Optimized adaptive MPC for lateral control of autonomous vehicles

Yassine Kebbati
IBISC-EA4526
Université Paris-Saclay
Evry, France
yassine.kebbati@univ-evry.fr

Vicenç Puig
CS2AC
UPC
Barcelona, Spain
vicenc.puig@upc.edu

Naima Ait-Oufroukh
IBISC-EA4526
Université Paris-Saclay
Evry, France
naima.aitoufroukh@univ-evry.fr

Vincent Vigneron
IBISC-EA4526
Université Paris-Saclay
Evry, France
vincent.vigneron@univ-evry.fr

Dalil Ichalal
IBISC-EA4526
Université Paris-Saclay
Evry, France
dalil.ichalal@univ-evry.fr

***Abstract*—Autonomous vehicles are the upcoming solution to most transportation problems such as safety, comfort and efficiency. The steering control is one of the main important tasks in achieving autonomous driving. Model predictive control (MPC) is among the fittest controllers for this task due to its optimal performance and ability to handle constraints. This paper proposes an adaptive MPC controller (AMPC) for the path tracking task, and an improved PSO algorithm for optimising the AMPC parameters. Parameter adaption is realised online using a lookup table approach. The propose AMPC performance is assessed and compared with the classic MPC and the Pure Pursuit controller through simulations.**

***Index Terms*—Autonomous Vehicles, Optimization, Model Predictive Control, Adaptive Control, Particle Swarm Optimization.**

I. Introduction

The increase in road accidents and traffic jams all around the world has encouraged research to further develop autonomous driving technologies [1]. This field has received great attention and has seen some important advances in the last decades [2], [3], especially urban autonomous driving that is still a big challenge today. Among the most famous competitions in the field was the DARPA grand challenge that took place in 2007 [4], where several research groups tested their autonomous driving systems in an urban-like environment but only six of them managed to finish the race.

Self-driving vehicles do not need human intervention to navigate their path, they are fully controlled by the control module that commands the different actuators such as the accelerator and the steering wheel. Autonomous vehicles are divided into 5 automation levels; level zero represents the conventional cars with no automation, level one includes simple automatic driving systems like adaptive cruise control and electronic stability control. Level two introduces advanced systems like speed and steering control or emergency braking systems. Level three allows the vehicle to sense the environment through multiple sensors and drive autonomously while requiring human intervention in infeasible situations. In level four, the vehicle is fully autonomous with only occasional human intervention and only for certain driving modes, while in level five the vehicle is fully autonomous in all driving modes and without any human intervention [5]. Path tracking is a fundamental part of autonomous driving, it ensures that the vehicle follows a predefined trajectory and this is achieved by controlling the vehicle lateral dynamics. Various control strategies for this task are reported in the literature. For example, authors of [6] developed a lateral control system based on the adaptive pure pursuit controller, they used a PI controller to reduce the lateral offset and improve the tracking error caused by the look ahead distance. In [7], an output-feedback robust controller is introduced to assist in the path tracking task, the controller considers different driver's characteristics described by uncertain parameters such as delay time and achieves good stability and tracking performance through regional pole placement. Han *et al.* [8] designed an adaptive neural network PID controller for path tracking, they used it with a second order vehicle model whose parameters are estimated using the forgetting factor least square estimation. Kebbati *et al.* [9] developed a self-adaptive PID controller for speed regulation using neural networks and then genetic algorithms. Hu *et al.* [10] introduced an integral sliding mode controller for path tracking of an independently four-wheel actuated vehicle, the proposed nonlinear feedback controller considers multi-input multi-output (MIMO) systems with time varying trajectories. However, most of the above mentioned strategies cannot handle constraints that are imposed by the safety and physical limitations of actuators.

Model predictive control (MPC) stands out in this field due to the fact that it systematically handles constraints on control, output and state signals, in addition to the fact that it easily handles MIMO systems. In [11], Zhang *et al.* designed an MPC controller for path following with low computation load, for which they used Laguerre functions to approximate the control signal, and exponential weight to improve the tracking accuracy. Guo *et al.* [12] developed a path-tracking MPC controller that considers measurable disturbances, the authors considered the varying road conditions and small-angle assumptions as a form of measurable disturbance and solved the control problem using the differential evolution algorithm. In MPC control methods, there are many parameters to tune such as the weighting matrices and prediction and control horizons which is not trivial. This has not been addressed in the above mentioned papers, but in [13] and [14] the authors used fuzzy inference systems (FIS) to tune the weighting matrices of the MPC. Papers [15] and [16] used genetic algorithms (GA) and particle swarm optimization (PSO) to optimize the MPC tunable parameters. Nevertheless, the controllers in these approaches remain non adaptive to varying parameters and external uncertainties. In [17], Lin *et al.* designed an adaptive MPC controller that adapts to estimated varying road friction and cornering stiffness coefficients and in [18] an adaptive MPC was designed for lane keeping systems where the steering offset is continuously learned from measured data and adapted.

This paper proposes an adaptive MPC controller for the lateral control task, the contributions of this work are threefold: first, the design of an optimized adaptive MPC controller with Laguerre functions for the lateral control task. Second, the optimization of MPC parameters and cost function weight matrices with a new improved PSO algorithm. Third, the adaptation of the MPC, cost function and Laguerre function parameters online using a lookup table. Section II of this paper presents the vehicle lateral dynamics modeling and MPC control formulation. The controller optimization with improved PSO algorithm is introduced in section III. The results of the proposed controller are evaluated and discussed in section IV. Section V gives final conclusions and some suggestions for the continuation of future research.

II. Vehicle Modeling and Controller Design

A. Vehicle Lateral Dynamics Model

Path tracking is associated with the lateral motion of the vehicle, which is the displacement on the y-axis and the yaw motion around the z-axis. Therefore, the bicycle (2-DOF) dynamics model shown in Fig. 1, can accurately represent the vehicle motion and it is used to formulate the MPC controller. Applying newton's law of motion along the y-axis and z-axis yields the following equations:

$$\begin{cases} m(\ddot{y}+\dot{x}\dot{\phi}) & = 2F_{yf}+2F_{yr} \\ I_z\ddot{\phi} & = 2l_fF_{yf}-2l_rF_{yr} \end{cases} \tag{1}$$

where, m represents the mass of the vehicle, y, x and ϕ are the respective lateral and longitudinal positions and the heading angle of the vehicle, F_{yf} and F_{yr} are the longitudinal forces of the front and rear wheels respectively and l_f and l_r are the distances from the front and rear wheel axles to the vehicle's center of gravity (CG). The longitudinal force F_y is a result of the nonlinear interaction between the tires and the road, In this work the linearized tire model under the assumption of small slip angles is used and it is given as follows:

