Two-Stage Trajectory Optimization for Autonomous Ground Vehicles Parking Maneuver

Runqi Chai, *Member, IEEE*, Antonios Tsourdos, Al Savvaris, Senchun Chai, and Yuanqing Xia, *Senior Member, IEEE*

***Abstract*—This paper proposes a two-stage optimization framework for generating the optimal parking motion trajectory of autonomous ground vehicles. The motivation for the use of this multilayer optimization strategy relies on its enhanced convergence ability and computational efficiency in terms of finding optimal solutions under the constrained environment. In the first optimization stage, the designed optimizer applies an improved particle swarm optimization technique to produce a near-optimal parking movement. Subsequently, the motion trajectory obtained from the first stage is used to start the second optimization stage, where gradient-based techniques are applied. The established methodology is tested to explore the optimal parking maneuver for a car-like autonomous vehicle with the consideration of irregularly parked obstacles. Simulation results were produced and comparative studies were conducted for different mission cases. The obtained results not only confirm the effectiveness but also reveal the enhanced performance of the proposed optimization framework.**

***Index Terms*—Autonomous ground vehicles, irregularly parked obstacles, optimal parking trajectory, particle swarm optimization (PSO), two-stage optimization.**

I. Introduction

MOTION planning or trajectory design problems have been widely researched over the last ten years due to their increasingly significance in industry and military fields [1]–[4]. A high-quality/well-designed trajectory is usually a key for stable movement and design of advanced control systems [5], [6]. Relative works on this topic can be found in a number of engineering practices, such as mobile robot movement design [7], [8], space vehicle maneuver planning [9]–[11], and autonomous ground vehicle dynamic missions [12], [13]. More precisely, the authors in [7] constructed a Riemannian metric-based approach to plan the path for tracked-robot on the raw point cloud. In addition, a collision-free space maneuver robot motion planning problem was established and addressed in [9], wherein a hybrid path planning strategy incorporating a collision detection algorithm and a polynomial interpolation technique was applied to produce feasible motion trajectories. Besides, Yin and Chen [12] studied an autonomous wheeled vehicle motion planning task by applying the spatio-temporal template.

The mission solved in the present work is a minimum-time parking trajectory planning for a wheeled vehicle. The core aim of this problem is to generate a path/trajectory, for the given autonomous vehicle, to reach the specified parking area in the shortest time without colliding other vehicles/obstacles in the environment. Although extensive research work has been carried out on the design of trajectories for different mission profiles and many effective methods were successfully developed for producing the path, it is only in the recent five years that there has been a growing interest in planning trajectories via optimization-based strategies [14], [15]. The motivation for the use of this kind of technique is that in many real-world trajectory design scenarios, not only the path feasibility should be preserved but also various performance indices are desired to be optimized. For this reason, in this paper, special attentions are given to the implementation of optimization-based trajectory design methods.

It should be noted that currently, there are mainly two types of optimization algorithms: the intelligent methods and traditional gradient-based methods. Contributions made to implement different optimization techniques in autonomous vehicle path generation problems can be found in the literature [16]–[20]. For example, Roberge *et al.* [16] combined genetic algorithm with particle swarm optimization (PSO) so as to generate near-optimal trajectories for fixed-wing unmanned aerial vehicles in three-dimensional (3-D) environment. Similarly, Kim and Lee [17] optimized the manipulator motion by using a PSO algorithm with modified initialization strategy. However, a main disadvantage of applying these intelligent optimization algorithms is that the computation effort required for the optimization process is usually high and can hardly be afforded online.

On the other hand, Li *et al.* [18] generated the optimal motions of a wheeled vehicle based on traditional interior point method (IPM) for fulfilling an automatic parking task. A similar application of IPM can also be found in [19], where a higher order manipulation optimal motion planning problem was solved successfully. Andreas *et al.* [20] formulated a multivehicle energy-optimal motion optimization problem with the consideration of obstacle avoidance and solved it by implementing a bar-

Manuscript received June 26, 2018; revised October 12, 2018; accepted November 12, 2018. Date of publication November 26, 2018; date of current version July 3, 2019. Paper no. TII-18-1659. *(Corresponding author: Runqi Chai.)*

R. Chai, A. Tsourdos, and A. Savvaris are with the School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield MK43 0AL, U.K. (e-mail: r.chai@cranfield.ac.uk; a.tsourdos@cranfield.ac.uk; a.savvaris@cranfield.ac.uk).

S. Chai and Y. Xia are with the School of Automation, Beijing Institute of Technology, Beijing 100081, China (e-mail: chaisc97@163.com; xia_yuanqing@bit.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2018.2883545

rier functional-based gradient algorithm. Although these results confirmed the effectiveness of using gradient-based optimization techniques, the sensitivity associated with the initial guess value is high and tends to increase the computational time significantly.

To effectively deal with the issue of using intelligent or traditional gradient-based optimization algorithms, a multistage optimization framework is designed to optimize the parking movement in this study. The novelty of the proposed computational framework is to incorporate an initial trajectory generator with the gradient-based inner optimizer. The technique used for this reference trajectory generator is a modified PSO method with enhanced local exploitation ability. Compared with traditional single-stage optimization strategy, using the proposed structure tends to have improved convergence performance and reduced computational cost. This will be shown in the simulation section of this paper.

The rest of this paper will be organized as follow: Section II constructs the mathematical formulation of the time-optimal wheeled vehicle automatic parking problem as well as the collision-free constraints. Following that, in Section III, the two-stage computational optimization framework is introduced. Detailed simulation results and comparative studies are illustrated in Section IV. Finally, the concluding remark is given in Section V.

II. Autonomous Ground Vehicle Parking Optimization Problem

In this section, the parking optimization model of the wheeled vehicle is formulated. Priory to presenting in detail the formulation of the parking optimization problem studied in this research, it is also worth recalling some related works regarding different vehicle models and mission scenarios investigated in the literature. Currently, there exist various vehicle dynamic systems that can be applied to describe or control the movement of intelligent vehicles. For instance, in [21], a three-degree-of-freedom vehicle dynamic model was constructed. This model was then applied in order to develop a main-servo loop integrated chassis control system. Considering the nonholonomic constraint that limits the wheels to roll with no slip, a kinematic car-like model was established and used in [22] to plan the trajectory for the autonomous lane change maneuver. In addition, an integrated vehicle dynamic model containing the visual recognition system, electrical servo braking system, and steering system was constructed in [23]. Based on this integrated model, a nonsingular fast terminal sliding mode-based emergency braking control strategy was designed. For the parking trajectory planning problem considered in this paper, alternatively, we use kinematics of a car-like vehicle to plan the time-optimal parking trajectories.

