Received August 3, 2020, accepted August 22, 2020, date of publication September 1, 2020, date of current version September 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3020859

An Autonomous Parking System of Optimally Integrating Bidirectional Rapidly-Exploring Random Trees* and Parking-Oriented Model Predictive Control

JYUN-HAO JHANG AND FENG-LI LIAN, (Senior Member, IEEE)
Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan
Institute of Mechanical and Mechatronics Systems, Industrial Technology Research Institute, Hsinchu 31040, Taiwan

Corresponding author: Feng-Li Lian (fengli@ntu.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 105-2221-E-002 -135-MY3 and Grant MOST 108-2634-F-002-0024, and in part by the Industrial Technology Research Institute, Hsinchu, Taiwan.

ABSTRACT Autonomous parking techniques can be used to tackle the lacking problem of parking spaces. In this paper, a sampling-based motion planner consisting of optimizing bidirectional rapidly-exploring random trees* (Bi-RRT*) and parking-oriented model predictive control (MPC) is proposed to properly deal with various parking scenarios. The optimal Bi-RRT* approach aims to improve the common defects of traditional sampling-based motion planners, such as uncertainties of path quality and consistency, and exploring inefficiency in narrow spaces. For this reason, the proposed motion planner is able to overcome strict environments with obstacles and narrow spaces. The parking-oriented MPC is then designed for steering and speed controls simultaneously for accurately and smoothly tracking parking paths. Furthermore, the proposed controller is dedicated to work under the practical scenarios, such as vehicle considerations, real-time control, and signal delay. To verify the effects of the proposed autonomous parking system, extensive simulations and experiments are conducted in common and strict parking scenarios, such as perpendicular parking, parallel parking. The simulation results not only verify the effects of each technical element, but also show the capability to deal with the various parking scenarios. Furthermore, various on-car experiments sufficiently demonstrate that the proposed system can be actually implemented in everyday life.

INDEX TERMS Autonomous parking system, sampling-based motion planning, parking-oriented vehicle control, bidirectional rapidly-exploring random trees* (Bi-RRT*), model predictive control (MPC), perpendicular parking, parallel parking.

I. INTRODUCTION

Autonomous vehicles aim to provide convenience and comfort for people. Numerous advanced driver assistance systems (ADAS) are actually applied in daily transportation vehicles. However, only a few fully autonomous driving applications are implemented due to safety and law considerations. Therefore, because of the relatively low risks and well-known environments, autonomous parking may be the first fully autonomous application in the near future.

In addition, parking becomes one of the major challenges in metropolitan cities recently because of the increment of vehicle numbers. Also, the size of parking slot and garage space can be decreased without human factor consideration. Therefore, autonomous parking is regarded as one of the autonomous vehicle major benefits [1].

In advance, regarding to the functions of autonomous techniques, autonomous vehicles are categorized into four research issues, localization, perception, design logic, and motion planning and control according to [2]–[4]. Except motion planning and vehicle control, most of the research domains already have the existing methods to deal with autonomous parking issues. When it comes to motion planning and vehicle control techniques, there are still more strict requirements in parking scenarios that should be properly overcome. Therefore, this paper is dedicated to developing

The associate editor coordinating the review of this manuscript and approving it for publication was Heng Wang.

a parking system combining a motion planner and a vehicle controller to work around the autonomous parking scenarios, namely, for the common perpendicular and parallel parking scenarios, and also the strict ones with obstacles.

The rest of the paper is organized as follows. Section II discusses the related literature survey. Section III formulates the problem studied in this paper. Sections IV and V present the methodology about motion planning and vehicle control, respectively. Section VI describes simulation study and detailed analysis to show the properties for autonomous parking. Section VII presents experimental tests of various parking scenarios in real-road considerations and settings. Section VIII summarizes this paper.

II. LITERATURE SURVEY

To deal with autonomous parking issues, motion planning and vehicle control techniques should be developed for corresponding requirements and challenges additionally but not implemented the existing methods from other scenarios.

For motion planning, several related methods are proposed to deal with parking issues. However, they still have some limitations and difficulties. Among the existing motion planning techniques, sampling-based planners are widely used in parking scenarios.

An RRT planner with only forward motion combining with solution templates for parking scenarios is implemented to deal with the common parking issues [5]. Nonetheless, the lack of backward kinematics and fixed template limit the motion directional changes. A kinematic Bi-RRT is implemented to grow two RRT trees from start and goal configurations simultaneously to find solutions more efficiently [6]. Bi-RRT* with hybrid curvature steer and cost function design is implemented to explore environments based on two RRT* trees to generate a solution with continuous curvature and the desired pattern [7]. Also, the desired orientation RRT (DO-RRT) is proposed to overcome the exploring inefficiency in narrow spaces [8]. Nevertheless, the uncertainty of path quality is still not solved and the method in [8] is limited to some template of goal configuration. Furthermore, a two-stage RRT planner is proposed in [9]. The first stage explores environments with Bi-RRT with the Reeds-Shepp curve, and the second stage utilizes the waypoint-guided RRT (WG-RRT) to connect both RRT trees to result in a solution for advanced smooth process. Unfortunately, the tree connection and smooth process are limited to the result of the first stage.

By considering the above-mentioned limitations and problems, the smooth-feedback Bi-RRT* is proposed to generate feasible and human-like parking paths with high quality and consistency efficiently [10]. Nevertheless, the optimization and exploring efficiency in narrow spaces are not good enough. Therefore, the proposed motion planner aims to improve both of them.

On the other hand, related vehicle control techniques are reviewed in [11]. Traditionally, vehicle controllers are classified into three categories, path stabilization, trajectory tracking control, and predictive control approaches. Most of vehicle controllers are implemented for on-road driving.

Path stabilization methods focus on local exponential stability of the predefined path and control the vehicle to follow the path, such as pure pursuit [12], rear wheel position-based feedback [13], and front wheel position-based feedback.

Trajectory tracking controllers are dedicated to maintain the local exponential stability of the predefined trajectory which is time-variant, such as control Lyapunov based design [14] and output feedback linearization [15].

Predictive control approaches use state-space model to describe the kinematics for multiple dimensions. Therefore, this approach can deal with the complex kinematics for autonomous vehicles. An unconstrained linear MPC is proposed to conduct steering control for autonomous vehicles [16]. However, the constraints of speed, steering angle, and steering speed are not considered due to the limitation of the proposed optimizer in MPC. A constrained linear MPC with steering dynamics model aims to work around the steering practical problems to provide accurate and smooth steering control for autonomous vehicles [17]. Also, a multiple-constrained linear MPC takes not only steering control limitation but also road boundaries as the MPC constraints to conduct steering control [18]. Unfortunately, they do not concern about speed control because they focus on the lateral control at constant speed in on-road scenarios. A nonlinear MPC with tire dynamics is proposed to conduct active steering control and the real-time requirements are achieved by online model linearization [19] and [20]. In addition, a model predictive controller with switched tracking error is proposed to reduce lateral tracking deviation and maintain vehicle stability even for high-speed condition [21]. Although the heavy computational load of nonlinear MPC is reduced by online model linearization, the parameter requirements of nonlinear tire dynamics make the model not friendly to various vehicles. Furthermore, a tube model predictive control and time delay motion prediction are combined to deal with signal delay in practice [22]. The concept can be taken as a reference for some delay compensation requirements.

In this paper, a parking-oriented MPC is proposed to real-time generate steering and speed control for directional motion changes in autonomous parking scenarios. Also, since the linear model of the proposed controller is general, so the proposed controller is friendly to various vehicles. Moreover, signal delay in practice [22] is able to be solved in the proposed vehicle controller.

When it comes to existing autonomous parking systems, there are still some defects in motion planning and vehicle control. Some of them consider only one motion direction [23]–[27]. A path planner combing saturated control is proposed to work around perpendicular parking [28]. Nonetheless, the planning method is merely designed for perpendicular parking and the vehicle controller lacks of speed control. A path generation algorithm is proposed to enter parking slots using curves with minimum turning radius for autonomous parking [29]. A model predictive parking control using time-state control form is proposed to deal with perpendicular parking [30]. However, both methods do not consider the conditions that obstacles locate around parking slots. An automatic parking system based on scene recognition is proposed to recognize a perpendicular parking scenario and have potential to complete parking mission by developed motion planner and vehicle controller [31]. Unfortunately, the vehicle controller is not simulated to track a perpendicular parking path. That is, the motion planner and vehicle controller have not been combined to deal with perpendicular parking scenario in [31].