$$\begin{cases} F_{yf} & = C_{yf}\alpha_f \\ F_{yr} & = C_{yr}\alpha_r \end{cases} \tag{2}$$

with $C_{y(f,r)}$ and $a_{(f,r)}$ being the cornering stiffness coefficient and the lateral slip angle for the front and rear wheels respectively. The lateral slip angles are given by the following equations:

$$\begin{cases} \alpha_f & = \delta_f-\gamma_f \\ \alpha_r & = \delta_r-\gamma_r \end{cases} \tag{3}$$

where $\delta_{(f,r)}$ is the front and rear wheel steering angles where only the front wheel is considered steerable ($\delta_r = 0$), $\gamma_{(f,r)}$ define the angles between the direction of the front/rear wheel directions and the longitudinal velocity given as:

$$\begin{cases} \tan(\gamma_f) & = \frac{\dot{y}+l_f\dot{\phi}}{\dot{x}} \\ \tan(\gamma_r) & = \frac{\dot{y}-l_r\dot{\phi}}{\dot{x}} \end{cases} \tag{4}$$

The transformation from the vehicle body frame to the inertial frame is obtained by equation (5):

$$\dot{Y} = \dot{x}\sin\phi+\dot{y}\cos\phi \tag{5}$$

Hence, by considering equation (5), using angle approximation and substituting equations (4) and (3) into (2) and then equation (2) into (1) we obtain the lateral dynamics model as the following:

$$\begin{cases} m(\ddot{y}+\dot{x}\dot{\phi}) & = 2[C_{yf}(\delta_f-\frac{\dot{y}+l_f\dot{\phi}}{\dot{x}})+C_{yr}\frac{l_r\dot{\phi}-\dot{y}}{\dot{x}}] \\ I_z\ddot{\phi} & = 2[l_fC_{yr}(\delta_f-\frac{\dot{y}+l_f\dot{\phi}}{\dot{x}})-l_rC_{yr}\frac{l_r\dot{\phi}-\dot{y}}{\dot{x}}] \end{cases} \tag{6}$$

The above mentioned model can be transformed into the following continuous state space representation:

$$\begin{cases} \dot{x}_c & = A_cx_c+B_cu \\ y_c & = C_cx_c \end{cases} \tag{7}$$

where $x_c = [\dot{y}\ \phi\ \dot{\phi}\ y]^T$ is the state vector, $y_c = y$ is the output of the system, $u = \delta_f$ is the control input and A_c, B_c and C_c are the respective state, control and output matrices given by:

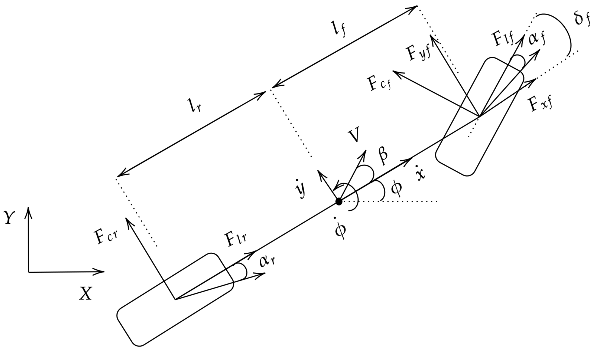

Fig. 1: 2-DOF Bicycle dynamic model.

$$A_c = \begin{bmatrix} -2\frac{C_{yf}+C_{yr}}{m\dot{x}} & 0 & -\dot{x} - 2\frac{C_{yf}l_f - C_{yr}l_r}{m\dot{x}} & 0 \\ 0 & 0 & 1 & 0 \\ -2\frac{C_{yf}l_f - C_{yr}l_r}{I_z\dot{x}} & 0 & -2\frac{C_{yf}l_f^2 + C_{yr}l_r^2}{I_z\dot{x}} & 0 \\ 1 & \dot{x} & 0 & 0 \end{bmatrix},$$

$$B_c = \begin{bmatrix} 2\frac{C_{yf}}{m} \\ 0 \\ 2\frac{C_{yf}l_f}{I_z} \\ 0 \end{bmatrix}, \qquad C_c = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}.$$

B. MPC Controller Design

The linear MPC controller with Laguerre functions is considered in this paper due to its high computational efficiency. The idea of MPC control is to use a descriptive model of the plant to predict its behaviour in the future along a prediction horizon N_p, and produce an optimal control sequence along the control horizon N_c that minimizes the error between the set-point and the output of the plant. In the context of path tracking, MPC aims to minimize the gap between the reference trajectory and the predicted trajectory. The optimal control sequence is achieved by minimizing a quadratic cost function subject to constraints, therefore, solving a constrained convex optimization problem. However, only the first term of this sequence is applied at each iteration and the whole prediction-optimization process is repeated iteratively, this is known as the receding horizon principle (Fig. 2). The linearized vehicle dynamic model of (7) is discretized to be used as the prediction model of the MPC controller:

$$\begin{cases} x_{(k+1)} & = A_k x_{(k)} + B u_{(k)} \\ y_{(k)} & = C x_{(k)} \end{cases} \tag{8}$$

To allow the MPC controller to be adaptive to varying longitudinal speeds, the state matrix A_k is iteratively updated with the current longitudinal velocity to produce an adaptive prediction model for varying speed profiles and not only fixed longitudinal speeds as is the case in most studies in the literature [11], [13], [17], [18]. This standard model is then augmented by embedding an integrator and introducing the output $y_{(k)}$ to the state vector as follows:

$$\begin{cases} \tilde{x}_{(k+1)} & = \tilde{A}_k \tilde{x}_{(k)} + \tilde{B} \Delta \tilde{u}_{(k)} \\ \tilde{y}_{(k)} & = \tilde{C} \tilde{x}_{(k)} \end{cases} \tag{9}$$

where the new state $\tilde{x} = \left[\Delta x_{(k)}\ y_{(k)}\right]^T$ is connected to the output, the new input signal is $\Delta \tilde{u}_{(k)}$ and the augmented system matrices are given by:

$$\tilde{A} = \begin{bmatrix} A_k & o_m^T \\ CA_k & I_{q \times q} \end{bmatrix}, \ \tilde{B} = \begin{bmatrix} B \\ CB \end{bmatrix}, \ \tilde{C} = [o_m \ I_{q \times q}].$$

