H4MPC: A Hybridization Toolbox for Model Predictive Control in Automated Driving

Gharavi, L.; De Schutter, B.H.K.; Baldi, S.

DOI
10.1109/AMC58169.2024.10505665

Publication date
2024

Document Version
Final published version

Published in
Proceedings of the IEEE 18th International Conference on Advanced Motion Control (AMC 2024)

2024 IEEE 18th International Conference on Advanced Motion Control (AMC) | 979-8-3503-4281-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/AMC58169.2024.10505665

H4MPC: A Hybridization Toolbox for Model Predictive Control in Automated Driving

Leila Gharavi
Delft Center for Systems and Control
Delft University of Technology
Delft, The Netherlands
L.Gharavi@tudelft.nl

Bart De Schutter
Delft Center for Systems and Control
Delft University of Technology
Delft, The Netherlands
B.DeSchutter@tudelft.nl

Simone Baldi
School of Mathematics
Southeast University
Nanjing, China
103009004@seu.edu.cn

***Abstract*—The computational complexity of nonlinear Model Predictive Control (MPC) poses a significant challenge in achieving real-time levels of 4 and 5 of automated driving. This work presents the open-access Hybridization toolbox for MPC (H4MPC), targeting computational efficiency of nonlinear MPC thanks to several modules to hybridize nonlinear MPC optimization problems commonly encountered in automated driving applications. H4MPC is designed as a user-friendly solution with a graphical user interface within the MATLAB environment. The toolbox facilitates intuitive and straightforward customization of the hybridization process for any given function appearing in the equality or inequality constraints within the MPC framework. The initial release, Version 1.0, is freely available from https://bit.ly/H4MPCV1. To provide a clear illustration of the toolbox capabilities, we present two case studies: one to hybridize a vehicle model and another one to approximate tire saturation constraints.**

***Index Terms*—Hybrid systems, Function approximation, MATLAB toolbox, Model predictive control, Automated driving**

I. Introduction

Nonlinearity of the MPC optimization problem in automated driving is a significant obstacle towards real-time vehicle control [1]. Approximating the nonlinearities is often done in many applications [2] to come up with improved computational efficiency in solving the nonlinear control optimization problem. In this line, hybridization techniques [3] approximate a nonlinear function using hybrid systems formalism, with both continuous and discrete-time dynamics involved in the approximation [4]. For more information on hybrid systems, the reader is referred to [5], [6].

Hybridization has been extensively employed in automated driving research, e.g., in vehicle control [7] by approximating the nonlinear model using a mixed-logical dynamics [8], or by approximating nonlinear tire forces using PieceWise-Affine (PWA) dynamics [9]–[11]. Efficiency of MPC after hybridization has been recently studied in [12], [13].

This work presents H4MPC [14], a hybridization toolbox in MATLAB that provides a user-friendly interface to formulate and solve optimization problems to approximate the nonlinearities in nonlinear MPC using hybrid systems formalism, in particular PWA modeling framework. The toolbox exploits the formulation from Max-Min-Plus-Scaling (MMPS) systems [15] to allow for an intuitive adjustment of the complexity level in the approximated form. Further, H4MPC facilitates approximation of the nonlinear constraints via covering the resulting non-convex feasible region by a union of convex subregions, namely ellipsoids or polytopes, where the latter are obtained using MMPS formalism as well.

This paper describes the H4MPC modules in detail and demonstrates its capabilities using two case studies: approximating a single-track vehicle model [16], and hybridizing the non-convex feasible region due to tire saturation limits, known as the Kamm circle constraint [17] for a Pacejka tire model [18]. The paper is structured as follows: Section II covers the preliminary definitions, Section III presents the architecture of H4MPC and Section IV illustrates the case studies and analysis of the results. Finally, Section V summarizes the results of this work.

II. Preliminaries

A. Nonlinear Problem Description

Consider a given discrete-time nonlinear system $s(k+1) = F(s(k),\, u(k))$ where $s \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ respectively represent the state and input vectors, and the domain of F is denoted by $\mathscr{D} \subseteq \mathbb{R}^{n+m}$. With the state and input vectors defined over the whole prediction horizon N_p as

$$\tilde{s}(k+1) = \begin{bmatrix} \hat{s}^T(k+1|k) & \hat{s}^T(k+2|k) & \dots & \hat{s}^T(k+N_\text{p}|k) \end{bmatrix}^T,$$

$$\tilde{u}(k) = \begin{bmatrix} u^T(k) & u^T(k+1) & \dots & u^T(k+N_\text{p}-1) \end{bmatrix}^T,$$

and $\hat{s}^T(k+i|k)$ with $i \in \{1,\dots,N_\text{p}\}$ representing the prediction of the states in step $k+i$ given the measured states at step k, the nonlinear MPC problem at step k is formulated in the general form:

