Deep Learning-Based Trajectory Planning and Control for Autonomous Ground Vehicle Parking Maneuver

Runqi Chai, *Member, IEEE*, Derong Liu, *Fellow, IEEE*, Tianhao Liu, Antonios Tsourdos, *Member, IEEE*, Yuanqing Xia, *Senior Member, IEEE*, and Senchun Chai, *Senior Member, IEEE*

***Abstract*— In this paper, a novel integrated real-time trajectory planning and tracking control framework capable of dealing with autonomous ground vehicle (AGV) parking maneuver problems is presented. In the motion planning component, a newly-proposed idea of utilizing deep neural networks (DNNs) for approximating optimal parking trajectories is further extended by taking advantages of a recurrent network structure. The main aim is to fully exploit the inherent relationships between different vehicle states in the training process. Furthermore, two transfer learning strategies are applied such that the developed motion planner can be adapted to suit various AGVs. In order to follow the planned maneuver trajectory, an adaptive learning tracking control algorithm is designed and served as the motion controller. By adapting the network parameters, the stability of the proposed control scheme, along with the convergence of tracking errors, can be theoretically guaranteed. In order to validate the effectiveness and emphasize key features of our proposal, a number of experimental studies and comparative analysis were executed. The obtained results reveal that the proposed strategy can enable the AGV to fulfill the parking mission with enhanced motion planning and control performance.**

***Note to Practitioners*—This article was motivated by the problem of optimal automatic parking planning and tracking control for autonomous ground vehicles (AGVs) maneuvering in a restricted environment (e.g., constrained parking regions). A number of challenges may arise when dealing with this problem (e.g., the model uncertainties involved in the vehicle dynamics, system variable limits, and the presence of external disturbances). Existing approaches to address such a problem usually exploit the merit of optimization-based planning/control techniques such as model predictive control and dynamic programming in order for an optimal solution. However, two practical issues may require further considerations: 1). The nonlinear (re)optimization process tends to consume a large amount of computing power and it might not be affordable in real-time; 2). Existing motion planning and control algorithms might not be easily adapted to suit various types of AGVs. To overcome the aforementioned issues, we present an idea of utilizing the recurrent deep neural network (RDNN) for planning optimal parking maneuver trajectories and an adaptive learning NN-based (ALNN) control scheme for robust trajectory tracking. In addition, by introducing two transfer learning strategies, the proposed RDNN motion planner can be adapted to suit different AGVs. In our follow-up research, we will explore the possibility of extending the developed methodology for large-scale AGV parking systems collaboratively operating in a more complex cluttered environment.**

***Index Terms*— Real-time trajectory planning, tracking control, autonomous ground vehicle, deep neural networks, adaptive learning tracking control.**

I. Introduction

THE development of autonomous ground vehicles (AGVs) or self-driving cars has received considerable attention during the last two decades due to its potential benefits in terms of alleviating urban traffic congestion, reducing harmful emissions, and enhancing road safety. The overall aim is to replace the role of manned vehicles in a less risky and more economic fashion [1]–[3]. Loosely speaking, four key modules are involved in an AGV system [4]: localization and mapping, environmental perception, motion planning, and motion control. The last two modules are mainly responsible for making driving decisions and steering vehicle movements [5], [6]. Consequently, they are usually identified as important indicators to reflect the intelligence level of an AGV platform.

The research executed in this work focuses on the automatic parking maneuver problem. This type of maneuver is a basic but typical usecase in the intelligent driving mode. However, it is undeniable that planning the parking maneuver for AGVs in an effective and timely manner is still challenging, as complex traffic environment or physical requirements must be frequently taken into account [6]–[8]. This task becomes even harder when certain performance index is required to be optimized. Therefore, both scholars and engineers are stimulated to design more promising motion planners capable to deal with various AGV parking maneuver scenarios. In [9], employing the double tree structure, an enhanced random tree-like motion planner was advocated to generate trajectories for mobile robotic systems. Similarly, a novel motion planning

Manuscript received 26 January 2022; revised 25 March 2022; accepted 13 June 2022. Date of publication 23 June 2022; date of current version 3 July 2023. This article was recommended for publication by Associate Editor A. Pietrabissa and Editor C. Seatzu upon evaluation of the reviewers' comments. *(Corresponding author: Runqi Chai.)*

Runqi Chai, Tianhao Liu, Yuanqing Xia, and Senchun Chai are with the School of Automation, Beijing Institute of Technology, Beijing 100081, China (e-mail: r.chai@bit.edu.cn; liutianhao233@163.com; xia_yuanqing@bit.edu.cn; chaisc97@163.com).

Derong Liu is with the Department of Electrical and Computer Engineering, University of Illinois at Chicago, Chicago, IL 60607 USA (e-mail: derong@uic.edu).

Antonios Tsourdos is with the School of Aerospace, Transport and Manufacturing, Cranfield University, Bedford MK43 0AL, U.K. (e-mail: a.tsourdos@cranfield.ac.uk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TASE.2022.3183610.

Digital Object Identifier 10.1109/TASE.2022.3183610

method, combining random tree and optimal control theory, was proposed in [10]. These two works belong to the class of sample-and-search methods where a finite set of mesh grids is first applied to discretize the search space. Then a satisfactory connection between the initial pose and the target pose is selected.

On the contrary, in [11] the authors utilized a particular dynamic optimization algorithm to plan the time-optimal parking trajectory for a class of wheeled mobile robots. Furthermore, benefiting from an initial guess generator, a dual-loop parking maneuver planner was established for the AGV in [12]. These two works mainly aim to construct an optimal control formulation consisting of the AGV dynamics, vehicle-related constraints, and other parking demands. Subsequently, well-developed optimization algorithms are utilized to produce the maneuver trajectories which will then be used for the automatic driving controller or advising human drivers. One advantage associate with this type of approach is that different limitations/demands can easily be included in the optimization process. However, these reported optimization-based strategies suffer from two critical issues: 1). They are less likely to be implemented in real-time due to huge computational demands; 2). The algorithm's convergence can significantly be affected by disturbances, uncertainties, local infeasible regions, etc.

Recently, some investigations revealed that it is possible to plan the optimal maneuver trajectory by using deep neural network (DNN)-based direct recalling [13], [14]. The core idea of this approach is to train the DNN on pre-generated trajectory ensembles such that it can learn and represent the relationship between optimal control actions and maneuver trajectories. This enables to plan the maneuver trajectory online by iteratively recalling the trained mapping relationship. Although previous works have reported successful applications of this approach and verified the effectiveness of its results, two critical problems still remain. First, different vehicle state variables are treated independently in the DNN, which means their inherent relations might not be fully exploited. Second, this method might not be adapted to suit different types of autonomous ground vehicles. Nevertheless, due to the high potential for real-time applications, we make an attempt to address the aforementioned issues and design an enhanced recurrent DNN-based (RDNN) motion planner for the problem considered in this paper.

Once a pre-planned trajectory has been produced, it should be provided to the motion controller to track during the actual maneuver phase. Currently, numerous contributions to the design of trajectory tracking controllers for autonomous vehicles can be reviewed in the literature [15]–[20]. For example, the problem of tracking control of robotic cars was addressed in [15], where an observer-based PID controller was constructed. A multi-AGVs formation control problem was considered in [16], where the authors applied a hybrid strategy combining robust control and neural network to deal with the uncertain parameters as well as the external disturbances. In [17], a human–machine co-driving AGVs control strategy was designed based on an adaptive command filtering method. The authors of [18] addressed the problem of steering a line of autonomous vehicles by using predetermined speed profiles imposed on the AGVs. The problem of AGV path tracking under different payloads was considered in [20], wherein the authors designed and applied a dynamic sliding-mode control to deal with uncertainties and achieve stable steering performance. Considering system uncertainties and actuator constraints, a T-S fuzzy output feedback control scheme was constructed to generate the steering wheel maneuver profiles in [21]. Another potentially effective control algorithm which can be applied to form the trajectory tracking system is the model predictive control (MPC). For example, by taking into account some dynamic public traffic restrictions, a model predictive guidance and control framework was established in [22] to plan and steer the vehicle's trajectory. Besides, the authors of [23] proposed an MPC-scheme, which is capable of anticipating the motion of the front vehicle, to fulfill the control of car following.

In addition, owing to the strong approximation, adaption and learning capabilities, applications of NN-based or deep learning-based algorithms on autonomous vehicle trajectory tracking control problems have also gained significant attention [24]–[26]. Such an intelligent controller has been widely investigated in recent years. For instance, the work presented by Lin *et al.* [27] indicated that by applying a fuzzy NN-based controller, the motion control mission can successfully be fulfilled and the trajectory tracking performance is likely to be improved. Yang *et al.* [28] investigated the possibility of applying an adaptive NN scheme in order to steer the motion of a wheeled inverted pendulum system. Besides, the authors addressed a micro-aerial vehicle trajectory following problem in [29], wherein a backstepping-free NN-based control law was derived to achieve robust tracking. Compared to other tracking control schemes, NN-based control algorithms tend to be easily constructed and initialized. Moreover, certain adaptive strategies can be designed and embedded in this control scheme, thereby allowing the NN learns online. Due to these reasons, in this paper, we are interested in designing an adaptive learning NN-based (ALNN) control algorithm to steer the AGV parking maneuver.

When applying the ALNN controller, two important issues required to be considered are the close-loop stability and the convergence of tracking errors. That is, whether the system state variable can be stabilized and whether the actual maneuver trajectory is able to follow the pre-planned reference as close as possible. Consequently, our next focus of this paper is to address these concerns. To do this, we have devoted efforts on designing an NN-based control law, and an adaptive learning law. The stability of the proposed ALNN control scheme, along with the convergence of tracking errors, can theoretically be guaranteed. This will be detailed in later sections of the paper.

The main contributions/novelties of the present work lie in the following aspects:

1) The idea of utilizing deep neural networks (DNNs) for approximating optimal parking trajectories proposed in [30] is further extended by taking advantages of a recurrent network structure. This modification (denoted as RDNN) helps the training process to better exploit the inherent relationships between different vehicle states,

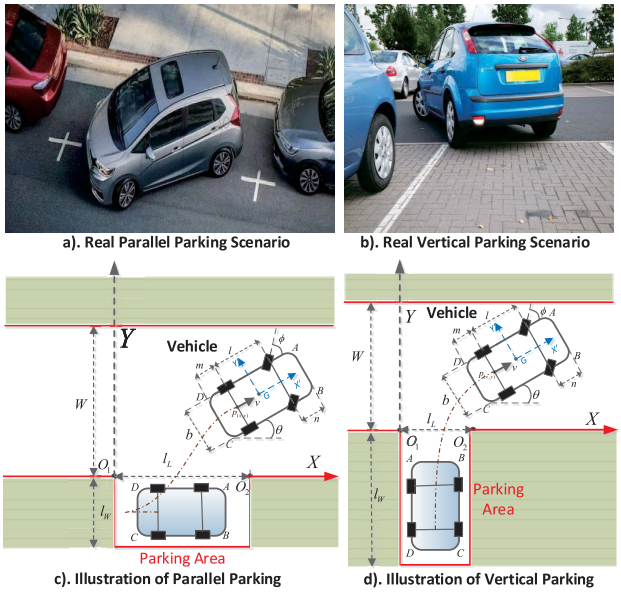

Fig. 1. Parking maneuver scenarios.

thereby resulting in an enhanced approximation performance.

2) Two transfer learning strategies are proposed such that the developed RDNN motion planner can be adapted to suit various types of AGVs.
3) An adaptive learning NN-based (ALNN) control algorithm is constructed to track the planned linear and angular velocities of the AGV. Theoretical results are provided to guarantee the tracking performance and stability of the ALNN motion controller.
4) Detailed comparative experiments and case studies were performed to demonstrate the effectiveness of the proposed trajectory planning and control scheme for the problem considered in this paper.

