A Novel Iterative Learning-Model Predictive Control Algorithm for Accurate Path Tracking of Articulated Steering Vehicles

Xuanwei Chen, Changlin Yang, Jiaqi Cheng, Huosheng Hu, *Life Senior Member, IEEE,* Guifang Shao, Yunlong Gao and Qingyuan Zhu, *Senior Member, IEEE*

***Abstract*—Refining the path tracking of articulated steering vehicles amidst terrain disturbances presents a formidable challenge. While model predictive control (MPC) offers promise in tackling this issue, its efficacy is often hindered by model intricacies and inaccuracies. In this communication, an innovative approach termed iterative learning-model predictive control (IL-MPC) is introduced to enhance path tracking performance on rugged terrains. Initially, an MPC controller grounded in a simplified kinematic model is established to ensure stability in path tracking. Subsequently, an iterative learning algorithm is integrated to meticulously capture and mitigate MPC controller errors. A comprehensive feedforward-feedback framework coupled with a spatial indexing method is proposed to synergize the strengths of iterative learning and MPC. Through rigorous evaluations across diverse paths and terrains, the method demonstrates robustness against terrain disturbances, affirming its efficacy in real-world scenarios.**

***Index Terms*—Articulated steering vehicle, path tacking, iterative learning, model predictive control, off-road terrain**

I. INTRODUCTION

Articulated steering vehicles (ASVs) are common engineering vehicles working on off-road terrain, and their steering relies on the relative rotation of their separated bodies [1]. ASVs are characterized by high flexibility, large load-carrying capacity, and small turning radius and widely used to transport materials from one place to another in mining, agriculture, and construction industries [2]. The urgency for autonomous driving in articulated steering vehicles is further intensified due to the harsh working environment and labor shortage.

Manuscript received: February 24, 2024; Revised April 23, 2024; Accepted June 19, 2024. This letter was recommended for publication by Editor Clement Gosselin upon evaluation of the Associate Editor and Reviewers' comments.

This work was supported in by University-Industry Collaborative Program of Fujian Province (Grant number 2021H6019), the National Natural Science Foundation of China (Grant number 52075461), and China Tobacco's "Selecting Best Projects by Opening Competition Mechanism" Program (Grant number 110202301013-2). (*Corresponding author: Qingyuan Zhu*)

Xuanwei Chen, Changlin Yang, Jiaqi Cheng, Guifang Shao, Yunlong Gao and Qingyuan Zhu are with Pen-Tung Sah Institute of Micro-Nano Science an d Technology, Xiamen University, Xiamen 361102, China. (email: xm_cxw @163.com; 2484724036@qq.com; chengjqss@163.com; gfshao@xmu.edu.c n; gaoyl@xmu.edu.cn; zhuqy@xmu.edu.cn)

Huosheng Hu is with School of Computer Science and Electronic Engineering, University of Essex, Colchester, CO4 3SQ, UK. (email: hhu@essex.ac.uk)

Path tracking is a basic function of autonomous vehicles, enabling vehicles to track specific paths automatically. Several modeling and control algorithms have been designed for ASVs to achieve path tracking, such as PID [3, 4], sliding mode [5], feedback linearization [6, 7], and linear quadratic [8]. These algorithms lack the capacity to consider vehicle constraints and do not incorporate the idea of feedforward control. MPC has the unique ability to manage constraints such as actuator saturation, safety, and stability, and attracted attention in path tracking control for ASVs [9-14].

Most MPC studies have used kinematic models without considering forces affected motion, such as a switching MPC that ensures optimal control under changing conditions [10], an LPV-MPC algorithm to improve computational efficiency [11] and a nonlinear MPC to enhance accuracy under extreme conditions [12]. On the other hand, some MPC studies were based on dynamics by considering interaction between motions and forces and validated through simulations. Shi et al. constructed an adaptive MPC based on loader's dynamic error model to ensure tracking accuracy under variable curvature conditions [13]. Shahirpour employed a simplified dynamic model for path tracking at high speeds [14].

Currently, accurate path tracking remains a big challenge for ASVs working on off-road terrain. Terrain disturbances will induce changes in vehicle attitude and tire slippage, thereby deteriorating path tracking accuracy. Although some vehicle-terrain interaction models were developed for other types of vehicles to mitigate terrain disturbances [15-17], a real-time solvable vehicle-terrain model for ASVs presents a significant challenge due to modeling errors, nonlinearities, and parameter time-variability caused by variable-structure vehicle bodies and complex vehicle-terrain relationships.

Some researchers used machine learning to provide an accurate model, aiming to reduce computational cost and improve MPC performance [18, 19]. Sonker et al. proposed a method for path tracking on uneven terrain using learning based MPC, which requires the measurement of terrain information ahead of the vehicle [20]. Besides, these methods usually require a large amount of pre-training and lack interpretability. Therefore, there is an urgent need for an efficient and accurate MPC path tracking method tailored to articulated steering vehicles.

As mining applications such as material transportation consistently need ASVs to repeatedly track the same path, presenting an opportunity to enhance tracking performance through repetitive. Similarly, in other applications that also

require repeated tracking, such as robotic arm control [21], CNC machining [22], additive manufacturing [23], researchers have applied iterative learning control (ILC) to reduce errors caused by model mismatch and disturbance. ILC does not rely on a precise model, and instead leverages deviations between actual value and desired value from prior iteration to correct control input in next iterations [24]. This iterative process enables the system error to converge, gradually improving tracking performance and achieving high-precision control.

ILC has drawn much attention in the field of autonomous driving due to its inherent self-learning capacity and ease of implementation. Dekker et al. combined PD iterative learning with feedback linearization control to correct vehicle path tracking errors [25]. Yang et al. used spatial-ILC to learn human driving behavior, allowing the controller to learn the steering input during manual driving [26]. Costa et al. used ILC to increase speed in a multi-lap car race [27]. It is important to recognize that ILC operates as a typical open-loop control method. Although it effectively diminishes repetition errors, it does not guarantee the closed-loop stability and robustness of the control system inherently.

In summary, it is difficult for ASVs to achieve accurate path tracking due to the non-negligible terrain disturbances. While complex modeling can reduce errors caused by disturbances, it also increases solving difficulty and the number of identification parameters. To solve the above problems, a novel iterative learning-MPC (IL-MPC) path tracking algorithm is proposed. The main contributions of this paper can be summarized below:

1) A novel application of IL-MPC tailored to articulated steering vehicles is proposed to mitigate path tracking errors resulting from simplified model and environmental uncertainties by iterative refinement.
2) An MPC based on a simplified kinematic model, with directly measured state variables, is designed to improve real-time performance. Furthermore, an iterative learning MPC with a feedforward-feedback framework and spatial indexing is introduced to utilize previous tracking data for better tracking performance.
3) The performance of the proposed algorithm are verified through co-simulation using Simulink and Adams, as well as field tests on different terrains and paths.

The rest of this paper is organized as follows. Section II describes the problem formulation, including a linear kinematic model and an error model. In Section III, the design process of the proposed IL-MPC controller is described. Simulation and field testing are conducted in Section IV with detailed data analysis on system performance. Finally, a brief conclusion and future work are given in Section V.