$o_m = \overbrace{[0\ 0...0]}^{n}$ is a zero vector and $I_{q \times q}$ is an identity matrix with n being the number of states, m the number of inputs and q the number of outputs. At $k_i > 0$ the state vector $x(k)$ is assumed available providing the current plant information, the future control trajectory along the control horizon N_c is given by:

$$\Delta u(k_i), \Delta u(k_i + 1), ..., \Delta u(k_i + N_c - 1)$$

The future plant information is predicted along the prediction horizon N_p using the model given in equation (9) and the future control parameters through the following equation:

$$Y = F x_{(k_i)} + \Phi \Delta U \tag{10}$$

where Y is the future outputs, F is the future prediction matrix, Φ is the future control matrix and ΔU is the future control sequence given by:

$$Y = [y(k_i + 1|k_i)\ y(k_i + 2|k_i y(k_i + 3|k_i) \dots y(k_i + N_p|k_i)]^T$$

$$\Delta U = [\Delta u(k_i)\ \Delta u(k_i + 1)\ \Delta u(k_i + 2) \dots \Delta u(k_i + N_c - 1)]^T$$

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix}$$

$$\Phi = \begin{bmatrix} CB & 0 & \dots & 0 \\ CAB & CB & \dots & 0 \\ CA^2B & CAB & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & \dots & CA^{N_p-N_c}B \end{bmatrix}$$

The goal of MPC is to find the best ΔU that minimizes the error $(R_s - Y)$, where $R_s = [1\ 1 \dots 1] r(k_i)$ is the reference trajectory containing N_p set-point information $r(k_i)$. Then, the constrained optimization problem is formulated as:

$$min\ J = (R_s - Y)^T Q (R_s - Y) + \Delta U^T R \Delta U \tag{11}$$

$$s.t: x(k+1) = Ax(k) + B\Delta u(k) \tag{12}$$

$$\Delta u_{min} \leq \Delta u \leq \Delta u_{max} \tag{13}$$

$$u_{min} \leq u \leq u_{max} \tag{14}$$

$$y_{min} \leq y \leq y_{max} \tag{15}$$

Equation (11) is the cost function to be minimized, (12) is the vehicle dynamic model and equations (13) to (15) represent the constraints on the control, control increment and output signals. Q and R are weighting matrices that penalise the tracking precision and control magnitude respectively. The set

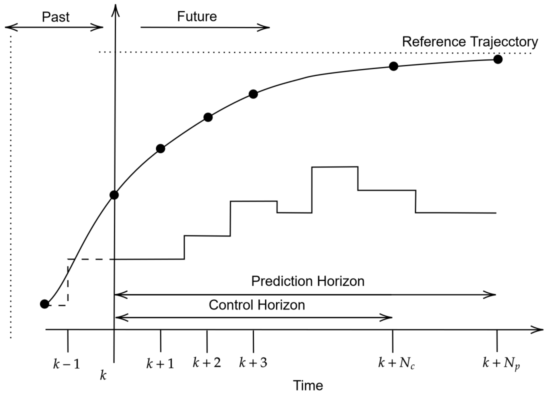

Fig. 2: MPC receding horizon concept.

of constraints can be reformulated and expressed in terms of the decision variable ΔU as:

$$M\Delta U \leq \gamma \tag{16}$$

such that M contains 3 reformulation sub-matrices and γ contains the set of upper and lower bounds on the control, control increment and output signals. The model predictive control becomes a quadratic programming (QP) problem with quadratic cost function and linear inequality constraints formulated in the following form:

$$\begin{cases} J = \frac{1}{2}x^T E x + x^T K \\ Mx \leqslant \gamma \end{cases} \tag{17}$$

where E, K, M and γ are compatible matrices and vectors and x is the decision variable (Δu) in the quadratic programming problem, E is assumed symmetric and positive definite.

1) Laguerre function approximation: The Laguerre function is used to reduce the computation burden by approximating the control sequence ΔU, the orthogonal feature of the Laguerre function allows to realize long control horizons without using a large number of parameters. The discrete form of a Laguerre function is given by:

$$\begin{aligned} \Gamma_1(z) &= \frac{\sqrt{1-\alpha^2}}{1-\alpha z^{-1}} \\ \Gamma_2(z) &= \frac{\sqrt{1-\alpha^2}}{1-\alpha z^{-1}} \frac{z^{-1}-\alpha}{1-\alpha z^{-1}} \\ &\vdots \\ \Gamma_N(z) &= \frac{\sqrt{1-\alpha^2}}{1-\alpha z^{-1}} (\frac{z^{-1}-\alpha}{1-\alpha z^{-1}})^{N-1} \end{aligned} \tag{18}$$

where $0 \leqslant \alpha \leqslant 1$ is the pole of the discretized Laguerre network called scaling factor and N is the number of terms. The orthonormality of the Laguerre functions in the time domain is expressed by:

$$\sum_{k=0}^{\infty} l_i(k) l_j(k) = 0 \qquad for \quad i \neq j. \tag{19}$$

$$\sum_{k=0}^{\infty} l_i(k) l_j(k) = 1 \qquad for \quad i = j. \tag{20}$$

The discrete Laguerre function expression is simplified into a vector form as the following:

$$L(k) = [l1(k)\ l_2(k) \ldots l_N(k)]^T \tag{21}$$

where $l_i(k)$ is the inverse z-transform of $\Gamma_i(z, \alpha)$:

$$\Gamma_k(z) = \Gamma_{k-1}(z) \frac{z^{-1}-\alpha}{1-\alpha z^{-1}} \tag{22}$$

Based on equation (22), the Laguerre sequence can be expressed recursively by:

$$L(k+1) = A_l L(k) \tag{23}$$

with A_l being an $N \times N$ matrix and a function of α and $\beta = (1-\alpha^2)$, and $L(0)$ as initial condition given by:

$$L(0)^T = \sqrt{\beta}[1 \ \ -\alpha \quad \alpha^2 \ \ -\alpha^3 \ \ \ldots \ \ (-1)^{N-1}\alpha^{N-1}].$$

For N=5, we have: $A_l = \begin{bmatrix} \alpha & 0 & 0 & 0 & 0 \\ \beta & \alpha & 0 & 0 & 0 \\ -\alpha\beta & \beta & \alpha & 0 & 0 \\ \alpha^2\beta & -\alpha\beta & \beta & \alpha & 0 \\ -\alpha^3\beta & \alpha^2\beta & -\alpha\beta & \beta & \alpha \end{bmatrix}$,

$$L(0)^T = \sqrt{\beta} \begin{bmatrix} 1 \\ -\alpha \\ \alpha^2 \\ -\alpha^3 \\ \alpha^4 \end{bmatrix}.$$