A. Vehicle Kinematic Model

In order to describe the movement of a front-steering vehicle, the equations of motion are first constructed. The vehicle is considered as a rigid-body and the sideslip problem is ignored. As a result, its equations of motion can be described as the following system of differential equations:

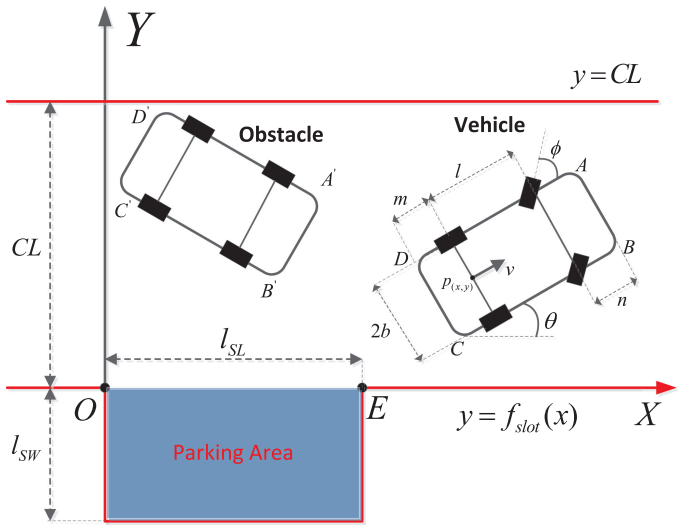


Fig. 1. Automatic parking mission.

$$\begin{cases} \dot{p}_x(t) = v(t)\cos(\theta(t)) \\ \dot{p}_y(t) = v(t)\sin(\theta(t)) \\ \dot{v}(t) = a(t) \\ \dot{a}(t) = \text{jerk}(t) \\ \dot{\theta}(t) = v(t)\tan(\phi(t))\frac{1}{l} \\ \dot{\phi}(t) = \omega(t) \end{cases}. \tag{1}$$

In (1), (p_x, p_y) is the central point of the rear wheel. $t \in [0, t_f]$ denotes the time; v and a are the vehicle velocity and acceleration variables, respectively. θ stands for the oriental angle, whereas ϕ refers to the steering angle with regard to the steering wheel. For brevity reasons, the state variables are defined as $x = [p_x, p_y, v, a, \theta, \phi]^T \in \mathbb{R}^6$. The control variables are composed by the jerk and the front wheel angular velocity ω. That is, $u = [\text{jerk}, \omega]^T \in \mathbb{R}^2$. To better illustrate the vehicle reference frame, a detailed illustration is plotted in Fig. 1. Other vehicle-dependent parameters appeared in Fig. 1 are the front overhang length n, length between the front and rear wheel l, rear overhang m, and the vehicle width $2b$, respectively.

B. Automatic Parking Process Constraints

Several mission constraints should be satisfied during the vehicle movement.

1) State Constraints: The tolerable range of the state variables can be described as follows:

$$\begin{cases} p_x(t) \in [p_x^{\min}, p_x^{\max}] \\ p_y(t) \in [p_y^{\min}, p_y^{\max}] \\ v(t) \in [v^{\min}, v^{\max}] \end{cases} \quad \begin{cases} a(t) \in [a^{\min}, a^{\max}] \\ \theta(t) \in [\theta^{\min}, \theta^{\max}] \\ \phi(t) \in [\phi^{\min}, \phi^{\max}] \end{cases}. \tag{2}$$

It should be noted that these state constraints are not just included to make the problem more complex but they do exist in reality. For example, in order to have enough reaction time for potential emergencies, the vehicle should maneuver at a relatively low speed. Hence, a box constraint is assigned to the

vehicle speed. Besides, it is not desired to have a significant variance in terms of the speed $v(t)$ as it brings discomfort to the drivers or passengers. Therefore, certain limits should be given to the magnitude of $a(t)$.

2) Control Constraints: Certain requirements should also be given to the control variables. For example,

$$|\text{jerk}(t)| \leq d_a^{\max} \tag{3a}$$

$$|\dot{k}(t)| \leq d_k^{\max}. \tag{3b}$$

In (3), $k = \frac{\tan\theta}{l}$ stands for the instantaneous curvature, whereas $\dot{k} = \frac{\omega}{l\cos^2\theta}$ is the corresponding derivative. $d_a^{\max}$ and $d_k^{\max}$ are the maximum allowable values of the jerk variable and $\dot{k}$, respectively. The aim for imposing a constraint on the jerk variable is to smoother the actual acceleration profile. As for the second control constraint (3b), similarly, it can avoid nonsmooth segment in the trajectory, thereby improving the ride comfort.

3) Parking Area Constraints and Terminal Conditions: To place the vehicle in the specific parking area shown in Fig. 1, constraints should be imposed during the vehicle maneuver and at the terminal time instant. Since rigid-body is treated as a rectangular in the 2-D plane, the four corners can be expressed by the following:

$$\begin{cases} (A_x, A_y) = (p_x + \cos(\theta)(l+n) - b\sin(\theta), \\ \qquad\qquad p_y + \sin(\theta)(l+n) + b\cos(\theta)) \\ (B_x, B_y) = (p_x + \cos(\theta)(l+n) + b\sin(\theta), \\ \qquad\qquad p_y + \sin(\theta)(l+n) - b\cos(\theta)) \\ (C_x, C_y) = (p_x - m\cos(\theta) + b\sin(\theta), \\ \qquad\qquad p_y - m\sin(\theta) - b\cos(\theta)) \\ (D_x, D_y) = (p_x - m\cos(\theta) - b\sin(\theta), \\ \qquad\qquad p_y - m\sin(\theta)) + b\cos(\theta)) \end{cases}. \tag{4}$$

After the definition of corner points, if a successful parking maneuver is achieved, the following inequality constraints should hold

$$\begin{cases} f_{\text{slot}}(A_x) \leq A_y \leq C_L \\ f_{\text{slot}}(B_x) \leq B_y \leq C_L \\ f_{\text{slot}}(C_x) \leq C_y \leq C_L \\ f_{\text{slot}}(D_x) \leq D_y \leq C_L \end{cases}. \tag{5}$$

where C_L is the width of the road. f_{slot} is given by $f_{\text{slot}}(x) = -(H(x) + H(x - l_{SL}))l_{SW}$. Here, $H(x)$ is the unit jump function. l_{SL} and l_{SW} represent the length and width of the parking area, respectively. Using the function $f_{\text{slot}}(x)$ can effectively describe the frontier of the desired parking area, and this function can easily be obtained via the linear combination of the translation and reflection of $H(x)$. Similarly, $y = CL$ is applied to describe the frontier on the other side of the road. The inequality (5) indicates the vehicle should move below $y = CL$ but above $y = f_{\text{slot}}(\cdot)$ during the entire time evolution.

To complete the entire mission, some state variables are required to reach specific values at the terminal time instant. That is

$$v(t_f) = 0, \quad a(t_f) = 0. \tag{6}$$

This implies the automatic parking will end with a full stop.

4) Obstacle Avoidance Constraints: In this research, we are interested in finding optimal parking trajectories in the presence of irregularly placed obstacles (see obstacle $A'B'C'D'$ in Fig. 1). To avoid colliding with other vehicles, obstacle avoidance constraints should be designed and entailed in optimization formulation. This is achieved by restricting that any corner point of one rectangular should be outside the other rectangular area. It is worth mentioning that if the following inequality holds true, then the edge point A can locate outside the obstacle $A'B'C'D'$.

$$S_{A'AB'} + S_{B'AC'} + S_{C'AD'} + S_{A'AD'} \geq S_{A'B'C'D'} \tag{7}$$

In (7), S stands for the area operation. As a result, if we use (7) to avoid the collision between the vehicle and the obstacle, eight inequalities should be imposed. Furthermore, during the optimization process, two additional collision-free constraints should also be taken into account in case that the vehicle will not hit the edge of the target parking slot (e.g., $O = (0, 0)$ and $E = (l_{SL}, 0)$). That is, the point O and E should locate outside the vehicle rectangular area

$$\begin{cases} S_{AOB} + S_{BOC} + S_{COD} + S_{AOD} \geq S_{ABCD} \\ S_{AEB} + S_{BEC} + S_{CED} + S_{AED} \geq S_{ABCD} \end{cases} \tag{8}$$

Remark 1: It should be noted that in [15], the authors achieved the collision-free with respect to O and E by transforming these two points to the vehicle's body frame. However, this intuitive restriction of the slot corner might introduce discontinuity, which will have negative effects for the optimization solver. Alternatively, we use an equivalent but continuous version shown in (8) to describe it in this study.