II. Problem Formulation

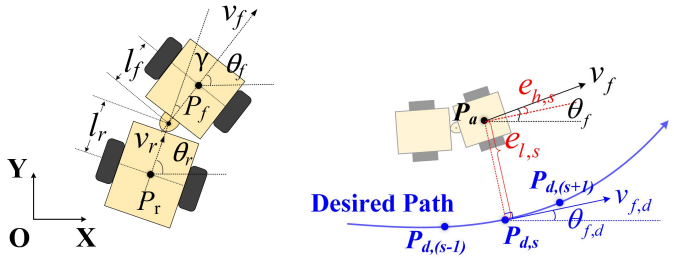
This section presents a linear kinematic model to establish the relationship between vehicle steering angle, velocity, and position. Subsequently, an error model is constructed to capture errors of distances and angles between desired path and actual path.

A. Discrete Kinematic Model

An ASV is composed of two parts, i.e., a front body and a rear body. The rotation between the two bodies makes the relationship between the steering angle and the heading angle differ from that of a front-wheel steering vehicle [28]. A specific model for ASVs is required for controller design. Existing models address vehicle dynamics on uneven terrains [29], but parameter identification and solution methods are complex. For feasibility and real-time performance, we developed a kinematic model assuming that the vehicle is a rigid body traveling forward without slip. The non-holonomic constraint is that no motion parallel to the wheel axles [10]. Table I lists the vehicle's variables and the proposed kinematic model are shown in Fig.1(a).

TABLE I
List of Nomenclature

Symbol	Description
$P_f(x_f, y_f)$ $P_r(x_r, y_r)$	Center points of the front and rear wheel axle
v_f , v_r	Velocities of the front and rear bodies
θ_f , θ_r	Heading angles of the front and rear bodies relative to the global coordinate system
l_f , l_r	Distances from the articulation point to front and rear axle
γ	Steering angle of the vehicle
ω	Steering angular velocity
e_l , e_h	Lateral error and heading error
s	Index of tracking point (subscript)
j	Index of iteration (subscript)



(a) (b)

Fig. 1. (a) Kinematic model of articulated steering vehicle (b) The error between the current point and desired path

Employing the center point of front wheel axle P_f as the reference point, $P_f(x_f, y_f)$ can be represented as:

$$\begin{cases} \dot{x}_f = v_f \cos\theta_f \\ \dot{y}_f = v_f \sin\theta_f \end{cases} \tag{1}$$

Assuming the vehicle is a rigid body, the front and rear body have the same velocity at the articulation point. Thus, relative velocity vector equation can be derived as:

$$\begin{cases} v_f = v_r \cos\gamma + \dot{\theta}_r l_r \sin\gamma \\ \dot{\theta}_f l_f = v_r \sin\gamma - \dot{\theta}_r l_r \cos\gamma \end{cases} \tag{2}$$

It can be seen in Fig.1(a) that

$$\theta_f = \theta_r - \gamma \tag{3}$$

From (2) and (3), the heading angles of the front body θ_f can be written as:

$$\dot{\theta}_f = \frac{v_f \sin\gamma + l_r\omega}{l_f\cos\gamma + l_r} \tag{4}$$

Combining (1) and (4), kinematic model is introduced as:

$$\begin{bmatrix}\dot{x}_f\\ \dot{y}_f\\ \dot{\theta}_f\\ \dot{\gamma}\end{bmatrix} = \begin{bmatrix}\cos\theta_f & 0\\ \sin\theta_f & 0\\ \dfrac{\sin\gamma}{l_f\cos\gamma + l_r} & \dfrac{l_r}{l_f\cos\gamma + l_r}\\ 0 & 1\end{bmatrix}\begin{bmatrix}v_f\\ \omega\end{bmatrix} \tag{5}$$

Assuming state variable $\boldsymbol{\chi} = [x_f, y_f, \theta_f, \gamma]^T$ and control variable $\boldsymbol{u} = [v_f, \omega]^T$, (5) is abbreviated as:

$$\dot{\boldsymbol{\chi}} = f(\boldsymbol{\chi}, \boldsymbol{u}) \tag{6}$$

To improve the calculation efficiency of controller, (6) is linearized and discretized. The linearization is realized by considering the relationship between current state $\boldsymbol{\chi}$ and desired state $\boldsymbol{\chi}_d$. The desired state is defined as:

$$\dot{\boldsymbol{\chi}}_d = f(\boldsymbol{\chi}_d, \boldsymbol{u}_d) \tag{7}$$

where $\boldsymbol{u}_d$ is the desired control variable.

Taylor expansion of (7) is performed at the desired point. A linear equation is obtained by ignoring higher-order terms.

$$\begin{aligned}\dot{\boldsymbol{\chi}} = f(\boldsymbol{\chi}_d, \boldsymbol{u}_d) &+ \left.\frac{\partial f(\boldsymbol{\chi}, \boldsymbol{u})}{\partial \boldsymbol{\chi}}\right|_{\substack{\boldsymbol{\chi}=\boldsymbol{\chi}_d\\ \boldsymbol{u}=\boldsymbol{u}_d}} (\boldsymbol{\chi} - \boldsymbol{\chi}_d)\\ &+ \left.\frac{\partial f(\boldsymbol{\chi}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{\chi}=\boldsymbol{\chi}_d\\ \boldsymbol{u}=\boldsymbol{u}_d}} (\boldsymbol{u} - \boldsymbol{u}_d)\end{aligned} \tag{8}$$

The rate of change of error between the current state and the desired state is define as:

$$\dot{\bar{\boldsymbol{\chi}}} = \boldsymbol{A}\bar{\boldsymbol{\chi}} + \boldsymbol{B}\bar{\boldsymbol{u}} \tag{9}$$

where $\bar{\boldsymbol{\chi}} = \boldsymbol{\chi} - \boldsymbol{\chi}_d$, $\bar{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_d$, $\boldsymbol{A} = \left.\dfrac{\partial f}{\partial \boldsymbol{\chi}}\right|_{\substack{\boldsymbol{\chi}=\boldsymbol{\chi}_d\\ \boldsymbol{u}=\boldsymbol{u}_d}}$ and $\boldsymbol{B} = \left.\dfrac{\partial f}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{\chi}=\boldsymbol{\chi}_d\\ \boldsymbol{u}=\boldsymbol{u}_d}}$.

According to forward Euler method, the kinematic model used in MPC is discretized for a step size T as:

$$\bar{\boldsymbol{\chi}}(k+1) = \boldsymbol{A}_k\bar{\boldsymbol{\chi}}(k) + \boldsymbol{B}_k\bar{\boldsymbol{u}}(k) \tag{10}$$

where $\boldsymbol{A}_k = \begin{bmatrix}1 & 0 & -v_{fd}\sin\theta_{fd}T & 0\\ 0 & 1 & v_{fd}\cos\theta_{fd}T & 0\\ 0 & 0 & 1 & \dfrac{v_{fd}(l_f + l_r\cos\gamma_d) + \omega_d l_f l_r\sin\gamma_d}{\left(l_f\cos\gamma_d + l_r\right)^2}T\\ 0 & 0 & 0 & 1\end{bmatrix}$,

$\boldsymbol{B}_k = \begin{bmatrix}\cos\theta_{fd}T & 0\\ \sin\theta_{fd}T & 0\\ \dfrac{\sin\gamma_d}{l_f\cos\gamma_d + l_r}T & \dfrac{l_r}{l_f\cos\gamma_d + l_r}T\\ 0 & T\end{bmatrix}$, $\bar{\boldsymbol{u}}(k) = \begin{bmatrix}v_f(k) - v_{fd}(k)\\ \omega(k) - \omega_d(k)\end{bmatrix}$,

$\bar{\boldsymbol{\chi}}(k) = \begin{bmatrix}x_f(k) - x_{fd}(k)\\ y_f(k) - y_{fd}(k)\\ \theta_f(k) - \theta_{fd}(k)\\ \gamma(k) - \gamma_d(k)\end{bmatrix}$, k denotes the step index.