For the case of $\alpha = 0$ the Laguerre function becomes just a set of pulses and the MPC design becomes equivalent to the traditional approach discussed earlier. Thus, the control increment at $(k+i)$ is represented by the Laguerre terms as the following:

$$\Delta u(k+i) = \sum_{j=1}^{N} c_j(k) l_j(i) = L(i)^T \eta, \ i = 1, 2, ..., N_c \tag{24}$$

where $\eta = [c_1 \ c_2 \ \ldots \ c_N]^T$ is the Laguerre fitting coefficient, solving the QP problem becomes a matter of finding the optimal fitting coefficient η_{opt}. The solution of the QP problem given in equations (17) can be divided into two cases; minimization with equality constraints, and minimization with inequality constraints. For equality constraints the QP problem is solved using Lagrange multipliers:

$$J = \frac{1}{2}x^T E x + x^T + \lambda^T(Mx - \gamma) \tag{25}$$

The Lagrange expression is subject to equality constraints ($Mx = \gamma$) and the solution is obtained by finding the optimal Lagrange multiplier λ and corresponding decision variable x through partial derivatives equated to zero:

$$\frac{\partial J}{\partial x} = Ex + F + M^T\lambda = 0 \tag{26}$$

$$\frac{\partial J}{\partial \lambda} = Mx - \gamma = 0 \tag{27}$$

Hence, the solution is the optimal λ_{opt} which contains the Lagrange multipliers, and the optimal decision variable x_{opt} which denotes ΔU in the context of predictive control:

$$\lambda_{opt} = -(ME^{-1}M^T)^{-1}(\gamma + ME^{-1}F) \tag{28}$$

$$x_{opt} = -E^{-1}(M^T\lambda_{opt} + F) \tag{29}$$

For inequality constraints, ($Mx \leqslant \gamma$) is divided into active ($Mx = \gamma$) and inactive ($Mx < \gamma$) constraints and these are defined in terms of Lagrange multipliers through the Kuhn-Tucker conditions:

$$\begin{aligned} Ex + F + M^T\lambda &= 0 \\ Mx - \gamma &\leqslant 0 \\ \lambda^T(Mx - \gamma) &= 0 \\ \lambda &\geqslant 0 \end{aligned} \tag{30}$$

The solution in this case requires the identification of active constraints first and then solving the optimization for those active constraints while inactive constraints are eliminated. In this paper, the Hildreth's QP method has been used to solve the QP problem of (19), for ample details on this method we refer the reader to [19].

III. Controller Optimization With Improved PSO

The particle swarm optimization is a swarm intelligence algorithm that mimics animal group behaviours, such as bird flocks and fish swarms. It is an evolutionary population-based algorithm where each individual (particle) represents a possible solution and the group of individuals make up a swarm. Due to its flexibility and ease of implementation, the PSO tool is used in various fields [20]–[22]. In an optimization problem with N dimensions, each i^{th} particle is attributed with a velocity vector $v_i = [v_{i1}, v_{i2}, ..., v_{in}]$ and a position vector $p_i = [p_{i1}, p_{i2}, ..., p_{in}]$. The classic algorithm is defined by:

$$\begin{cases} v_i(k+1) = & \omega v_i(k) + c_1 r_1 (Pb_i(k) - x_i(k)) \\ & + c_2 r_2 (Gb(k) - x_i(k)) \\ x_i(k+1) = & x_i(k) + v_i(k+1) \end{cases} \tag{31}$$

where at each iteration k, v_i and p_i are the velocity and position of particle i, and ω is the inertia weight. The coefficients $c_{1,2}$ are learning factors called cognitive and social accelerations respectively, and $r_{1,2} \in [0,1]$ are random numbers. Pb_i represents the position with the best fitness score for particle i and Gb is the global best position in the swarm. In the classic algorithm ω and $c_{1,2}$ are constant values, several improved versions have been reported in the literature such as [23] and [24] where w and $c_{1,2}$ are dynamic according to equation (32) and table (I):

$$\omega = \omega_{max} - (\omega_{max} - \omega_{min}) \frac{g}{G} \tag{32}$$

where ω_{max} and ω_{min} are the maximum and minimum of the inertia weight, g and G represent the number of the actual generation and the maximum number of generations respectively.

TABLE I: Rules for updating c_1 and c_2.

Phase	c_1	c_2
Exploration	Big increase	Big decrease
Exploitation	Small increase	Small decrease
Convergence	Small increase	Small increase
Jumping out	Big decrease	Big Increase

In table (I), the cognitive acceleration coefficient c_1 is increased during the exploration phase to explore the swarm and pull the particle towards its best position Pb. During the exploitation phase, the social acceleration coefficient c_2 is increased to push the swarm towards the global best Gb. In the convergence phase, the swarm begins to find the globally optimal region and increasing c_2 while slightly decreasing c_1 helps accelerate the convergence. At the finale phase of the optimization, the global best particle moves from a local optimum, where most particles are clustered, towards a better optimum (the new global best). Therefore, all clustered particles must move as fast as possible towards the new global best. To achieve this, c_2 is greatly increased during this phase while c_1 is greatly decreased. to balance the four phases in table (I), we control c_1 and c_2 by the following logic which promotes the global search capabilities of PSO:

$$\begin{cases} c_1(k+1) = c_1(k) + \alpha \\ c_2(k+1) = c_2(k) + \beta \\ \alpha = -\beta = 0.05 \quad \text{for} \quad \frac{g}{G} \leq 20\% \\ \alpha = -\beta = 0.02 \quad \text{for} \quad 20\% \leq \frac{g}{G} \leq 35\% \\ \alpha = -\beta = -0.035 \quad \text{for} \quad 35\% \leq \frac{g}{G} \leq 75\% \\ \alpha = -\beta = -0.0015 \quad \text{for} \quad \frac{g}{G} \geq 75\% \end{cases} \tag{33}$$

where g is the current generation and G is the maximum number of generations.

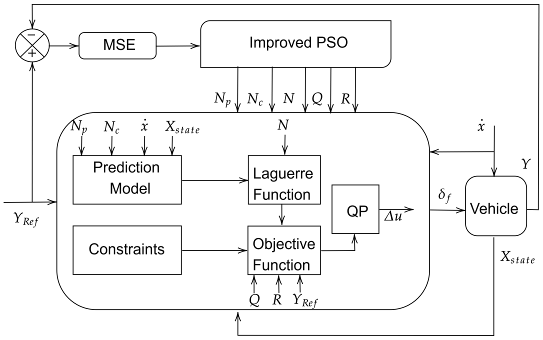

Fig. 3: Schematic diagram of the proposed approach.

According to (32), ω decreases linearly over iterations. A small ω promotes the local search of PSO [24]. A new formula for updating the inertia weight is proposed in this paper (34), which ensures an exponential decrease rather than linear.

$$\omega = \omega_{min} + \frac{\exp\left(\omega_{max} - \lambda_1(\omega_{max} + \omega_{min}) \frac{g}{G}\right)}{\lambda_2} \tag{34}$$

This has been found to effectively enhance the overall search capabilities of PSO when coupled with the control logic of $c_{1,2}$. Unlike a linear decrease, an exponential decrease of ω ensures faster convergence and better overall optimization. $\lambda_{1,2}$ are adjustable constants to achieve a decrease from ω_{max} to ω_{min}.