In this paper, the proposed motion planner aims to improve the inefficient exploration in narrow spaces in [10]. Also, parking-oriented MPC is proposed to track the resulting parking path accurately and smoothly for real-time requirements. Moreover, the proposed autonomous parking system is applied to deal with the real-world parking issues.

III. PROBLEM FORMULATION

A. VEHICLE MODEL

The vehicle model and the definition of related parameters used in this paper are presented in Figure 1. The commonly used bicycle model is applied to represent the kinematics of four wheels. The modeling details are discussed in [32] and [33]. Moreover, because the vehicle system is assumed to operate at low speed, the Ackermann steering geometry is utilized. As a result, the vehicle position, (x, y), in the world coordinate corresponds to the rear centroid of the vehicle. The heading angle, θ, is defined as the angle between the heading orientation of vehicle and the x-axis of the world coordinate. The vehicle speed, v, is defined as the scalar alone the vehicle axis. When it comes to steering kinematics, the steering angle, δ, is defined as the angle between the heading orientation of vehicle and the orientation of the front wheel. The turning radius, R_{turn}, is determined based on the dimensions of vehicle. In addition, the length of wheelbase, WB, is the distance between the axis of front wheels and that of rear wheels.

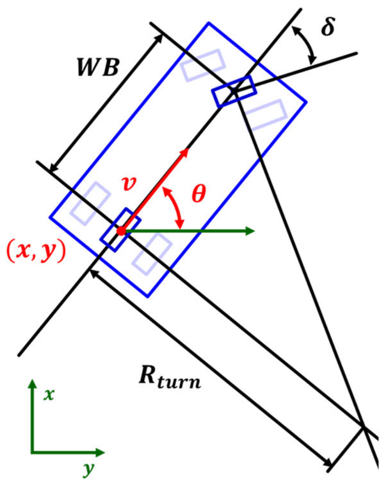

FIGURE 1. The vehicle model.

B. MPC PERIODS

When it comes to the periods of MPC, there are three kinds of periods in the proposed vehicle controller, namely, control period, T, prediction horizon, T_p, and control horizon, T_c. The control period, T, indicates that the control commands are sent to the plant in periods of T (sec).

The definition of prediction horizon, T_p, is described as follows:

$$T_p = N_p T, \quad when\ N_p \in Z^+. \tag{1}$$

The prediction horizon, T_p, indicates that each MPC execution predicts the vehicle states and control commands up to prediction horizon of next N_p periods.

On the other hand, the definition of control horizon, T_c, is described as follows:

$$T_c = N_c T, \quad when\ N_c \in Z^+. \tag{2}$$

The control horizon, T_c, indicates that the first N_c control commands among the prediction horizon are taken. In other words, the control horizon also represents the re-plan period of MPC. Usually, a control horizon of length N_c is determined based on the predictive uncertainties.

C. LINEAR STEERING AND SPEED KINEMATIC MODEL

The proposed vehicle controller aims to control steering and speed simultaneously, so the linear steering and speed kinematic model in [34] and [33] is implemented as reference. The detail is presented as follows:

$$z_{k+1} = Az_k + Bu_k + C, \tag{3}$$

where the vehicle state, z_k, and the control input, u_k, are defined as follows:

$$z_k = \begin{bmatrix} x_k \\ y_k \\ v_k \\ \theta_k \end{bmatrix}, \quad u_k = \begin{bmatrix} a_k \\ \delta_k \end{bmatrix}, \tag{4}$$

and the state space matrices A, B, and C, are described as follows:

$$A = \begin{bmatrix} 1 & 0 & cos\left(\bar{\theta}\right) dt & -\bar{v} sin\left(\bar{\theta}\right) dt \\ 0 & 1 & sin\left(\bar{\theta}\right) dt & \bar{v} cos\left(\bar{\theta}\right) dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \dfrac{tan(\bar{\delta})}{L} & 1 \end{bmatrix}, \tag{5}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ dt & 0 \\ 0 & \dfrac{\bar{v}}{L\ cos^2(\bar{\delta})} dt \end{bmatrix}, \tag{6}$$

$$C = \begin{bmatrix} \bar{v}\, sin(\bar{\theta})\, \bar{\theta} dt \\ -\bar{v}\, cos(\bar{\theta})\, \bar{\theta} dt \\ 0 \\ \dfrac{\bar{v} * \bar{\delta}}{L\, cos^2(\bar{\delta})} \end{bmatrix}. \tag{7}$$

This linear state space model aims to utilize acceleration, a, and steering angle, δ, as control input to control the vehicle state like position, (x, y), speed, v, and heading, θ. Moreover, the state space matrices A, B, and C, are determined by the prediction state and control input in discrete-time format. Therefore, the implemented linear state space model is a discrete-time model and control for autonomous vehicles.

IV. OPTIMAL FEEDBACK BIDIRECTIONAL RRT* WITH REEDS-SHEPP CURVE

A motion planner using smooth-feedback Bi-RRT* is utilized to plan parking paths in high path quality and consistency for various parking scenarios [10]. Moreover, the proposed motion planner revises the original framework to improve exploration and optimization. Also, two additional samplers using Gaussian distribution and uniform distribution to enhance exploration in narrow spaces. The proposed framework is described in the following and illustrated in **Algorithm 1**.

A. COLLISION-AVOIDABLE SAMPLER

The collision-avoidable sampler is proposed to make the resulting nodes avoid obstacles and boundaries efficiently (Lines 1 & 2). The collision-avoidable sampler aims to locate sampling nodes near the occupancy grid with a certain distance based on the collision checking radius, $R_{collision}$. Furthermore, the heading of the sampling nodes should be perpendicular to the vector from the occupancy grid to the sampling node. Therefore, the sampling nodes of the collision-avoidable sampler are able to help RRT* trees avoid obstacles and boundaries. An example of the desired random state distribution of $State_{rand}$ is shown in Figure 2.

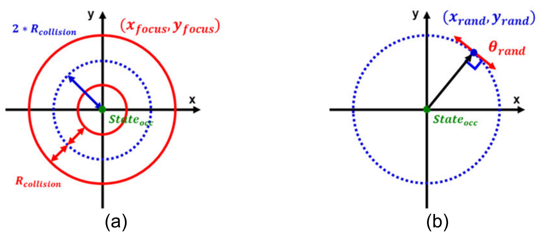

FIGURE 2. The desired distribution results of collision-avoidable sampler. (a) The desired position distribution (x, y) based on collision checking radius $R_{collision}$. (b) The desired heading θ when $State_{occ}$ equals to $(0, 0, \pi/2)$.

To achieve this goal, the collision-avoidable sampler is defined as follows. For a starter, an occupied grid is picked up from the contour of obstacles and boundaries in map randomly as the reference point $State_{occ} = (x_{occ}, y_{occ}, \theta_{occ})$. Then, the polar coordinates of the new random nodes

Algorithm 1 Optimize-Feedback Bi-RRT* With RS Curve

```
1: $MixedSampler_{start} \leftarrow Sampler_{free}, Sampler_{avoid}$
2: $MixedSampler_{goal} \leftarrow Sampler_{radial}, Sampler_{free},$
     $Sampler_{avoid}$
3: $Tree\ Tree_{start} \leftarrow$ Setup($Start\_point, MixedSampler_{start}$)
4: $Tree\ Treeg_{goal} \leftarrow$ Setup(($Gaol\_point$,
     $MixedSampler_{goal}$)
5: $LoopNum \leftarrow 0$
6: while $ExecutionTime < LimitedTime.$ do
7:    if $LoopNum\%2 == 0$ then
8:       $Node_{start} \leftarrow Tree_{start}$.iterate$_{rand}$()
9:       $Node_{goal} \leftarrow Tree_{goal}$.iterate$_{connect}$($Node_{start}.State$)
10:   else
11:      $Node_{goal} \leftarrow Tree_{goal}$.iterate$_{rand}$()
12:      $Node_{start} \leftarrow Tree_{start}$.iterate$_{connect}$($Node_{goal}.State$)
13:   end if
14:   if $Node_{start}.State == Node_{goal}.State$ then
15:      $Sol_{connected} \leftarrow$ Connect($Node_{start}, Node_{goal}$)
16:      $Sol_{interpolated} \leftarrow$ Interpolate($Sol_{connected}$
         $Res_{interpolate}$)
17:      $Tree\ Tree_{optimize} \leftarrow$ Setup($Start\_point$)
18:      $Sol_{optimal} \leftarrow Tree_{optimize}$ Optimize($Sol_{interpolated}$ )
19:      if $Sol_{optimal} =$ NULL then
20:         continue
21:      end if
22:      if $Sol_{optimal}$.cost() $< Sol_{best}$.cost() then
23:            OptimizeFeedback( $Sol_{optimal}, Tree_{start},$
             $Tree_{goal}$)
24:            $Sol_{best} \leftarrow$ Sampling($Sol_{optimal}, Res_{sample}$)
25:            StateSpaceConvergence($Sol_{best}$)
26:         end if
27:   end if
28:   $LoopNum$ ++
29: end while
30: return $Sol_{best}$
```

