RNN-based linear parameter varying adaptive model predictive control for autonomous driving

Yassine Kebbati, Naïma Aït Oufroukh, Dalil Ichalal, Vincent Vigneron

RNN-based linear parameter varying adaptive model predictive control for autonomous driving

Yassine Kebbati [a], Naima Ait-Oufroukh [a], Dalil Ichalal [a], Vincent Vigneron [a,b]

[a] *IBISC-EA4526, Université Paris-Saclay, Evry, France*

[b] *School of Applied Sciences (FCA),UNICAMP, Limeira, Brazil*

Abstract

Autonomous driving is a complex and highly dynamic process that ensures controlling the coupled longitudinal and lateral vehicle dynamics. Model predictive control, distinguished by its predictive feature, optimal performance, and ability to handle constraints, makes it one of the most promising tools for this type of control application. The content of this article handles the problem of autonomous driving by proposing an adaptive linear parameter varying model predictive controller (LPV-MPC), where the controller's prediction model is adaptive by means of a recurrent neural network. The proposed LPV-MPC is further optimized by a hybrid Genetic and Particle Swarm Optimization Algorithm (GA-PSO). The developed controller is tested and evaluated on a challenging track under variable wind disturbance.

Key words: Autonomous Driving, Linear Parameter Varying, Model Predictive Control, Neural Networks, Optimization.

1 INTRODUCTION

The ever-increasing number of vehicles is inducing terrible traffic conditions and increasing air pollution in today's world. With most people having to spend countless hours to commute between home and work, driving has become a source of strain and stress, increasing the possibility of road accidents. Therefore, the research community has been striving to accelerate the shift toward autonomous driving by replacing human drivers with automatic control systems. This shift will improve traffic safety and enhance mobility while boosting human productivity since driving time can be used to do productive tasks instead. The considerable advances in artificial intelligence and information processing technologies further accelerate the shift to autonomous driving. The latter is a complex multidisciplinary process, involving sensing, perception, planning, and control. Control is the final and most important step of the process, it can be divided into longitudinal control, in charge of speed tracking, and lateral control, which handles the steering.

Research works can be divided into two main categories; the first one addresses the longitudinal and lateral controls separately, and the second one couples both tasks together. For instance, Xu *et al.* [1] introduced an optimized controller for speed regulation, where road slope, speed profile, and vehicle dynamics are integrated into the model. Paper [2] addressed the longitudinal control by a self-adaptive PID controller, whose optimization and adaptation were based on neural networks and genetic algorithms. An adaptive neural network PID controller was developed by Han *et al.* [3] for the path-tracking task. The authors applied it to a second-order vehicle model, and they used a forgetting factor least square algorithm to estimate model parameters. Guo *et al.* [4] dealt with path tracking. They developed an MPC controller that takes into account the changing road conditions and small-angle assumptions as a form of measurable disturbance. The authors used the differential evolution algorithm to solve the control problem. Authors of [5] developed a model predictive controller (MPC) for lateral control and optimized its weighting matrices using fuzzy inference systems (FIS). Corno *et al.* [6] developed an LPV H_∞ lateral controller, where they exploited the lateral error and look-ahead distance of the vehicle to ensure better robustness and account for actuator nonlinearities under low speeds.

⋆ Corresponding author Yassine Kebbati.

Email addresses: yassine.kebbati@univ-evry.fr (Yassine Kebbati), naima.aitoufroukh@univ-evry.fr (Naima Ait-Oufroukh), dalil.ichalal@univ-evry.fr (Dalil Ichalal), vincent.vigneron@univ-evry.fr (Vincent Vigneron).

The authors tested their controller in high-speed driving scenarios and evasive maneuvers. Authors of [7] worked on adaptive MPC designed with Laguerre functions for path tracking, they optimized the controller tuning with an improved PSO algorithm. A lookup table approach was used to achieve online controller adaptation. However, the employed method cannot account for all possible cases despite the good results that were achieved. Additionally, the same authors enhanced their approach in [8] and replaced the lookup table approach with neural networks and adaptive neuro-fuzzy inference systems so that the adaptions generalize beyond the lookup table data. Although significant tracking improvements were achieved, this approach still requires long offline optimizations.

In [9], a coordinated lateral and longitudinal control using LPV-MPC for lateral control with PSO-PID for speed regulation was proposed. In other works, Yao *et al.* [10] developed an (MPC) path tracking controller that includes longitudinal speed compensation, their approach aims to overcome the assumption of constant longitudinal speed along the control horizon. This technique seeks to minimize the control deviation caused by fast speed and acceleration variations. In [11], Wang *et al.* designed an improved (MPC) control strategy that includes an adaptive fuzzy controller, the latter aims to change the weights of the cost function to tackle the problem of ride discomfort caused by fixed weights in the standard MPC. Li *et al.* proposed in his paper [12] an (NMPC) for trajectory tracking, their controller was based on nonlinear vehicle dynamics, and Pacejka tire model [13], and it tracks the yaw angle and lateral position. Most of the above-mentioned studies ensure autonomous driving using coordination between lateral and longitudinal control. However, full autonomy requires handling both lateral and longitudinal controls simultaneously, especially during highly dynamic maneuvers.

Kebbati *et al.* [14] addressed the coupled lateral and longitudinal control, their solution was based on an LPV-MPC approach with genetic algorithm optimization. A Takagi-Sugeno-based MPC (TS-MPC) for autonomous driving was proposed by Alcala *et al.* [15]. This data-driven approach was merely used to learn a Takagi-Sugeno representation of the vehicle dynamics, which was used by the MPC controller with a Moving Horizon Estimator (MHE) to achieve coupled longitudinal and lateral control. Papers [16–18] used the nonlinear model predictive control (NMPC) for autonomous driving and parking applications. Kabzan *et al.* [19] proposed an online learning MPC controller for autonomous racing by learning the model errors online using Gaussian process regression. Similarly, paper [20] addressed autonomous racing using an online learning NMPC with Gaussian process regression and online moving horizon state estimation. Paper [21] dealt with autonomous race driving, the authors introduced an (NMPC) for the reduced F1/10 platform. The authors interpolated the circuit boundaries using 3^{rd} order polynomials and implemented them as inequality constraints on the lateral position to keep the vehicle inside the track. Kloeser *et al.* [22] proposed an (NMPC) to tackle autonomous racing for a 1:43 scale race car using a singularity-free path parametric model. The authors used partial spatial reformulation of the prediction model to exclude singularities and implemented obstacle avoidance in the optimization problem as a constraint with the objective of maximizing progress on the path. A review of the most widely used control strategies for autonomous driving is provided in [23].

This paper contributes to the above-mentioned literature by proposing an improved controller for coupled speed and steering control. The main contributions are threefold; First, to ensure real-time application with minimal computing resources and to overcome the heavy computations of NMPC, an adaptive LPV-MPC is developed for autonomous driving. Second, a novel hybrid GA-PSO algorithm is proposed for optimizing the controller's cost function to achieve optimal control actions and automate the controller's tuning process. Third, a Jordan recurrent neural network is designed and trained to learn the tire lateral dynamics by predicting the cornering stiffness coefficients from measurable parameters only, such as velocities and accelerations.

