Autonomous driving using GA-optimized neural network based adaptive LPV-MPC controller

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

Autonomous driving using GA-optimized neural network based adaptive LPV-MPC controller

1st Yassine Kebbati
IBISC-EA4526
University of Paris-Saclay
Evry, France
yassine.kebbati@univ-evry.fr

2nd Naima Ait-Oufroukh
IBISC-EA4526
University of Paris-Saclay
Evry, France
naima.aitoufroukh@univ-evry.fr

3rd Vicenc Puig
CS2AC
University of Catalonia
Barcelona, Spain
vicenc.puig@upc.edu

4th Dalil Ichalal
IBISC-EA4526
University of Paris-Saclay
Evry, France
dalil.ichalal@univ-evry.fr

5th Vincent Vigneron
IBISC-EA4526
University of Paris-Saclay
Evry, France
vincent.vigneron@univ-evry.fr

***Abstract*— Autonomous vehicles are complex systems that operate in dynamic environments, where automated driving seeks to control the coupled longitudinal and lateral vehicle dynamics to follow a certain behaviour. Model predictive control is one of the most promising tools for this type of application due to its optimal performance and ability to handle input and output constraints. This paper addresses autonomous driving by introducing an adaptive linear parameter varying model predictive controller (LPV-MPC), whose prediction model is adapted online by a neural network. Moreover, the controller's cost function is optimized by an improved Genetic Algorithm. The proposed controller is evaluated on a challenging track subject to variable wind disturbances.**

***Index Terms*— Autonomous Driving, LPV-Systems, Model Predictive Control, Neural Networks, Genetic Algorithms.**

I. Introduction

Scientists and engineers in top automobile companies have been striving to achieve full driving autonomy by replacing human drivers with automatic control systems. Consequently, autonomous vehicles are emerging as a top technology aiming at improving traffic safety and enhancing mobility. The transition from manual to automatic driving is further accelerated by the huge leap in artificial intelligence and information processing technologies. Such a technological shift can also boost human productivity, in the sense that time spent on driving can be used for other productive tasks instead. Autonomous driving, being a multidisciplinary field, involves sensing, perception, decision making and planning. In addition, motion control is among the most important tasks for achieving full autonomy. The latter can be divided into longitudinal control in charge of speed tracking and lateral control which handles the steering.

Literature research is divided into two categories; those who address the longitudinal and lateral control separately, and those who couple both tasks together. For instance, Xu *et al.* developed an optimal preview controller for speed regulation that integrates road slope, speed profile and vehicle dynamics [1]. Kebbati *et al.* [2] addressed speed control by designing a self-adaptive PID controller based on neural networks and genetic algorithms. On the other hand, Han *et al.* [3] designed an adaptive neural network PID controller for path tracking. They applied it to a second order vehicle model whose parameters were estimated by a forgetting factor least square algorithm. LPV H_∞ was developed for high-speed driving and evasive maneuvers by Corno *et al.* [4]. The design was based on the lateral error and look-ahead distance of the vehicle to ensure better robustness and account for actuator nonlinearities under low speeds. In [5], authors developed a model predictive controller (MPC) for lateral control and used fuzzy inference systems (FIS) to tune its weighting matrices. Guo *et al.* [6] developed a path-tracking MPC controller that considers the varying road conditions and small-angle assumptions as a form of measurable disturbance and solved the control problem using the differential evolution algorithm. Authors of [7] worked on adaptive MPC for path tracking that optimizes the controller tuning with an improved PSO algorithm [8]. Online controller adaptation was achieved by a lookup table approach, but this technique cannot account for all possible cases despite the good results that were obtained. However, the same authors improved their approach in [9] by replacing the lookup table method with neural networks and adaptive neuro-fuzzy inference systems to generalize the adaptation beyond the data in the lookup table. Although significant tracking improvements were achieved, this approach still requires long offline optimizations.

