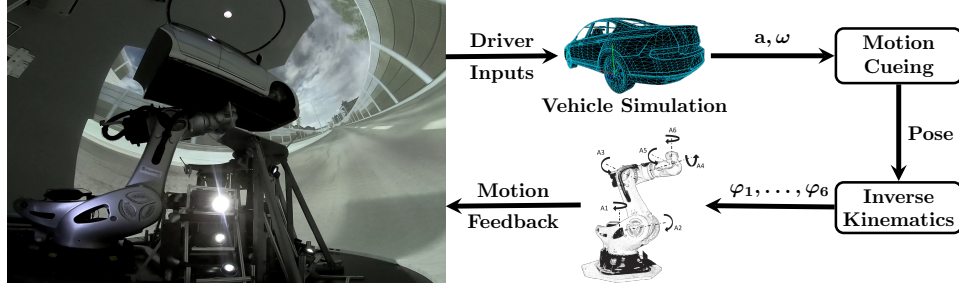


Fig. 3 Signal flow of RODOS[®]

5.1 Tilt Coordination

The reproduction of sustained accelerations using direct translational platform motion is fundamentally limited by the bounded workspace. The technique of *tilt coordination* addresses this limitation by exploiting a characteristic of the human vestibular system, namely the gravito-inertial ambiguity of the otolith organs [12]. These sensory organs respond to the resultant specific force vector rather than to acceleration and gravity

independently. The specific force $\mathbf{f}$ acting on the vestibular system is defined as

$$\mathbf{f} := \mathbf{a} + \mathbf{g},$$

where $\mathbf{a}$ denotes the translational acceleration and $\mathbf{g} = [0 \ 0 \ -g]^\top$ the gravitational acceleration, with $g = 9.81 \text{ m/s}^2$.

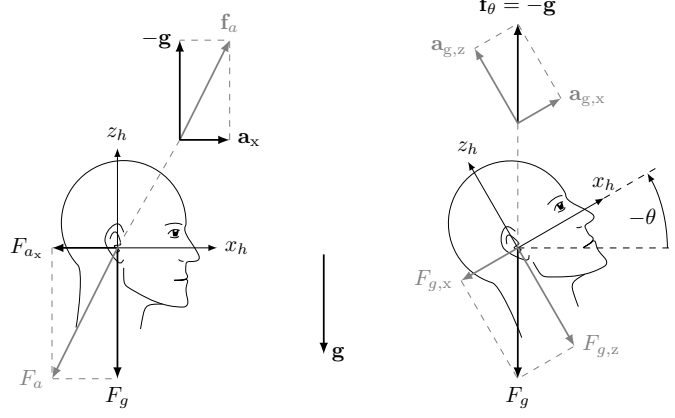


Fig. 4 Principle of tilt coordination

The principle of tilt coordination is illustrated in Fig. 4. For an upright orientation, a longitudinal translational acceleration a_x produces the specific force $\mathbf{f}_a = [a_x \ 0 \ -g]^\top$, sensed as inertial force $F_a = -\mathbf{f}_a = [-a_x \ 0 \ g]^\top$. Tilting the head backward by an angle θ redistributes the specific force $\mathbf{f}_\theta = [0 \ 0 \ g]^\top$ into head-fixed components, yielding $F_g = [F_{g,x} \ 0 \ F_{g,z}] = [g \sin \theta \ 0 \ -g \cos \theta]^\top$. Comparing the longitudinal components reveals that identical vestibular stimuli arise when $a_x = -g \sin \theta$. Hence, sustained translational accelerations can be substituted by appropriate platform tilt angles.

For small tilt angles, the approximations $\sin \theta \approx \theta$ and $\cos \theta \approx 1$ hold with sufficient accuracy for motion cueing purposes. Under this linearization, the perceived specific forces are related to the platform accelerations and tilt angles by

$$\begin{aligned} f_x &= a_x - g \theta_{\text{tilt}}, \\ f_y &= a_y + g \phi_{\text{tilt}}, \\ f_z &= a_z, \end{aligned} \tag{24}$$

where ϕ_{tilt} and θ_{tilt} denote the roll and pitch tilt angles, respectively. The effectiveness of tilt coordination depends critically on the angular rate at which the tilt angle changes. To avoid generating spurious rotational cues, the tilt rate should remain below the semicircular canal detection threshold, typically reported in the range of $0.5^\circ/\text{s}$ to $4^\circ/\text{s}$ [13, 34].

