EVS35 Symposium
Oslo, Norway, June 11-15, 2022

A double-layer data-driven motion planning and control method for parallel parking

Shaoyu Song, Hui Chen
Shaoyu Song (corresponding author), School of Automotive Studies, Tongji University, Shanghai, 201804, China, s.song@tongji.edu.cn

Executive Summary

Automated parking system (APS) is essential to an intelligent electric vehicle, especially for electric vehicles (EVs) with wireless charging solutions using wireless power transfer (WPTS) technology. In this study, the data generated by the onboard sensors and off-line simulation are used to improve the parking performance. The upper-level motion planner learns the trajectory planned by nonlinear programming. The lower-level trajectory-following control method combines model predictive control and iterative learning control. The proposed method is tested in Simulink and high-fidelity CarSim simulation and compared with the open-loop online planning method. The mean precisions of parking position in the *y*-axis for one-layer open-loop control and double-layers control are 0.010 m and 0.016 m, respectively. And the parking time of the double-layer data-driven method is shorter than the open-loop method. Moreover, the adaptability to the different initial positions, parking slot size, and road conditions has been confirmed.

1 Introduction

1.1 Motivation

Automated parking system (APS) provides humans with greater mobility, productivity and leisure by freeing a driver's parking maneuver operation [1, 2]. Owing to the sensors and actuators equipped on the vehicle, APS usually performs better precision of parking position compared to a novice driver. The high parking precision is beneficial to electric vehicles (EVs) with wireless charging solutions based on near-field wireless power transfer (WPTS), where the power transfer efficiencies significantly reduce as the distances of the two sub-systems increase [3]. Motion planning and path-following control modules in APSs are two decisive factors of the final precision of parking position. First, the planning module generates parking path using methods such as the curve-based method [4], the sampling-based method [5], and the search-based method [1]. Second, the path is followed by the path-following control module, which is usually derived from the vehicle model, such as linear quadratic regulator [6], pure pursuit [7], and model predictive control (MPC) [2]. The parking function has been realized in previous research and some products. However, little research has been reported to reuse the historical data generated by the onboard motion sensors in the vehicle and offline optimization calculation. These data are valuable and contain beneficial system characteristics. APS provides an example of how the data generated by the vehicle can be used to improve the performance in terms of final precision of parking position and parking time.

1.2 Related Work

Data-driven methods receive attention in planning and decision-making after their significant success in the environmental perception of self-driving. Motion data were used to reduce the computational complexity and to improve vehicle motion quality. The usage of motion data in motion planning can be

divided into two categories: to guide conventional global path planners and act as local planners. In the first category, Banzhaf [8] used a deep convolutional neural network (CNN) to learn the poses distribution and guide a bidirectional RRT* planner. Alois et al. [9] used deep reinforcement learning (DRL) to speed up the search in A* planner. In the second category, the planning is usually reformulated as a Markov decision process (MDPs), where the speed and steering angle are decision variables. Song et al. [10, 11] proposed local online motion planning methods with low computational complexity based on Monte Carlo tree search (MCTS). An online tree search method was used to generate fast open-loop parking control signal. Xiong et al. [12] proposed an end-to-end DRL-based parking method using the data collected on the driving simulator and the test platform. Other methods include directly mapping the vehicle states with control command by learning the data with supervised learning [13, 14].

Although MCTS [10, 11] has realized the integrated planning and control, the lower layer of the motion control frame used an open-loop control method that does not use the motion data. The speed response lagged behind the commands. This work uses the MCTS-based parking in Song [10] and further strengthens the system performance by improving the trajectory-following ability. In principle, the motion planner is trained with the hypothesis that the planned trajectory can be precisely tracked. It is reasonable to improve the consistency between the expected behavior given by the planning layer and the practical trajectory-following performance of the control layer.

1.3 Contributions

The contributions of this work are twofold: First, a lower layer data-driven trajectory-following control method strengthens the upper layer data-driven planning module. Simulation results reveal that a better speed following performance is beneficial to the parking time performance. Second, the generalization ability of the proposed method to different initial positions, parking slot size, and road conditions has been validated in high-fidelity CarSim simulation. That results show the potential of the proposed method. It is expected to inspire studies exploring the advantages of data generated by the controlled plant.

The rest of this paper is organized as follows: Section 2 introduces the preliminaries for the upper layer motion planning and the lower layer trajectory following control method in the existing literature. Section 3 introduces the proposed double-layer method. It is followed by results and conclusions in Sections 4 and 5, respectively.

2 Preliminaries

2.1 Monte Carlo Tree Search

MCTS is an online planning method, which incrementally builds a search tree with a set of statistical values on its edge [10]:

$$\{P(\boldsymbol{s},a),N(\boldsymbol{s},a),W(\boldsymbol{s},a),Q(\boldsymbol{s},a)\}, \tag{1}$$

where $P(\boldsymbol{s}, a)$ is the probability of selecting action a at station $\boldsymbol{s}$, $N(\boldsymbol{s}, a)$ is the number of sampling, $W(\boldsymbol{s},a)$ is the total rewards, $Q(\boldsymbol{s}, a)$ is the value of action a at $\boldsymbol{s}$. By sampling in action space, the actual value is estimated by [15]:

$$Q(\boldsymbol{s},a)=\frac{1}{N(\boldsymbol{s},a)}\sum_{i=1}^{N(s)}\mathrm{I}_i(\boldsymbol{s},a)z_i, \tag{2}$$

where I(.) is used to count the number of selecting a at state $\boldsymbol{s}$. As shown in Fig. 1, a search tree can be obtained by performing four steps: selection, expansion, evaluation, and backup. With the increase of sampling numbers, the estimated value gets closer to the actual value.