The article is divided as follows: Section 2 discusses the model of the vehicle's coupled lateral and longitudinal dynamics. Section 3 explains the development of the proposed controller, the adaptation approach using recurrent neural networks, and the optimization of the controller's cost function through the proposed hybrid GA-PSO algorithm. Evaluation results of the learning approach, optimization, and control are presented and analyzed in section 4. Finally, Section 5 provides conclusions and gives perspectives for future work.

2 VEHICLE MODELING

The vehicle is modeled in this article using the common bicycle dynamics model [24, 25], which is considered accurate for control design and easy to implement for real-time control applications as it does not require long computations. Its simplicity is in lumping the front wheels together as well as the rear wheels to form a single-track or bicycle representation as illustrated in Fig. 1. The tire lateral forces, being a function of the slip angles, govern the vehicle's lateral dynamics. As expressed in equations (1), the model takes into account the longitudinal, the lateral, and the yaw dynamics and includes the heading and lateral position errors as well:

$$\begin{cases} \dot{v}_x &= \alpha_x + \omega v_y - \frac{1}{m}(F_{yf}\sin\delta + F_d) \\ \dot{v}_y &= \frac{1}{m}(F_{yf}\cos\delta + F_{yr}) - \omega v_x \\ \dot{\omega} &= \frac{1}{I}(F_{yf} l_f \cos\delta - F_{yr} l_r) \\ \dot{y}_e &= v_x \sin\theta_e + v_y \cos\theta_e \\ \dot{\theta}_e &= \omega - \frac{v_x \cos\theta_e - v_y \sin\theta_e}{1 - y_e k} \\ F_{yf} &= C_f \alpha_f \\ F_{yr} &= C_r \alpha_r \\ F_d &= \mu m g + \frac{1}{2}\rho C_d A v_x^2 \end{cases} \tag{1}$$

The linear longitudinal and lateral velocities and yaw rate in the body frame are represented by v_x, v_y, and ω, respectively. $F_{y(f,r)}$ express the lateral forces of the front and rear tires, respectively. The total drag force is expressed by F_d, where C_d, ρ, and A represent the drag coefficient, air density, and the vehicle cross-sectional area, respectively. The parameters θ_e and y_e represent the heading and lateral position errors, where k is the road curvature. The inertia and the mass of the vehicle are represented by I and m, and $l_{(f,r)}$ are the distances between the front/rear wheel axles and the vehicle's center of gravity, respectively. The terms α_x and δ are the acceleration and steering controls, and the parameters μ and g represent the friction coefficient and the gravity. Finally, the front/rear tire cornering stiffness coefficients are given by $C_{(f,r)}$, and $\alpha_{(f,r)}$ design the slip angles for the front/rear wheels with ϵ being an additional term to avoid singularities in the model, which are respectively given by:

$$\begin{cases} \alpha_f &= \delta - \tan^{-1}\left(\frac{v_y}{v_x+\epsilon} - \frac{l_f \omega}{v_x+\epsilon}\right) \\ \alpha_r &= -\tan^{-1}\left(\frac{v_y}{v_x+\epsilon} + \frac{l_r \omega}{v_x+\epsilon}\right) \end{cases} \tag{2}$$

The full model can be considered as a non-linear function that maps the state vector (x) with the input vector (u) and the road curvature (k) as follows:

$$\dot{x} = f(x, u, k) \tag{3}$$

where $x = [v_x\ v_y\ \omega\ y_e\ \theta_e]^T$ and $u = [\delta\ \alpha_x]^T$.

3 CONTROLLER DESIGN

The control strategy is based on the LPV approach since it allows capturing model nonlinearities, and ensures real-time application with minimal computing resources, therefore, overcoming the heavy computations of NMPC [24]. Furthermore, the LPV approach is adaptive to varying parameters, thereby, increasing the model's accuracy. The model presented in Section 2 is reformulated in an LPV form and transformed into a state space representation. Thus, the state and control

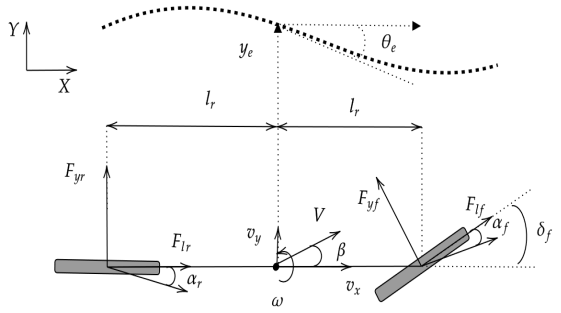

Fig. 1. Bicycle dynamic model with tracking error.

matrices will depend on a scheduling vector of varying parameters. Therefore, the nonlinearities of the model are captured by embedding linear varying parameters into the system matrices, which provides a simple but accurate model for control design. LPV systems are known as a class of linear systems with their parameters being functions of scheduling signals that can be external or internal. The LPV state space formulation of the system is given by (4) based on the following scheduling vector $\psi = [\delta\ v_x\ v_y\ \theta_e\ y_e\ k]^T$.

$$\dot{x} = A(\psi)x + B(\psi)u \tag{4}$$

The state matrix $A(\psi)$ and control matrix $B(\psi)$ can then be derived as the following:

$$A(\psi) = \begin{bmatrix} A_{11} & A_{12} & A_{13} & 0 & 0 \\ 0 & A_{22} & A_{23} & 0 & 0 \\ 0 & A_{32} & A_{33} & 0 & 0 \\ A_{41} & A_{42} & 0 & 0 & 0 \\ A_{51} & A_{52} & 1 & 0 & 0 \end{bmatrix}, \tag{5}$$

$$B(\psi) = \begin{bmatrix} B_{11} & 1 \\ B_{21} & 0 \\ B_{31} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \tag{6}$$

where the terms are given by:
$A_{11} = \frac{-\mu g}{v_x} - \frac{\rho C_d A v_x}{2m}$, $A_{12} = \frac{C_f \sin\delta}{m v_x}$,
$A_{13} = \frac{C_f l_f \sin\delta}{m v_x} + v_y$, $A_{22} = -\frac{C_r + C_f \cos\delta}{m v_x}$,
$A_{23} = -\frac{C_f l_f \cos\delta - C_r l_r}{m v_x} - v_x$, $A_{32} = -\frac{C_f l_f \cos\delta + C_r l_r}{I v_x}$,
$A_{33} = -\frac{C_f l_f^2 \cos\delta + C_r l_r^2}{I v_x}$, $A_{41} = \sin\theta_e$, $A_{42} = \cos\theta_e$,
$A_{45} = v_x$, $A_{51} = -\frac{k\cos\theta_e}{1 - y_e k}$, $A_{52} = \frac{k\cos\theta_e}{1 - y_e k}$,
$B_{11} = -\frac{C_f \sin\delta}{m}$, $B_{21} = -\frac{C_f \cos\delta}{m}$, $B_{31} = -\frac{C_f l_f \cos\delta}{I}$.