The remaining sections of this paper are outlined as follows. In Section II, the parking maneuver mission and AGV equations of motion are briefly introduced. Following that, in Section III we construct the real-time RDNN-based trajectory planning method and demonstrate the transfer learning strategy. Subsequently, in Section IV, the ALNN trajectory tracking controller, along with some theoretical results, will be detailed. Section V provides a number of comparative case studies in order to validate the effectiveness of the proposed design. Finally, in Section VI, key findings observed from the experimental results will be summarised and concluded.

II. Problem Formulation

A. Parking Maneuver Scenarios

Parking maneuver can be regarded as an important usecase among the intelligent driving modes. In this study, we focus on two typical parking maneuver scenarios which commonly exist in the real world. They are, namely, the parallel parking as well as the vertical parking. Their illustrations and some notations are visualized in Fig. 1, in which the bold solid line indicates the boundary of the road.

Prior to present the parking maneuver optimization model in detail, the AGV equations of motion as well as the collision avoidance constraints will be introduced in the following subsections.

B. AGV Equations of Motion

The AGV equations of motion can be described as

$$\begin{cases} \dot{p}_x(t) = v(t)\cos(\theta(t)) \\ \dot{p}_y(t) = v(t)\sin(\theta(t)) \\ \dot{\theta}(t) = \dfrac{v(t)\tan(\phi(t))}{L} \\ \dot{\phi}(t) = \eta_1(t) \\ \dot{v}(t) = a(t) \\ \dot{a}(t) = \eta_2(t) \end{cases} \tag{1}$$

where

(p_x, p_y) : represents the pose of the AGV;
(θ, ϕ) : represents the heading and steering angles;
(v, a) : represents the speed and acceleration;
(η_1, η_2) : represents the steering rate and jerk;
(L, t) : represents the wheel base of the AGV; and the time variable.

Defining the state variables of the controlled AGV as $x(t) = [p_x(t), p_y(t), \theta(t), \phi(t), v(t), a(t)]^T$ with a given initial condition $x(0) = x_{\text{ini}}$, (1) can be abbreviated as $\dot{x}(t) = f(x(t), u(t))$. Here, $u = [\eta_1(t), \eta_2(t)]^T$ stands for the control variable consisting of the steering rate η_1 and jerk η_2, while $f(x(t), u(t))$ denotes the right hand side of (1). During the parking maneuver, the state variable of the AGV is required to satisfy its mechanical limit which is expressed as $x(t) \in [x^{\min}, x^{\max}]^T$. To enhance the continuity and smoothness of the acceleration and steering angle trajectories, two additional system equations with respect to a and ϕ are introduced in (1). Moreover, rate constraints are also imposed on $\dot{a}(t)$ and $\dot{\phi}(t)$, resulting in $u(t) \in [u^{\min}, u^{\max}]^T$.

C. Collision Avoidance Constraints

The region occupied by the vehicle can be denoted as $\mathbb{S}(x)$, which is a subset of $\mathbb{R}^n$. One simple way to establish the collision avoidance constraint is to consider the controlled AGV as a point mass, thereby resulting in $\mathbb{S}(x) = p(t)$ for any $t \in [0, t_f]$. Here, $p(t) = [p_x(t), p_y(t)]$ stands for the position of the AGV at t and this assumption is applied in a number of literature works. However, an on-road object such as an obstacle or a parked vehicle cannot be simply treated as a point mass in reality [31], [32]. Alternatively, they should be considered as a full-dimensional object. As a result, motivated by [32], a collision-free AGV maneuver trajectory should satisfy

$$\mathbb{S}(x) \cap \mathbb{O}^{(m)} = \emptyset, \quad \forall m = 1, \ldots, N_o, \tag{2}$$

with

$$\begin{cases} \mathbb{S}(x) = Q(x)\mathbb{W} + T(x), & \text{(3a)} \\ \mathbb{W} = \{z \in \mathbb{R}^n | Gz \le g\}, & \text{(3b)} \\ \mathbb{O}^{(m)} = \{z \in \mathbb{R}^n | A^{(m)} z \le b^{(m)}\}. & \text{(3c)} \end{cases}$$

In (2) and (3), we apply $\mathbb{O}^{(m)}$ to represent the region occupied by the m-th obstacle, and its shape is regulated by $(A^{(m)}, b^{(m)})$. Here, $m = 1, \ldots, N_o$ and N_o stands for the number of obstacles. Note that in real-world scenarios, multiple obstacles may exist in the environment. (G, g) regulates the shape of the initial compact set $\mathbb{W}$. $Q(x)$ can be understood as a rotation matrix, whereas $T(x)$ outputs a translation vector. From the definition of $\mathbb{O}^{(m)}$, it is obvious that the obstacles are modeled as convex compact sets with zero as the interior point. Note that for nonconvex obstacles, they can usually be approximated by a number of convex ones. Although both $\mathbb{S}(x)$ and $\mathbb{O}^{(m)}$ are modeled as convex regions, constraint (2) is nondifferentiable and nonconvex. To preserve the continuity and model smoothly the collision avoidance constraint, the following inequality can be applied:

$$\text{dist}(\mathbb{S}(x), \mathbb{O}) > \underline{d} \geq 0. \tag{4}$$

where $\mathbb{O}$ denotes the region occupied by the obstacle, while $\underline{d}$ can be regarded as a safety margin. In addition, the definition of distance is utilized, which is given by [32], [33]:

$$\text{dist}(\mathbb{S}(x), \mathbb{O}) := \min_r \{\|r\| : (\mathbb{S}(x) + r) \cap \mathbb{O} = \emptyset\}. \tag{5}$$

Simply imposing condition (4) for the entire time domain $t \in [0, t_f]$ may result in numerical difficulties for gradient-based optimization algorithms, as condition (4) requires a solution to the sub-optimization problem. Alternatively, an equivalent form is used by exploring the next proposition established in [32], [33]:

Proposition 1: Given that the obstacle and the AGV are in the form of (3a) and (3c), then imposing inequality constraint (4) is equivalent to

$$\begin{aligned}
&\text{Find } \lambda^{(m)} \geq 0, \mu^{(m)} \geq 0 \\
&\text{s.t. } \forall t \in [0, t_f] \\
&\quad (A^{(m)}T(x) - b^{(m)})^T \lambda^{(m)} - g^T \mu^{(m)} > \underline{d} \\
&\quad Q^T(x) A^{(m)^T} \lambda^{(m)} + G^T \mu^{(m)} = 0 \\
&\quad \|A^{(m)^T} \lambda^{(m)}\| \leq 1
\end{aligned} \tag{6}$$

where $\lambda^{(m)}$ and $\mu^{(m)}$ are dual variables.

By applying Proposition 1, a smooth transformation of the original collision avoidance constraint (4) can be obtained, thereby alleviating the burden of the optimization process.

D. Parking Maneuver Optimization Model

To describe the mission profile in a straightforward manner, a constrained optimal control formulation is presented. The established formulation contains multiple constraints reflecting the physical limits and safety factors. Specifically, we model the parking maneuver optimization problem in the form of

$$\begin{aligned}
\min_{x, u, \lambda^{(m)}, \mu^{(m)}} \quad & J = \int_0^{t_f} L(x(t), u(t)) dt + \Phi(x(t_f), t_f) \\
\text{s.t. } & \forall t \in [0, t_f], \quad \forall m \in \{1, \ldots, N_o\} \\
& \dot{x}(t) = f(x(t), u(t)) \\
& x(0) = x_0 \\
& x(t_f) = x_f \\
& A^{(m)}T(x) - b^{(m)})^T \lambda^{(m)} - g^T \mu^{(m)} > \underline{d} \\
& Q^T(x) A^{(m)^T} \lambda^{(m)} + G^T \mu^{(m)} = 0 \\
& \|A^{(m)^T} \lambda^{(m)}\| \leq 1 \\
& x(t) \in [x^{\min}, \quad x^{\max}] \\
& u(t) \in [u^{\min}, \quad u^{\max}] \\
& \lambda^{(m)} \geq 0, \quad \mu^{(m)} \geq 0
\end{aligned} \tag{7}$$

where $L(x(t), u(t))$ stands for the process cost, while $\Phi(x(t_f), t_f)$ denotes the terminal cost. Note that the optimization problem (7) is in a continuous-time form. A discretized version will be presented in the next section in company with the optimal trajectory ensemble generation process.

In this paper, we aim to fulfill the parking mission within the shortest time possible, thus resulting in $J = t_f$. In (7), x_f denotes the target final state value which is directly related to the considered parking scenarios. It should be noted that for the parking maneuver scenarios described in Fig. 1, the regions where the AGV cannot enter are modeled as obstacles. Specifically, for the vertical parking case, the no-enter zones can be described by three axis-aligned rectangles (e.g., regions highlighted in solid color as shown in Fig. 1(d)). Similarly, for the parallel parking case, the no-enter zones can be described by three rectangles of a different size. This idea can also be extended to other irregular parking cases.

III. RDNN-Based Optimal Parking Trajectory Generation

The trajectory optimization model (7) can be solved with an acceptable accuracy offline using off-the-shelf numerical optimizers [11]. However, in reality, the trajectory should be planned in real-time. To alleviate the high computational burden caused by the numerical optimization process and improve the algorithm's real-time convergence performance, efforts can be devoted to searching a near-optimal initial reference solution as suggested in [12] and [34]. However, designing a high-quality initial guess of trajectory for a constrained optimization problem is inherently difficult. Besides, there is no guarantee that one preassigned initial reference trajectory can work perfectly well for different parking maneuver cases.

To explore an effective alternative, a DNN-based real-time parking maneuver planner was proposed in our previous work [30], where DNNs were trained on the pre-generated parking trajectory dataset to learn the mapping relationship between the optimized states and control inputs. Later, the trained networks were used to serve as onboard trajectory planner, producing near-optimal control actions with vehicle current states as input of the network. In this way, the

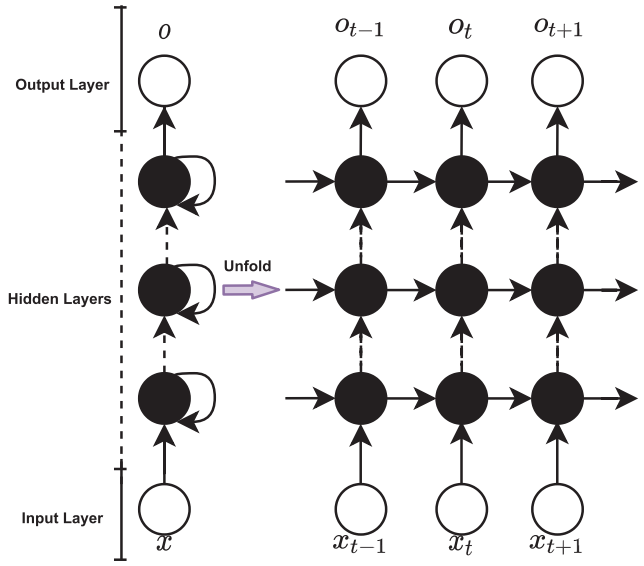

Fig. 2. An example of RDNN with 3 hidden layers.

long-standing challenge of addressing the trajectory optimization model in real-time becomes unnecessary. Developed upon our previous works, the RDNN parking maneuver planner introduced in this paper further extends the DNN-based method by taking advantages of a recurrent network structure. Moreover, a transfer learning strategy is designed such that the developed RDNN motion planner has the capability of suiting various types of AGVs. Specifically, the developed RDNN contains four main steps:

1) Optimal parking trajectory dataset generation;
2) RDNN construction;
3) Data aggregation and training;
4) Transfer layer construction.

Next, these four components will be discussed in detail.