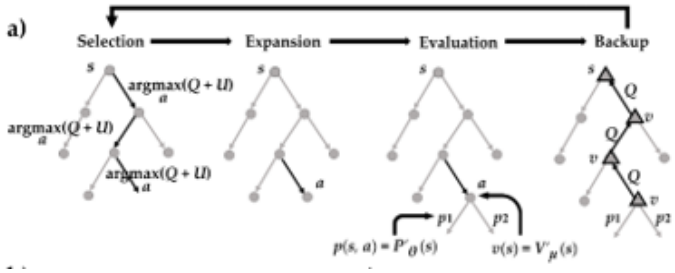

Figure 1: Learnable Monte Carlo tree search (MCTS)

A powerful formula to select the action in the first step of Fig. 1 is the predictor + upper confidence bounds tree (PUCB). The action in the tree search is selected by maximizing a scalar:

$$a_t(\boldsymbol{s}) = \arg\max_a (Q(\boldsymbol{s},a) + cP(\boldsymbol{s},a)\frac{\sqrt{\sum_b N(\boldsymbol{s},b)}}{1+N(\boldsymbol{s},a)}) \tag{3}$$

where c is a constant to balance the exploration and exploitation, $P(\boldsymbol{s}, a)$ is obtained by a predictor, b is the visited number of the parent node of action a.

2.2 Model Predictive Control

MPC solves a finite horizon open-loop optimal problem to obtain the control inputs. The optimization problem should consider the system model and constrains [16]:

$$\begin{aligned} &\min_{\mathrm{U}_t, \xi_{1,t}, \xi_{2,t}, \ldots, \xi_{Nc,t,t}} && J_{Nc}(\xi, U_t) \\ &s.t. && \xi_{k+1,t} = f(\xi_{k,t}, u_{k,t}), k = 0,..., N_p - 1 \\ & && \xi_{k,t} \in X \\ & && u_{k,t} \in U \end{aligned} \tag{4}$$

where J is the cost function, N_c is the length of control inputs, N_p is the number of prediction steps, $u_{k,t}$ is control input, ξ is the system state, X is state space, and U is input space.

2.3 Iterative Learning Control

Consider a system in state space:

$$\begin{aligned} x^l{}_{k+1} &= A^l x^l{}_k + B^l u^l{}_k \\ y^l{}_k &= C^l x^l{}_k \end{aligned} \tag{5}$$

where superscript l stands for l-th control loop. Let $x^l{}_{k+1} = qx^l{}_k$, the expression of $x^l{}_{k+1}$ can be obtained:

$$x^l{}_{k+1} = (q\mathrm{I} - A^l)^{-1} u^l{}_k \tag{6}$$

For j-th running:

$$y^l{}_{jk} = P(q)u^l{}_{jk} + C^l(A^l)^k x^l{}_0, P(q) \equiv C^l(qI - A^l)^{-1}B^l \tag{7}$$

where the elements in $P(q)$ are:

$$p_{mk} = \begin{cases} 0, m < k \\ C^l B^l, m = k \\ C^l A^l{}_{m-1} ... A^l{}_k B^l{}_k, m > k \end{cases} \tag{8}$$

Consider this form of ILC control law:

$$u^l{}_{j+1,k} = Q^l(u^l{}_{j,k} + L^l e^l_{j,k+1}) \tag{9}$$

where Q^l is a filter, L^l is learning matrix and $e^l_{j,k+1}$ is control errors of j-th running at time k+1.

Quadratically optimal ILC (Q-type ILC) [17] is used to consider control errors and control effort, which is expressed as:

$$J^l = (e^l)^{\mathrm{T}} \mathbf{T}_{\mathrm{LG}} e^l + (u^l)^{\mathrm{T}} \mathbf{R}_{\mathrm{LG}} u^l + (\delta^l)^{\mathrm{T}} \mathbf{S}_{\mathrm{LG}} \delta^l \tag{10}$$

The solution is:

$$\begin{aligned} Q^l{}_{\mathrm{opt}} &= \left(P^{\mathrm{T}} \mathbf{T}_{\mathrm{LG}} P + \mathbf{R}_{\mathrm{LG}} + \mathbf{S}_{\mathrm{LG}}\right)^{\mathrm{T}} u^l + (P^{\mathrm{T}} \mathbf{T}_{\mathrm{LG}} P + \mathbf{S}_{\mathrm{LG}}) \\ L^l_{\mathrm{opt}} &= \left(P^{\mathrm{T}} \mathbf{T}_{\mathrm{LG}} P + \mathbf{S}_{\mathrm{LG}}\right)^{\mathrm{T}} P^{\mathrm{T}} \mathbf{T}_{\mathrm{LG}} \end{aligned} \tag{11}$$

where $\mathbf{T}_{\mathrm{LG}}$, $\mathbf{R}_{\mathrm{LG}}$ and $\mathbf{S}_{\mathrm{LG}}$ are weights for tracking errors control input and run-to-run input increment.

3 Method

Although MCTS has been used in the automated parking system of our previous work [10], the planned steering angle and speed commands were used to control the vehicle directly. In this work, we study the influence of tracking control methods on the overall parking performance.

3.1 Overview

The proposed double-layered data-driven parking method diagram is shown in Fig. 2. The double-layers parking methods consist of two parts. In the upper level, MCTS developed in Song et al. [10] is used to

plan the reference parking trajectory associated with reference speed and steering angle commands in the upper level. In the lower level, ILC and MPC control are used for the longitudinal and lateral aspects. MCTS used the data of nonlinear programming to train the artificial neural networks in Song et al. [10] without considering the speed and steering angle response characteristic of the vehicle. ILC uses the data of the control plant to simplify the computation of MPC. The time delay in speed following is compensated by using the speed errors collected on the test platform.