Generally speaking, the MPC approach exploits the plant model to foresee its behavior over a prediction horizon N_p. Based on these predictions, it generates an optimal control sequence by solving a constrained convex optimization problem. The MPC approach is based on the receding horizon principle, where only the first term of the optimal control sequence is used. The LPV model, discretized with T_s sampling time, is used to build the MPC prediction model. This means that the scheduling vector is used to instantiate the LPV model iteratively. The parameters of the scheduling vector can be obtained from sensors, planners, or previous MPC predictions. In this regard, the MPC problem is formulated as the following constrained quadratic optimization:

$$\begin{aligned}
&\min_{\Delta U_k} J_k = \sum_{i=0}^{N_p-1} \Big((r_{k+i} - x_{k+i})^T Q (r_{k+i} - x_{k+i}) + \\
&\qquad \Delta u_{k+i} R \Delta u_{k+i} \Big) + x_{k+N_p}^T Q x_{k+N_p} \\
&s.t: \\
&\qquad x_{k+i+1} = x_{k+i} + A(\psi_{k+i}) x_{k+i} + B(\psi_{k+i}) u_{k+i} dt \\
&\qquad u_{k+i} = u_{k+i-1} + \Delta u_{k+i} \\
&\qquad \Delta u_{min} \leq \Delta u_k \leq \Delta u_{max} \\
&\qquad u_{min} \leq u_k \leq u_{max} \\
&\qquad x_{min} \leq x_k \leq x_{max}
\end{aligned} \tag{7}$$

The terms x, u, and N_p define the state vector, the control vector, and the prediction horizon, respectively. The weighting matrices $Q \in \mathbb{R}^{5\times5}$ and $R \in \mathbb{R}^{2\times2}$ are semi-positive definite, and they penalize the states and the control effort. The longitudinal speed profile is given by the reference vector r_{k+i}, and the upper and lower bounds on the control actions, control increments, and states are respectively expressed as $[u_{min}, u_{max}]$, $[\Delta u_{min}, \Delta u_{max}]$ and $[x_{min}, x_{max}]$. The last term of the cost function is added to increase stability, a terminal cost and a terminal set are needed to ensure the asymptotic stability of MPC with a quadratic stage cost. Otherwise, a sufficiently long prediction horizon is required to guarantee asymptotic stability as illustrated by Franz *et al.* [26]. In addition, paper [27] states that adding a terminal cost to the MPC formulation and using a sufficiently long prediction horizon ensures MPC stability without the need for terminal constraints.

3.1 Controller Adaptation with Jordan Network

In most cases, research works are based on linearized tire models, where the tire lateral force depends linearly on the slip angle through a constant known as the cornering stiffness coefficient. However, this is only valid for relatively small slip angles and not during fast and challenging maneuvers, the so-called cornering stiffness coefficient may be time-varying and changes during different types of maneuvers. To deal with this issue, we use machine learning tools to predict the cornering stiffness coefficient online using information from measurable parameters of the vehicle's dynamics. We assume that measurable parameters such as the longitudinal (v_x) and lateral (v_y) velocities, the steering angle (δ), the acceleration (α_x), and the yaw rate (ω)) are enough to capture the tire dynamics. Hence, the prediction model of the LPV-MPC controller is adapted online using this approach (see Fig. 2), this strategy improves the prediction capability and precision of the LPV-MPC. In this article, we propose the use of a modified deep Jordan network to learn tire dynamics. Jordan networks are a simple type of recurrent neural network where the delayed output signal from the output layer is reinjected with the inputs to account for temporal dependencies and improve network predictions, meaning that the previous network predictions become inputs for future predictions. Fig. 3 illustrates the simplest form of a Jordan network where i represents the inputs, y is the output, ω are the weights, and b is the bias. The network consists of two inputs, one output, and one hidden layer with n hidden neurons. It can be modeled as follows:

$$y(k) = f(u(k-1), y(k-1)) \tag{8}$$

Considering $\omega_{j,k}^{(n)}$ as the weights of the n^{th} layer between neurons j and k of the previous and actual layers, respectively, the output signal can be expressed as:

$$y(k) = \omega_0^{(2)} + \sum_{i=0}^{n} \omega_i^{(2)} \zeta(z_i(k)) \tag{9}$$

The term ζ is the activation function of the hidden layer, and $z_i(k)$ represents the sum of i^{th} hidden node, and it is given by:

$$z_i(k) = \omega_{0,i}^{(1)} + \omega_{1,i}^{(1)} u(k-1) + \omega_{2,i}^{(1)} y(k-1) \tag{10}$$

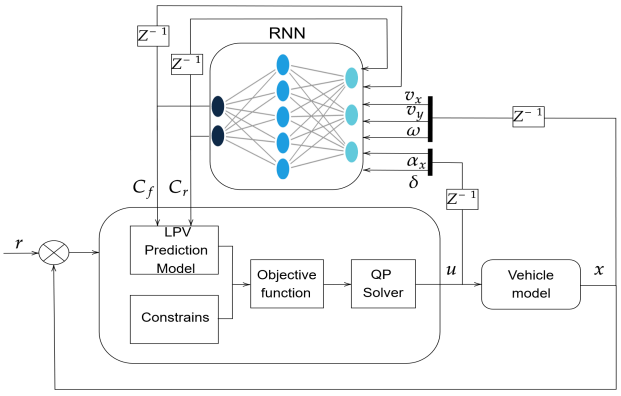

Fig. 2. Adaptive LPV-MPC approach.

3.2 *Controller Tuning with Hybrid GA-PSO*

Tuning the MPC manually is a difficult task that requires expertise and time, and eventually, it may not result in optimal performance. Thus, to optimize the designed LPV-MPC, we propose a hybrid Genetic Algorithm (GA) and Particle Swarm Optimizer to tune the weighting matrices of the quadratic cost function by minimizing the MPC tracking root mean squared error (RMSE) as a fitness function. GAs are a global optimization technique based on Darwin's biological evolution theory. They can find the optimums of discontinuous and nondifferentiable objective functions [28]. Generally speaking, the genetic algorithm initializes a population set that contains encoded solutions, which are called chromosomes in the genetic jargon. These possible solutions are improved iteratively and their optimality is assessed by a fitness function. GA evolution operations include the selection, the crossover, and the mutation processes, which control the search capability and the quality of the solutions. They consist of different functions that impact the performance of the algorithm at different degrees [28]. For instance, the selection process selects the best chromosomes to be enhanced by the crossover and mutation operations. Alternatively, the crossover seeks to produce high-quality solutions by mixing genetic data, while mutation introduces new genes to complement the crossover as illustrated in Fig. 4. The most common selection operations in the literature, are the roulette wheel (RWS) and tournament selection (TS) [28, 29]. For the crossover operation, one finds single/multi-point, uniform, and shuffle crossover. Similarly, mutation operations include inversion and random resetting. Researchers have essentially worked towards improving these operations to further optimize genetic algorithms.

