Nonlinear Curvature Modeling for MPC of Autonomous Vehicles*

Gonçalo Collares Pereira[1,2], Pedro F. Lima[1], Bo Wahlberg[2], Henrik Pettersson[1], Jonas Mårtensson[2]

***Abstract*— This paper investigates how to compensate for curvature response mismatch in lateral Model Predictive Control (MPC) of an autonomous vehicle. The standard kinematic bicycle model does not describe accurately the vehicle yaw-rate dynamics, leading to inaccurate motion prediction when used in MPC. Therefore, the standard model is extended with a nonlinear function that maps the curvature response of the vehicle to a given request. Experimental data shows that a two Gaussian functions approximation gives an accurate description of this mapping. Both simulation and experimental results show that the corresponding modified model significantly improves the control performance when using Reference Aware MPC for autonomous driving of a Scania heavy-duty construction truck.**

I. Introduction

Autonomous vehicles are becoming a reality and lower-level [1] autonomy starts to be available to the general public (for example, consider Tesla's autopilot functionality [2]). Already these systems provide more comfort to the users, but autonomous vehicles are bound to revolutionize the road-transport industry by making it greener, safer, more comfortable, and more economically viable [3]. Autonomous vehicle research has been booming in recent years, and major manufacturers, suppliers, technology companies, and universities have started projects and collaborations to speed up the development of the technology. The heavy-duty vehicle (HDV) industry has an important and particular interest in high-levels [1] of automation. This industry will, most likely, be an early adopter of the technology in more controlled and constrained environments, such as, e.g., ports and mining sites [4].

One of the core components of an autonomous driving system is the motion controller. Control methods for autonomous vehicles have been widely researched. One of them is model predictive control (MPC), which is a very effective and capable controller due to its ability to handle both input and state constraints explicitly [5]. MPC has been widely used to control autonomous vehicles, in [6], Borrelli et al. propose a nonlinear MPC (NMPC) approach for active-steering control on a double lane change scenario on slippery surfaces. In [7], Falcone et al. have applied a linear time-varying MPC (LTV-MPC) to control an autonomous vehicle on icy roads with speeds up to $21\,\mathrm{m\,s^{-1}}$. In [8], Lima et al. present an LTV-MPC based controller called smooth and accurate MPC (SA-MPC), showcasing its advantages *vs* other controllers experimentally with an HDV. Furthermore, Lima et al., in [9], extend previous work by considering terminal cost and constraints to guarantee stability for a reference tracking LTV-MPC, verifying the design experimentally with an HDV on an evasive maneuver scenario. More recently, in [10], we propose the Reference Aware MPC (RA-MPC), extending previous work by redesigning the optimization problem and proposing a method to systematically handle references received by motion planners. In [10], the RA-MPC is compared against the SA-MPC where its advantages are clear. In this paper, the RA-MPC controller is used in both simulation and experiments to showcase and verify the contribution.

MPC is highly dependent on the model used to predict the system. Vehicle models can be very complex, where, e.g., tire forces, road friction, slip angles, and suspension forces, can be considered, or they can be very simple, e.g., consider the kinematic bicycle model with no slip [11]. The complexity of the model influences the complexity of the MPC making it harder to solve in real-time. The trade-off between very accurate vehicle model and computational power needs to be made when considering real-time systems. In [7], Falcone et al. uses a six state dynamic bicycle model that considers tire forces with the Pacejka tire model [12], this model is vehicle specific and requires good identification of the vehicle parameters. In [13], Kong et al. study both kinematic and dynamic vehicle models to motivate an MPC controller, concluding that depending on the discretization step a kinematic model can perform well compared to a dynamic model. Lima et al. in [8], and we in [10], both use a kinematic bicycle model without forces or slip angles and use vehicle curvature as control input, showing that it is possible to laterally control an HDV without using highly complex dynamic models. In this paper, the work done in [10] with the RA-MPC controller, is extended by improving the prediction model to account better for the vehicle's curvature response.

Summarizing, the main contributions of this paper are:

1) a model modification to the standard kinematic bicycle model, using a nonlinear two Gaussian function to better approximate the curvature response of the vehicle;
2) applicability of the proposed model to control real-time systems using the RA-MPC, with both simulation and experimental results comparing to the standard model.

The rest of this paper is organized as follows. The problem is formulated in Section II. Section III presents the vehicle modeling, including the standard kinematic bicycle

*This work was partially supported by the Wallenberg Artificial Intelligence, Autonomous Systems, and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

[1]Research and Development, Scania CV AB, Södertälje, Sweden.

[2]Division of Decision and Control Systems, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden.

Email addresses: gpcp@kth.se, pedro.lima@scania.com, bo@kth.se, henrik_x.pettersson@scania.com, jonas1@kth.se .

Authorized licensed use limited to: Univ of Calif Santa Barbara. Downloaded on June 22,2021 at 20:19:40 UTC from IEEE Xplore. Restrictions apply.

Fig. 1: A modified R580 Scania truck used for the experiments.

