

Autonomous Vehicle Trajectory Following: A Flatness Model Predictive Control Approach With Hardware-in-the-Loop Verification

Zejiang Wang^{ID}, Jingqiang Zha, and Junmin Wang^{ID}, *Senior Member, IEEE*

Abstract—Trajectory following of autonomous vehicle is a challenging task because of the multiple constraints imposed on the plant. Therefore, Model Predictive Control (MPC) is becoming prevail in vehicle motion control as it can explicitly handle system constraints. However, MPC, grounded in real-time iterative optimization, entails a considerable computational burden for current electronic control units. To mitigate the MPC execution load, a popular strategy is to linearize the original (nonlinear) system around the current working point and then design a Linear Time-Varying MPC (LTV MPC). Nevertheless, the successive linearization introduces extra modeling errors, which may impair the control performance. Indeed, if the plant model satisfies the ‘differential flatness’ condition, it can be *exactly* linearized to the Brunovsky’s canonical form. In contrast to the LTV model, this newly appeared linear form reserves all the nonlinear features of the native plant model. Based on this equivalent linear system, a Flatness Model Predictive Controller (FMPC) can be formulated. FMPC on the one hand, improves the control performance over an LTV MPC because it avoids extra modeling errors from the local linearization. On the other hand, it entails a much lighter computational load versus a nonlinear MPC thanks to its linear nature. Real-time simulations conducted on a hardware-in-the-loop system indicate the advantages of the proposed FMPC in autonomous vehicle trajectory following.

Index Terms—Autonomous vehicle, differential flatness, model predictive control, trajectory following.

I. INTRODUCTION

AS A technique to systematically handle constraints, Model Predictive Control (MPC) has been commonly used for ground vehicle motion control [1], [2]. However, the iterative online optimization of MPC imposes a heavy computational burden on the vehicular embedded computing units. To shorten the MPC execution time and avoid overrun, several methods [3] have been proposed such as the high-efficiency optimization solvers [4], [5], and the ‘moving block’ technique [6]. Further, an explicit MPC [7] was proposed to separate the calculation-extensive optimization from the fast online optimal result search. Among all the existing computation-burden-alleviation strategies, Linear Time-Varying Model Predictive Controller (LTV MPC) arguably constitutes the most

popular option [2], [8]. LTV MPC successively linearizes the original (nonlinear) system around the current working point and converts a nonlinear optimization problem into a Quadratic Programming (QP) problem, which can be solved with QP solvers [9], [10]. However, the local linearization of LTV MPC brings additional approximation errors, which affect its control performance.

In contrast to the local linearization, a differentially flat system [11] can be *exactly* linearized by using the flat output, and the native nonlinear system can be expressed as an equivalent linear one in its Brunovsky’s canonical form. However, it remains hard to include system constraints into the flatness controller design. Existing non-optimization-based methods focus on designing special reference trajectory of the flat output for handling the input constraints [12]. Rather, combining differential flatness with MPC may treat the plant constraints systematically. The resulting Flatness Model Predictive Controller (FMPC) inherits the light computational burden of LTV MPC and maintains the high control performance of nonlinear MPC (NMPC) as no local linearization error is involved.

There exist several control structures in the literature integrating differential flatness with predictive control. A flatness predictive controller was illustrated in [13], where the predicted flat output was used to determine the feedforward control, and a simple feedback controller was included for disturbance and modeling error rejection. The differential flatness property of an aerial vehicle was exploited in [14] to generate the referential state trajectories from the desired flat output. Then, an MPC was utilized for tracking the referential state. In [15] and [16], system inputs and states were firstly parameterized as functions of the flat output and the flat output itself was further expressed as a linear combination of basic functions. Accordingly, the original optimization problem was transformed into a Nonlinear Programming (NLP). However, as indicated in [15], it is indeed difficult to prove that NLP leads to the global optima. Moreover, the computing period of an NLP can be even longer than resolving a QP-based MPC [16]. Similar to this paper, applying MPC on an equivalent linear flat system can be found in [17] and [18]. The nonlinear plant model was first converted into its Brunovsky’s canonical form. Next, an MPC on the basis of this Linear Time-Invariant (LTI) model was established to obtain the optimal flat inputs and states, from which the control of the primary system was decided.

This paper proposes an FMPC for autonomous vehicle trajectory following. A new kinodynamic vehicle model

Manuscript received June 19, 2019; revised October 6, 2019 and January 26, 2020; accepted April 10, 2020. Date of publication April 29, 2020; date of current version September 1, 2021. This work was supported in part by the National Science Foundation under Award 1901632. The Associate Editor for this article was J. W. Choi. (Corresponding author: Junmin Wang.)

The authors are with the Walker Department of Mechanical Engineering, The University of Texas at Austin, Austin, TX 78712 USA (e-mail: jwang@austin.utexas.edu).

Digital Object Identifier 10.1109/TITS.2020.2987987

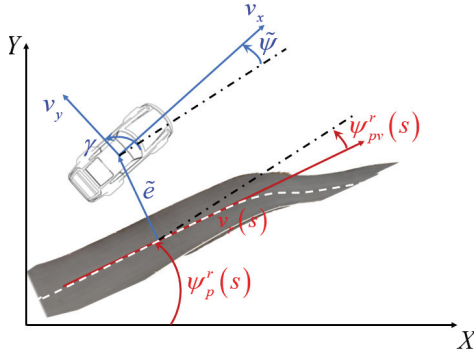


Fig. 1. Kinodynamic trajectory following model schematic diagram.

is developed, and its differential flatness property is proved. Subsequently, a high-level FMPC is designed for calculating the vehicle's global longitudinal force, lateral force, and yaw moment. Afterward, a simple control allocation algorithm is employed to allocate the high-level forces and moment into four wheels' steering and spinning torques. We assume that the autonomous vehicle is both four-wheel-independent driven (4WID) and four-wheel-independent steered (4WIS) with in-wheel motors and steering-by-wheel systems. The spinning torque of each wheel can be individually regulated, which can be positive (driving torque) or negative (braking torque). The advantages of the proposed FMPC over an LTV MPC and an NMPC are shown through Hardware-in-the-Loop (HIL) real-time simulations.

The rest of this paper is organized as follows. The vehicle kinodynamic model for trajectory following is illustrated in Section II. The differential flatness property of this new model is then proved in Section III. Thereafter, the global forces and moment generation from either an LTV MPC, an NMPC, or the FMPC is illustrated in Section IV, where the control allocation module is also mentioned. HIL simulation setup and results are presented in Section V, and Section VI concludes this paper.

II. KINODYNAMIC TRAJECTORY FOLLOWING MODEL

Unifying the linear tracking-error kinematics and the vehicle dynamics in [19], a nonlinear kinodynamic model diagram for ground vehicle trajectory following is shown in Fig. 1.

This model contains five states as:

$$x = [\tilde{e} \quad \tilde{\psi} \quad \tilde{v} \quad v_y \quad \gamma]. \quad (1)$$

In (1), $\tilde{e}$ represents the minimum distance between the vehicle's Center of Gravity (CG) and the centerline of the reference path. $\tilde{\psi}$ indicates the vehicle yaw error with respect to the desired vehicle heading: $\psi_p^r(s) + \psi_{pv}^r(s)$, where $\psi_p^r(s)$ stands for the referent path direction and $\psi_{pv}^r(s)$ accounts for the turning vehicle sideslip amendment [20]. $\tilde{v}$ is the velocity tracking error in regard to the desired tangential speed along the referential path: $v_r(s)$. Finally, v_y specifies the vehicle lateral speed at CG and γ indicates the vehicle yaw rate. External signals $\psi_p^r(s)$, $\psi_{pv}^r(s)$, and $v_r(s)$, are functions of the station s , which is the arc-length traveled by the car along the referential path centerline. We assume that there exists a planning module for generating both the desired trajectory and the corresponding external signals.

Remark 1: Longitudinal velocity v_x , though not included in the states x , is assumed a known variable for computing $\tilde{v}$.

In addition, the system admits three inputs, as:

$$u = [F_x \quad F_y \quad M_z], \quad (2)$$

where F_x and F_y represent the vehicle's global longitudinal and lateral force, and M_z indicates the global yaw moment around CG.

According to Fig. 1, we can directly express the first-order derivative of $\tilde{e}$ as:

$$\dot{\tilde{e}} = v_x \sin(\tilde{\psi} + \psi_{pv}^r(s)) + v_y \cos(\tilde{\psi} + \psi_{pv}^r(s)). \quad (3)$$

Also, vehicle yaw error can be calculated as:

$$\tilde{\psi} = \psi - (\psi_p^r(s) + \psi_{pv}^r(s)). \quad (4)$$

By treating $\psi_{pv}^r(s)$ as the steady-state sideslip angle [21], we obtain:

$$\psi_{pv}^r(s) = -l_r \kappa_r(s) + \frac{m l_f v_r(s)^2 \kappa_r(s)}{2 C_y (l_f + l_r)}. \quad (5)$$

In (5), l_f and l_r respectively represent the distance from vehicle CG to the front and rear axle, m indicates the vehicle mass, C_y symbolizes the cornering stiffness of a single tire, and $\kappa_r(s)$ is the curvature of the desired path. Eq. (5) is derived on the basis of the linear tire model and the small sideslip angle assumption, which holds true for general steady-state turning.

From (4), we have naturally,

$$\dot{\tilde{\psi}} = \gamma - \left(\frac{\partial \psi_p^r(s)}{\partial s} + \frac{\partial \psi_{pv}^r(s)}{\partial s} \right) (\tilde{v} + v_r(s)), \quad (6)$$

as we define:

$$\tilde{v} = \dot{s} - v_r(s). \quad (7)$$

In practice, we can assume $\tilde{e} \ll 1/\kappa_r(s)$. Then, $\dot{s}$ in (7) is approximated as:

$$\dot{s} \approx v_x \cos(\tilde{\psi} + \psi_{pv}^r(s)) - v_y \sin(\tilde{\psi} + \psi_{pv}^r(s)). \quad (8)$$

Based on (7) and (8), the first-order derivative of $\tilde{v}$ can be computed as:

$$\begin{aligned} \dot{\tilde{v}} = & \frac{F_x}{m} \cos(\tilde{\psi} + \psi_{pv}^r(s)) - \frac{F_y}{m} \sin(\tilde{\psi} + \psi_{pv}^r(s)) \\ & + \frac{\partial \psi_p^r(s)}{\partial s} (\tilde{v} + v_r(s)) \dot{\tilde{e}} - \frac{\partial v_r(s)}{\partial s} (\tilde{v} + v_r(s)), \end{aligned} \quad (9)$$

where $\dot{\tilde{e}}$ is expressed in (3).