The proposed PSO algorithm is used to optimize the parameters of the MPC designed with the Laguerre function in section II-B. These parameters include; the prediction and control horizons N_p and N_c, the weighting matrices Q and R and the number of Laguerre terms N. To the best of our knowledge, existing research has not addressed the tuning and adaptation of these parameters all-together with the proposed PSO algorithm. Thus, the proposed approach allows us to optimize the controller performance in terms of its tunable parameters and Laguerre approximation at the same time without losing the computation efficiency. The mean squared error (MSE) is used as a fitness function to assess the optimality of the solutions. The schematic diagram of our approach is given in Fig. 3, where Y_{ref}, Y, $\dot{x}$ and X_{state} are the reference path, the vehicle lateral position, the longitudinal velocity and the states vector respectively.

IV. Results and Discussion

The performance of the proposed PSO algorithm is compared to the improved version of [24], the modified PSO of [23] and the latter enhanced with a damped inertia weight instead of a constant one ($\omega(k+1) = \omega(k) \times \omega_{dmp}$). The sphere function ($f_1(x) = \sum_{i=1}^{n} x_i^2$) is used as a benchmark test similar to [24], and the optimization is carried out with the parameters given in table (II) and the logic stated by equations (33). The obtained results in Fig. 4, show that the proposed PSO is much faster than the other PSO versions and able to find better optimal solutions. After 41 generations, the proposed PSO reached a fitness of 10^{-3} compared to 2.1 and 3.11 for the improved PSO and the classic damped PSO respectively, while the classic PSO reached only 5.59.

A high fidelity vehicle model has been built using the Vehicle Dynamics Blockset of `MATLAB`, the latter consists of a 3 DOF dual track lateral dynamics block with nonlinear Pacejka tire formula [26]. In addition, a simplified powertrain block is added to accommodate varying speed profiles with the predictive longitudinal driver block. The proposed MPC controller is optimized with the proposed PSO using 15 generations and 20 particles, the rest of the parameters are identical to table (II). The optimization is done for multiple longitudinal speeds ($\dot{x}(m/s) \in [3, 27]$) and reference lateral positions ($y_{ref}(m) \in [-15, 15]$). The adaptive MPC (AMPC) has been tested against the pure pursuit controller (P.P) and the classic MPC (tuned by trial and error) for three scenarios:

1) Sc_1 consists of a double lane change with constant longitudinal velocity $v_x = 9m/s$.
2) Sc_2 is a double lane change with varying longitudinal velocity (Fig. 8) where the AMPC is adapted to both trajectory and velocity profile.
3) Sc_3 where the AMPC is tested for a general trajectory with varying longitudinal velocity (Fig. 15) subject to wind gust as an external disturbance (Fig. 21).

The parameters of the dynamic model (6) and the MPC controller are listed in table (III). The results of (Sc_1) show that the three controllers are able to track the double lane change trajectory. However, the proposed AMPC outperforms the other two controllers where it achieved an MSE value of $e_1 = 2.6$ compared to $e_2 = 11.27$ and $e_3 = 12.89$ for the classic MPC and pure pursuit respectively. Fig. 5-7 compare the tracking performance ($Y_{position}$), the steering angles (δ_f) and the heading rates ($\dot{\phi}$) for the three controllers.

TABLE II: PSO hyper parameters.

Parameter	Interpretation	Value
G	Number of generations	15
N_{Pop}	Number of particles	20
ω_{max}	Maximum inertia weight	0.99
ω_{min}	Minimum inertia weight	0.1
ω_{dmp}	Damping constant	0.99
c_{1i}	Initial cognitive acceleration coefficient	2
c_{2i}	Initial social acceleration coefficient	2
λ_1	Constant	30
λ_2	Constant	3

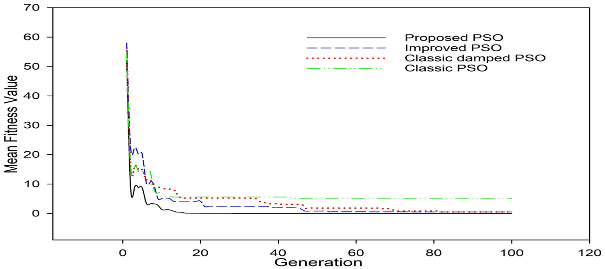

Fig. 4: Performance of the proposed PSO.

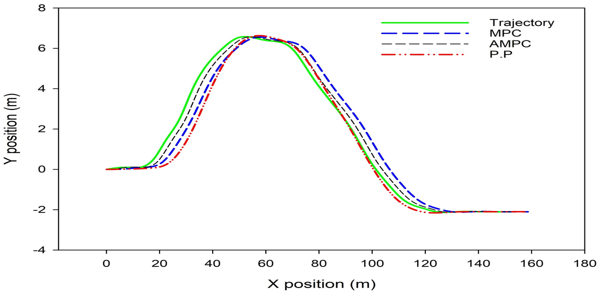

Fig. 5: Path tracking for Sc_1.

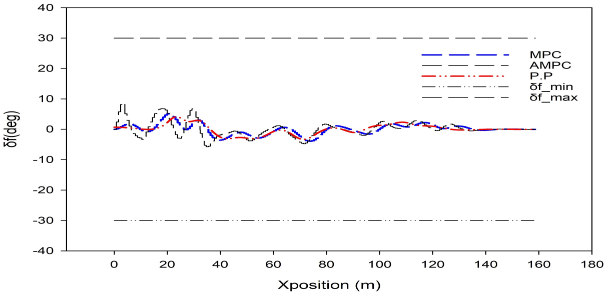

Fig. 6: Control signal for Sc_1.

On the other hand, the P.P controller could not perform the double lane change of (Sc_2), while the proposed AMPC tracked the trajectory significantly better than the classic MPC. Their respective MSE values are $e_1 = 3.59$ against $e_2 = 14.2$. Fig. 9-11 show the tracking performance, the control signals and the heading rates.