To enable independent parameterization of tilt-based acceleration rendering and physical platform rotation, we introduce two dedicated virtual tilt channels that are

decoupled from the physical orientation angles. This approach, following the methodology in [42], permits separate constraint specification: perceptual threshold constraints are imposed on the tilt angular velocities while the physical platform angular velocities remain subject to the mechanical limits of the robot.

5.2 Platform Model

Based on the tilt coordination concept introduced above, we derive an 8-DOF state-space model comprising six physical platform degrees of freedom and two virtual tilt channels. The motion platform dynamics are modeled as decoupled double integrators, one for each degree of freedom. To derive the discrete-time LTI state space representation (1), the input vector

$$\mathbf{u} = [\mathbf{a}^\top \ \boldsymbol{\alpha}^\top \ \boldsymbol{\alpha}_{\text{tilt}}^\top]^\top \in \mathbb{R}^8$$

is introduced, comprising the translational accelerations $\mathbf{a} = [a_x \ a_y \ a_z]^\top$, the angular accelerations $\boldsymbol{\alpha} = [\alpha_\phi \ \alpha_\theta \ \alpha_\psi]^\top$ corresponding to roll, pitch, and yaw rotations, and the tilt coordination angular accelerations $\boldsymbol{\alpha}_{\text{tilt}} = [\alpha_{\phi,\text{tilt}} \ \alpha_{\theta,\text{tilt}}]^\top$. The state vector

$$\mathbf{x} = [\mathbf{v}^\top \ \boldsymbol{\omega}^\top \ \boldsymbol{\omega}_{\text{tilt}}^\top \ \mathbf{p}^\top \ \boldsymbol{\rho}^\top \ \boldsymbol{\rho}_{\text{tilt}}^\top]^\top \in \mathbb{R}^{16} \quad (25)$$

contains the translational velocities $\mathbf{v} = [v_x \ v_y \ v_z]^\top$, the angular velocities $\boldsymbol{\omega} = [\omega_\phi \ \omega_\theta \ \omega_\psi]^\top$, the tilt angular velocities $\boldsymbol{\omega}_{\text{tilt}} = [\omega_{\phi,\text{tilt}} \ \omega_{\theta,\text{tilt}}]^\top$, the translational positions $\mathbf{p} = [p_x \ p_y \ p_z]^\top$, the platform orientation angles $\boldsymbol{\rho} = [\phi \ \theta \ \psi]^\top$, and the virtual tilt angles $\boldsymbol{\rho}_{\text{tilt}} = [\phi_{\text{tilt}} \ \theta_{\text{tilt}}]^\top$.

The complete continuous-time state equation describing double integrator dynamics for all degrees of freedom is given by

$$\dot{\mathbf{x}}(t) = \mathbf{A}_c \mathbf{x}(t) + \mathbf{B}_c \mathbf{u}(t),$$

with the continuous-time system matrices

$$\mathbf{A}_c = \begin{bmatrix} \mathbf{O}_8 & \mathbf{O}_8 \\ \mathbf{I}_8 & \mathbf{O}_8 \end{bmatrix} \in \mathbb{R}^{16 \times 16} \quad \text{and} \quad \mathbf{B}_c = \begin{bmatrix} \mathbf{I}_8 \\ \mathbf{O}_8 \end{bmatrix} \in \mathbb{R}^{16 \times 8}.$$

Applying zero-order hold discretization with sampling period T_s yields the discrete-time state equation

$$\mathbf{x}(k+1) = \mathbf{A} \mathbf{x}(k) + \mathbf{B} \mathbf{u}(k),$$

with

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_8 & \mathbf{O}_8 \\ T_s \mathbf{I}_8 & \mathbf{I}_8 \end{bmatrix} \in \mathbb{R}^{16 \times 16} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} T_s \mathbf{I}_8 \\ \frac{1}{2} T_s^2 \mathbf{I}_8 \end{bmatrix} \in \mathbb{R}^{16 \times 8}.$$

While the components of the state vector (25) are determined by the platform dynamics model, the output vector can be tailored to include all quantities relevant for tracking and washout behavior (see Sect. 5.5). Accordingly, the output vector is defined as

$$\mathbf{y} = [\mathbf{f}^\top \ \mathbf{x}^\top]^\top \in \mathbb{R}^{19}$$

comprising the perceived specific force $\mathbf{f}(\mathbf{x}, \mathbf{u}) = [f_x \ f_y \ f_z]^\top$ and the complete platform state. According to the tilt coordination relationships (24), the specific force $\mathbf{f}$ depends linearly on the virtual tilt angles contained in $\mathbf{x}$ and the translational accelerations contained in $\mathbf{u}$. This structure yields the output equation