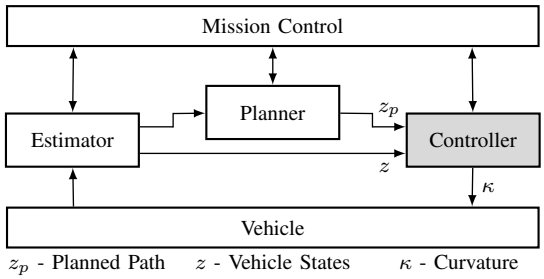


Fig. 2: Simplified closed-loop system architecture.

model, and the model modification to account for the vehicle curvature response. A brief introduction to the RA-MPC is done in Section IV. In Section V, evaluation results of the proposed model for both simulation and experiments with a heavy-duty vehicle are presented. Finally, Section VI presents concluding remarks and outlines future work.

A. Nomenclature and definitions

The notation $a_{i|t}$ means the value of variable a for time $t+i$ predicted at time t. The notation $\mathbf{a}_t = [a_{0|t}, a_{1|t}, ..., a_{N-1|t}]$, where N is the number of prediction steps. The notation $\mathbb{N}_0^N$, denotes the set of all natural numbers from 0 to N. $d(x_t, \mathcal{S})$ denotes the infimum Euclidean distance between the point x_t and any other point in set $\mathcal{S}$, i.e., $d(x_t, \mathcal{S}) = \inf\{|x_t - s| \mid s \in \mathcal{S}\}$.

II. Problem Statement

In previous work [10], we have presented the RA-MPC controller for path following of an autonomous vehicle. This controller is experimentally tested with the heavy-duty vehicle shown in Fig. 1. This type of vehicles have rather slow time-varying dynamics (e.g., empty *vs* full bucket load), which make the task of path following challenging.

Fig. 2 presents a simplified architecture of the autonomous vehicle closed-loop system. The block diagram is not extensive, i.e., neither all the modules necessary nor interfaces are presented, but it is a sufficient representation for the purpose of this work. Mission control represents all the higher-level interface between system and user. Furthermore, the considered system includes, a controller that receives feedback from the vehicle through the sensor fusion and

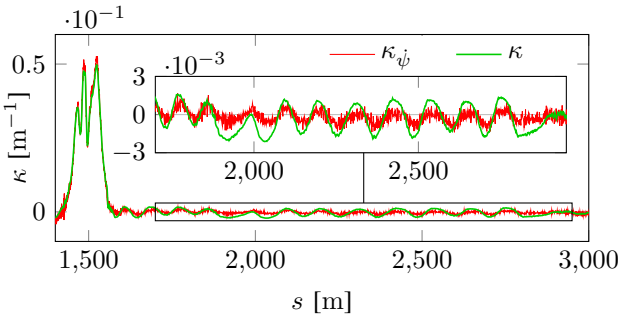


Fig. 3: Curvature measured ($\kappa_{\dot{\psi}}$) and curvature request (κ) for the experiment with RA-MPC in [10].

estimation modules, and receives a path or trajectory from the planner module. The controller sends a high-level control request for the system to achieve the desired behavior. The system, in this case, also includes low-level controllers implemented to track the received request, which for the purpose of this work cannot be modified and are considered as a black-box.

The objective of the control module is to follow the path reference received by the planner (z_p) while maintaining certain error bounds, respecting the actuation constraints, and providing a safe and comfortable ride. When a curvature (κ) is requested the system does not follow exactly the request, because of both external and internal disturbances. Consider Fig. 3, where a snippet of the measured curvature[1] (in red, $\kappa_{\dot{\psi}} = \frac{\dot{\psi}}{v}$, the quotient between measured yaw-rate and velocity) is compared to the curvature request (in green, κ) for the RA-MPC experiment presented in [10]. Clearly, the predicted motion by the MPC is not followed exactly. The objective of this work is to model better the nonlinear curvature response of the system and use the improved model, together with the RA-MPC presented in [10], to improve the overall performance of the system.

III. Vehicle Model

This section presents both the standard road-aligned spatial-based kinematic bicycle model, and the vehicle curvature response modeling, including how to include it in the kinematic model.

A. Road-Aligned Spatial-Based Kinematic Bicycle Model

The model is a coordinate transformation of the conventional kinematic bicycle model. Fig. 4 presents the relation between the *road-aligned* and the *global* coordinate frames for the vehicle model.

The movement of a car-like nonholonomic vehicle at low speeds, with negligible lateral dynamics, can be approximately described by its time-domain equations according to

[1]The signal presented is shifted to account for system delay.

Authorized licensed use limited to: Univ of Calif Santa Barbara. Downloaded on June 22,2021 at 20:19:40 UTC from IEEE Xplore. Restrictions apply.

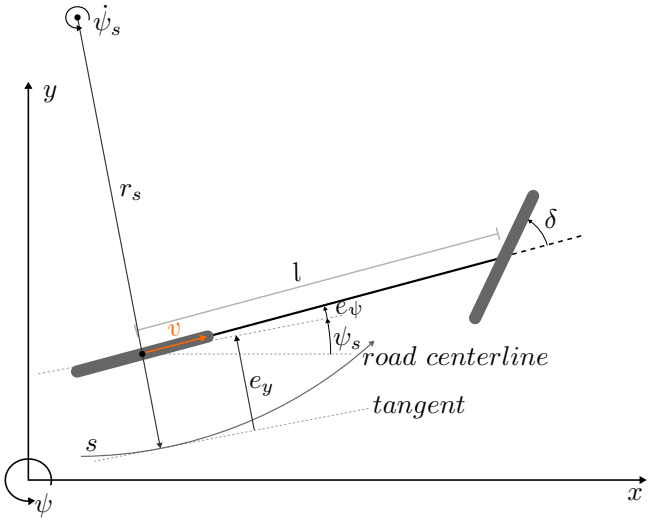


Fig. 4: Nonlinear bicycle model with both the *global* and *road-aligned* coordinate frames depicted.

