IEEE Transactions on Neural Networks and Learning Systems, Volume 33, Issue 4, April 2022, pp. 1400-1413
DOI: 10.1109/TNNLS.2020.3042120

Design and Implementation of Deep Neural Network-Based Control for Automatic Parking Maneuver Process

Runqi Chai, *Member, IEEE,* Antonios Tsourdos, *Member, IEEE,* Al Savvaris, Senchun Chai, *Senior Member, IEEE,* Yuanqing Xia, *Senior Member, IEEE,* and C. L. Philip Chen, *Fellow, IEEE,*

***Abstract*—This paper focuses on the design, test and validation of a deep neural network (DNN)-based control scheme capable of predicting optimal motion commands for autonomous ground vehicles (AGVs) during the parking maneuver process. The proposed design utilizes a multi-layer structure. In the first layer, a desensitized trajectory optimization method is iteratively performed to establish a set of time-optimal parking trajectories with the consideration of noise-perturbed initial configurations. Subsequently, by using the pre-planned optimal parking trajectory dataset, several DNNs are trained in order to learn the functional relationship between the system state-control actions in the second layer. To obtain further improvements regarding the DNN performances, a simple yet effective data aggregation approach is designed and applied. These trained DNNs are then utilized as the motion controllers to generate the feedback actions in real-time. Numerical results were executed to demonstrate the effectiveness and the real-time applicability of using the proposed control scheme to plan and steer the AGV parking maneuver. Experimental results were also provided to justify the algorithm performance in real-world implementations.**

***Index Terms*—Deep neural network, autonomous ground vehicles, trajectory optimization, motion controller, parking maneuver.**

I. Introduction

ADVANCED driver assistance systems currently play a key component of intelligent vehicle technology. In order to achieve a higher level of automation and to satisfy different autonomous driving demands, a large amount of efforts have been paid by both the academic and industrial communities on researching this topic during the last decade [1–3]. Among these developments, a particular focus is the design and control of automatic parking systems. Automatic parking usually refers to placing a ground vehicle in a pre-determined parking area and it is a typical use case in the autonomous driving mode [4, 5]. Due to the existence of uncertainty in the traffic environment, it is still challenging to design a reliable real-time parking control system which can ensure the passenger's comfort whilst completing the maneuver in a safe and quick way.

The entire automatic parking task is often fulfilled by performing two steps: trajectory planning and motion control [6–8]. Generally speaking, in the first step, the system aims to generate a feasible parking path connecting the vehicle's current position and the desired position in the parking slot without colliding with obstacles (e.g. the edge of the road, parking area boundaries, or objects). It should be noted that this step is fundamental yet important, as it is likely to obtain enhanced motion control performance if a well-designed parking trajectory can be recorded. Contributions on researching trajectory planning methods for autonomous vehicles (AVs) can be found in the literature. For example, a number of well-developed geometric path planners have been proposed [6, 9]. The core idea of these approaches is to shape and smooth the vehicle maneuver profile using spline or polynomial techniques. Although geometric-based approaches are computationally friendly, they have a number of limitations. Commonly, a geometric path planner may produce a solution on the basis of specific structures. When complicated traffic environments are required to consider, such a planner may fail to satisfy vehicle dynamic constraints under a certain scenario. Also, it is likely to result in a relatively-large difference between the actual maneuver trajectories and the designed curves.

As a potential alternative, researchers have recently investigated the possibility of applying optimization or artificial intelligence-based methods to plan the motion of AVs [10–13]. For example, the study carried out by Choi and Huhtala [14] investigated the feasibility of using gradient-based optimization to search the global path of the AV. Their work showed that, given a proper set of algorithm parameters, it is possible to calculate the shortest path using the proposed approach. In addition, in [5] a swarm intelligent optimization algorithm was constructed to plan the parking trajectory of the wheeled vehicle.

Once a pre-planned trajectory is generated, the motion control system should be designed so as to steer the vehicle to fulfill the entire maneuver. A commonly-used strategy is the reference-tracking-based control [15, 16]. This type of method is motivated by modern control theories and it has been broadly utilized in the literature [17–21]. For instance, Xu et

R. Chai, A. Tsourdos and A. Savvaris are with the School of Aerospace, Transport and Manufacturing, Cranfield University, UK, e-mail: (r.chai@cranfield.ac.uk or r.chai@ieee.org), (a.tsourdos@cranfield.ac.uk), and (a.savvaris@cranfield.ac.uk).

S. Chai and Y. Xia are with the school of Automation, Beijing Institute of Technology, Beijing, China, e-mail: (chaisc97@163.com), and (xia_yuanqing@bit.edu.cn).

C. L. P. Chen is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China; with the Navigation College, Dalian Maritime University, Dalian 116026, China; and also with the Faculty of Science and Technology, University of Macau, Macau 999078, China, e-mail: (philip.chen@ieee.org).

© 2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

al. [19] derived a neural network-based (NN) dynamic control law in order to steer and track the motion of an autonomous flight vehicle, whereas in [20], an NN-assisted adaptive back-stepping law was derived and successfully applied to address an AV attitude tracking problem. In addition, an NN-based mechanism, coupled with a robust adaptive strategy, was advocated in [21] in order to control the AV to track the desired trajectory. From the reported verification results, these methods can work properly for the considered cases. However, for many complex nonlinear or hybrid systems, an explicit control policy which can guarantee system stability under dynamic constraints and various model uncertainties might not be easily derived [22]. To circumvent the problem, multiple model-related assumptions are usually provided. This will inevitably introduce conservatism in the design, thereby limiting the practical application of the proposed control law.

In recent years, there is a growing trend in terms of extending the optimization-based motion planners to real-time motion control applications. This is usually achieved by constructing a receding horizon replanning procedure and the control command is re-optimized at each sampling instant. In the literature, a number of works on this topic can be found [23–25]. The authors in [24] designed a nonlinear model predictive control (NMPC) scheme to produce the guidance commands and control profiles for AVs under a public traffic environment. Similarly, an NMPC-driven scheme was developed in [25] to steer the motion of AVs with the consideration of moving obstacles. One important feature is that a B-spline parametrization method is used to decrease the scale of the nonlinear optimization process. Though the results reported in the aforementioned works confirmed the effectiveness of using these optimization-based motion control algorithms, they are less likely to be applied in real systems due to the high computational burden and insufficient robustness of the online optimization process.

To ease the computational burden while producing optimal feedback actions in real-time, in this paper, we are interested in designing, testing, and validating a deep neural network-based (DNN) control scheme for the automatic parking problem. The motivation of applying DNN mainly relies on its ability in representing complex functional relationship, thereby making it a useful tool for approximation of optimal state-control actions. Early studies suggested that DNN-based direct representations have the potential to be applied in autonomous vehicle trajectory planning/control problems, and a number of works have been reported in DNN for autonomous driving [26–28]. For example, Grigorescu et al. [26] provided a comprehensive review regarding artificial intelligence (AI) techniques applied in autonomous driving. In this paper, the authors systematically outlined the self-driving architecture including key modules constructed via deep learning algorithms. In [27], the authors introduced a parallel learning concept in order to form AI-based systems for engineering problems. Later in [28], a first attempt was made to combine transfer leaning and parallel learning such that the network can learn the parking skill more effectively. Motivated by these impressive achievements, a bi-level integrated framework is proposed in the present work. Employing such a multi-layer design, we are able to merge advanced trajectory optimization and deep learning algorithms together, thus preserving the structure of the optimal solution and reducing the online computing demand.

In the upper level of the proposed control scheme, a set of optimal parking trajectories with time duration minimization is generated by sequentially utilizing a newly-developed trajectory optimization approach. Following that, the obtained parking trajectory dataset is provided to the lower level, where deep neural networks are constructed to learn the structure of the optimal state-control relations. To further strengthen the network approximation performance, a data aggregation strategy has been developed. In this way, the trained DNNs can gain an understanding of the parking trajectory-steering mapping relationship such that they can be applied for controlling the vehicle to achieve the optimal trajectory online. It is noteworthy that compared with online replanning-based approaches (e.g., dynamic programming (DP) or model predictive control (MPC)-based methods), the proposed DNN-driven method is likely to save considerable computational time and resources.

The remainder of this work is structured as follows. In Sec II, the trajectory optimization model of the parking maneuver problem is established. Following that, the process of generating the optimal parking trajectory ensemble, along with the establishment of the DNN-based control scheme, is outlined in Sec III. Furthermore, in Sec IV the DNN-driven motion control performance is studied in detail by performing a number of experiments. Key findings as well as highlights of obtained results are then concluded in Sec V.

II. Optimization Model of the Parking Maneuver Process

A. System Model of the Vehicle

The trajectory optimization model of the parking maneuver problem is introduced in this section. The general parking scenarios (e.g., the vertical parking and parallel parking) can be visualized in Fig.1.