However, decoupling vehicle dynamics leads to inaccuracies and reduces controller performance, coupled dynamics are more suitable for highly dynamic maneuvers and fast driving such as in racing applications. To address the mixed lateral and longitudinal control, Alcala *et al.* developed in [10] a solution for trajectory tracking. Their control strategy was divided into a cascade control scheme, with an internal layer for controlling vehicle dynamics and an external one for vehicle kinematics. The same authors developed in [11] a Takagi-Sugeno based MPC (TS-MPC) for autonomous driving. They used a data-driven approach to learn a Takagi-Sugeno representation of the vehicle dynamics, which was used in MPC with Moving Horizon Estimator (MHE) for achieving coupled longitudinal

and lateral control. Paper [12] proposed a coordinated lateral and longitudinal control strategy using LPV-MPC for lateral control with PSO-PID for speed regulation, the strategy includes exponential weight to the MPC cost function to improve tracking performance. The use of nonlinear model predictive control (NMPC) was proposed in [13] where the authors aimed at exploring a parking lot autonomously and performing the parking maneuver. Finally, paper [14] introduced an online learning MPC controller for autonomous racing by learning the model errors online using Gaussian process regression.

This paper contributes to the above mentioned literature by developing an enhanced controller for the coupled longitudinal and lateral control. The contributions of this work are threefold: First, an adaptive LPV-MPC is developed for realizing autonomous driving. Second, an improved genetic algorithm is proposed for tuning the weighting matrices of the cost function to achieve optimal control actions. Third, a deep neural network is developed and trained to learn the tire lateral dynamics by predicting the cornering stiffness coefficients from measurable parameters such as velocities and accelerations. The paper is organized as follows: Section II presents the modeling of the vehicle dynamics. Section III details the design of the LPV-MPC controller, the controller adaptation approach using machine learning and the controller optimization via the improved GA version. Evaluation results of the learning approach, optimization and control are presented and analyzed in section IV. Finally, section V provides summarized conclusions and gives directions for future work.

II. Vehicle Modeling

The commonly used bicycle dynamic model [15], [16] is adopted in this paper as it is accurate for control design whilst being simple for real-time implementation. In this model, the two front wheels similar to the rear ones are lumped to form a single track representation (see Fig. 1). The lateral dynamics are governed by the tire lateral forces which are a function of the slip angles. The full model accounts for longitudinal, lateral and yaw dynamics (1) as well as tire forces (2) and heading and lateral position errors (3):

$$\begin{cases} \dot{v}_x &= \alpha_x + \omega v_y - \frac{1}{m}(F_{yf}\sin\delta + F_d) \\ \dot{v}_y &= \frac{1}{m}(F_{yf}\cos\delta + F_{yr}) - \omega v_x \\ \dot{\omega} &= \frac{1}{I}(F_{yf} l_f \cos\delta - F_{yr} l_r) \end{cases} \tag{1}$$

$$\begin{cases} Fyf &= C_f \alpha_f \\ Fyr &= C_r \alpha_r \\ F_d &= \mu m g + \frac{1}{2}\rho C_d \Lambda v_x^2 \end{cases} \tag{2}$$

$$\begin{cases} \dot{y}_e &= v_x \sin\theta_e + v_y \cos\theta_e \\ \dot{\theta}_e &= \omega - \frac{v_x \cos\theta_e - v_y \sin\theta_e}{1 - y_e k} \end{cases} \tag{3}$$

The parameters v_x, v_y and ω represent the linear longitudinal and lateral velocities and yaw rate in the body frame. $F_{y(f,r)}$ are the tire lateral forces of the front and rear wheels respectively. F_d is the total drag force where C_d and Λ represent the drag coefficient and the vehicle cross sectional area. y_e and θ_e represent the lateral position error and the heading error where k is the road curvature. I and m are the inertia and the mass of the vehicle and $l_{(f,r)}$ are the distances

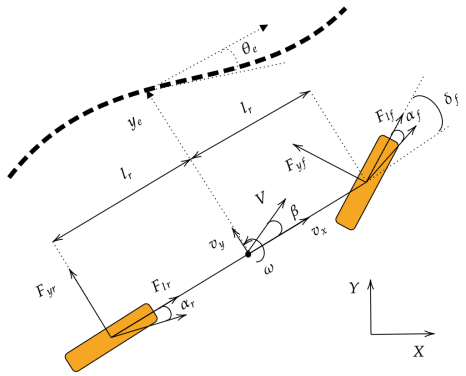

Fig. 1: Bicycle dynamic model with tracking error.