Finally, the lateral dynamics of the vehicle can be summarized as:

$$\dot{v}_y = F_y/m - v_x \gamma, \quad (10)$$

and

$$\dot{\gamma} = M_z/I_z, \quad (11)$$

where I_z is vehicle yaw inertia.

Equations (3), (6), (9), (10), and (11) constitute the complete kinodynamic model for trajectory following.

III. DIFFERENTIAL FLATNESS PROPERTY PROOF

In this section, we will show that the kinodynamic model in Section II is differentially flat.

A. Differentially Flat System

A dynamic system $\dot{\mathbf{x}} = \mathbf{g}(\mathbf{x}, \mathbf{u})$ with $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$ is differentially flat [11] if it satisfies the following conditions:

- There is a set of differentially independent variables $\boldsymbol{\eta} = (\eta_1 \ \eta_2 \ \dots \ \eta_m)$, which can be expressed as smooth functions: $\eta = p(x, u, \dots, u^{(r)})$, $r \in \mathbb{N}$.
- Both system states $\mathbf{x}$ and inputs $\mathbf{u}$ can be expressed as smooth functions of $\boldsymbol{\eta}$ and its finite-order derivatives. In other words, $\mathbf{x} = \boldsymbol{\Lambda}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \dots, \boldsymbol{\eta}^{(j)})$, $\mathbf{u} = \boldsymbol{\Theta}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \dots, \boldsymbol{\eta}^{(j+1)})$, $j \in \mathbb{N}$.
- The following equations hold: $\frac{d\boldsymbol{\Lambda}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \dots, \boldsymbol{\eta}^{(j)})}{dt} = \mathbf{g}(\boldsymbol{\Lambda}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \dots, \boldsymbol{\eta}^{(j)}), \boldsymbol{\Theta}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \dots, \boldsymbol{\eta}^{(j+1)}))$.

The variable set $\boldsymbol{\eta}$ is called the ‘flat output’, which has the same dimension as the system inputs $\mathbf{u}$.

B. Differential Flatness Property of the Kinodynamic Model

According to the definition of differential flatness, we first need to identify the flat output and then show that all the system inputs and states can be expressed as functions of the flat output and its finite-order derivatives. Here, we prove that the kinodynamic system composed of equations (3), (6), (9), (10), and (11) is differentially flat with the flat output as: $\boldsymbol{\eta} = [\tilde{e}, \tilde{\psi}, \tilde{v}]$.

Proof: First of all, the three elements in $\boldsymbol{\eta}$ are differentially independent, which means there does not exist a non-trivial differential equation of the form $\Phi(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \dots) = \mathbf{0}$. What is more, $\boldsymbol{\eta}$ already includes three elements of x in (1). Therefore, we only need to show that the rest two states v_y, γ and the three inputs F_x, F_y, M_z can be formulated as functions of $\boldsymbol{\eta}$ and its finite-order derivatives.

Rearranging (6) gives the expression of γ as:

$$\gamma = \dot{\tilde{\psi}} + \left(\frac{\partial \psi_p^r(s)}{\partial s} + \frac{\partial \psi_{pv}^r(s)}{\partial s} \right) (\tilde{v} + v_r(s)). \quad (12)$$

By combining (3), (7), and (8), we are able to express the vehicle lateral velocity as:

$$v_y = \dot{\tilde{e}} \cos(\tilde{\psi} + \psi_{pv}^r(s)) - (\tilde{v} + v_r(s)) \sin(\tilde{\psi} + \psi_{pv}^r(s)). \quad (13)$$

Subsequently, the second-order derivative of $\tilde{e}$ and $\tilde{\psi}$ can be calculated as:

$$\begin{aligned} \ddot{\tilde{e}} &= \frac{F_x}{m} \sin(\tilde{\psi} + \psi_{pv}^r(s)) \\ &+ \frac{F_y}{m} \cos(\tilde{\psi} + \psi_{pv}^r(s)) - \frac{\partial \psi_p^r(s)}{\partial s} (\tilde{v} + v_r(s))^2, \end{aligned} \quad (14)$$

and

$$\begin{aligned} \ddot{\tilde{\psi}} &= \frac{M_z}{I_z} - \left(\frac{\partial^2 \psi_p^r(s)}{\partial^2 s} + \frac{\partial^2 \psi_{pv}^r(s)}{\partial^2 s} \right) (\tilde{v} + v_r(s))^2 \\ &- \left(\frac{\partial \psi_p^r(s)}{\partial s} + \frac{\partial \psi_{pv}^r(s)}{\partial s} \right) \left(\dot{\tilde{v}} + \frac{\partial v_r(s)}{\partial s} (\tilde{v} + v_r(s)) \right), \end{aligned} \quad (15)$$

where $\dot{\tilde{v}}$ in (15) is defined in (9).

An in-depth examination of (9), (14), and (15) reveals that the dynamic extension algorithm [22] is satisfied. As a result,

the kinodynamic model is indeed differentially flat. Literally, the parameterization of the system inputs can be shown as:

$$\begin{aligned} \frac{F_x}{m} &= \sin(\tilde{\psi} + \psi_{pv}^r(s)) \left(\ddot{\tilde{e}} + \frac{\partial \psi_p^r(s)}{\partial s} (\tilde{v} + v_r(s))^2 \right) \\ &+ \cos(\tilde{\psi} + \psi_{pv}^r(s)) \left(\dot{\tilde{v}} - \frac{\partial \psi_p^r(s)}{\partial s} (\tilde{v} + v_r(s)) \dot{\tilde{e}} \right. \\ &\quad \left. + \frac{\partial v_r(s)}{\partial s} (\tilde{v} + v_r(s)) \right), \end{aligned} \quad (16)$$

$$\begin{aligned} \frac{F_y}{m} &= \cos(\tilde{\psi} + \psi_{pv}^r(s)) \left(\ddot{\tilde{e}} + \frac{\partial \psi_p^r(s)}{\partial s} (\tilde{v} + v_r(s))^2 \right) \\ &+ \sin(\tilde{\psi} + \psi_{pv}^r(s)) \left(-\dot{\tilde{v}} + \frac{\partial \psi_p^r(s)}{\partial s} (\tilde{v} + v_r(s)) \dot{\tilde{e}} \right. \\ &\quad \left. - \frac{\partial v_r(s)}{\partial s} (\tilde{v} + v_r(s)) \right), \end{aligned} \quad (17)$$

and finally,

$$\begin{aligned} \frac{M_z}{I_z} &= \ddot{\tilde{\psi}} + \left(\frac{\partial^2 \psi_p^r(s)}{\partial^2 s} + \frac{\partial^2 \psi_{pv}^r(s)}{\partial^2 s} \right) (\tilde{v} + v_r(s))^2 \\ &+ \left(\frac{\partial \psi_p^r(s)}{\partial s} + \frac{\partial \psi_{pv}^r(s)}{\partial s} \right) \left(\dot{\tilde{v}} + \frac{\partial v_r(s)}{\partial s} (\tilde{v} + v_r(s)) \right). \end{aligned} \quad (18)$$

As stated in (12), (13), (16), (17), and (18), both the remaining two states: v_y, γ and all the system inputs: F_x, F_y, M_z can be parameterized with $\boldsymbol{\eta}$ and its derivatives up to the second order.

Remark 2: We have proved that the nonlinear kinodynamic model composed by (3), (6), (9), (10), and (11) with inputs $\mathbf{u} = [F_x \ F_y \ M_z]$ and states $\mathbf{x} = [\tilde{e} \ \tilde{\psi} \ \tilde{v} \ v_y \ \gamma]$ is differentially flat with the flat output: $\boldsymbol{\eta} = [\tilde{e}, \tilde{\psi}, \tilde{v}]$. However, the complicated tire-road friction forces, which in general are modeled as nonlinear functions of the tire sideslip angles and tire slip ratios, are not considered in this high-level kinodynamic model. Therefore, even though the external signal $\psi_{pv}^r(s)$ in (5) was derived on the basis of the linear tire model and the small-angle assumption, it does not affect the conclusion that the kinodynamic model is differentially flat. For instance, if we simply assign $\psi_{pv}^r(s) = 0$, the differential flatness property proof still holds.

C. Flatness Property Verification

To validate the proof in Section III-B, we can substitute the Brunovsky’s feedback items:

$$\begin{cases} \dot{\tilde{v}} = \dot{\tilde{v}}_r + K_{v0} (\tilde{v} - \tilde{v}_r), \\ \ddot{\tilde{e}} = \ddot{\tilde{e}}_r + K_{e1} (\dot{\tilde{e}} - \dot{\tilde{e}}_r) + K_{e0} (\tilde{e} - \tilde{e}_r), \\ \ddot{\tilde{\psi}} = \ddot{\tilde{\psi}}_r + K_{\psi1} (\dot{\tilde{\psi}} - \dot{\tilde{\psi}}_r) + K_{\psi0} (\tilde{\psi} - \tilde{\psi}_r), \end{cases} \quad (19)$$

where

$$\dot{\tilde{v}}_r = \dot{\tilde{v}}_r = 0; \quad \ddot{\tilde{e}}_r = \dot{\tilde{e}}_r = \ddot{\tilde{e}}_r = 0; \quad \ddot{\tilde{\psi}}_r = \dot{\tilde{\psi}}_r = \ddot{\tilde{\psi}}_r = 0, \quad (20)$$

back to the parameterized system inputs in (16), (17), and (18), and observe whether the flat output: $\tilde{e}, \tilde{\psi}$, and $\tilde{v}$ can be stabilized. The simulation result is given in Fig. 2. The feedback gains were fixed as: $K_{v0} = -10$, $K_{e1} = K_{\psi1} = -20$, $K_{e0} = K_{\psi0} = -100$.