$(\rho_{rand}, \varphi_{rand})$ with $State_{occ}$ as the center should be derived. The polar radius ρ_{rand} can be obtained based on normal distribution as follows:

$$\rho_{rand} = \mathcal{N}\left(2 * R_{collision}, (0.5 * R_{collision})^2\right), when$$
$$\rho_{rand} \geq R_{collision}. \tag{8}$$

On the other hand, the polar angle φ_{rand} is calculated based on the uniform distribution as follows:

$$\varphi_{rand} = \mathcal{U}(-\pi, \pi). \tag{9}$$

According to (8) and (9), most of the random polar coordinates focus on the region between the circles with radius $R_{collision}$ and $3R_{collision}$, respectively, as shown in Figure 2(a).

For the derivation of $State_{rand}$, x_{rand} and y_{rand} can be derived based on $State_{occ}$ and the random polar coordinates $(\rho_{rand}, \varphi_{rand})$ as follows:

$$\begin{cases} x_{rand} = x_{occ} + \rho_{rand} * cos(\varphi_{rand}) \\ y_{rand} = y_{occ} + \rho_{rand} * sin(\varphi_{rand}) \end{cases} \tag{10}$$

Moreover, θ_{rand} is defined as follows:

$$\theta_{rand} = \begin{cases} atan2^{-1}\dfrac{x_{occ} - x_{rand}}{y_{rand} - y_{occ}}, if\ \mathcal{U}(-1, 1) \geq 0 \\ atan2^{-1}\dfrac{x_{occ} - x_{rand}}{y_{rand} - y_{occ}} + \pi, if\ \mathcal{U}(-1, 1) < 0. \end{cases} \quad (11)$$

Therefore, θ_{rand} can be perpendicular to the vector from (x_{occ}, y_{occ}) to (x_{rand}, y_{rand}) as shown in Figure 2(b).

According to the design of the collision-avoidable sampler, the resulting sampling nodes are able to help RRT* tree avoid obstacles and boundaries, and even improve the exploration efficiency in narrow spaces.

B. THE REVISED RADIAL SAMPLER

The proposed radial sampler is revised from [35] to make the RRT* tree grow out from narrow parking slots. With vehicle kinematic consideration, the revised radial sampler aims to focus the sampling nodes around the goal configuration and especially along with the goal heading. Moreover, the heading of sampling nodes should distribute around the goal heading but not lose the probabilistic property to explore the environment near the goal. An example of desired random state distribution of $State_{rand}$ is shown in Figure 3.

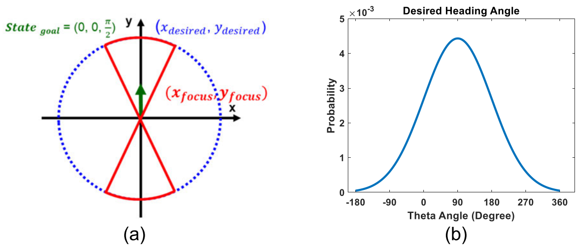

FIGURE 3. The desired distribution results of the revised radial sampler. (a) The desired position distribution. (b) The desired heading θ when $State_{goal}$ equals to (0, 0, π/2).

To achieve this goal, the revised radial sampler is defined as follows. First of all, the proposed radial sampler regards the goal configuration, $State_{goal} = (x_{goal}, y_{goal}, \theta_{goal})$, as the reference center. Then, the polar coordinates are generated by the uniform distribution according to reference center. The random polar radius ρ_{rand} depending on the RRT* search radius R_{search} is defined as follows:

$$\rho_{rand} = R_{search} * \mathcal{U}(0, 1). \quad (12)$$

As a result, the distribution of the distance between sampling nodes and goal configuration can be determined by the design of $\rho_{desired}$.

On the other hand, the polar orientation φ_{rand} which makes the sampling state focus on the front and behind locations of the goal configuration as follows:

$$\varphi_{rand} = \begin{cases} \theta_{goal} + \dfrac{\pi}{4}\mathcal{N}(0, 1), & if\ \mathcal{U}(-1, 1) \geq 0 \\ \theta_{goal} + \dfrac{\pi}{4}\mathcal{N}(0, 1) + \dfrac{\pi}{2}, & if\ \mathcal{U}(-1, 1) < 0 \end{cases} \quad (13)$$

Furthermore, the x and y values of the random state can be derived based on $State_{goal}$ and the random polar coordinates $(\rho_{rand}, \varphi_{rand})$ as follows:

$$\begin{cases} x_{rand} = x_{goal} + \rho_{rand} * cos(\varphi_{rand}) \\ y_{rand} = y_{goal} + \rho_{rand} * sin(\varphi_{rand}). \end{cases} \quad (14)$$

Based on (12) and (13), the random position distribution of (x_{rand}, y_{rand}) is able to perform as Figure 3(a). In addition, θ is randomized based on θ_{goal} as follows:

$$\theta_{rand} = \theta_{goal} + \frac{\pi}{2} * \mathcal{N}(0, 1). \quad (15)$$

Regarding to the design of (15), the random heading angle θ_{rand} can focus around the goal heading angle θ_{goal} as shown in Figure 3(b). Finally, the random state $State_{rand}$ can be derived.

Based on the revised radial sampler, the iterations of $Tree_{goal}$ can focus around the goal configuration along the heading of the goal. Moreover, the heading of the random state also focuses around the heading of the goal. Therefore, the low probability of feasible node generation for narrow parking slot can be improved.

C. OPTIMAL FEEDBACK PROCESS

The proposed process aims to improve the optimization and exploring efficiency in [10]. As shown in Figure 4, The former two steps are to optimize the connected solutions, interpolated by the Reeds-Shepp curve, with RRT* as the original version. The original process in [10] feeds the smooth process results back to RRT* trees for each found solution. However, it may result in heavy load for the following computation and fall into a local minimum easily. Compared to the one in [10], the proposed process conducts the feedback function only when the optimal solution is better than the best solution (Lines 22 & 23). The most critical information can improve the following exploration and optimization.

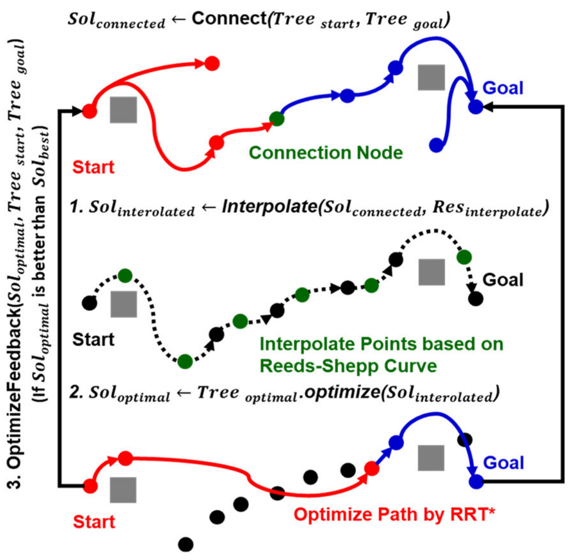
$Sol_{connected} \leftarrow$ Connect($Tree_{start}$, $Tree_{goal}$)
Start
Connection Node
Goal
1. $Sol_{interolated} \leftarrow$ Interpolate($Sol_{connected}$, $Res_{interpolate}$)
Start
Interpolate Points based on Reeds-Shepp Curve
Goal
2. $Sol_{optimal} \leftarrow Tree_{optimal}$.optimize($Sol_{interolated}$)
Start
Optimize Path by RRT*
Goal
3. OptimizeFeedback($Sol_{optimal}$, $Tree_{start}$, $Tree_{goal}$)
(If $Sol_{optimal}$ is better than Sol_{best})

FIGURE 4. The flowchart of optimal feedback process.