A. Generation of Optimal Maneuver Trajectory Dataset

To construct the optimal maneuver trajectory dataset $\mathbb{D}$ for a particular parking scenario, the discretized version of the optimization problem (7) should be established and addressed iteratively with noise-perturbed initial conditions. Specifically, the dataset $\mathbb{D}$ which contains N_d optimal parking maneuver trajectories (e.g., $\|\mathbb{D}\| = N_d$) can be generated by following the main steps presented in Algorithm 1. Note that in (8), the temporal nodes are denoted as $\{t_k\}_{k=1}^{N_k}$ with $t_1 = 0$ and $t_{N_k} = t_f$. Here, N_k denotes the number of discrete time points. The vehicle state and control variables at t_k can then be abbreviated as x_k and u_k, respectively.

B. RDNN Construction and Training

Once the dataset $\mathbb{D}$ is formed, an RDNN network with long short-term memory (LSTM) architecture is constructed to learn the mapping relationships between the optimized states and control actions. Specifically, for the automatic parking mission, we aim to construct two RDNNs such that a near-optimal $u_k = [\eta_1(t_k), \eta_2(t_k)]^T$ can be produced with AGV states as the network input:

$$\begin{aligned}\eta_1(t_k) &\approx \mathscr{N}_1(x_k)\\ \eta_2(t_k) &\approx \mathscr{N}_2(x_k)\end{aligned} \tag{9}$$

In (9), $\mathscr{N}_1$ and $\mathscr{N}_2$ can be treated as the acceleration control network and the steering control network, respectively. Different from the fully-connected DNN applied in [30], the RDNN implicitly embeds memory effects into the model. It has been shown in a number of related works [35], [36] that a neural network with LTSM recurrent structure is able to

Algorithm 1 Optimal Parking Maneuver Dataset Generation

Input: N_d and $\mathbb{D} = \emptyset$;
Output: The final parking trajectory set $\mathbb{D}$;
/*Main procedure*/
Step 1: Generate time nodes $\{t_k\}_{k=1}^{N_k}$;
while $\|\mathbb{D}\| < N_d$ **do**
  Step 2: Randomly initialize the noise value $\Delta\zeta_x$;
  Step 3 Assign

$$x_{\text{ini}} = x_0 + \Delta\zeta_x$$

  Step 4: Form the parking trajectory optimization model (7)
    with noise-perturbed initial conditions;
  Step 5: Discretize the optimization model to form

$$\begin{aligned}\min_{x_k,u_k,\lambda_k^{(m)},\mu_k^{(m)}} \quad & J = \sum_{k=0}^{N_k} L(x_k,u_k) + \Phi(x_{N_k},t_k)\\ \text{s.t. } & \forall k \in \{1,\ldots,N_k\}, \quad \forall m \in \{1,\ldots,N_o\}\\ & x_{k+1} = f(x_k,u_k)\\ & x_0 = x_{\text{ini}}\\ & x_{N_k} = x_f\\ & (A^{(m)}T(x_k) - b^{(m)})^T\lambda_k^{(m)} - g^T\mu_k^{(m)} > \underline{d}\\ & Q^T(x_k)A^{(m)^T}\lambda_k^{(m)} + G^T\mu_k^{(m)} = 0\\ & \|A^{(m)^T}\lambda_k^{(m)}\| \le 1\\ & x^{\min} \le x_k \le x^{\max}\\ & u^{\min} \le u_k \le u^{\max}\\ & \lambda_k^{(m)} \ge 0, \quad \mu_k^{(m)} \ge 0\end{aligned} \tag{8}$$

  Step 6: Access the two-step parking trajectory planner [12]
    to address the optimization model (8);
  if the solver can successfully terminate (tolerance ϵ) **then**
    Step 7: Output the optimal result and perform

$$\mathbb{D} = \mathbb{D} \cup \{(x_k^*, u_k^*)\}$$

  else
    Step 8: Discard the current solution and perform

$$\mathbb{D} = \mathbb{D} \cup \emptyset$$

  end if
end while
Output the final dataset $\mathbb{D}$

outperform other networks in regards to predicting time series data, which might be more suitable for the considered mission scenario.

Fig. 3 provides a graphical illustration of an RDNN with 3 hidden layers, from where it is obvious that a recurrent structure allows neurons to form a direct cycle, thereby making the internal state to exhibit dynamic behaviour.

From Fig. 3, the output of the g-th network unit in j-th layer at discrete time point t_k can be determined by:

$$\begin{cases} z_{j,g,t_k} = \sigma_s(W_z x_{j,g,t_k} + U_z o_{j,g,t_{k-1}} + b_z) \\ u_{j,g,t_k} = \sigma_s(W_u x_{j,g,t_k} + U_u o_{j,g,t_{k-1}} + b_u) \\ h_{j,g,t_k} = \sigma_s(W_h x_{j,g,t_k} + U_h o_{j,g,t_{k-1}} + b_h) \\ \hat{c}_{j,g,t_k} = \sigma_h(W_c x_{j,g,t_k} + U_c o_{j,g,t_{k-1}} + b_c) \\ c_{j,g,t_k} = z_{j,g,t_k} c_{j,g,t_{k-1}} + u_{j,g,t_k} \hat{c}_{j,g,t_k} \\ o_{j,g,t_k} = h_{j,g,t_k} \sigma_h(c_{j,g,t_k}) \end{cases} \tag{10}$$

in which $j = 1, \ldots, N_l$ and $g = 1, \ldots, N_o$. Here, N_l and N_o represent the number of layer and neurons at each layer. x_{j,g,t_k} and o_{j,g,t_k} are, respectively, the input and output vectors of the LSTM unit. z_{j,g,t_k}, u_{j,g,t_k} and h_{j,g,t_k} denote the forget, update and output activation vectors. $\hat{c}_{j,g,t_k}$ and c_{j,g,t_k} are the cell input and state vectors. The activation functions σ_s and σ_h are given by:

$$\sigma_s(x) = \frac{1}{1+e^{-x}}, \quad \sigma_h(x) = \frac{e^{2x}-1}{e^{2x}+1}. \tag{11}$$

In (10), $W_{(\cdot)}$, $U_{(\cdot)}$ and $b_{(\cdot)}$ are weight matrices and bias vectors. They are updated during the network training process. To be more specific, the network training process can be viewed as an optimization problem with weight matrices and bias vectors being the decision variables. The performance index can be written in the form of:

$$\min \; L = \frac{1}{N_{\mathrm{Tr}}} \sum_{i=1}^{N_{\mathrm{Tr}}} \frac{1}{N_k} \sum_{j=1}^{N_k} [x(t_{jT}) - \hat{x}(t_{jT})]^2 \tag{12}$$

where $x(t_{jT})$ and $\hat{x}(t_{jT})$ represent the target and approximated network outputs. N_{Tr} stands for the size of the training dataset, and $T = \frac{t_f}{N_k}$. Note that we divide $\mathbb{D}$ into three subsets (e.g., $\mathbb{D} = \mathbb{D}_{\mathrm{Tr}} \cup \mathbb{D}_{\mathrm{Te}} \cup \mathbb{D}_{\mathrm{V}}$). They are, respectively, the training set $\mathbb{D}_{\mathrm{Tr}}$ containing 70% data of $\mathbb{D}$, the testing set $\mathbb{D}_{\mathrm{Te}}$ containing 20% data of $\mathbb{D}$, and the validation set $\mathbb{D}_{\mathrm{V}}$ (10% data of $\mathbb{D}$). These subsets are used to train, test, and validate the network such that it can acquire the ability to map relationship between optimized states and control variables. The Adam optimizer [37], which leverages the stochastic gradient descent (SGD) method and an adaptive learning rate strategy, is applied to train the network model. The SGD algorithm process is exactly the same as suggested in [37]. For the adaptive learning process, if ζ stands for the learning rate, then its updated value $\hat{\zeta}$ can be calculated by $\hat{\zeta} = \zeta e^{-\lambda s_g / s_d}$, in which $\lambda \in [0.7, 1]$. s_g and s_d are the current iteration index and the decay step.

C. Data Aggregation and Retraining

To enhance the network approximation accuracy, a data aggregation (DA) strategy is designed and applied. The central idea of this strategy is to retrain the RDNN on an updated training dataset. Specifically, by testing the RDNN on $\mathbb{D}_{\mathrm{Te}}$, we are able to detect a number of parking maneuver samples which cannot be accurately approximated. These samples will be collected to a so-called bad performance dataset $\mathbb{D}_{\mathrm{d}}$ and adhered to $\mathbb{D}_{\mathrm{Tr}}$. Subsequently, the network is re-trained on $\mathbb{D}_{\mathrm{Tr}}$. The motivation for the use of DA is to allow some emphasis on the detected $\mathbb{D}_{\mathrm{d}}$ such that the network can recover from past mistakes. To be more precise, the DA and re-training process are summarised in the pseudocode (see Algorithm 2).

Algorithm 2 Data Aggregation and Training Process

procedure
  Step 1: Initialize $\mathbb{D}_{\mathrm{d}} = \emptyset$;
  Step 2: Specify the acceptable thresholds of trajectory approximation error ϵ_a and terminal state error ϵ_t;
  Step 3: Train the RDNN on $\mathbb{D}_{\mathrm{Tr}}$;
  Step 4: Test the trained RDNN on $\mathbb{D}_{\mathrm{Te}}$;
  for $m := 1, 2, \ldots, \|\mathbb{D}_{\mathrm{V}}\|$ **do**
    if the approximation error and terminal state error of the mth trajectory among $\mathbb{D}_{\mathrm{V}}$ can stay within the threshold **then**
      Step 5: Perform
      $$\mathbb{D}_{\mathrm{d}} = \mathbb{D}_{\mathrm{d}} \cup \emptyset$$
    else
      Step 6: Insert the m-th parking trajectory to $\mathbb{D}_{\mathrm{d}}$;
    end if
  end for
  Step 7: Perform
  $$\mathbb{D}_{\mathrm{Tr}} = \mathbb{D}_{\mathrm{Tr}} \cup \mathbb{D}_{\mathrm{d}}$$
  Step 8: Retrain the RDNN on $\mathbb{D}_{\mathrm{Tr}}$;
end procedure

D. Transfer Learning Strategy

One limitation of the proposed RDNN parking maneuver planner is that this method might not be easily adapted to suit different types of AGVs. This is because the network is trained on the parking trajectory set of a specific vehicle. A direct implementation of the trained RDNN to another AGV may result in algorithm performance degradation. However, in real-world automatic parking scenarios, differences always exist in terms of the AGV dynamics. Owing to the high potential for real-time applications, we make an attempt to address the this issue and extend the RDNN motion planner such that it can be adapted to suit various types of AGVs. The proposed method takes the advantage of transfer learning (TL) techniques. More specifically, let us assume a large set of parking trajectories $\mathbb{D}_A$ is generated for a particular AGV (e.g., vehicle A). Besides, we define $\mathbb{D}_B$ as a limited-size parking trajectory dataset for another AGV (e.g., vehicle B). The main objective of transfer learning is to reinforce the mapping relationship $\mathscr{N}_A(x)\colon x \mapsto u, (x,u) \in \mathbb{D}_A$ by maximally exploiting the knowledge of $\mathscr{N}_B(x)\colon x \mapsto u, (x,u) \in \mathbb{D}_B$. To achieve this goal, the

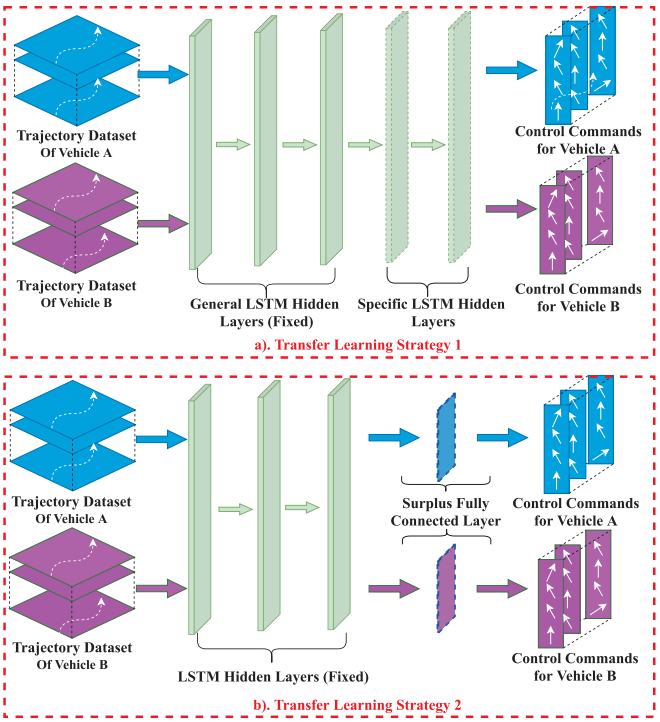

Fig. 3. Two TL schemes.

structure of the RDNN motion planner is modified and two transfer learning strategies are proposed and visualized in Fig. 4:

- TL strategy 1: The LSTM layers are divided into general hidden layers and specific hidden layers. After training the network on the optimal parking trajectory dataset of vehicle A, we fix the network parameters of the general hidden layers and retrain the specific layer parameters on the dataset of vehicle B.
- TL strategy 2: A surplus layer will be inserted between the last hidden LTSM layer and the final output layer after training the RDNN on the dataset of vehicle A. A retraining process will be carried out for this surplus layer using the limited-size parking trajectory dataset of vehicle B. Note that in this strategy, only the weight and bias parameters of the surplus layer will be updated.