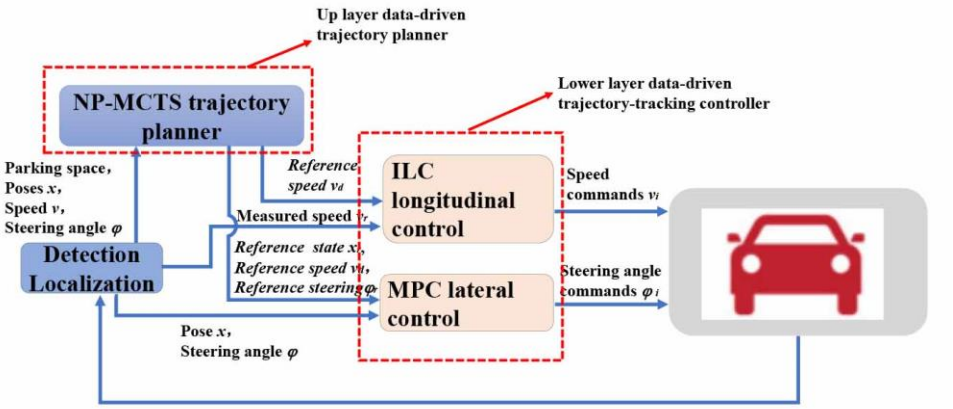

Figure 2: Overall diagram of the parking motion planning and trajectory following control system

The iterative cycle of the proposed double-layers data-driven system is similar to Partial Motion Planning [18], as shown in Fig. 3. MCTS performs online trajectory search with fixed time intervals. The nearest partial trajectory in the tree is fed to trajectory following modular that uses MPC and ILC. The reference point in the trajectory following modular is aligned in time spaces.

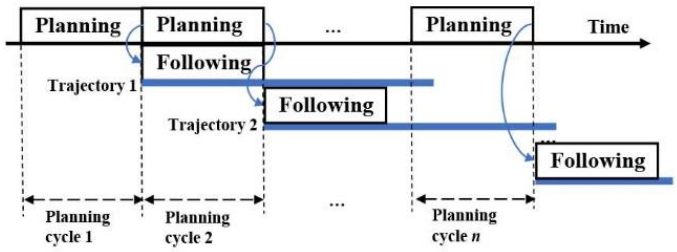

Figure 3: Iterative cycle, the vehicle and parking space information is our root model that is different from [18]

3.2 Lower-level data-driven learning control

Unlike path planning methods, where the temporal information of the trajectory was usually ignored, and the speed was associated with distance information [19], our planner outputs trajectory points with times stamp, as shown in Fig. 4. The advantage is that it regulates the speed controller if speed errors appear.

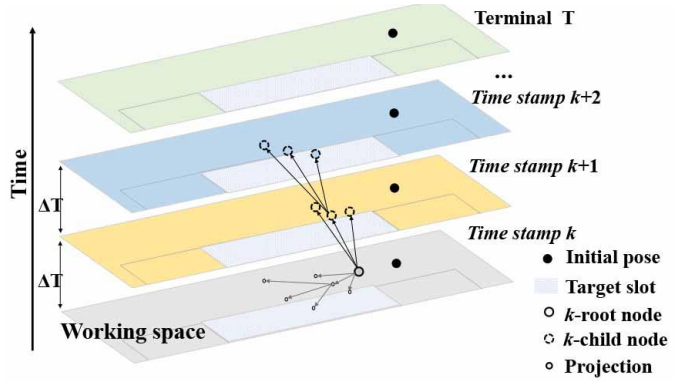

Figure 4: Search in spatiotemporal space in MCTS

Because the speed of parking is low, a single-track kinematic model is reasonable and is frequently used to describe vehicle motion:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix}=\begin{bmatrix}\cos(\theta)\\ \sin(\theta)\\ \tan(\varphi)/L\end{bmatrix}v \tag{12}$$

where (x, y) is the position of real axis midpoint, θ is yaw angle, v is, φ is the steering wheel angle. Writing state with $\boldsymbol{x}=(x,y,\theta)$, then the vehicle motion can be expressed with $\dot{\boldsymbol{x}}=f(\boldsymbol{x},\boldsymbol{u})$. Because the reference trajectory $\dot{\boldsymbol{x}}_r=f(\boldsymbol{x}_r,\boldsymbol{u}_r)$ has been obtained by MCTS, the error model can be obtained by subtraction:

$$\dot{\tilde{\boldsymbol{x}}}=\begin{bmatrix}0&0&-v_r\sin(\theta_r)\\ 0&0&v_r\cos(\theta_r)\\ 0&0&0\end{bmatrix}\begin{bmatrix}x-x_r\\ y-y_r\\ \theta-\theta_r\end{bmatrix}+\begin{bmatrix}\cos(\theta_r)&0\\ \sin(\theta_r)&0\\ \tan(\delta_r)/L&v_r/(L\cos^2\delta_r)\end{bmatrix}\begin{bmatrix}v-v_r\\ \delta-\delta_r\end{bmatrix} \tag{13}$$