$$\min_{\tilde{u}(k)} \quad \|\Theta_s \tilde{s}(k)\|_\rho + \|\Theta_u \tilde{u}(k)\|_\rho \tag{1a}$$

$$\text{s.t.} \quad \hat{s}(k+i|k) = F(\hat{s}(k+i-1|k), u(k+i-1)), \quad \forall i \in \{1,\dots,N_\text{p}\}, \tag{1b}$$

$$G(\hat{s}(k+i-1|k), u(k+i-1)) \leqslant 1, \quad \forall i \in \{1,\dots,N_\text{p}\}, \tag{1c}$$

This research is funded by the Dutch Science Foundation NWO-TTW within the EVOLVE project (no. 18484).

where (1b) represents the equality constraints due to the prediction model, and (1c) expresses the non-convex feasible region via the normalized nonlinear constraint function G resulting from physics-based constraints such as tire saturation or vehicle stability. Without loss of generality, we assume G to be a scalar function. The objective function (1a) is a sum of the $\rho-$norm of the state and input vectors with $\rho \in \{1, 2, \infty\}$, induced by weight matrices Θ_s and Θ_u.

B. Approximation of the Nonlinear Problem

Hybridization of the nonlinear MPC problem is done in two steps: (1) approximating the prediction model, i.e. F, and (2) approximating the nonlinear constraints, i.e. G. For a more compact representation, we use the augmented state vector $x = \begin{bmatrix} s^T & u^T \end{bmatrix}$ to define $F(x) := F(s,u)$ and $G(x) := G(s,u)$.

1) Model Approximation: We approximate each component F_w of $F = \begin{bmatrix} F_1 & \dots & F_n \end{bmatrix}^T$ by an MMPS function f_w in the Kripfganz form [19] as

$$\begin{aligned} f_w(x) = \max &\left\{ \phi^+_{w,1}(x), \phi^+_{w,2}(x), \dots, \phi^+_{w,P^+_w}(x) \right\} \\ - \max &\left\{ \phi^-_{w,1}(x), \phi^-_{w,2}(x), \dots, \phi^-_{w,P^-_w}(x) \right\}, \\ &\forall w \in \{1, \dots, n\}, \end{aligned} \tag{2}$$

where the vectors $\phi^\eta_s : \mathbb{R}^{m+n} \to \mathbb{R}^{P^\eta}$ with $\eta \in \{+,-\}$ are affine functions of x, also referred to as dynamic modes. Figure 1 shows an illustrative example for a 1-dimensional case with $(P^+, P^-) = (3, 4)$.

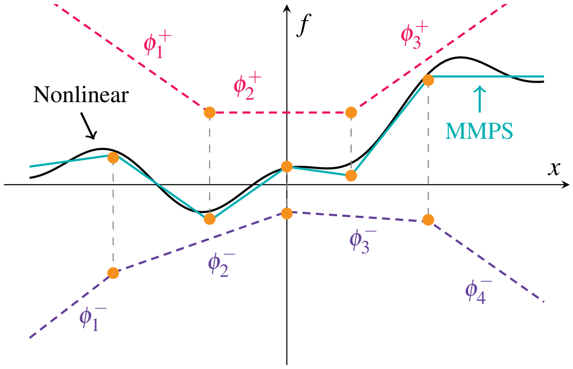


Fig. 1: MMPS approximation of a nonlinear function using the difference of two max functions

2) Constraint Approximation: With the nonlinear, non-convex constraints given as $G(x) \leqslant 1$, we approximate the feasible region $\mathscr{C} := \{x \in \mathscr{D} \mid G(x) \leqslant 1\}$ by a union of convex subregions $\mathscr{R}$. The shape of the subregions in $\mathscr{R}$ can either be polytopic, which we obtain by an MMPS approximation of the boundary, or ellipsoidal.

Similar to the prediction model, MMPS approximation of the constraints is expressed by

$$\begin{aligned} g_{\text{MMPS}}(x) = \max &\left\{ \gamma^+_1(x), \gamma^+_2(x), \dots, \gamma^+_{R^+}(x) \right\} \\ - \max &\left\{ \gamma^-_1(x), \gamma^-_2(x), \dots, \gamma^-_{R^-}(x) \right\}, \end{aligned} \tag{3}$$

with the vectors $\gamma^\eta : \mathbb{R}^{m+n} \to \mathbb{R}^{R^\eta}$ and $\eta \in \{+,-\}$ being affine functions of x.