[14] by,

$$\begin{aligned} \dot{x} &= \frac{\mathrm{d}x}{\mathrm{d}t} = v\cos(\psi), \\ \dot{y} &= \frac{\mathrm{d}y}{\mathrm{d}t} = v\sin(\psi), \\ \dot{\psi} &= \frac{\mathrm{d}\psi}{\mathrm{d}t} = \frac{v}{l}\tan(\delta), \end{aligned} \tag{1}$$

where x and y are the vehicle's coordinates in the *global* coordinate system, ψ is the yaw angle, v is the longitudinal velocity measured in the vehicle local coordinate system, l is the distance between the front and rear axles, and δ is the steering angle of the front wheel. The vehicle curvature κ is described by $\kappa = \frac{\tan\delta}{l}$.

This work only considers the improvement of the lateral controller of the system. The lateral controller objective is to track a path and not a trajectory. So, the vehicle is modeled in the space domain and in a *road-aligned* coordinate frame along the reference path to exclude the time and velocity from the model equations.

The derivation of the spatial-based vehicle model follows [15], where the distance along the reference path s, the lateral error to the path e_y and the yaw error e_ψ are introduced. The following geometric relations can be derived from Fig. 4,

$$\begin{aligned} \dot{e_y} &= \frac{\mathrm{d}e_y}{\mathrm{d}t} &&= v\sin(e_\psi), \\ \dot{e_\psi} &= \frac{\mathrm{d}e_\psi}{\mathrm{d}t} &&= \dot{\psi} - \dot{\psi_s}, \\ \dot{s} &= \frac{\mathrm{d}s}{\mathrm{d}t} &&= \frac{v\cos(e_\psi)}{1-\kappa_s e_y}, \end{aligned} \tag{2}$$

where ψ_s is the yaw angle of the path and κ_s is the curvature of the path. The lateral and yaw errors, e_y and e_ψ, and curvature of the path, κ_s, are computed using orthogonal projection of the vehicle onto the path.

Assuming the vehicle is moving, i.e., $v \neq 0$, and that the velocity is a continuous function, the *chain rule* of calculus can be used to compute the derivative of the composition of two functions $\left(\frac{\mathrm{d}(\cdot)}{\mathrm{d}s} = \frac{\mathrm{d}(\cdot)}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}s}\right)$. Noting that $\frac{\mathrm{d}t}{\mathrm{d}s} = \frac{1}{\dot{s}}$ it is possible to obtain the spatial-based representation of (2),

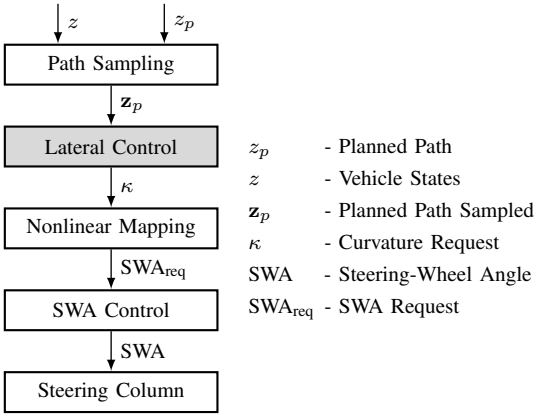


Fig. 5: Simplified lateral controller actuation architecture.

$$\begin{aligned} e_y' &= \frac{\dot{e_y}}{\dot{s}} &&= (1-\kappa_s e_y)\tan(e_\psi), \\ e_\psi' &= \frac{\dot{e_\psi}}{\dot{s}} &&= \frac{(1-\kappa_s e_y)\kappa}{\cos(e_\psi)} - {\psi_s}', \end{aligned} \tag{3}$$

where ${\psi_s}' = \kappa_s$ is the spatial derivative of the path's yaw.

Finally, consider the system states, z_t, and the control input, u_t, at time t, and do the linearization of (3) around reference values. For the states $z = [e_y, e_\psi]^\mathsf{T}$, $[e_{y,r}, e_{\psi,r}]^\mathsf{T} = [0,0]^\mathsf{T}$, and for the control input[2] $u = \kappa$, $\kappa_r = \kappa_s$. Afterwards, do zero-order hold (ZOH) using a spatial interval $\Delta_{s_t} = \Delta T_t v_t$ to obtain the system written in matrix form as $\breve{z}_{t+1} = A_t\breve{z}_t + B_t\breve{u}_t$, where the notation $\breve{z}_t = z_t - z_{r,t}$ and the matrices

$$\begin{aligned} A_t &= \begin{bmatrix} \cos(\Delta_{s_t}\kappa_{s_t}) & \sin(\Delta_{s_t}\kappa_{s_t})/(\kappa_{s_t}) \\ -\kappa_{s_t}\sin(\Delta_{s_t}\kappa_{s_t}) & \cos(\Delta_{s_t}\kappa_{s_t}) \end{bmatrix}, \\ B_t &= \begin{bmatrix} -(\cos(\Delta_{s_t}\kappa_{s_t})-1)/\kappa_{s_t}^2 \\ \sin(\Delta_{s_t}\kappa_{s_t})/\kappa_{s_t} \end{bmatrix}, \end{aligned} \tag{4}$$

are derived symbolically using MATLAB.