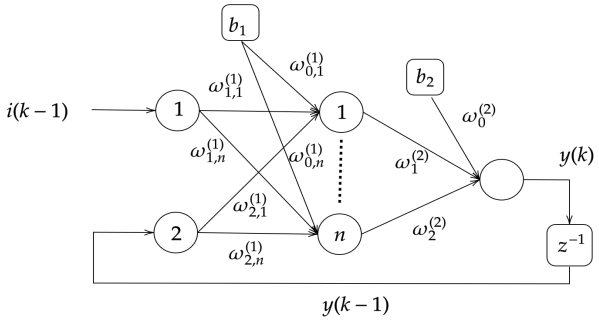

Fig. 3. Jordan network structure.

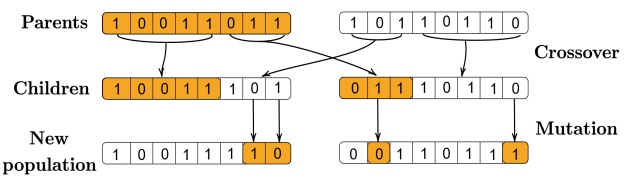

Fig. 4. Genetic operations

In this article, we propose a combination of RWS and TS selection operations to improve the selection of potential genes, a critical GA phase. Briefly speaking, The RWS method provides a higher chance for good genes to be selected, and this improves the exploitation and accelerates the convergence of the algorithm. However, since this approach is mainly based on the fitness value, premature convergence by selecting the same dominant genes is an open issue. On the other hand, the TS approach allows controlling the selection pressure, where smaller tournament sizes ensure more chances for weak genes to be selected unlike RWS. This feature retains the diversity of the search space, which in turn increases the possibility of converging to a global optimum at the expense of slower convergence. In this paper, both methods are used with random percentages at each iteration to increase both convergence speed and optimality by combining the advantages of both methods. Additionally, uniform crossover has been used with mutation based on Gaussian distribution (see Algorithm 1).

Algorithm 1 Proposed Genetic Algorithm

Require: Gen_{max}, N_p ▷ Generations, Population size
  $Pop \leftarrow N_p\ Parents$ ▷ Random population
  while $Generation < Gen_{max}$ **do**
    $Child \leftarrow emptyPop$ ▷ Create child population
    while $Child \leq full$ **do**
      $RWS \leftarrow \%_r$ ▷ Generate RWS percentage
      $TS \leftarrow \%_t$
      if $\%_r \geq \%_t$ **then**
        $Parent1 \leftarrow RWS(Pop)$ ▷ RWS Selection
        $Parent2 \leftarrow RWS(Pop)$
      else
        $Parent1 \leftarrow TS(Pop)$ ▷ TS Selection
        $Parent2 \leftarrow TS(Pop)$
      end if
      $Child1, 2 \leftarrow UCrossover(Parent1, Parent2)$
        ▷ Perform Uniform Crossover
      $Child1, 2 \leftarrow GMutation(Child1, Child2)$
        ▷ Perform Mutation
      $Fitness \leftarrow Evaluate(Child1, Child2)$
        ▷ Evaluate new offsprings
      $Offspring \leftarrow Child1, Child2$
    end while
    $Pop \leftarrow Offspring$ ▷ Replace Population
  end while
  $Solution \leftarrow Best\ fitness$ ▷ Save best solution

On the other hand, particle swarm optimization is a well-known algorithm for meta-heuristic optimization [30, 31], its classic algorithm is defined as follows:

$$\begin{cases} v_i(k+1) = \omega v_i(k) + c_1 r_1 (Pb_i(k) - x_i(k)) \\ \qquad\qquad + c_2 r_2 (Gb(k) - x_i(k)) \\ x_i(k+1) = x_i(k) + v_i(k+1) \end{cases} \tag{11}$$

The terms v_i and x_i define the velocity and the position of particle (i), a particle is a solution to the optimization problem. The terms ω, c_1, and c_2 are respectively known as inertia weight, cognitive, and social accelerations, while $r_{1,2} \in [0, 1]$ are just random constants. Parameters Pb and Gb are the best local and global positions, respectively. In the classic algorithm, ω and $c_{1,2}$ are constants, but in the improved version of this work, they are dynamic and change according to the following equations [7]:

$$\omega = \omega_{min} + \frac{\exp\left(\omega_{max} - \lambda_1(\omega_{max} + \omega_{min})\frac{g}{G}\right)}{\lambda_2} \tag{12}$$

$$\begin{cases} c_1(k+1) = c_1(k) + \alpha \\ c_2(k+1) = c_2(k) + \beta \\ \alpha = -\beta = 0.05 \quad \text{for} \quad \frac{g}{G} \leq 20\% \\ \alpha = -\beta = 0.02 \quad \text{for} \quad 20\% \leq \frac{g}{G} \leq 35\% \\ \alpha = -\beta = -0.035 \quad \text{for} \quad 35\% \leq \frac{g}{G} \leq 75\% \\ \alpha = -\beta = -0.0015 \quad \text{for} \quad \frac{g}{G} \geq 75\% \end{cases} \tag{13}$$

The terms ω_{min} and ω_{max} are the upper and lower bounds of the inertia weight and $\lambda_{1,2}$ are adjustable parameters to control the decrease from ω_{max} to ω_{min}. The terms g and G represent the actual and the last generations. The advantage of this improved version over the standard one is that it enhances the overall search capabilities of the PSO algorithm [7]. When ω decreases exponentially it accelerates the convergence towards the global best solution. Furthermore, increasing cognitive acceleration c_1 enhances the exploration phase where particles are pulled towards Pb, and increasing c_2 enhances the exploitation phase where particles converge towards Gb and vice versa. Compared to GA, PSO algorithms are a bit more intelligent as they incorporate memory by retaining knowledge of good solutions by all the particles as they share information in the swarm. In contrast, a GA would discard all the previous knowledge of the problem once it changes populations.

The proposed hybrid algorithm (see Fig. 5) runs offline and exploits the improved GA and the PSO algorithm iteratively. Briefly speaking, at each iteration of the algorithm, the solutions found by the GA and PSO algorithms are compared, and only the best-found solution is retained. At the next iteration stage, both algorithms will run with the best previously found solution. This strategy allows combining both search efforts of the GA and PSO algorithms towards finding the best solution to the problem. The interest in hybridizing these algorithms lies in the fact that it allows us to overcome the weak searching ability and slow convergence of the GA. In fact, when individuals are not selected in the GA algorithm, their information is completely lost. This is not the case with PSO, since it has memory. On the other hand, PSO algorithms do not have selection operators, which means the algorithm will likely run computations on unfit individuals. Simply put, combining GA with PSO retains the advantages of both algorithms, where the GA excels at reaching the global solution region, and the group search feature of the PSO algorithm boosts the search for exact optimal solutions.