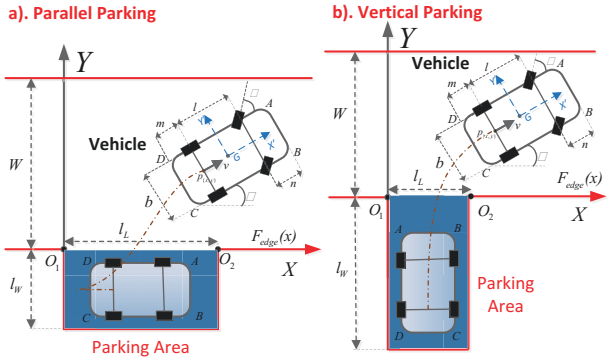


Fig. 1: General parking scenarios

To describe the movement of the wheeled vehicle, a set of differential equations are established, which can be abbreviated to the following affine nonlinear system [7]:

$$\dot{x} = f(x(t)) + Bu(t) \tag{1}$$

In Eq.(1), $t \in [t_0, t_f]$ is the time variable. $x(t) = [p_x(t), p_y(t), v(t), a(t), \theta(t), \phi(t)]^T$ is the system state variable consisting of the center location of the rear wheel $(p_x(t), p_y(t))$, the velocity and acceleration $(v(t), a(t))$, the oriental angle $\theta(t)$, and the steering angle $\phi(t)$, respectively. $u(t) = [\eta(t), \omega(t)]^T$ represents the control input, where η and ω are the jerk and angular velocity, respectively. The nonlinear function $f(\cdot)$ and B are given by:

$$f(x(t)) = \begin{bmatrix} v(t)\cos(\theta(t)) \\ v(t)\sin(\theta(t)) \\ a(t) \\ 0 \\ v(t)\frac{\tan(\phi(t))}{l} \\ 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{2}$$

where l denotes the length between the two wheels.

B. Parking Maneuver Constraints

Multiple process or boundary constraints are required to be taken into account during the parking maneuver. For instance, the starting and ending conditions of the vehicle should be specified. This can be described as:

$$\begin{array}{llll} p_x(t_0) = p_{x_0} & p_y(t_0) = p_{y_0} & v(t_0) = 0 & a(t_0) = 0 \\ \theta(t_0) = 0 & \phi(t_0) = 0 & p_x(t_f) = p_{x_f} & p_y(t_f) = p_{y_f} \\ v(t_f) = 0 & a(t_f) = 0 & \theta(t_f) = \theta_f & \phi(t_f) = 0 \\ A_y(t_f) \leq 0 & B_y(t_f) \leq 0 & C_y(t_f) \leq 0 & D_y(t_f) \leq 0 \end{array} \tag{3}$$

It is noteworthy that the target final conditions may vary from scenario to scenario. For example, the terminal oriental angle value for the vertical parking mission should be $\theta(t_f) = 90°$, while for the parallel parking mission, the target $\theta(t_f)$ becomes $0°$. In Eq.(3), (A_y, B_y, C_y, D_y) can be calculated via:

$$\begin{cases} A_y(t) = \sin(\theta(t))(l+n) + 0.5b\cos(\theta(t)) + p_y(t) \\ B_y(t) = \sin(\theta(t))(l+n) - 0.5b\cos(\theta(t)) + p_y(t) \\ C_y(t) = p_y(t) - 0.5b\cos(\theta(t)) - m\sin(\theta(t)) \\ D_y(t) = p_y(t) + 0.5b\cos(\theta(t)) - m\sin(\theta(t))) \end{cases} \tag{4}$$

Here, m and n are, respectively, the rear and front overhangs. b is the width of the vehicle. Apart from the boundary constraints, each system state or control variable should satisfy the following path constraint:

$$\begin{array}{ll} |p_x(t)| \leq p_x^{max} & |a(t)| \leq a^{max} \\ |p_y(t)| \leq p_y^{max} & |\phi(t)| \leq \phi^{max} \\ |v(t)| \leq v^{max} & |\eta(t)| \leq \eta^{max} \\ |\theta(t)| \leq \theta^{max} & |\omega(t)| \leq \omega^{max} \end{array} \tag{5}$$

As shown in Fig.1, the edge of the parking slot is modeled as $F_{edge}(x) = -(H(x) + H(x - l_L))l_W$, where $H(x)$ is the Heaviside step function. During the parking maneuver, we need to guarantee that the vehicle will not collide with both the edge of the parking area and the edge of the road. To achieve this goal, the following path constraint should be imposed:

$$\begin{array}{l} F_{edge}(A_x) \leq A_y \leq W \\ F_{edge}(B_x) \leq B_y \leq W \\ F_{edge}(C_x) \leq C_y \leq W \\ F_{edge}(D_x) \leq D_y \leq W \end{array} \tag{6}$$

In Eq.(6), the width of the road is denoted as W. (A_x, B_x, C_x, D_x) can be calculated via:

$$\begin{cases} A_x(t) = \cos(\theta(t))(l+n) - 0.5b\sin(\theta(t)) + p_x(t) \\ B_x(t) = 0.5b\sin(\theta(t)) + \cos(\theta(t))(l+n) + p_x(t) \\ C_x(t) = p_x(t) + 0.5b\sin(\theta(t)) - m\cos(\theta(t)) \\ D_x(t) = p_x(t) - 0.5b\sin(\theta(t)) - m\cos(\theta(t)) \end{cases} \tag{7}$$

Besides, a collision-free constraint should be constructed such that the vehicle will not collide with two corner points of the parking slot during the entire maneuver (e.g., O_1 and O_2 shown in Fig.1). This is achieved via the following inequalities:

$$\begin{array}{l} \mathbb{A}_{AO_1B} + \mathbb{A}_{BO_1C} + \mathbb{A}_{CO_1D} + \mathbb{A}_{DO_1A} > \mathbb{A}_{ABCD} \\ \mathbb{A}_{AO_2B} + \mathbb{A}_{BO_2C} + \mathbb{A}_{CO_2D} + \mathbb{A}_{DO_2A} > \mathbb{A}_{ABCD} \end{array} \tag{8}$$

where $\mathbb{A}_\cdot$ stands for the area. $\mathbb{A}_{ABCD}$ is the region occupied by the wheeled vehicle. Eq.(8) indicates the two corner points of the parking slot can always locate outside the region occupied by the vehicle.

C. Mission Objective

In this paper, we are interested in steering the autonomous vehicle to the specified parking area in the shortest time possible. Therefore, the objective function of the considered problem can be written as:

$$J = \min \int_{t_0}^{t_f} 1 \mathrm{dt} \quad \text{or} \quad J = \min \int_{s_0}^{s_f} \frac{1}{\dot{s}(\tau)} \mathrm{d}\tau \tag{9}$$

where t_0 and t_f denotes the initial and terminal time instants. $\dot{s}(\tau)$ is the velocity along the path s. Both of these two formulation are widely applied in the literature to reflect the traveling time [4, 29]. In the former one, $t \in [t_0, t_f]$ is considered as an independent variable. In the later one, $\dot{s}$ can be formulated as a function of velocity, curvature radius or other geometrical considerations. In this paper, we use to former one to construct our trajectory optimization model.

The entire optimization model can be understood as follows: we aim to determine the vector of optimal states $x^*(t)$ and controls $u^*(t)$ that minimize the total time duration (9) when the autonomous vehicle starts its maneuver from an initial configuration at time t_0, and moves to a final configuration at t_f, while being subjected to the process constraints introduced in Sec II.B. Note that the nonlinear vehicle equations of motion (1) are considered as dynamic constraints and adhered in the optimization model.

III. Maneuver Planning and Control via a DNN-based Approach

Indeed, the parking maneuver optimization problem constructed in the previous section yields an open-loop control solution which is not implementable for real AGVs. This stimulates a technology development program that calls for high precise control systems capable of delivering optimal feedback actions in real-time. To achieve this goal, significant amount of efforts have been devoted by researchers to investigate and improve the possibility of using MPC-based approaches [24, 25]. That is, the re-optimization process is performed at the current time instant and only a certain segment of the resulting optimal control solution is acting on

the system. Although some related work demonstrated that the MPC-based methods have the potential to steer AGVs in an optimal way, a main drawback is that the time-consuming re-optimization process can hardly be afforded in real-world applications (e.g., it may cause large time delays in receiving the control command). Consequently, the main objective of this section is to design a computationally-friendly control framework for the AGV parking system. Motivated by the previous work [23, 28], a bi-level integrated framework is proposed and visualized in Fig. 2.

A. Dataset of Optimal Parking Maneuver Profiles

As shown in Fig. 2, the upper level of the designed framework is mainly responsible for producing a number of optimal parking maneuver profiles. This is achieved by solving the optimal maneuver planning problem defined in the previous section with noise-perturbed initial configurations. Due to the existence of system nonlinearity and path constraints, deriving the analytical form of the optimal solution becomes much more difficult. In this case, numerical methods become an effective tool to tackle the problem. In this paper, we resort to the method of pseudospectral collocation which is commonly applied in wheeled vehicle control applications [30] for transcribing the continuous-time parking maneuver planning problem into a nonlinear static version. In this approach, the system states and controls are discretized on a set of temporal nodes $\{t_k\}_{k=1}^{N_k}$ (denoted as $x(t_k) = x_k$, and $u(t_k) = u_k$). Then the mission objective, system equations and dynamic constraints become either a function of the parameterized state and control or a value at the collocation node. In this way, the original optimal parking maneuver planning problem is transformed into a static constrained version which is solvable using standard optimization algorithms. More specifically, the aim of the optimization process becomes finding (x_k, u_k) at every temporal node $t_k, k = 1, ..., N_k$ such that the discretized performance index can be optimized and the discretized constraints are satisfied.

Priory to trigger the solution-finding iteration, two additional steps are performed to further improve the robustness of the numerical optimization process.