If this error model is used in MPC, the speed and steering angle should be calculated. In order to reduce the number of decision variables, the model is simplified as:

$$\dot{\tilde{\boldsymbol{x}}}=\begin{bmatrix}0&0&-v_r\sin(\theta_r)\\ 0&0&v_r\cos(\theta_r)\\ 0&0&0\end{bmatrix}\begin{bmatrix}x-x_r\\ y-y_r\\ \theta-\theta_r\end{bmatrix}+\begin{bmatrix}\cos(\theta_r)&0\\ \sin(\theta_r)&0\\ \tan(\delta_r)/L&v_r/(L\cos^2\delta_r)\end{bmatrix}\begin{bmatrix}v-v_r\\ \delta-\delta_r\end{bmatrix} \tag{14}$$

by assuming that the speed can be well followed. The matrix form of the model is:

$$\dot{\tilde{\boldsymbol{x}}}(k+1)=A_{k,t}\tilde{\boldsymbol{x}}(k)+B_{k,t}\tilde{\boldsymbol{u}}(k) \tag{15}$$

And the i-th coefficient matrices $A_{k,t}$ and $B_{k,t}$

$$A_{k,t}=\begin{bmatrix}1&0&-v_r\sin(\theta_r)T\\ 0&1&v_r\cos(\theta_r)T\\ 0&0&1\end{bmatrix},\ B_{k,t}=\begin{bmatrix}\cos(\theta_r)T&0\\ \sin(\theta_r)T&0\\ T\tan(\delta_r)/L&Tv_r/(L\cos^2(\delta_r))\end{bmatrix}. \tag{16}$$

Both the trajectory following errors and input efforts can be considered, so a new variable $\xi(k\,|\,t)=[\tilde{x}(k\,|\,t),\tilde{u}(k-1\,|\,t)]^{\mathrm{T}}$ is added and the model is rewritten with:

$$\xi(k+1\,|\,t)=\tilde{A}_{k,t}\xi(k\,|\,t)+\tilde{B}_{k,t}\Delta\tilde{\boldsymbol{u}}(k\,|\,t),\eta(k\,|\,t)=\tilde{C}_{k,t}\xi(k\,|\,t) \tag{17}$$

Finally, the prediction equation can be written as:

$$\boldsymbol{Y}(t)=\boldsymbol{\Psi}_t\xi(t)+\boldsymbol{\Theta}_t\Delta\mathbf{U}(t) \tag{18}$$

with

$$Y(t)=\begin{bmatrix}\eta(k+1\,|\,t)\\ \eta(k+2\,|\,t)\\ \dots\\ \eta(k+N_p\,|\,t)\end{bmatrix},\Psi_t=\begin{bmatrix}\tilde{C}_t\tilde{A}_t\\ \tilde{C}_t\tilde{A}_t^2\\ \dots\\ \tilde{C}_t\tilde{A}_t^{N_p}\end{bmatrix},\Delta\mathrm{U}(t)=\begin{bmatrix}\Delta\tilde{\boldsymbol{u}}(t\,|\,t)\\ \Delta\tilde{\boldsymbol{u}}(t+1\,|\,t)\\ \dots\\ \Delta\tilde{\boldsymbol{u}}(t+N_c\,|\,t)\end{bmatrix},\Theta_t=\begin{bmatrix}\tilde{C}_t\tilde{B}_t&\dots&0\\ \tilde{C}_t\tilde{A}_t\tilde{B}_t&\dots&0\\ \dots&\dots&\dots\\ \tilde{C}_t\tilde{A}_t^{Np-1}\tilde{B}_t&\dots&\tilde{C}_t\tilde{A}_t^{N_p-N_c-1}\tilde{B}_t\end{bmatrix},$$

$$\tilde{A}_{k,t}=\begin{bmatrix}A_{k,t}&B_{k,t}\\ 0_{mn}&I_m\end{bmatrix},\tilde{B}_{k,t}=\begin{bmatrix}B_{k,t}\\ I_m\end{bmatrix},\tilde{C}_{k,t}=\begin{bmatrix}C_{k,t},0\end{bmatrix}, \tag{19}$$

where the coefficient matrixes in the model are simplified and assumed to equal.

$$\tilde{A}_{k,t}=\tilde{A}_t,\tilde{B}_{k,t}=\tilde{B}_t,k=0,2,...,N_p-1 \tag{20}$$

The constraints should be considered in the calculation of the control law. We consider the limits of front-wheel angle, and its change rate in this work:

$$\mathbf{u}_{\min}(t+k) \le \mathrm{u}(t+k) \le \mathbf{u}_{\max}(t+k), \Delta\mathbf{u}_{\min}(t+k) \le \Delta\,\mathrm{u}(t+k) \le \Delta\mathbf{u}_{\max}(t+k)\,, \tag{21}$$

which can be written in matrix form:

$$\mathbf{U}_{\min} - \mathbf{U}_{\mathrm{t}} \le \mathbf{A}\Delta\mathbf{U}_{\mathrm{t}} \le \mathbf{U}_{\max} - \mathbf{U}_{\mathrm{t}}, \Delta\mathbf{U}_{\min} \le \Delta\mathbf{U}_{\mathrm{t}} \le \Delta\mathbf{U}_{\max} \tag{22}$$

with

$$\mathbf{U}_{\mathrm{t}} = \mathbf{I}_{Nc} \otimes \mathbf{u}(k-1) = \begin{bmatrix} u(k-1) \\ u(k-1) \\ \ldots \\ u(k-1) \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 0 \\ . & . & . & . \\ 1 & 1 & 1 & 1 \end{bmatrix}_{Nc\times Nc} \otimes \mathbf{I}_m = \begin{bmatrix} \mathbf{I}_m & 0 & \ldots & 0 \\ \mathbf{I}_m & \mathbf{I}_m & \ldots & 0 \\ . & . & . & . \\ \mathbf{I}_m & \mathbf{I}_m & \mathbf{I}_m & \mathbf{I}_m \end{bmatrix}_{Nc\times Nc}. \tag{23}$$