In (10), x_f, y_f and θ_f are derived from the measurements of position and Euler angle data by an inertial navigation system (INS). γ are acquired through the measurement by a rotary encoder installed at the articulation point.

B. Error Model

ILC corrects control variables according to tracking error. We use the lateral error e_l and heading error e_h illustrated in Fig. 1(b) as error indicators. The subscript s denotes the index of tracking point, where $s = 1, 2, ..., S$. e_l represents distance error between current point P_a and closest point $P_{d,s}$ along the desired path. e_h represents angle error between vehicle heading and tangent to the closest point on the desired path. According to [2], the lateral and heading errors for point $P_{d,s}$ can be expressed as:

$$\begin{cases}e_{l,s} = -(x_f - x_{fd,s})\sin\theta_{fd,s} + (y_f - y_{fd,s})\cos\theta_{fd,s}\\ e_{h,s} = \theta_f - \theta_{fd,s}\\ \dot{e}_{l,s} = v_f\sin e_{h,s}\\ \dot{e}_{h,s} = \dot{\theta}_f - \dot{\theta}_{fd,s}\end{cases} \tag{11}$$

Let $\boldsymbol{\varepsilon}_s = [e_{l,s}, e_{h,s}]^T$. The relationship between $\boldsymbol{\varepsilon}_s$ and $\bar{\boldsymbol{\chi}}$ is:

$$\boldsymbol{\varepsilon}_s = \begin{bmatrix}-\sin\theta_{fd,s} & \cos\theta_{fd,s} & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\bar{\boldsymbol{\chi}} = \boldsymbol{D}\bar{\boldsymbol{\chi}} \tag{12}$$

III. Path Tracking Controller Design

In this section, we present the design process of the proposed IL-MPC controller. As shown in Fig. 2, the controller consists of an MPC feedback controller and an ILC feedforward controller.

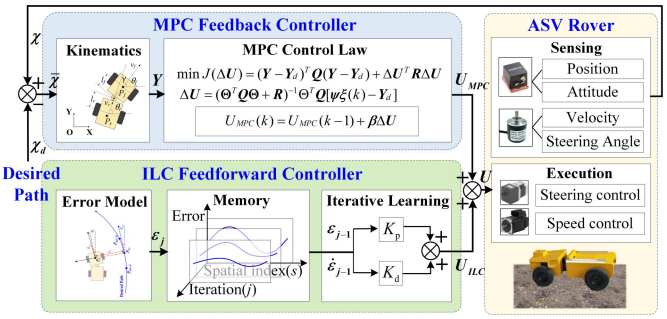


Fig. 2. Schematics of the proposed IL-MPC

Firstly, MPC performs optimization over a specified time horizon based on the linear kinematic model. It calculates the control variable U_{MPC} that corresponds to the desired path to ensure system stability. Secondly, as the simplified model in MPC may bring tracking errors (analyzed in Section I), ILC is incorporated as a feedforward controller to correct the control variable based on the previous tracking errors $\boldsymbol{\varepsilon}_{j-1}$, which are

calculated by error model and stored in memory. Finally, the control variables of feedforward and feedback controller are added together as the control input to the ASV. It is noteworthy that measuring and implementing the input and output variables of the controller can be easy.

A. Design of model predictive control

It is necessary to build a vehicle state predictive model based on kinematic model to predict the state at the future moment. We define a new state variable vector $\boldsymbol{\xi}(k)=\begin{bmatrix}\bar{\boldsymbol{\chi}}(k)\\ \bar{\boldsymbol{u}}(k-1)\end{bmatrix}$ and an output vector $\boldsymbol{\eta}(k)=\bar{\boldsymbol{\chi}}(k)$. Based on the model in (10), the new state variable is calculated as:

$$\boldsymbol{\xi}(k+1)=\bar{\boldsymbol{A}}\boldsymbol{\xi}(k)+\bar{\boldsymbol{B}}\Delta\boldsymbol{u}(k) \tag{13}$$

$$\boldsymbol{\eta}(k)=\bar{\boldsymbol{C}}\boldsymbol{\xi}(k) \tag{14}$$

where

$$\Delta\boldsymbol{u}=\bar{\boldsymbol{u}}(k)-\bar{\boldsymbol{u}}(k-1)\quad,\quad \bar{\boldsymbol{A}}=\begin{bmatrix}\boldsymbol{A}_k & \boldsymbol{B}_k\\ \boldsymbol{0}_{m\times n} & \boldsymbol{I}_m\end{bmatrix}\quad,\quad \bar{\boldsymbol{B}}=\begin{bmatrix}\boldsymbol{B}_k\\ \boldsymbol{I}_m\end{bmatrix}\quad,$$

$\bar{\boldsymbol{C}}=\begin{bmatrix}\boldsymbol{I}_n & \boldsymbol{0}_m\end{bmatrix}$, n and m represent dimensions of state and control variables.

Let N_p be predictive horizon and N_c be control horizon. By expanding (13) and (14), we obtain the state predictive equation at future time steps:

$$\boldsymbol{Y}=\boldsymbol{\Psi}\boldsymbol{\xi}(k)+\boldsymbol{\Theta}\Delta\boldsymbol{U} \tag{15}$$

where

$$\boldsymbol{\Psi}=\begin{bmatrix}\bar{\boldsymbol{C}}\bar{\boldsymbol{A}}\\ \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^2\\ \vdots\\ \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_c}\\ \vdots\\ \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_p}\end{bmatrix},\boldsymbol{\Theta}=\begin{bmatrix}\bar{\boldsymbol{C}}\bar{\boldsymbol{B}} & 0 & 0 & \cdots & 0\\ \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}\bar{\boldsymbol{B}} & \bar{\boldsymbol{C}}\bar{\boldsymbol{B}} & 0 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_c-1}\bar{\boldsymbol{B}} & \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_c-2}\bar{\boldsymbol{B}} & \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_c-3}\bar{\boldsymbol{B}} & \cdots & \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{0}\bar{\boldsymbol{B}}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_p-1}\bar{\boldsymbol{B}} & \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_p-2}\bar{\boldsymbol{B}} & \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_p-3}\bar{\boldsymbol{B}} & \cdots & \bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N_p-N_c}\bar{\boldsymbol{B}}\end{bmatrix},$$

$$\Delta\boldsymbol{U}=\begin{bmatrix}\Delta\bar{\boldsymbol{u}}(k)\\ \Delta\bar{\boldsymbol{u}}(k+1)\\ \Delta\bar{\boldsymbol{u}}(k+2)\\ \vdots\\ \Delta\bar{\boldsymbol{u}}(k+N_c-1)\end{bmatrix},\ \boldsymbol{Y}=\begin{bmatrix}\boldsymbol{\eta}(k+1)\\ \boldsymbol{\eta}(k+2)\\ \boldsymbol{\eta}(k+3)\\ \vdots\\ \boldsymbol{\eta}(k+N_p)\end{bmatrix}.$$