V. PARKING-ORIENTED MODEL PREDICTIVE CONTROL

The proposed vehicle controller aims to overcome the autonomous parking issues and control the vehicle according to a planned parking path. To overcome strict environments and frequently directional changes in autonomous parking, the proposed vehicle controller aims to control vehicle speed and steering simultaneously to track the results of the proposed motion planner accurately and smoothly. Also, the proposed vehicle controller aims to work around some practical problems, such as real-time difficulties due to simultaneous speed and steering control, the difference between steering command and the actual steering response of vehicle, and even the time delay of computation. The main structure of the proposed vehicle controller is developed based on the model predictive speed and steer control in [34].

A. OVERVIEW OF THE PROPOSED VEHICLE CONTROLLER

The system structure of the proposed vehicle controller is shown in Figure 5. As usual, there are three parts in an MPC controller, namely, the state space model, the cost function and constraints design, and the optimizer. Moreover, the simultaneous computation and control framework is implemented in the proposed vehicle controller. Linear state space model $\hat{z} = [\hat{x}, \hat{y}, \hat{v}, \hat{\theta}]^T$ is determined based on the compensated state of vehicle $z_{compensated}$ from the simultaneous computation and control framework and the past input command sequence $\hat{u} = [\hat{a}, \hat{\delta}]^T$. Then, the path tracking problem is formulated by the parking-oriented cost function and constraints. According to the reference trajectory $z_{ref} = [x_{ref}, y_{ref}, v_{ref}, \theta_{ref}]^T$, the state space model and tracking problem formulation, the optimizer [36] solves the optimization problem to derive a sequence of optimal outputs. Then, the sequence of optimal outputs is smoothened by the proposed steering command smoother to obtain the control sequence $u = [a, \delta]^T$. In advance, the simultaneous computation and control framework stores the resulting control sequence $u = [a, \delta]^T$, and outputs the current control command $u_i = [v_i, \delta_i]^T$ to the autonomous vehicle which requires speed commands instead of acceleration unlike simulations. When the proposed controller requires to subscribe the vehicle state $z_v = [x_v, y_v, v_v, \theta_v]^T$ and $u_v = [\delta_v]^T$, the simultaneous computation and control framework will predict the state $z_{compensated}$ based on the execution time delay and signal delay in practice by time-varying motion prediction.

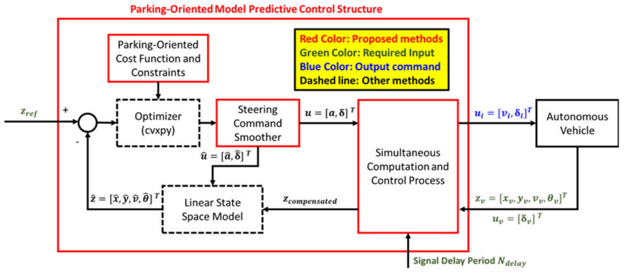
Parking-Oriented Model Predictive Control Structure
Parking-Oriented Cost Function and Constraints
Red Color: Proposed methods
Green Color: Required Input
Blue Color: Output command
Dashed line: Other methods
z_{ref}
+
-
Optimizer (cvxpy)
Steering Command Smoother
$u = [a, \delta]^T$
$u_i = [v_i, \delta_i]^T$
Autonomous Vehicle
$\hat{u} = [\hat{a}, \hat{\delta}]^T$
Simultaneous Computation and Control Process
$\hat{z} = [\hat{x}, \hat{y}, \hat{v}, \hat{\theta}]^T$
Linear State Space Model
$z_{compensated}$
$z_v = [x_v, y_v, v_v, \theta_v]^T$
$u_v = [\delta_v]^T$
Signal Delay Period N_{delay}

FIGURE 5. The system structure of the proposed vehicle controller.

B. TRACKING STRATEGY FOR AUTONOMOUS PARKING

To take both vehicle mechanism and tracking accuracy into account, the tracking strategy for autonomous parking should be defined first.

1) SEGMENT BY SEGMENT TRACKING

In practice, the vehicle should stop completely and change motion direction. However, the speed control of the MPC controller usually does not consider this situation properly. Therefore, the strategy of segment-by-segment tracking is implemented in the proposed vehicle controller to track a single motion direction each time as shown in Figure 6.

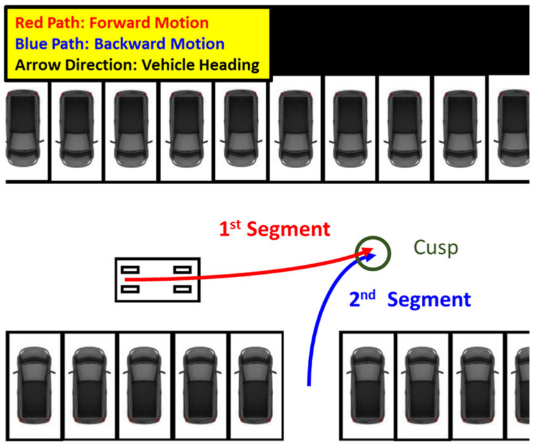

FIGURE 6. An example of tracking strategy for autonomous parking.

2) STEERING ANGLE INITIALIZATION FOR EACH SEGMENT

In addition, to overcome the strict environment, the results of motion planner may require turning the steering angle to opposition at cusps as shown in Figure 6. Therefore, the steering angle initialization for each segment should be implemented to prevent the tracking error by steering angle difference.

Moreover, the formulation of steering angle initialization should be defined. For a starter, the judging distance, d_{judge}, of the steering angle initialization should be determined. Because the beginning waypoints are highly concerned about the initial steering angle, the judging distance, d_{judge}, is determined as follows:

$$d_{judge} = v_{target} * \frac{\delta_{max}}{\Delta\delta_{max}}, \tag{16}$$

where v_{target} is the target velocity, δ_{max} is the maximum steering angle, and $\Delta\delta_{max}$ is maximum steering speed.

For physical meaning, d_{judge} indicates the distance which the vehicle moves during the switching time of steering angle from 0 to δ_{max}. The initialized steering is defined as follows:

$$\delta_0 = \begin{cases} sgn\left(\kappa_{avg}\right) * \delta_{max}, & if \ \left|\kappa_{avg}\right| \geq 0.5\left|\kappa_{max}\right| \\ 0, & else, \end{cases} \quad (17)$$

where

$$\kappa_{avg} = \frac{\sum_{i=0}^{N-1} \kappa_i}{N}, \quad where\ N = \left\lceil \frac{d_{judge}}{Res_{sample}} \right\rceil. \quad (18)$$

The value of κ_{avg} is determined by the judging distance, d_{judge}, and the sampling resolution of the proposed motion planner, Res_{sample}.

Therefore, as shown in (17), if the value of average curvature κ_{avg} within the judging distance d_{judge} is larger than half of maximum curvature κ_{max}, the initial steering angle δ_0 should be set as δ_{max} with corresponding sign.

Based on the steering angle initialization, the tracking error due to the large curvature of the beginning waypoints can decrease.

C. PARKING-ORIENTED COST FUNCTION AND CONSTRAINTS

The parking-oriented cost function and constraints are designed to overcome the large curvature in parking scenarios. Most of model predictive controllers are designed for on-road scenarios at high speed [16]–[22]. To maintain lateral stability, the limitation of the max steering angle decreases as the increment of vehicle speed [17]. Therefore, the parking-oriented tracking problem is formulated in the paper to solve path tracking with large curvature.