between the vehicle centre of gravity and the front and rear wheel axles, respectively. α_x and δ are the acceleration and steering controls and μ and g represent the friction coefficient and the gravity pull. Finally, $C_{(f,r)}$ represent the respective front and rear tire cornering stiffness coefficients, and $\alpha_{(f,r)}$ are the slip angles of the front and rear wheels with ϵ being an additional term to avoid singularities in the model, these are respectively given by:

$$\begin{cases} \alpha_f &= \delta - \tan^{-1}\left(\frac{v_y}{v_x+\epsilon} - \frac{l_f \omega}{v_x+\epsilon}\right) \\ \alpha_r &= -\tan^{-1}\left(\frac{v_y}{v_x+\epsilon} + \frac{l_r \omega}{v_x+\epsilon}\right) \end{cases} \tag{4}$$

The full model can be summarized as a non-linear function of the state vector (x), input vector (u) and the road curvature (k) as follows:

$$\dot{x} = f(x, u, k) \tag{5}$$

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

III. Controller Design

The control task is achieved based on LPV-MPC approach [15], where the model presented in Section II is reformulated in an LPV form which allows to capture model nonlinearities as in the nonlinear model without the high computation burden. The model is then transformed into a state space representation where the state and control matrices are functions that depend on a scheduling vector of varying parameters. Embedding linear varying parameters into these matrices allows to capture the nonlinearities of the full model that provides a simple but accurate model for control design. Generally speaking, LPV systems are a class of linear systems whose parameters are functions of external or internal scheduling signals. The LPV state space formulation of the system is given by (6) based on the scheduling vector $\psi = [\delta\ v_x\ v_y\ \theta_e\ y_e\ k]^T$.

$$\dot{x} = A(\psi)x + B(\psi)u \tag{6}$$

According to [15], [17], the state matrix $A(\psi)$ and control matrix $B(\psi)$ can be derived as the following:

$$A(\psi) = \begin{bmatrix} A_{11} & A_{12} & A_{13} & 0 & 0 \\ 0 & A_{22} & A_{23} & 0 & 0 \\ 0 & A_{32} & A_{33} & 0 & 0 \\ A_{41} & A_{42} & 0 & 0 & 0 \\ A_{51} & A_{52} & 1 & 0 & 0 \end{bmatrix}, \tag{7}$$

$$B(\psi) = \begin{bmatrix} B_{11} & 1 \\ B_{21} & 0 \\ B_{31} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \tag{8}$$

where the terms are given by:

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

Model predictive control uses the plant model to predict its behaviour over a prediction horizon N_p. Then, it generates an optimal control sequence by solving a constrained convex optimization problem. The receding horizon principle is then applied and only the first term of the optimal control sequence is used. The LPV model, discretized with T_s sampling time, is used as the MPC prediction model. Hence, at each iteration the scheduling vector is used to instantiate the LPV model. The parameters of the scheduling vector can be obtained from sensors, planners or previous MPC predictions. In this regard, the MPC problem can be formulated as the following constrained quadratic optimization:

$$\begin{aligned} \min_{\Delta U_k} \; J_k = & \sum_{i=0}^{N_p-1} \Big((r_{k+i} - x_{k+i})^T Q (r_{k+i} - x_{k+i}) + \\ & \Delta u_{k+i} R \Delta u_{k+i} \Big) + x_{k+N_p}^T Q x_{k+N_p} \\ s.t: & \\ & x_{k+i+1} = x_{k+i} + A(\psi_{k+i}) x_{k+i} + B(\psi_{k+i}) u_{k+i} dt \\ & u_{k+i} = u_{k+i-1} + \Delta u_{k+i} \\ & \Delta u_{min} \leq \Delta u_k \leq \Delta u_{max} \\ & u_{min} \leq u_k \leq u_{max} \\ & x_{min} \leq x_k \leq x_{max} \end{aligned} \tag{9}$$

The terms x, u and N_p are the previously defined state vector, control vector and prediction horizon. $Q \in \mathbb{R}^{5\times5}$ and $R \in \mathbb{R}^{2\times2}$ are semi-positive definite weighting matrices that penalise the states and the control effort. r_{k+i} is the reference vector and the terms $[u_{min}, u_{max}]$, $[\Delta u_{min}, \Delta u_{max}]$ and $[x_{min}, x_{max}]$ are the upper and lower bounds on the control actions, control increments and states, respectively.