In the ellipsoidal approach, G is approximated by g_{ELLP} as

$$g_{\text{ELLP}}(x) = \min_{e \in \{1,\dots,n_e\}} \left\{ (x - x_{e,0})^T Q_e (x - x_{e,0}) - 1 \right\}, \tag{4}$$

with Q_e being positive definite matrices and $x_{e,0}$ representing the center coordinates of the (possibly rotated) n_e ellipsoids. Figure 2 represents a schematic view of MMPS and ellipsoidal constraint approximations.

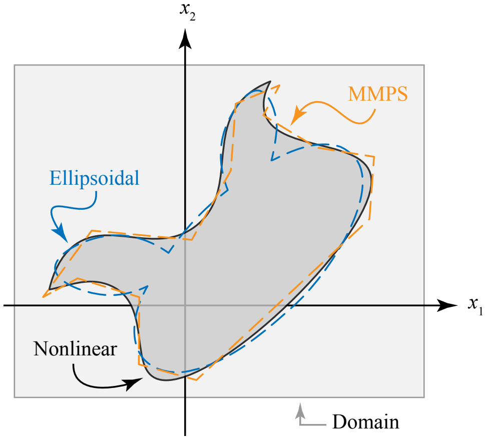


Fig. 2: MMPS and ellipsoidal constraint approximation.

C. Approximation Problem Formulation

All the nonlinear functions $H \in \{F_1, \dots, F_n, G\}$ are approximated by their respective hybrid formulations $h \in \{f_1, \dots, f_n, g\}$ for $g \in \{g_{\text{MMPS}}, g_{\text{ELLP}}\}$ via solving the nonlinear optimization problem

$$\min_{\mathscr{A}} \int_{\mathscr{D}} \frac{\|H(x) - h(x)\|_2}{\|H(x)\|_2 + \varepsilon_0} \, dx, \tag{5}$$

where $\mathscr{A}$ represents the decision variables used to define h and the positive value $\varepsilon_0 > 0$ added to the denominator avoids division by very small values for $\|H(x)\|_2 \approx 0$. For the nonlinear constraint, (5) approximates the boundary of the feasible region, and therefore we call this approach “boundary-based”.

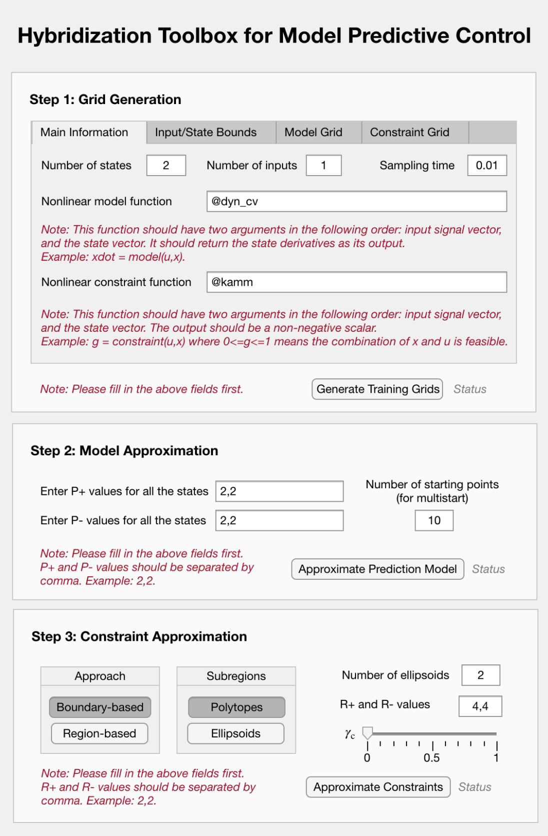
Another method to formulate the constraint approximation problem is the “region-based” approach where we formulate the optimization problem as

$$\min_{\mathscr{A}} \ \gamma_{\text{c}} \frac{\mathscr{V}\{\mathscr{C} \setminus \mathscr{R}\}}{\mathscr{V}\{\mathscr{C}\}} + (1 - \gamma_{\text{c}}) \frac{\mathscr{V}\{\mathscr{R} \setminus \mathscr{C}\}}{\mathscr{V}\{\mathscr{D} \setminus \mathscr{C}\}}, \tag{6}$$

where the operator $\mathscr{V}$ gives the size or “volume” of the region, and $\gamma_{\text{c}} \in [0, 1]$ is a tuning parameter to adjust the relative penalization weight for the misclassification errors regarding inclusion error $\mathscr{C} \setminus \mathscr{R}$, i.e., failing to cover the feasible region, and the violation error $\mathscr{R} \setminus \mathscr{C}$ which corresponds to violating the constraints.