TABLE III: MPC and dynamic model parameters

Parameter	Interpretation	Value
m	vehicle mass	1575 (kg)
I_z	moment of inertia	$2875 (kg.m^2)$
l_f	front axle to (CG)	$1.2 (m)$
l_r	rear axle to (CG)	$1.6 (m)$
C_f	front cornering stiffness	$19000 (N/rad)$
C_r	rear cornering stiffness	$33000 (N/rad)$
T	sampling time	$0.1 (s)$
N_p	Prediction horizon	45
N_c	Control horizon	15
$\Delta u_{max/min}$	Control increment constraints	$\pm\frac{\pi}{12} (rad)$
$u_{max/min}$	Control constraints	$\pm\frac{\pi}{6} (rad)$
R	Control magnitude weight	0.01
Q_y	Tracking precision weight	10
N	Number of Laguerre terms	5
α	Laguerre scale factor	0.75

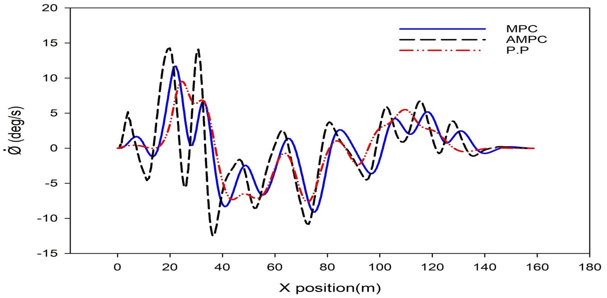

Fig. 7: Heading rate for Sc_1.

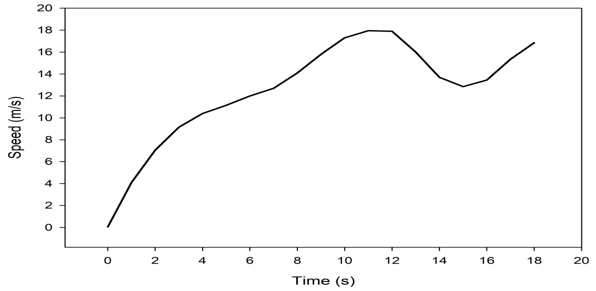

Fig. 8: Longitudinal velocity profile for Sc_2.

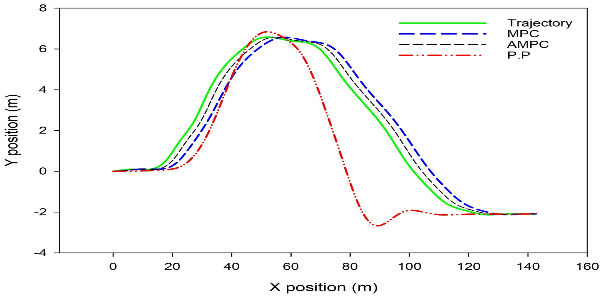

Fig. 9: Path tracking for Sc_2.

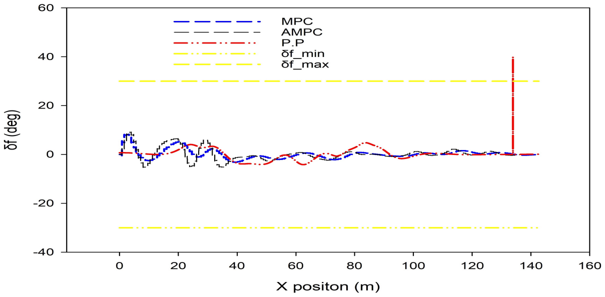

Fig. 10: Control signal for Sc_2.

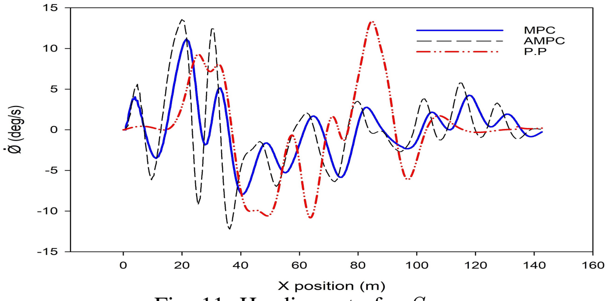

Fig. 11: Heading rate for Sc_2.

The results for (Sc_3) confirm better tracking ability of the proposed AMPC. Fig. 12-14 show the tracking performance for a constant velocity $v_x = 11m/s$. The MSE values are $e_1 = 176.73$, $e_2 = 347.79$ and $e_3 = 1255.3$ for the AMPC, MPC and P.P respectively. In Fig. 16-18, the same trajectory is simulated with a varying velocity profile (Fig. 15). The performance of AMPC is much improved compared to regular MPC, but both are able to track the trajectory and the resulting MSE values for AMPC and MPC are $e_1 = 121.97$ and $e_2 = 423.71$. The parameters adaptation for this case is illustrated in Fig. 19-20.

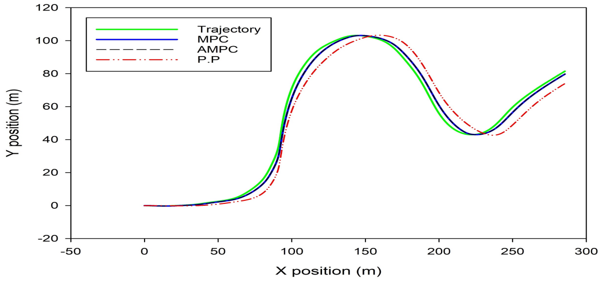

Fig. 12: Path tracking for Sc_3 with constant velocity.

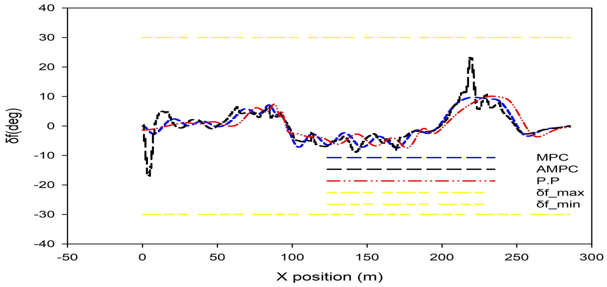

Fig. 13: Control signal for Sc_3 with constant velocity.

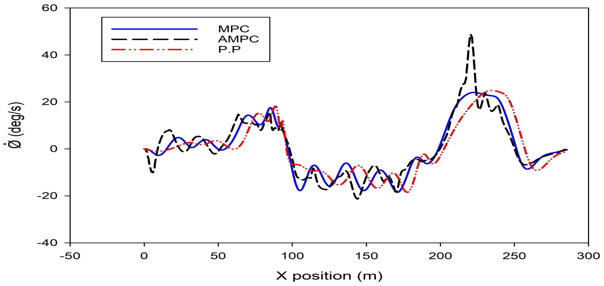

Fig. 14: Heading rate for Sc_3 with constant velocity.

For the same trajectory and velocity profile, the lateral wind disturbance profile shown in Fig. 21 is applied to the high fidelity vehicle model to verify the controller robustness. The simulation results (Fig. 22-24) show that at certain wind speeds, the classic MPC is not able to reject the external disturbance and its performance deteriorates and diverges. On the other hand, the proposed AMPC still manages to track the trajectory despite the introduced disturbances. This is due to its adaptive feature with the optimized parameters which allows the AMPC to overcome the perturbations.