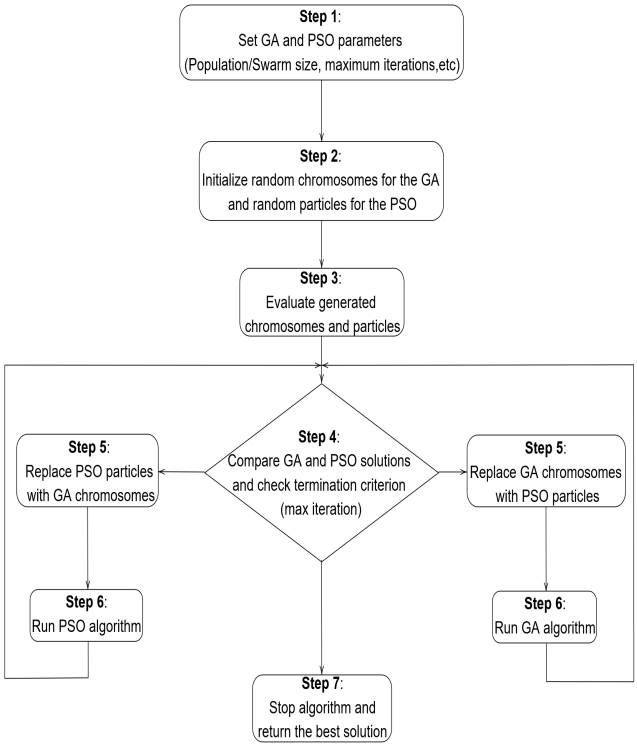

Fig. 5. Proposed hybrid GA-PSO algorithm.

4 RESULTS AND DISCUSSION

The proposed control strategy is tested using a Renault Zoe vehicle, whose behavior is simulated in `Matlab` using a high fidelity nonlinear dynamic model [24] with the Pacejka formula for the lateral tire forces [13]. Table 3 properly lists all model parameters.

4.1 Learning and Optimization Results

Since the cornering stiffness coefficients are learned from data, multiple driving scenarios are performed in Carsim [32] to collect data for training the neural network. The Jordan recurrent neural network consists of an input layer with 7 neurons, and two hidden layers with 8 and 5 hidden neurons, respectively. In addition to one output layer with two neurons corresponding to the cornering stiffness coefficients for front and rear wheels. All the hidden layers are activated with the sigmoid function, while the identity is used for the output layer. The model is developed in Keras-Tensorflow [33] and trained for

100 epochs with a batch size of 16. The data set contains around 10000 data points, which were split into 75% for training and 25% for validation. The adaptive movement estimation algorithm was used as the optimizer with a learning rate of $5e^{-4}$. Fig. 6 shows the training curve of the neural network, which proves the ability of the model to learn the data without over-fitting. The resulting validation loss value is as low as 0.002, while the training loss reached 0.005. The obtained R^2 scores on the training and the test data sets are 99.8% and 99.7%, respectively. Fig. 7 shows a comparison between the predicted values and the expected ones over a few data points of the test data set, which illustrates the accuracy of the model.

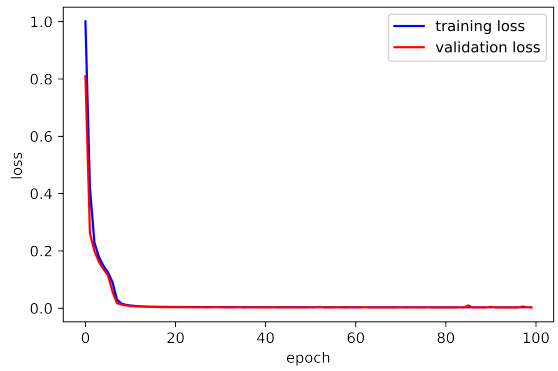

Fig. 6. Learning curve.

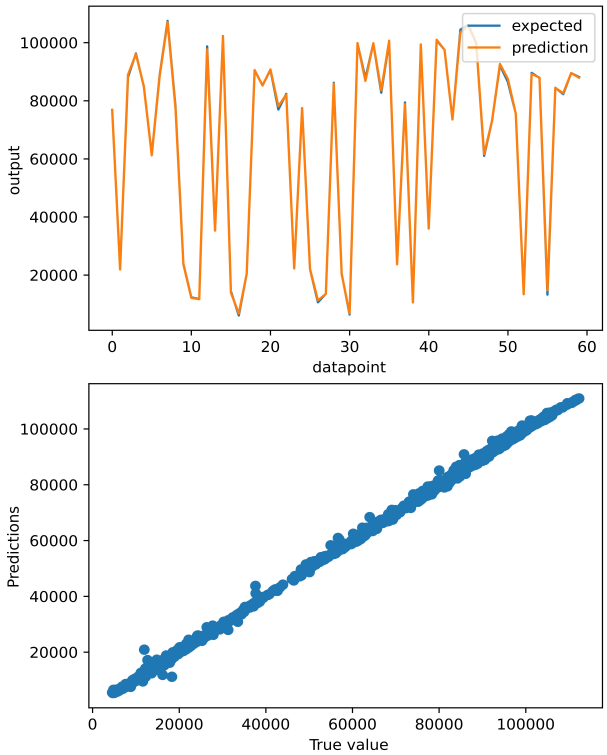

Fig. 7. Prediction model performance.

The different parameters of the proposed hybrid GA-PSO algorithm are tuned intuitively and iteratively until the desired performance is achieved, the algorithm is evaluated on a 5D sphere function $\left(f(x) = \sum_{i=1}^{5} x_i^2\right)$ as a benchmark test [34]. The resulting performance of the proposed GA-PSO algorithm over 100 iterations is compared to improved GA and improved PSO, respectively. Fig. 8 shows that the hybrid GA-PSO is indeed faster and able to further optimize the solutions. It managed to reach a minimum cost value of $4.24e^{-8}$ compared to $4.82e^{-6}$ and $2.95e^{-5}$ for the improved PSO and GA algorithms, respectively. Tables 1 and 2 list the parameters used in the GA-PSO algorithm for the optimization of the LPV-MPC controller. The fitness function, in this case, was selected as RMSE for longitudinal velocity, lateral position, and heading tracking. The GA-PSO optimization managed to achieve a minimum RMSE score of 0.0069, 0.0191, and 0.0212 for the position, heading, and velocity tracking, respectively. The optimized weighting matrices are as follows:
$Q = \text{diag}(50,\ 1e^{-5},\ 0.01,\ 47.43,\ 1e^{-3})^T$,
$R = \text{diag}(0.003,\ 1e^{-4})^T$.