III. Toolbox Architecture

The graphical user interface of the H4MPC toolbox is shown in Fig. 3 and consists of three steps, which correspond to the three modules in H4MPC as shown in Fig. 4: grid generation, model approximation, and constraint approximation. In each module, during grid generation, model approximation, and constraint approximation, the toolbox saves the results as separate `.mat` files in case the user is only interested in the output from one of the modules. The arrows in Fig. 4 illustrate the possible flow of using each module within H4MPC.



Hybridization Toolbox for Model Predictive Control

Step 1: Grid Generation

Number of states: 2
Number of inputs: 1
Sampling time: 0.01

Nonlinear model function: @dyn_cv

Note: This function should have two arguments in the following order: input signal vector, and the state vector. It should return the state derivatives as its output. Example: xdot = model(u,x).

Nonlinear constraint function: @kamm

Note: This function should have two arguments in the following order: input signal vector, and the state vector. The output should be a non-negative scalar. Example: g = constraint(u,x) where 0<=g<=1 means the combination of x and u is feasible.

Note: Please fill in the above fields first.

Step 2: Model Approximation

Enter P+ values for all the states: 2,2
Enter P- values for all the states: 2,2
Number of starting points (for multistart): 10

Note: Please fill in the above fields first. P+ and P- values should be separated by comma. Example: 2,2.

Step 3: Constraint Approximation

Approach: Boundary-based
Subregions: Polytopes
Number of ellipsoids: 2
R+ and R- values: 4,4

Note: Please fill in the above fields first. R+ and R- values should be separated by comma. Example: 2,2.

Fig. 3: Graphical user interface of H4MPC.

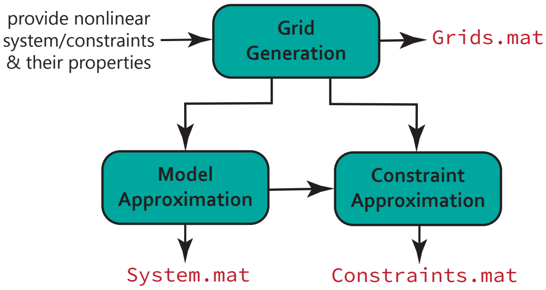


Fig. 4: Architecture of the H4MPC toolbox.

A. Grid Generation Module

In the first step, the user is asked to provide information on:

- number of states and inputs,
- function handles for the nonlinear model and constraint functions,
- sampling time for forward Euler discretization if the model function is continuous-time (can be set to 0 if the provided function is discrete-time), and
- input and state bounds.

For the highly-nonlinear model or constraint functions in automated driving, analytical closed-form solutions for (5)–(6) often do not exist. Therefore, a grid is generated on $\mathscr{D}$ to solve these optimization problems by minimizing the objective function across the grid points. For model approximation, the user can select among four methods described in [12]:

- **Domain-based:** [*points are directly sampled from $\mathscr{D}$*]
  - **Uniform (U):** the points are generated by picking n_{samp} uniformly-spaced points along each axis in $\mathscr{D}$.
  - **Random (R):** a total of n_{rand} points are randomly selected from $\mathscr{D}$.
- **Trajectory-based:** [*n_{sim} open-loop simulations with n_{step} steps of F are run using random inputs from $\mathscr{D}$*]
  - **Steady-state (S):** the initial state of each simulation is selected as the steady-state solution w.r.t. the initial input, i.e., it is assumed that each simulation starts from a steady state.
  - **Randomly-initiated (T):** the initial state of each simulation is randomly selected from $\mathscr{D}$.

For constraint approximation, the grid points are sampled from the whole domain $\mathscr{D}$. Since the region close to the boundary of the feasible region where $G(x) = 1$ is of more interest, the constraint approximation grid is generated by combining a uniform grid (U) with a random grid (R) on the boundary region with width ε_{b}, where $|G(x) - 1| \leqslant \varepsilon_{\text{b}}$. The user can select the number of uniform and boundary grid points, as well as ε_{b} in the user interface. After clicking on the "Generate Training Grids" button, the parameters are saved in the `params` struct and the model and constraint approximation grids, respectively `SM` and `SC`, are generated.

B. Model Approximation Module

In this module, the user is asked to provide (P^+, P^-) values for model approximation, for each state/input separately, i.e. for components of F. We solve (5) for the user-defined values of (P^+, P^-) by MATLAB's nonlinear least squares optimizer, `lsqnonlin`, using the trust-region-reflective algorithm, in view of the number of optimization variables. We solve the problem with $n_{\text{multistart}}$ random initial guesses, where $n_{\text{multistart}}$ is provided by the user, and we select the one with the lowest objective value as the optimal solution. By clicking on the "Approximate Prediction Model" button, the model approximation optimization problem is solved and the hybridized model is saved in the `system` struct.