The cost function J_{MPC} of the proposed vehicle controller with in prediction horizon length N_p is defined as follows:

$$\begin{aligned} J_{MPC} &= z_{err,k+f}^T Q_f z_{err,k+f} \\ &+ \sum_{i=0}^{f=N_p-1} \begin{matrix} z_{err,k+i}^T Q z_{err,k+i} \\ +u_{k+i}^T R u_{k+i} \\ +u_{jerk,k+i}^T Rd u_{jerk,k+i} \end{matrix}, \end{aligned} \quad (19)$$

where the state error is presented as $z_{err,i} = \left[z_{ref,i} - \hat{z}_i\right]$ and the control jerk is presented as $u_{jerk,i} = [u_{i+1} - u_i]$.

There are four cost terms (19), namely, the state error cost, the final state error cost, the control energy cost, and the control jerk cost, respectively. The state error cost and the final state error cost aim to decrease the value of state error to perform accuracy in path tracking. The state error weighting matrix Q is a 4 by 4 matrix to correspond to the elements of vehicle state $z = [x, y, v, \theta]^T$ as follows:

$$Q = Q_f = diag\left(q_x, q_y, q_v, q_\theta\right). \quad (20)$$

Then, the control energy cost aims to compress the control input energy for smooth control. The control energy weighting matrix R is a 2 by 2 matrix to represent the cost weight of control energy at straight and curve described as follows:

$$R = \begin{cases} R_{straight} = diag\left(r_{straight,a}, r_{straight,\delta}\right), \\ \qquad if\ \kappa_{N_p-1} < 0.5\kappa_{max} \\ R_{curve} = diag\left(r_{curve,a}, r_{curve,\delta}\right), \\ \qquad if\ \kappa_{N_p-1} \geq 0.5\kappa_{max} \end{cases} \quad (21)$$

Finally, the control jerk cost is designed to prevent dramatic change of control input so that control smooth can be improved. The control jerk weighting matrix Rd is a 2 by 2 matrix to represent the cost weight of control jerk at straight and curve defined as follows:

$$Rd = \begin{cases} Rd_{straight} = diag\left(rd_{straight,a}, rd_{straight,\delta}\right), \\ \qquad if\ \kappa_{N_p-1} < 0.5\kappa_{max} \\ Rd_{curve} = diag\left(rd_{curve,a}, rd_{curve,\delta}\right), \\ \qquad if\ \kappa_{N_p-1} \geq 0.5\kappa_{max} \end{cases} \quad (22)$$

When it comes to the parking-oriented cost function design, the considerations of state error and control smoothness are taken into account. Moreover, different controls related weighting matrices at curve are designed to overcome the curves with large curvature in parking scenarios. An example of parking-oriented cost function is presented in Figure 7. Based on the Reeds-Shepp curve, these curve and straight segments can be clearly classified by curvature. Therefore, different cost functions can be implemented in straight and curve segments, respectively.

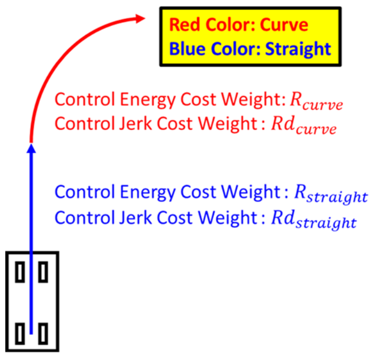
Red Color: Curve
Blue Color: Straight
Control Energy Cost Weight: R_{curve}
Control Jerk Cost Weight: Rd_{curve}
Control Energy Cost Weight: $R_{straight}$
Control Jerk Cost Weight: $Rd_{straight}$

FIGURE 7. An example of parking-oriented cost function.

Moreover, the optimization problem can be formulated based on (19), so the optimization problem is defined as follows:

$$\min_u J_{MPC}. \quad (23)$$

Subject to the state space model equation in (3), and the constraints are defined as follows:

$$\begin{aligned} z_0 &= z_{v,0}, & (24) \\ \left|\delta_{k+1} - \delta_k\right| &\leq \Delta\delta_{max} * dt, & (25) \\ \left|\delta_0 - \delta_{last}\right| &\leq \Delta\delta_{max} * dt, & (26) \\ \left|\delta_k\right| &\leq \delta_{max}, & (27) \\ \left|a_k\right| &\leq a_{max}, & (28) \\ \left|v_k\right| &\leq v_{max}, & (29) \end{aligned}$$

where δ_{last} indicates the last steering command.

(24) indicates that the initial state of MPC should be equal to the vehicle state for each MPC execution. (25) indicates that the steering speed constraints. Also, the steering speed

constraints should be obeyed between the last steering command and the first steering output in (26). (27)-(29) present the max values of steering angle, acceleration, and speed constraints of the proposed vehicle controller.

With the linear state space model in (19) and the tracking problem formulation in this section, the implemented optimizer can result in the optimal control sequence within the prediction horizon T_p for vehicle control.

D. SIMULTANEOUS COMPUTATIONAL AND CONTROL FRAMEWORK

This section is composed of two elements, multi-thread framework and execution time compensation, to deal with the real-time control problems. Because the MPC computational load of the simultaneous speed and steering control is too heavy, all the process in [34] is hard to be executed with the control period T matching real-time requirements. In addition, for practical consideration, the longer control period T leads to the larger control jerk so that the vehicle actuator tracks the next desired control command with longer delay time. For this reason, the MPC computation and control command sending are required to be executed simultaneously based on multi-thread framework. However, resulting execution time delay of MPC exists in this process. Therefore, execution time compensation is implemented to deal with this side effect.

1) MULTI-THREAD FRAMEWORK

To achieve real-time control when both steering and speed controls are conducted, a framework with two threads is proposed in this paper.

In [34] and [37], the original coding framework is presented in Figure 8(a). The MPC part is executed to predict the prediction horizon T_p of control inputs and states in periods of the control period T. Then, the process updates the control sequence and publishes the current control input to the autonomous vehicle in periods of the control period T. In this kind of framework, the execution time of MPC should be limited to the control period T. However, once the computational load is too heavy, the control period T is difficult for real-time control.

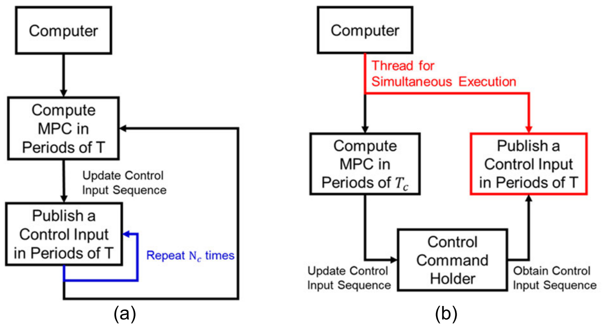
Computer
Compute MPC in Periods of T
Update Control Input Sequence
Publish a Control Input in Periods of T
Repeat N_c times
(a)
Computer
Thread for Simultaneous Execution
Compute MPC in Periods of T_c
Publish a Control Input in Periods of T
Control Command Holder
Update Control Input Sequence
Obtain Control Input Sequence
(b)

FIGURE 8. The flowchart of the coding frameworks. (a) The original framework. (b) Multi-thread framework.

Therefore, multi-thread framework is developed to work around this problem. As shown in Figure 8(b), multiple threads are implemented to conduct MPC computation and control input sending simultaneously. Each time the MPC finishes an execution, the resulting control input sequence will be updated to the global control command holder. Then, the other thread publishes the current control input in periods of the control period T.

Based on multi-thread framework, the execution time limitation of MPC can be expanded up to the control horizon T_c. Moreover, the control inputs within the control horizon T_c can be utilized. In other words, the control period T in the proposed vehicle controller can be $\frac{1}{N_c}$ times than the original framework so that controllers with heavy computational load can be executed for real-time control.

2) EXECUTION TIME COMPENSATION

Moreover, a side effect of multi-thread framework occurs due to the MPC execution time. Therefore, the execution time should be compensated for the vehicle controller. To work around this problem, the concept of the time delay motion prediction in [22] is taken as the reference to propose the execution time compensation.

Figure 9 shows an example. For a starter, the control period is set as T and the execution time of MPC is set as 3T. For the original process in [34], at State 0, the MPC predicts the next state and calculate the sequence of the control input and trajectory for State 1. However, the MPC execution time may not be within the control period T in multi-thread framework. In this case, the results of the execution at State 0 will be received at State 3. Apparently, the received control input sequence has 2-period delay by State 3. To overcome this problem, 2-period delay should be compensated in the proposed vehicle controller. In other words, for the MPC execution at State 0, the proposed vehicle controller should take State 3 as the reference state to derive the corresponding control input sequence.