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k),$$

with

$$\mathbf{C} = \begin{bmatrix} \mathbf{O}_{3 \times 14} & \mathbf{G}_{\text{tilt}} \\ & \mathbf{I}_{16} \end{bmatrix} \in \mathbb{R}^{19 \times 16}, \quad \mathbf{D} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{O}_{3 \times 5} \\ & \mathbf{O}_{16 \times 8} \end{bmatrix} \in \mathbb{R}^{19 \times 8}.$$

The tilt coordination matrix

$$\mathbf{G}_{\text{tilt}} = \begin{bmatrix} 0 & -g \\ g & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{3 \times 2}$$

maps the virtual tilt angles $\boldsymbol{\rho}_{\text{tilt}} = [\phi_{\text{tilt}} \ \theta_{\text{tilt}}]^\top$ to the corresponding specific force components in accordance with (24).

5.3 Constraint Specification

The admissible constraint sets are determined by the physical capabilities of the RODOS[®] platform. The velocity and acceleration limits were experimentally identified using sine-sweep excitation with increasing frequency. Table 1 summarizes the resulting bounds for each physical degree of freedom.

Table 1 Physical limits of the RODOS[®] platform

DoF	Acceleration	Velocity	Position
x	$\pm 6.0 \text{ m/s}^2$	$\pm 1.5 \text{ m/s}$	$[-0.55 \text{ m}, 0.3 \text{ m}]$
y	$\pm 8.0 \text{ m/s}^2$	$\pm 2.0 \text{ m/s}$	$\pm 1.6 \text{ m}$
z	$\pm 7.0 \text{ m/s}^2$	$\pm 1.0 \text{ m/s}$	$\pm 0.1 \text{ m}$
ϕ	$\pm 200^\circ/\text{s}^2$	$\pm 60^\circ/\text{s}$	$\pm 35^\circ$
θ	$\pm 200^\circ/\text{s}^2$	$\pm 60^\circ/\text{s}$	$\pm 15^\circ$
ψ	$\pm 50^\circ/\text{s}^2$	$\pm 60^\circ/\text{s}$	$\pm 50^\circ$

The polyhedral constraint sets $\mathcal{U}$ and $\mathcal{X}$ are constructed from these bounds according to (3) as demonstrated in Example 2. For the decoupled tilt coordination channels, a portion of the rotational workspace is allocated to ensure subthreshold tilt motion. Specifically, the virtual tilt channels are subject to the constraints

$$|\phi_{\text{tilt}}|, |\theta_{\text{tilt}}| \leq 5^\circ, \quad |\omega_{\phi, \text{tilt}}|, |\omega_{\theta, \text{tilt}}| \leq 5^\circ/\text{s}, \quad |\alpha_{\phi, \text{tilt}}|, |\alpha_{\theta, \text{tilt}}| \leq 20^\circ/\text{s}^2.$$

The tilt angle bounds of $\pm 5^\circ$ remain within the range where small-angle approximation in (24) holds with sufficient accuracy. The tilt rate constraint of $5^\circ/\text{s}$ is selected to lie slightly above typical semicircular canal detection thresholds [34, p. 26] while providing sufficient margin for smooth tilt coordination. By partitioning the rotational workspace, the physical platform angular velocities $\boldsymbol{\omega}$ and orientations $\boldsymbol{\rho}$ retain the remaining capacity according to Table 1 for reproducing high-frequency rotational movements.

5.4 Reference Generation

The performance-mode optimization (17) requires a reference trajectory $\{\mathbf{r}(i)\}_{i=k}^{k+N_p-1}$ spanning the entire prediction horizon. In passive driving simulation, where the vehicle trajectory is predetermined, the future reference values are directly available from precomputed scenario data. However, in interactive driving simulation, the future vehicle motion depends on driver inputs that are inherently unknown at the current time step k . This fundamental distinction necessitates reference prediction strategies for interactive applications.