C. Constraint Approximation Module

The third module approximates the nonlinear constraints using the desired approach and subregion types that can be selected by the user. Similar to the model approximation module, the boundary-based approach leads to a smooth problem, which H4MPC solves for the user-defined values of (R^+, R^-) or n_e. The boundary-based approach offers greater flexibility in fine-tuning the trade-off between encompassing the non-linear region and potentially infringing upon it, ultimately resulting in improved coverage of the non-convex region. However, if the application demands strict adherence to non-linear constraints, it is advisable to opt for the region-based approach in (6), which results in a non-differentiable objective function. The user can then select the parameter γ_c to adjust the weight factor in (6). H4MPC solves the approximation problem using the particle swarm optimizer in MATLAB, which does not require the problem to be differentiable where -based on extensive numerical experiments- we select the swarm size to be 10 times larger than the number of decision variables. The user can then click on the "Approximate Constraints" button, to get the hybridized constraints as the `const` struct.

IV. Case Study

In this section, two examples are investigated to showcase model and constraint approximations handled by H4MPC. The first example involves the single-track vehicle model from [16] as an illustrative model approximation example, and the second considers the Kamm circle constraint [17] as a function of the lateral and longitudinal slips to represent a non-convex feasible region. Multiple selections in H4MPC are tested to highlight the effect of various tuning parameters for hybridization.

A. Nonlinear Vehicle Model

The lateral dynamics of the single-track vehicle model in Fig. 5 is characterized by

$$\dot{r} = \frac{1}{I_{zz}} \left(l_\mathrm{f} \, F_\mathrm{yf} - l_\mathrm{r} \, F_\mathrm{yr} \right), \tag{7a}$$

$$\dot{\beta} = \arctan\left(\frac{F_\mathrm{yf} + F_\mathrm{yr}}{m v_x} - r \right), \tag{7b}$$

with the tire forces described by the Pacejka tire model [18] as

$$F_{xa} = F_{za} \, \mu \, \sin\left(C_\kappa \arctan\left(B_\kappa \kappa_a\right)\right), \tag{8a}$$

$$F_{ya} = F_{za} \, \mu \, \sin\left(C_\alpha \arctan\left(B_\alpha \alpha_a\right)\right), \tag{8b}$$

the κ_a representing the slip ratio on axle $a \in \{\mathrm{f}, \mathrm{r}\}$, and the slip angles being

$$\alpha_\mathrm{f} = \arctan\left(\beta + \frac{l_\mathrm{f} r}{v_x} \right) - \delta, \tag{9a}$$

$$\alpha_\mathrm{r} = \arctan\left(\beta - \frac{l_\mathrm{r} r}{v_x} \right). \tag{9b}$$

The system parameters are shown in Table I. The tire forces should satisfy the tire saturation limits, i.e. Kamm circle constraint [17]

$$F_{x\mathrm{f}}^2 + F_{y\mathrm{f}}^2 \leqslant \left(\mu F_{z\mathrm{f}}\right)^2, \tag{10a}$$

$$F_{x\mathrm{r}}^2 + F_{y\mathrm{r}}^2 \leqslant \left(\mu F_{z\mathrm{r}}\right)^2. \tag{10b}$$

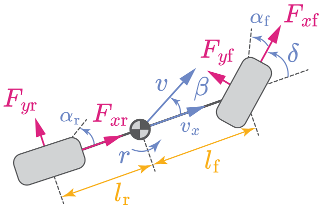


Fig. 5: Single-track vehicle model.