1) Normalization: A normalization process should be performed on each decision variable, which can be written as:

$$\begin{array}{ll} \bar{p_x} = p_x/p_x^{max} & \bar{p_y} = p_y/p_y^{max} \\ \bar{v} = v/v^{max} & \bar{a} = a/a^{max} \\ \bar{\theta} = \theta/\pi & \bar{\phi} = \phi/\pi \\ \bar{\eta} = \eta/\eta^{max} & \bar{\omega} = \omega/\omega^{max} \end{array} \tag{10}$$

It is noteworthy that this step is simple yet non-neglectable. It was shown in [5, 13] that poor scaling can damage the algorithm robustness and it may lower the convergence speed. Therefore, we use Eq.(10) (e.g., the nondimensionalization equation) to regulate the AGV system variables and desensitize these potentially negative effects.

2) Initial Parking Maneuver Profile: The consideration of perturbed configurations may also damage the convergence rate of the optimization process. To deal with this issue, a desensitized trajectory optimization algorithm specifically designed for AGV parking problems [5] is applied. The idea of this algorithm is to pre-generate a feasible parking maneuver trajectory between the noise-perturbed initial point and the target point via a swarm-intelligent optimization approach. Subsequently, this trajectory is applied as the initial guess value to warmly trigger the main solution-finding loop. It was analyzed in [5] that using the pre-generated guess trajectory, the convergence speed and the successful rate for finding the optimal parking maneuver profile can be significantly improved. As a result, we choose this method as the maneuver planner in this work.

Suppose that the initial configuration of the vehicle is perturbed by some uncertain noises. That is, the initial state value becomes

$$x(t_0) = x_0 + \zeta_x^i, \quad i = 1, ..., N_i \tag{11}$$

In Eq.(11), the noise ζ_x is a random variable taken from a normal distribution. That is, $\{\zeta_x^i\}_{i=1}^{N_i} \in N(0, G^2)$, where G^2 represents the variance and N_i is the sample size. A Monte-Carlo technique is then used to generate trajectories in order to form the optimal parking maneuver dataset D_{Tr}. These steps will be carried out offline and they are summarised in Algorithm 1. The algorithm will be terminated until the size of the dataset reaches a certain value specified by the designer.

Algorithm 1 Construction of the optimal parking maneuver dataset

```
Generate the temporal points {t_k}_{k=1}^{N_k};
for i := 1, 2, ... do
    Generate the noise value ζ_x^i randomly;
    Construct the noise-perturbed model based on (11) and (8);
    Discretize the noise-perturbed optimization model at
    (x_k, u_k);
    Initialize the initial parking maneuver generator [5];
    Pre-solve the problem using the initial parking maneuver
    generator;
    Assign the output of the generator as the starting point;
    Resolve the problem using gradient-based approach;
    if the algorithm can successfully converge (tolerance ϵ) then
        Collect the optimal result (x_k^*, u_k^*) to D_Tr;
        Set i = i + 1;
    else
        Discard the current solution and set i = i + 1;
    end if
end for
Output the final dataset D_Tr
```

B. Learning the Optimal Structure of State-Control Actions

Following the discussion provided in the previous subsection, it is assumed that an optimal parking maneuver dataset has already been produced. As mentioned earlier, open loop solution can make limited contribution in real world applications. Hence, one goal of the proposed framework, especially in the lower level, is to construct an optimal feedback motion controller which can be applied in real-time. To do this, the idea is to train DNNs $\mathscr{N}$ on the produced dataset such that the network can explore and estimate the relationship between the optimal state-control pairs. That is,

$$\begin{array}{l} u_1(t_k) \approx \mathscr{N}_1(x(t_k)) \\ u_2(t_k) \approx \mathscr{N}_2(x(t_k)) \end{array} \tag{12}$$

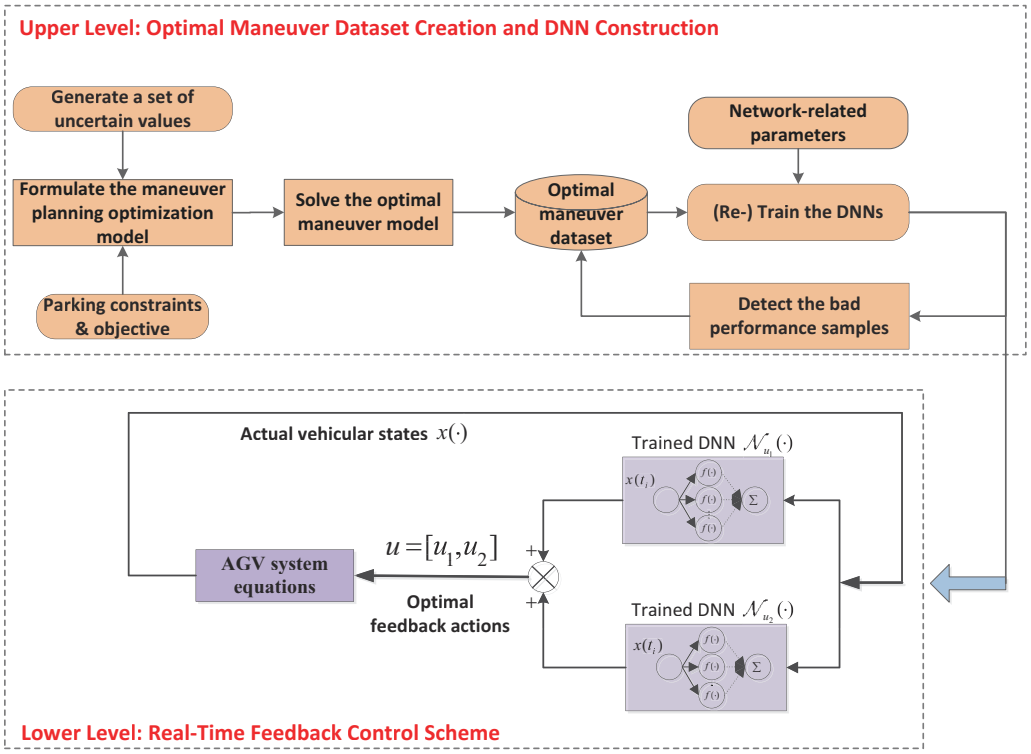


Fig. 2: Bi-level integrated framework

Here, $\mathcal{N}_1$ can be understood as the acceleration control subsystem. Analogically, $\mathcal{N}_2$ denotes the steering control subsystem.

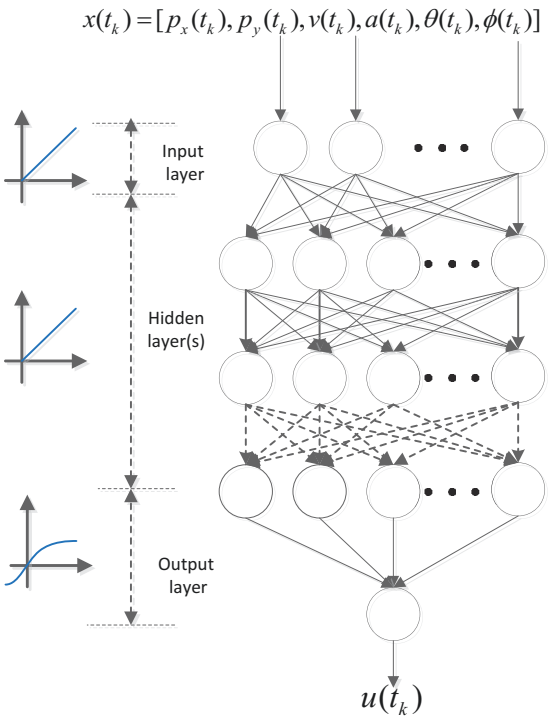


Fig. 3: An illustration of DNN

The considered DNNs (e.g., $\mathcal{N}_1$ and $\mathcal{N}_2$) use a feed-forward neural network structure including an input layer, an output layer and multiple hidden layers. An illustration of the constructed DNN is given by Fig. 3, from where it can be observed that multiple node points (neurons) may exist in each layer and once the numbers of layer and neuron (e.g., N_l and N_s) are specified, the structure of the DNN is determined. The output of the jth neuron at ith layer can be written as

$$S_{i,j} = f_i(\omega_{i,j} S_{i-1} + b_{i,j}) \tag{13}$$

Here the term S_{i-1} is defined as

$$S_{i-1} = \sum_{i=1}^{N_s^{i-1}} f_{i-1}(\omega_{i-1,j} S_{i-1,j} + b_{i-1,j}) \tag{14}$$

In Eq.(13) and Eq.(14), the activation function f used in the hidden layer is the rectified linear function $f(x) = \max(0, x)$, while in the output layer, the $\tanh$ function is used as the mapping function. The weight and bias variables $[\omega, b]$ will be adjusted during the network training process such that the DNN can optimally approximate the functional relationship between the input and the target output values. In other words, the training process aims to update $[\omega, b]$ so that a loss function in the form of (15) can be minimized.

$$L = \frac{1}{N} \sum_{i=1}^{N} (\widehat{o}(x_i) - o(x_i))^2 \tag{15}$$

In Eq.(15), the variables N, $\widehat{o}$ and o stand for, respectively, the size of data, the network output value, and the target output value. Similarly, the network approximation performance can be assessed by introducing the error and the mean squared

error (MSE) functions:

$$e_i = |x_i) - o(x_i)|/o(x_i$$
$$\text{MSE} = \sqrt{\frac{1}{N}\sum_i^N e_i^2} \tag{16}$$

It is worth noting that gradient descent (GD) method is commonly-used in training the neural network. Specifically,

$$\boldsymbol{\omega} = \boldsymbol{\omega} + \Delta\boldsymbol{\omega} \quad \boldsymbol{b} = \boldsymbol{b} + \Delta\boldsymbol{b} \tag{17}$$

where the increment vectors $\Delta\boldsymbol{\omega}$ and $\Delta\boldsymbol{b}$ are written as:

$$\Delta\boldsymbol{\omega} = \xi\frac{\partial L}{\partial\boldsymbol{\omega}} \quad \Delta\boldsymbol{b} = \xi\frac{\partial L}{\partial\boldsymbol{b}} \tag{18}$$

where ξ represents the learning rate value. In recent years, various modified GD approaches have also been proposed in the literature to obtain enhanced training performance. In this paper, a state-of-the-art stochastic gradient descent method (SGD) proposed in [31] is selected to train the weight and bias variables.