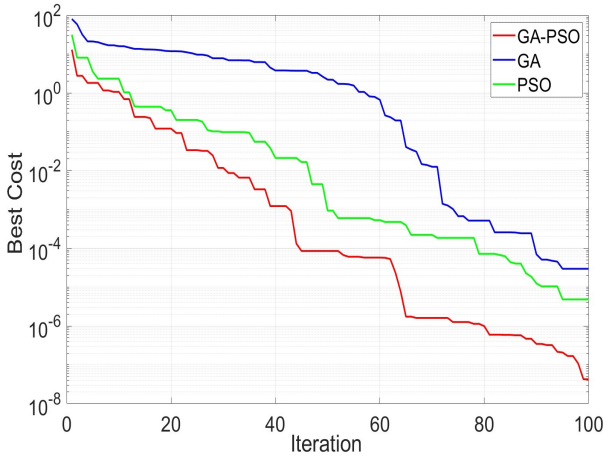

Fig. 8. Performance of the improved hybrid GA-PSO.

Table 1
GA parameters

Parameter	Name	Value
Gen	Generation	15
N_P	Size of population	25
O_p	Percentage of offsprings	0.8
β	Selection pressure	0.75
μ_r	Mutation rate	0.3
σ	Mutation variance	0.15

4.2 Control Results

The proposed controller is coded in `Yalmip` platform and solved using `Gurobi` solver. The algorithm runs at 95Hz

Table 2
PSO hyper-parameters.

Parameter	Interpretation	Value
N	Generation	15
N_{Pop}	Swarm particles	25
ω_{max}	Maximum inertia weight	0.99
ω_{min}	Minimum inertia weight	0.1
c_{1i}	Initial cognitive acceleration	2
c_{2i}	Initial social acceleration	2
λ_1	Constant	30
λ_2	Constant	3

on a Ryzen7 laptop with 32gb of RAM. The LPV-MPC is implemented and evaluated in `Matlab` simulations using the high fidelity nonlinear dynamic model [24] with the Pacejka formula for the lateral tire forces [13]. Table 3 presents the MPC parameters. The evaluation is performed for a double lane change trajectory and speed profile (see Fig. 9,10). Overall, the proposed controller performs the double lane change maneuver very well with minimal tracking errors. Similarly, the controller is further tested on a more challenging general trajectory and speed profile under wind disturbances varying between 20 and 50 m/s, (see Fig. 11,12). Furthermore, it is compared to another MPC based on the linear bicycle model as introduced in [9], which is used in coordination with an optimized PSO-PID to address the combined longitudinal and lateral dynamics. We denote this controller LMPC for linear MPC. Such a comparison shows that the proposed controller handles both lateral and longitudinal dynamics and outperforms the decoupled control strategy of [9], which dedicates two optimized controllers for the same task. The LMPC was tested on the same trajectory with the same speed profile and wind disturbance. Fig. 11 shows the velocity profile variying between 5 and 25 m/s. It can be seen that the LPV-MPC is slightly more accurate in speed tracking. The MSE was evaluated at 0.02 compared to 0.19 for the PSO-PID, whose parameters were already optimized. In addition, PSO-PID was found to be more aggressive as it cannot handle constraints.

The results in Fig. 12 show that the proposed LPV-MPC performed much better than LMPC in terms of tracking accuracy. The obtained MSE score was as low as 0.007 compared to 0.32 for LMPC, this means the LPV-MPC tracking is almost ideal compared to its rival. The zoomed regions of the figure further illustrate the big difference in tracking accuracy, this is partly due to the different models used to develop the MPC controller.

The corresponding steering and acceleration controls and the lateral velocity of the vehicle are shown in Fig. 13. The predicted cornering stiffness coefficients for the

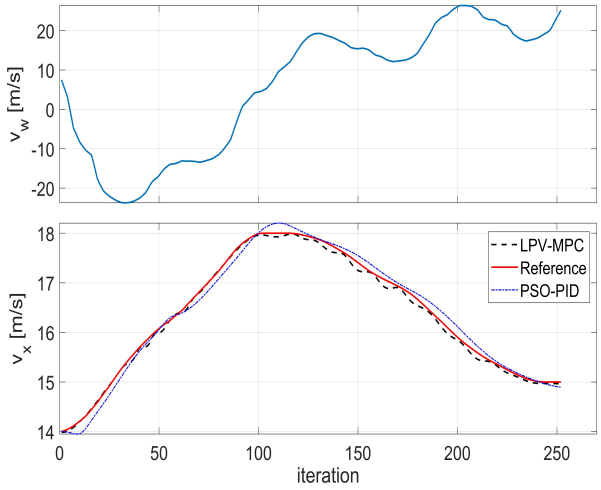

Fig. 9. Wind velocity and speed tracking for double lane change.

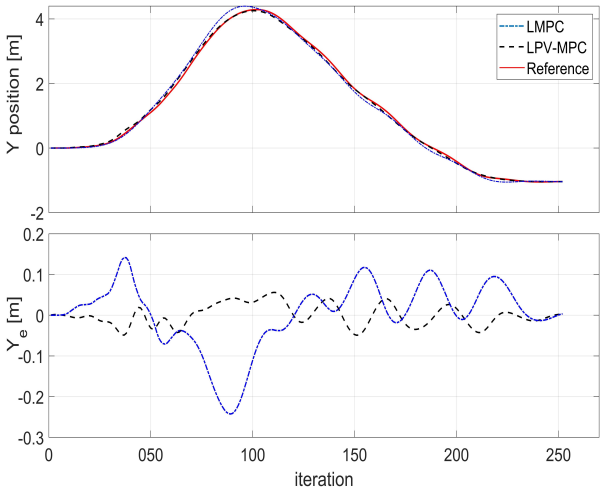

Fig. 10. Trajectory tracking for double lane change.

Table 3
MPC and model parameters

Parameter	Value	Parameter	Value
m	1575 (kg)	C_d	0.29
I_z	2875 $(kg.m^2)$	A	1.6 (m^2)
l_f	1.2 (m)	$y_{e_{max/min}}$	0.3 (m)
l_r	1.6 (m)	$u_{max/min}$	$\pm\frac{\pi}{6}$ (rad)
ρ	1.225 (kgm^3)	$\Delta u_{max/min}$	$\pm\frac{\pi}{12}$ (rad)
μ	0.82	N_p	10
g	9.81 (m/s^2)	T_s	0.033 s

tested trajectory are reported in Fig. 14, and Fig. 15 shows the corresponding tracking errors for longitudinal velocity, heading, and lateral position, respectively.

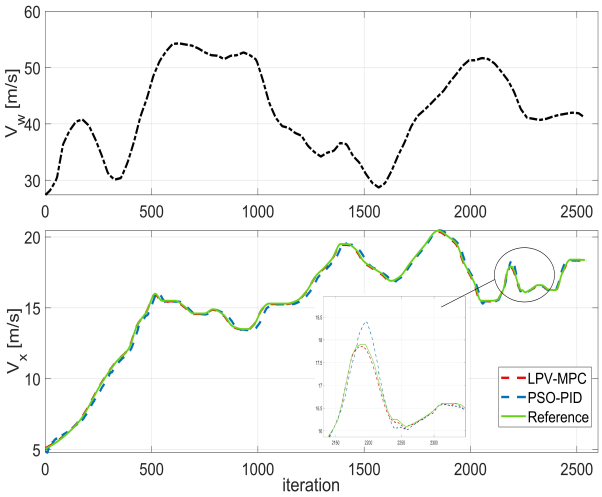

Fig. 11. Wind velocity and speed tracking performance.