B. Vehicle Curvature Response Modeling

In (1), the vehicle yaw-rate is assumed to be $v\kappa$, this assumption is most likely inaccurate when referring to a real vehicle. Consider Fig. 5, where a simplified lateral controller actuation architecture is presented. When the curvature request (κ) is made to the vehicle this value is transformed into a steering-wheel angle (SWA) by a nonlinear mapping. The SWA is then physically actuated by a servo motor on the vehicle's steering-wheel, which turns the steering column resulting in the front-axle wheels turning. Besides the nonlinearities of the curvature to steering conversion, and the nonlinearities and different responses of the actuators, there are also tire forces and external disturbances to consider. In the end, the requested curvature will most likely not lead to the MPC predicted motion of the vehicle. This problem is highly related to understeer and oversteer of the vehicle, but instead of looking at steering angle to yaw-rate, this paper considers modeling the system from curvature request

[2]For the special case where $\kappa_s = 0$, it is possible to substitute κ_s prior to the discretization and then do the ZOH to obtain the system matrices.

Authorized licensed use limited to: Univ of Calif Santa Barbara. Downloaded on June 22,2021 at 20:19:40 UTC from IEEE Xplore. Restrictions apply.

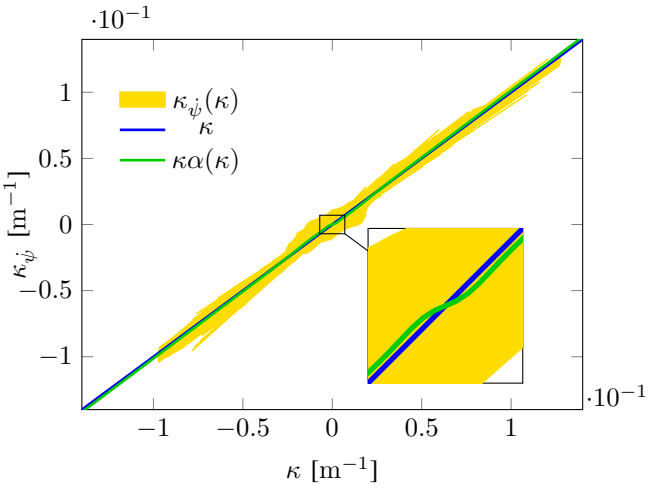


Fig. 6: Curvature response for the vehicle presented in Fig. 1 in yellow. Kinematic bicycle model curvature in blue. Proposed curvature model in green.

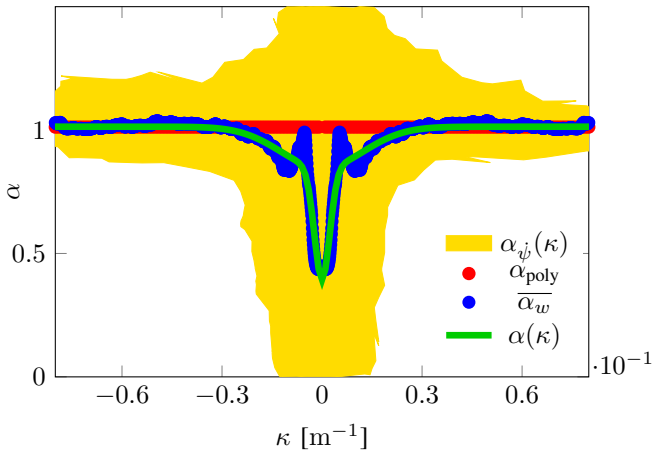


Fig. 7: Variable α measured for the vehicle presented in Fig. 1 in yellow. Polynomial fit (using symmetrical $\kappa_{\dot{\psi}}$) in red, and in blue the sliding window average. Proposed $\alpha(\kappa)$ function in green.

to measured vehicle curvature (according to the kinematic model) and everything else is treated as a black-box.

Fig. 3 shows that there is a mismatch between the measured curvature and the MPC requested curvature. The idea is to change the MPC model to account for this curvature response. Note that this mismatch could also be accounted for outside the MPC by doing an extra nonlinear mapping or controller, but this has the big disadvantage that the control input constraints would most likely not be respected for some points. For example, if the MPC requests maximum curvature rate and then the actual curvature request would be increased even more, the maximum curvature rate request limit would not be respected.

Fig. 6, presents curvature request against curvature measured ($\kappa_{\dot{\psi}}(\kappa)$ in yellow[3]) for multiple experiments with the vehicle presented in Fig. 1. As expected, the same request has a large range of responses due to noise and nonlinearities. Furthermore, Fig. 6 illustrates the kinematic bicycle curvature model presented in Section III-A, κ, in blue.

Consider the variable $\alpha = \frac{\dot{\psi}}{v\,\kappa} = \frac{\kappa_{\dot{\psi}}}{\kappa}$, i.e., the quotient between measured curvature and requested. Fig. 7 presents the measured α points ($\alpha_{\dot{\psi}}(\kappa)$ in yellow[3]), which have some symmetry around the y-axis. This property is explored for easier mathematical modeling, so $\alpha(\kappa)$ is considered an even function. If $\alpha(\kappa)$ is even, the curvature response is odd, i.e., symmetrical with respect to the origin. To account for this symmetry the original data is doubled and the origin symmetrical points are included.