Several approaches have been proposed to address this challenge. The simplest strategy is constant extrapolation, where the current vehicle accelerations and angular velocities are held constant over the prediction horizon [18]. A refinement employs prerecorded reference trajectories from previous test drives on a known track, using adaptive scaling factors to account for driver variability [43]. Machine learning approaches have also been explored, where neural networks trained on past motion cues predict future vehicle dynamics based on learned driver behavior patterns [44]. More sophisticated model-based methods combine simplified vehicle models with linear feedback control laws for path and speed tracking to anticipate future driver inputs based on known route information [45]. The most elaborate approaches model the driver as an optimal controller pursuing a predicted path, formulating and solving an additional optimization problem at each time step to estimate future driver inputs [34, 46].

For the implementation presented herein, we adopt constant reference extrapolation due to its computational simplicity:

$$\mathbf{r}(i) = \mathbf{r}(k) \quad \text{for } i \in \{k, k+1, \dots, k+N_p-1\}.$$

The receding horizon principle ensures that the reference trajectory is updated at each sampling instant, thereby incorporating the latest vehicle state information.

5.5 Washout Mechanism

The desired *washout* effect ensures that the motion platform returns toward its neutral configuration when vehicle motion demands subside. Within the proposed dual-mode architecture, washout can be achieved in two ways: implicitly through appropriate parameterization of the weighting matrices in the performance-mode optimization, or explicitly through manual activation of the safety mode.

5.5.1 Implicit Washout via Cost Function Design

The implicit washout behavior emerges from the reference structure and the weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ in the performance-mode optimization problem (17). The reference vector at time step k is constructed as

$$\mathbf{r}(k) = [\mathbf{a}_{\text{veh}}(k)^\top \mathbf{0}_3^\top \boldsymbol{\omega}_{\text{veh}}(k)^\top \mathbf{0}_{10}^\top]^\top \in \mathbb{R}^{19},$$

where $\mathbf{a}_{\text{veh}}(k) \in \mathbb{R}^3$ denotes the simulated vehicle accelerations and $\boldsymbol{\omega}_{\text{veh}}(k) \in \mathbb{R}^3$ represents the vehicle angular velocities at time k . The zero entries correspond to translational velocities $\mathbf{v}$, tilt angular velocities $\boldsymbol{\omega}_{\text{tilt}}$, translational positions $\mathbf{p}$, platform orientations $\boldsymbol{\rho}$, and tilt angles $\boldsymbol{\rho}_{\text{tilt}}$.

The cost function (17a) simultaneously pursues two objectives: tracking the nonzero reference components, which requires reproducing vehicle accelerations and angular velocities, and driving the remaining output components toward zero. The latter objective effects the washout behavior, as minimizing the tracking error for zero-referenced states penalizes deviations from the neutral platform configuration. It is important to note that the weighting matrix $\mathbf{Q} \in \mathbb{R}^{19 \times 19}$ directly parameterizes the trade-off between motion fidelity and washout aggressiveness. Larger weights on the specific force components f_x , f_y , f_z prioritize accurate reproduction of vehicle accelerations, while larger weights on position and orientation components promote faster return toward the neutral configuration. The input weighting matrix $\mathbf{R} \in \mathbb{R}^{8 \times 8}$ penalizes control effort, thereby influencing the smoothness of both tracking and washout motions.

5.5.2 Explicit Washout via Safety Mode Activation

The safety mode provides an alternative mechanism for initiating platform washout. By activating the stabilizing state feedback $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$, the platform is driven toward the origin. This explicit washout capability is particularly relevant in scenarios requiring immediate return to the neutral position, such as emergency stops, simulation termination, or operator-initiated session endings. Since any state within $\mathcal{O}_{\mathbf{K}}^\infty$ can be asymptotically stabilized to the origin under the safety-mode feedback while respecting all input and state constraints, the explicit washout is guaranteed to succeed without constraint violations.

5.6 Practical Advantages

The dual-mode architecture offers several practical advantages beyond the formal stability and feasibility guarantees established in Section 3. First, the architecture provides computational robustness through an inherent solver fallback mechanism. If the QP solver fails to converge within the sampling period T_s , the safety-mode feedback $\mathbf{u}(k) = -\mathbf{K}\mathbf{x}(k)$ can be applied immediately without risking constraint violations, since the current state necessarily satisfies $\mathbf{x}_k \in \mathcal{O}_{\mathbf{K}}^\infty$. This enables aggressive solver configurations without compromising safety. Second, the weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ can be modified during closed-loop operation without invalidating stability guarantees. Since asymptotic stability is certified by the safety mode independently

of the performance-mode parameters, online tuning for driver personalization or context-adaptive motion cueing becomes feasible. Third, the externalization of stability certification removes the need for terminal costs or terminal equality constraints in the MPC formulation. This design freedom permits the cost function to be tailored exclusively for motion fidelity, and the prediction horizon N_p and control horizon N_c can be selected based on preview availability and computational budget rather than stability considerations.