C. Performance Enhancement

One important conclusion obtained from [23] is that the learning rate value tends to be sensitive with respect to the stability and accuracy of approximating the optimal state-control relation. More specifically, a relatively small ξ value might result in insufficient approximation accuracy. However, a large ξ value tends to result in insufficient robustness in terms of the convergence history. To enhance the network performance, an effective strategy is to start the training process with a large learning rate value and gradually decrease ξ as the number of training iteration increases. This is achieved by introducing a natural exponential decay equation:

$$\bar{\xi} = \xi e^{-\lambda\tau} \quad \tau = \frac{s_g}{s_d} \tag{19}$$

in which s_g stands for the current iteration step while s_d is the decay step. $\lambda \in [0.8, 1]$ denotes the decay rate.

In order to further improve the network approximation ability, another simple yet effective approach is designed and investigated in this paper. The proposed strategy can be treated as an imitating learning process, where the DNN aims to imitate an action specified by the experts so as to improve its performance. Specifically, for the considered problem, we apply a data aggregation (DA) technique. The motivation for the usage of DA mainly relies on its simplicity and capability of recovering from past mistakes.

To perform the proposed strategy, the constructed DNNs are firstly trained on the optimal maneuver dataset D_{Tr}. Once the training has been completed, we test the network prediction ability on the test dataset D_{Te} including N_{Te} testing trajectories. Following that, according to the terminal state conditions (3) and the MSE function (16), the parking maneuver index (among the test dataset) which cannot be accurately predicted is detected and the corresponding parking trajectory will be collected to a bad performance dataset D_b. Subsequently, we adhere the detected bad performance dataset into the training dataset and re-train the DNN, thereby allowing more emphases on this particular group. The general procedures of constructing the bad performance dataset for the automatic parking problem are summarised and presented in Algorithm 2.

Algorithm 2 Bad performance dataset construction

procedure
  Step 1: Use the pre-generated optimal maneuver dataset D_{Tr} to train DNNs;
  Step 2: Apply the trained DNNs on the test dataset D_{Te};
  Step 3: Assign the acceptable threshold with respect to the MSE and the terminal state errors;
  for $q := 1, 2, ..., N_{Te}$ **do**
    (a). If the qth trajectory among D_{Te} cannot satisfy the acceptable threshold;
    (b). Add the qth maneuver trajectory to D_b;
    (c). Set $q = q + 1$;
  end for
  Step 4: Adhere D_b into D_{Tr} via:
    $D_{Tr} = D_{Tr} \cup D_b$
  Step 5: Re-train the DNNs on D_{Tr} for further improvement;
end procedure

D. Overall Algorithm Framework

The overall framework of the proposed DNN-driven parking maneuver controller is structured in Algorithm 3.

Algorithm 3 Procedures of the DNN-driven parking maneuver

Offline: Construct the parking maneuver optimization model via Eq.(8)
Generate the optimal maneuver dataset via Algorithm 1;
Assign the network structural parameters to establish the DNNs;
Train the DNNs on the pre-generated optimal trajectory dataset;
Construct the bad performance dataset via Algorithm 2;
Re-train the DNNs on the detected bad performance dataset;
/*Main Loop*/
Online: At each time point $k := 0, 1, ...$;
  (i). Obtain the current vehicle state x_k;
  (ii). Predict the optimal control commands using the trained DNNs $u_k := \mathscr{N}(x_k)$;
  (iii). Implement u_k to the vehicle motion system until the next sampling instant;
  (iv). Set $k = k + 1$;
  (v). Compute the state variables at the next time point by numerically integrating $\dot{x} = f(x, \mathscr{N}(x))$;
  (vi). Repeat the steps i)-v) for the next sampling time instant;

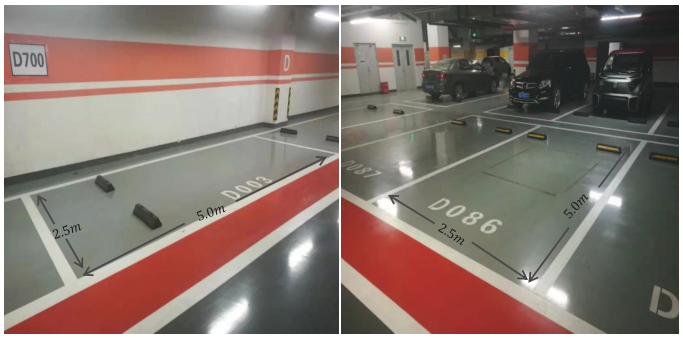
IV. Results and Discussions

In this section, we focus on the test and validation of the proposed DNN-based strategy. Specifically, detailed numerical and physical studies were executed and the obtained results are presented to show the availability as well as the real-time applicability of applying the proposed control scheme to plan and steer the parking maneuver of the AGV.

In the following tests, two real-world parking scenarios are mainly considered:

- Scenario 1: Vertical parking as visualized by Fig. 1(a) and Fig. 4(a)
- Scenario 2: Parallel parking as visualized by Fig. 1(b) and Fig. 4(b)

Some vehicle-related parameters existing in the optimization model (8), along with the algorithm-related parameters are provided in Table I. On the other hand, the initial/terminal state values of the AGV, together with some scenario-related parameters are assigned in Table II.



a) Scenario 1: Parallel parking b) Scenario 2: Vertical parking

Fig. 4: Two Parking Scenarios

TABLE I: Vehicle/algorithm-related parameters

Parameters	Values	Parameters	Values	Parameters	Values
m	0.4	v^{max}	2	N_k	60
l	1.45	θ^{max}	90	N_i	50000
n	0.55	a^{max}	0.5	N_l	7
b	1.54	ϕ^{max}	30	N_s	128
W	4	η^{max}	0.5	λ	0.9
p_x^{max}	15	ω^{max}	35	ϵ	1×10^{-8}
p_y^{max}	5	t_f^{max}	50	ξ	$0.1 - 0.0001$

TABLE II: Scenario-related parameters

Scenario 1		Scenario 2	
$p_x(t_0) = 10.4$	$p_x(t_f) = 1.745$	$p_x(t_0) = -4.5$	$p_x(t_f) = 1.25$
$p_y(t_0) = 1.615$	$p_y(t_f) = -1.25$	$p_y(t_0) = 1.615$	$p_y(t_f) = -3.255$
$v(t_0) = 0$	$v(t_f) = 0$	$v(t_0) = 0$	$v(t_f) = 0$
$a(t_0) = 0$	$a(t_f) = 0$	$a(t_0) = 0$	$a(t_f) = 0$
$\theta(t_0) = 0$	$\theta(t_f) = 0$	$\theta(t_0) = 0$	$\theta(t_f) = 90$
$\phi(t_0) = 0$	$\phi(t_f) = 0$	$\phi(t_0) = 0$	$\phi(t_f) = 0$
$l_L = 2.5$	$l_W = 5$	$l_L = 5$	$l_W = 2.5$

To establish the DNN-based controller, we need to produce a number of optimal parking maneuver profiles for the two considered parking scenarios. As illustrated by Algorithm 1, for each $i \in \{1, 2, ..., N_i\}$, the desensitized maneuver planner proposed in [5] is iteratively applied to calculate the optimal parking maneuver profiles from the noise-perturbed initial configuration to the target final point. It is assumed that ζ_{p_x}, ζ_{p_y} and ζ_θ are normally distributed on $[-2, 2]m$, $[-0.5, 0.5]m$ and $[-5, 5]^\circ$, respectively.

A. Determination of Network-Related Parameters

In this subsection, we determine the combination of network-related parameters such that the trained DNN can acquire enhanced approximation performance. It should be noted that three important factors, including the data used for training the network N_d, the depth of the network N_l and the number of units N_o, may have impacts on the DNN approximation performance. To explore the optimal combination, we divide these factors into five levels and design the orthogonal experiments (as shown in Table IV). The detailed factor specifications can be found in Table III. Note that 80% of the data is utilized for training the DNN, while 10% of the data is applied for testing the network. The rest 10% forms the validation dataset, which is primarily used to provide a feedback while the training process is performing. In this way, the training can be terminated if the error value on the validation dataset increases, as this can be viewed as a sign of over-fitting. The MSE value on the test set is used as the main indicator to reflect the final DNN performance under a certain parameter combination. This result has been reported in the last column of Table IV.