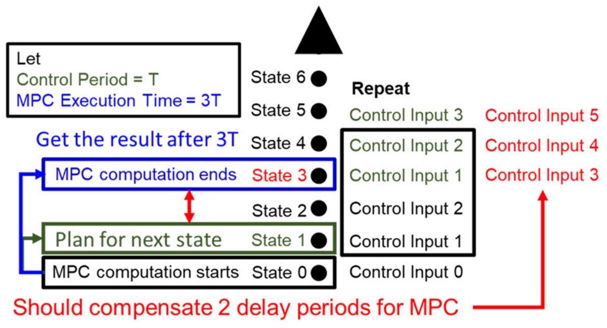

FIGURE 9. An example of execution time compensation.

To achieve execution time compensation, the time-varying motion prediction in [34] is implemented. The state translations of the motion prediction are described as follows:

$$\begin{cases} x_{k+1} = x_k + v_k * \cos(\theta_k) * T \\ y_{k+1} = y_k + v_k * \sin(\theta_k) * T \\ v_{k+1} = v_k + a_k * T \\ \theta_{k+1} = \theta_k + \dfrac{v_k}{WB} * \tan(\delta_k) * T, \end{cases} \tag{30}$$

where $(x_k, y_k, v_k, \theta_k)$ is the current state of vehicle, (a_k, δ_k) is the current control input for the MPC controller, T is control period, and WB is the wheelbase length of vehicle. Then the required iteration number $N_{compensation}$ for execution time compensation is defined as follows:

$$N_{compensation} = \left\lceil \frac{t_{execution}}{T} \right\rceil, \tag{31}$$

where $t_{execution}$ is the last execution time of MPC controller, and T is the control period.

Moreover, if the signal delay t_{signal} in practice is considered, (31) should be revised as follows:

$$N_{compensation} = \left\lceil \frac{t_{execution} + t_{signal}}{T} \right\rceil, \tag{32}$$

Based on (30) and (31), the compensated state $z_{k+N_{compensation}}$ can be derived for the MPC controller. Compared to the motion prediction proposed in [22], the implemented time-varying prediction utilizes the calculated control inputs instead of only the current state to predict the future state. Therefore, the prediction in this paper is more accurate so that the performance of the controller can be better.

E. STEERING COMMAND SMOOTHER

The steering command smoother is proposed to prevent steering commands from damping in high frequency. Based on this function, the smoothed command sequence should have the same effect as the original and match the steering mechanism.

In autonomous parking scenarios, the steering angle usually changes dramatically and frequently. Also, the optimal solution of steering command may contain some damping to track more accurately. In aspect of practice, these damping commands are difficult for steering actuator to track and even may hurt the steering actuator. Therefore, steering command smoother is proposed to cancel the damping of steering results without the steering speed change of the original command sequence.

Steering command smoother is detailed as follows. First of all, to avoid huge difference with the last steering command, the last steering angle δ_{last} is also considered to calculate the average steering angle δ_{avg} with the new steering sequence as follows:

$$\delta_{avg} = \frac{\delta_{last} + \sum_{i=0}^{N_p-1} \delta_i}{N_p + 1}. \tag{33}$$

Then, the new steering speed $\Delta\delta'$ in new steering sequence is derived based on δ_{avg} ad δ_{last} as follows:

$$\Delta\delta' = \frac{2 * (\delta_{avg} - \delta_{last})}{N_p}. \tag{34}$$

Finally, the new steering sequence δ' is obtained as follows:

$$\delta'_i = \delta_{last} + (i + 1) * \Delta\delta', \tag{35}$$

for i = 0, 1, 2 . . .$N_p - 1$.

With the proposed steering smoother, the damping steering control of the original sequence is solved by the constant steering speed $\Delta\delta'$. Moreover, the steering speed of the smooth sequence is the same as the original one because of the same average steering angle δ_{avg}. For this reason, the effects of the smooth sequence are equal to the original and even the smooth sequence can be followed more accurately in practice.

VI. SIMULATION RESULTS AND ANALYSIS

In this section, the motion planner comparison to [10] is simulated in parking scenarios first. In advance, the individual effects of the technical elements in the proposed vehicle controller are discussed by simulation tests.

To make the simulation results correspond to experiments, the parameters are identical based on Chrysler Pacifica which is the plant in this paper. The dimension of Chrysler Pacifica are presented in Table 1.

TABLE 1. The dimension of Chrysler Pacifica.

Parameters	Value
Vehicle Length (L)	5.170 m
Vehicle Width (W)	2.020 m
Side Mirror Width (SMW)	0.190 m
Rear Wheel to Back (RB)	0.935 m
Wheel Base (WB)	3.089 m

When it comes to the proposed motion planner, the parameter setups are presented in Table 2. The minimum turning radius R_{turn} is determined as 8.0 m. The search radius R_{search} of $Tree_{start}$ and $Tree_{goal}$ is set as 8.0 m according to the scenario requirements. Collision buffer $Buffer$ which is defined for uncertainties and collision checking resolution is 0.05 m. Collision disk radius $R_{collision}$ is derived based on the formulation in [10]. Finally, to prevent from falling into local minimum, state space expansion is set the same as R_{search} for exploration.

TABLE 2. The parameters of the proposed motion planner.

Parameters	Value
Minimum Turning Radius (R_{turn})	8.0 m
Search Radius (R_{search})	8.0 m
Collision Buffer ($Buffer$)	0.05 m
Collision Disk Radius ($R_{collision}$)	1.25 m
State Space Expansion	8.0 m

The parameter setups of the proposed vehicle controller are presented in Table 3. These parameters are designed for

TABLE 3. The parameters of the proposed controller.

Parameters	Value
Control Period (T)	0.05 s
Control Horizon ($T_c = 10T$)	0.5 s
Prediction Horizon ($T_p = 10T$)	0.5 s
Control Cost at Straight ($R_{straight}$)	diag [0.01, 0.01]
Control Cost at Curve (R_{curve})	diag [0.01, 0.00]
Control Jerk Cost at Straight ($Rd_{straight}$)	diag [0.01, 0.00]
Control Jerk Cost at Curve (Rd_{curve})	diag [0.01, 0.05]
State Cost (Q)	diag [1.5, 1.5, 0.50, 0.75]
Max Magnitude of Steering Angle ($\|\varphi_{max}\|$)	24 degree
Max Steering Speed Magnitude	18 degree/s
Max Magnitude of Speed Magnitude ($\|v_{max}\|$)	5.0 km/h
Max Magnitude of Acceleration (a_{max})	0.5 m/s^2
Target Speed	1.5 km/h

Chrysler Pacifica. For a starter, the periods of the proposed MPC controller are defined. Also, the cost weighting matrices and constraints are detailed in this table. Finally, the target speed of the speed profile is set as 1.5 km/h.

The map and configuration setups of each design parking scenario are presented in Figure 10. The left figure represents the common parking scenarios, and the right figure represents the strict parking scenarios. In addition, the grid size is 1 m by 1 m so that the relative location information can be clearly learned. For the coordinate, the positive x-axis corresponds to the downside of the pictures and the positive y-axis corresponds to the right.

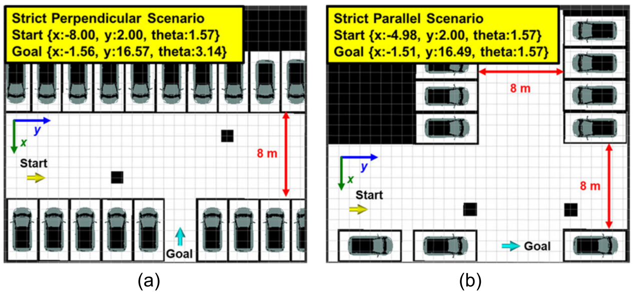
Strict Perpendicular Scenario
Start {x:-8.00, y:2.00, theta:1.57}
Goal {x:-1.56, y:16.57, theta:3.14}
y
x
Start
8 m
Goal
(a)
Strict Parallel Scenario
Start {x:-4.98, y:2.00, theta:1.57}
Goal {x:-1.51, y:16.49, theta:1.57}
8 m
y
x
Start
8 m
Goal
(b)

FIGURE 10. The parking scenarios for simulations. (a) Strict perpendicular Scenario (Scenario 1). (b) Strict parallel scenario (Scenario 2).