And

$$\Delta\mathbf{U}_{\min} = \mathbf{I}_{Nc} \otimes \Delta\mathbf{u}_{\min} = \begin{bmatrix} \Delta\mathbf{u}_{\min} \\ \Delta\mathbf{u}_{\min} \\ \ldots \\ \Delta\mathbf{u}_{\min} \end{bmatrix}, \Delta\mathbf{U}_{\max} = \mathbf{I}_{Nc} \otimes \Delta\mathbf{u}_{\max} = \begin{bmatrix} \Delta\mathbf{u}_{\min} \\ \Delta\mathbf{u}_{\min} \\ \ldots \\ \Delta\mathbf{u}_{\min} \end{bmatrix}. \tag{24}$$

The final control law of MPC is

$$[\Delta\mathbf{U}, \varepsilon]^{\mathrm{T}}[\boldsymbol{\Theta}_t{}^T \boldsymbol{Q}\boldsymbol{\Theta}_t][\Delta\mathbf{U}, \varepsilon] + 2[\Delta\mathbf{U}, \varepsilon]\boldsymbol{\Theta}_t{}^T \boldsymbol{Q}\boldsymbol{\Psi}_t\xi \tag{25}$$

subject to

$$\begin{bmatrix} \mathbf{A} & 0 \\ -\mathbf{A} & 0 \end{bmatrix}[\Delta\mathbf{U}, \varepsilon]^{\mathrm{T}} \le \begin{bmatrix} \Delta\mathbf{U}_{\max} - \mathbf{U}_t \\ 0 \\ -\Delta\mathbf{U}_{\max} + \mathbf{U}_t \\ 0 \end{bmatrix}, [\mathbf{L}_b, \varepsilon] \le [\Delta\mathbf{U}, \varepsilon] \le [\mathbf{U}_b, \varepsilon]\,. \tag{26}$$

3.3 Speed compensating

As mentioned above, ILC is used to improve the longitudinal control performance of the vehicle. First, the linear system of the speed control is identified to obtain A^l, B^l, C^l in longitudinal linearized system Eq. (5). Second, the matrix P is calculated by Eq. (8). Then, the learning matrix L^l and filter matrix Q^l are obtained by Eq. (11). Equation (9) is used as a speed control law. The errors are calculated by:

$$\begin{bmatrix} e^l_{j,1} \\ e^l_{j,2} \\ \ldots \\ e^l_{j,N} \end{bmatrix} = \begin{bmatrix} y^l_{d,1} \\ y^l_{d,2} \\ \ldots \\ y^l_{d,N} \end{bmatrix} - \begin{bmatrix} y^l_{j,1} \\ y^l_{j,2} \\ \ldots \\ y^l_{j,N} \end{bmatrix} \tag{27}$$

where y_d is the desired signal, y_j is the measured signal.

The parallel parking process can be divided into two phases: park-in and shuttling in the parking spaces.Because the trajectory in the second phase is sensitive to the final poses of the park-in stage, the trajectory is used in the first stage, and open-loop control is used to adjust the parking space. The error signals of speed control should be stored. To ensure the learned experience can be used in other initial poses, we divide the parking-in stage into acceleration and deceleration phases, as shown in Fig. 5. When the errors experience of the acceleration is running out but the desired speed does not decrease, the last errors of the acceleration phase will be used. In other cases, if the desired speed decrease in advance, the pointer will jump to the deceleration phase.

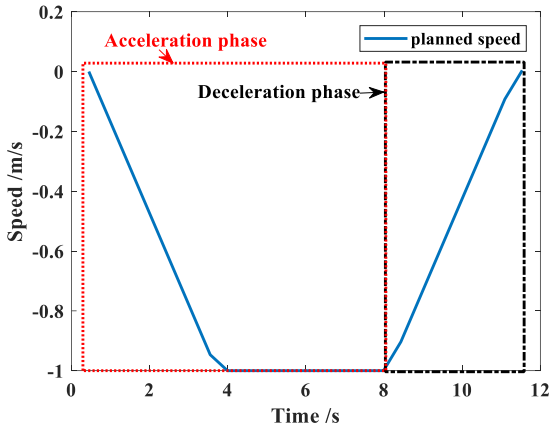

Figure 5: Phases of the speed during the parking

4 Comparison of test and simulation

The objective of the experiments is to confirm the effectiveness of the lower-level data-driven trajectory following the control method and to study the adaptivity of the proposed double-layers parking method to different work conditions.

4.1 Test conditions

The test is carried out in Simulink and CarSim. The planning period is 50 ms, the period of the trajectory following control is 5 ms. The parameters of the up-layer motion planner are the same as Song et al. [10]. The parameters of the vehicle and lower-layer control are listed in Table 1.