TABLE III: Factor specification

Factor	Level				
	Level A	Level B	Level C	Level D	Level E
Data Size N_d	2×10^5	3×10^5	4×10^5	5×10^5	6×10^5
Layer N_l	3	4	5	6	7
Unit N_o	8	16	32	64	128

TABLE IV: Results: different combinations

Experiment	Level			Result
	N_d	N_L	N_o	MSE value
Test case 1	A	A	A	6.46E-04
Test case 2	A	B	B	4.92E-04
Test case 3	A	C	C	3.43E-04
Test case 4	A	D	D	2.12E-04
Test case 5	A	E	E	8.01E-05
Test case 6	B	A	B	5.91E-04
Test case 7	B	B	C	4.44E-04
Test case 8	B	C	D	2.39E-04
Test case 9	B	D	E	9.57E-05
Test case 10	B	E	A	6.84E-05
Test case 11	C	A	C	5.42E-04
Test case 12	C	B	D	4.04E-04
Test case 13	C	C	E	2.25E-04
Test case 14	C	D	A	2.14E-04
Test case 15	C	E	B	6.00E-04
Test case 16	D	A	D	5.12E-04
Test case 17	D	B	E	3.61E-04
Test case 18	D	C	A	3.13E-04
Test case 19	D	D	B	9.77E-05
Test case 20	D	E	C	5.87E-05
Test case 21	E	A	E	5.10E-04
Test case 22	E	B	A	4.41E-04
Test case 23	E	C	B	2.78E-04
Test case 24	E	D	C	1.09E-04
Test case 25	E	E	D	4.64E-05

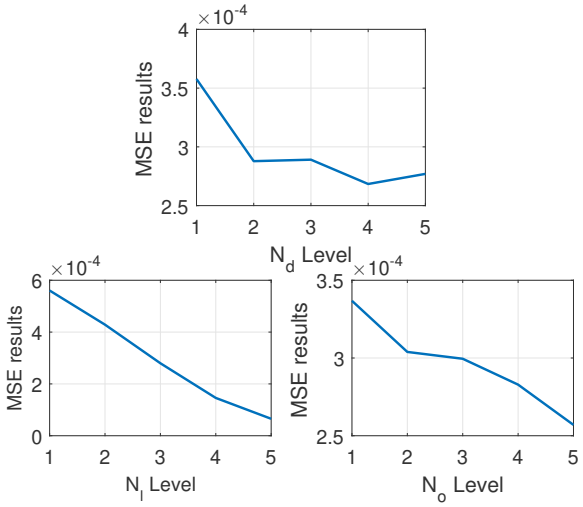


Fig. 5: Level trend results

After carrying out all the test cases, the level trends for different factors can be visualized in Fig. 5. A jump in the level trend curve may be viewed as a sign of over-training. From this figure, we can see that the optimal network approximation ability can be obtained if the N_l and N_o factors can be assigned to level E, while the N_d factor can be set to Level

D (e.g., $N_l = 7$, $N_o = 128$ and $N_d = 5 \times 10^5$) for the considered problem. Interestingly, from the data size level trend profile, the DNN performance is not always improved as the dataset becomes larger. Also, it can be observed from Fig.5 that compared with the impact caused by the data size and the number of units, the depth of the network tends to have more significant influences with respect to the DNN approximation performance.

B. DNN Results and Real-Time Performance Evaluation

Priori to directly apply the proposed method on the real vehicle, semi-physical experiments were executed in a laboratory environment. The testing environment is visualized in Fig. 6, from where it can be observed that the semi-physical system contains three main parts:

1) A server (Dell EMC PowerEdge R930 rack): Generating dataset and training DNNs.
2) An industrial PC (IPC-610MB-30LDE/I5-2400/DDR3): Creating executable files via LabVIEW real-time mode.
3) An embedded controller (NI PXI-8820): Testing the control performance.



a) Semi-physical system overview b) Real-time simulator front view

Fig. 6: Semi-physical testing environment

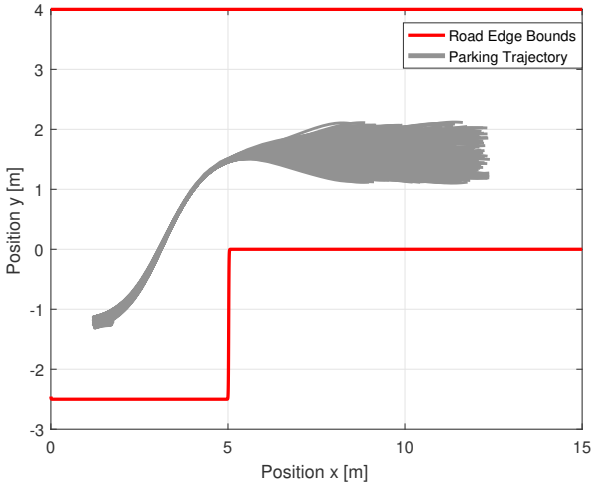


Fig. 7: Parking maneuver profile: Scenario 1

Monte-Carlo (MC) approach is adopted to assess the control performance and the real-time capability of the designed DNN-based algorithm. 500 MC trials were performed for the two mission scenarios and the DNN-driven parking trajectories are depicted in Fig. 7 and Fig. 8. The corresponding state and

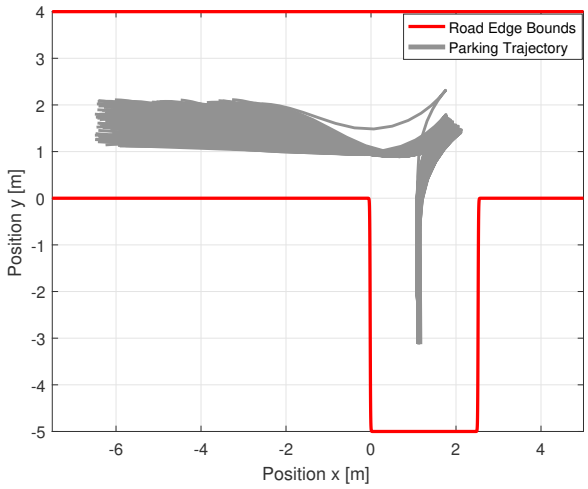


Fig. 8: Parking maneuver profile: Scenario 2

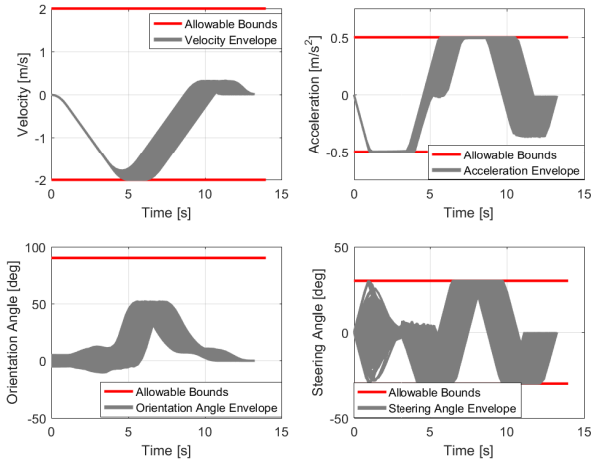


Fig. 9: State-control profiles: Scenario 1

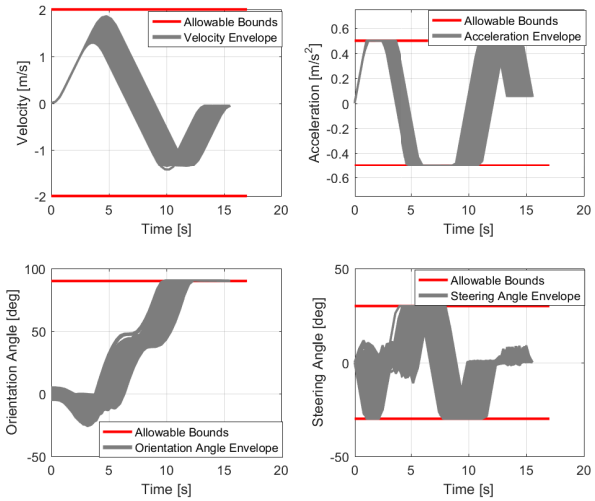


Fig. 10: State-control profiles: Scenario 2

control profiles are plotted in Fig. 9 and Fig. 10, respectively. From the results presented in Fig. 7 and Fig. 8, the proposed algorithm can successfully steer the AGV from the noise-perturbed initial pose to the target final pose for the two parking scenarios. In addition, by viewing the state/control profiles shown in Fig. 9 and Fig. 10, all the state and control path constraints can be satisfied, which confirms the validity of the obtained maneuver results.

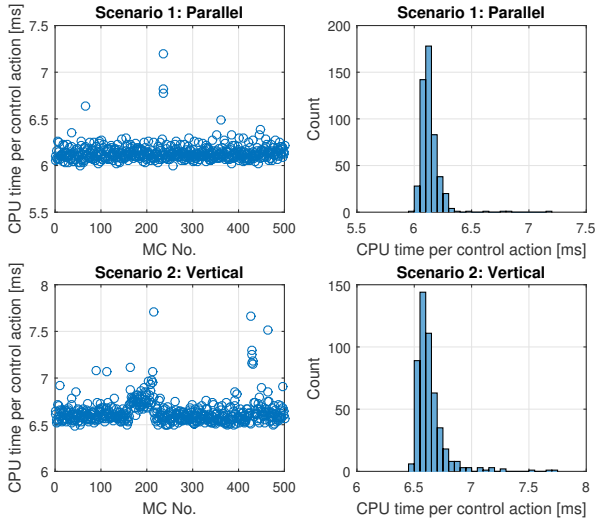


Fig. 11: Algorithm execution performance

As for the real-time applicability of the proposed algorithm, results regarding the average processing time per control action for MC each trial are visualized in Fig. 11, from where we can observe that the algorithm execution time can be kept in a small scale. This is because in real-time applications, only a finite number of forward operation is required by the algorithm, while other primary steps of the algorithm are mainly carried out offline.