In TL strategy 1, it assumes that the general hidden layers can abstract the behaviour of the parking maneuver once the first training is completed. Subsequently, the retraining on specific hidden layers plays a dominant role in determining the maneuver action for other vehicles. In TL strategy 2, on the other hand, it is assumed that the maneuver action of another vehicle can be captured by learning additional features of the dataset B.

E. Overall Algorithm Structure

To detail the proposed RDNN-based real-time parking trajectory planning scheme, general offline and online operations are structured and presented in Algorithm 3.

Algorithm 3 RDNN-Based Parking Motion Control Process

1: **Offline:** Establish the trajectory optimization model given by (7)
2: Perform Algorithm 1 to generate the trajectory dataset $\mathbb{D}$;
3: Construct and train the RDNN on $\mathbb{D}_{\text{Tr}}$;
4: Perform Algorithm 2 to enhance the approximation ability of the RDNN;
5: /*Main Iteration*/
6: **Online:** At each time node $k := 0, 1, \ldots$;
7: (a) Obtain the current AGV state x_k;
8: (b) Compute optimal control actions via

$$u_k := \mathcal{N}(x_k)$$

9: (c) Apply u_k to the AGV kinematic system (1) until t_{k+1};
10: (d) Calculate x_{k+1} via numerical integration:

$$x_{k+1} = x_k + \int_{t_k}^{t_{k+1}} f(x_k, \mathcal{N}(x_k))dt$$

11: (e) Set $k = k + 1$;
12: (f) Re-perform steps a)-e) for t_{k+1}.

IV. ALNN-Based Trajectory Tracking Controller

In this section, an ALNN-based trajectory tracking controller is introduced so as to provide the analytical form of the control variables and to track the AGV reference linear and angular velocity profiles generated via the RDNN-based trajectory planner. Theoretical results are established to guarantee the tracking performance and stability of the proposed tracking controller.

A. AGV Dynamics

The control of the two front wheels of the AGV is considered in this section. The dynamics of the AGV can be written in the following general form [38]:

$$M(q)\dot{R}+C(q,\dot{q})R+G(q)+F(\dot{q})+\tau_d=B(q)\tau \tag{13}$$

where $q = [p_x, p_y, \theta]^T$, $\omega = \frac{\upsilon \tan(\phi)}{L}$, and $R = [\upsilon, \omega]^T$. $\tau = [\tau_1, \tau_2]^T$ denotes the control input. Here, τ_1 and τ_2 are, respectively, the toques produced by the left and right motors. The matrices $M(q)$, $C(q,\dot{q})$, $G(q)$, $F(\dot{q})$, and $B(q)$ are in the form of

$$\begin{aligned} M(q) &= \begin{bmatrix} 2\frac{I_\omega}{r^2} + 2m_\omega + m_\upsilon & 0 \\ 0 & 2\frac{b^2}{r^2}I_\omega + I_z \end{bmatrix}, \\ C(q,\dot{q}) &= \begin{bmatrix} 0 & -dm_\upsilon\dot{\theta} \\ dm_\upsilon\dot{\theta} & 0 \end{bmatrix}, \quad G(q) = 0, \\ F(\dot{q}) &= F_c\dot{q}, \quad B(q) = \begin{bmatrix} \frac{2}{r} & \frac{2}{r} \\ -\frac{2}{r}b & \frac{2}{r}b \end{bmatrix}. \end{aligned} \tag{14}$$

Definitions of these matrices as well as some other vehicle-related parameters are introduced in Table I. Note that in this paper $F(\dot{q}) = F_c\dot{q}$ is supposed to be an uncertain term with the unknown coefficient matrix F_c. Based on (13), the dynamic model with respect to R can be written as

$$\dot{R} = -M^{-1}(q)(C(q,\dot{q})R + D(\dot{q}) - B(q)\tau) \tag{15}$$

TABLE I
DEFINITIONS OF MATRICES AND PARAMETERS

$M(q)$:	The inertia matrix
$C(q,\dot{q})$:	The centripetal matrix
$G(q)$:	The gravitation term
$F(\dot{q})$:	The surface friction matrix
$B(q)$:	The input transformation matrix
τ_d:	The unknown bounded disturbance
m_v, m_ω:	The mass of the AGV and the motorized wheel
I_ω, I_z:	The inertia of the motorized wheel and the AGV
r, $2b$:	The radius of the wheel and the length of the rear axle
d:	The distance between the centre of mass and the centre of the back axle

where $D(\dot{q}) = F(\dot{q}) + \tau_d$.

B. ALNN-Based Controller Design and Stability Analysis

From the definition of $M(q)$ and $C(q,\dot{q})$ given by (14), it is easy to verify that the $M(q)$ is symmetric and positive definite and $M - 2C(q,\dot{q})$ is skew-symmetric. If the error between the pre-planned velocity vectors $R_r = [v_r, \omega_r]^T$ and R is denoted as $e(t) = R_r(t) - R(t)$, then the main objective of the ALNN tracking controller is to robustly steer the error term $e(t)$ to a small neighbourhood around the origin in the presence of unknown disturbances and uncertainties $D(\dot{q})$. Before presenting the proposed ALNN-based controller, some assumptions are made below.

Assumption 1: The AGV reference velocity trajectories and their derivatives $(R_r, \dot{R}_r)$ are considered to be continuous and bounded.

Assumption 2: The system uncertain term $D(\dot{q})$ is considered to be bounded. That is, $\|D(\dot{q})\| \le \bar{D}$. Here, $\bar{D}$ is an unknown positive constant.

Assumption 3: For any $(q,\dot{q}) \in \mathbb{R}^2$, $C(q,\dot{q})$ is bounded. That is, $\underline{C} \le \|C(q,\dot{q})\| \le \bar{C}$, where $\underline{C}$ and $\bar{C}$ are two positive constants.

Assumption 1 prevents the vehicle from jerking. The static friction limit is related to the tire, road friction coefficient and positive pressure. Besides, the tires rely on friction to provide centripetal force. Therefore, the other two assumptions are also reasonable.

Differentiating the error term e, the tracking error dynamics can be written as

$$\begin{aligned}\dot{e} &= \dot{R}_r - \dot{R} \\ &= \dot{R}_r + M^{-1}(q)(C(q,\dot{q})R + D(\dot{q}) - B(q)\tau) \end{aligned} \tag{16}$$

Define a Lyapunov function in the form of $V_1 = \frac{1}{2}e^T M e$, then by taking the derivative of V_1, we can obtain

$$\begin{aligned}\dot{V}_1 &= e^T M\dot{e} + \frac{1}{2}e^T\dot{M}e \\ &= e^T\left(M(\dot{R}_r - \dot{R}) + \frac{1}{2}\dot{M}e\right) \\ &= e^T\left(M\dot{R}_r + CR + D - B\tau + \frac{1}{2}\dot{M}e\right) \\ &= e^T\left(M\dot{R}_r + CR_r + D - B\tau + \left(\frac{1}{2}\dot{M} - C\right)e\right) \\ &= e^T\left(M\dot{R}_r + CR_r + D - B\tau\right) \end{aligned} \tag{17}$$

Note that in (17), the skew-symmetry property of $M-2C(q,\dot{q})$ is applied to replace $\frac{1}{2}e^T\dot{M}e$ by the term e^TCe. Define $\xi = [R, R_r, e]^T$ and suppose there exists a continuous input function τ^* such that the error term can be steered to a small neighbourhood of the origin, leading to $\lim_{t\mapsto+\infty}|R_r - R| = 0$. Then, there exists an ideal NN-based control law $\tau^*(\xi) = -B^{-1}(W^{*^T}S(V^{*^T}\xi))$ with constant weight matrices (W^*, V^*) such that R_r can be tracked. More precisely, substituting $\tau^*(z)$ into the AGV system yields

$$-W^{*^T}S(V^{*^T}\xi) + \epsilon_\tau = M\dot{R}_r + CR_r$$

in which $\epsilon_\tau(z)$ denotes the NN approximation error, and $S(\cdot)$ is the sigmoid activation function.

Assumption 4: The ideal weight matrices and the NN approximation error are upper bounded. That is, $\|W^*\| \le \bar{W}$, $\|V^*\| \le \bar{V}$, and $|\epsilon_\tau \le \bar{\epsilon}_l|$, where $\bar{W}$, $\bar{V}$ and $\bar{\epsilon}_l$ are three positive constants.

Now, let us construct a robust NN tracking control law in the form of

$$\begin{cases}\tau = -B^{-1}(\tau_{\rm nn} + \tau_b) \\ \tau_{\rm nn} = \hat{W}^T S(\hat{V}^T\xi) \\ \tau_b = -\left[K_p(\|\hat{S}'\hat{V}^T\xi\|^2 + \|\xi\hat{W}^T\hat{S}'\|^2 + 1) + K_s\right]e\end{cases} \tag{18}$$

where $\tau_{\rm nn}$ is the approximation of the optimal control law τ^* with $\hat{W}$ and $\hat{V}$ being the estimation of W^* and V^*, respectively. τ_b can be viewed as a bounding control term, which is responsible for restricting the state bounds of the AGV dynamics. K_p and K_s are positive definite diagonal matrices. $\hat{S} = S(\hat{V}^T\xi)$, and $\hat{S}'$ denotes its derivative.