Strict perpendicular scenario (Scenario 1) is presented in Figure 10(a). This is a challenging scenario to test the proposed motion planner whether can avoid obstacles and accomplish the perpendicular parking mission. There are two obstacles placed in this map. The first one locates at the place in front of the start configuration, and the second one is placed near the location of the only cusp in common perpendicular scenario of [10]. With the above obstacles, the proposed methods should be able to avoid the first one in the narrow spaces, and use another path pattern to overcome the second one for this perpendicular parking.

Strict parallel scenario (Scenario 2) is presented in Figure 10(b). The design concept is similar to strict perpendicular scenario (Scenario 2), the obstacles are placed to block the start configuration and path pattern for entering the slot respectively. The start configuration is also set at the location with 2 m distance from the down side of the boundary. Moreover, the distance between the obstacles are set as the length of the parking space. As a result, the parking path should overcome not only the challenges mentioned in strict perpendicular scenario (Scenario 1) but also the narrow parking space for the vehicle.

A. MOTION OLANNER COMPARISON IN PARKING SCENARIOS

To validate the proposed motion planner can be able to deal with the parking issues in theory, the simulations in strict perpendicular and parallel scenarios (Scenarios 1 & 2) as [10] were conducted for further comparisons.

The original motion planner performs not well enough for exploration in narrow spaces. For this reason, some failure occurs in the simulation scenarios in [10] when the minimum turning radius increases to 8 m for Chrysler Pacifica. Therefore, the improvement of the proposed motion planner will be demonstrated in the following simulations.

The simulation process is shown in Figure 11. There are four steps in the simulations of this section. Firstly, the proposed method has to read the map which is plotted as a picture previously. Then, the start and goal configurations are set with position (x, y) and heading angle θ. Third, the proposed method grows RRT* trees to explore the environment for one result in 3 seconds each time. If no feasible path is found in 3 seconds, the exploration is regarded as a failure one. Finally, Step 3 is repeated 5 times, and 5 individual results are outcome. For cost comparison, the results from best to worst are presents in red, green, blue, black, and purple in order. As a result, the path geometries and statistical results can correspond to the improvements of the proposed motion planner in aspect of path quality and consistency, and even the exploring efficiency.

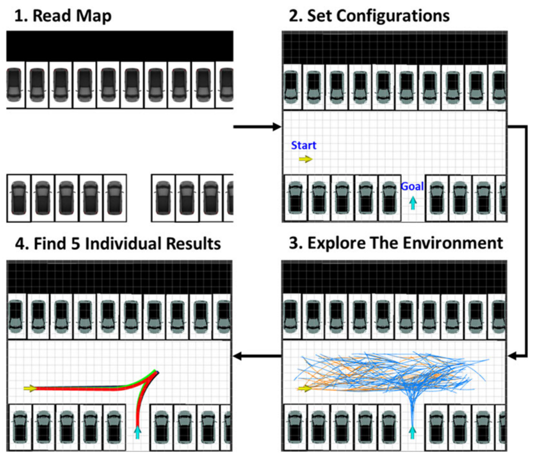

FIGURE 11. The simulation process of the proposed motion planner.

1) PERPENDICULAR PARKING

When it comes to the perpendicular parking simulation tests, the proposed motion planner and vehicle controller are simulated in Strict Perpendicular Scenario (Scenario 1). the five individual path results and cost convergence profiles in both scenarios are shown in Figure 12. Furthermore, the statistical results of cost and cost function elements are recorded in Table 4. Strict perpendicular scenario (Scenario 1) places two obstacles to break the resulting path pattern in common perpendicular scenario in [10]. The stricter environment makes the original motion planner result in an exceptional paths (purple) with 5 cusps as shown in Figure 12(a). Because of the two proposed samplers, the resulting paths of the proposed motion planner require 3 cusps to complete the parking mission consistently. The detail information can be learned from the statistics of cost number in Table 4.

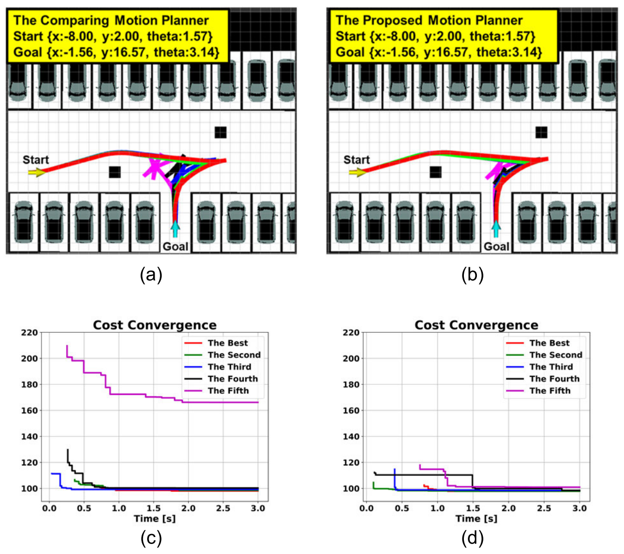

FIGURE 12. The path results of the comparing motion planner (left) and the proposed motion planner (right) in strict perpendicular scenario. (a) and (b) The five individual results. (c) and (d) The cost convergence profiles.

TABLE 4. The statistical results of motion planner simulations in perpendicular parking scenarios.

		[10]		Proposed Planner	
		Mean	Std. Dev.	Mean	Std. Dev.
Strict Perpendicular Scenario (Scenario 1)	Cost	112.44	30.03	98.70	1.30
	Path Length (m)	34.7	3.83	33.26	0.45
	Backward Length (m)	13.4	2.77	12.3	0.32
	Curve Length (m)	21.50	4.96	19.01	0.64
	Cusp Number	4.2	2.68	3	0
	Failure	0		0	

2) PARALLEL PARKING

In aspect of parallel parking simulation tests, both motion planners are conducted in strict parallel scenario (Scenario 2). Also, the five individual path results and cost convergence profiles in both scenarios are shown in Figure 13. Furthermore, the statistical results of cost and cost function elements are recorded in Table 5.

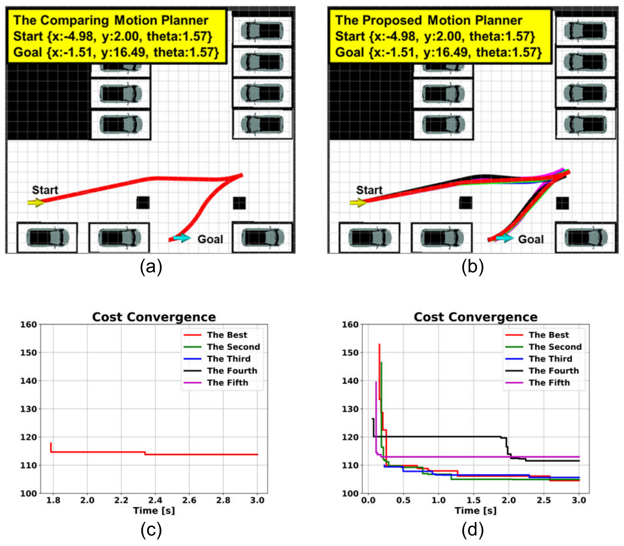

FIGURE 13. The path results of the comparing motion planner (left) and the proposed motion planner (right) in strict parallel scenario. (a) and (b) The five individual results. (c) and (d) The cost convergence profiles.

TABLE 5. The statistical results of motion planner simulations in parallel parking scenarios.

		[10]		Proposed Planner	
		Mean	Std. Dev.	Mean	Std. Dev.
Strict Parallel Scenario (Scenario 2)	Cost	113.79	0.00	107.92	4.00
	Path Length (m)	35.60	0.00	35.40	0.41
	Backward Length (m)	11.80	0.00	11.70	0.21
	Curve Length (m)	20.60	0.00	17.77	2.11
	Cusp Number	5.0	0.00	5.0	0.00
	Failure	4		0	

Strict parallel scenario (Scenario 2) becomes much stricter than common parallel scenario of [10]. The original motion planner has difficulty to generate a feasible path. The only resulting path of the original motion planner is presented in Figure 13(a) and Table 5. As shown in Figure 13(c) and (d), the cost of only resulting path is worse than each results of the proposed motion planner. In advance, the proposed motion planner generates the results with 5 cusps consistently corresponding to the data in Table 5. This phenomenon indicates the proposed motion planner is able to explore the narrow spaces and make frequent motion directional changes around the parking slot. The simulation results in this scenario corresponds to the improvement of the proposed motion planner in narrow space exploration.