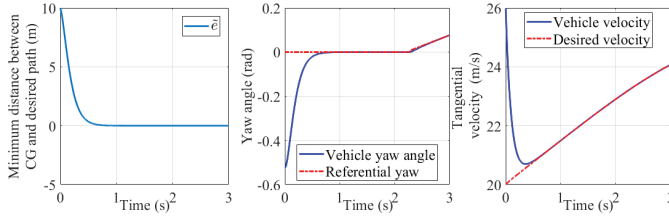


Fig. 2. Flat output stabilization and referential trajectory following.

Clearly, the flat output $\eta = [\tilde{e}, \tilde{\psi}, \tilde{v}]$, which includes the position tracking error $\tilde{e}$, the yaw angle tracking error $\tilde{\psi}$, and the velocity tracking error $\tilde{v}$, quickly converges to zero.

Remark 3: The Brunovsky's feedback control law in (19) does not consider any system constraints.

IV. CONTROLLER DESIGN

As indicated in Section I, the controller has a hierarchical structure. The higher-level MPC is responsible for generating the global forces and moment: $[F_x \ F_y \ M_z]$ for stabilizing the flat output: $\eta = [\tilde{e}, \tilde{\psi}, \tilde{v}]$. Then, a lower-level module is used to allocate the global forces and moment into four wheels' steering and spinning torques.

In addition to the FMPC, an LTV MPC and an NMPC are also conceived for comparison purpose.

A. Linear Time Varying Model Predictive Controller

The differentially flat kinodynamic model in Section II can be compactly written as:

$$\begin{cases} \dot{x} = f(x, u), \\ y = x, \end{cases} \quad (21)$$

where the states x are explained in (1) and the system inputs u are defined in (2).

System constraints for the LTV MPC can be listed as:

$$\begin{cases} x_{k+1,t} = A_t x_{k,t} + B_t u_{k,t} + d_{k,t}, & k = t \dots t + H_p, \\ y_{k,t} = x_{k,t}, & k = t \dots t + H_p, \\ -\max(v_{y_t}) - \zeta_{v_y} \leq v_{y_{k,t}} \leq \max(v_{y_t}) + \zeta_{v_y}, & k = t + 1 \dots t + H_p, \\ -\max(\gamma_t) - \zeta_{\gamma} \leq \gamma_{k,t} \leq \max(\gamma_t) + \zeta_{\gamma}, & k = t + 1 \dots t + H_p. \end{cases} \quad (22)$$

In (22), $x_{t,t}$ indicates the latest measured states whereas $x_{k,t}, k = t + 1 \dots t + H_p$ are the predicted states within the prediction horizon H_p . Besides, $u_{k,t}, k = t \dots t + H_p$ corresponds to the control sequence. To decrease the dimension of the optimized variables, we enforce $u_{t+k,t} = u_{t+H_c-1,t}, k = H_c \dots H_p$, with H_c as the control horizon. In addition, A_t, B_t indicate the discrete Jacobian matrices of the successively linearized system (21). Finally, $d_{k,t}$ represent the linearization residues. We assume that system states can be either measured or estimated.

To ensure vehicle stability, the last two equations in (22) restrict the vehicle lateral velocity v_y and the yaw rate γ within

the prediction horizon. The threshold values: $\max(v_{y_t})$ and $\max(\gamma_t)$ come from [23] and are calculated as:

$$\begin{cases} \max(v_{y_t}) = 0.02 \mu g v_{x_{t,t}}, \\ \max(\gamma_t) = 0.85 \mu g / v_{x_{t,t}}. \end{cases} \quad (23)$$

In (23), μ stands for the tire-road friction coefficient, $v_{x_{t,t}}$ is the current vehicle longitudinal velocity, and $g = 9.81 \text{ m/s}^2$. To guarantee the iterative feasibility, the constraints on v_y and γ are set soft with slack variables $\zeta_{v_y}, \zeta_{\gamma}$.

Then, the minimization problem of the LTV MPC can be formulated as:

$$\begin{aligned} \min_{F_x, F_y, M_z, \zeta_{v_y}, \zeta_{\gamma}} & \frac{1}{2} \sum_{i=1}^{H_p} \left(\|\tilde{e}(k+i|k)\|_{Q_e}^2 + \|\tilde{\psi}(k+i|k)\|_{Q_{\psi}}^2 \right. \\ & \left. + \|\tilde{v}(k+i|k)\|_{Q_v}^2 \right) \\ & + \frac{1}{2} \sum_{i=0}^{H_c-1} \left(\|F_x(k+i|k) - F_x^r(k+i|k)\|_{R_{F_x}}^2 \right) \\ & + \frac{1}{2} \sum_{i=0}^{H_c-1} \left(\|F_y(k+i|k) - F_y^r(k+i|k)\|_{R_{F_y}}^2 \right) \\ & + \frac{1}{2} \sum_{i=0}^{H_c-1} \left(\|M_z(k+i|k) - M_z^r(k+i|k)\|_{R_{M_z}}^2 \right) \\ & + \rho_{v_y} \zeta_{v_y} + \rho_{\gamma} \zeta_{\gamma}. \end{aligned} \quad (24)$$

In (24), the first item accumulates the tracking errors within the prediction horizon H_p . The second, third, and fourth items represent the sum of the discrepancies between the actual system input u and its desired value: $u^r = [F_x^r \ F_y^r \ M_z^r]$ within the control horizon H_c . The last two terms correspond to the soft constraint violation penalties associated with v_y and γ , where $\rho_{v_y}, \rho_{\gamma}$ are the penalty coefficients.

To determine the desired system inputs u^r , the differential flatness property of the kinodynamic model is exploited again. By substituting $\tilde{e} = \dot{\tilde{e}} = \ddot{\tilde{e}} = 0$, $\tilde{\psi} = \dot{\tilde{\psi}} = \ddot{\tilde{\psi}} = 0$, and $\tilde{v} = \dot{\tilde{v}} = 0$ back into (16), (17), and (18), we have:

$$\begin{aligned} F_x^r &= m \cos(\psi_{pv}^r(s)) \left(\frac{\partial v_r(s)}{\partial s} (v_r(s)) \right) \\ &+ m \sin(\psi_{pv}^r(s)) \left(\frac{\partial \psi_p^r(s)}{\partial s} (v_r(s))^2 \right), \end{aligned} \quad (25)$$

$$\begin{aligned} F_y^r &= m \cos(\psi_{pv}^r(s)) \left(\frac{\partial \psi_p^r(s)}{\partial s} (v_r(s))^2 \right) \\ &- m \sin(\psi_{pv}^r(s)) \left(\frac{\partial v_r(s)}{\partial s} (v_r(s)) \right), \end{aligned} \quad (26)$$

and

$$\begin{aligned} M_z^r &= I_z \left(\frac{\partial^2 \psi_p^r(s)}{\partial s^2} + \frac{\partial^2 \psi_{pv}^r(s)}{\partial s^2} \right) (v_r(s))^2 \\ &+ I_z \left(\frac{\partial \psi_p^r(s)}{\partial s} + \frac{\partial \psi_{pv}^r(s)}{\partial s} \right) \left(\frac{\partial v_r(s)}{\partial s} (v_r(s)) \right). \end{aligned} \quad (27)$$

Repeatedly solving the optimization problem (24) under the constraints in (22) gives us the optimal higher-level control signals: $u^* = [F_x^* \ F_y^* \ M_z^*]$.

B. Nonlinear Model Predictive Controller

The sole difference between the LTV MPC and the NMPC is to replace the successively linearized system dynamics in (22) by: $x_{k+1,t} = x_{k,t} + T_s f(x_{k,t}, u_{k,t})$, $k = t \dots t + H_p$, where T_s stands for the sampling period and $f(\cdot)$ is the nonlinear system dynamics in (21). Due to this nonlinear constraint, the optimization problem of NMPC is not quadratic anymore and QP solvers cannot be utilized. Optimization solvers will be detailed in Section V.

C. Flatness Model Predictive Controller

Contrary to the LTV MPC, which locally linearizes the kinodynamic model (21), FMPC *exactly* converts the nonlinear model (21) into an equivalent LTI model, according to which an MPC controller is designed.

Based on (9), (14), and (15), we can assign $\dot{v} \triangleq v_v$, $\ddot{e} \triangleq v_e$, and $\ddot{\psi} \triangleq v_\psi$. Then, the states of the equivalent LTI system can be grouped as:

$$x^{LTI} = \begin{bmatrix} \tilde{e} & \dot{\tilde{e}} & \tilde{\psi} & \dot{\tilde{\psi}} & \tilde{v} \end{bmatrix}. \quad (28)$$

Hence, the state-space expression can be formulated as:

$$\begin{bmatrix} \dot{\tilde{e}} \\ \ddot{\tilde{e}} \\ \dot{\tilde{\psi}} \\ \ddot{\tilde{\psi}} \\ \dot{\tilde{v}} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{e} \\ \dot{\tilde{e}} \\ \tilde{\psi} \\ \dot{\tilde{\psi}} \\ \tilde{v} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_e \\ v_\psi \\ v_v \end{bmatrix}, \quad (29)$$

which can be abbreviately written like:

$$\dot{x}^{LTI} = A^{LTI} x^{LTI} + B^{LTI} v^{LTI}. \quad (30)$$

In (30), we call:

$$v^{LTI} = \begin{bmatrix} v_e & v_\psi & v_v \end{bmatrix}^T, \quad (31)$$

the ‘flat input’ and x^{LTI} the ‘flat state’.

Remark 4: As illustrated in Section III, all the states in (1) and inputs in (2) of the original system (21) can be parameterized with x^{LTI} and v^{LTI} .