If the estimation errors of weight matrices are defined as $\tilde{W} = \hat{W} - W^*$ and $\tilde{V} = \hat{V} - V^*$, then based on Taylor expansion, the approximation error between $\tau_{\rm nn}$ and τ^* can be described as

$$\begin{aligned}\tau_{\rm nn}(\xi) - \tau^*(\xi) &= \hat{W}^T S(\hat{V}^T\xi) - W^{*^T}S(V^{*^T}\xi) \\ &= \hat{W}^T\hat{S}'\tilde{V}^T\xi + \tilde{W}^T(\hat{S} - \hat{S}'\hat{V}^T\xi) + \Delta_\tau \end{aligned} \tag{19}$$

in which Δ_τ stands for the residual term. It was shown in [39] that Δ_τ satisfies

$$\|\Delta_\tau\| \le \|W^*\|\|\hat{S}'\hat{V}^T\xi\| + \|V^*\|\|\xi\hat{W}^T\hat{S}'\| + \|W^*\| \tag{20}$$

By substituting (18) and (20) into (17), $\dot{V}_1$ can be further written as

$$\begin{aligned}\dot{V}_1 &= e^T\left(-W^{*^T}S(V^{*^T}\xi) + \epsilon_\tau + D + \hat{W}^T S(\hat{V}^T\xi) + \tau_b\right) \\ &= e^T\left(\hat{W}^T\hat{S}'\tilde{V}^T\xi + \tilde{W}^T(\hat{S} - \hat{S}'\hat{V}^T\xi) + \epsilon_\tau + D + \Delta_\tau + \tau_b\right) \\ &\le e^T\left(\Lambda_W + \epsilon_\tau + D + \|W^*\|\|\hat{S}'\hat{V}^T\xi\| + \|V^*\|\|\xi\hat{W}^T\hat{S}'\|\right. \\ &\quad \left. + \|W^*\| - K_p\|\xi\hat{W}^T\hat{S}'\|^2 e - K_p\|\hat{S}'\hat{V}^T\xi\|^2 e - K_p e - K_s e\right) \\ &\le e^T\left(\Lambda_W + \bar{\epsilon}_l + \bar{D} + \bar{W} - K_p e - K_s e\right) + \bar{V}\|\xi\hat{W}^T\hat{S}'\|e \\ &\quad + \bar{W}\|\hat{S}'\hat{V}^T\xi\|e - K_p\|\xi\hat{W}^T\hat{S}'\|^2 e^2 - K_p\|\hat{S}'\hat{V}^T\xi\|^2 e^2 \\ &\le e^T\left(\Lambda_W + \bar{\epsilon}_l + \bar{D} + \bar{W} - K_p e - K_s e\right) + \frac{\bar{V}^2}{4K_p} + \frac{\bar{W}^2}{4K_p} \end{aligned} \tag{21}$$

where $\Lambda_W = \hat{W}^T\hat{S}'\tilde{V}^T\xi + \tilde{W}^T(\hat{S} - \hat{S}'\hat{V}^T\xi)$. According to (21), an adaptive learning law-based robust NN controller can be

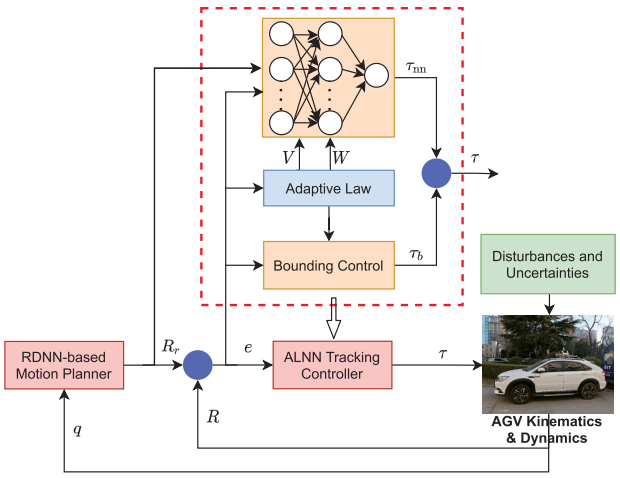

Fig. 4. ALNN-based parking control scheme.

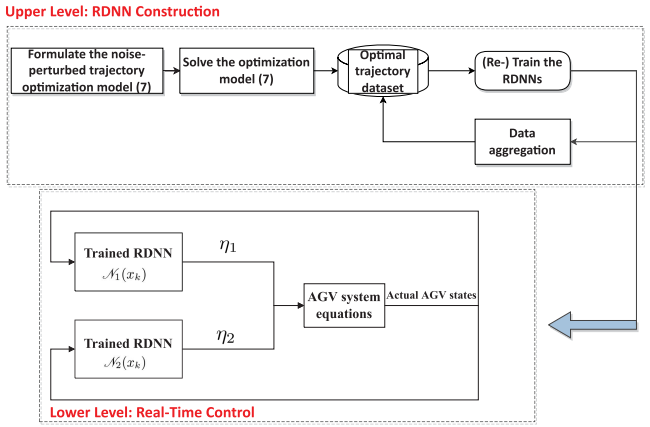

Fig. 5. Kinematic control process.

designed (annotated as ALNN), and we are able to derive the following theorem.

Theorem 1: Given the AGV dynamics and tracking error systems in the form of (13) and (16). If Assumptions 1–4 can be satisfied and the multiple layer ALNN weights are tuned via:

$$\begin{cases} \dot{\hat{V}} = -\Gamma_v\left(\delta_v \hat{V} + \xi \hat{W}^T \hat{S}' e\right) \\ \dot{\hat{W}} = -\Gamma_w\left(\delta_w \hat{W} + (\hat{S} - \hat{S}' \hat{V}^T \xi) e\right) \end{cases} \tag{22}$$

in which δ_v and δ_w are two positive design parameters, $\Gamma_v = \Gamma_v^T > 0$ and $\Gamma_w = \Gamma_w^T > 0$ are constant matrices, then we have the following properties by applying the control law (18):

(1) The close-loop variables $(R, e, \hat{V}, \hat{W})$ will be maintained in a compact set during the control process;

(2) The trajectory tracking error e, together with the weight estimation error $(\tilde{V}, \tilde{W})$, will be eventually steer to a small neighbourhood of the origin.

Proof: The proof is detailed in the Appendix. ■

C. Tracking Controller Framework

To clearly demonstrate how the RDNN-based motion planner and the ALNN-based parking control scheme work, the key algorithm processes are visualized in Fig. 4 and Fig. 5.

V. Experimental Study

This section validates the effectiveness of the proposed RDNN-based motion planning method, the transfer learning strategies as well as the ALNN tracking control algorithm for the AGV automatic parking problem. The experiment was executed on two AGV models: a simulated autonomous vehicle and a real intelligent car (2012 BYD Su Rui 1.5 TID) as illustrated in Fig. 6(a) and Fig. 6(b). Meanwhile, we consider two real-world parking scenarios: a parallel parking case and a vertical parking case as shown in Fig. 6(c) and Fig. 6(d), respectively. The AGV model/parking scene-related parameters are tabulated in Table II. The ranges of variables

TABLE II
AGV Model/Parking Scene-Related Parameters

Parameter	Value		Parameter	Value	
	Vehicle A	Vehicle B		Scenario 1	Scenario 2
m, kg	1723	1395	W, m	5	5
I_z, kg·m^2	4175	4192	l_L, m	5	2.5
S_l, m	4	4.680	l_W, m	2.5	5
S_w, m	1.988	1.765	$p_x(0)$, m	-5.14	-6.14
b, m	0.872	0.76	$p_y(0)$, m	1.41	1.62
d, m	1.468	1.62	$\theta(0)$, rad	0.1047	0.1309
r, m	0.335	0.335	$v(0)$, m·s^{-1}	0	0
l, m	2.70	2.66	$\phi(0)$, rad	0	0

appeared in the optimization model (7) are assigned as:

$$\begin{cases} |p_x| \le 20\text{m} & |p_y| \le 5\text{m} \quad |\theta| \le 1.5708\text{rad} \\ |v| \le 2\text{m/s} & |a| \le 0.5\text{m/s}^2 \quad |\phi| \le 0.5760\text{rad} \\ |\eta_1| l \le \dot{k}^{\max} \phi^{\max} & |\eta_2| \le 0.5\text{m/s}^3 \quad t \le 50\text{s} \end{cases}$$

Here, $\dot{k}^{\max} = 0.6$ and $\phi^{\max} = 0.5760$. By following the procedure detailed in Algorithm 1, 100000 optimal parking trajectories were produced for vehicle A, while a limited 20000 trajectories were generated for vehicle B. The parking trajectory datasets are used to construct the RDNN-based parking maneuver planner. Here, the initial condition perturbation for $(p_x(0), p_y(0), \theta(0))$ was supposed to be normally distributed on $[-1, 1\text{m}]$, $[-0.5, 0.5\text{m}]$ and $[-0.0873, 0.0873\text{rad}]$, respectively. As for the uncertain coefficient matrix F_c and external disturbance τ_d, their elements are supposed to be uniformly distributed on $[0, 0.01]$ and $[-0.01, 0.01]$, respectively. Regarding the RDNN motion planner and ALNN tracking controller, their structural parameters are assigned as follows: $N_l = 5$, $N_o = 30$, $N_k = 100$, $\lambda = 0.8$, $\zeta_0 = 10^{-4}$, Γ_v, $\Gamma_w = \text{diag}\{5\}$, $\delta_w = 0.5$, and $\delta_v = 0.25$. The network structural parameters of the RDNN motion planner are trained on the pre-generated parking trajectory dataset offline, while the structural parameters of the ALNN motion controller are updated using the online update law (22).

As shown in [30], the depth of the network tends to have significant impact on the approximation of optimal parking trajectories. Hence, extensive test trials were carried out to determine a proper number of RDNN layers for the automatic parking problem and the pre-generated optimal trajectory

Fig. 6. AGV models and parking scenarios.

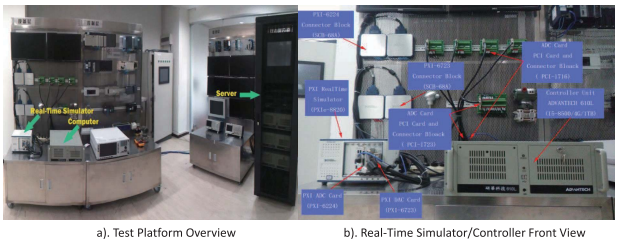

Fig. 7. Test platform.

dataset. The results reveal that constructing the RDNN with more than five hidden layers is unnecessary for the investigated problem. However, for other motion planning problems where the system state-control relationships are relatively complex or highly nonlinear, a deeper network might bring more benefits. Besides, efforts have been devoted to select a proper length of the RDNN input sequence. After several tuning tests, we found that five sequential AGV kinematic states can result in best final approximation results. The initial weights of the ALNN are all assigned as zero, whereas the diagonal components of K_p and K_s are set to 0.25 and 0.5, respectively. The real-time sampling interval is set to $T = 0.1$s. Regarding the test platform, an illustration is visualized in Fig. 7, from where it can be observed that three main components are included in the platform: 1). A Dell EMC PowerEdge R930 rack server which is responsible for producing the parking trajectory dataset and training the RDNN; 2). An IPC-610MB-30LDE/I5-2400/DDR3 industrial PC which is responsible for creating executable scripts via LabVIEW real-time mode; and 3). An NI PXI-8820 embedded controller which is responsible for testing the performance of the RDNN motion planner and the ALNN tracking control algorithm.

A. Results and Discussion

First, attention is given to highlight the merit of applying the recurrent network structure. We evaluate the trajectory planning and control performance by applying the DNN-based direct recalling strategy reported in [30] and the RDNN-based strategy designed in this study for the two considered parking scenarios. In addition, to analyze the impact of the DA strategy on the algorithm performance, comparison between the RDNN-based approach with and without equipping the DA process (denoted as RDNN-DA and RDNN) is also carried out.