5.7 Simulation Results

To demonstrate the dual-mode switching capability, we consider a longitudinal acceleration maneuver where the safety mode is activated during operation. The simulation employs the discrete-time double integrator model with the state feedback gain $\mathbf{K}$ from Example 1. The MPI set $\mathcal{O}_{\mathbf{K}}^\infty$ for the x -axis is taken from Example 2, respectively. The performance-mode optimization (17) is configured with prediction horizon $N_p = 20$ and control horizon $N_c = 12$. The weighting parameters are chosen as

$$\begin{aligned} q_{f_x} &= 1.0, & q_{v_x} &= 0.1, & q_{p_x} &= 50.0, & r_{a_x} &= 1.0, \\ q_{\omega_{\theta, \text{tilt}}} &= 50.0, & q_{\theta_{\text{tilt}}} &= 50.0, & r_{\alpha_x} &= 2.0. \end{aligned}$$

The simulation comprises a 10 s scenario with the vehicle reference acceleration shown in Fig. 5. At $t = 6$ s, the safety mode is manually activated, simulating an operator-initiated session termination.

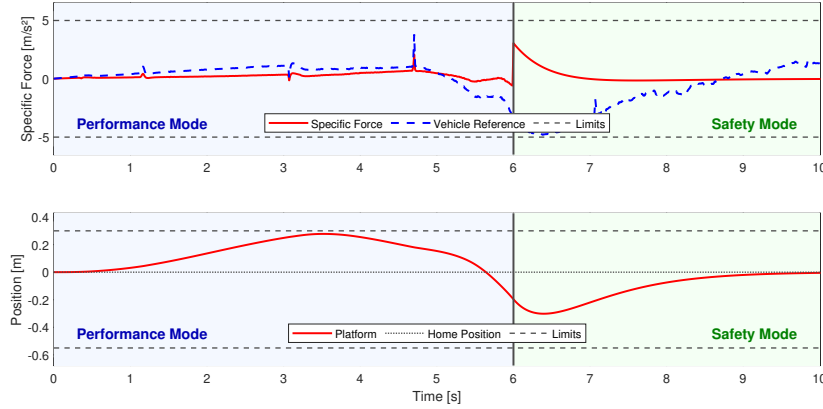


Fig. 5 Dual-mode motion cueing for acceleration maneuver

The upper plot displays the vehicle reference acceleration and the resulting specific force acceleration. During the performance mode ($t < 6$ s), the MPC tracks the reference while respecting all constraints according to Table 1. Upon safety mode activation, the platform acceleration transitions to the stabilizing feedback response

$a_x(k) = -\mathbf{K}\mathbf{x}(k)$. The lower plot shows the platform position p_x , which remains within the workspace bounds throughout the maneuver. After safety mode activation, the platform returns smoothly toward the neutral position $p_x = 0$ without constraint violations, confirming the guaranteed safe return capability.

6 Conclusion

This paper has presented a dual-mode model predictive control architecture for motion cueing that systematically separates safety certification from performance optimization. The safety mode, employing a stabilizing state feedback law and an associated maximal positive invariant set, provides formal guarantees for constraint satisfaction and asymptotic stability. By constraining all predicted states to remain within this invariant set, the performance mode can be designed exclusively for motion fidelity without requiring terminal costs or terminal equality constraints. Recursive feasibility and closed-loop stability follow directly from the invariance properties of the safety mode, independently of the MPC weighting parameters or horizon selection. The framework has been implemented in the robot-based driving simulator RODOS[®], which showcases the integration of tilt coordination through an expanded 8-DOF platform model with dedicated virtual tilt channels. Simulation results confirm the guaranteed safe return capability upon safety mode activation, validating the theoretical guarantees established in the dual-mode formulation. The architecture offers several practical advantages, including computational robustness through an inherent solver fallback mechanism, the ability to modify tuning parameters during closed-loop operation, and design freedom in the MPC formulation.