C. Objective and Overall Optimization Formulation

Since it is desirable to fulfill the parking movement in the shortest time, minimizing t_f is selected as the main objective function. The objective function J, together with the physical path constraints and obstacle avoidance constraints, formulates the automatic parking maneuver optimization model. The overall formulation is summarized as follows:

$$\begin{aligned} &\text{minimize } J = t_f \\ &\text{subject to } \forall t \in [t_0, t_f] \\ &\qquad \text{Equation}(1) \qquad\quad \text{(dynamic constraints)} \\ &\qquad \text{Equations}(2) \text{ and } (3) \quad \text{(state/control constraints)} \\ &\qquad \text{Equation}(5) \qquad\quad \text{(path constraints)} \\ &\qquad \text{Equation}(6) \qquad\quad \text{(terminal condition)} \\ &\qquad \text{Equations}(7) \text{ and } (8) \quad \text{(collision-free constraint)} \end{aligned} \tag{9}$$

III. Two-Stage Trajectory Optimization Framework

In this section, the two-stage trajectory design method is introduced and applied to address the autonomous vehicle parking maneuver problem. The discretized version of the optimal parking problem is first defined in Section III-A. Then, an initial parking movement generator is designed in Section III-B so as

to produce a feasible and near-optimal parking trajectory. Subsequently, the generated initial parking trajectory is provided to the inner gradient optimization solver that will be discussed in Section III-C. The overall structure of this two-stage method is summarized in Section III-D.

A. Discretized Optimal Parking Model

It is important to remark that the optimal control model (9) is not solvable in its present form. In order to optimize the control variables of the automatic parking problem, a necessary procedure is to parameterize the continuous-time model. Suppose that the time interval $[0, t_f]$ is divided into N_k segments. The goal of the optimization becomes finding the optimal control values at all the discrete time instants subject to the terminal time can be minimized and the constraints can be satisfied. The state can be obtained by integrating the dynamics numerically. That is, the optimization model becomes [10]

$$\begin{aligned} &\text{minimize} \;\; J = t_{N_k} \\ &\text{subject to} \;\; \forall t_k, k \in \{1, 2, \ldots, N_k\} \\ &\qquad x_{k+1} = x_k + h_k \sum_{i=1}^{s} q_i f(x_{ki}, u_{ki}) \\ &\qquad C(x_k, u_k) \leq 0 \\ &\qquad \Phi(x_{N_k}, t_{N_k}) = 0 \\ &\qquad i = 1, \ldots, s \end{aligned} \tag{10}$$

in which $C(\cdot, \cdot)$ stands for the general form of inequality constraints, whereas $\Phi(x_{N_k}, t_{N_k}) = [v_{N_k}, a_{N_k}]^T$ is the terminal condition. q_i is the discretization coefficient, and s is the stage of the integration. x_{kj} and u_{kj} are the intermediate variables defined on the current time interval $[t_k, t_{k+1}]$. Compared with the original model (9), the continuous-time system equations are transcribed into a series of algebraic equations. This static version can then be solved by standard optimization techniques [24].

B. Initial Parking Maneuver Planner

Following the establishment of discretized model, it is desired to find an efficient optimization algorithm. In this paper, a two-stage optimization structure is used to search the optimal solution of (10). The motivation for the use of this design philosophy relies on its enhanced convergence and computational ability. Traditional optimizers tend to be sensitive with the user-provided initial guess value and they are likely to get stuck at an infeasible point or local optimal point. This issue becomes more severe when the number of decision variable increases. To effectively deal with this issue, an initial parking movement generator is designed. The method used in this stage is an adaptive gradient PSO (AGPSO) method. For completeness, a brief description of this approach is stated below.

PSO is a simple swarm-based intelligent algorithm. Each particle among the swarm has a position as well as a velocity vector. That is

$$\begin{aligned} u_j(G) &= [u_{j,1}(G), \ldots, u_{j,D}(G)] \\ v_j(G) &= [v_{j,1}(G), \ldots, v_{j,D}(G)] \end{aligned} \tag{11}$$

in which $j = 1, 2, \ldots, N_j$; N_j stands for the size of the swarm. D denotes the dimensionality index of the searching space. In this way, each particle located at position u_j can be treated as a candidate solution. During the evolution process, we can denote the best position of the jth particle as $p_j(G) = [p_{j,1}, \ldots, p_{j,D}]$, whereas the best position among the swarm can be recorded as $g(G) = [g_1(G), \ldots, g_D(G)]$. Subsequently, the new velocity vector is updated according to the definitions of g and p, which can be written as follows:

$$\begin{aligned} v_{j,d}(G+1) = w v_{j,d}(G) + r_1 c_1 (p_{j,d}(G) - u_{j,d}(G)) \\ r_2 c_2 (g_d(G) - u_{j,d}(G)). \end{aligned} \tag{12}$$

In (12), several parameters are introduced. For example, w is the inertia weight parameter and is usually assigned as a constant. c_1 and c_2 are two acceleration parameters corresponding to the cognitive component $(p_{j,d}(G) - u_{j,d}(G))$ and the social component $(g_d(G) - u_{j,d}(G))$, respectively. r_1 and r_2, on the other hand, are two random parameters defined on $[0, 1]$. Based on (11) and (12), the new position of the jth particle can be computed via

$$u_{j,d}(G+1) = u_{j,d}(G) + v_{j,d}(G+1). \tag{13}$$

To evaluate the quality of the particle, the fitness function should be introduced. For general unconstrained problems, the fitness value can simply be set as the objective value. However, for the constrained parking maneuver planning problem, the constraint violation value should also be included in the fitness function so as to reflect the solution feasibility. To do this, an effective strategy is to calculate the total violation degree of the particle $Vol_{j,d}(G) \in [0, 1]$. The way to compute this value can be found in [25] and is omitted for space reasons. Consequently, the augmented fitness value associated with each particle is computed via

$$J_{j,d}^{\text{aug}}(G) = \begin{cases} J_{j,d}(G), & \text{if } Vol_{j,d}(G) = 0; \\ J^*(G) + J^*(G) Vol_{j,d}(G), & \text{if } Vol_{j,d}(G) > 0. \end{cases} \tag{14}$$

where $J^*(G)$ is the worst objective value among the Gth iteration.

It is well known that PSO method has a strong global exploration ability. In order to further enhance its local exploitation ability, a local gradient operation is embedded in the algorithm framework. Supposing that Vol and J are differentiable in their searching space, the Jacobian vector of Vol and J can be expressed by [26]

$$\begin{aligned} \nabla_u J_j &= \left[\frac{\partial J_j}{\partial u_{j,1}}, \ldots, \frac{\partial J_j}{\partial u_{j,d}} \right] \\ \nabla_u Vol_j &= \left[\frac{\partial Vol_j}{\partial u_{j,1}}, \ldots, \frac{\partial Vol_j}{\partial u_{j,d}} \right]. \end{aligned} \tag{15}$$

Based on the Jacobian vector, a local searching direction that minimizes the objective and constraint violation can be written as $e_j = -(a_1 \frac{\nabla_u J_j}{\|\nabla_u J_j\|} + a_2 \frac{\nabla_u Vol_j}{\|\nabla_u Vol_j\|})$. It is easy to verify that a decrease with regard to the augmented fitness function can be achieved by moving the current position along e. Specifically, if one calculates the inner product of $\langle e_j, -(\nabla J_j/\|\nabla J_j\|)\rangle$ or $\langle e_j, -(\nabla Vol_j/\|\nabla Vol_j\|)\rangle$, the result will be negative [26]. It is worth noting that a_1 and a_2 are two positive parameters. Since the primary task for the initial parking movement generator is to produce a feasible and near-optimal reference, these two parameters should be adjusted in an adaptive way. This is achieved by setting $a_1 = N_f/N_j$ and $a_2 = 1 - a_1$. Here, N_f refers to the number of feasible candidates among the current generation. In this way, more priorities can be given to optimizing the objective when there are more feasible candidates among the swarm and vice versa. After calculating the descent direction e_j, the current solution is updated by

$$\bar{u}_j(G) = u_j(G) + s_j e_j \tag{16}$$

where s stands for the step length along e_j.