Table 1: Vehicle and controller parameters

Item	Value	Item	Value
Vehicle length	3.569 m	Front overhang	0.72 m
Vehicle width	1.551 m	Rear overhang	0.54 m
Wheelbase	2.305 m	Trans. ratio	16.68
Vehicle length	3.569 m	Front overhang	0.72 m
Steering -MPC	400 °/s	Q weight -MPC	*diag*(1, 0.1, 2)
Prediction -MPC	20 steps	R weight-MPC	$1.5\times I_{Nc}$
Control -MPC	15 steps	$\mathbf{T}_{LG}$ -LQR	$1 \times I_{N\times N}$
Relaxation -MPC	5	$\mathbf{R}_{LG}$ -LQR	$0.1\times I_{N\times N}$
		$\mathbf{S}_{LG}$ -LQR	$0.01\times I_{N\times N}$

As shown in Fig. 6, the initial parking poses are distributed in x = [1.7 m, 3.7 m], y = [1.25 m, 2.25 m] with zeros initial angle. The values are determined using parking standard ISO-16787. The data to learn the lower-level longitudinal control law is obtained in the position of the red circle in Fig. 6. To save the computer memory consumption and test the generalization ability, we used only one pose to learn lower-level. To sum up, the learning of the upper-level layer used 25 poses. Moreover, the lower-level layer used one pose.

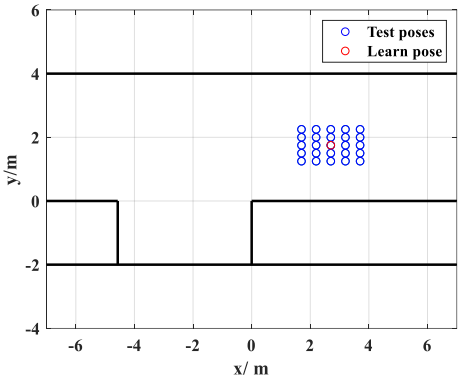

Figure 6: Initinal positions of traning in the double-layers data driven method, where the the data in circles are used in motion planning layers and red circle is used to learn speed compensate

4.2 Results on training positions - verification of lower layer

The method without lower-level trajectory tracking is denoted as MCTS. MCTS+ILC+MPC denotes the double-layer method. The planners of both MCTS and MCTS+ILC+MPC are trained using 25 data in the poses in Fig. 6. Speed-following and trajectory-following errors at an arbitrary pose are shown in Figs. 7-8 to confirm the benefit of the lower-level trajectory-following method. The error of motion control is reduced after adding the tracking module. On the one hand, the reason is that in the longitudinal direction, ILC used the learning experience to take the historical error as a feedforward to compensate for the output speed of MPC. In addition, MPC performed tracking control in the lateral direction according to the pose feedback. The phenomenon in Fig. 8 is similar to that in Fig. 7. The tracking error fluctuates in the time domain. The reason is that although there is a low-pass filter matrix Q in the control rate of ILC, the vehicle speed error is collected according to the discrete period.

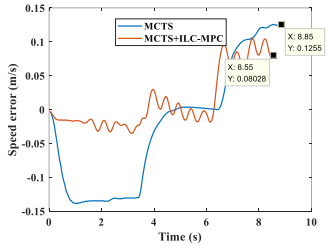

Figure 7: Comparsion of speed following errors at (1.7 m, 1.25 m, 0°)

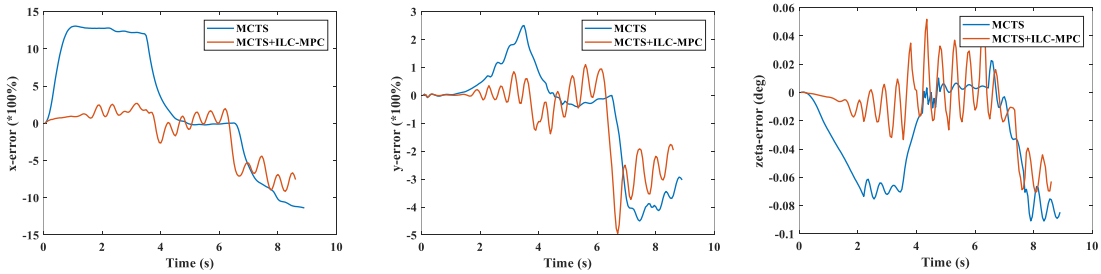

Figure 8: Comparsion of trajectory-following control errors at (1.7 m, 1.25 m, 0°): (a) *x*-coordinate; (b) *y*-coordinate, and (c) θ

Table 2 shows the speed following errors, trajectory errors, and the time spent in the whole process of 25 parking times. In terms of speed errors, although the standard deviation increases, the mean, maximum, and mean square values of speed error of single parking are reduced after adding tracking control.

Table 2: Statistical results of parking process in the training poses, 25 trials

		Open loop		With tracking	
		Mean	Std	Mean	Std
Speed errors/ m/s	Mean	**0.06**	0.00	**0.03**	0.01
	MSE	**0.003**	0.000	**0.001**	0.004
	Maximum	**0.13**	0.001	**0.11**	0.08

	Mean	**6.170**	0.41	**2.67**	1.48
X errors/ %	MSE	**27.43**	0.92	**10.62**	25.16
	Maximum	**13.07**	0.12	**10.12**	6.40
Time/ s	/	**10.02**	0.66	**9.88**	0.76

As the direction of the motions in the parking space changes frequently, trajectory following is only used out the parking space. The comparison of open-loop control and double-layer planning and control method is shown in Fig. 9. In 23 cases out of 25, the proposed method's parking time is shorter than open-loop control. That confirms the effectiveness of lower-level trajectory-following control.

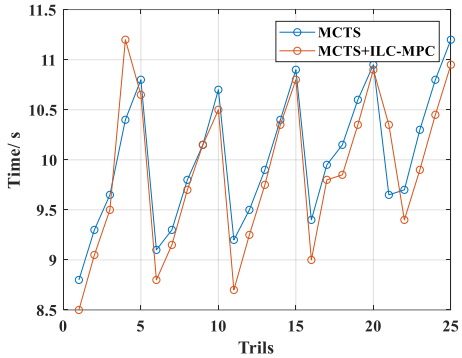

Figure 9: Motion time of parking in different initial positions

The speed at pose four is also shown in Fig. 10, where sudden deceleration is avoided by adding the lower-level control method. In previous studies, open-loop planning and control are shown to be feasible. The learning compensation in ILC speed following further improved the overall performance.