To solve $\Delta\boldsymbol{U}$ that minimizes tracking error and the change of control variables within predictive horizon N_p at time step k, we refer to [30] to define the cost function as follows:

$$\min J(\Delta\boldsymbol{U})=(\boldsymbol{Y}-\boldsymbol{Y}_d)^T\boldsymbol{Q}(\boldsymbol{Y}-\boldsymbol{Y}_d)+\Delta\boldsymbol{U}^T\boldsymbol{R}\Delta\boldsymbol{U} \tag{16}$$

where $\boldsymbol{Q}\in\mathbb{R}^{2N_p\times 2N_p}$ is error weight matrix and $\boldsymbol{R}\in\mathbb{R}^{N_p\times N_p}$ is control weight matrix.

Substituting (15) to (16) and calculating the optimal $\Delta\boldsymbol{U}$ by solving $\frac{\partial J}{\partial\Delta\boldsymbol{U}}=0$ as:

$$\Delta\boldsymbol{U}=(\boldsymbol{\Theta}^T\boldsymbol{Q}\boldsymbol{\Theta}+\boldsymbol{R})^{-1}\boldsymbol{\Theta}^T\boldsymbol{Q}[\boldsymbol{\psi}\boldsymbol{\xi}(k)-\boldsymbol{Y}_d] \tag{17}$$

In practice, only the 1st variable in sequence calculated by (17) will be used to update MPC control variable and the other variables are just used for state prediction. Thus, the final control variable of MPC can be acquired as:

$$\boldsymbol{U}_{MPC}(k)=\boldsymbol{U}_{MPC}(k-1)+\boldsymbol{\beta}\Delta\boldsymbol{U} \tag{18}$$

where $\boldsymbol{\beta}=[\underbrace{1,0,\cdots,0}_{N_c}]$ and $\boldsymbol{U}_{MPC}(k-1)$ is the control variable of MPC in the last time.

Since the cost function is based on linear model without state and input constraints, the solution is done in a single step to minimize the computational cost. Although constraints are not considered in the solution, control inputs are limited after feed-forward corrections to ensure proper actuator operation. During field tests, the speed and the rate change of the steering angle were constrained within ±1 m/s and ±0.5 rad/s, respectively.

B. Design of iterative learning control

Considering vehicle dynamics becomes essential during off-road travel, but this will raise MPC solving burden. Thus, we introduced ILC as a feedforward controller, which enhances accuracy without relying on dynamic model. ILC leverages past tracking errors to derive reference information and improves performance by integrating tracking error data into subsequent iterations. In the limit of an infinite number of iterations, the vehicle state ideally converges to the desired state, as depicted in (19).

$$\lim_{j\to\infty}\boldsymbol{\chi}_j=\boldsymbol{\chi}_d \tag{19}$$

When ILC is designed independently, it is usually expressed as:

$$\boldsymbol{u}_j=\boldsymbol{F}\boldsymbol{u}_{j-1}+\boldsymbol{L}\boldsymbol{\varepsilon}_{j-1} \tag{20}$$

where $\boldsymbol{F}$ is a filter matrix, $\boldsymbol{u}_{j-1}$ is control variable from the previous iteration, $\boldsymbol{L}$ is a learning matrix, and j is iterative index [31].

To improve the control performance of MPC without relying on high-precision models, MPC is combined with ILC. Taking $\boldsymbol{U}_{MPC}$ to instead $\boldsymbol{u}_{j-1}$, and modifying matrix $\boldsymbol{L}$. A PD-type ILC controller with MPC is obtained as

$$\boldsymbol{U}=\boldsymbol{U}_{MPC}+K_p\boldsymbol{\varepsilon}_{(j-1)s}+K_d\dot{\boldsymbol{\varepsilon}}_{(j-1)s} \tag{21}$$

where $\boldsymbol{U}$ is the control variable of IL-MPC, $\boldsymbol{\varepsilon}_{(j-1)s}$ is the error at the same point in the last iteration, K_p and K_d are learning gains.

Under the premise of tracking the same path, errors caused by terrain disturbances and model simplifications are recorded for subsequent correction of MPC control inputs, as outlined in (21). Path tracking for vehicles differs from cycle control in industrial production [23], which runs strictly following the cycle time. It is difficult for vehicles to keep the same position and attitude at the same moment when tracking the path repeatedly. Using the temporal index to record and recall prior error information may lead to data mismatch.

Although vehicles may not have the same position and attitude at the same moment in different trials, vehicle will travel along desired path under the action of the controller. In addition, errors at the same point are approximately equal when operation environment is unchanged.

Therefore, we consider path tracking to exhibit spatially

repetitive and adopt a spatial indexing method for error recording, where indexing is based on the desired path point number s. Fig.3 shows the structure of the error memory. At the start of each iteration, the error recorded last iteration is assigned to the error memory of new trial to prevent the clearance of errors not tracked in the current iteration. Then the error memory is calculated by (11) and recorded by index of the closest point. Based on the error memory, control value U is corrected by (21) when tracking to the same point again.

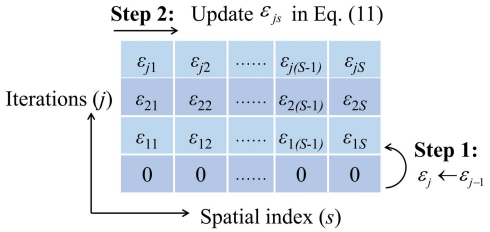


Fig. 3. Error memory by spatial indexing

We analyze the convergence of iterative learning to ensure that the algorithm corrects errors when it repeatedly tracks the path.

Lemma 1: For linear systems $z_{k+1} = \Gamma \cdot z_k$ when $k \to \infty$, the sufficient condition for the system to asymptotically converge to zero is $\rho(\Gamma) < 1$ (where $\rho(\Gamma)$ represents the spectral radius of the matrix Γ) [31].

Assumption 1: The initial conditions and the starting points remain consistent when tracking the same path, namely:

$$\chi_j(0) = \chi_{j+1}(0) = \chi_d(0) \tag{22}$$

Theorem 1: When assumption is satisfied, the sufficient condition for $\lim_{j\to\infty} \chi_j = \chi_d$ to be true is

$$\rho(\boldsymbol{I} - K_d \bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}) < 1 \tag{23}$$

Proof: Let the error $\boldsymbol{E}_{j+1}(k)$ of the $j+1$ iteration be:

$$\begin{aligned} \boldsymbol{E}_{j+1}(k) &= \boldsymbol{\eta}_d(k) - \boldsymbol{\eta}_{j+1}(k) \\ &= \boldsymbol{\eta}_d(k) - \boldsymbol{\eta}_j(k) + \boldsymbol{\eta}_j(k) - \boldsymbol{\eta}_{j+1}(k) \\ &= \boldsymbol{E}_j(k) - \bar{\boldsymbol{C}}\Delta\boldsymbol{\xi}_j(k) \end{aligned} \tag{24}$$

where $\Delta\boldsymbol{\xi}_j(k) = \boldsymbol{\xi}_{j+1}(k) - \boldsymbol{\xi}_j(k)$.