C. Control Performance and Stability Analysis

Since the proposed control algorithm is acting on the AGV nonlinear system with different initial conditions, the stability issue becomes a key and must be analyzed. It is worth noting that the main objective of the parking mission is to control the AGV to reach the specified point in the parking slot. Therefore, from a practical point of view, one important indicator that can reflect the stability and quality of the parking maneuver is the terminal state errors $e_x(t_f) = x_f - x(t_f)$. Fig. 12 and Fig. 13 reveal the distribution of the terminal state errors for the two mission cases. According to the obtained results, we can see that the proposed approach is able to steer the vehicle state variable to a small neighbourhood of the target point (e.g., the terminal state error can locate in a small region around the origin). A comparative analysis was performed by applying the DNN-based algorithm with and without the data aggregation part. The resulting final error distributions are also revealed in the histograms (see Fig. 12 and Fig. 13). Clearly, the distribution of error becomes closer to the origin if the data aggregation strategy can be applied.

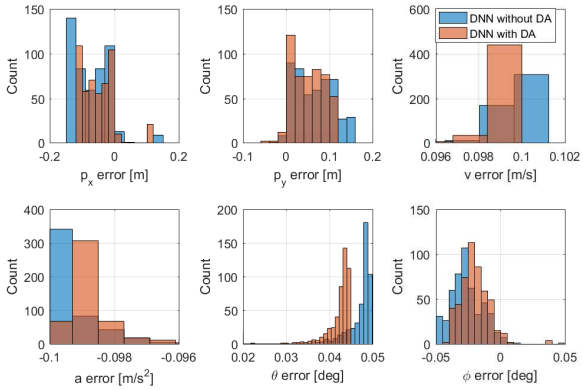


Fig. 12: Distribution of final state errors: Scenario 1

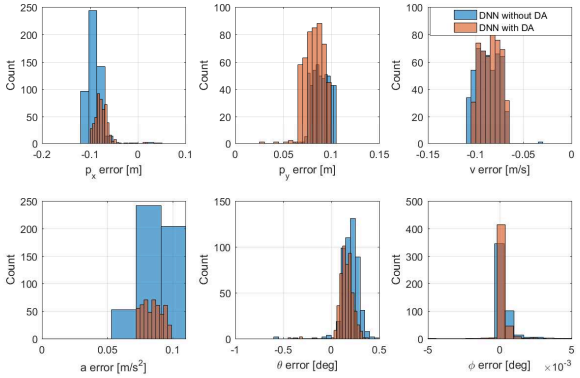


Fig. 13: Distribution of final state errors: Scenario 2

TABLE V: Final error results

Terminal error	Deisreable region	p-value Scenario 1		p-value Scenario 2	
		DNN	DNN-DA	DNN	DNN-DA
e_{p_x}	$[-0.1, 0.1]$	66.4%	78.2%	67.0%	98.6%
e_{p_y}	$[-0.1, 0.1]$	74.0%	89.4%	83.6%	99.4%
e_v	$[-0.1, 0.1]$	92.4%	99.6%	80.4%	94.6%
e_a	$[-0.1, 0.1]$	98.6%	100%	79.6%	98.8%
e_θ	$[-0.5, 0.5]$	99.8%	100%	99.6%	100%
e_ϕ	$[-0.05, 0.05]$	99.4%	100%	99.8%	100%

Detailed comparative results are summarised and tabulated in Table V, where a highly desirable region is defined to better reflect the performance enhancement achieved by using the DA. In Table V, p-value denotes the probability that e_{x_f} can locate inside the desired region. Based on the data shown in Table V, it is obvious that the results acquired via the DNN algorithm with DA tend to be much closer to the targeted final conditions and have higher probability to locate inside the desirable region. This further confirms the advantage of applying the DA established in Algorithm 2 to achieve better network control stability.

Apart from the final error distribution, another clue which can also reveal the stability of the proposed method is the DNN-driven parking trajectory behaviour after touching the target position. Ideally, these trajectory profiles should maintain a stable behaviour around the pre-specified target point. Fig. 14 illustrates the corresponding results for the parallel and

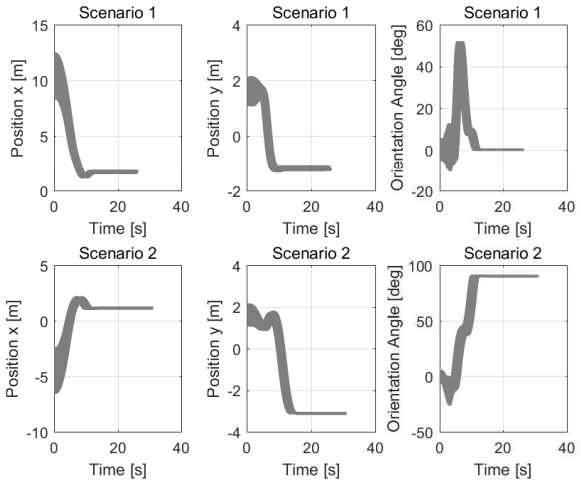


Fig. 14: Trajectory Profiles After Reaching the Target Point

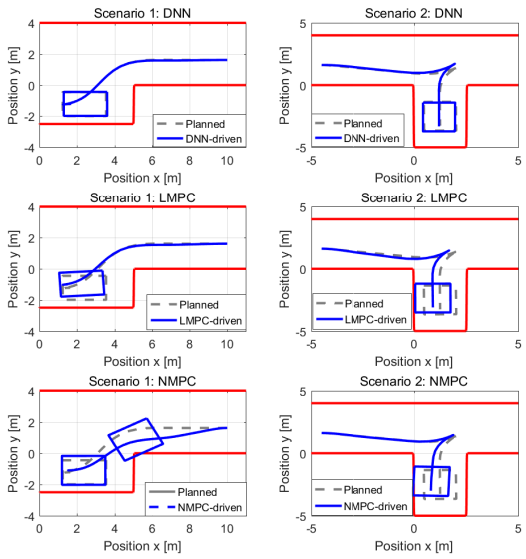


Fig. 15: Maneuver Trajectories Obtained Using Different Methods

vertical parking trials. From Fig.11, a stable behaviour can be witnessed in the vehicle position and orientation trajectory profiles (e.g., p_x, p_y and θ) for both of the two parking cases. These results further reflect the stability of the proposed design. That is, the DNN-based method is able to keep the AGV states remaining close to x_f long after the target parking point has been reached.

D. Comparative Studies Against Optimization-based Methods

Apart from the comparison carried out to study the impact of the imitation learning process on the control performance, another attempt is made to compare the proposed control scheme with other optimization-based motion control methods. Specifically, two MPC-based trajectory controllers, the linear MPC scheme (denoted as LMPC) and the nonlinear MPC scheme (denoted as NMPC), have been modified such that they are able to tackle the considered parking maneuver problem. Based on our experiments, the maximum optimization iteration should be set to 400 so as to guarantee the real-time applicability. After reaching this maximum allowable value, the MPC algorithm will terminate the current optimization process and jump to the next control loop. The two parking missions were re-executed by using the three algorithms and the obtained maneuver trajectories are displayed in Fig. 15.

Fig. 15 clearly indicates that the quality of MPC-based parking maneuver solutions is not as comparable as the one achieved using the proposed DNN-driven method. For example, the actual maneuver trajectories driven by the proposed algorithm are almost identical with the simulated optimal solutions, whereas a relatively-large deviation can be found in the LMPC solutions for both parking cases. As for the NMPC solution, a collision with the edge of the parking slot can be detected, which indicates the obtained maneuver trajectory is infeasible. Actually, one problem of applying the NMPC is that it needs to solve a nonconvex optimization model at each time step and the global convergence of the solution can hardly be guaranteed. Moreover, the consideration of various constraints might also result in heavy burdens on the optimization process. This will affect the accuracy of the obtained solution and degrade the computational performance significantly. From our observations, most of the online optimization processes were partly-solved and some of them were terminated after getting stuck at local infeasible solutions. As for the LMPC method, solving the optimization model becomes easier as the problem is transformed to a linear version. However, if the accumulation of errors caused by the successive linearization process is large, the actual maneuver trajectory is likely to diverge from the simulated optimal reference, thereby degrading the control and computational performance.

E. Experimental Results

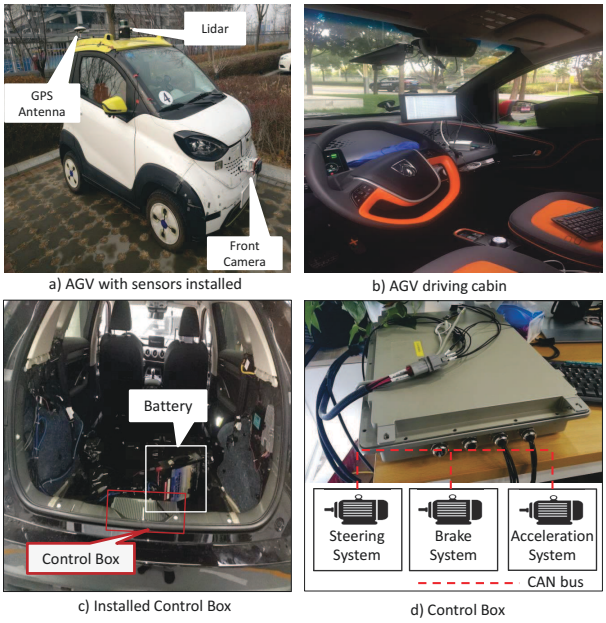


Fig. 16: Experimental Setup: Vehicle Platform

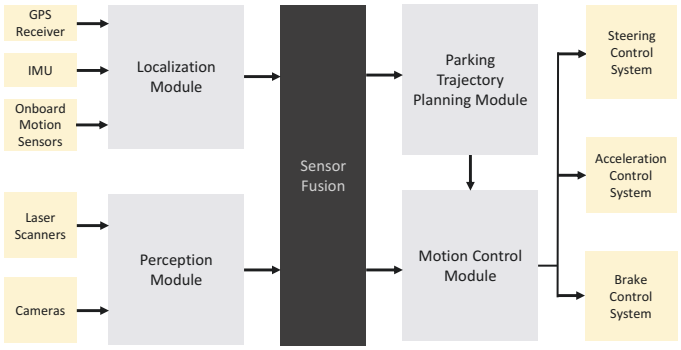


Fig. 17: An Illustration of the Entire System Structure

Although results presented in previous subsections can demonstrate remarkable performance of the proposed method in both qualitative and quantitative sides, real-world application results are much more desirable for solidly convincing evaluations. Consequently, we test our proposal in a real-world autonomous system, which is illustrated in Fig. 16(a) and Fig. 16(b). The vehicle-related parameters are identical with the setting provided in Table I. Fig. 16(c) and Fig. 16(d) provide an illustration of the vehicle control box, while the battery shown in Fig.16(c) is applied as the main power supplier of the control box.