Moreover, for the symmetrical data, consider a first degree polynomial fit (α_{poly}, presented in red in Fig. 7) and the sliding window average ($\overline{\alpha_w}$, presented in blue in Fig. 7), made with $1 \times 10^{-3}\text{m}^{-1}$ window centered on the desired point[4]. The sliding window average, $\overline{\alpha_w}$, presented in Fig. 7 resembles two negative Gaussian or exponentials of the form,

$$\alpha(\kappa) = a_1 e^{-\left(\frac{\kappa}{a_2}\right)^2} + b_1 e^{-\left(\frac{\kappa}{b_2}\right)^2} + c_1, \tag{5}$$

where a_1, a_2, b_1, and b_2 are parameters that define the size and shape of the exponentials, and c_1 defines the position on the y-axis. Then the curve fitting toolbox function *fit* from MATLAB, is used to fit the $\alpha(\kappa)$ parameters for (5) to both the linear polynomial and moving average points, using tuned weights such that close to zero the points are well fitted. This leads to the function $\alpha(\kappa)$ presented in green in Fig. 7, and to the $\kappa\alpha(\kappa)$ proposed curvature model presented in green in Fig. 6. For the vehicle presented in Fig. 1 the function parameters are:

a_1	a_2	b_1	b_2	c_1
-0.1669	0.0174	-0.4513	0.0032	1.0136

TABLE I: $\alpha(\kappa)$ function parameters.

This function is now used in the prediction model by replacing the yaw-rate in (1) for

$$\dot{\psi}_\alpha = v\,\kappa\,\alpha(\kappa). \tag{6}$$

The function $\alpha(\kappa)$ changes the model behavior for low-curvature values, reducing the yaw-rate of the standard model down to $\approx 40\%$ around zero for the same curvature request. In practice, to follow low curvature paths, the controller will request more curvature to compensate for the response of the vehicle. This model leads to a better prediction of the vehicle states, due to the improved prediction of the vehicle's yaw-rate using (6), which includes the curvature response (5). Finally, to obtain the A and B matrices for the MPC,

[3]This is presented in the form of a set, being easier to understand. The number of measured points is very large so in the end it would look very similar.

[4]Both, α_{poly} and $\overline{\alpha_w}$, are actually obtained using the symmetrical $\kappa_{\dot{\psi}}(\kappa)$ and then divided by κ, such that, all points are included and there is no loss of information.

Authorized licensed use limited to: Univ of Calif Santa Barbara. Downloaded on June 22,2021 at 20:19:40 UTC from IEEE Xplore. Restrictions apply.

the *road-aligned* model is derived and then linearized and discretized in the same manner as before for (4).

IV. Controller Design

This section briefly introduces the Reference Aware MPC (RA-MPC). For a more detailed description of the design choices the reader is referred to [10].

The RA-MPC optimization problem is formulated as follows,

$$\begin{aligned}
\min_{\breve{\mathbf{u}}_t,\lambda_1,\lambda_2} \quad & J_{RA} \\
\text{s.t.} \quad & \breve{z}_{i+1|t} = A_{i|t}\breve{z}_{i|t} + B_{i|t}\breve{u}_{i|t}, \\
& \bar{z}_{i|t} = \breve{z}_{i|t} + z_{e,i|t}, \\
& \bar{z}_{0|t} = \bar{z}_t = \breve{z}_{0|t} = \breve{z}_t = z_t, \\
& \breve{u}_{i|t} = u_{i|t} - u_{r,i|t},\ u_{i|t} \in \mathcal{U}_{i|t}, \\
& d(\dot{u}_{i|t}, \dot{\mathcal{U}}_{i|t}) \leqslant \lambda_1,\ d(\ddot{u}_{i|t}, \ddot{\mathcal{U}}_{i|t}) \leqslant \lambda_2, \\
& \lambda_m \geqslant 0,\ \forall i \in \mathbb{N}_0^{N-1},\ \forall m \in \mathbb{N}_1^2
\end{aligned} \tag{7}$$

$$\begin{aligned}
J_{RA} = & \sum_{m=1}^{2} \left({\lambda_m}^{\mathsf{T}} \Lambda \lambda_m + \Lambda \lambda_m \right) + \bar{z}_{N|t}^{\mathsf{T}} Q \bar{z}_{N|t} + \\
& \sum_{i=0}^{N-1} \left(\bar{z}_{i|t}^{\mathsf{T}} Q \bar{z}_{i|t} + \dot{u}_{i|t}^{\mathsf{T}} R_1 \dot{u}_{i|t} + \ddot{u}_{i|t}^{\mathsf{T}} R_2 \ddot{u}_{i|t} \right)
\end{aligned}$$

where the control input is curvature, $u = \kappa$, and the states are the lateral and yaw errors to path, $z = [e_y, e_\psi]$. The matrix

$$Q = \begin{bmatrix} Q_{e_y} + Q_{e_{y,f}} & l\ Q_{e_{y,f}} \\ l\ Q_{e_{y,f}} & l^2\ Q_{e_{y,f}} + Q_{e_\psi} \end{bmatrix},$$

where, Q_{e_y}, $Q_{e_{y,f}}$, and Q_{e_ψ}, are positive weights that penalize the lateral error to path of both the rear and front axles, and the yaw error to path, respectively. The weights R_1 and R_2 are also positive tuning parameters that penalize the curvature rate and acceleration, respectively. To compute the rates and accelerations, the following discrete time differences are used,

$$\dot{\kappa}_t = \frac{\kappa_t - \kappa_{t-1}}{\Delta T_t}, \qquad \ddot{\kappa}_t = \frac{\dot{\kappa}_t - \dot{\kappa}_{t-1}}{\Delta T_t}.$$

The notation z_t is the system states measurement assumed available at each instant. The variable $\breve{z}_t = z_t - z_{r,t}$ is the difference between system states and the vehicle model reference. The error $z_{e,t} = z_{r,t} - z_{p,t}$ is the difference between vehicle prediction model references and the planned path to follow.