TABLE I: System parameters and variables*

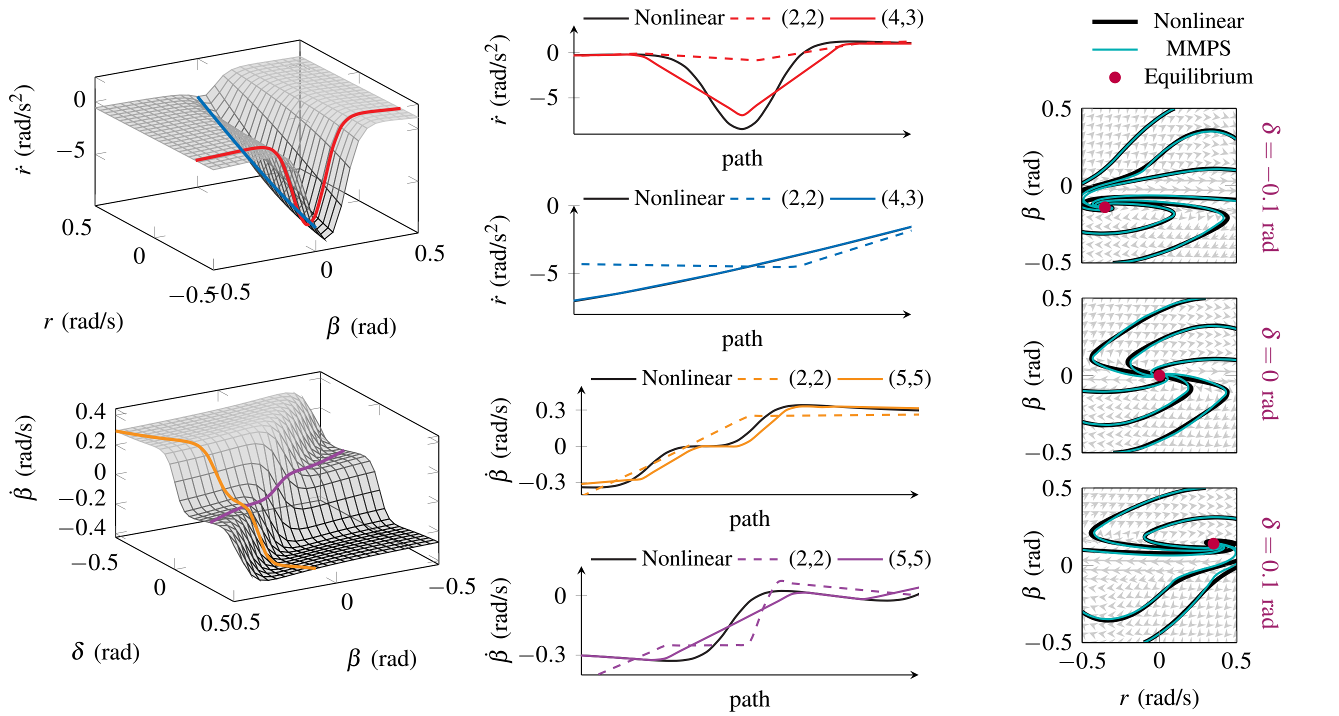
	System parameters		
Par.	**Definition**	**Value**	**Unit**
m	Vehicle mass	1725	kg
I_{zz}	Inertia moment about z-axis	1300	kg/m^2
l_f	CoG** to front axis distance	1.35	m
l_r	CoG to rear axis distance	1.15	m
μ	Friction coefficient	1	–
v_x	Longitudinal velocity	20	m/s
$F_{z\mathrm{f}}$	Normal load on the front axis	5000	N
$F_{z\mathrm{r}}$	Normal load on the rear axis	5000	N
B_κ	Pacejka tire coefficients	11.4	–
C_κ		1.4	–
B_α		10.0	–
C_α		1.6	–
	System variables		
Var.	**Definition**	**Bound**	**Unit**
β	Sideslip angle	[-0.3,0.3]	rad
r	Yaw rate	[-0.5, 0.5]	rad/s
δ	Steering angle (road)	[-0.3, 0.3]	rad
$F_{x\mathrm{f}}$	Longitudinal force on the front axis	[-5000, 0]	N
$F_{x\mathrm{r}}$	Longitudinal force on the rear axis	[-5000, 5000]	N
$F_{y\mathrm{f}}$	Lateral force on the front axis	[-5000, 5000]	N
$F_{y\mathrm{r}}$	Lateral force on the rear axis	[-5000, 5000]	N
α_f	Front slip angle	[-0.4,0.4]	rad
α_r	Rear slip angle	[-0.4,0.4]	rad
κ_f	Front slip ratio	[-1,1]	–
κ_r	Rear slip ratio	[-1,1]	–
s	State vector $:= \begin{bmatrix} r & \beta \end{bmatrix}^T$	–	–
u	Input vector $:= \delta$	–	–

*These kinematic parameters are from [16].
**Center of Gravity

B. Approximation of the Vehicle Dynamics

We use the vehicle model (7) as a case study to investigate model approximation with two states, represented by the variables r and β, and the input δ. To showcase the impact of the parameters (P^+, P^-) on the level of complexity of the hybrid formulation, we examine two sets of parameter values: $(P^+, P^-) \in \{(2,2),(4,3)\}$ for $\dot{r}$ and $(P^+, P^-) \in \{(2,2),(5,5)\}$ for $\dot{\beta}$.

As visualizing a three-dimensional input/state space can be not easy to read, we plot two specific cases in Fig. 6a: $\dot{r}(\beta, r)$ for $\delta = 0$ rad and $\dot{\beta}(\beta, \delta)$ for $r = -0.3$ rad. For more clarity, we provide two cuts along different paths in each plot with their own distinguished colors. We compare the original nonlinear function with its MMPS approximations by using dashed lines for a simpler MMPS approximation and solid lines for a more complex one. Notably, the more complex MMPS approximation, which includes additional terms, provides a more accurate representation of the nonlinear function. For a more clear dynamic comparison, the phase portrait of



(a) Comparison of two MMPS approximations of the vehicle model with different (P^+, P^-) values. For a more clear representation, the functions are plotted along four paths as 2-dimensional cuts of the 3-dimensional function representation.