Remark 1. Multiple sensors are installed on the vehicle such that the AGV can acquire the capability of localization and perception. Specifically, the vehicle is equipped with onboard motion sensors such as the wheel speed sensor and the steering angle sensor. Their information is shared via the controller area network (CAN). To obtain the position of the AGV, a real-time kinematics GPS receiver is installed on the vehicle (as shown in Fig. 16(a)). Besides, in order to measure the accelerations, angular rates and other attitude variables of the AGV, an inertial measurement unit (IMU) is installed and placed in the geometric center of the vehicle. Furthermore, laser scanners and cameras are also installed on the vehicle. To better present the connection between sensors and actuators, a graphical illustration of the entire automatic parking system is shown in Fig. 17, where four key components are included: localization, environment perception, parking trajectory planning and motion control. The cameras and laser scanners are mainly responsible for the environment perception module while other sensors are mainly responsible for the localization module.

Remark 2. It is important to note that multiple types of measurement noises or bias may exist in the localization module and the environment perception module. For example, in the localization module, the measured GPS raw data can be easily polluted by poor satellite signal conditions, blockage or other perturbations. As a result, the position and velocity information used to construct the trajectory optimization model may be unreliable, thereby significantly damaging the operation of the proposed automatic parking algorithm. To deal with this problem and improve the accuracy of the localization module, a GPS noise/bias correction method can be utilized [32]. The core idea of this method is to focus on the fusion of GPS data with other sensor outputs (e.g., the IMU and onboard motion sensors), thereby forming a fusion-based localization module. That is, as shown in Fig.17, multiple sensor outputs will be provided to the localization module, where a particle filter can then be applied to estimate and compensate the bias.

Under each mission scenario, we have performed four parking cases. The initial configurations of these test cases are tabulated in Table VI.

TABLE VI: Test cases and results

Case No.	Scenario 1			Final error		
	$p_x(t_0)$	$p_y(t_0)$	$\theta(t_0)$	e_{p_x}	e_{p_y}	e_θ
No. 1	12.21	2.02	-1.08	-0.12	0.13	0.07
No. 2	12.38	1.22	2.85	-0.09	0.07	0.05
No. 3	8.62	1.94	4.60	-0.14	0.10	0.05
No. 4	8.44	1.17	-4.72	-0.05	-0.03	0.04
Case No.	Scenario 2			Final error		
	$p_x(t_0)$	$p_y(t_0)$	$\theta(t_0)$	e_{p_x}	e_{p_y}	e_θ
No. 1	-6.38	1.99	-1.46	-0.11	0.15	0.11
No. 2	-6.49	1.26	-3.74	-0.12	0.15	0.24
No. 3	-2.56	1.79	4.04	-0.14	0.16	0.34
No. 4	-4.41	1.63	-0.63	-0.14	0.15	0.11

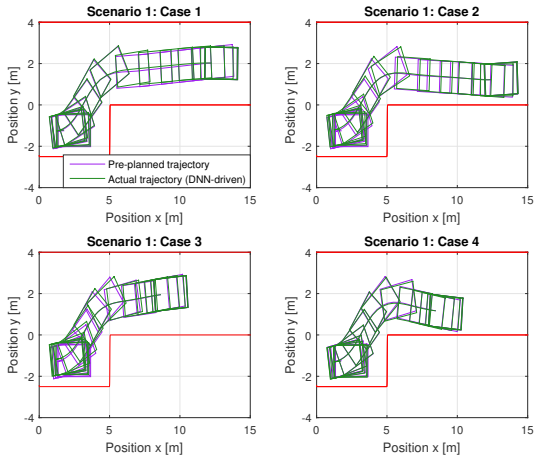


Fig. 18: Maneuver profiles for all test cases: Scenario 1

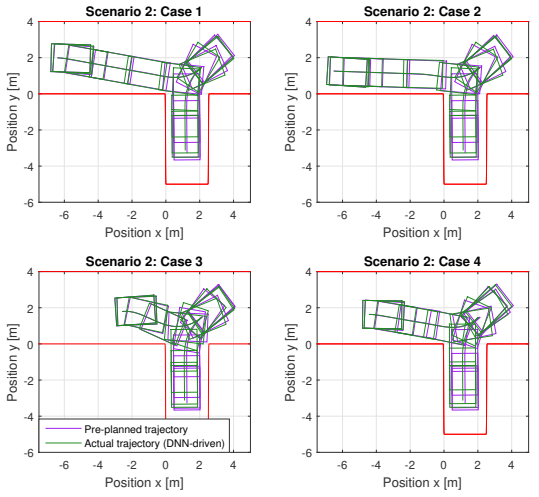


Fig. 19: Maneuver profiles for all test cases: Scenario 2

The actual parking trajectories for all the test cases are shown in Fig. 18 and Fig. 19, while the results of final position errors are included in Table VI. It can be seen from Fig. 18 and Fig. 19 that by applying the DNN-driven guidance approach, the AGV does not collide the edge of the road as well as the corner points of the parking area during the entire maneuver phase. From Table IV, the difference between the achieved final pose and the targeted final pose is relatively-small. Moreover, by viewing the results presented in Fig. 18 and Fig. 19, the actual maneuver profiles for all test cases are almost identical with respect to the simulated optimal solutions. Consequently, the results provided earlier validate the effectiveness of the proposed design and justify its enhanced performance.

Remark 3. It is true that differences can always be found in the parking environments or in the dynamics of various intelligent vehicles. In these cases, the trajectory-steering mapping relationship may vary from case to case and the collected data might be outdated. Since the neural network was trained only for two specific parking scenarios, the algorithm performance tends to be degraded if it is applied to another complex scenario. However, the proposed algorithm has the potential to be extended for other parking environments. For example, we can take the advantage of transfer learning [33]. Specifically, suppose we have a set of parking trajectories $\mathbf{D}_s$ under a particular parking scenario and a corresponding set of control actions $\mathbf{C}_s$ which can steer the vehicle to the given position. Let $\mathbf{D}_T$ and $\mathbf{C}_T$ represent the parking trajectory and control action sets of another parking scenario. The primary aim of transfer learning is to boost and reinforce the learning of mapping relationship $f_T : \mathbf{D}_T \mapsto \mathbf{C}_T$ by fully exploiting the knowledge of $f_s : \mathbf{D}_s \mapsto \mathbf{C}_s$. That is, training will be initially performed on the source domain (e.g., $\mathbf{D}_s$ and $\mathbf{C}_s$). Subsequently, a retraining process will be executed for specific hidden layers based on the target domain data (e.g., $\mathbf{D}_T$ and $\mathbf{C}_T$). In this way, a good transferability is likely to be achieved.

V. Conclusion

In this paper, a multi-layer control scheme merging optimal trajectory planning and deep learning was designed and used to steer the parking maneuver of the autonomous ground vehicles. To improve the approximation accuracy of the network, a data aggregation strategy was applied such that the DNNs can recover from past mistakes. This process was performed by re-training the DNNs on a selected trajectory ensemble containing all the wrong predicted samples. Detailed numerical studies was carried out to illustrate some important findings of the DNN-driven parking maneuver solutions. From the results, it can be concluded that:

1) The proposed intelligent control strategy has the capability of predicting optimal motion commands and steering the vehicle to the pre-specified parking slot.
2) The developed data aggregation approach can indeed provide contributions with respect to the approximation accuracy of the DNNs.
3) The proposed DNN-driven scheme is able to produce optimal feedback actions online and its real-time applicability can be verified.

Field experiments were also executed and the results verify the availability of adopting the DNN-based approach in real-wold applications.

References

[1] 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 Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3444–3457, 2019.

[2] S. Glaser, B. Vanholme, S. Mammar, D. Gruyer, and L. Nouveliere, "Maneuver-based trajectory planning for highly autonomous vehicles on real road with traffic and driver interaction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 589–606, 2010.

[3] U. Rosolia, S. D. Bruyne, and A. G. Alleyne, "Autonomous vehicle control: A nonconvex approach for obstacle avoidance," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 469–484, 2017.

[4] B. Li, K. Wang, and Z. Shao, "Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3263–3274, 2016.

[5] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, and Y. Xia, "Two-stage trajectory optimization for autonomous ground vehicles parking maneuver," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 3899–3909, 2019.

[6] Y. Kim and B. K. Kim, "Time-optimal trajectory planning based on dynamics for differential-wheeled mobile robots with a geometric corridor," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 7, pp. 5502–5512, 2017.