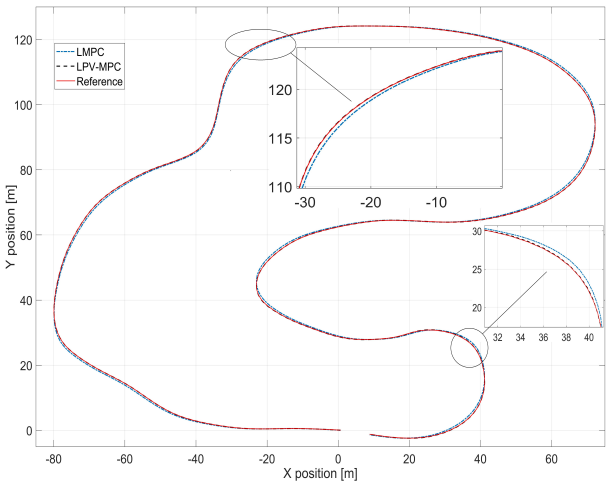

Fig. 12. Trajectory tracking.

As seen in the figure, the velocity tracking error does not exceed 0.095 m/s, and the maximum heading and position tracking errors are kept below 4.5° and 2.3 cm, respectively. Moreover, the execution time of the LPV-MPC is very suitable for real-time applications as observed in Fig. 16 with a mean computation time of (0.011 s).

5 CONCLUSIONS

This article addressed the coupled control task in autonomous driving with an LPV-MPC controller. The developed controller is capable of simultaneously controlling the lateral and longitudinal dynamics. A machine learning approach has been introduced to predict the model's tire cornering stiffness coefficients online, using only measurable parameters. This approach adapts the LPV-MPC prediction model for more accurate predictions. For tuning and optimizing the

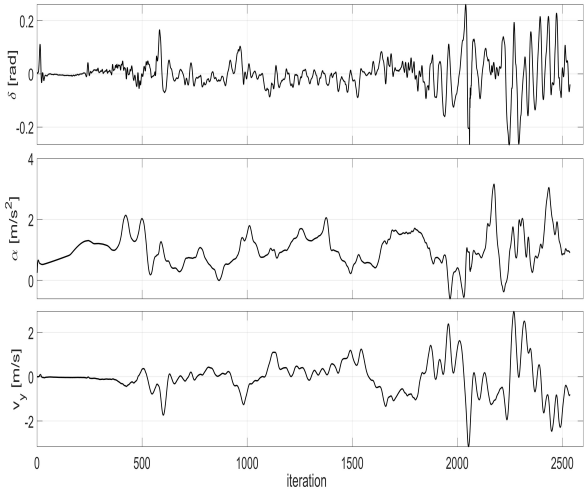

Fig. 13. Steering, acceleration, and lateral velocity signals.

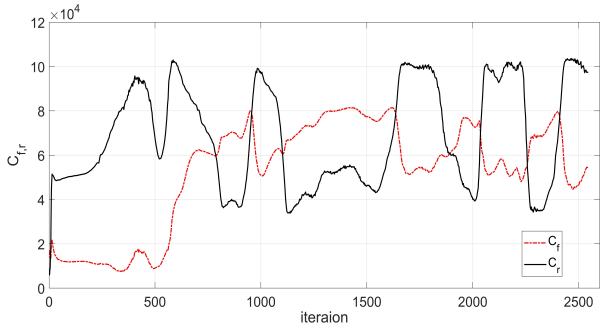

Fig. 14. Cornering stiffness coefficients.

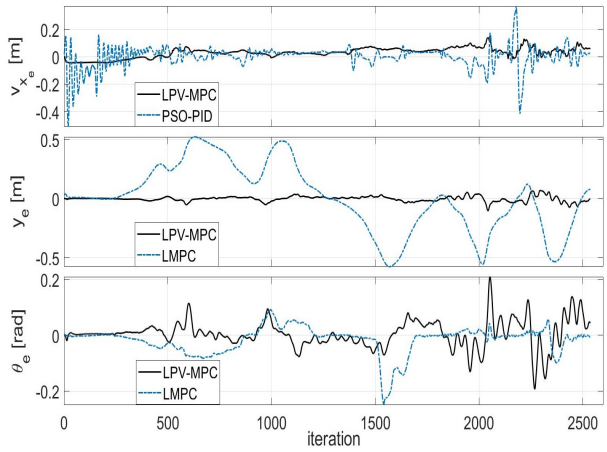

Fig. 15. Tracking performance.

proposed controller, an improved hybrid GA-PSO algorithm has been proposed. The developed controller has been evaluated on a challenging track and compared to another variant of LPV-MPC. The obtained results showed superior performance of the proposed controller, which ensures high speed and trajectory tracking accuracy. Future works shall target online learning for more advanced autonomous driving applications.

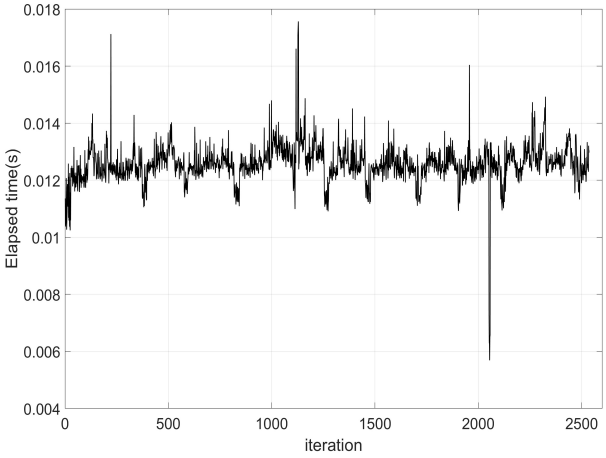

Fig. 16. MPC computation time.

6 ACKNOWLEDGEMENTS

The authors would like to thank the University of Paris-Saclay for the financial support provided to conduct this research.

7 AUTHORS' CONTRIBUTIONS

Yassine Kebbati: Conceptualization, Methodology, Writing original draft, Reviewing and Editing; **Naima Ait-Oufroukh**: Review and Validation; **Dalil Ichalal**: Review and Supervision; **Vincent Vigneron**: Supervision, Review and Validation.

8 CONFLICTS of INTEREST

The authors declare having no conflict of interest for the publication of this article.

References

[1] S. Xu, H. Peng, Z. Song, K. Chen, and Y. Tang, "Accurate and smooth speed control for an autonomous vehicle," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1976–1982, IEEE, 2018.

[2] Y. Kebbati, N. Ait-Oufroukh, V. Vigneron, D. Ichalal, and D. Gruyer, "Optimized self-adaptive pid speed control for autonomous vehicles," in *2021 26th International Conference on Automation and Computing (ICAC)*, pp. 1–6, IEEE, 2021.