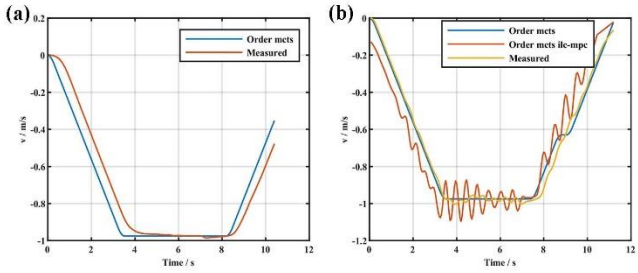

Figure 10: Speed order and response at pose [3.2 m, 1.25 m]: (a) without following control; (b) with following control

The results of final position accuracy and parking process are shown in Table 3, where the effective rate is defined as the percentage of faster parking trials obtained by the tracking layer among the 25 positions. It can be seen that the average *y*-axis errors of double-layers parking are almost the same with one-layer open-loop control. The tracking layer reduces parking time of 72% of the initial position at the expense of higher *y*-axis errors and standard deviation. It indicates that up-layer online planning is important to the high and stable position precision. With a more complex planning and control structure in our case, the worst performance increases.

Table 3: Final parking results of position accuracy and parking process, 25×2 trials

	Item	Open-loop	With tracking
Y errors/ m	Mean	**0.010**	**0.016**
	Max	**0.055**	**0.114**
	Min	9.79e-05	0.003
	Std.	**0.015**	**0.023**
θ errors/ °	Mean	1.137	1.185
	Max	1.318	1.327
	Min	0.984	1.013
	Std.	0.091	0.090
Gear Changes times	Mean	2.00	2.00
	Max	2	2
	Min	2	2

	Std.	0.00	0.00
Time/ s	Mean	**16.904**	**16.9**
	Std.	**0.715**	1.11
	Effective rate	**72%**	

4.3 Adaptability to different initial positions - verification of upper layer

The motion planning policy is tested by changing the initial vehicle angle from 0 to [−8°, 8°]. The results of the parking paths are shown in Fig. 11, where the requirements of safety are satisfied, and the final bounding boxes of the vehicle are in the parking spaces.

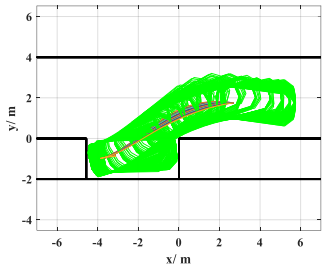

Figure 11: Parking trajectories when change the initial angle of vehicle from 0° to [−8°, 8°] at (2.7 m, 1.75 m)

4.4 Adaptability to different sizes of slots

To further test the adaptability of the proposed double-layer date-driven parking method, we changed the length of the slots. The proposed system learned at parking space with a length of 4.57 m. The results are shown in Fig. 12. The final angles of the vehicle are 1.186°, 1.095°, 1.133°, and 3.537°, respectively. The parking task is succeeded except for the smallest parking space, where the parking time is more than 50 seconds set in the experiment. The adaptability to different slots has been confirmed.

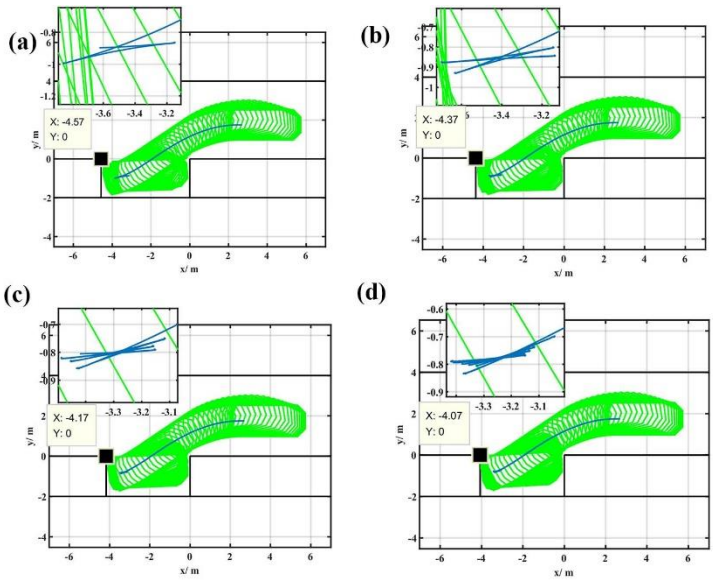

Figure 12: Parking trajectories uing same model and parameters with different parking slot length: (a) 4.57 m; (b) 4.37 m; (c) 4.17 m; (d) 4.07 m

4.5 Adaptability to different road conditions

The road friction coefficient is 0.85 in the data collection stage of ILC speed control. The friction is reduced to test the adaptability to different road conditions. Figure 13 shows the speed response and the parking trajectories with different road friction. It can be seen that the measured speed in the different roads is different. As the parking speed is low, the difference in the trajectories with friction coefficients above 0.05 is small. The friction of 0.05 is too small to provide enough driving force. It also shows that the response lag in the deceleration could not be completely avoided. The delay comes from the recognition of the deceleration stage in Fig. 5.