From (10) and (12), the relationship between the error $\boldsymbol{E}(k)$ and path tracking error is

$$\boldsymbol{\varepsilon}(k) = \boldsymbol{D}\boldsymbol{E}(k) \tag{25}$$

From (13), the relationship between $\Delta\boldsymbol{\xi}$ and $\Delta\boldsymbol{u}$ can be represented by the formula:

$$\Delta\boldsymbol{\xi}_j(k+1) = \bar{\boldsymbol{A}}\Delta\boldsymbol{\xi}_j(k) + \bar{\boldsymbol{B}}\Delta(\Delta\boldsymbol{u}_j(k)) \tag{26}$$

Combining (21) and (25), it follows that:

$$\Delta(\Delta\boldsymbol{u}_j(k)) = K_p\boldsymbol{D}\boldsymbol{E}_j(k) + K_d\boldsymbol{D}(\boldsymbol{E}_j(k+1) - \boldsymbol{E}_j(k)) \tag{27}$$

Combining (26) and (27), it follows that:

$$\Delta\boldsymbol{\xi}_j(k+1) = \bar{\boldsymbol{A}}\Delta\boldsymbol{\xi}_j(k) + K_p\bar{\boldsymbol{B}}\boldsymbol{D}\boldsymbol{E}_j(k) + K_d\bar{\boldsymbol{B}}\boldsymbol{D}\delta\boldsymbol{E}_j(k+1) \tag{28}$$

where $\delta\boldsymbol{E}_j(k+1) = \boldsymbol{E}_j(k+1) - \boldsymbol{E}_j(k)$

From *Assumption 1* we know $\Delta\boldsymbol{\xi}_j(0) = 0$ and $\boldsymbol{E}_j(0) = 0$, Eq.(24) is extended to:

$$\begin{cases} \boldsymbol{E}_{j+1}(1) = \boldsymbol{E}_j(1) - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}\delta\boldsymbol{E}_j(1) \\ \boldsymbol{E}_{j+1}(2) = [\boldsymbol{I} - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}]\boldsymbol{E}_j(2) + K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}\boldsymbol{E}_j(1) \\ \qquad -K_p\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}\boldsymbol{E}_j(1) - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{A}}\bar{\boldsymbol{B}}\boldsymbol{D}\delta\boldsymbol{E}_j(1) \\ \qquad \vdots \\ \boldsymbol{E}_{j+1}(N) = [\boldsymbol{I} - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}]\boldsymbol{E}_j(N) + K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}\boldsymbol{E}_j(N-1) \\ \qquad -\bar{\boldsymbol{C}}[K_d\bar{\boldsymbol{A}}^{N-1}\bar{\boldsymbol{B}}\boldsymbol{D}\boldsymbol{E}_j(1) + K_d\bar{\boldsymbol{A}}^{N-2}\bar{\boldsymbol{B}}\boldsymbol{D}\delta\boldsymbol{E}_j(2) + \ldots \\ \qquad +K_d\bar{\boldsymbol{A}}\bar{\boldsymbol{B}}\boldsymbol{D}\delta\boldsymbol{E}_j(N-1)] - \bar{\boldsymbol{C}}[K_p\boldsymbol{A}^{N-2}\bar{\boldsymbol{B}}\boldsymbol{E}_j(1) \\ \qquad +K_p\boldsymbol{A}^{N-3}\bar{\boldsymbol{B}}\boldsymbol{E}_j(2) + \ldots + K_p\bar{\boldsymbol{B}}\boldsymbol{E}_j(N-1)] \end{cases} \tag{29}$$

Abbreviated as:

$$\boldsymbol{M}_{j+1} = \boldsymbol{Z}\boldsymbol{M}_j \tag{30}$$

where $\boldsymbol{M}_j = [\boldsymbol{E}_j(1), \boldsymbol{E}_j(2), \ldots, \boldsymbol{E}_j(N)]^T$,

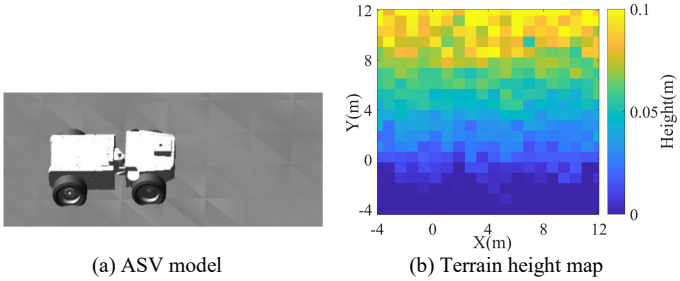
$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{I} - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D} & 0 & \cdots & 0 \\ K_d\bar{\boldsymbol{C}}(\bar{\boldsymbol{B}}\boldsymbol{D} - \bar{\boldsymbol{A}}\bar{\boldsymbol{B}}\boldsymbol{D}) - K_p\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D} & \boldsymbol{I} - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D} & \cdots & 0 \\ -K_d\bar{\boldsymbol{C}}(\bar{\boldsymbol{A}}^2\bar{\boldsymbol{B}}\boldsymbol{D} + \boldsymbol{A}\bar{\boldsymbol{B}}\boldsymbol{D}) - K_p\bar{\boldsymbol{C}}\bar{\boldsymbol{A}}\bar{\boldsymbol{B}}\boldsymbol{D} & K_d\bar{\boldsymbol{C}}(\bar{\boldsymbol{B}}\boldsymbol{D} - \bar{\boldsymbol{A}}\bar{\boldsymbol{B}}\boldsymbol{D}) - K_p\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -K_d\bar{\boldsymbol{C}}(\bar{\boldsymbol{A}}^{N-1}\bar{\boldsymbol{B}}\boldsymbol{D} + \boldsymbol{A}^{N-2}\boldsymbol{B}\boldsymbol{D}) - K_p\bar{\boldsymbol{C}}\bar{\boldsymbol{A}}^{N-2}\bar{\boldsymbol{B}}\boldsymbol{D} & \cdots & \cdots & \boldsymbol{I} - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D} \end{bmatrix}.$$

We know from *Lemma 1* that (30) converges when $\rho(\boldsymbol{Z}) < 1$, that is $\rho(\boldsymbol{I} - K_d\bar{\boldsymbol{C}}\bar{\boldsymbol{B}}\boldsymbol{D}) < 1$. ■

IV. Results and Discussion

A. Simulation tests

In this part, the performance of IL-MPC algorithm is validated through co-simulation using Simulink and Adams. The Adams model of an ASV is employed as the controlled object in simulation tests. Its basic parameters are shown in Table II, Details of the model are described in [32]. The traveling terrain with random heights ranges from 0 to 0.1m serving as external disturbances. Fig.4 shows the Adams model and the terrain height map in the test.