Based on the LTI model (30), the system constraints of the FMPC can be clustered together as:

$$\begin{cases} x_{k+1,t}^{LTI} = A_{ds}^{LTI} x_{k,t}^{LTI} + B_{ds}^{LTI} v_{k,t}^{LTI}, & k = t \dots t + H_p, \\ y_{k,t}^{LTI} = x_{k,t}^{LTI}, & k = t \dots t + H_p, \\ -\max(v_{y_l}) - \zeta_{v_y} \leq v_y(x_{k,t}^{LTI}) \leq \max(v_{y_l}) + \zeta_{v_y}, \\ & k = t + 1 \dots t + H_p, \\ -\max(\gamma_l) - \zeta_\gamma \leq \gamma(x_{k,t}^{LTI}) \leq \max(\gamma_l) + \zeta_\gamma, \\ & k = t + 1 \dots t + H_p. \end{cases} \quad (32)$$

In (32), A_{ds}^{LTI} and B_{ds}^{LTI} are the discretized matrices of A^{LTI} and B^{LTI} in (30), which can be approximated as:

$$\begin{cases} A_{ds}^{LTI} = I + T_s A^{LTI}, \\ B_{ds}^{LTI} = T_s B^{LTI}, \end{cases} \quad (33)$$

where I is the identity matrix with the appropriate dimension. Like (22), $x_{t,t}^{LTI}$ in (32) indicates the LTI system’s current states and $x_{k,t}^{LTI}$, $k = t + 1 \dots t + H_p$ are the predicted LTI system states within the prediction horizon H_p . Also, $v_{k,t}^{LTI}$, $k = t \dots t + H_p$ are the control sequence where $v_{t+k,t}^{LTI} = v_{t+H_c-1,t}^{LTI}$, $k = H_c \dots H_p$.

The constraints on the vehicle’s lateral velocity and yaw rate remain as the same in (22). However, for the LTI system (30), v_y and γ are not system states anymore. Instead, they must be parametrized with the new flat states x^{LTI} in (28).

As illustrated in (12) and (13), the predicted vehicle lateral velocity $v_y(x_{k,t}^{LTI})$ and yaw rate $\gamma(x_{k,t}^{LTI})$ in (32) can be separately expressed as:

$$\begin{aligned} v_y(x_{k,t}^{LTI}) &= \dot{\tilde{e}}_{k,t} \cos(\tilde{\psi}_{k,t} + \psi_{pv}^r(s_{k,t})) \\ &\quad - (\tilde{v}_{k,t} + v_r(s_{k,t})) \sin(\tilde{\psi}_{k,t} + \psi_{pv}^r(s_{k,t})), \end{aligned} \quad (34)$$

and

$$\begin{aligned} \gamma(x_{k,t}^{LTI}) &= \dot{\tilde{\psi}}_{k,t} + \left(\frac{\partial \psi_p^r(s_{k,t})}{\partial s} + \frac{\partial \psi_{pv}^r(s_{k,t})}{\partial s} \right) \\ &\quad \times (\tilde{v}_{k,t} + v_r(s_{k,t})). \end{aligned} \quad (35)$$

In (34) and (35), $\psi_{pv}^r(s_{k,t})$, $v_r(s_{k,t})$, $\frac{\partial \psi_p^r(s_{k,t})}{\partial s}$, $\frac{\partial \psi_{pv}^r(s_{k,t})}{\partial s}$ are the predicted external signals along the desired trajectory.

In contrast to (22), the simple box constraints on v_y and γ have been converted into the state-dependent inequalities in (32). Moreover, the parameterized $v_y(x_{k,t}^{LTI})$ becomes a nonlinear function of the predicted flat state $x_{k,t}^{LTI}$. Following the idea of LTV MPC, this newly-appearing nonlinear constraint went through a similar online successive linearization process, as in [24]. Through the first-order Taylor expansion, $v_y(x_{k,t}^{LTI})$ in (34) can be expressed as:

$$\begin{aligned} v_y(x_{k,t}^{LTI}) &= g(\dot{\tilde{e}}_{k,t}, \tilde{\psi}_{k,t}, \tilde{v}_{k,t}, \psi_{pv}^r(s_{k,t}), v_r(s_{k,t})) \\ &\approx g(\dot{\tilde{e}}_{k,t-1}^*, \tilde{\psi}_{k,t-1}^*, \tilde{v}_{k,t-1}^*, \psi_{pv}^r(s_{k,t}), v_r(s_{k,t})) \\ &\quad + \frac{\partial g}{\partial \dot{\tilde{e}}} \bigg|_{\substack{\dot{\tilde{e}}_{k,t-1}^* \\ \tilde{\psi}_{k,t-1}^* \\ \tilde{v}_{k,t-1}^*}} [\dot{\tilde{e}}_{k,t} - \dot{\tilde{e}}_{k,t-1}^*] \\ &\quad + \frac{\partial g}{\partial \tilde{\psi}} \bigg|_{\substack{\dot{\tilde{e}}_{k,t-1}^* \\ \tilde{\psi}_{k,t-1}^* \\ \tilde{v}_{k,t-1}^*}} [\tilde{\psi}_{k,t} - \tilde{\psi}_{k,t-1}^*] \\ &\quad + \frac{\partial g}{\partial \tilde{v}} \bigg|_{\substack{\dot{\tilde{e}}_{k,t-1}^* \\ \tilde{\psi}_{k,t-1}^* \\ \tilde{v}_{k,t-1}^*}} [\tilde{v}_{k,t} - \tilde{v}_{k,t-1}^*], \end{aligned} \quad (36)$$

where the expressions of partial derivatives are:

$$\frac{\partial g}{\partial \dot{\tilde{e}}} \bigg|_{\substack{\dot{\tilde{e}}_{k,t-1}^* \\ \tilde{\psi}_{k,t-1}^* \\ \tilde{v}_{k,t-1}^*}} = \cos(\tilde{\psi}_{k,t-1}^* + \psi_{pv}^r(s_{k,t})),$$

$$\begin{aligned}
\frac{\partial g}{\partial \tilde{v}} \bigg|_{\substack{\dot{\tilde{e}}_{k,t-1}^* \\ \dot{\tilde{\psi}}_{k,t-1}^* \\ \dot{\tilde{v}}_{k,t-1}^*}} &= -\sin(\tilde{\psi}_{k,t-1}^* + \psi_{pv}^r(s_{k,t})), \\
\frac{\partial g}{\partial \tilde{\psi}} \bigg|_{\substack{\dot{\tilde{e}}_{k,t-1}^* \\ \dot{\tilde{\psi}}_{k,t-1}^* \\ \dot{\tilde{v}}_{k,t-1}^*}} &= -\dot{\tilde{e}}_{k,t-1}^* \sin(\tilde{\psi}_{k,t-1}^* + \psi_{pv}^r(s_{k,t})) \\
&\quad - (\tilde{v}_{k,t-1}^* + v_r(s_{k,t})) \cos(\tilde{\psi}_{k,t-1}^* + \psi_{pv}^r(s_{k,t})).
\end{aligned} \tag{37}$$

In (36) and (37), $\dot{\tilde{e}}_{k,t-1}^*$, $\dot{\tilde{\psi}}_{k,t-1}^*$, and $\dot{\tilde{v}}_{k,t-1}^*$ correspond to the optimal predicted flat states obtained at the last step $t-1$.

Note that we did not explicitly impose constraints on the model inputs $u = [F_x \ F_y \ M_z]^T$. Two reasons account for this decision. Firstly, it is hard to determine the upper bounds on the available global forces and moment in real-time [25] and inappropriate thresholds will fundamentally affect the tracking performance. Secondly, the low-level control allocation module will handle excessive high-level signals.

As for the physical constraints: $F_x^2(t) + F_y^2(t) \leq (\mu mg)^2$, we can linearize it again in both the formulations of LTV MPC and FMPC. For the sake of simplicity, we ignore this constraint in our paper. In fact, another approach recently proposed in [12] can potentially treat input constraints: By comparing (16), (17), (18) with (25), (26), (27), we can make the conclusion that the magnitudes of $[F_x \ F_y \ M_z]^T$ shall be close to their desired value: $u^r = [F_x^r \ F_y^r \ M_z^r]^T$, because the tracking errors and their derivatives: $\tilde{\psi}, \dot{\tilde{\psi}}, \ddot{\tilde{\psi}}, \tilde{v}, \dot{\tilde{v}}$ should be anyhow limited in the neighborhood of zero under the efforts of our controller. In this way, by carefully designing the referential values, the peak magnitude of $F_x(t)$, $F_y(t)$, $M_z(t)$ could be well handled. This approach does not require computationally expensive online optimization, and we will study it in the future.

Akin to (24), the minimization problem of the FMPC can be expressed as:

$$\begin{aligned}
\min_{v_e, v_\psi, v_v, \zeta_{vy}, \zeta_\gamma} & \frac{1}{2} \sum_{i=1}^{H_p} \left(\|\tilde{e}(k+i|k)\|_{Q_e}^2 + \|\tilde{\psi}(k+i|k)\|_{Q_\psi}^2 \right. \\
& \left. + \|\tilde{v}(k+i|k)\|_{Q_v}^2 \right) \\
& + \frac{1}{2} \sum_{i=0}^{H_c-1} \left(\|v_e(k+i|k)\|_{R_{v_e}}^2 \right. \\
& \left. + \|v_\psi(k+i|k)\|_{R_{v_\psi}}^2 + \|v_v(k+i|k)\|_{R_{v_v}}^2 \right) \\
& + \rho_{vy} \zeta_{vy} + \rho_\gamma \zeta_\gamma.
\end{aligned} \tag{38}$$

In (38), the discrepancies accumulated within the control horizon H_c are formulated with respect to the flat input v^{LTI} in (31) and its desired value: $v_r^{LTI} = [0 \ 0 \ 0]^T$.