TABLE III
PARKING PLANNING AND CONTROL RESULTS: VEHICLE A

Different Methods	Scenario 1 (Process error)			Scenario 1 (Terminal error)		
	$\int_0^{t_f} e_x^2\,dt$	$\int_0^{t_f} e_y^2\,dt$	$\int_0^{t_f} e_\theta^2\,dt$	$\lvert p_{xf}\rvert$	$\lvert p_{yf}\rvert$	$\lvert \theta_f\rvert$
DNN in [26]	0.1018	0.1652	0.0348	0.018	0.11	0.0262
RDNN	0.0617	0.1180	0.0343	0.014	0.05	0.0125
RDNN-DA	0.0470	0.0179	0.0277	0.010	0.00	0.0087
Different Methods	Scenario 2 (Process error)			Scenario 2 (Terminal error)		
	$\int_0^{t_f} e_x^2\,dt$	$\int_0^{t_f} e_y^2\,dt$	$\int_0^{t_f} e_\theta^2\,dt$	$\lvert p_{xf}\rvert$	$\lvert p_{yf}\rvert$	$\lvert \theta_f\rvert$
DNN in [26]	0.0338	0.0681	0.0063	0.1287	0.15	0.0082
RDNN	0.0049	0.0077	0.0009	0.0996	0.11	0.0074
RDNN-DA	0.0011	0.0049	0.0004	0.0208	0.07	0.0054

Fig. 8 demonstrates the trajectory planning and actual tracking results of using the three strategies for the parallel parking scenario. Specifically, in the left column of Fig. 8, the AGV maneuver trajectories are shown, whereas in the middle column of Fig. 8, the error profiles between the optimized reference and the actual posture are shown. Besides, in the right column of Fig. 8, the linear and angular velocity profiles obtained via different methods are visualized. From the obtained results, it is clear that differences can be detected in the actual parking trajectory profiles. The RDNN-DA methods tends to have the best trajectory planning and control performance than its counterparts, as the corresponding maneuver trajectories are closer to the optimized references. Moreover, according to the linear and angular velocity profiles, it can be verified that the ALNN algorithm is able to track the AGV motion command profiles.

Fig. 9 illustrates the maneuver planning and command tracking results obtained via the three strategies for the vertical parking scenario. Similar to the results presented in Fig. 8, the best maneuver planning and command tracking performance are achieved by applying the RDNN-DA approach. Hence, we may conclude that certain advantages can be acquired if a recurrent network structure is employed to approximate the optimal trajectory-steering relationship and then act as the online motion planner. This can be explained by the fact that an RDNN is more likely to fully exploit the functional relationship between the optimal state and control variables, as they are all time series data in nature.

Furthermore, by comparing the results obtained via the RDNN and RDNN-DA, one may conclude that using the DA process in the network training phase is helpful for enhancing the approximation accuracy, as the network tends to recover from past mistakes.

B. Comparative Studies

Monte-Carlo experiments are employed to further test and compare the performance and robustness of applying different parking maneuver planning and control approaches.

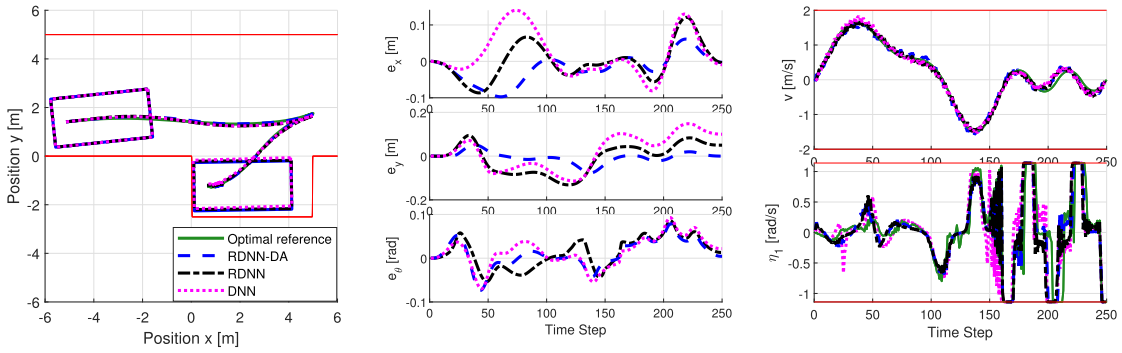

Fig. 8. Parking maneuver and tracking results for vehicle A: Scenario 1 (The target posture is $[0.65; -1.25; 0]$).

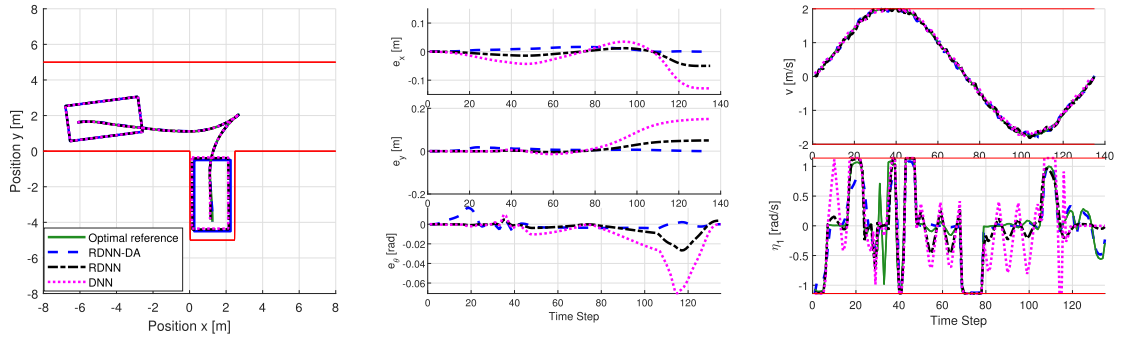

Fig. 9. Parking maneuver and tracking results for vehicle A: Scenario 2 (The target posture is $[1.25; -3.97; 1.5708]$).

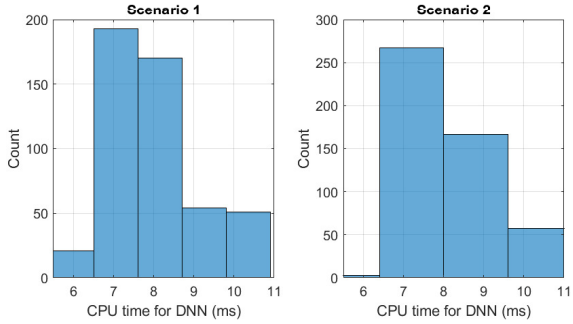

Fig. 10. Computational performance of DNN-based approach.

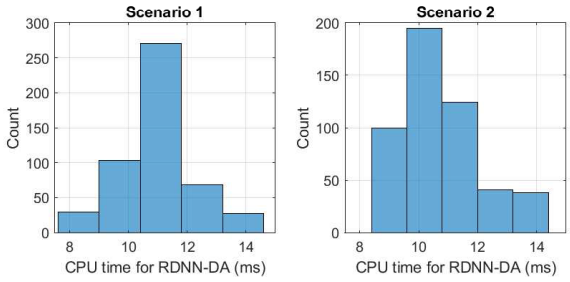

Fig. 11. Computational performance of RDNN-DA approach.

Specifically, 500 trials were executed and statistical results are tabulated in Table III. Note that in this table, two types of performance indicators are adopted. That is, the process error ($\int_0^{t_f} e_x^2 dt, \int_0^{t_f} e_y^2 dt, \int_0^{t_f} e_\theta^2 dt$) and the terminal error ($|p_{xf}|, |p_{yf}|, |\theta_f|$). According to the results reported in Table III, it is obvious that implementing the proposed RDNN-DA approach can result in a relatively-higher probability of guiding the AGV to the target parking point. That is, a smaller terminal posture error, together with a smaller tracking process error, is likely to be obtained. This further confirms the advantages of using the proposed approach for the considered automatic parking maneuver problems.

To analyze the real-time applicability of the proposed RDNN-DA algorithm, attention is given to the average process time required to execute the online motion planning algorithm. The computational results for the Monte-Carlo experiments are collected to construct the histograms and visualized in Fig. 10 and Fig. 11. The average computational effort (process time) required by the proposed RDNN-DA approach in two scenarios are 10.7178ms and 10.9719ms, respectively, while the DNN-based approach requires 7.8099ms and 7.9109ms. Although the process time is increased compared to the DNN-based approach, it is still less than the sampling interval, which confirms the real-time applicability of the algorithm. Moreover, the statistical results reported in Table III indicate that the proposed approach has higher precision in parking maneuver. We hereby conclude that the proposed approach is able to produce near optimal results, while requiring significantly lower computational efforts. This allows more possibilities in real-world applications.

C. Performance Evaluation of Transfer Learning Schemes

In the previous subsection, we have shown the effectiveness and potential of applying the designed parking maneuver

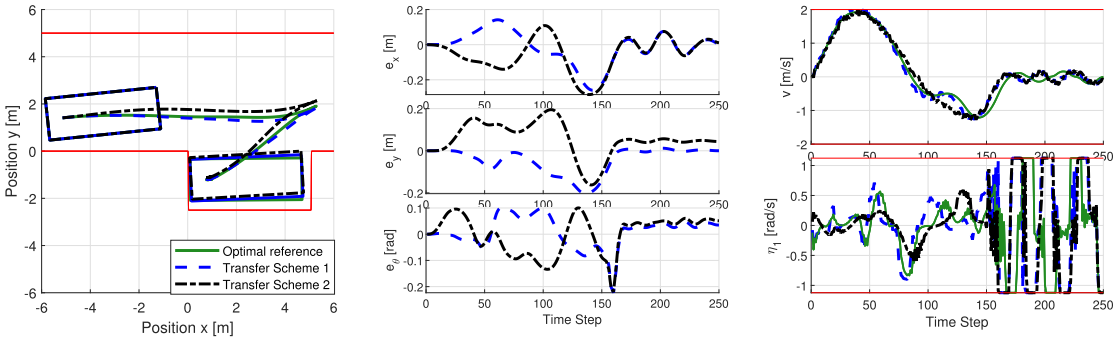

Fig. 12. Performance evaluation of different TL schemes on vehicle B: Scenario 1 (The target posture is $[0.7508; -1.25; 0]$).

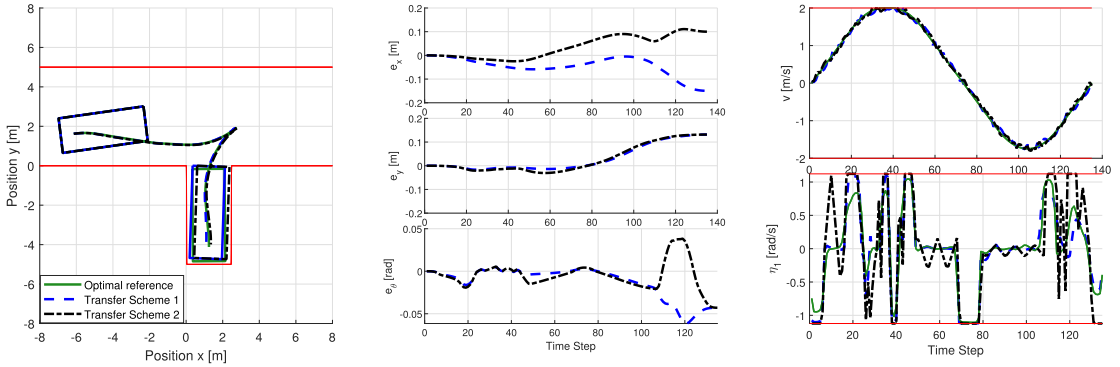

Fig. 13. Performance evaluation of different TL schemes on vehicle B: Scenario 2 (The target posture is $[1.25; -4.12; 1.5708]$).