The sets $\mathcal{U}_t$, $\dot{\mathcal{U}}_t$, and $\ddot{\mathcal{U}}_t$, are time-varying and velocity dependent sets. These sets take into consideration the constraints of the steering actuator, as well as, the safety and comfort limitations implemented.

Finally, the optimization problem (7) is posed as a quadratic program (QP) and solved online at every iteration.

V. Results

This section starts by setting up the controllers to be tested. Then, a initial analysis of the contribution made in this paper is done through simulations. Finally, experimental results obtained with the heavy-duty vehicle in Fig. 1 are presented.

Both the scenario and the key performance indicators (KPIs) used for the analysis are presented in previous work by the authors [10]. In summary, the KPIs are the lateral error to path, e_y, the curvature request rate, $\dot{\kappa}$, and the curvature request acceleration, $\ddot{\kappa}$. For all KPIs, both the maximum of the absolute value, $|\cdot|_{\max}$, and average of the absolute values, $\overline{|\cdot|}$, are presented.

A. Controller Setup

The controller runs at $50\,\text{Hz}$. It is assumed that a measurement of the system is available at every iteration. Furthermore, the vehicle speed, used in the controller's prediction horizon, is assumed constant and the same as the current measurement. This is a reasonable assumption, since a heavy-duty vehicle has slow dynamics. The velocity control is assumed to be done by a separate controller. All controllers use 10 prediction steps and the optimization problems are solved on-line as QPs using qpOASES [16].

Results are obtained for two different models using the RA-MPC:

1) RA_s: standard model presented in III-A;
2) RA_α: $\alpha(\kappa)$ model presented in III-B.

The controllers use the same control parameters to be straight forward to compare the results between them, their parameters are presented in Table II:

Q_{e_y}	Q_{e_ψ}	$Q_{e_{y,f}}$	R_1	R_2	Λ	ΔT_t
1	25	1	1.5	0.001	10^6	0.44

TABLE II: Control parameters.

Furthermore, the tuning is performed using the real vehicle and the same control parameters are used in simulation to allow for a direct comparison between the simulation and experimental results.

Remark 1: It is important to note that the $\alpha(\kappa)$ function parameters presented in Table I are only used for the real vehicle. For simulation the $\alpha(\kappa)$ function parameters are designed using the procedure presented in Section III-B, but for data collected with the simulator. This represents one of the disadvantages of using a vehicle specific model, i.e., the model needs to be fit to each vehicle being controlled.

The control input constraints are set up in the same manner as in the authors previous work [10].

B. Simulation

Simulation allows to verify the approach and showcase the importance of different controller aspects.

1) Simulator: The simulation environment is fully developed using MATLAB/Simulink. The vehicle simulator is succinctly described in Lima's PhD thesis [3, Ch.3.1]. Additionally, the current version of the simulator also includes the safety limitations and constraints on the steering wheel actuation. The simulator also generates virtual sensor data, that is then fed to a vehicle estimation module, which in turn outputs the current vehicle states to the controller. These modules are courtesy of Scania CV AB.

Authorized licensed use limited to: Univ of Calif Santa Barbara. Downloaded on June 22,2021 at 20:19:40 UTC from IEEE Xplore. Restrictions apply.

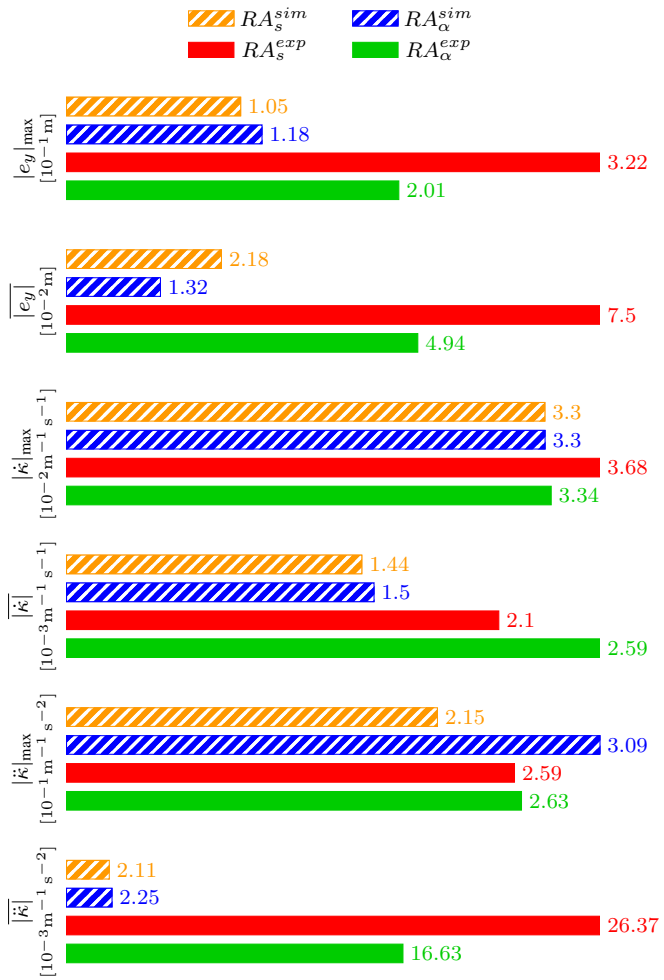
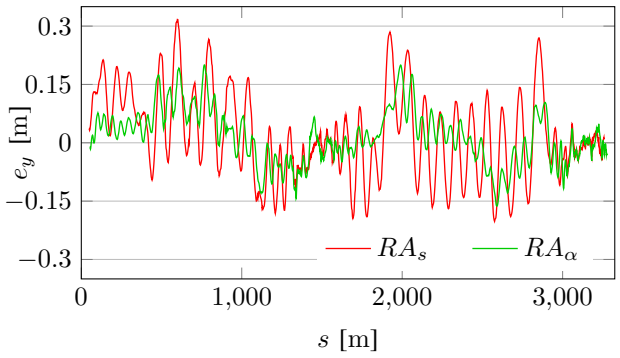