For future work, it would be interesting to apply the proposed dual-mode concept to other dynamic driving simulator architectures, particularly hexapod-based Stewart platforms, which represent the most widely deployed motion platform configuration in professional driving simulation. Furthermore, the incorporation of vestibular perception models into the framework could enhance motion fidelity by accounting for human sensory dynamics in the cost function design. The architectural separation between safety certification and performance optimization suggests a natural extension to nonlinear MPC formulations, since the safety mode guarantees remain valid independently of the specific cost function structure employed in the performance mode. Finally, investigating advanced reference generation strategies that model the driver as an optimal controller pursuing a predicted path could improve preview exploitation in interactive driving simulation scenarios.

Author contributions

Conceptualization: TN; Methodology: TN; Formal analysis and investigation: TN; Writing - original draft preparation: TN; Writing - review and editing: TN, SE, MB, MG

Funding

Not applicable.

Data Availability

Not applicable.

Materials Availability

Not applicable.

Code Availability

Not applicable.

Declarations**Competing interests**

The authors declare no competing interests.

Ethics approval and consent to participate

Not applicable.

References

- [1] Baumgartner, E., Ronellenfitsch, A., Reuss, H.-C., Schramm, D.: Using a dynamic driving simulator for perception-based powertrain development. *Transportation research part F: traffic psychology and behaviour* **61**, 281–290 (2019) <https://doi.org/10.1016/j.trf.2017.08.012>
- [2] Bruck, L., Haycock, B., Emadi, A.: A Review of Driving Simulation Technology and Applications. *IEEE Open Journal of Vehicular Technology* **2**, 1–16 (2021) <https://doi.org/10.1109/OJVT.2020.3036582>
- [3] Markkula, G., Romano, R., Waldram, R., Giles, O., Mole, C., Wilkie, R.: Modelling visual-vestibular integration and behavioural adaptation in the driving simulator. *Transportation Research Part F: Traffic Psychology and Behaviour* **66**, 310–323 (2019) <https://doi.org/10.1016/j.trf.2019.07.018>
- [4] Bruschetta, M., Winkel, K.N., Mion, E., Pretto, P., Beghi, A., Bühlhoff, H.H.: Assessing the contribution of active somatosensory stimulation to self-acceleration perception in dynamic driving simulators. *PLOS ONE* **16**(11), 1–18 (2021) <https://doi.org/10.1371/journal.pone.0259015>
- [5] Himmels, C.: A use-case driven approach to establishing driving simulator validity. PhD thesis, Johannes Kepler University Linz (2025). <https://resolver.obvsg.at/urn:nbn:at:at-ubl:1-88046>
- [6] Kolff, M.J.C.: Motion Cueing Quality in Driving Simulation. PhD thesis, Delft University of Technology (2025). <https://doi.org/10.4233/UUID:A9C8ABB2-C45D-400E-8EF1-ED1C76515B44>

- [7] Ellensohn, F.: Urban motion cueing algorithms. PhD thesis, Technische Universität München (2020). <https://nbn-resolving.org/urn:nbn:de:bvb:91-diss-20200115-1515163-1-9>
- [8] Conrad, B., Schmidt, S.F.: Motion drive signals for piloted flight simulators. Technical Report NASA-CR-1601, National Aeronautics and Space Administration, Ames Research Center (1970)
- [9] Reid, L.D., Nahon, M.A.: Flight simulation motion-base drive algorithms: part 1. Developing and testing equations. Technical report, Canadian Transportation Development Center (1985)
- [10] Nahon, M.A., Reid, L.D.: Simulator motion-drive algorithms-a designer's perspective. *Journal of Guidance, Control, and Dynamics* **13**(2), 356–362 (1990) <https://doi.org/10.2514/3.20557>
- [11] Grant, P.R., Reid, L.D.: Motion washout filter tuning: Rules and requirements. *Journal of Aircraft* **34**, 145–151 (1997) <https://doi.org/10.2514/2.2158>
- [12] Angelaki, D.E., Wei, M., Merfeld, D.M.: Vestibular Discrimination of Gravity and Translational Acceleration. *Annals of the New York Academy of Sciences* **942**, 114–127 (2001) <https://doi.org/10.1111/j.1749-6632.2001.tb03739.x>
- [13] Colombet, F., Fang, Z., Kemeny, A.: Tilt thresholds for acceleration rendering in driving simulation. *SIMULATION* **93** (2016) <https://doi.org/10.1177/0037549716675955>
- [14] Garrett, N.J., Best, M.: Driving simulator motion cueing algorithms—a survey of the state of the art. In: 10th International Symposium on Advanced Vehicle Control (AVEC), Loughborough (2010). <https://hdl.handle.net/2134/8325>
- [15] Jamson, A.H.J.: Motion cueing in driving simulators for research applications. PhD thesis, University of Leeds (2010). <https://etheses.whiterose.ac.uk/id/eprint/1915/>
- [16] Qazani, M.R.C., Asadi, H., Rostami, M., Mohamed, S., Lim, C.P., Nahavandi, S.: Adaptive motion cueing algorithm based on fuzzy logic using online dexterity and direction monitoring. *IEEE Systems Journal* **16**(2), 1945–1953 (2022) <https://doi.org/10.1109/JSYST.2021.3059285>
- [17] Dagdelen, M., Reymond, G., Kemeny, A., Bordier, M., Maïki, N.: Mpc based motion cueing algorithm: Development and application to the ultimate driving simulator. In: DSC 2004 Europe (driving Simulation Conference), pp. 221–233 (2004)
- [18] Beghi, A., Bruschetta, M., Maran, F.: A real time implementation of MPC based