The optimization problem (38) under the constraints in (32) is repeatedly solved online and its solution corresponds to the optimal flat input: $v^{LTI*} = [v_e^* \ v_\psi^* \ v_v^*]^T$. After substituting $\ddot{\tilde{e}} = v_e^*$, $\ddot{\tilde{\psi}} = v_\psi^*$, and $\dot{\tilde{v}} = v_v^*$ back into the

equations (16), (17), and (18), the optimal control signals: $u^* = [F_x^* \ F_y^* \ M_z^*]^T$ in (2) can be determined.

Remark 5: When MPC is implemented online, the terminal cost for improving system stability is commonly omitted [26]. The stability issue is out of the scope of this paper and interested reader is referred to [27], which discusses the stability conditions of the finite-horizon MPC without a terminal cost.

D. Control Allocation

The MPC controller gives us the optimal higher-level control signals: F_x^*, F_y^*, M_z^* . Subsequently, they need to be allocated to the four wheels' spinning torques and steering. A pseudo-inverse allocation architecture from the authors' previous work [28] is used here.

The kinematics relation between the global forces/moment and four wheels' longitudinal and lateral tire forces can be expressed as:

$$[F_x \ F_y \ M_z]^T = B_{sys} u_{F_{x,y}}, \tag{39}$$

where $u_{F_{x,y}} = [F_{xfl}, F_{xrl}, F_{xfr}, F_{xrr}, F_{yfl}, F_{yrl}, F_{yfr}, F_{yrr}]^T$ and the expression of the 3*8 system matrix B_{sys} can be found in [28].

To uniquely determine $u_{F_{x,y}}$, another optimization problem is proposed as:

$$\begin{aligned}
\min_{u_{F_{x,y}}^*} J &= u_{F_{x,y}}^{*T} W u_{F_{x,y}}^* + (B_{sys} u_{F_{x,y}}^* - [F_x, F_y, M_z]^T)^T Q_{sys} \\
&\quad \times (B_{sys} u_{F_{x,y}}^* - [F_x, F_y, M_z]^T).
\end{aligned} \tag{40}$$

In (40), $u_{F_{x,y}}^*$ indicates the optimally allocated longitudinal and lateral tire forces of each wheel. The first term in (40) is used to restrain excessively demanded tire forces whereas the second term aims at restricting allocation errors.

As an 8*8 diagonal matrix, the eight diagonal elements: $w(F_{xfl})$, $w(F_{xfr})$, $w(F_{xrl})$, $w(F_{xrr})$, $w(F_{yfl})$, $w(F_{yfr})$, $w(F_{yrl})$, and $w(F_{yrr})$ in W are chosen as:

$$\begin{cases} w(F_{xij}) = \tan\left(\frac{\pi}{2} \left(1 - \frac{\partial \hat{F}_{xij}}{\partial s_{ij}} / C_x\right)\right), \\ w(F_{yij}) = \tan\left(\frac{\pi}{2} \left(1 - \frac{\partial \hat{F}_{yij}}{\partial \alpha_{ij}} / C_y\right)\right). \end{cases} \tag{41}$$

In (41), $\hat{F}_{xij}$, $\hat{F}_{yij}$ represent the estimated longitudinal and lateral tire forces of each wheel, which were calculated by using the brush tire model in [23]. Besides, s_{ij} indicates tire slip ratio and α_{ij} is tire sideslip angle. Finally, C_x and C_y are the tire longitudinal and cornering stiffness.

Normally, $w(F_{xij})$, $w(F_{yij})$ in (41) are in the order 1e-2 or 1e-3. Therefore, Q_{sys} is chosen as the 3*3 identity matrix for simplicity. However, when an estimated tire force enters into the nonlinear region, the corresponding weighting factor w can sharply increase toward infinity. Hence, the cost function (40) balances the allocation accuracy and the friction force usage.

The optimization problem (40) follows a canonical quadratic form, so its global minimum can be analytically calculated as:

$$u_{F_{x,y}}^* = (B_{sys}^T Q_{sys} B_{sys} + W)^{-1} B_{sys}^T Q_{sys} (u^*)^T, \tag{42}$$

where $u^* = [F_x^*, F_y^*, M_z^*]$ is the optimal command from the high-level MPC controller.

By neglecting the wheel slip dynamics and adopting the small-angle assumption, each wheel's spinning torque T_{ij}^* and steering δ_{ij}^* can be finally determined by the following naive control laws:

$$\begin{cases} T_{ij}^* = R_{ew} F_{xij}^*, \\ \delta_{ij}^* = K_1 \int (F_{yij}^* - \bar{F}_{yij}) dt + K_2 \int \int (F_{yij}^* - \bar{F}_{yij}) dt dt, \end{cases} \quad (43)$$

where R_{ew} is the effective tire radius, $K_{1,2}$ are positive gains, and $\bar{F}_{yij}$ indicates the actual lateral tire force feedback, which can be either measured via special tire force sensors [29], [30] or estimated from tire force estimation algorithms [31], [32].

Remark 6: As this paper emphasizes on the higher-level FMPC, the lower-level allocation algorithm remains indeed plain. Although the dynamic weighting matrix W intends to restrain excessively generated tire forces, we did not rigorously prove its stability when all the four wheels operate near their physical limits. In fact, these emergency situations are not common for a ‘trajectory following’ controller, as the reference path from a high-level planner should indeed be smooth and easy-to-track. Additionally, we did not explicitly include the friction circle constraints in (40). There exist many other allocation methods in the literature, such as [19], [33], and [34], which can treat tire force saturation and coupling issues. Nonetheless, these methods, similar to MPC, involves numerical optimization. Even some of them can be converted into a QP problem, they still entail a much higher computational burden in contrast to the arithmetic pseudo-inverse approach.

V. HARDWARE-IN-THE-LOOP SIMULATIONS

In this Section, both the control performance and the entailed computational load of the FMPC will be compared with the counterparts of LTV MPC and NMPC through real-time HIL simulations.

A. Hardware-in-the-Loop Experiment Setup

In the HIL setup, a high-fidelity vehicle model, dSPACE Automotive Simulation Model (ASM) ran in a Scalexio real-time system. ASM is an industry-proven vehicle simulation software, which includes dynamics models for suspension, steering, brake, tire force, vehicular embedded electric components, propulsion systems, etc.

In the meantime, the higher-level MPC controller was built into a MicroAutoBox. The MPC controller was programmed in Simulink and it received six raw states of the vehicle from ASM, including the longitudinal position X , the lateral position Y , the vehicle yaw angle ψ , the longitudinal velocity v_x , the lateral velocity v_y , and the yaw rate γ .

Combining these six raw states with the desired path and the referential external references: $\psi_p^r(s)$, $\psi_{pv}^r(s)$ and $v_r(s)$, we can then obtain the kinodynamic system states: $\tilde{e}$, $\tilde{\psi}$, and $\tilde{v}$ in (1) as well as the LTI system states: $\tilde{e}$, $\tilde{\psi}$ in (28).

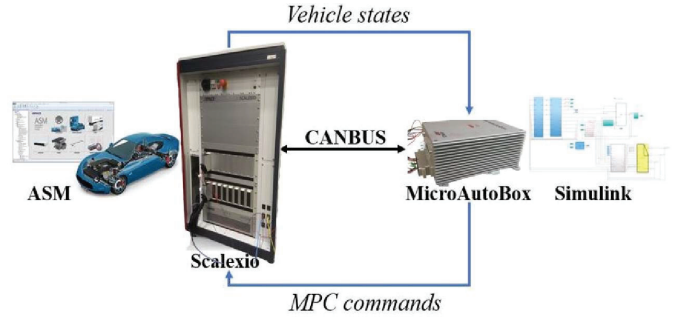


Fig. 3. HIL setup architecture.

The higher-level MPCs periodically computed the optimal global forces and moment: $u^* = [F_x^*, F_y^*, M_z^*]$. Then, these higher-level commands were transferred back to the Scalexio system, where the control allocation algorithm in Section IV-D was executed. Finally, four wheels' spinning torques T_{ij}^* and steering δ_{ij}^* in (43) were fed back to the ASM vehicle model to close the loop. As the computationally-effective pseudo-inverse allocation method in (42) and the actuator-control algorithms in (43) had a very limited influence on the overall computational load (less than 0.1%) during our HIL simulations, we decided to separate the control allocation and actuator-control algorithms from the MicroAutoBox to accurately measure the computational burden associated with the higher-level MPCs.

The bidirectional communications between Scalexio and MicroAutoBox went through the CAN Bus, which is the most widely-used industrial standard for communications between different electronic parts in a vehicle. HIL system architecture is depicted in Fig. 3.

An MPC controller must find the (sub)-optimal solution of a constrained optimization problem before the next sampling moment. Thus, the numerical solver plays a crucial role in delivering high-performance commands in real-time. Both the LTV MPC and the FMPC in Section IV satisfied a quadratic programming framework, therefore the state-of-the-art QP solver: CVXGEN [35] was used. CVXGEN employs an interior-point method, which is robust to ill-conditioned problem and leads to a relatively low algorithm complexity if the dimension of the optimization problem is not too large [35]. In contrast, a projected gradient-based NMPC solver: GRAMPC [26] was employed for solving the NMPC. GRAMPC can outperform the long-tested NMPC solvers, such as ACADO [36] in terms of both the control performance and the execution speed [26].

Since CVXGEN and GRAMPC exploit different internal optimization mechanisms, an absolutely fair comparison seems unfeasible. However, a bunch of parameters and options in GRAMPC can be adjusted to make these two solvers share as many similarities as possible. Precisely speaking, the state constraints handling method in GRAMPC was changed from the default augmented Lagrangian approach to the outer penalty function. In such a manner, the NMPC constraints on v_y and γ became soft. In addition, the convergence threshold for inequality constraints in GRAMPC was set as $1e-4$, which was commensurate to the default settings in CVXGEN.

Finally, the penalty coefficients for the inequality constraints in both GRAMPC and CVXGEN were equally set as $1e6$.