(b) Phase portraits of the nonlinear and the MMPS approximation with $(P^+, P^-) \in \{(4,3),(5,5)\}$.

Fig. 6: MMPS approximations vs. nonlinear vehicle model: (a) comparing two approximations with different complexity levels, and (b) the resulting state trajectories of the nonlinear model and the more complex MMPS approximation on the phase portrait for three input values.

the more accurate MMPS approximation is compared with the nonlinear one from [16] in Fig. 6b for three values of δ, which shows that the MMPS approximation can generate sufficiently-accurate trajectories on the phase portrait as well.

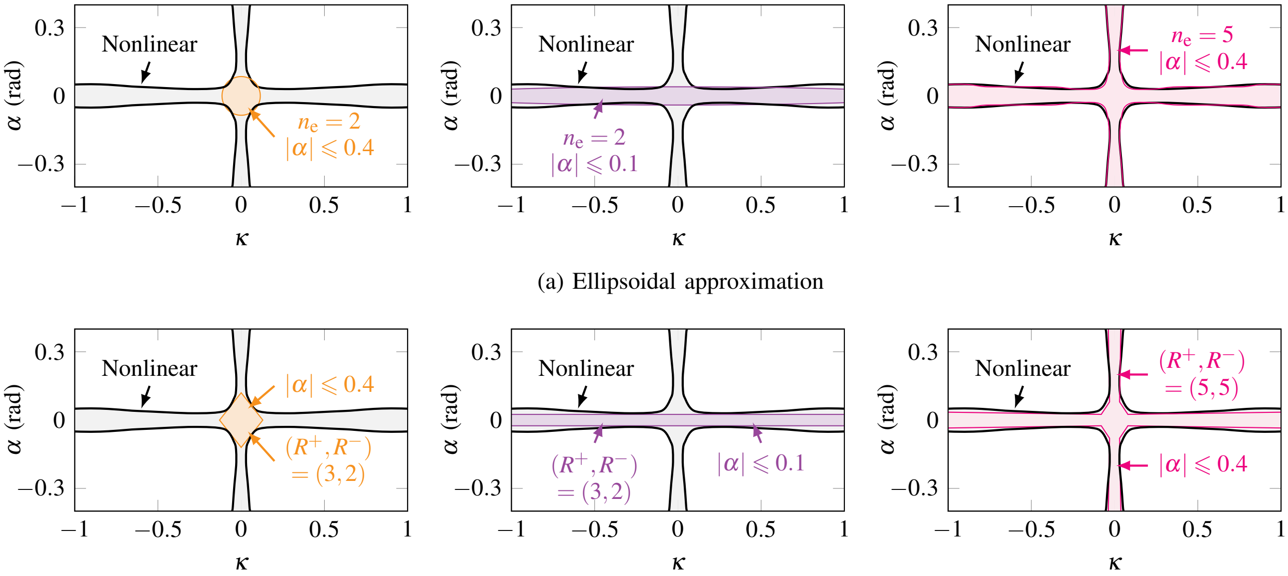
C. Approximation of the Kamm Circle Constraint

To demonstrate the use of hybrid approximation for non-convex feasible regions, we provide the following example with a more intuitive interpretation: approximate the feasible region associated with the Kamm circle constraint defined in (10) within the κ-α plane as its two input/states. Therefore, we employ region-based approximation techniques while ensuring that the constraint violation error is maintained at zero. Figure 7 showcases the hybrid approximations using the ellipsoidal and MMPS methodologies for different values of n_e and (R^+, R^-). Figure 7 illustrates the obtained hybrid approximations.

Additionally, to emphasize the significance of defining boundaries properly, we examine simpler cases with $n_e = 2$ for the ellipsoidal approach and $(R^+, R^-) = (2,2)$ for the MMPS method under two scenarios: setting the boundary on the domain in terms of α to either 0.1 or 0.4 radians. A comparison between the orange and purple approximations for both the ellipsoidal and MMPS approaches reveals that, as the optimizer minimizes the error across the domain, it converges to a more accurate approximation in the vicinity of the origin when $|\alpha|$ is less than or equal to 0.1. However, when the domain is extended to $|\alpha| \leqslant 0.4$, the optimizer converges to an ellipsoidal or polytopic approximation of the feasible region close to the origin, which cannot adequately capture the complexity of the nonlinear constraint further from the origin. By increasing the complexity of the approximation, such as using five ellipsoids or $(R^+, R^-) = (5,5)$, the optimizer finds hybrid approximations that provide better coverage of the feasible region. This improved approximation is represented in pink in both figures.