- Motion Cueing strategy for driving simulators. In: 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), pp. 6340–6345. IEEE, Maui, HI, USA (2012). <https://doi.org/10.1109/CDC.2012.6426119>
- [19] Chalak Qazani, M.R., Asadi, H., Zhang, L., Tabarsinezhad, F., Mohamed, S., Lim, C.P., Nahavandi, S.: A new prepositioning technique of a motion simulator platform using nonlinear model predictive control and recurrent neural network. *IEEE Transactions on Intelligent Transportation Systems* **23**(12), 23268–23277 (2022) <https://doi.org/10.1109/TITS.2022.3195964>
 - [20] Cleij, D., Venrooij, J., Pretto, P., Katliar, M., Bühlhoff, H.H., Steffen, D., Hoffmeyer, F.W., Schöner, H.-P.: Comparison between filter-and optimization-based motion cueing algorithms for driving simulation. *Transportation Research Part F: Traffic Psychology and Behaviour* **61**, 53–68 (2019) <https://doi.org/10.1016/j.trf.2017.04.005>
 - [21] Biemelt, P., Böhm, S., Gausemeier, S., Trächtler, A.: Subjective evaluation of filter- and optimization-based motion cueing algorithms for a hybrid kinematics driving simulator. In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1619–1626 (2021). <https://doi.org/10.1109/SMC52423.2021.9658974>
 - [22] Rawlings, J.B., Mayne, D.Q., Diehl, M.: *Model Predictive Control: Theory, Computation, and Design*, 2nd edition edn. Nob Hill Publishing, Madison, Wisconsin (2017)
 - [23] Grüne, L., Pannek, J.: *Nonlinear Model Predictive Control*. Communications and Control Engineering. Springer, Cham (2017). <https://doi.org/10.1007/978-3-319-46024-6>
 - [24] Fang, Z., Kemeny, A.: An efficient Model Predictive Control-based motion cueing algorithm for the driving simulator. *SIMULATION* **92**(11), 1025–1033 (2016) <https://doi.org/10.1177/0037549716667835>
 - [25] Qazani, M.R.C., Asadi, H., Nahavandi, S.: A Motion Cueing Algorithm Based on Model Predictive Control Using Terminal Conditions in Urban Driving Scenario. *IEEE Systems Journal* **15**(1), 445–453 (2021) <https://doi.org/10.1109/JSYST.2020.2994154>
 - [26] Lyapunov, A.M.: *The General Problem of the Stability of Motion*. Taylor & Francis, London (1992). Reprint of 1907 French translation
 - [27] Bof, N., Carli, R., Schenato, L.: *Lyapunov Theory for Discrete Time Systems*. arXiv (2018). <https://doi.org/10.48550/ARXIV.1809.05289>
 - [28] Sontag, E.D.: *Mathematical Control Theory*, 2nd edn. Texts in Applied Mathematics. Springer, New York, NY (1998). <https://doi.org/10.1007/>