B. Tuning of Model Predictive Controllers

MPC tuning also has a pivotal influence on the control performance and the computational load. Parameters of MPCs in Section IV can be divided into two groups:

- Fundamental parameters: Control horizon H_c , prediction horizon H_p , and sampling period T_s .
- Weighting factors in the cost function: Q_e , Q_ψ , Q_v , R_{F_x} , R_{F_y} , R_{M_z} in (24) of LTV MPC and NMPC and Q_e , Q_ψ , Q_v , R_{v_e} , R_{v_ψ} , R_{v_v} in (38) of FMPC.

As GRAMPC enforces $u(\tau) \equiv u_0(\tau)$, $\tau \in [t, t + H_p T_s]$, we fixed $H_c = 1$. Then, due to the limited flash memory of the MicroAutobox (1st Generation), the prediction horizon was tuned as: $H_p = 3$ to make sure that the local memory can entirely contain the compiled .ppc file. Since the prediction horizon was small, the sampling period T_s was tuned as 180ms to guarantee that the preview time $H_p T_s$ was long enough than the system settling time [23]. The tuned H_p , H_c , and T_s were uniform for LTV MPC, NMPC, and FMPC.

Furthermore, the range-based tuning rule in [23] was employed to decide the weighting factors in the cost functions (24) and (38): Before the authentic HIL simulations, an offline pre-simulation was conducted. During this offline simulation, the desired high-level control signals: $u^* = [F_x^*, F_y^*, M_z^*]$ were firstly generated by use of the Brunovsky's feedback in Section III-C, where the initial errors: $\tilde{e}$, $\tilde{\psi}$, and $\tilde{v}$ were set as zero. After that, the high-level control signals were allocated to four wheels' spinning torques and steering via (42) and (43).

From the offline simulation data, the weighting factors can be determined by:

$$Q_\rho = 1 / \max(|\rho|), \quad R_\sigma = 1 / \max(|\sigma|), \quad (44)$$

where ρ represents e , ψ , and v in both (24) and (38) while σ indicates either F_x , F_y , M_z for the LTV MPC and NMPC in (24) or v_e , v_ψ , v_v for the FMPC in (38). Unfortunately, we found that neither the LTV MPC nor the NMPC produced satisfactory tracking results by strictly following (44). Consequently, the tracking error weights Q_e , Q_ψ , and Q_v of both the LTV MPC and the NMPC were determined by trials and errors (from order of magnitude $1e1$ to $1e5$) with the aim to make the Root Mean Square (RMS) of the tracking errors of the three MPCs (LTV MPC, NMPC, and FMPC) commensurate with each other. The ultimately tuned weighting factors for LTV MPC and NMPC are: $Q_e = 10225$, $Q_\psi = 28846$, $Q_v = 12220$, $R_{F_x} = 3.41e-4$, $R_{F_y} = 2.13e-4$, $R_{M_z} = 4.50e-3$ and for FMPC are: $Q_e = 34.08$, $Q_\psi = 96.15$, $Q_v = 40.73$, $R_{v_e} = 1.46$, $R_{v_\psi} = 9.13$, $R_{v_v} = 4.07$. Indeed, it is debatable that the LTV MPC and NMPC tunings had parameters that were 8 orders of magnitude different. By penalizing the control efforts (R_{F_x} , R_{F_y} , R_{M_z}) in KN/KNm and setting the weights in a comparable range, the control performance of LTV MPC and NMPC could presumably be improved.

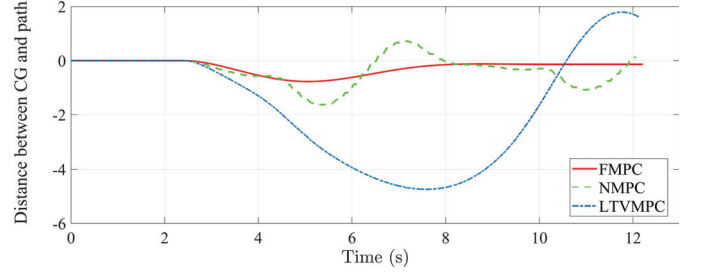


Fig. 4. Minimum distance between CG and the centerline of the reference path.

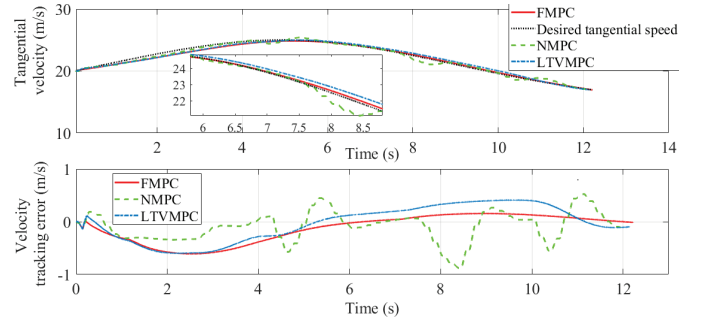


Fig. 5. Velocity tracking error.

Remark 7: The weighting factors of LTV MPC and NMPC by trials and errors were not optimal. Parameter fine-tuning will be studied in the future.

C. Simulation Result and Analysis

A trajectory-following scenario was created to compare the control performance and computational burden of the three MPCs. The reference path was composed of a straight line and an arc with a radius equaling 250m. The desired speed $v_r(s)$ was set as a time-varying sinusoidal function.

To begin with, the minimum distances between the vehicle's CG and the path centerline ($\tilde{e}$) are compared in Fig. 4.

Clearly, FMPC gave us the best tracking whereas LTV MPC resulted in a high $\tilde{e}$ with peak value over 4 meters. Such a huge tracking error from LTV MPC is unusual. The non-optimally tuned weighting factors in LTV MPC partially explained the result. Besides, the short prediction horizon and low control loop frequency are also to blame. In fact, an exactly same LTV MPC [37] with a longer prediction step and a faster update rate can reduce the LTV MPC tracking error to around 1m.

Afterwards, the velocity tracking results and the tracking errors $\tilde{v}$ are depicted in Fig. 5.

All three MPCs can largely track the desired speed $v_r(s)$. Furthermore, LTV MPC and FMPC generated a similar tracking error pattern, but FMPC led to a much smaller $\tilde{v}$ especially after 5s. NMPC produced an oscillating speed tracking error.

The unsatisfactory results from NMPC in Fig. 4 and Fig. 5 were principally boiled down to the short prediction horizon, slow update frequency, and the non-optimally tuned parameters. Virtually, when the GRAMPC were tested offline with $H_p = 10$ and $T_s = 50ms$, the tracking performance of the NMPC was much improved. In addition, the huge tracking error weights led to the oscillating behavior of the NMPC.

Vehicle yaw tracking results are given in Fig. 6.

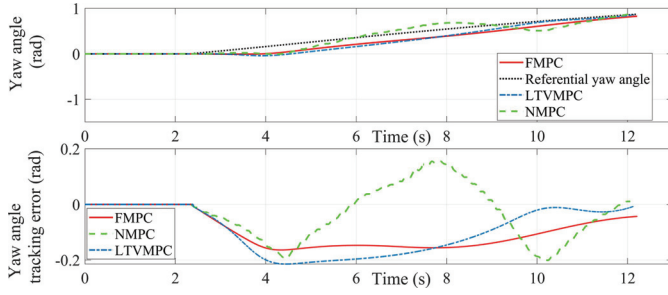


Fig. 6. Vehicle yaw tracking results.

TABLE I
RMS OF TRACKING ERRORS

	$RMS(\tilde{e})$	$RMS(\tilde{\psi})$	$RMS(\tilde{v})$
LTVMPC	2.6925	0.1244	0.3319
NMPC	0.6393	0.0953	0.2938
FMPC	0.3481	0.1141	0.2862

None of the three MPCs could promptly react to the sudden target yaw change (from a straight line to a curve). Actually, the derivative of the target yaw: $\partial\psi_p^r(s)/\partial s$ underwent an abrupt change from 0 ($1/\infty$) to 0.04 ($1/250$). So, the second-order derivative: $\partial^2\psi_p^r(s)/\partial^2s$ in (18) and (27) becomes infinity at the transition point. However, due to the relatively long sampling period T_s and the small prediction horizon H_p , it is clear that the information of this critical point was not included in the three MPCs. On the one hand, this avoided the drastic change of M_z^* at the transition point [37]. On the other hand, it indeed affected the target yaw tracking performance. By the use of a clothoid with a linearly-varying curvature, the transition from the straight line to the curve will be smoothed and the unsatisfactory yaw tracking shall be substantially improved. This strategy will be studied in the future.

The RMS of tracking errors are summarized in Table I.

Hence, the tracking performances of the FMPC are much improved in contrast to the LTVMPC. Because, firstly, the exact linearized LTI model in (29) conserved all the nonlinear feature of the kinodynamic model (21). Secondly, the parameterized system inputs (16), (17), and (18) indeed contained the open-loop control terms, which substantially alleviated the online feedback control efforts. This second reason became especially important as the prediction horizon was short. A similar tracking performance enhancements of FMPC over an LTVMPC were also witnessed in [37] when both the FMPC and the LTVMPC enjoyed a sufficiently long prediction horizon and a fast enough sampling rate. In contrast, we do not assert that the FMPC can over-perform the NMPC in terms of the tracking performance, because the NMPC also conserved all the nonlinear features of the kinodynamic model (21). Indeed, there did not exist obvious tracking performance discrepancies between the FMPC and the (detuned) NMPC in Table I. Instead, as will be revealed in Fig. 10, the fundamental advantage of FMPC over NMPC is in principle the reduced computational load.

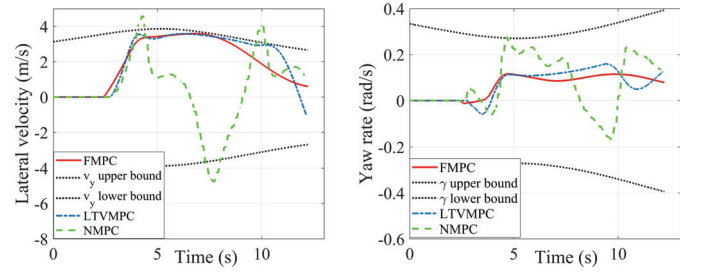


Fig. 7. Lateral velocity and yaw rate constraints.