V. Summary

H4MPC is a open-sources toolbox for hybridization of nonlinear model and constraints in MPC to allow a hybrid formulation of the nonlinear optimization problem. The toolbox includes three modules and provides a user-friendly interface to allow the user to customize the approximation. In this paper, a single-track vehicle model and the tire saturation limits were investigated as two examples to showcase multiple approximation approaches handled in H4MPC and to highlight the influence of their corresponding parameters. We expect the toolbox to be useful in a variety of applications such as



(a) Ellipsoidal approximation

(b) MMPS approximation

Fig. 7: Illustration of MMPS and ellipsoidal approximation of the Kamm circles (10) as a function of α and κ.

automated driving or control of robotic systems. The H4MPC toolbox is freely available from https://bit.ly/H4MPCV1. The next versions of the toolbox will include controller design using the hybridized form of the nonlinear model and physics-based constraints to investigate the effect of approximation complexity on computation time of the MPC optimization problem.

References

[1] P. Stano, U. Montanaro, D. Tavernini, M. Tufo, G. Fiengo, L. Novella, and A. Sorniotti, "Model predictive path tracking control for automated road vehicles: A review," *Annual Reviews in Control*, vol. 55, pp. 194–236, 2023.

[2] N. Groot, B. De Schutter, and H. Hellendoorn, "Integrated model predictive traffic and emission control using a piecewise-affine approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 587–598, 2013.

[3] E. Asarin, T. Dang, and A. Girard, "Hybridization methods for the analysis of nonlinear systems," *Acta Informatica*, vol. 43, no. 7, p. 451, 2007.

[4] J. Lunze and F. Lamnabhi-Lagarrigue, *Handbook of Hybrid Systems Control: Theory, Tools, Applications*. Cambridge University Press, 2009.

[5] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, no. 7, pp. 1085–1091, 2001.

[6] F. Torrisi and A. Bemporad, "HYSDEL-a tool for generating computational hybrid models for analysis and synthesis problems," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 2, pp. 235–249, 2004.

[7] X. Sun, H. Zhang, Y. Cai, S. Wang, and L. Chen, "Hybrid modeling and predictive control of intelligent vehicle longitudinal velocity considering nonlinear tire dynamics," *Nonlinear Dynamics*, vol. 97, no. 2, pp. 1051–1066, 2019.

[8] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[9] S. Di Cairano, H. E. Tseng, D. Bernardini, and A. Bemporad, "Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1236–1248, 2013.

[10] N. Guo, X. Zhang, Y. Zou, B. Lenzo, and T. Zhang, "A computationally efficient path-following control strategy of autonomous electric vehicles with yaw motion stabilization," *IEEE Transactions on Transportation Electrification*, vol. 6, no. 2, pp. 728–739, 2020.

[11] D. Jagga, M. Lv, and S. Baldi, "Hybrid adaptive chassis control for vehicle lateral stability in the presence of uncertainty," in *Mediterranean Conference on Control and Automation (MED 2018)*, no. 3, 2018, pp. 529–534.

[12] L. Gharavi, B. De Schutter, and S. Baldi, "Efficient MPC for emergency evasive maneuvers, part I: Hybridization of the nonlinear problem," *arXiv preprint arXiv:2310.00715*, 2023.

[13] ——, "Efficient MPC for emergency evasive maneuvers, part II: Comparative assessment for hybrid control," *arXiv preprint arXiv:2310.00716*, 2023.

[14] L. Gharavi, "Hybridization Toolbox for Model Predictive Control," 4TU.ResearchData, Delft University of Technology, 2023. [Online]. Available: https://doi.org/10.4121/2a4a7bed-63b9-43d9-a4d2-192bc9163dd1

[15] B. De Schutter, T. van den Boom, J. Xu, and S. S. Farahani, "Analysis and control of max-plus linear discrete-event systems: An introduction," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 30, no. 1, pp. 25–54, 2020.

[16] C. G. Bobier-Tiu, C. E. Beal, J. C. Kegelman, R. Y. Hindiyeh, and J. C. Gerdes, "Vehicle control synthesis using phase portraits of planar dynamics," *Vehicle System Dynamics*, vol. 57, no. 9, pp. 1318–1337, 2019.

[17] R. Rajamani, *Vehicle Dynamics and Control*. Springer Science & Business Media, 2011.

[18] H. Pacejka, *Tire and Vehicle Dynamics*. Elsevier, 2005.

[19] A. Kripfganz and R. Schulze, "Piecewise affine functions as a difference of two convex functions," *Optimization*, vol. 18, no. 1, pp. 23–29, 1987.