A. Controller adaptation with neural network

The majority of research in the literature consider a linearized tire model, where the relation between tire force and slip angle is governed by a constant called cornering stiffness coefficient. However, this is only valid for relatively small slip angles and not during fast and challenging maneuvers. To remedy this, we use machine learning to predict the cornering stiffness coefficient online using measurable parameters from the vehicle dynamics. We assume these parameters (namely, 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. Using this approach (see Fig. 2), the prediction model of the LPV-MPC is adapted online to improve the prediction capability and precision.

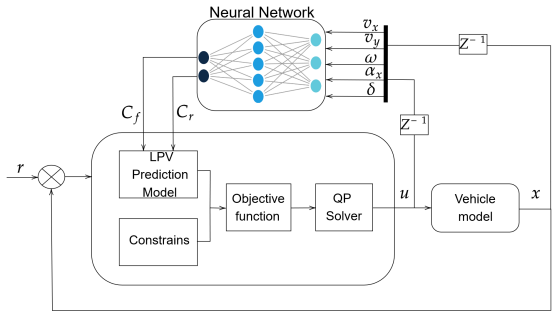

Fig. 2: Adaptive LPV-MPC approach.

B. Controller tuning with improved GA Algorithm

Manually tuning the MPC is time consuming and may not result in optimal performance. Thus, to optimize the designed LPV-MPC, we propose an improved Genetic Algorithm (GA) 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 biological evolution theory. They can find the optimum of an objective function even if it is not continuous or differentiable [18]. The algorithm initializes a population set of possible encoded solutions named chromosomes which are genetically enhanced over iterations and evaluated by a fitness function to determine their optimality. The GA operations are the selection, the crossover and the mutation processes, which control the search capability and the quality of the solutions. These operations consist of different functions which influence the algorithm at different degrees [18]. The selection process chooses the best chromosomes to be enhanced by the crossover and mutation operations. 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. 3.

To improve the GA performance, researchers have essentially tried to enhance the genetic operations. For instance, the most common selection operations are the roulette wheel (RWS) and tournament (TS) [18], [19]. Similarly, crossover versions include single/multi-point, uniform and shuffle crossover, while in mutation, one finds swap, inversion and random resetting. Details about these methods and GA can be found in [20]. Rather than improving the operators themselves, we combine RWS and TS selection operations to improve the selection of potential genes which is a critical

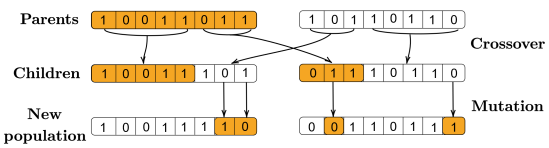

Fig. 3: Genetic operations

phase of the GA. In particular, RWS strategy provides a higher chance for the good genes to be selected, which enhances exploitation and accelerates the convergence of the algorithm. However, RWS is mainly based on the fitness value, which makes it prone to premature convergence by possibly selecting the same dominant genes every time. On the other hand, the TS method allows to control the selection pressure. A smaller tournament size ensures more chances for weak genes to be selected unlike RWS. This feature retains diversity of the search space and 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. Furthermore, 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

IV. Results and Discussion

For testing the proposed approach, a Renault Zoe vehicle is used. The behaviour of this vehicle is simulated in `Matlab` using a high fidelity nonlinear dynamic model [15], with the Pacejka formula for the lateral tire forces [21]. Model parameters are presented in Table II.