- [29] Bachtiar, V., Kerrigan, E.C., Moase, W.H., Manzie, C.: Continuity and monotonicity of the MPC value function with respect to sampling time and prediction horizon. *Automatica* **63**, 330–337 (2016) <https://doi.org/10.1016/j.automatica.2015.10.042>
- [30] Morgenstern, D.: Model Predictive Control with Stabilizing Prediction Horizon of Systems Described by Time-Discretized ODEs and DAEs. Fraunhofer-Gesellschaft (2024). <https://doi.org/10.24406/PUBLICA-3618>
- [31] Bellman, R.: Dynamic Programming. *Science* **153**(3731), 34–37 (1966) <https://doi.org/10.1126/science.153.3731.34>
- [32] Rossiter, J.A.: Model-Based Predictive Control: A Practical Approach, 1st edn. CRC Press, Boca Raton (2017). <https://doi.org/10.1201/9781315272610>
- [33] Khusro, Y.R., Zheng, Y., Grotoli, M., Shyrokau, B.: MPC-Based Motion-Cueing Algorithm for a 6-DOF Driving Simulator with Actuator Constraints. *Vehicles* **2**(4), 625–647 (2020) <https://doi.org/10.3390/vehicles2040036>
- [34] Lamprecht, A., Steffen, D., Nagel, K., Haecker, J., Graichen, K.: Online Model Predictive Motion Cueing With Real-Time Driver Prediction. *IEEE Transactions on Intelligent Transportation Systems* **23**(8), 12414–12428 (2022) <https://doi.org/10.1109/TITS.2021.3114003>
- [35] Ackermann, J.: Der Entwurf linearer Regelungssysteme im Zustandsraum. *auto* **20**(1-12), 297–300 (1972) <https://doi.org/10.1524/auto.1972.20.112.297>
- [36] Kautsky, J., Nichols, N.K., Van Dooren, P.: Robust pole assignment in linear state feedback. *International Journal of Control* **41**(5), 1129–1155 (1985) <https://doi.org/10.1080/0020718508961188>
- [37] Blanchini, F.: Set invariance in control. *Automatica* **35**(11), 1747–1767 (1999) [https://doi.org/10.1016/S0005-1098\(99\)00113-2](https://doi.org/10.1016/S0005-1098(99)00113-2)
- [38] Gilbert, E.G., Tan, K.T.: Linear systems with state and control constraints: the theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control* **36**(9), 1008–1020 (1991) <https://doi.org/10.1109/9.83532>
- [39] Arnström, D.: Real-Time Certified MPC : Reliable Active-Set QP Solvers. Linköping Studies in Science and Technology. Dissertations, vol. 2324. Linköping University Electronic Press, Linköping (2023). <https://doi.org/10.3384/9789180752190>
- [40] Ferreau, H.J., Kirches, C., Potschka, A., Bock, H.G., Diehl, M.: qpOASES:

- a parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation* **6**(3), 327–363 (2014) <https://doi.org/10.1007/s12532-014-0071-1>
- [41] Arnstrom, D., Bemporad, A., Axehill, D.: A Dual Active-Set Solver for Embedded Quadratic Programming Using Recursive LDL^T Updates. *IEEE Transactions on Automatic Control* **67**(8), 4362–4369 (2022) <https://doi.org/10.1109/TAC.2022.3176430>
 - [42] Asadi, H., Bellmann, T., Qazani, M.C., Mohamed, S., Lim, C.P., Nahavandi, S.: A Novel Decoupled Model Predictive Control-Based Motion Cueing Algorithm for Driving Simulators. *IEEE Transactions on Vehicular Technology* **72**(6), 7024–7034 (2023) <https://doi.org/10.1109/TVT.2023.3237317>
 - [43] Bruschetta, M., Cenedese, C., Beghi, A., Maran, F.: A motion cueing algorithm with look-ahead and driver characterization: Application to vertical car dynamics. *IEEE Transactions on Human-Machine Systems* **48**(1), 6–16 (2018) <https://doi.org/10.1109/THMS.2017.2776207>
 - [44] Mohammadi, A., Asadi, H., Nelson, K., Nahavandi, S.: Future reference prediction in model predictive control based driving simulators. In: *Australasian Conference on Robotics and Automation (ACRA2016)* (2016)
 - [45] Drop, F.M., Olivari, M., Katliar, M., Bühlhoff, H.H.: Model predictive motion cueing: Online prediction and washout tuning. In: Kemeny, A., Colombet, F., Merienne, F., Espié, S. (eds.) *Proceedings of the Driving Simulation Conference 2018 Europe VR*, Antibes, France, pp. 71–78 (2018). Driving Simulation Association
 - [46] Biemelt, P., Link, C., Gausemeier, S., Trächtler, A.: A Model-Based Online Reference Prediction Strategy for Model Predictive Motion Cueing Algorithms. *IFAC-PapersOnLine* **53**(2), 6082–6088 (2020) <https://doi.org/10.1016/j.ifacol.2020.12.1681>