Remark 2: The evolution process of the constructed AGPSO algorithm will continue until the number of generation reaches the limit. Since the goal of the initial movement generator is only to produce a qualified guess for the second-stage optimization process, the maximum value of G is limited to $G^{\max} = 30$ for the sake of computational burden.

Remark 3: One advantage of using the AGPSO algorithm in the first stage is that it requires no physical and theoretical knowledge of the problem. Moreover, the robustness and convergence with respect to the particle position and velocity can be guaranteed by selecting the control parameters w, c_1, and c_2 properly.

C. Optimization Strategy in the Second Stage

The optimization strategy applied in the second stage is traditional gradient method (e.g., sequential quadratic programming (SQP) or IPM). It was investigated in [25] that if a gradient-based technique starts its Newton iteration from an initial point which is near to the optimal solution, the algorithm can successfully converge to desired point with less iteration and computational time. Therefore, the initial parking maneuver trajectory obtained from the first stage is provided to the second stage gradient-based algorithm as a "warm start." At this point, most of the constraints are less likely to become active and the search direction is less restricted.

D. Implementation Consideration of the Two-Stage Optimization

In order to better illustrate the proposed parking maneuver optimization framework, the overall algorithm implementation flowchart is depicted in Fig. 2, while the two-stage optimization process is extracted and summarized in the pseudocode (see, e.g., Algorithm 1).

The two-stage optimization process is implemented over sequential calls to several function files carrying out the particle

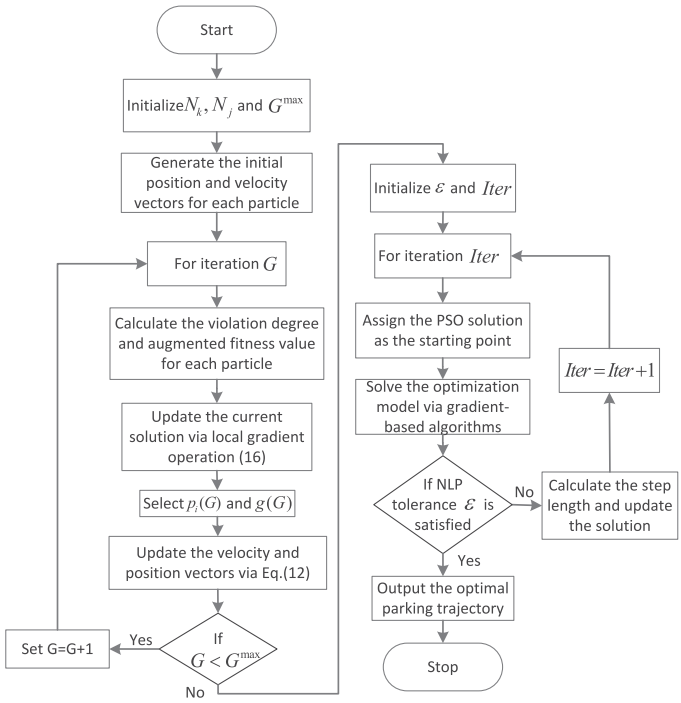


Fig. 2. Overall algorithm implementation flowchart.

Algorithm 1: Framework of the Two-stage Solver.

/*1st Stage Optimization*/
Input: Control parameters of the AGPSO w, c_1, c_2, r_1, r_2, N_j, a_1, a_2, N_k and G;
/*Main Loop*/
Step 1: Initialize the position and velocity vector;
Step 2: Calculate the augmented fitness value for each particle via (14);
Step 3: Update the particle via local gradient operation (16);
Step 4: Select $g(G)$ from the current swarm;
Step 5: Update a_1 and a_2;
Step 6: Update $v_{j,d}$ and $u_{j,d}$ via (12) and (13);
Step 7: Check if $G \leq G^{\max}$ is satisfied, if not, set $G = G + 1$ and go back to Step 2;
Output: The initial parking movement trajectory;
/*2nd Stage Optimization*/
Input: The convergence tolerance of gradient method ϵ, $Iter = 0$ and the near-optimal parking trajectory obtained from Stage 1;
/*Main Loop*/
Step 1: Check the convergence tolerance of the gradient solver:
a). if is greater than ϵ, go to Step 2;
b). if not, stop the iteration;
Step 2: Search new solution via Newton iteration [27];
Step 3: Compute the step length via the line search method and update the current solution [10], [24];
Step 4: Set $Iter = Iter + 1$ and go back to Step 1;
Output: The optimal parking maneuver trajectories;

TABLE I
VEHICLE/MISSION DEPENDENT PARAMETERS

Parameters	Values	Variables	Ranges
l_{SL}, m	5	p_x, m	$[-10, 15]$
l_{SW}, m	2	p_y, m	$[-2, 3.5]$
CL, m	3.5	v, m/s	$[-2, 2]$
n, m	0.8	a, m/s^2	$[-0.75, 0.75]$
l, m	2.5	θ, deg	$[-180°, 180°]$
m, m	0.7	ϕ, deg	$[-33°, 33°]$
$2b$, m	1.771	t, s	$[0, 50]$

TABLE II
CONTROL PARAMETERS OF THE ALGORITHM

Parameters	Values/ranges	Parameters	Values/ranges
w	$(1 + r_1)/2$	a_1	$[0, 1]$
c_1	1.49445	a_2	$[0, 1]$
c_2	1.49445	N_k	50
r_1	$[0, 1]$	$G^{\max}$	30
r_2	$[0, 1]$	ϵ	10^{-6}
N_j	100	$Iter^{\max}$	5000

TABLE III
AUTOMATIC PARKING MISSION CASES

Case No.	Initial conditions	Obstacle position O^i
1	$p_x(0) = 10.70$, $p_y(0) = 1.5$, $\theta(0) = 0$	No obstacle
2	$p_x(0) = 10.70$, $p_y(0) = 1.5$, $\theta(0) = 0$	Obstacles 1and2
3	$p_x(0) = 10.70$, $p_y(0) = 1.5$, $\theta(0) = 0$	Obstacles 1and3
4	$p_x(0) = 10.70$, $p_y(0) = 1.5$, $\theta(0) = 0$	Obstacles 1and2and3
5	$p_x(0) = 9.70$, $p_y(0) = 2.40$, $\theta(0) = -5$	Obstacles 4and5
6	$p_x(0) = 9.70$, $p_y(0) = 2.4$, $\theta(0) = -5$	Obstacles 4and5and6

initialization, computing the local gradient direction, performing the Newton iteration, and calculating the step length. A number of function files are created for different components of the algorithm. For instance, in the first optimization stage, functions are defined to produce:

1) The temporal nodes and discretization coefficients.
2) The initial position and velocity vectors of all the particles.
3) The total violation degree of each particle.
4) The augmented fitness value, $p_j(G)$ and $g(G)$.
5) The descent direction e_j and the locally updated solution.

Besides, several function files are defined in the second optimization stage so as to calculate:

1) The first and second-order derivatives of the objective function.
2) The derivative of the parking movement path constraints.
3) The step length regulated by the Goldstein condition [24].