[3] G. Han, W. Fu, W. Wang, and Z. Wu, "The lateral tracking control for the intelligent vehicle based on adaptive pid neural network," *Sensors*, vol. 17, no. 6, 2017.

[4] H. Guo, D. Cao, H. Chen, Z. Sun, and Y. Hu, "Model predictive path following control for autonomous cars considering a measurable disturbance: Implementation, testing, and verification," *Mechanical Systems and Signal Processing*, vol. 118, pp. 41–60, 2019.

[5] M. S. Akbari, A. A. Safavi, N. Vafamand, T. Dragičević, and J. Rodriguez, "Fuzzy mamdani-based model predictive load frequency control," in *2020 IEEE 11th International Symposium on Power Electronics for Distributed Generation Systems (PEDG)*, pp. 7–12, IEEE, 2020.

[6] M. Corno, G. Panzani, F. Roselli, M. Giorelli, D. Azzolini, and S. M. Savaresi, "An LPV Approach to Autonomous Vehicle Path Tracking in the Presence of Steering Actuation Nonlinearities," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 4, pp. 1766–1774, 2020.

[7] Y. Kebbati, V. Puig, N. Ait-Oufroukh, V. Vigneron, and D. Ichalal, "Optimized adaptive mpc for lateral control of autonomous vehicles," in *2021 9th International Conference on Control, Mechatronics and Automation (ICCMA)*, pp. 95–103, IEEE, 2021.

[8] Y. Kebbati, N. Ait-Oufroukh, V. Vigneron, and D. Ichalal, "Neural network and anfis based auto-adaptive mpc for path tracking in autonomous vehicles," in *2021 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, vol. 1, pp. 1–6, IEEE, 2021.

[9] Y. Kebbati, N. Ait-Oufroukh, V. Vigneron, and D. Ichalal, "Coordinated pso-pid based longitudinal control with lpv-mpc based lateral control for autonomous vehicles," in *2022 European Control Conference (ECC)*, pp. 518–523, IEEE, 2022.

[10] Q. Yao and Y. Tian, "A model predictive controller with longitudinal speed compensation for autonomous vehicle path tracking," *Applied Sciences (Switzerland)*, vol. 9, no. 22, 2019.

[11] H. Wang, B. Liu, X. Ping, and Q. An, "Path Tracking Control for Autonomous Vehicles Based on an Improved MPC," *IEEE Access*, vol. 7, pp. 161064–161073, 2019.

[12] S. Li, Z. Li, B. Zhang, S. Zheng, X. Lu, and Z. Yu, "Path tracking for autonomous vehicles based on nonlinear model: Predictive control method," *SAE Technical Papers*, vol. 2019-April, no. April, pp. 1–7, 2019.

[13] H. B. Pacejka, "Vehicle System Dynamics : International Journal of Vehicle Mechanics and Mobility," *International Journal of Vehicle Mechanics and Mobility*, no. August 2012, pp. 37–41, 2008.

[14] Y. Kebbati, N. Ait-Oufroukh, V. Puig, D. Ichalal, and V. Vigneron, "Autonomous driving using ga-optimized neural network based adaptive lpv-mpc controller," in *2022 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, pp. 1–6, IEEE, 2022.

[15] E. Alcala, O. Sename, V. Puig, and J. Quevedo, "Ts-mpc for autonomous vehicle using a learning approach," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15110–15115, 2020.

[16] J. V. Frasch, A. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear mpc algorithm for real-time obstacle avoidance of ground vehicles," in *2013 European Control Conference (ECC)*, pp. 4136–4141, 2013.

[17] 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, no. 3, pp. 566–580, 2007.

[18] M. Rick, J. Clemens, L. Sommer, A. Folkers, K. Schill, and C. Büskens, "Autonomous Driving Based on Nonlinear Model Predictive Control and Multi-Sensor Fusion," *IFAC-PapersOnLine*, vol. 52, no. 8, pp. 458–473, 2019.

[19] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-Based Model Predictive Control for Autonomous Racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, 2019.

[20] Y. Kebbati, N. Ait-Oufroukh, R. Andreas, D. Ichalal, and V. Vigneron, "Learning-based model predictive control with moving horizon state estimation for autonomous racing," *International Journal of Control*, pp. 1–9, 2024.

[21] A. Tătulea-Codrean, T. Mariani, and S. Engell, "Design and simulation of a machine-learning and model predictive control approach to autonomous race driving for the f1/10 platform," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6031–6036, 2020.

[22] D. Kloeser, T. Schoels, T. Sartor, A. Zanelli, G. Prison, and M. Diehl, "Nmpc for racing using a singularity-free path-parametric model with obstacle avoidance," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 14324–14329, 2020.

[23] Y. Kebbati, N. Ait-Oufroukh, D. Ichalal, and V. Vigneron, "Lateral control for autonomous wheeled vehicles: A technical review," *Asian Journal of Control*, 2022.

[24] E. Alcalá, V. Puig, J. Quevedo, and U. Rosolia, "Autonomous racing using Linear Parameter Varying-Model Predictive Control (LPV-MPC)," *Control Engineering Practice*, vol. 95, no. March 2019, p. 104270, 2020.

[25] D. Schramm, M. Hiller, and R. Bardini, "Vehicle dynamics," *Modeling and Simulation. Berlin, Heidelberg*, vol. 151, 2014.

[26] F. Rußwurm, W. Esterhuizen, K. Worthmann, and S. Streif, "On MPC without terminal conditions for dynamic non-holonomic robots," *IFAC-PapersOnLine*, vol. 54, no. 6, pp. 133–138, 2021.

[27] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[28] Y. Song, F. Wang, and X. Chen, "An improved genetic algorithm for numerical function optimization," *Applied Intelligence*, vol. 49, no. 5, pp. 1880–1902, 2019.

[29] S. L. Yadav and A. Sohal, "Comparative study of different selection techniques in genetic algorithm," *International Journal of Engineering, Science and Mathematics*, vol. 6, no. 3, pp. 174–180, 2017.

[30] B. Song, Z. Wang, and L. Zou, "An improved pso algorithm for smooth path planning of mobile robots using continuous high-degree bezier curve," *Applied Soft Computing*, vol. 100, p. 106960, 2021.

[31] Y. Xie and W. Zhang, "A novel tuning method for pid controller based on improved pso algorithm for unstable plants with time delay," in *2019 Chinese Control Conference (CCC)*, pp. 4270–4275, IEEE, 2019.

[32] R. F. Benekohal and J. Treiterer, "Carsim: Car-following model for simulation of traffic in normal and stop-and-go conditions," *Transportation research record*, vol. 1194, pp. 99–111, 1988.

[33] N. Ketkar and N. Ketkar, "Introduction to keras," *Deep learning with python: a hands-on introduction*, pp. 97–111, 2017.

[34] Y. Kaya, M. Uyar, *et al.*, "A novel crossover operator for genetic algorithms: ring crossover," *arXiv preprint arXiv:1105.0355*, 2011.