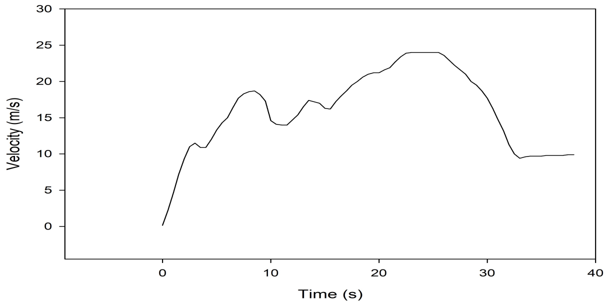

Fig. 15: Velocity profile for Sc_3.

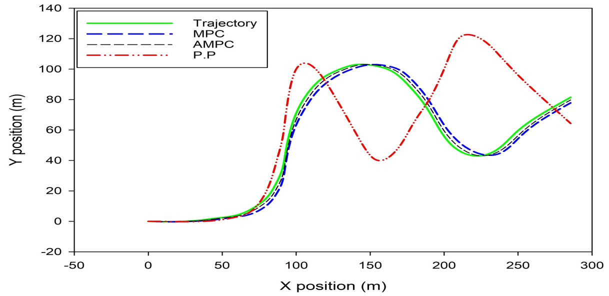

Fig. 16: Path tracking for Sc_3 with varying velocity.

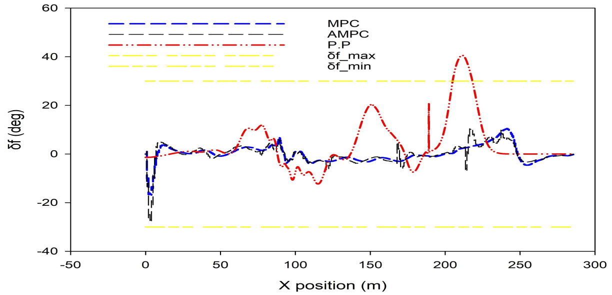

Fig. 17: Control signal for Sc_3 with varying velocity.

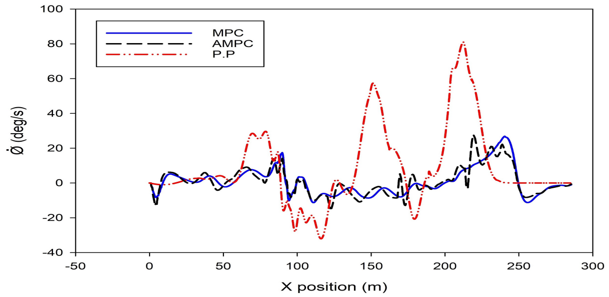

Fig. 18: Heading rate for Sc_3 with varying velocity.

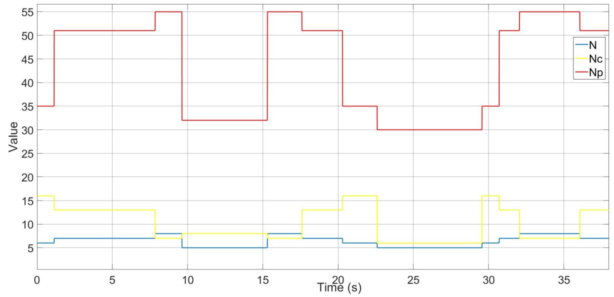

Fig. 19: MPC parameters adaptation for Sc_3 with varying velocity.

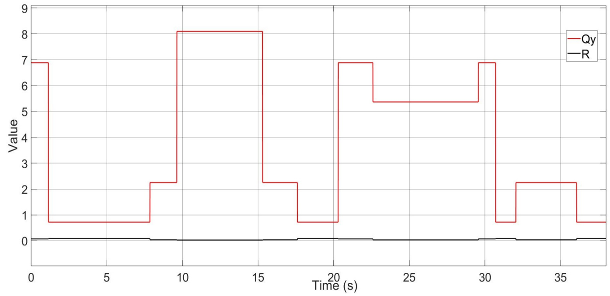

Fig. 20: Cost function adaptation for Sc_3 with varying velocity.

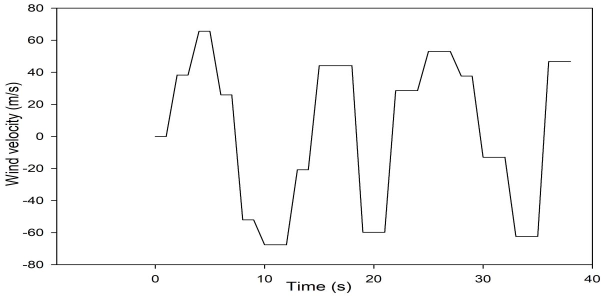

Fig. 21: Wind disturbance profile.

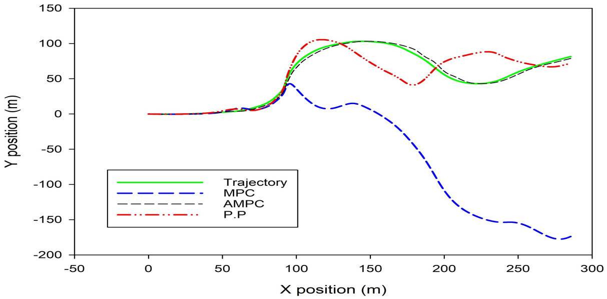

Fig. 22: Path tracking for Sc_2 under wind disturbance.

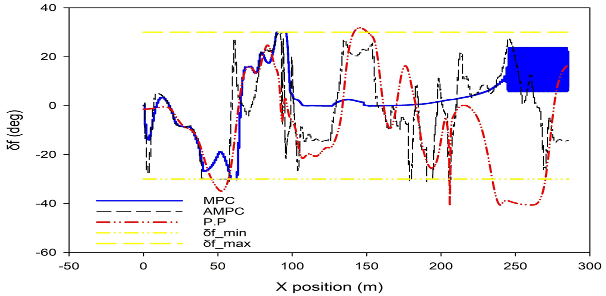

Fig. 23: Control signal for Sc_3 under wind disturbance.

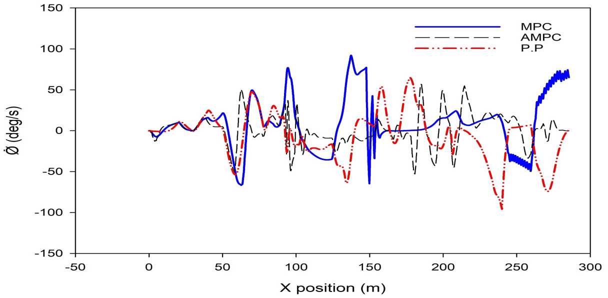

Fig. 24: Heading rate for Sc_3 under wind disturbance.