IV. SIMULATION STUDY

In this section, the simulation results of applying the designed two-stage optimization framework to the autonomous ground vehicle parking trajectory planning problem constructed in Section II are shown. The assignment of vehicle-dependent and mission-dependent parameters is displayed in Table I, whereas control parameters for the proposed optimization scheme are tabulated in Table II.

In order to validate the performance of the proposed design, several mission cases are established and tested. The initial conditions, along with the collision-free constraint setting for different case studies, are summarized in Table III. The positional information for different obstacles O^p, $p = 1, \ldots, 6$ is given as this unnumbered eq. shown at the bottom of this page.

The state boundary values at the terminal time t_f should satisfy $x_f = [v_f, a_f]$=[0, 0]. The control constraints assigned to the jerk and curvature derivative are given by jerk$(t) \in [-0.5, 0.5]$ and $\dot{k}(t) \in [-0.6, 0.6]$, respectively. All the simulation results were generated applying MATLAB 2016b under Windows 10 and Intel (R) Core(TM) i7-4790 CPU, with 12.00 GB RAM.

$$O^1\begin{cases} A_x = 6.01, A_y = 2.61 \\ B_x = 9.95, B_y = 3.30 \\ C_x = 9.64, C_y = 5.05 \\ D_x = 5.70, D_y = 4.35 \end{cases} \quad O^2\begin{cases} A_x = 8.03, A_y = 0.66 \\ B_x = 11.97, B_y = -0.03 \\ C_x = 11.66, C_y = -1.78 \\ D_x = 7.72, D_y = -1.08 \end{cases}$$

$$O^3\begin{cases} A_x = 0.27, A_y = -0.92 \\ B_x = -3.59, B_y = -1.96 \\ C_x = -4.05, C_y = -0.24 \\ D_x = -0.19, D_y = 0.79 \end{cases} \quad O^4\begin{cases} A_x = -1.35, A_y = 2.53 \\ B_x = 2.21, B_y = 0.71 \\ C_x = 3.02, C_y = 2.29 \\ D_x = -0.55, D_y = 4.11 \end{cases}$$

$$O^5\begin{cases} A_x = 5.25, A_y = 0.50 \\ B_x = 9.18, B_y = 1.26 \\ C_x = 9.51, C_y = 0.48 \\ D_x = 5.59, D_y = -1.24 \end{cases} \quad O^6\begin{cases} A_x = 0.52, A_y = -1.25 \\ B_x = -3.43, B_y = -1.91 \\ C_x = -3.72, C_y = -0.16 \\ D_x = 0.22, D_y = 0.50 \end{cases}$$

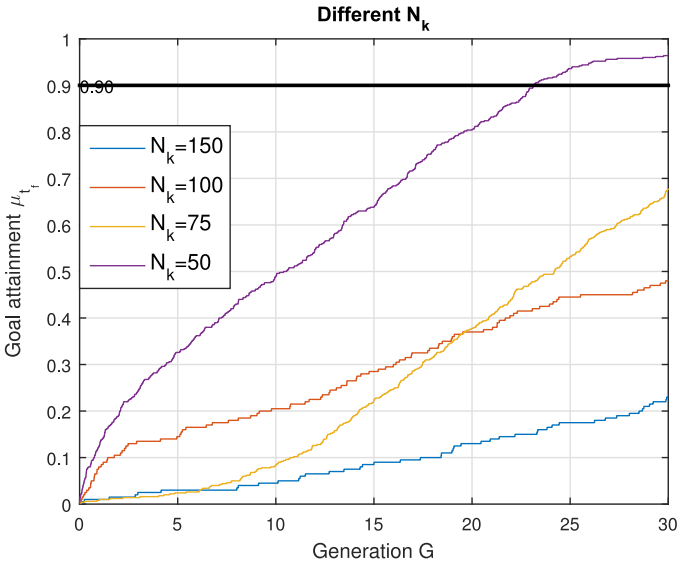


Fig. 3. Convergence results with respect to N_k.

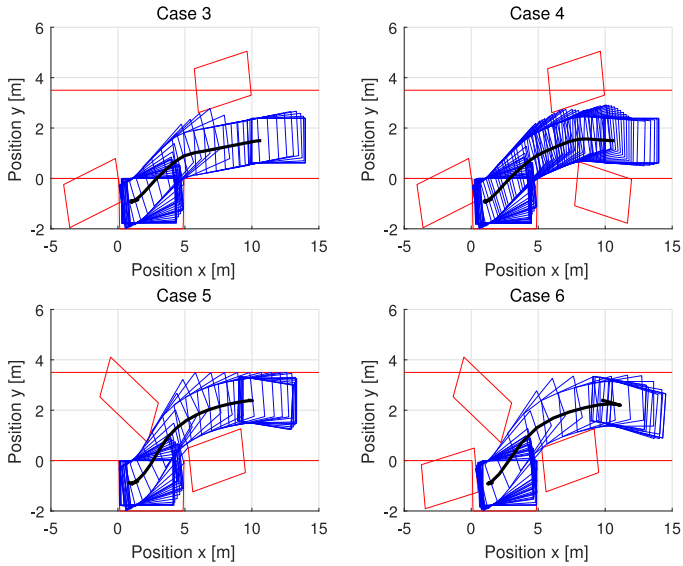


Fig. 4. Parking maneuver for Cases 3–6.

TABLE IV
SENSITIVITY ANALYSIS WITH RESPECT TO ϵ

Values for ϵ	1e-4	1e-5	1e-6	1e-7	1e-8
t_f (s)	15.072	15.045	14.140	14.128	14.122
t_p (s)	44.384	44.775	45.324	66.541	82.337

A. Parameter Analysis

In this section, the impact of different control parameters on the computational time and convergence ability is first studied. This analysis includes: First, the influence of the number of temporal node N_k; and Second, the impact of the convergence tolerance value ϵ.

To assess the convergence speed of the AGPSO optimization with respect to N_k, the goal attainment value is used as the primary indicator. Take mission Case 2 as an example, the goal attainment value $\mu_{t_f} \in [0, 1]$ is computed by $\mu_{t_f} = 1 - \frac{J^{\text{aug}} - t_f^*}{t_f^{\max} - t_f^*}$, where t_f^* and $t_f^{\max}$ are set to 15 and 20, respectively. By specifying $N_k = (50, 75, 100, 150)$, the results are depicted in Fig. 3.

Fig. 3 presents the results on the goal attainment value versus the number of generation G plane. Although using a large N_k can improve the approximation accuracy of the continuous-time model, the number of decision variables will be increased. As a result, the convergence speed of the AGPSO is decreased significantly. Since the aim of the first-stage optimization is only to produce a near-optimal parking movement to warm start the second-stage optimization, a relatively small N_k value (e.g., $N_k = 50$) is used.

Furthermore, it is found that the second-stage optimization process tends to be sensitive with respect to the convergence tolerance value ϵ. Using Case 1 as an instance, a sensitivity analysis was performed by specifying different values of ϵ and the results are displayed in Table IV. According to Table IV, a better solution can be obtained by using a smaller index of accuracy ϵ. However, the computational time t_p is monotonically increasing as ϵ becomes tighter. To balance the solution optimality and the computational burden, ϵ is set to 10^{-6} for the test trials.