(a) ASV model (b) Terrain height map

Fig. 4. The Adams model and test terrain

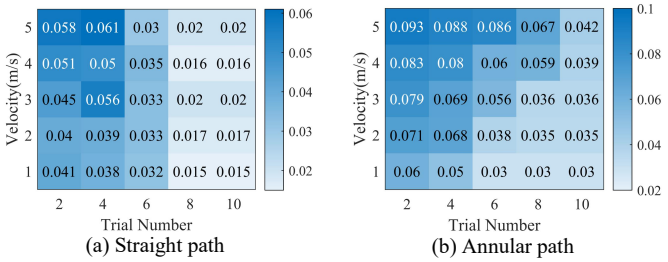
TABLE II
Main Parameters of Vehicle Model

Parameters	Unit	Value
Distance from articulated point to front axle (l_f)	m	1.68
Distance from articulated point to rear axle (l_r)	m	1.87
Maximum velocity (v)	m·s^{-1}	10
Maximum steering angle (γ)	rad	0.52

To evaluate the tracking performance of the controllers, we conduct experiments on both straight and annular paths, and make a comparison with the traditional MPC algorithm. The sampling frequency is 10Hz. The predictive horizon and control horizon of controllers are set to 10 and 5 steps. Five sets of experiments are carried out at different velocities, ranging from 1 m/s to 5 m/s with an interval of 1 m/s between each set. e_l and e_h are calculated using (11). Their RMS (root mean square) values are employed to evaluate the controllers. The RMS error for a total number of steps N in a trace is given as follows:

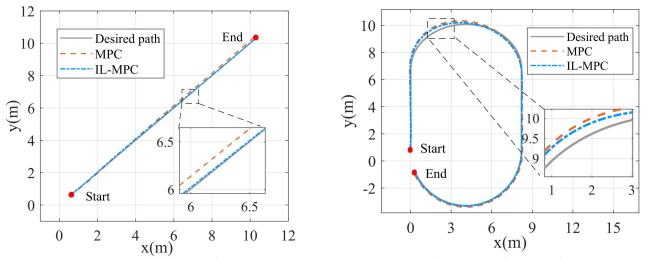
$$e_{RMS} = \sqrt{\frac{1}{N}\sum_{k=1}^{N} e(k)^2} \tag{31}$$

The RMS values of lateral error under various velocities and iterations are depicted in Fig. 5. The lateral errors increase with velocity. IL-MPC reduces tracking errors through iteration, resulting in similar error levels across different velocities. Fig.6 and Fig.7 present tracking results at a velocity of 5m/s. It's evident that MPC exhibits larger tracking errors than IL-MPC. At marked points along the annular path influenced by disturbances (terrain height around 0.1m) and the change in curvature, MPC reaches a maximum lateral error of 0.29m, whereas the error of IL-MPC achieves only 0.14m. At the second turn, the terrain height is approximately 0.01m and tracking errors reduce. The tracking results demonstrate that IL-MPC exhibits better robustness than MPC.



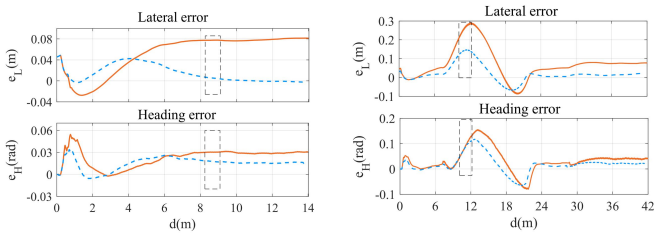
(a) Straight path (b) Annular path

Fig. 5. The RMS of lateral error under different velocities and iterations.



(a) Straight path (b) Annular path

Fig. 6 Path Tracking results.



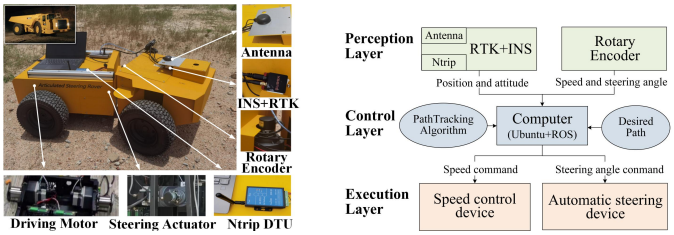
(a) Straight path tracking (b) Annular path tracking

Fig. 7 Lateral error and heading error. The solid orange line represents the results of MPC, and the dashed blue line represents the results of IL-MPC.

B. Field experiment setup

Field tests using an articulated steering rover further validate the robustness and precision of IL-MPC, comparing it with PID [3], traditional MPC [11] and feedback linearized iterative learning (FBL-ILC) [25]. To validate the algorithm's robustness against disturbances, experiments were conducted on both flat road and off-road terrain, as shown in Fig. 9.

The desired tracking path consists of a U-shaped path and a curvature-varying path, with a velocity of 1m/s. The U-shaped path involves three segments of straight lines, each having a length of 6m, and two semicircular arcs with a radius of 2 meters. The path-planner generated the curvature-varying path [11], with the start point at $(0.1, 0.4)$. We conduct three sets of experiments: tracking the U-shaped path on flat terrain (flat-U), tracking the U-shaped path on off-road terrain (off-road-U), and tracking the curvature-varying path on off-road terrain (off-road-C). For all the experiments, the desired path is defined by plane coordinates (x, y) and heading angle θ . The initial state of the rover is $(x, y, \theta) = (0, 0, 0)$.



(a) Articulated steering rover (b) System hardware framework

Fig. 8 The vehicle platform used in testing

Fig.9 The test site in the field experiments

TABLE III

THE KEY PARAMETERS OF THE ROVER

Parameters	Unit	Value
External dimensions	mm	1250×710×410
Weight	kg	114.25
Front wheel tread	mm	555
Rear wheel tread	mm	585
Distance from front axle to articulated point	mm	287
Distance from rear axle to articulated point	mm	475
Steering range	rad	-0.52 ~ 0.52
Velocity	$m \cdot s^{-1}$	0 ~ 2.2

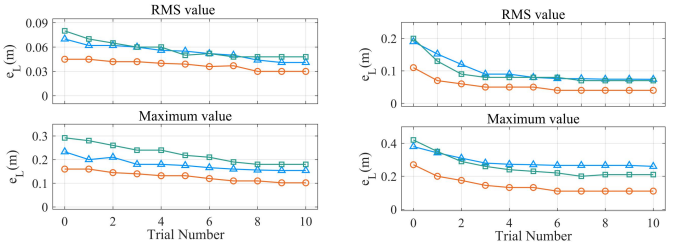
Fig. 8 shows articulated steering rover used in this test. The perception layer consists of three rotary encoders and an INS with RTK. The rotary encoders are used to measure the rover's steering angle and velocity. Xsens MTi-680G RTK/INS, installed at the center point of the front wheel axle, is used to collect filtered position (longitude and latitude) and heading angle data at a sampling frequency of 10Hz. Combined with Ntrip DTU, INS achieves a positioning accuracy of 1 cm + 1 ppm CEP and a heading accuracy of 1°.