A. Learning and optimization results

As discussed in Section III-A, the cornering stiffness coefficient is learned from data. Carsim is used to perform multiple driving scenarios and collect data for training. Then, a deep neural network consisting of an input layer with 5 neurons, four hidden layers with 16, 28, 16 and 9 hidden neurons respectively, and an output layer with two neurons corresponding to the rear and front wheel cornering stiffness coefficients. The Sigmoid function is used for activation in all the layers except the output layer which uses Identity since this is a regression task. The model is built using Keras-Tensorflow and trained for 2500 epochs with a batch-size of 64. The whole data-set consists of 10752 data-points which were split into 75% for training and 25% for validation. Adam was chosen as the optimizer with a learning rate of $5{\times}10^{-4}$. The training curve of the neural network in Fig. 4 shows that the model is able to learn the data without over-fitting. The final validation loss value is minimized to 0.226, while the training loss reached 0.196. The evaluation of the model on the training and the test data-sets achieved an R2 score of 0.88 and 0.78, respectively. The proposed GA algorithm is tested on a 5D sphere function $\left(f(x) = \sum_{i=1}^{5} x_i^2\right)$ as a benchmark test [22], and the resulting performance over 100 iterations is compared to GA with either RWS or TS methods separately. Fig. 6 shows that the improved version is indeed faster and able to find more optimal solutions, reaching a cost value of 25.029 compared to 27.725 and 28.722 for the GA with RWS and TS respectively. Table I lists the parameters used in the GA algorithm for optimizing the weighting matrices of the LPV-MPC cost function. The fitness function is chosen as

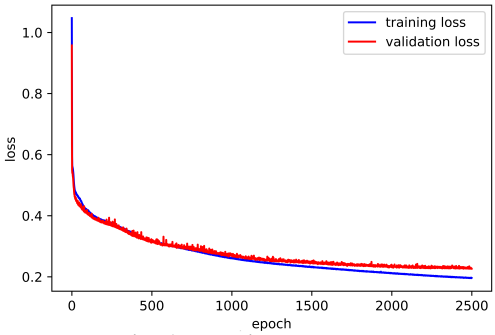

Fig. 4: Learning curve.

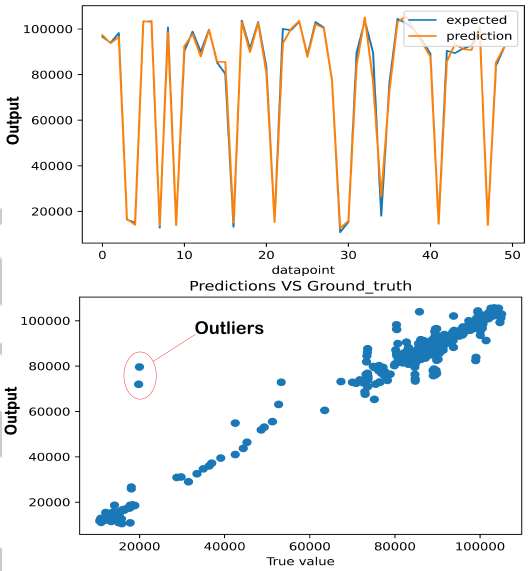

Fig. 5: Prediction model performance.

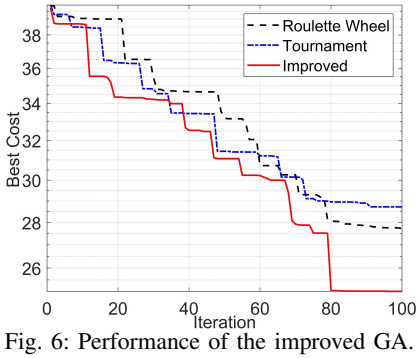

Fig. 6: Performance of the improved GA.