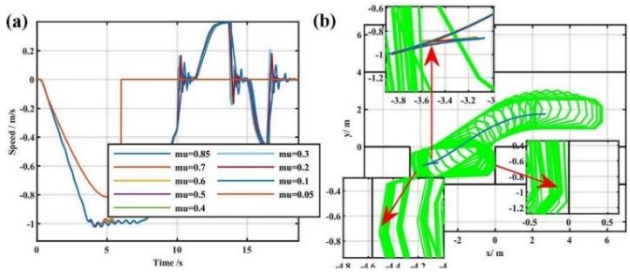

Figure 13: (a) Speed response, and (b) parking trajectories with different road friction 0.1-0.85, without 0.05

5 Conclusions

A method of double-layer data-driven motion planning and control for parallel parking is proposed in this work. Although the one-layer open-loop integrated planning and control method has been previously confirmed. The lower layer trajectory-following method with speed compensating is beneficial to the parking time performance and avoid sudden deceleration at the expense of higher *y*-axis errors. The mean precisions of parking position in the *y*-axis for one-layer open-loop control and double-layers control are 0.010 m and 0.016 m, respectively. Moreover, the generalization ability of the proposed method is confirmed. In the future, we will apply the proposed method to more complex driving scenes, where a physical model is hard to obtain.

Acknowledgments

This study received no external fund.

References

[1] H. Banzhaf, D. Nienhuser, S. Knoop et al., *The future of parking: A survey on automated valet parking with an outlook on high density parking*, 2017 28th IEEE Intelligent Vehicles Symposium, ISSN 1931-0587, 2017, 1827-1834.

[2] C. Jang, C. Kim, S. Lee et al., *Re-plannable automated parking system with a standalone around view monitor for narrow parking lots*, IEEE Transactions on Intelligent Transportation Systems, ISSN 1524-9050, 21(2019), 1-14.

[3] P. Machura, Q. Li, *A critical review on wireless charging for electric vehicles*, Renewable & Sustainable Energy Reviews, ISSN 1364-0321, 104(2019), 209-234.

[4] S. Upadhyay, A. Ratnoo, *A point-to-ray framework for generating smooth parallel parking maneuvers*, IEEE Robotics and Automation Letters, ISSN 2377-3774, 3(2018), 1268-1275.

[5] D. Kim, W. Chung, S. Park, *Practical motion planning for car-parking control in narrow environment*, IET Control Theory & Applications, ISSN 1751-8644, 4(2010), 129-139.

[6] Z. Fan, H. Chen, *Study on path following control method for automatic parking system based on LQR*, SAE International Journal of Passenger Cars - Electronic and Electrical Systems, ISSN 1946-4614, 10(2016), 41-49.

[7] J. Ahn, S. Shin, M. Kim et al., *Accurate path tracking by adjusting look-ahead point in pure pursuit method*, International Journal of Automotive Technology, ISSN 1229-9138, 22(2021), 119-129.

[8] H. Banzhaf, P. Sanzenbacher, U. Baumann et al., *Learning to predict ego-vehicle poses for sampling-based nonholonomic motion planning*, IEEE Robotics and Automation Letters, ISSN 2377-3774, 4(2019), 1053-1060.

[9] J. Bernhard, R. Gieselmann, K. Esterle et al. *Experience-based heuristic search: robust motion planning with deep Q-learning*, 2018 21st International Conference on Intelligent Transportation Systems, ISSN 2153-0009, 2018, 3175-3182.

[10] S. Song, H. Chen, H. Sun et al., *Time-optimized online planning for parallel parking with nonlinear optimization and improved Monte Carlo tree search*, IEEE Robotics and Automation Letters, ISSN 2377-

3774, 2022.

[11] S. Song, H. Chen, H. Sun et al., *Data efficient reinforcement learning for integrated lateral planning and control in automated parking system*, Sensors, ISSN 1424-8220, 20(2020).

[12] P. Zhang, L. Xiong, Z. Yu et al., *Reinforcement learning-based end-to-end parking for automatic parking system*, Sensors, ISSN 1424-8220, 19(2019).

[13] N. Kalkovaliev, G. Mirceva, *Autonomous driving by using convolutional neural network*, 2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), ISBN 9781665440585, 2021.

[14] Y. Lin, L. Li, X.Y. Dai et al., *Master general parking skill via deep learning*, 2017 IEEE Intelligent Vehicles Symposium, ISSN 1931-0587, 2017, 941-946.

[15] C.B. Browne, E. Powley, D. Whitehouse et al., *A survey of Monte Carlo tree search methods*, IEEE Transactions on Computational Intelligence and Ai in Games, ISSN 1943-068X, 4(2012), 1-43.

[16] P. Falcone, *Nonlinear model predictive control for autonomous vehicles*. Department of Engineering. Benevento: University of Sannio, 2007.

[17] D. A. Bristow, B. Hencey, *A Q, L factorization of norm-optimal iterative learning control*, 47th IEEE Conference on Decision and Control, ISSN 0743-1546, 2008, 2380-2384.

[18] S. Petti, T. Fraichard, *Safe motion planning in dynamic environments*, IEEE/RSJ International Conference on Intelligent Robots and Systems, ISBN 0-7803-8912-3, 2005, 3726-3731.

[19] Y. Kuwata, J. Teo, G. Fiore et al., *Real-time motion planning with applications to autonomous urban driving*, IEEE Transactions on Control Systems Technology, ISSN 1063-6536, 17(2009), 1105-1118.

Presenter Biography

Shaoyu Song is a Ph.D. candidate at the School of Automotive Studies, Tongji University, engaged in path planning and intelligent control technology of wheeled vehicles. He received his bachelor's and master's degree from Chongqing University in 2015 and 2018, respectively.