V. Conclusions

This paper addressed the lateral control task in autonomous vehicles for which an AMPC controller is proposed. A new improved PSO algorithm is also proposed to optimize and tune the AMPC controller for varying working conditions. The MPC controller is formulated with a linearized bicycle model and Laguerre functions and solved through quadratic programming. The parameter optimization is performed for different scenarios, and the generated optimal parameters are used for controller adaptation. The performance of the proposed controller is tested against the classic MPC and the Pure Pursuit controller using a high fidelity vehicle model.

The results demonstrate that the designed AMPC is able to perform much better tracking accuracy especially for varying longitudinal velocities. Moreover, the AMPC is able to handle varying working conditions and reject external disturbances. These good performances are achieved thanks to its adaptive feature and well optimized parameters through the proposed PSO algorithm. Future research shall address the possibility of using neural networks and adaptive neuro-fuzzy inference systems to learn the optimized parameters by the proposed PSO algorithm for adaptation purposes.

References

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, “A Survey of Autonomous Driving: Common Practices and Emerging Technologies,” IEEE Access, vol. 8, pp. 58443–58469, 2020

[2] Q. Yao, Y. Tian, Q. Wang, and S. Wang, “Control Strategies on Path Tracking for Autonomous Vehicle: State of the Art and Future Challenges,” IEEE Access, vol. 8, pp. 161211–161222, 2020

[3] S. Bacha, R. Saadi, M. Y. Ayad, A. Aboubou, and M. Bahri, “A review on vehicle modeling and control technics used for autonomous vehicle path following,” Int. Conf. Green Energy Convers. Syst. GECS 2017

[4] M. Buehler, K. Iagnemma, and S. Singh, The DARPA urban challenge: autonomous vehicles in city traffic. Springer, 2009

[5] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, “A survey of motion planning and control techniques for self-driving urban vehicles,” IEEE Trans. Intell. Veh., vol. 1, no. 1, pp. 33–55, 2016

[6] M.-W. Park, S.-W. Lee, and W.-Y. Han, ‘Development of lateral control system for autonomous vehicle based on adaptive pure pursuit algorithm,’ in Proc. 14th Int. Conf. Control, Autom. Syst. (ICCAS), Oct. 2014, pp. 1443–1447.

[7] J.X. Wang, M.M. Dai, G.D. Yin, and N. Chen Output-feedback robust control for vehicle path tracking considering different human drivers’ characteristics. Mechatronics 2018; 50: 402–412

[8] G. Han, W. Fu, W. Wang, and Z. Wu, “The lateral tracking control for the intelligent vehicle based on adaptive PID neural network,” Sensors (Switzerland), vol. 17, no. 6, pp. 1–15, 2017

[9] Y. Kebbati, N. Ait-oufroukh, V. Vigneron, D. Ichalal, and D. Gruyer, ”Optimized self-adaptive PID speed control for autonomous vehicles,” ICAC 2021 - 2021 26th IEEE Int. Conf. Autom. Comput. no. September, pp. 81–86, 2021.

[10] C. Hu, R. Wang, and F. Yan, “Integral Sliding Mode-Based Composite Nonlinear Feedback Control for Path Following of Four-Wheel Independently Actuated Autonomous Vehicles,” IEEE Trans. Transp. Electrif., vol. 2, no. 2, pp. 221–230, 2016

[11] B. Zhang, C. Zong, G. Chen, and B. Zhang, “Electrical Vehicle Path Tracking Based Model Predictive Control with a Laguerre Function and Exponential Weight,” IEEE Access, vol. 7, pp. 17082–17097, 2019

[12] 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,” Mech. Syst. Signal Process., vol. 118, pp. 41–60, 2019

[13] 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

[14] M. S. Akbari, A. A. Safavi, N. Vafamand, T. Dragicevic, and J. Rodriguez, “Fuzzy Mamdani-based Model Predictive Load Frequency Control,” 2020 IEEE 11th Int. Symp. Power Electron. Distrib. Gener. Syst. PEDG 2020, pp. 7–12, 2020

[15] V. Ramasamy, R. K. Sidharthan, R. Kannan, and G. Muralidharan, “Optimal tuning of model predictive controller weights using genetic algorithm with interactive decision tree for industrial cement kiln process,” Processes, vol. 7, no. 12, 2019

[16] H. Moumouh, N. Langlois, and M. Haddad, “A Novel Tuning approach for MPC parameters based on Artificial Neural Network,” IEEE Int. Conf. Control Autom. ICCA, vol. 2019-July, pp. 1638–1643, 2019

[17] F. Lin, Y. Chen, Y. Zhao, and S. Wang, “Path tracking of autonomous vehicle based on adaptive model predictive control,” Int. J. Adv. Robot. Syst., vol. 16, no. 5, pp. 1–12, 2019

[18] M. Bujarbaruah, X. Zhang, H. E. Tseng, and F. Borrelli, “Adaptive MPC for Autonomous Lane Keeping,” 2018

[19] L. Wang,Model Predictive Control System Design and Implementation Using MATLAB, vol. 53. 2016.

[20] B. Song, Z. Wang, and L. Zou, “An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve,” Appl. Soft Comput., vol. 100, p. 106960, 2021

[21] Y. Xie and W. Zhang, “A novel tuning method for PID controller based on improved PSO algorithm for unstable plants with time delay,” Chinese Control Conf. CCC, vol. 2019-July, pp. 4270–4275, 2019

[22] Z. Tian, Y. Ren, and G. Wang, “Short-term wind speed prediction based on improved PSO algorithm optimized EM-ELM,” Energy Sources, Part A Recover. Util. Environ. Eff., vol. 41, no. 1, pp. 26–46, 2019

[23] Y. Shi and R. Eberhart, “Modified particle swarm optimizer,” Proc. IEEE Conf. Evol. Comput. ICEC, no. February 2015, pp. 69–73, 1998

[24] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung, “Adaptive particle swarm optimization,” IEEE Trans. Syst. Man, Cybern. Part B Cybern., vol. 39, no. 6, pp. 1362–1381, 2009

[25] Y. Del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J. C. Hernandez, and R. G. Harley, “Particle swarm optimization: Basic concepts, variants and applications in power systems,” IEEE Trans. Evol. Comput., vol. 12, no. 2, pp. 171–195, Apr. 2008

[26] H.B. Pacejka, (2008). Vehicle System Dynamics : International Journal of Vehicle Mechanics and Mobility. International Journal of Vehicle Mechanics and Mobility, (August 2012), 37–41