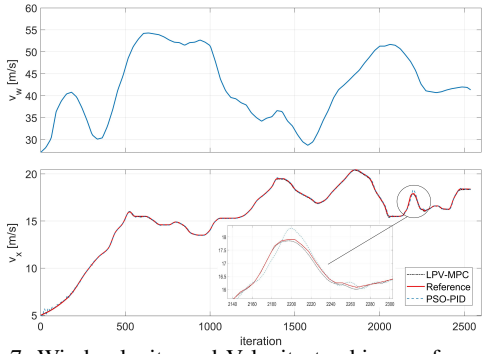

Fig. 7: Wind velocity and Velocity tracking performance.

the root mean squared error (RMSE) for longitudinal velocity, heading and lateral position tracking. The GA optimization achieved minimum RMSE scores of 0.0217, 0.0465 and 0.101 for the position, heading and velocity tracking, respectively. The resulting weighting matrices are as follows:
$Q = \text{diag}(0.008,\ 0.0007,\ 0.0133,\ 3,\ 0.021,\ 5)^T$,
$R = \text{diag}(0.0337,\ 0.0117)^T$.

B. Control results

The LPV-MPC is coded in `Yalmip` and solved with `Gurobi`, and its corresponding parameters are listed in Table II. The algorithm runs at 95Hz on a legion 5 pro with $3.2Ghz$ Ryzen7 $5800H$ and $32gb$ of RAM. The proposed controller is evaluated on a challenging trajectory and speed profile subject to wind disturbances varying between 25 and 50 m/s (see Fig. 7). Furthermore, it is compared to another LPV-MPC based on the linear bicycle model [12]. The latter, denoted LMPC, is used in a coordinated approach with an optimised PSO-PID to address longitudinal and lateral dynamics which were tested for the same trajectory, speed profile and wind disturbance. Such comparison shows the significance of accurate modelling and of the coupling of lateral and longitudinal dynamics for controller design. Fig. 7 shows the velocity profile varying between 5 and 21 m/s with challenging accelerations. The LPV-MPC outperforms the PSO-PID and is more accurate in speed tracking where its MSE is evaluated at 0.0476 compared to 0.187 for the PSO-PID whose parameters were already optimized. The better performance of LPV-MPC is achieved thanks to the dynamic coupling and the predictive feature of MPC which compensates for tracking errors beforehand unlike PID control which merely reacts to observed errors. In addition, PSO-PID is more aggressive as it cannot handle constraints. On the other hand, the results in Fig. 8 show that the proposed LPV-MPC is indeed superior to LMPC in terms of lateral tracking accuracy. In fact, its MSE score is as low as

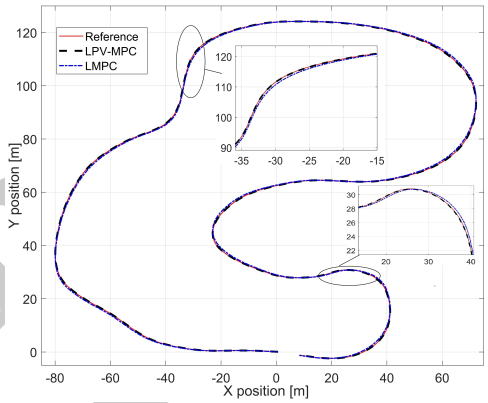

Fig. 8: Trajectory tracking.

0.02 compared 0.32 for LMPC which means the LPV-MPC tracking is almost ideal. The zoomed regions of the figure illustrate the significant difference in tracking accuracy considering the scale of the figure. This is partly due to the more accurate model used in LPV-MPC, in addition to the online adaptation of the cornering stiffness coefficients. Furthermore, unlike LMPC whose parameters were tuned iteratively, the parameters of LPV-MPC are well optimized by the proposed GA algorithm. Fig. 9 shows the steering and acceleration controls, and the lateral velocity of the vehicle. The tracking errors for longitudinal velocity, heading and lateral position are respectively shown in Fig. 10. The maximum velocity tracking error does not exceed 0.087 m/s, while the heading and position tracking errors are kept below $11°$ and 8 cm, which corresponds to the challenging section of the trajectory. Finally, Fig. 11 presents the computation time required by the LPV-MPC for solving the optimization problem. The mean computation time is evaluated at $(0.0113\ s)$ which is very suitable for real-time applications.