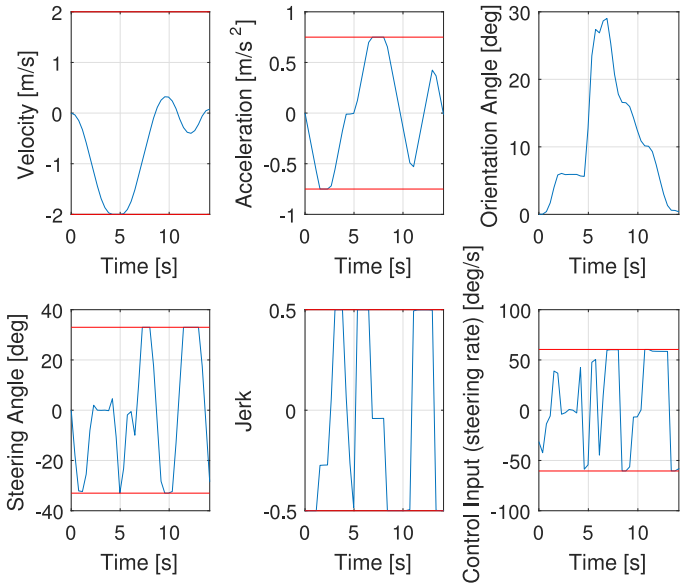


Fig. 5. Optimal trajectories for Case 1.

B. Optimal Parking Maneuver Trajectories

The optimal parking maneuver trajectory for different mission cases is partly displayed and the effectiveness of the obtained results is analyzed in this section. More precisely, the maneuver profiles for Cases 3–6 are plotted in Fig. 4, whereas the corresponding optimized state and control profiles are depicted in Figs. 5–9. Different from most previous works [14], [28], wherein regularly parked obstacles were considered, we are interested in finding optimal parking trajectories with the consideration of irregularly placed obstacles. Moreover, in Case 4 and Case 6, the target parking area is occupied partly by other vehicles. As can be seen from Figs. 4–9, the proposed two-stage optimization framework is able to optimize the

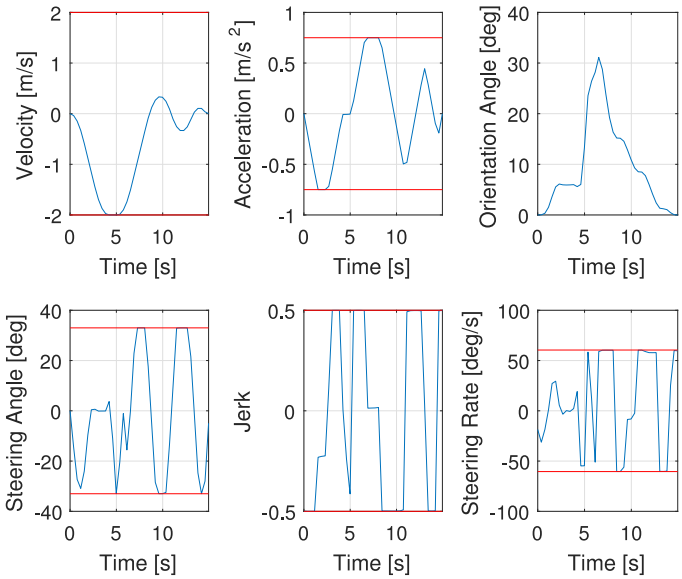


Fig. 6. Optimal trajectories for Case 3.

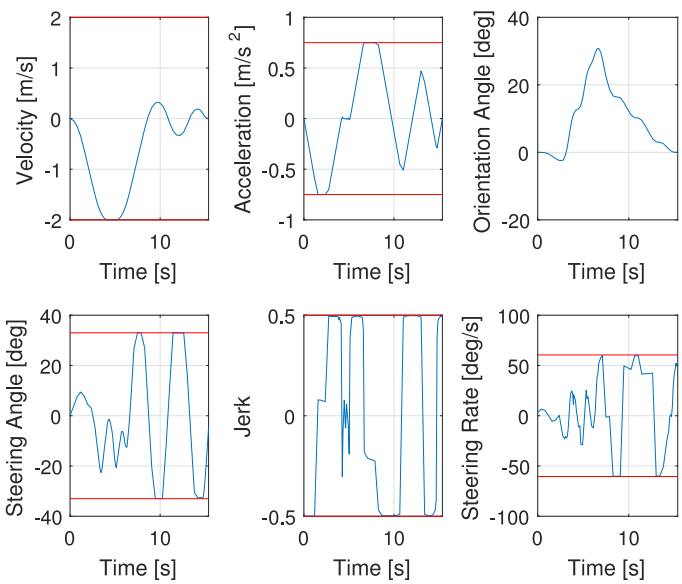


Fig. 7. Optimal trajectories for Case 4.

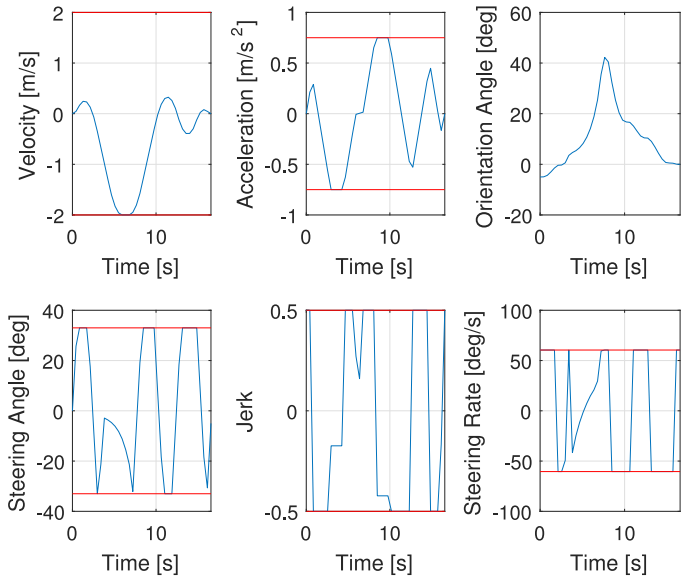


Fig. 8. Optimal trajectories for Case 5.

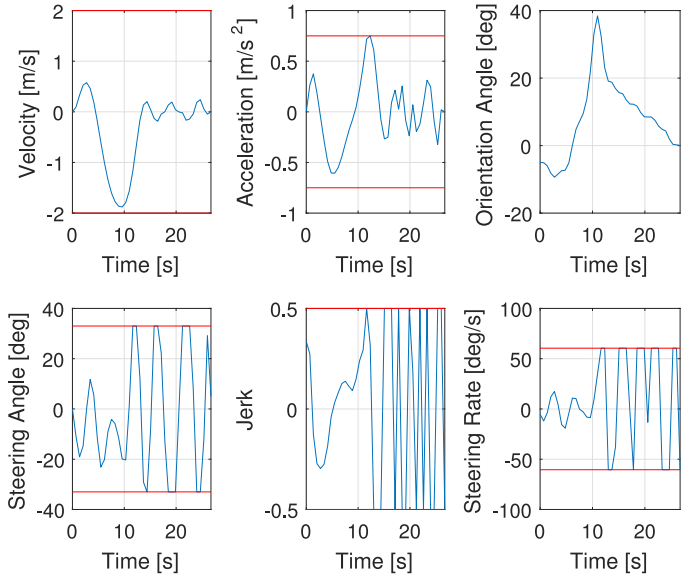


Fig. 9. Optimal trajectories for Case 6.

parking maneuver without violating the vehicle path, terminal, and collision-free constraints. In terms of the solution optimality, this can be partly reflected by the calculated control evolution profiles (especially the jerk trajectory). Since the jerk variable does not appear in the path constraint and it appears in the dynamics linearly, we can expect the jerk has a bang-singular-bang behavior for any $t \in [0, t_f]$. This conclusion can also be analogized according to the Proposition 3 stated in [10].

A significant difference of the parking movement can be found between Case 5 and Case 6. This can be explained that in Case 6, the desired parking area is largely occupied by another vehicle. When the vehicle enters the parking area, it only has limited rooms to adjust its attitude. As a result, to meet the specific final boundary conditions, more number of maneuver are required at the expense of objective value (e.g., final time t_f).