[7] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[8] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 11, pp. 2004–2016, 2014.

[9] T. Fraichard and A. Scheuer, "From reeds and shepp's to continuous-curvature paths," *IEEE Transactions on Robotics*, vol. 20, no. 6, pp. 1025–1035, 2004.

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

[11] J. J. Kim and J. J. Lee, "Trajectory optimization with particle swarm optimization for manipulator motion planning," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 620–631, 2015.

[12] C. Sanchez-Sanchez and D. Izzo, "Real-time optimal control via deep neural networks: Study on landing problems," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 5, pp. 1122–1135, 2018.
[13] B. A. Conway, "A survey of methods available for the numerical optimization of continuous dynamic systems," *Journal of Optimization Theory and Applications*, vol. 152, no. 2, pp. 271–306, 2012.
[14] J. Choi and K. Huhtala, "Constrained global path optimization for articulated steering vehicles," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 1868–1879, 2016.
[15] C. Mu, Z. Ni, C. Sun, and H. He, "Air-breathing hypersonic vehicle tracking control based on adaptive dynamic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 3, pp. 584–598, 2017.
[16] H. Gui and A. H. J. d. Ruiter, "Adaptive fault-tolerant spacecraft pose tracking with control allocation," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 2, pp. 479–494, 2019.
[17] P. Shi, Y. Zhang, M. Chadli, and R. K. Agarwal, "Mixed h-infinity and passive filtering for discrete fuzzy neural networks with stochastic jumps and time delays," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 903–909, 2016.
[18] N. Wang, S. Su, X. Pan, X. Yu, and G. Xie, "Yaw-guided trajectory tracking control of an asymmetric underactuated surface vehicle," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3502–3513, 2019.
[19] B. Xu, C. Yang, and Y. Pan, "Global neural dynamic surface tracking control of strict-feedback systems with application to hypersonic flight vehicle," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2563–2575, 2015.
[20] X. Cao, P. Shi, Z. Li, and M. Liu, "Neural-network-based adaptive backstepping control with application to spacecraft attitude regulation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 9, pp. 4303–4313, 2018.
[21] G. Lai, Z. Liu, Y. Zhang, and C. L. P. Chen, "Adaptive position/attitude tracking control of aerial robot with unknown inertial matrix based on a new robust neural identifier," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 1, pp. 18–31, 2016.
[22] Z.-S. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Information Sciences*, vol. 235, pp. 3–35, 2013.
[23] R. Chai, A. Tsourdos, A. Savvaris, Y. Xia, and S. Chai, "Real-time reentry trajectory planning of hypersonic vehicles: A two-step strategy incorporating fuzzy multi-objective transcription and deep neural network," *IEEE Transactions on Industrial Electronics*, pp. 1–10, 2019.
[24] T. Weiskircher, Q. Wang, and B. Ayalew, "Predictive guidance and control framework for (semi-)autonomous vehicles in public traffic," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 6, pp. 2034–2046, 2017.
[25] T. Mercy, R. V. Parys, and G. Pipeleers, "Spline-based motion planning for autonomous guided vehicles in a dynamic environment," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 6, pp. 2182–2189, 2018.
[26] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.
[27] L. Li, Y. Lin, N. Zheng, and F. Wang, "Parallel learning: a perspective and a framework," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 3, pp. 389–395, 2017.
[28] Y. Lin, L. Li, X. Dai, N. Zheng, and F. Wang, "Master general parking skill via deep learning," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, Conference Proceedings, pp. 941–946.
[29] R. Verschueren, S. D. Bruyne, M. Zanon, J. V. Frasch, and M. Diehl, "Towards time-optimal race car driving using nonlinear mpc in real-time," in *53rd IEEE Conference on Decision and Control*, 2014, Conference Proceedings, pp. 2505–2510.
[30] P. Khalaf and H. Richter, "Trajectory optimization of robots with regenerative drive systems: Numerical and experimental results," *IEEE Transactions on Robotics*, pp. 1–16, 2019.
[31] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[32] 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 Transactions on Industrial Electronics*, vol. 62, no. 8, pp. 5119–5132, 2015.
[33] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.



Runqi Chai (S'15-M'18) received the Ph.D. degree in Aerospace Engineering from Cranfield University, Cranfield, U.K, in August 2018. He is currently a research fellow at Cranfield University. His research interests include trajectory optimization, networked control systems, multi-agent control systems, and autonomous vehicle motion planning and control.



Antonios Tsourdos (M'99) obtained a MEng on Electronic, Control and Systems Engineering from the University of Sheffield, in 1995, an MSc on Systems Engineering from Cardiff University in 1996 and a PhD on Nonlinear Robust Autopilot Design and Analysis from Cranfield University in 1999. He joined the Cranfield University in 1999 as lecturer, appointed Head of the Centre of Autonomous and Cyber-Physical Systems in 2007 and Professor of Autonomous Systems and Control in 2009 and Director of Research - Aerospace, Transport and Manufacturing in 2015. Prof. Tsourdos is Chair of the IFAC Technical Committee on Aerospace Control, an member of the IFAC Technical Committee on Networked Systems, Discrete Event and Hybrid Systems, and Intelligent Autonomous Vehicles. Professor Tsourdos is also member of the AIAA Technical Committee on Guidance, Control and Navigation; AIAA Unmanned Systems Program Committee; and IEEE Control System Society Technical Committee on Aerospace Control (TCAC). He has published over 150 peer-reviewed journal and conference papers.

Prof. Tsourdos is an editorial board member on several international publications including IMechE and IEEE. He is an Associate Editor of the *Proceedings of the Institution of Mechanical Engineers Part G Journal of Aerospace Engineering*; *International Journal of Systems Science*; *IEEE Transactions on Aerospace and Electronic Systems*; and *Aerospace Science and Technology*.



Al Savvaris received the M.Eng. degree in aerospace systems engineering from the University of Hertfordshire, Hertfordshire, U.K., in 1998 and the Ph.D. degree in radiowave propagation and system design from the University of South Wales, Pontypridd, U.K., in 2004.

He is a Reader with the Centre for Cyber-Physical Systems, Cranfield University, Cranfield, U.K. He established the Autonomous Vehicle Dynamics and Control M.Sc. course and the COMAC training programme at Cranfield. His current research interests include systems integration, hybrid energy management, communication systems, embedded systems, guidance, and control. He is currently researching on Innovate U.K. Funded AirStart and USMOOTH Projects. In the past, he researched on the FLAVIIR and ASTRAEA UAS Projects, developing new technologies for unmanned systems, researching on hardware and system integration. He has published over 100 peer-reviewed journal and conference papers.



Senchun Chai (M'19-SM'19) received the B.S. and master's degrees in control science and engineering from Beijing Institute of Technology, Beijing, China, in 2001 and 2004, respectively. He received the Ph.D. degree in Networked Control System from University of South Wales, Pontypridd, U.K., in 2007.

He is currently a professor of School of Automation with Beijing Institute of Technology. He was a research fellow at Cranfield University, UK, from 2009 to 2010, and was a visiting scholar at University of Illinois at Urbana-Champaign Urbana, USA, from January 2010 to May 2010. He has published over 100 journal and conference papers. His current research interests focus on flight control system, networked control systems, embedded systems and multi-agent control systems.



Yuanqing Xia (M'15-SM'16) was born in Anhui Province, 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. His current research interests are in the fields of networked control systems, robust control and signal processing, active disturbance rejection control and flight control. He has published 8 monographs with Springer and Wiley, and more than 200 papers in journals. He has obtained Second Award of the Beijing Municipal Science and Technology (No. 1) in 2010, Second National Award for Science and Technology (No. 2) in 2011, and Second Natural Science Award of The Ministry of Education (No. 1) in 2012. He is a Deputy Editor of the Journal of the Beijing Institute of Technology, 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.



C. L. Philip Chen (S'88 - M'88 - SM'94 - F'07) received the M.S. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 1985 and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1988.

He is the Chair Professor and Dean of the College of Computer Science and Engineering, South China University of Technology. Being a Program Evaluator of the Accreditation Board of Engineering and Technology Education (ABET) in the U.S., for computer engineering, electrical engineering, and software engineering programs, he successfully architects the University of Macau's Engineering and Computer Science programs receiving accreditations from Washington/Seoul Accord through Hong Kong Institute of Engineers (HKIE), of which is considered as his utmost contribution in engineering/computer science education for Macau as the former Dean of the Faculty of Science and Technology. He is a Fellow of IEEE, AAAS, IAPR, CAA, and HKIE; a member of Academia Europaea (AE), European Academy of Sciences and Arts (EASA), and International Academy of Systems and Cybernetics Science (IASCYS). He received IEEE Norbert Wiener Award in 2018 for his contribution in systems and cybernetics, and machine learnings. He is also a highly cited researcher by Clarivate Analytics in 2018 and 2019.



https://dspace.org/

Design and implementation of deep neural network-based control for automatic parking maneuver process

Chai, Runqi; Tsourdos, Antonios; Savvaris, Al; Chai, Senchun; Xia, Yuanqing ...

2022-04-01
IEEE
https://doi.org/10.1109/TNNLS.2020.3042120
Chai R, Tsourdos A, Savvaris A, et al., (2022) Design and implementation of deep neural network-based control for automatic parking maneuver process. IEEE Transactions on Neural Networks and Learning Systems, Volume 33, Issue 4, April 2022, pp. 1400-1413

Downloaded from a DSpace repository. https://dspace.org/