Fig. 8: Performance metrics comparison for both simulation (orange and blue, with striped bars) and experimental results (red and green, with solid bars).

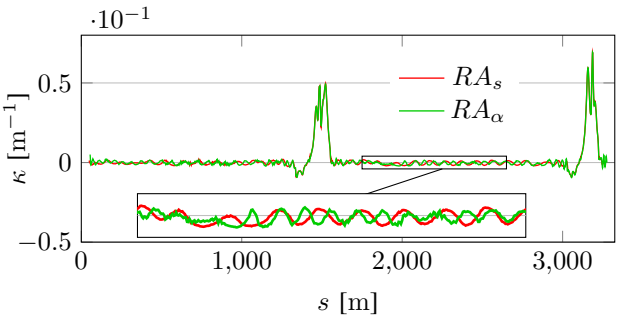
2) Simulation Results: Fig. 8 presents bar charts with two simulation (marked with super-script sim and with striped bars) and two experimental (marked with super-script exp and with solid bars) results.

First, it is important to notice that both controllers operate within the safety limits for the lane, since $|e_y|_{\max} \leqslant 0.5\,\mathrm{m}$. The results presented in Fig. 8 show that the RA_α gives better overall results than the RA_s, because the average lateral error ($\overline{|e_y|}$) to path is significantly smaller, about 40%. Note that the small $1\,\mathrm{cm}$ increase in maximum lateral error ($|e_y|_{\max}$) is not a concern, since both controllers maximum lateral error ($|e_y|_{\max}$) is very small. Most of the other metrics are very similar between the controllers, with the exception of maximum curvature acceleration ($|\ddot{\kappa}|_{\max}$). The bigger maximum curvature acceleration ($|\ddot{\kappa}|_{\max}$) is considered to be a good trade-off, because of the major improvement in average lateral error ($\overline{|e_y|}$) and the fact that the average curvature acceleration ($\overline{|\ddot{\kappa}|}$) is very similar for both controllers.

The computation times for a iteration of the RA-MPC with the proposed model are measured, and the worst time in multiple runs is less than $2.9\,\mathrm{ms}$. Since the controller runs at $50\,\mathrm{Hz}$ it is real-time implementable. These measurements were done with a laptop running an Intel(R) Core(TM) i7-8850H CPU at $2.6\,\mathrm{GHz}$, Windows 10, and MATLAB 2016b.



(a) Lateral error to path.



(b) Controller curvature request.

Fig. 9: Experimental results.

C. Experiments

After verifying the controller and models in simulation they can be tested in a real vehicle (Fig. 1). Fig. 8 presents the experimental results of both RA_s and RA_α MPCs (with super-script exp). The lateral error to path, e_y, and the curvature request, κ, along the path, for these runs are presented in Fig. 9. Note that these results were obtained on a wet asphalt road, which lead to a slight deterioration of performance when compared to the results presented in [10] for the RA_s controller.

Both controllers achieve a performance well below the safety limits of $0.5\,\mathrm{m}$. The results presented in Fig. 8 show that the RA_α is better than the RA_s, since it is able to follow the path with better accuracy in almost all metrics, specially in the lateral error to path. There are two metrics where the standard model behaves better than the modified model, $|\ddot{\kappa}|_{\max}$ and $\overline{|\dot{\kappa}|}$. The maximum curvature acceleration ($|\ddot{\kappa}|_{\max}$) being higher for RA_α than for RA_s, is not a concern since this is just one point, and because the average curvature acceleration is significantly smaller for RA_α, so over the whole run RA_α uses less curvature acceleration. The average curvature rate ($\overline{|\dot{\kappa}|}$) being higher for RA_α than for RA_s is of more concern, but when compared to the major improvement

Authorized licensed use limited to: Univ of Calif Santa Barbara. Downloaded on June 22,2021 at 20:19:40 UTC from IEEE Xplore. Restrictions apply.

on lateral error to path this is considered a good trade-off.