C. Comparison Against Other Techniques

This section presents a comparative study in terms of the optimal parking movement achieved by performing the proposed two-stage strategy and other optimization-based trajectory planning techniques reported in the literature. For instance, an IPM-based direct transcription method outlined in [15] and an artificial bee colony-based (ABC) intelligent optimization method studied in [29]. For the purpose of comparison, the default setting suggested in the original paper is used. Detailed optimization results including the objective and process time t_p for different parking scenarios are summarized in Table V.

It is important to note that in this study, we are interested in finding an optimization strategy such that the computational time t_p can be decreased yet the solution optimality can be maximally preserved. Hence, compared with the t_f results, t_p is

TABLE V
COMPUTATIONAL RESULTS FOR DIFFERENT METHODS

Case No.	IPM-based		ABC-based		Proposed	
	t_f (s)	t_p (s)	t_f (s)	t_p (s)	t_f (s)	t_p (s)
1	14.140	27.996	15.091	157.263	14.140	45.324
2	14.943	179.311	15.224	163.408	14.929	68.232
3	14.982	222.891	15.576	161.367	14.955	76.734
4	15.558	264.382	15.971	159.192	15.374	88.380
5	16.707	211.600	17.053	147.135	16.569	75.121
6	27.147	289.523	28.506	156.381	26.723	147.192

TABLE VI
COMPARATIVE RESULTS FOR DIFFERENT METHODS

Case No.	IPM-based		ABC-based		Proposed	
	Ind_1	Ind_2	Ind_1	Ind_2	Ind_1	Ind_2
1	0.50	1.0720	0.50	1.2175	0.50	1.0720
2	0.50	0.3063	0.50	0.5384	0.50	0.3061
3	0.50	1.0024	0.50	1.4783	0.50	1.0049
4	0.50	0.6831	0.50	0.7347	0.50	0.6775
5	0.50	0.0805	0.50	0.1177	0.50	0.0798
6	0.50	2.2058	0.48	2.8562	0.50	2.0141

rather important and should be given more attentions when assessing the performance of different algorithms. From the results presented in Table V, it can be seen that the optimal objective results t_f calculated using the proposed method and single-stage IPM-based approach investigated in [15] are comparable. For the normal case (e.g., Case 1), the IPM-based direct method can generate the parking trajectory with the smallest computational cost. However, when more irregular collision-free constraints are considered in the optimization model, it becomes difficult for a single-stage optimization structure to converge. Alternatively, the two-stage optimization framework tends to perform better in terms of achieving enhanced convergence speed (small t_p values) and objective values. This further confirms the superiority of using the investigated scheme to compute the optimal parking maneuver.

Apart from the objective value, attentions should also be given to the passenger's comfort of the obtained solutions. To assess the comfort of passengers, certain evaluation metrics are desired. It is worth noting that one important factor that could have significant influences with respect to the passenger's comfort is the path smoothness. By applying the information of the jerk and $\dot{k}$, a comfort indicator and a path smoothness indicator are defined. For example, the peak jerk value $\text{Ind}_1 = \max(|\text{jerk}|)$ and $\text{Ind}_2 = \int_0^{t_f} \dot{k} dt$, respectively. The comparative results for different parking scenarios are summarized and tabulated in Table VI.

As can be observed from Table VI, all the algorithms can produce the parking trajectory without violating the jerk path constraint. The proposed method can produce smoother results than its counterparts for most test cases. Interestingly, a relatively uneven performance can be found in the ABC-based results. This is because it applies the random initialization process and stochastic evolutionary strategies. Hence, it tends to contain more fluctuations with respect to the control variables [24].

TABLE VII
DISPERSIONS OF INITIAL CONDITIONS

Initial states	Distribution	3-σ range
$p_x(0)$, m	Uniform	$\pm 5\%$
$p_x(0)$, m	Uniform	$\pm 5\%$
$v(0)$, m/s	Zero-mean Gaussian	0.25
$a(0)$, m/s^2	Zero-mean Gaussian	0.25
$\theta(0)$, deg	Uniform	$\pm 5\%$
$\phi(0)$, deg	Uniform	$\pm 5\%$

TABLE VIII
CONVERGENCE RESULTS FOR DIFFERENT STRATEGIES

Different methods	N_s	N_{inf}	N_m	Successful rate (%)
Single optimization (Pure SQP)	183	116	701	18.3
Single optimization (Pure IPM)	210	78	712	21.0
Two-stage optimization (AGPSO+SQP)	719	28	253	71.9
Two-stage optimization (AGPSO+IPM)	733	19	248	73.3

Remark 4: From Table V, the computational cost results obtained via a single gradient optimizer might experience a significant variance between different cases. This is mainly caused by the collision-free constraints. One obstacle will result in eight path constraints and the increasing number of constraints entailed in the optimization model will tighter the searching space of the optimization process which in turn slow down the convergence speed.

D. Convergence and Robustness Analysis

Another attempt is carried out so as to analyze the convergence ability and robustness of the proposed two-stage optimization structure. A dispersion experiment was performed for the parking Case 5 with 1000 Monte-Carlo trials. The initial conditions of the vehicle are perturbed and the random initialization data are summarized in Table VII.

Comparative study was made to analyze the performance of different gradient optimization strategies, such as the single IPM-based and single SQP-based and the two-stage optimization framework. It is important to mention that in this test, the maximum number of Newton iteration $\text{Iter}^{\max}$ is limited to 500. The convergence results of the dispersion experiment are established in Table VIII.

As can be observed from Table VIII, three performance indicators are included. That is, the times of successful solution found N_s, the times of infeasible solution detected N_{inf}, and the times of maximum iteration exceeded N_m. It is obvious that compared with SQP optimization method, the IPM tends to have better solution-finding ability for the problem investigated in this study. This can be reflected by the fact that N_s achieved using IPM is generally larger than that of SQP method. Besides, the two-stage approach is able to achieve a higher successful rate with respect to finding optimal solutions compared with the single optimization structure. Therefore, it can be concluded that

the implementation of stage 1 (initial parking movement generator) can have positive influences for improving the computational time and convergence ability of the parking optimization process.

V. Concluding Remarks and Future works

In this investigation, a hybrid optimization structure was constructed and applied to address the autonomous vehicle parking motion planning problem. In order to effectively handle the sensitivity issue, an initial parking movement generator based on an AGPSO algorithm was designed. In this way, the stage-two gradient-based solver can start the solution-finding iteration at a near-optimal point, thereby improving the computational efficiency as well as the convergence ability. This conclusion was verified by carrying out a number of case studies and comparative simulations. Moreover, dispersion experiments were also performed and the results revealed the proposed two-stage strategy can have a more robust performance than traditional single-stage optimization structures.

Our follow-up work will focus on enhancing the stability of the proposed two-stage optimization algorithm such that it can be applied in dealing with more complicated vehicle models and automatic driving mission scenarios. Furthermore, it would be worthwhile to take into account some potential model errors and environmental uncertainties during the optimization process. Moreover, an integrated parking trajectory planning and tracking control system will also be designed by applying multilayer optimization-based techniques.

References

[1] G. Demesure, M. Defoort, A. Bekrar, D. Trentesaux, and M. Djemai, "Decentralized motion planning and scheduling of AGVS in an FMS," *IEEE Trans. Ind. Inform.*, vol. 14, no. 4, pp. 1744–1752, Apr. 2018.

[2] H. Guo, C. Shen, H. Zhang, H. Chen, and R. Jia, "Simultaneous trajectory planning and tracking using an MPC method for cyber-physical systems: A case study of obstacle avoidance for an intelligent vehicle," *IEEE Trans. Ind. Inform.*, vol. 14, no. 9, pp. 4273–4283, Sep. 2018.