The control layer is composed of a computer with an Intel Core i7 - 8550U processor and 8GB of RAM. The path tracking algorithm runs under Ubuntu 18.04+ROS Melodic system, executing the path tracking algorithm in real-time. In the execution layer, movement and steering are performed by high-power servo motors. A PID algorithm is used to implement closed-loop control of steering angle and speed. Table III shows the key parameters of the rover.

C. Analysis of the Field Experiments

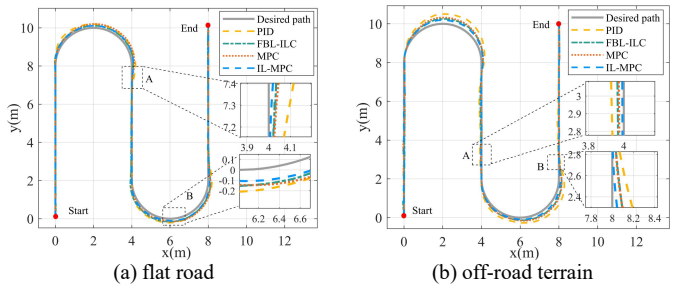
FBL-ILC, like IL-MPC, requires iterative processes to reduce tracking errors. In the context of identical starting conditions, we perform 10 tracking trials. Fig. 10 shows the RMS and maximum value of the lateral error in each trial. Notably, the initial error of FBL-ILC is greater than that of IL-MPC. The errors decrease as the number of iterations increases. Within 10 trials, the error converges to a low level, demonstrating their capability in reducing tracking errors during repetitive tracking tasks.

Fig.11 and 12 show the results of tracking the U-shaped path. Notably, the tracking performances of all algorithms exhibit similarities when following a straight line on flat road. However, deviations become more pronounced during turning maneuvers. From the enlarged map A, it is apparent that IL-MPC demonstrates superior tracking performance, enabling faster and smoother transitions into straight-line driving areas. Additionally, from the enlarged map B, it is observed that IL-MPC requires less turning space during maneuvers.



(a) IL-MPC (b) FBL-ILC

Fig. 10 RMS and maximum values of lateral errors for learning iterations. Orange line with circular markers represents the result of flat-U, blue line with triangular markers represents the result of off-road-U and green line with square markers is the result of off-road-C (Excluding start point error).

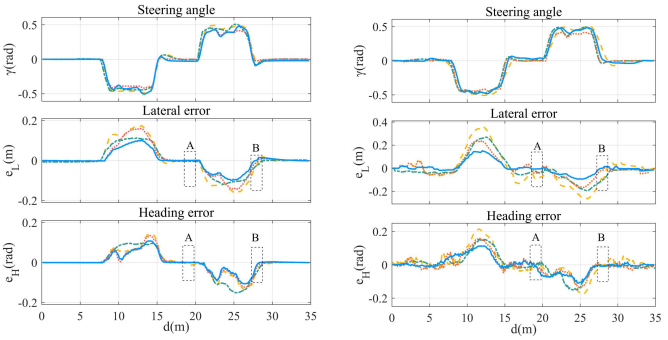


(a) flat road (b) off-road terrain

Fig. 11 U-shaped tracking paths.

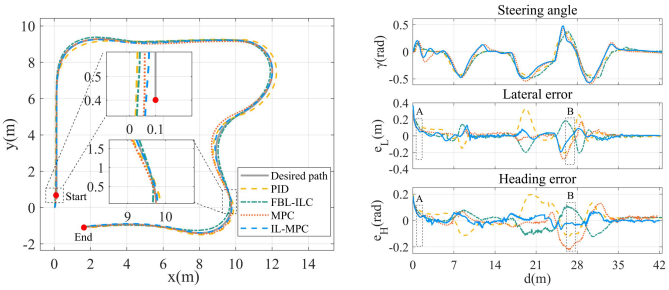
Different from tracking on flat road, the enlarged map A in Fig.11(b) shows slight deviations on straight path for all three algorisms, which are caused by the terrain disturbances. Notably, IL-MPC demonstrates the least deviation among them. The enlarged map B shows larger steering overshoots on the curved path, among which PID shows the greatest impact with an increase in tracking errors. Furthermore, Fig.12(b) shows the heading and lateral errors when the vehicle are operated on off-road terrain. Compared to driving on flat roads, the heading angle data exhibits greater noise. In summary, IL-MPC overperformed PID, FBL-ILC and MPC by highlighting its capacity to effectively correct tracking errors induced by disturbances.

In the curvature-varying path tracking experiment, although the path's starting point differs from the rover's initial position, the results at the marked range A in Fig. 13 demonstrate that all algorithms converge to the desired path. Notably, IL-MPC exhibits the smallest error among them. Within the variable curvature segment (17-35m), all algorithms show increased errors. However, IL-MPC produces the smallest number of errors. Due to the large errors in the first trial, FBL-ILC experiences overcompensation during iterative correction, resulting in tracking errors that are opposite to those of other algorithms in the variable curvature segment. The tracking results show that our proposed algorithm demonstrates good tracking performance in curvature-varying path on off-road terrain.



(a) flat road (b) off-road terrain

Fig. 12 U-shaped tracking input values and tracking errors. Solid blue line represents the results of proposed IL-MPC, chain orange line represents the results of MPC, dash-dot green line represents the results of FBL-ILC and dashed yellow line represents the results of PID.



(a) tracking path (b) input value and tracking errors

Fig. 13 The result of off-road-C. The solid blue line represents IL-MPC, the chain orange line represents MPC, the dash-dot green line represents FBL-ILC, and the dashed yellow line represents PID.

Table IV summarizes the lateral tracking errors for all three sets of field experiments after 10 cycles of iterative learning. In the flat road test, the surface imposes minimal disturbances on the rover. The proposed IL-MPC achieves significant improvements i.e., a reduction of 37.5% and 25% in the maximum value and RMS value of the lateral error, respectively. When tracking path on off-road terrains, all algorithms exhibit increased errors due to bumps and slippage.

Even under these challenging conditions, IL-MPC outperforms PID, FBL-ILC and MPC. Notably, the maximum and the RMS value of the lateral error are reduced by 34.8% and 42.8% when tracking the U-path. In tracking curvature-varying paths, IL-MPC also exhibits the smallest tracking errors. These results indicate the proposed IL-MPC algorithm can effectively mitigate tracking errors through iterative processes. Besides, IL-MPC can meet real-time requirements at 10Hz, like the MPC with a simplified model.

TABLE IV
LATERAL ERROR IN FIELD TESTS

	Test No.	PID	FBL-ILC	MPC	IL-MPC
RMS	Flat-u	0.05	0.04	0.04	0.03
	rough-u	0.09	0.07	0.07	0.04
	Rough-c	0.09	0.07	0.08	0.05
Max	Flat-u	0.17	0.11	0.16	0.10
	rough-u	0.35	0.26	0.23	0.15
	Rough-c	0.33	0.21	0.29	0.18

V. CONCLUSION

This letter presents a novel IL-MPC (Iterative Learning-Model Predictive Control) path tracking algorithm that addresses tracking errors resulting from simplified models and external disturbances through iterative learning. The algorithm is specifically designed for articulated steering vehicles using a kinematic model. It achieves accurate tracking without the need for high-precision dynamic modeling. The convergence of the proposed IL-MPC was first validated through both theoretical proofs and simulation tests. Then, field tests demonstrated that the IL-MPC outperforms PID, FBL-ILC, and traditional MPC in terms of tracking performance. This improvement in accuracy is beneficial for engineering vehicles in transportation and other operational scenarios.