Fig. 9 shows that both models have very similar behavior on the high curvature parts of the track (around $1500\,\mathrm{m}$ and $3200\,\mathrm{m}$). This behavior is expected since $\alpha(\kappa)$ is very close to 1 for high curvature values (Fig. 7), i.e., the models are very similar for these values. On the higher speed and straighter parts of the track the tracking error is significantly smaller for RA_α and the oscillations have significantly smaller amplitude. However, the oscillations have a higher frequency for RA_α than for RA_s.

A frequency analysis of both oscillatory parts of the e_y signal shows that the peak frequencies (in these experiments) for RA_s are $0.140\,\mathrm{Hz}$ and $0.118\,\mathrm{Hz}$, with $0.077\,\mathrm{m}$ and $0.053\,\mathrm{m}$ amplitude, respectively, and for RA_α are $0.210\,\mathrm{Hz}$ and $0.1779\,\mathrm{Hz}$, with $0.025\,\mathrm{m}$ and $0.019\,\mathrm{m}$, respectively. From these values it is hard to say which one provides a more comfortable experience since the RA_s oscillates slower but with more than double the amplitude. Subjectively, the in vehicle experience is more or less the same, being comfortable for both, and maybe feeling slightly safer for the RA_α since there is less lateral movement on the lane. Finally, it is important to mention, that the simulation results do not present these oscillations, meaning that their cause comes from the more complex and difficult to control real system.

In conclusion, the proposed model is considered to be a good improvement to the controller. This conclusion is mostly linked to the major improvement in path following lateral error accuracy, which improved by more than $30\,\%$ in both metrics.

VI. Conclusion and Future Work

This paper has investigated a vehicle specific model modification for lateral control of autonomous heavy-duty vehicles. The yaw-rate state of the model is modified to account better for the nonlinearities of the vehicle's curvature response. This new model is then compared against the standard kinematic yaw-rate model, by using both models together with the RA-MPC presented in [10]. The proposed model with the RA-MPC achieves better control performance, since the lateral error to path is more than $30\,\%$ smaller when compared to the standard kinematic bicycle model with the same controller.

Future work includes testing how the vehicle handles different environments and how the curvature response changes. Furthermore, the authors would like to test the controller in different vehicles and check if and how the proposed model can be adapted to different vehicles and if the achieved performance is in pair between vehicles. Moreover, stability guarantees for the controller are also part of future work. Finally, it would be interesting to consider other models and methods, such that, disturbance rejection and offset-free path following can be achieved.

References

[1] SAE International, *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, jun 2018.

[2] Tesla, "Autopilot," 2019. Available at https://www.tesla.com/support/autopilot.

[3] P. F. Lima, "Optimization-based motion planning and model predictive control for autonomous driving: With experimental evaluation on a heavy-duty construction truck," 2018.

[4] J. Peters, "The future of autonomous vehicles runs off roads and on to farms, construction sites and mines," 2019. Available at https://techcrunch.com/2019/07/10/autonomous-vehicle-startups-are-dead-long-live-autonomous-vehicle-startups/.

[5] A. Rupp and M. Stolz, *Survey on Control Schemes for Automated Driving on Highways*, pp. 43–69. Cham: Springer International Publishing, 2017.

[6] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat, "Mpc-based approach to active steering for autonomous vehicle systems," *Int. J. of Vehicle Autonomous Systems*, vol. 3, no. 2/3/4, 2005.

[7] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on Control Systems Technology*, vol. 15, pp. 566–580, May 2007.

[8] P. F. Lima, M. Nilsson, M. Trincavelli, J. Mårtensson, and B. Wahlberg, "Spatial model predictive control for smooth and accurate steering of an autonomous truck," *IEEE Transactions on Intelligent Vehicles*, vol. 2, pp. 238–250, Dec 2017.

[9] P. F. Lima, G. Collares Pereira, J. Mårtensson, and B. Wahlberg, "Experimental validation of model predictive control stability for autonomous driving," *Control Engineering Practice*, vol. 81, pp. 244 – 255, 2018.

[10] G. Collares Pereira, P. F. Lima, B. Wahlberg, H. Pettersson, and J. Mårtensson, "Reference aware model predictive control for autonomous vehicles," 2020. *To appear in Proceedings of the 2020 IEEE Intelligent Vehicles Symposium (IV)*.

[11] R. Rajamani, *Vehicle Dynamics and Control*. Mechanical Engineering Series, 2nd ed., 2012.

[12] E. Bakker, L. Nyborg, and H. B. Pacejka, "Tyre modelling for use in vehicle dynamics studies," *SAE Transactions*, vol. 96, pp. 190–204, 1987.

[13] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1094–1099, June 2015.

[14] A. De Luca, G. Oriolo, and C. Samson, "Feedback control of a nonholonomic car-like robot," in *Robot motion planning and control*, pp. 171–253, Springer, 1998.

[15] Y. Gao, A. Gray, J. V. Frasch, T. Lin, E. Tseng, J. K. Hedrick, and F. Borrelli, "Spatial predictive control for agile semi-autonomous ground vehicles," *Proceedings of the 11th International Symposium on Advanced Vehicle Control*, vol. VD11, no. 2, pp. 1–6, 2012.

[16] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, pp. 327–363, Dec 2014.

Authorized licensed use limited to: Univ of Calif Santa Barbara. Downloaded on June 22,2021 at 20:19:40 UTC from IEEE Xplore. Restrictions apply.