[3] Y. Wang, S. Wang, and M. Tan, "Path generation of autonomous approach to a moving ship for unmanned vehicles," *IEEE Trans. Ind. Electron.*, vol. 62, no. 9, pp. 5619–5629, Sep. 2015.

[4] A. Rucco, G. Notarstefano, and J. Hauser, "An efficient minimum-time trajectory generation strategy for two-track car vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 4, pp. 1505–1519, Jul. 2015.

[5] B. Tian, W. Fan, R. Su, and Q. Zong, "Real-time trajectory and attitude coordination control for reusable launch vehicle in reentry phase," *IEEE Trans. Ind. Electron.*, vol. 62, no. 3, pp. 1639–1650, Mar. 2015.

[6] G. Xie, H. Gao, L. Qian, B. Huang, K. Li, and J. Wang, "Vehicle trajectory prediction by integrating physics- and maneuver-based approaches using interactive multiple models," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5999–6008, Jul. 2018.

[7] M. Liu, "Robotic online path planning on point cloud," *IEEE Trans. Cybern.*, vol. 46, no. 5, pp. 1217–1228, May 2016.

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

[9] X. Chu, Q. Hu, and J. Zhang, "Path planning and collision avoidance for a multi-arm space maneuverable robot," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 54, no. 1, pp. 217–232, Feb. 2018.

[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 Trans. Cybern*, pp. 1–14, Dec. 2017.

[11] R. Chai, A. Savvaris, and A. Tsourdos, "Fuzzy physical programming for space manoeuvre vehicles trajectory optimization based on hp-adaptive pseudospectral method," *Acta Astronaut.*, vol. 123, pp. 62–70, 2016.

[12] X. Yin and Q. Chen, "Trajectory generation with spatio-temporal templates learned from demonstrations," *IEEE Trans. Ind. Electron.*, vol. 64, no. 4, pp. 3442–3451, Apr. 2017.

[13] B. Muller, J. Deutscher, and S. Grodde, "Continuous curvature trajectory design and feedforward control for parking a car," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 541–553, May 2007.

[14] C.-K. Lee, C.-L. Lin, and B.-M. Shiu, "Autonomous vehicle parking using hybrid artificial intelligent approach," *J. Intell. Robot. Syst.*, vol. 56, no. 3, pp. 319–343, 2009.

[15] B. Li and Z. Shao, "Simultaneous dynamic optimization: A trajectory planning method for nonholonomic car-like robots," *Adv. Eng. Softw.*, vol. 87, pp. 30–42, 2015.

[16] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Trans. Ind. Inform.*, vol. 9, no. 1, pp. 132–141, Feb. 2013.

[17] J. J. Kim and J. J. Lee, "Trajectory optimization with particle swarm optimization for manipulator motion planning," *IEEE Trans. Ind. Inform.*, vol. 11, no. 3, pp. 620–631, Jun. 2015.

[18] 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.

[19] A. Reiter, A. Muller, and H. Gattringer, "On higher order inverse kinematics methods in time-optimal trajectory planning for kinematically redundant manipulators," *IEEE Trans. Ind. Inform.*, vol. 14, no. 4, pp. 1681–1690, Apr. 2018.

[20] A. J. Hausler, A. Saccon, A. P. Aguiar, J. Hauser, and A. M. Pascoal, "Energy-optimal motion planning for multiple robotic vehicles with collision avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 3, pp. 867–883, May 2016.

[21] F. You, R. Zhang, G. Lie, H. Wang, H. Wen, and J. Xu, "Trajectory planning and tracking control for autonomous lane change maneuver based on the cooperative vehicle infrastructure system," *Expert Syst. Appl.*, vol. 42, no. 14, pp. 5932–5946, 2015.

[22] R. Zhang, Z. He, H. Wang, F. You, and K. Li, "Study on self-tuning tyre friction control for developing main-servo loop integrated chassis control system," *IEEE Access*, vol. 5, pp. 6649–6660, 2017.

[23] R. Zhang, K. Li, Z. He, H. Wang, and F. You, "Advanced emergency braking control based on a nonlinear model predictive algorithm for intelligent vehicles," *Appl. Sci.*, vol. 7, no. 5, 2017.

[24] R. Chai, A. Savvaris, A. Tsourdos, S. Chai, and Y. Xia, "Improved gradient-based algorithm for solving aeroassisted vehicle trajectory optimization problems," *J. Guid., Control Dyn.*, vol. 40, no. 8, pp. 2093–2101, 2017.

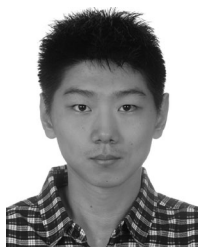
[25] R. Chai, A. Savvaris, and A. Tsourdos, "Violation learning differential evolution-based hp-adaptive pseudospectral method for trajectory optimization of space maneuver vehicle," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 4, pp. 2031–2044, Aug. 2017.

[26] R. Chai, A. Savvaris, A. Tsourdos, S. Chai, and Y. Xia, "Unified multiobjective optimization scheme for aeroassisted vehicle trajectory planning," *J. Guid., Control, Dyn.*, vol. 41, no. 7, pp. 1521–1530, 2018.

[27] A. Wachter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.

[28] F. Gomez-Bravo, F. Cuesta, A. Ollero, and A. Viguria, "Continuous curvature path generation based on b-spline curves for parking manoeuvres," *Robot. Auton. Syst.*, vol. 56, no. 4, pp. 360–372, 2008.

[29] H. Duan and S. Li, "Artificial bee colony based direct collocation for reentry trajectory optimization of hypersonic vehicle," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 51, no. 1, pp. 615–626, Jan. 2015.



Runqi Chai (S'15–M'18) was born in Beijing, China, in 1993. He received the Ph.D. degree in aerospace engineering from Cranfield University, Cranfield, U.K., in 2018.

His research interests include trajectory optimization, guidance and control.

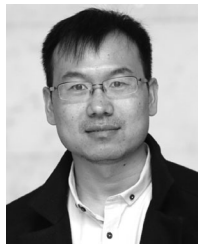
Antonios Tsourdos received the Ph.D. degree in nonlinear robust missile autopilot design and analysis from Cranfield University, Cranfield, U.K., in 1999.

He is currently a Professor in autonomous systems and control with Cranfield University. He was appointed the Head of the Autonomous Systems Group in 2007, the Head of the Centre of Autonomous and Cyber-Physical Systems in 2012, and the Director of Research—Aerospace, Transport and Manufacturing in 2015. He leads the research theme 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.

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, Bedfordshire, U.K. He established the Autonomous Vehicle Dynamics and Control M.Sc. course and the COMAC training programme at Cranfield. He has authored and coauthored more than 100 peer-reviewed journal and conference papers. His research interests and activities include systems integration, hybrid energy management, communication systems, embedded systems, guidance and control.

Dr. Savvaris was a member of the Autonomous Systems National Technical Committee, EPSRC College Review Member and Reviewer in several international publications, including IMechE and IEEE.



Senchun Chai received the Ph.D. degree in networked control systems and then held a Postdoctoral fellowship with the School of Electronics, University of Glamorgan, Pontypridd, U.K., in 2007 and 2009, respectively.

His current research interests include design of unmanned aerial vehicles (UAV), wireless sensor network control, networked control systems, and multiagent control systems.

Yuanqing Xia (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.

He has published eight monographs with Springer and Wiley, and more than 200 papers in journals. His current research interests include the fields of networked control systems, robust control and signal processing, active disturbance rejection control and flight control.

Dr. Xia 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*.