TABLE IV

TL SCHEMES PERFORMANCE EVALUATION ON TARGET DOMAIN (VEHICLE B)

Different TL schemes	Scenario 1 (Process error)			Scenario 1 (Terminal error)		
	$\int_0^{t_f} e_x^2\,dt$	$\int_0^{t_f} e_y^2\,dt$	$\int_0^{t_f} e_\theta^2\,dt$	$\lvert p_{xf}\rvert$	$\lvert p_{yf}\rvert$	$\lvert \theta_f\rvert$
TL scheme 1	0.2265	0.1504	0.0881	0.011	0.0441	0.0436
TL scheme 2	0.3130	0.2422	0.1292	0.006	0.1319	0.0599
Different TL schemes	Scenario 2 (Process error)			Scenario 2 (Terminal error)		
	$\int_0^{t_f} e_x^2\,dt$	$\int_0^{t_f} e_y^2\,dt$	$\int_0^{t_f} e_\theta^2\,dt$	$\lvert p_{xf}\rvert$	$\lvert p_{yf}\rvert$	$\lvert \theta_f\rvert$
TL scheme 1	0.0476	0.0487	0.0067	0.1496	0.1316	0.0349
TL scheme 2	0.0462	0.0539	0.0036	0.1000	0.1317	0.0351

planning and control algorithm in real-time applications. Here, we focus on the test and validation of the proposed two parking transfer learning schemes. By defining the trajectory dataset of vehicle A as the source domain, the TL scheme 1 and TL scheme 2 illustrated in Fig. 4 are performed to retrain the network on the trajectory dataset of vehicle B (target domain). Subsequently, the trained network will serve as the online parking maneuver planner for vehicle B. Note that for the transfer learning scheme 1, the first three hidden layers of the RDNN are specified as general hidden layers, while the last two are specified as the specific hidden layers.

Figs. 12 and 13 demonstrates, respectively, the AGV maneuver profiles and the velocity tracking trajectories obtained via the two TL schemes for different parking scenarios. Compared to the parking maneuver profiled illustrated in Figs. 8 and 9, the posture errors shown in Figs. 12 and 13 become larger. However, the algorithm can still steer the vehicle into a small neighbourhood of the target point. 500 Monte-Carlo experiments were carried out and the statistical results are shown in Table IV. From Table IV and Figs. 12 and 13, it is clear that both the process and terminal errors experience an increase in comparison to the results shown in Table III. However, the two parking missions can still be fulfilled with a relatively high accuracy, which indicates that the two proposed TL schemes are able to achieve a good transferability for the considered parking problem.

VI. CONCLUSION

In this paper, we investigated the problem of real-time parking maneuver planning and tracking control for autonomous ground vehicles (AGVs). A novel RDNN-based motion planning approach, along with two transfer learning strategies, has been structured to plan the parking movement for different AGVs in real-time. Subsequently, an ALNN-oriented control algorithm has been introduced so as to track the planned maneuver trajectory. Based on the obtained experimental results and comparative analysis, we have revealed that:

- Using the proposed RDNN-based motion planner can better exploit the functional relationship between the optimal state and control variables, thus resulting in enhanced approximation accuracy to the optimal linear and angular velocity command profiles.
- Adopting the ALNN-based control algorithm is able to accurately track the optimized AGV motion command profiles with guaranteed tracking performance and algorithm stability.

- The trust on the real-time applicability of the proposed deep learning-based automatic parking motion planning and control scheme can be appreciated.
- The proposed TL strategies have the capability of extending the developed RDNN motion planner such that it can be adapted to suit different types of AGVs.

As a consequence, the deep learning-based trajectory planning and control strategy suggested in this study may serve as a promising tool for the considered automatic parking problem.

APPENDIX
PROOF OF THEOREM 1

Proof: For Part (1), due to the fact that V^* and W^* are constant vectors, we have $\dot{\tilde{V}} = \dot{\hat{V}}$ and $\dot{\tilde{W}} = \dot{\hat{W}}$, respectively. Define $V_2 = \frac{1}{2}\tilde{V}^T\Gamma_v^{-1}\tilde{V} + \frac{1}{2}\tilde{W}^T\Gamma_w^{-1}\tilde{W}$. Then for the Lyapunov function $V_3 = V_1 + V_2$, the following inequality holds

$$\begin{aligned}\dot{V}_3 &= \dot{V}_1 + \tilde{V}^T\Gamma_v^{-1}\dot{\tilde{V}} + \tilde{W}^T\Gamma_w^{-1}\dot{\tilde{W}} \\ &= \dot{V}_1 + \tilde{V}^T\Gamma_v^{-1}\dot{\hat{V}} + \tilde{W}^T\Gamma_w^{-1}\dot{\hat{W}} \\ &\le e^T\left(\bar{\epsilon}_l + \bar{D} + \bar{W} - K_p e - K_s e\right) \\ &\quad + \frac{\bar{V}^2}{4K_p} + \frac{\bar{W}^2}{4K_p} - \delta_v\tilde{V}^T\hat{V} - \delta_w\tilde{W}^T\hat{W} \end{aligned} \tag{23}$$

Notice that one has

$$\begin{aligned} 2\tilde{V}^T\hat{V} &= \|\tilde{V}\|^2 + \|\hat{V}\|^2 - \|V^*\|^2 \ge \|\tilde{V}\|^2 - \|V^*\|^2 \\ \tilde{W}^T\hat{W} &= \|\tilde{W}\|^2 + \|\hat{W}\|^2 - \|W^*\|^2 \ge \|\tilde{W}\|^2 - \|W^*\|^2 \end{aligned} \tag{24}$$

Then, (23) can be further transcribed to

$$\begin{aligned}\dot{V}_3 &\le e^T\left(\bar{\epsilon}_l + \bar{D} + \bar{W} - K_p e - K_s e\right) + \frac{\bar{V}^2 + \bar{W}^2}{4K_p} \\ &\quad - \frac{\delta_v}{2}\|\tilde{V}\|^2 - \frac{\delta_w}{2}\|\tilde{W}\|^2 + \frac{\delta_v}{2}\bar{W}^2 + \frac{\delta_w}{2}\bar{W}^2 \\ &\le (\bar{\epsilon}_l + \bar{D} + \bar{W})|e| - K_p|e|^2 - e^T K_s e - \frac{\delta_v}{2}\|\tilde{V}\|^2 \\ &\quad - \frac{\delta_w}{2}\|\tilde{W}\|^2 + \frac{\bar{V}^2 + \bar{W}^2}{4K_p} + \frac{\delta_v}{2}\bar{W}^2 + \frac{\delta_w}{2}\bar{W}^2 \end{aligned} \tag{25}$$

Define (α_1, α_2) in the form of

$$\begin{aligned}\alpha_1 &= \bar{\epsilon}_l + \bar{D} + \bar{W} \\ \alpha_2 &= \frac{\bar{V}^2 + \bar{W}^2}{4K_p} + \frac{\delta_v}{2}\bar{W}^2 + \frac{\delta_w}{2}\bar{W}^2 \end{aligned} \tag{26}$$

Based on (25) and (26), it is straightforward to have

$$\begin{aligned}\dot{V}_3 &\le \alpha_1|e| - K_p|e|^2 - e^T K_s e - \frac{\delta_v}{2}\|\tilde{V}\|^2 - \frac{\delta_w}{2}\|\tilde{W}\|^2 + \alpha_2 \\ &\le -e^T K_s e - \frac{\delta_v}{2}\|\tilde{V}\|^2 - \frac{\delta_w}{2}\|\tilde{W}\|^2 + \alpha_2 + \frac{\alpha_1^2}{4K_p} \\ &= -\varrho V_3 + L \end{aligned} \tag{27}$$

in which

$$\begin{aligned}\varrho &= \min\left\{\frac{2\underline{\lambda}(K_s)}{\bar{\lambda}(M)}, \frac{\delta_v}{\bar{\lambda}(\Gamma_v^{-1})}, \frac{\delta_w}{\bar{\lambda}(\Gamma_w^{-1})}\right\} \\ L &= \alpha_2 + \frac{\alpha_1^2}{4K_p} \end{aligned} \tag{28}$$

In (28), $\underline{\lambda}(\cdot)$ and $\bar{\lambda}(\cdot)$ are the minimum and maximum eigenvalues of the matrix. By multiplying the left and right sides of $\dot{V}_3 \le -\varrho V_3 + L$ by $e^{\varrho t}$ and integrating the outcome over $[0, t]$, one can easily obtain the following inequality

$$\begin{aligned}V_3(t) &\le e^{-\varrho t}\left(V_3(0) - \frac{L}{\varrho}\right) + \frac{L}{\varrho} \\ &\le \frac{L}{\varrho} + V_3(0) \end{aligned} \tag{29}$$

According to (23)–(29), it is obvious that

$$\begin{aligned}|e| &\le \sqrt{\frac{N}{\underline{\lambda}(M)}}, \quad |R| \le \max_{[0,t]}(R_r) + \sqrt{\frac{N}{\underline{\lambda}(M)}}, \\ \|\tilde{V}\| &\le \sqrt{\frac{N}{\underline{\lambda}(\Gamma_v^{-1})}}, \quad \|\tilde{W}\| \le \sqrt{\frac{N}{\underline{\lambda}(\Gamma_w^{-1})}}. \end{aligned} \tag{30}$$

where $N = 2(V_3(0) + L/\varrho)$. This means the close-loop variables can be maintained in the compact sets defined as

$$\begin{aligned}\mathbb{O}_e &= \left\{e\colon |e| \le \sqrt{\frac{N}{\underline{\lambda}(M)}}\right\} \\ \mathbb{O}_R &= \left\{R\colon |R| \le \max_{[0,t]}(R_r) + \sqrt{\frac{N}{\underline{\lambda}(M)}}\right\} \\ \mathbb{O}_{\tilde{v}} &= \left\{\tilde{V}\colon \|\tilde{V}\| \le \sqrt{\frac{N}{\underline{\lambda}(\Gamma_v^{-1})}}\right\} \\ \mathbb{O}_{\tilde{w}} &= \left\{\tilde{W}\colon \|\tilde{W}\| \le \sqrt{\frac{N}{\underline{\lambda}(\Gamma_w^{-1})}}\right\} \end{aligned} \tag{31}$$

For Part (2), starting from (29) and (30), we can derive

$$|e| \le \sqrt{\frac{2e^{-\varrho t}\left(V_3(0) - \frac{L}{\varrho}\right) + \frac{2L}{\varrho}}{\underline{\lambda}(M)}} \tag{32}$$

When $V_3(0) = \frac{L}{\varrho}$, we have $\forall t \ge 0$, $|e| \le \sqrt{2L/\varrho\underline{\lambda}(M)}$. Otherwise, for any $e_s > 2L/\varrho\underline{\lambda}(M)$, a time point t_s exists such that $|e| \le e_s$ and $\lim_{t\mapsto\infty}|e| = \sqrt{2L/\varrho\underline{\lambda}(M)}$ for any $t > t_s$. Similar procedures can be re-performed for $\tilde{V}$ and $\tilde{W}$, resulting in compact regions

$$\begin{aligned}\mathbb{O}_e^f &= \left\{e\colon \lim_{t\mapsto\infty}|e| \le \sqrt{\frac{2L}{\varrho\underline{\lambda}(M)}}\right\} \\ \mathbb{O}_{\tilde{v}}^f &= \left\{\tilde{V}\colon \lim_{t\mapsto\infty}\|\tilde{V}\| \le \sqrt{\frac{2L}{\varrho\underline{\lambda}(\Gamma_v^{-1})}}\right\} \\ \mathbb{O}_{\tilde{w}}^f &= \left\{\tilde{W}\colon \lim_{t\mapsto\infty}\|\tilde{W}\| \le \sqrt{\frac{2L}{\varrho\underline{\lambda}(\Gamma_w^{-1})}}\right\} \end{aligned} \tag{33}$$

This implies that $(e, \tilde{V}, \tilde{W})$ can be eventually steered to a small neighbourhood of the origin determined by $(K_p, K_s, \delta_v, \delta_w, \bar{\epsilon}_l)$ and it completes the proof. ■

REFERENCES

[1] X. Meng and C. G. Cassandras, "Eco-driving of autonomous vehicles for nonstop crossing of signalized intersections," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 1, pp. 320–331, Jul. 2020, doi: 10.1109/TASE.2020.3029452.