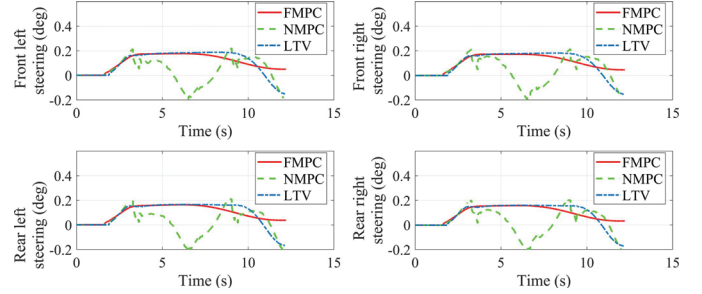


Fig. 8. Allocated four wheel's steering.

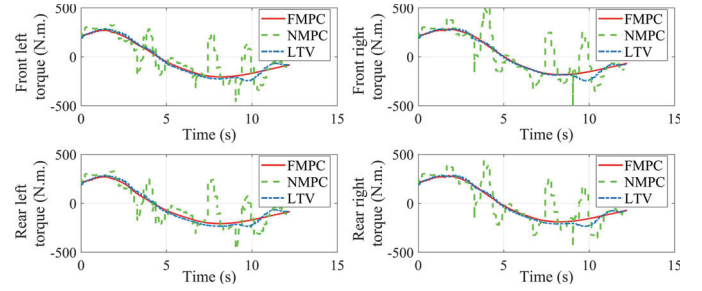


Fig. 9. Allocated four wheel's torque.

Next, the lateral velocity and yaw rate are depicted in Fig. 7.

Obviously, both the LTVMPC and the FMPC obeyed the constraints on v_y and γ . On the contrary, the NMPC violated the constraint on v_y around 4s, 8s, and 10s. This was principally because the state constraints handling method in GRAMPC was changed from the default augmented Lagrangian approach to the outer penalty function, which disabled the online update of the Lagrangian multiplier as well as the penalty coefficients.

Afterwards, we will compare the allocated four wheels' steering and spinning torques. The steering angles of each wheel are demonstrated in Fig. 8 while the torques are depicted in Fig. 9.

In Fig. 8, we can recognize that the pattern of four wheels' steering from each MPC was very similar to its corresponding lateral velocity in Fig. 7. Particularly, NMPC produced a counter-steering behavior from 6s to 8s. This was because, during that period, NMPC simultaneously entailed a large positive position tracking error $\tilde{e}$ (Fig. 4) and yaw error $\tilde{\psi}$ (Fig. 6). To counteract these errors, GRAMPC resulted in a significantly negative v_y , as demonstrated in Fig. 7. To be compatible with such a negative v_y , all four wheels' steering counter-steered.

In Fig. 9, the FMPC and the LTVMPC produced similar four-wheel spinning torques. However, the allocated torques

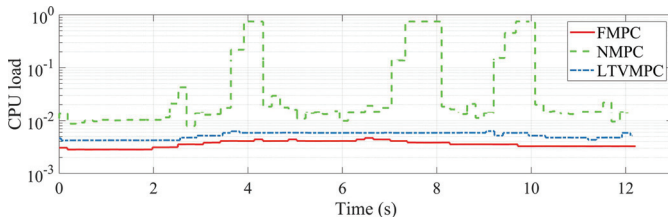


Fig. 10. Computational burden comparisons.

from NMPC suffered from strong oscillations, which directly affected the velocity tracking result in Fig. 5. As mentioned before, this was presumably because of the excessive tracking error weights, the short prediction horizon, and the detuned settings (for the fair comparison between the two solvers) of GRAMPC.

Finally, we define the computational load as [23]:

$$\lambda_{CPU} = TAT/T_s, \quad (45)$$

where TAT represents the controller turnaround time inside the MicroAutoBox, and $T_s = 180ms$ indicates the uniformed sampling period. Intuitively, real-time execution necessitates $\lambda_{CPU} \leq 1$ to avoid overrun. Computational loads are plotted in the semi-logarithmic graph in Fig. 10.

Two conclusions can be drawn from Fig. 10. Firstly, FMPC required the least computational resource, as the peak value of computational load from FMPC was merely 4.8%, which was less than LTV MPC (6.3%) and far less than NMPC (75%). As the current embedded vehicular computing unit has a quite limited computational capacity, this slight execution burden of FMPC can definitely facilitate its online implementation. Secondly, both the computational burdens of FMPC and LTV MPC remained stable without obvious fluctuations. Instead, NMPC incurred a severely changing computational load, especially when the soft constraint on the lateral velocity was violated around 4s, 8s, and 10s. At these moments, the NMPC solver would execute more iterative calculations within one sampling step trying to find a suboptimal solution. These extra-executed iterations caused the sharply increased computational load of NMPC.

In fact, the computational advantage of FMPC was also demonstrated in [38], where system parameterization via differential flatness effectively reduced the dimension of the optimization problem. However, this was not our case. Instead, the reasons why the FMPC incurred less computational load than the LTV MPC can be summarized into two points. Firstly, the LTI model did not require online successive linearization and matrix elements update. Secondly, in contrast to the dense Jacobian matrices A_t, B_t in (22), the LTI system matrices $A_{ds}^{LTI}, B_{ds}^{LTI}$ in (32) were sparse. This sparse data structure was fully exploited by CVXGEN to speed up execution.

VI. CONCLUSION

This paper proposed a flatness model predictive controller for autonomous vehicle trajectory following. Grounded in a newly developed differentially flat kinodynamic model (21), the original nonlinear plant was exactly linearized and an equivalent LTI system (29) appeared. This LTI system maintained however all the nonlinear features of the original

kinodynamic model. Then, an MPC was applied to this newly appeared LTI system for determining the optimal flat input. System parameterization subsequently converted the optimized flat input into the high-level control forces and moment, which were ultimately allocated to the four wheels' spinning torques and steering. HIL simulation demonstrated the advantages of the proposed FMPC over an LTV MPC in terms of the tracking performance, and an NMPC in terms of the online computational burden.

Future work will concentrate on FMPC optimal parameter tuning and stability analysis under disturbances.

REFERENCES

- [1] G. V. Raffo, G. K. Gomes, J. E. Norney-Rico, C. R. Kelber, and L. B. Becker, "A predictive controller for autonomous vehicle path tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 1, pp. 92–102, Mar. 2009, doi: [10.1109/TITS.2008.2011697](https://doi.org/10.1109/TITS.2008.2011697).
- [2] B. Gütjahr, L. Groll, and M. Werling, "Lateral vehicle trajectory optimization using constrained linear time-varying MPC," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1586–1595, Oct. 2016, doi: [10.1109/TITS.2016.2614705](https://doi.org/10.1109/TITS.2016.2614705).
- [3] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 2, pp. 267–278, Mar. 2010, doi: [10.1109/TCST.2009.2017934](https://doi.org/10.1109/TCST.2009.2017934).
- [4] M. Kogel and R. Findeisen, "Fast predictive control of linear systems combining Nesterov's gradient method and the method of multipliers," in *Proc. IEEE Conf. Decis. Control Eur. Control Conf.*, Dec. 2011, pp. 501–506.
- [5] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl, "From linear to nonlinear MPC: Bridging the gap via the real-time iteration," *Int. J. Control*, vol. 93, no. 1, pp. 62–80, Jan. 2020, doi: [10.1080/00207179.2016.1222553](https://doi.org/10.1080/00207179.2016.1222553).
- [6] S. E. Li, Z. Jia, K. Li, and B. Cheng, "Fast online computation of a model predictive controller and its application to fuel economy-oriented adaptive cruise control," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1199–1209, Jun. 2015, doi: [10.1109/TITS.2014.2354052](https://doi.org/10.1109/TITS.2014.2354052).
- [7] P. Tøndel, T. A. Johansen, and A. Bemporad, "An algorithm for multi-parametric quadratic programming and explicit MPC solutions," *Automatica*, vol. 39, no. 3, pp. 489–497, Mar. 2003, doi: [10.1016/S0005-1098\(02\)00250-9](https://doi.org/10.1016/S0005-1098(02)00250-9).
- [8] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat, "Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation," *Int. J. Robust Nonlinear Control*, vol. 18, no. 8, pp. 862–875, May 2008, doi: [10.1002/rnc.1245](https://doi.org/10.1002/rnc.1245).
- [9] A. Bemporad, "A quadratic programming algorithm based on nonnegative least squares with applications to embedded model predictive control," *IEEE Trans. Autom. Control*, vol. 61, no. 4, pp. 1111–1116, Apr. 2016, doi: [10.1109/TAC.2015.2459211](https://doi.org/10.1109/TAC.2015.2459211).
- [10] A. Bemporad, "A multiparametric quadratic programming algorithm with polyhedral computations based on nonnegative least squares," *IEEE Trans. Autom. Control*, vol. 60, no. 11, pp. 2892–2903, Nov. 2015, doi: [10.1109/TAC.2015.2417851](https://doi.org/10.1109/TAC.2015.2417851).
- [11] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: Introductory theory and examples," *Int. J. Control*, vol. 61, no. 6, pp. 1327–1361, Jun. 1995, doi: [10.1080/00207179508921959](https://doi.org/10.1080/00207179508921959).
- [12] H. Mounier, S.-I. Niculescu, A. Cela, and M. Stefan Geamanu, "Flatness-based longitudinal vehicle control with embedded torque constraint," *IMA J. Math. Control Inf.*, vol. 36, no. 3, pp. 729–744, Sep. 2019, doi: [10.1093/imamci/dny005](https://doi.org/10.1093/imamci/dny005).
- [13] M. Fliess and R. Marquez, "Continuous-time linear predictive control and flatness: A module-theoretic setting with examples," *Int. J. Control*, vol. 73, no. 7, pp. 606–623, Jan. 2000, doi: [10.1080/002071700219452](https://doi.org/10.1080/002071700219452).
- [14] I. Prodan, S. Olaru, R. Bencatel, J. Borges de Sousa, C. Stoica, and S.-I. Niculescu, "Receding horizon flight control for trajectory tracking of autonomous aerial vehicles," *Control Eng. Pract.*, vol. 21, no. 10, pp. 1334–1349, Oct. 2013, doi: [10.1016/j.conengprac.2013.05.010](https://doi.org/10.1016/j.conengprac.2013.05.010).
- [15] J. A. De Doná, F. Suryawan, M. M. Seron, and J. Lévine, "A flatness based iterative method for reference trajectory generation in constrained NMPC," in *Nonlinear Model Predictive Control* (Lecture Notes in Control and Information Sciences), vol. 384. Berlin, Germany: Springer, 2009, pp. 325–333, doi: [10.1007/978-3-642-01094-1_27](https://doi.org/10.1007/978-3-642-01094-1_27).