TABLE I: GA parameters

Parameter	Name	Value
Gen	Number of generations	15
N_P	Size of population	20
O_p	Percentage of offsprings	0.8
β	Selection pressure	0.75
μ	Mutation rate	0.3
σ	Mutation variance	0.15

TABLE II: MPC and model parameters

Parameter	Value	Parameter	Value
m	$1575\ (kg)$	C_d	0.29
I_z	$2875\ (kg.m^2)$	Λ	$1.6\ (m^2)$
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	0.2	N_p	10
g	$9.81\ (m/s^2)$	T_s	$0.033\ s$

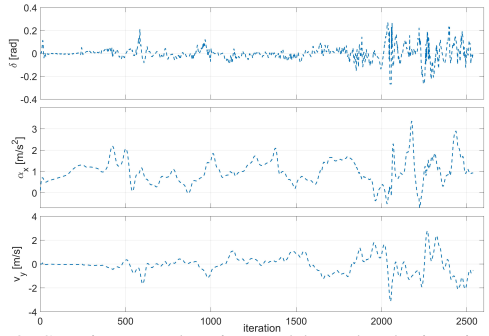

Fig. 9: Steering, acceleration and lateral velocity signals.

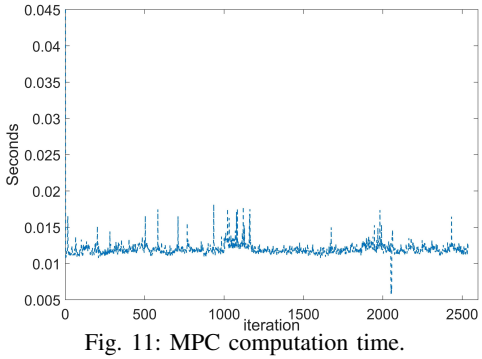

Fig. 11: MPC computation time.

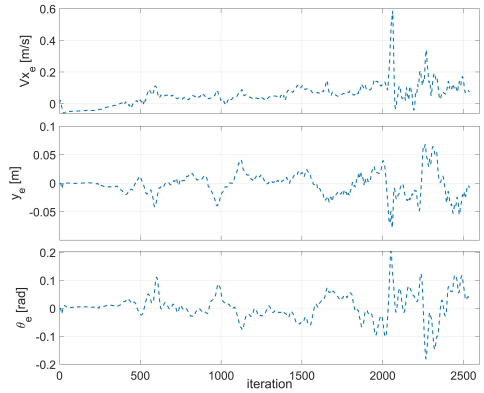

Fig. 10: Tracking performance.

V. Conclusions

This paper addressed the control task in autonomous driving by developing an adaptive LPV-MPC controller that handles both lateral and longitudinal dynamics. A data-driven approach based on Machine Learning has been proposed to predict the tire cornering stiffness coefficients online from measurable parameters only, and then adapt the LPV-MPC prediction model for more accurate predictions. Furthermore, an improved Genetic Algorithm has been proposed to optimize the cost function of the LPV-MPC controller by tuning its weighting matrices. The proposed control strategy has been tested on a challenging track against another variant of LPV-MPC. The obtained results proved that the proposed controller has superior performance and ensures high speed and trajectory tracking accuracy. Future research shall address the development of an online learning-based LPV-MPC for autonomous racing.

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] 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.

[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] 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.

[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 and L. Baghli, "Design, modeling and control of a hybrid grid-connected photovoltaic-wind system for the region of adrar, algeria," *International Journal of Environmental Science and Technology*, pp. 1–28, 2022.

[9] 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.

[10] E. Alcalá, V. Puig, and J. Quevedo, "LPV-MPC Control for Autonomous Vehicles," *IFAC-PapersOnLine*, vol. 52, no. 28, pp. 106–113, 2019.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

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

[17] A. Kwiatkowski, M.-T. Boll, and H. Werner, "Automated generation and assessment of affine lpv models," in *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 6690–6695, IEEE, 2006.

[18] 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.

[19] 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.

[20] M. T. Ahvanooey, Q. Li, M. Wu, and S. Wang, "A survey of genetic programming and its applications," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 13, no. 4, pp. 1765–1794, 2019.

[21] 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.

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