[2] S. Liu, Z. Hou, T. Tian, Z. Deng, and Z. Li, "A novel dual successive projection-based model-free adaptive control method and application to an autonomous car," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3444–3457, Nov. 2019.

[3] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car—Part I: Distributed system architecture and development process," *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 7131–7140, Dec. 2014.

[4] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car—Part II: A case study on the implementation of an autonomous driving system based on distributed architecture," *IEEE Trans. Ind. Electron.*, vol. 62, no. 8, pp. 5119–5132, Aug. 2015.

[5] I. Draganjac, D. Miklić, Z. Kovačić, G. Vasiljević, and S. Bogdan, "Decentralized control of multi-AGV systems in autonomous warehousing applications," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 4, pp. 1433–1447, Oct. 2016.

[6] C. Jiang *et al.*, "R2-RRT: Reliability-based robust mission planning of off-road autonomous ground vehicle under uncertain terrain environment," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 2, pp. 1030–1046, Apr. 2022, doi: 10.1109/TASE.2021.3050762.

[7] W. Liu, Z. Li, L. Li, and F.-Y. Wang, "Parking like a human: A direct trajectory planning solution," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 12, pp. 3388–3397, Dec. 2017.

[8] S. G. Manyam, K. Sundar, and D. W. Casbeer, "Cooperative routing for an air–ground vehicle team—Exact algorithm, transformation method, and heuristics," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 537–547, Jan. 2020.

[9] L. Chen, Y. Shan, W. Tian, B. Li, and D. Cao, "A fast and efficient double-tree RRT*-like sampling-based planner applying on mobile robotic systems," *IEEE/ASME Trans. Mechtronics*, vol. 23, no. 6, pp. 2568–2578, Dec. 2018.

[10] Y. Li, R. Cui, Z. Li, and D. Xu, "Neural network approximation based near-optimal motion planning with kinodynamic constraints using RRT," *IEEE Trans. Ind. Electron.*, vol. 65, no. 11, pp. 8718–8729, Nov. 2018.

[11] B. Li, K. Wang, and Z. Shao, "Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3263–3274, Nov. 2016.

[12] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, and Y. Xia, "Two-stage trajectory optimization for autonomous ground vehicles parking maneuver," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 3899–3909, Jul. 2019.

[13] C. Sánchez-Sánchez and D. Izzo, "Real-time optimal control via deep neural networks: Study on landing problems," *J. Guid., Control, Dyn.*, vol. 41, no. 5, pp. 1122–1135, Feb. 2018.

[14] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, and C. L. P. Chen, "Six-DOF spacecraft optimal trajectory planning and real-time attitude control: A deep neural network-based approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 5005–5013, Nov. 2020.

[15] K. Shojaei, "A prescribed performance PID control of robotic cars with only posture measurements considering path curvature," *Eur. J. Control*, vol. 65, May 2022, Art. no. 100616.

[16] K. Shojaei, "Neural adaptive PID formation control of car-like mobile robots without velocity measurements," *Adv. Robot.*, vol. 31, no. 18, pp. 947–964, 2017.

[17] J. Zhang *et al.*, "Command-filter-adaptive-based lateral motion control for autonomous vehicle," *Control Eng. Pract.*, vol. 121, Apr. 2022, Art. no. 105044.

[18] S. Geller, I. Avrahami, and N. Shvalb, "Control of autonomous vehicles flow using imposed speed profiles," *J. Intell. Transp. Syst.*, pp. 1–15, Sep. 2021, doi: 10.1080/15472450.2021.1932495.

[19] C.-L. Hwang and W.-L. Fang, "Global fuzzy adaptive hierarchical path tracking control of a mobile robot with experimental validation," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 3, pp. 724–740, Jun. 2016.

[20] C.-L. Hwang, C.-C. Yang, and J. Y. Hung, "Path tracking of an autonomous ground vehicle with different payloads by hierarchical improved fuzzy dynamic sliding-mode control," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 2, pp. 899–914, Apr. 2018.

[21] C. Zhang, H.-K. Lam, J. Qiu, P. Qi, and Q. Chen, "Fuzzy-model-based output feedback steering control in autonomous driving subject to actuator constraints," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 3, pp. 457–470, Mar. 2021.

[22] T. Weiskircher, Q. Wang, and B. Ayalew, "Predictive guidance and control framework for (semi-)autonomous vehicles in public traffic," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 6, pp. 2034–2046, Nov. 2017.

[23] N. Wan, C. Zhang, and A. Vahidi, "Probabilistic anticipation and control in autonomous car following," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 1, pp. 30–38, Jan. 2019.

[24] H. Hu, S. Song, and C. L. P. Chen, "Plume tracing via model-free reinforcement learning method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 8, pp. 2515–2527, Aug. 2019.

[25] J. Zhu, J. Zhu, Z. Wang, S. Guo, and C. Xu, "Hierarchical decision and control for continuous multitarget problem: Policy evaluation with action delay," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 464–473, Feb. 2019.

[26] L. Liu, D. Wang, Z. Peng, C. L. P. Chen, and T. Li, "Bounded neural network control for target tracking of underactuated autonomous surface vehicles in the presence of uncertain target dynamics," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1241–1249, Apr. 2019.

[27] F. J. Lin and P. H. Chou, "Adaptive control of two-axis motion control system using interval type-2 fuzzy neural network," *IEEE Trans. Ind. Electron.*, vol. 56, no. 1, pp. 178–193, Jan. 2009.

[28] S. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 11, pp. 2004–2016, Nov. 2014.

[29] J. Li, Q. Yang, B. Fan, and Y. Sun, "Robust state/output-feedback control of coaxial-rotor MAVs based on adaptive NN approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3547–3557, Dec. 2019.

[30] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, and C. L. P. Chen, "Design and implementation of deep neural network-based control for automatic parking maneuver process," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 4, pp. 1400–1413, Apr. 2022, doi: 10.1109/TNNLS.2020.3042120.

[31] R. Jiang, H. Zhou, H. Wang, and S. S. Ge, "Road-constrained geometric pose estimation for ground vehicles," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 2, pp. 748–760, Apr. 2020.

[32] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 3, pp. 972–983, May 2021.

[33] R. Soloperto, J. Köhler, F. Allgöwer, and M. A. Müller, "Collision avoidance for uncertain nonlinear systems with moving obstacles using robust model predictive control," in *Proc. 18th Eur. Control Conf. (ECC)*, Naples, Italy, Jun. 2019, pp. 811–817.

[34] R. Chai, A. Savvaris, A. Tsourdos, S. Chai, and Y. Xia, "Trajectory optimization of space maneuver vehicle using a hybrid optimal control solver," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 467–480, Feb. 2019.

[35] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, May 2015.

[36] X. Wang, R. Jiang, L. Li, Y. Lin, X. Zheng, and F.-Y. Wang, "Capturing car-following behaviors by deep learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 910–920, Mar. 2018.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[38] H. Guo, D. Cao, H. Chen, C. Lv, H. Wang, and S. Yang, "Vehicle dynamic state estimation: State of the art schemes and perspectives," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 2, pp. 418–431, Mar. 2018.

[39] S. S. Ge and J. Zhang, "Neural-network control of nonaffine nonlinear system with zero dynamics by state and output feedback," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 900–918, Jul. 2003.

Runqi Chai (Member, IEEE) received the B.S. degree in information and computing science from the North China University of Technology, Beijing, China, in 2015, and the Ph.D. degree in aerospace engineering from Cranfield University, Cranfield, U.K., in August 2018. He is currently an Associate Professor with the Beijing Institute of Technology. His research interest includes unmanned vehicle trajectory optimization, guidance, and control.

Derong Liu (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the University of Notre Dame, Notre Dame, IN, USA, in 1994. He was a Staff Fellow with the General Motors Research and Development Center, Warren, MI, USA, from 1993 to 1995. He was an Assistant Professor with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA, from 1995 to 1999. He joined the University of Illinois at Chicago, Chicago, IL, USA, in 1999, where he became a Full Professor of Electrical and Computer Engineering and Computer Science in 2006. He has published 19 books. He is a fellow of the International Neural Network Society and the International Association of Pattern Recognition. He was selected for the "100 Talents Program" by the Chinese Academy of Sciences, Beijing, China, in 2008, where he was the Associate Director of the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, from 2010 to 2015. He received the Faculty Early Career Development Award from the National Science Foundation in 1999, the University Scholar Awards from the University of Illinois from 2006 to 2009, the Overseas Outstanding Young Scholar Award from the National Natural Science Foundation of China in 2008, the Outstanding Achievement Award from the Asia–Pacific Neural Network Assembly in 2014, the INNS Gabor Award in 2018, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS Outstanding Paper Award in 2018, the IEEE SMC Society Andrew P. Sage Best Transactions Paper Award in 2018, and the IEEE/CAA JOURNAL OF AUTOMATICA SINICA Hsue-Shen Tsien Paper Award in 2019. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS from 2010 to 2015. He is also the Editor-in-Chief of *Artificial Intelligence Review* (Springer).

Tianhao Liu received the B.S. degree in automatic control from the Nanjing University of Science and Technology, Nanjing, China, in 2020. He is currently pursuing the M.S. degree in control science and engineering with the Beijing Institute of Technology, Beijing, China.

Antonios Tsourdos (Member, IEEE) received the M.Eng. degree in electronic, control, and systems engineering from the University of Sheffield in 1995 and the M.Sc. degree in systems engineering and the Ph.D. degree in nonlinear robust autopilot design and analysis from Cardiff University in 1996 and 1999, respectively. He joined Cranfield University in 1999 as a Lecturer, the Appointed Head of the Centre of Autonomous and Cyber-Physical Systems in 2007, and a Professor of Autonomous Systems and Control in 2009 and the Director of Research—Aerospace, Transport, and Manufacturing in 2015. He leads the Research Team on Autonomous Systems within the School of Aerospace, Transport and Manufacturing, Cranfield University. He has diverse expertise in both unmanned and autonomous vehicles as well as networked systems. He conducts basic and applied research in the fields of guidance, control, and navigation for single and multiple unmanned autonomous vehicles as well as research on cyber-physical systems.

Yuanqing Xia (Senior Member, IEEE) was born in Anhui, China, in 1971. He received the B.S. degree in fundamental mathematics from the Department of Mathematics, Chuzhou University, Chuzhou, China, in 1991, the M.S. degree in fundamental mathematics from Anhui University, Wuhu, China, in 1998, and the Ph.D. degree in control theory and control engineering from the Beijing University of Aeronautics and Astronautics, Beijing, China, in 2001. He has published eight monographs with Springer and Wiley, and more than 300 articles in journals. His current research interests are in the fields of networked control systems, robust control and signal processing, and active disturbance rejection control and flight control. He has obtained the Second Award of the Beijing Municipal Science and Technology (No. 1) in 2010, the Second National Award for Science and Technology (No. 2) in 2011, and the Second Natural Science Award of The Ministry of Education (No. 1) in 2012. He is the Deputy Editor of the *Journal of Beijing Institute of Technology*; and an Associate Editor of *Acta Automatica Sinica*, *Control Theory and Applications*, the *International Journal of Innovative Computing, Information and Control*, and the *International Journal of Automation and Computing*.

Senchun Chai (Senior Member, IEEE) received the B.S. and master's degrees in control science and engineering from the Beijing Institute of Technology, Beijing, China, in 2001 and 2004, respectively, and the Ph.D. degree in networked control systems from the University of South Wales, Pontypridd, U.K., in 2007. He was a Research Fellow with Cranfield University, U.K., from 2009 to 2010; and a Visiting Scholar with the University of Illinois at Urbana–Champaign, Urbana, USA, from January 2010 to May 2010. He is currently a Professor with the School of Automation, Beijing Institute of Technology. His current research interests focus on flight control systems, networked control systems, embedded systems, and multi-agent control systems.