- [16] M. Joševski and D. Abel, "Flatness-based model predictive control for the fuel optimization of hybrid electric vehicles," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 464–470, Sep. 2015, doi: [10.1016/j.ifacol.2015.11.322](https://doi.org/10.1016/j.ifacol.2015.11.322).
- [17] G. Allibert, E. Courtial, and Y. Touré, "A flat model predictive controller for trajectory tracking in image based visual servoing," *IFAC Proc. Volumes*, vol. 40, no. 12, pp. 993–998, 2007, doi: [10.3182/20070822-3-ZA-2920.00165](https://doi.org/10.3182/20070822-3-ZA-2920.00165).
- [18] M. Greeff and A. P. Schoellig, "Flatness-based model predictive control for quadrotor trajectory tracking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 6740–6745.
- [19] H. Park and J. C. Gerdes, "Optimal tire force allocation for trajectory tracking with an over-actuated vehicle," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2015, pp. 1032–1037.
- [20] C. Hu, R. Wang, F. Yan, and N. Chen, "Should the desired heading in path following of autonomous vehicles be the tangent direction of the desired path?" *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3084–3094, Dec. 2015, doi: [10.1109/TITS.2015.2435016](https://doi.org/10.1109/TITS.2015.2435016).
- [21] R. Rajamani, "Lateral vehicle dynamics and steering control for automated lane keeping," in *Vehicle Dynamics and Control* (Mechanical Engineering Series). Boston, MA, USA: Springer, 2011, ch. 2, pp. 15–54, doi: [10.1007/978-1-4614-1433-9_2](https://doi.org/10.1007/978-1-4614-1433-9_2).
- [22] H. Nijmeijer and W. Respondek, "Dynamic input-output decoupling of nonlinear control systems," *IEEE Trans. Autom. Control*, vol. 33, no. 11, pp. 1065–1070, Nov. 1988, doi: [10.1109/9.14420](https://doi.org/10.1109/9.14420).
- [23] Z. Wang, Y. Bai, J. Wang, and X. Wang, "Vehicle path-tracking linear-time-varying model predictive control controller parameter selection considering central process unit computational load," *J. Dyn. Syst., Meas., Control*, vol. 141, no. 5, May 2019, Art. no. 051004, doi: [10.1115/1.4042196](https://doi.org/10.1115/1.4042196).
- [24] W. R. van Soest, Q. P. Chu, and J. A. Mulder, "Combined feedback linearization and constrained model predictive control for entry flight," *J. Guid., Control, Dyn.*, vol. 29, no. 2, pp. 427–434, Mar. 2006, doi: [10.2514/1.14511](https://doi.org/10.2514/1.14511).
- [25] M. Jonasson, J. Andreasson, B. Jacobson, and A. S. Trigell, "Global force potential of over-actuated vehicles," *Vehicle Syst. Dyn.*, vol. 48, no. 9, pp. 983–998, Sep. 2010, doi: [10.1080/00423110903243232](https://doi.org/10.1080/00423110903243232).
- [26] T. Englert, A. Völz, F. Mesmer, S. Rhein, and K. Graichen, "A software framework for embedded nonlinear model predictive control using a gradient-based augmented lagrangian approach (GRAMPC)," *Optim. Eng.*, vol. 20, no. 3, pp. 769–809, Sep. 2019, doi: [10.1007/s11081-018-9417-2](https://doi.org/10.1007/s11081-018-9417-2).
- [27] W.-H. Chen, "Stability analysis of classic finite horizon model predictive control," *Int. J. Control, Autom. Syst.*, vol. 8, no. 2, pp. 187–197, Apr. 2010, doi: [10.1007/s12555-010-0202-z](https://doi.org/10.1007/s12555-010-0202-z).
- [28] R. Wang, C. Hu, Z. Wang, F. Yan, and N. Chen, "Integrated optimal dynamics control of 4WD4WS electric ground vehicle with tire-road frictional coefficient estimation," *Mech. Syst. Signal Process.*, vols. 60–61, pp. 727–741, Aug. 2015, doi: [10.1016/j.ymssp.2014.12.026](https://doi.org/10.1016/j.ymssp.2014.12.026).
- [29] K. Nam, H. Fujimoto, and Y. Hori, "Lateral stability control of in-wheel-motor-driven electric vehicles based on sideslip angle estimation using lateral tire force sensors," *IEEE Trans. Veh. Technol.*, vol. 61, no. 5, pp. 1972–1985, Jun. 2012, doi: [10.1109/TVT.2012.2191627](https://doi.org/10.1109/TVT.2012.2191627).
- [30] S. Kerst, B. Shyrokau, and E. Holweg, "A model-based approach for the estimation of bearing forces and moments using outer ring deformation," *IEEE Trans. Ind. Electron.*, vol. 67, no. 1, pp. 461–470, Jan. 2020, doi: [10.1109/TIE.2019.2897510](https://doi.org/10.1109/TIE.2019.2897510).
- [31] A. Rezaeian *et al.*, "Novel tire force estimation strategy for real-time implementation on vehicle applications," *IEEE Trans. Veh. Technol.*, vol. 64, no. 6, pp. 2231–2241, Jun. 2015, doi: [10.1109/TVT.2014.2345695](https://doi.org/10.1109/TVT.2014.2345695).
- [32] M. Doumiati, A. C. Victorino, A. Charara, and D. Lechner, "Onboard real-time estimation of vehicle lateral tire-road forces and sideslip angle," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 4, pp. 601–614, Aug. 2011, doi: [10.1109/TMECH.2010.2048118](https://doi.org/10.1109/TMECH.2010.2048118).
- [33] J. Wang and R. G. Longoria, "Coordinated and reconfigurable vehicle dynamics control," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 3, pp. 723–732, May 2009, doi: [10.1109/TCST.2008.2002264](https://doi.org/10.1109/TCST.2008.2002264).
- [34] Y. Chen and J. Wang, "Fast and global optimal energy-efficient control allocation with applications to over-actuated electric ground vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 5, pp. 1202–1211, Sep. 2012, doi: [10.1109/TCST.2011.2161989](https://doi.org/10.1109/TCST.2011.2161989).
- [35] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optim. Eng.*, vol. 13, no. 1, pp. 1–27, Mar. 2012, doi: [10.1007/s11081-011-9176-9](https://doi.org/10.1007/s11081-011-9176-9).
- [36] B. Houska, H. J. Ferreau, and M. Diehl, "ACADO toolkit—An open-source framework for automatic control and dynamic optimization," *Optim. Control Appl. Methods*, vol. 32, no. 3, pp. 298–312, May 2011, doi: [10.1002/oca.939](https://doi.org/10.1002/oca.939).
- [37] Z. Wang, J. Zha, and J. Wang, "Flatness-based model predictive control for autonomous vehicle trajectory tracking," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 4146–4151.
- [38] J. Liu, G. Li, and H. K. Fathy, "An extended differential flatness approach for the health-conscious nonlinear model predictive control of lithium-ion batteries," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 5, pp. 1882–1889, Sep. 2017, doi: [10.1109/TCST.2016.2624143](https://doi.org/10.1109/TCST.2016.2624143).



Zejiang Wang received the B.E. degree (Hons.) in mechanical engineering and automation from Southeast University, Nanjing, China, in 2014, the Dipl.Ing. degree from ENSTA ParisTech, Palaiseau, France, in 2017, and the M.S. degree in design, modeling, and architecture of complex industrial systems from École Polytechnique, University of Paris-Saclay, Palaiseau, in 2017. He is currently pursuing the Ph.D. degree with the Walker Department of Mechanical Engineering, The University of Texas at Austin, Austin, TX, USA.

His research interests include vehicle dynamics and control, combustion engine control, and cyber-physical system.



Jingqiang Zha received the B.E. degree in mechanical engineering and automation from Shanghai Jiao-tong University, Shanghai, China, in 2013, and the M.S. degree in mechanical engineering from The University of Texas at Austin, Austin, TX, USA, in 2015, where he is currently pursuing the Ph.D. degree with the Walker Department of Mechanical Engineering.

His research interests include vehicle dynamics and control, robust control, and LPV control.



Junmin Wang (Senior Member, IEEE) received the B.E. degree in automotive engineering and the M.S. degree in power machinery and engineering from Tsinghua University, Beijing, China, in 1997 and 2000, respectively, the M.S. degrees in electrical engineering and mechanical engineering from the University of Minnesota, Twin Cities, Minneapolis, MN, USA, in 2003, and the Ph.D. degree in mechanical engineering from The University of Texas at Austin, Austin, TX, USA, in 2007.

He is an Accenture Endowed Professor with the Walker Department of Mechanical Engineering, The University of Texas at Austin. In 2008, he started his academic career at Ohio State University, where he founded the Vehicle Systems and Control Laboratory, was early promoted to Associate Professor in September 2013 and then very early promoted to Full Professor in June 2016. He also gained five years of full-time industrial research experience at Southwest Research Institute (San Antonio, TX, USA) from 2003 to 2008. He is the author or coauthor of more than 320 peer-reviewed publications, including 165 journal articles and 13 U.S. patents. He is an IEEE Vehicular Technology Society Distinguished Lecturer, SAE fellow, and ASME fellow. He was a recipient of numerous international and national honors and awards.