Future works will focus on reducing the amount of error data stored in memory to alleviate cache burdens. Additionally, we plan to explore the application of the IL-MPC path tracking algorithm in various field operations, such as non-repetitive tasks on hills and mountains.

REFERENCES

[1] X. Chen et al., "A novel data-driven rollover risk assessment for articulated steering vehicles using RNN," *J Mech. Sci. Technol,* vol. 34, no. 5, pp. 2161-2170, 2020.

[2] J. Marshall, T. Barfoot, and J. Larsson, "Autonomous underground tramming for center-articulated vehicles," *J Field. Robot,* vol. 25, no. 6-7, pp. 400-421, 2008.

[3] Rains, C. et al., "Evaluation of a Simple Pure Pursuit Path-Following Algorithm for an Autonomous, Articulated-Steer Vehicle," *Applied Engineering in Agriculture,* pp. 367-374, 2014.

[4] S. Guan *et al*., "Dynamic hyperparameter tuning-based path tracking control for robotic rollers working on earth-rock dam under complex construction conditions," *Automat. Constr.,* vol. 143, 2022.

[5] T. Nayl *et al*., "Design and experimental evaluation of a novel sliding mode controller for an articulated vehicle," *Robot Auton. Syst.,* vol. 103, pp. 213-221, 2018.

[6] X. Zhao et al., "Feedback Linearization Control for Path Tracking of Articulated Dump Truck," *TELKOMNIKA,* vol. 13, no. 3, 2015.

[7] Y. Bian *et al*., "Kinematics and Path Following Control of an Articulated Drum Roller," *Chin. J. Mech. Eng-En,* vol. 30, no. 4, pp. 888-899, 2017.

[8] H. Yu *et al*., "Pre-Work for the Birth of Driver-Less Scraper (LHD) in the Underground Mine: The Path Tracking Control Based on an LQR Controller and Algorithms Comparison," *Sensors,* vol. 21, no. 23, 2021.

[9] T. Nayl *et al*., "Effect of kinematic parameters on MPC based on-line motion planning for an articulated vehicle," *Robot Auton Syst,* vol. 70, pp. 16-24, Aug 2015.

[10]T. Nayl, G. Nikolakopoulos, and T. Gustafsson, "A full error dynamics switching modeling and control scheme for an articulated vehicle," *Int. J. Control Autom.,* vol. 13, no. 5, pp. 1221-1232, 2015.

[11]R. Song *et al.*, "Autonomous Wheel Loader Trajectory Tracking Control Using LPV-MPC,", *2022 American Control Conference (Acc), Atlanta, GA, USA,* pp. 2063-2069, 2022.

[12]Bai G, *et al.,* "Path tracking of mining vehicles based on nonlinear model predictive control," *Applied Sciences*, vol. 9, no.7, 2019.

[13]J. Shi *et al.*, "Planning the trajectory of an autonomous wheel loader and tracking its trajectory via adaptive model predictive control," *Robot Auton Syst,* vol. 131, 2020.

[14]A. Shahirpour and D. Abel, "Simulation and Successive Sideslip-Compensating Model Predictive Control for Articulated Dump Trucks," IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 2022, pp. 3907-3913.

[15]K. Iagnemma *et al*, "Online Terrain Parameter Estimation for Wheeled Mobile Robots With Application to Planetary Rovers," *IEEE T. Robot,* vol. 20, no. 5, pp. 921-927, 2004.

[16]M. Fnadi *et al.*, "Constrained Model Predictive Control for dynamic path tracking of a bi-steerable rover on slippery grounds," *Control Eng. Pract.,* vol. 107, 2021.

[17]L. Li, *et al*, "Trajectory tracking control for a wheel mobile robot on rough and uneven ground," *Mechatronics,* vol. 83, 2022.

[18]Norouzi A, *et al.*, "Integrating machine learning and model predictive control for automotive applications: A review and future directions," Engineering Applications of Artificial Intelligence, vol. 120, 2023.

[19]R. Yang *et al.*, "Learning-Based Predictive Path Following Control for Nonlinear Systems Under Uncertain Disturbances," *IEEE Robot Autom. Let,* vol. 6, no. 2, pp. 2854-2861, 2021.

[20] R. Sonker and A. Dutta, "Adding Terrain Height to Improve Model Learning for Path Tracking on Uneven Terrain by a Four Wheel Robot," *IEEE Robot Autom. Let.,* vol. 6, no. 1, pp. 239-246, 2021.

[21]C. Hu *et al.*, "Task Space Contouring Error Estimation and Precision Iterative Control of Robotic Manipulators," *IEEE Robot Autom. Let,* vol. 7, no. 3, pp. 7826-7833, 2022.

[22]Z. Wang *et al.*, "Newton-ILC Contouring Error Estimation and Coordinated Motion Control for Precision Multiaxis Systems With Comparative Experiments,", *IEEE T. Ind. Electron.,* vol. 65, no. 2, pp. 1470-1480, Feb 2018.

[23]Z. Afkhami, *et al*., "Robust Higher-Order Spatial Iterative Learning Control for Additive Manufacturing Systems," *IEEE T Contr. Syst. T.,* vol. 31, no. 4, pp. 1692-1707, 2023.

[24]D. A. Bristow, *et al.*, "A survey of iterative learning control,", *IEEE Contr. Syst. Mag.,* vol. 26, no. 3, pp. 96-114, Jun 2006.

[25]L. G. Dekker, *et al.* , "Experiments in feedback linearized iterative learning-based path following for center-articulated industrial vehicles," *J Field Robot,* vol. 36, no. 5, pp. 955-972, 2019.

[26]L. Yang, *et al*, "Iterative learning of an unknown road path through cooperative driving of vehicles," *Iet Intell. Transp. Sy.,* vol. 14, no. 5, pp. 423-431, 2020.

[27]G. Costa *et al.*, "Online learning of MPC for autonomous racing," *Robot Auton Syst,* vol. 167, 2023.

[28]M. Murillo *et al.*, "Improving path-tracking performance of an articulated tractor-trailer system using a non-linear kinematic model," *Comput. Electron. Agr.,* vol. 196, 2022.

[29]Y. Gao *et al.*, "Path-following control by dynamic virtual terrain field for articulated steer vehicles," *Veh. Syst. Dyn.,* vol. 58, no. 10, pp. 1528-1552, 2020.

[30]Y. Ding *et al.*, "Model predictive control and its application in agriculture: A review," *Comput. Electron. Agr.,* vol. 151, pp. 104-117, 2018.

[31]H.S. Ahn, *et al.*, *Iterative learning control: robustness and monotonic convergence for interval systems*. Springer, 2007.

[32]C. Yang *et al.*, "An unscented Kalman filter based velocity estimation method for articulated steering vehicle using a novel dynamic model," *P. I. Mech. Eng. K-J. Mul.,* vol. 237, no. 3, pp. 389-405, 2023.