B. METHOD EFFECTS OF THE PROPOSED VEHICLE CONTROLLER

To examine the effects of each technical elements in the proposed vehicle controller, all the simulations were conducted to track the designed reference path in common perpendicular scenario (Scenario 1). Moreover, the heading of the start configuration is adjusted to result in more curve segments. To confirm the reference path of simulation is identical, the information of the reference path in Figure 14 is written as a csv profile.

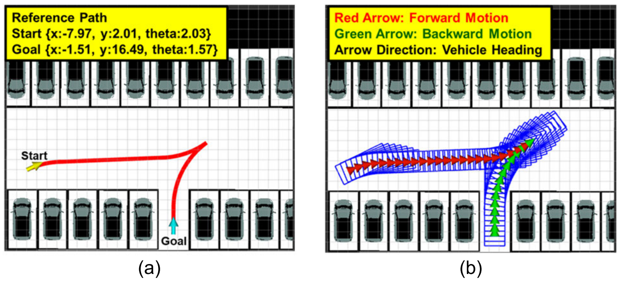
Reference Path
Start {x:-7.97, y:2.01, theta:2.03}
Goal {x:-1.51, y:16.49, theta:1.57}
Start
Goal
(a)
Red Arrow: Forward Motion
Green Arrow: Backward Motion
Arrow Direction: Vehicle Heading
(b)

FIGURE 14. The reference path for the simulation in the section. (a) The reference waypoints. (b) The reference path with vehicle shape models.

The simulation process of the proposed vehicle controller in this section is shown in Figure 15. As shown in step 1, the csv profile of the reference path is read for the comparing and proposed vehicle controllers. For step 2, the reference path information is regarded as the input for both controllers. Finally, the simulation is conducted based on the reference path, and the simulation results are utilized to show the difference between both methods.

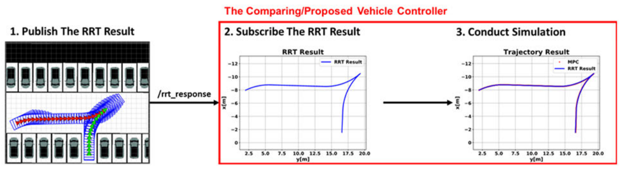
The Comparing/Proposed Vehicle Controller
1. Publish The RRT Result
/rrt_response
2. Subscribe The RRT Result
RRT Result
3. Conduct Simulation
Trajectory Result

FIGURE 15. The simulation process of the proposed vehicle controller.

The following simulations are compared with the results of the proposed vehicle controller in Figure 16, Figure 17 and Table 6. As shown in Figure 16, the results of tracking trajectory, distance and heading error are presented. Also, acceleration and steering angle profiles as control input, and speed and heading profiles as plant responses, are recorded in Figure 17. In advance, the RMSE results in aspect of distance and heading are recorded in Table 6.

1) TRACKING STRATEGY FOR AUTONOMOUS PARKING

In this case, the proposed tracking strategy is compared to the original one in [34] for the validation in advance. The

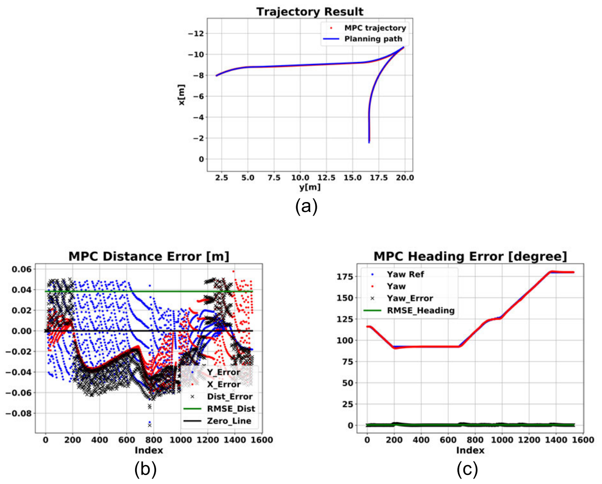

FIGURE 16. The tracking results of the proposed vehicle controller for effect comparison. (a) The tracking trajectory. (b) The distance error. (c) The heading error.

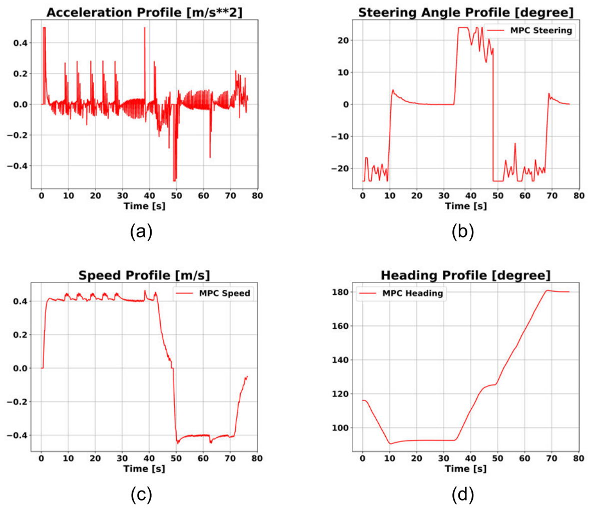

FIGURE 17. The tracking profiles of the proposed vehicle controller for effect comparison. (a) The acceleration profiles. (b) The steering angle profiles. (c) The speed profiles. (d) The heading profile.

TABLE 6. The simulation results of the comparison for effect comparison.

	The Proposed Tracking Strategy
Distance RMSE (m)	0.038
Heading RMSE (degree)	0.523
Max Distance Error (m)	0.092
Max Heading Error (degree)	1.885
Final *x* Error (m)	-0.171
Final *y* Error (m)	0.018
Final Yaw Error (degree)	0.182

simulation results are shown in, Figure 18 and Table 7. For the original tracking strategy, a dramatic tracking error can be learned from Figure 18(a) and Table 2. Due to lack of steering angle initialization, the steering angles at the start point and the cusp are much different from the desired

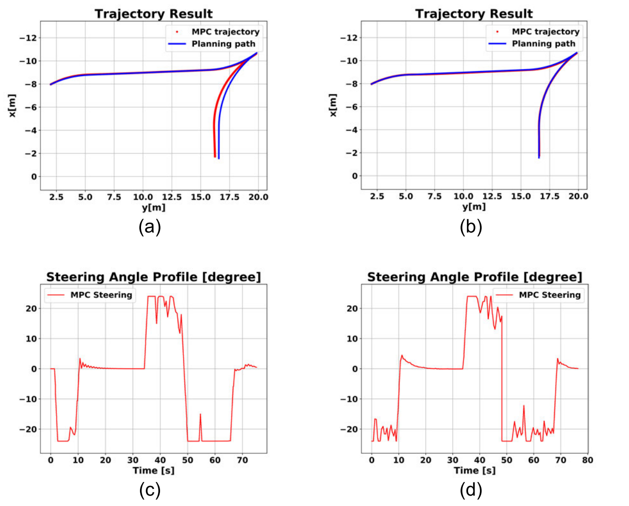

FIGURE 18. The tracking results of comparison for tracking strategy for autonomous parking. The left presents the results with the original tracking strategy in [34]. The right presents the results with the proposed tracking strategy. (a) and (b) The trajectory results. (c) and (d) The steering profiles.

TABLE 7. The simulation results of the comparison for the proposed tracking strategy.

	The Original Tracking Strategy	The Proposed Tracking Strategy
Distance RMSE (m)	0.204	0.038
Heading RMSE (degree)	2.242	0.523
Max Distance Error (m)	0.434	0.092
Max Heading Error (degree)	7.363	1.885
Final x Error (m)	-0.153	-0.171
Final y Error (m)	-0.329	0.018
Final Yaw Error (degree)	1.873	0.182

steering angles. For this reason, the huge lateral tracking errors are caused especially at the cusp. In advance, the difference tracking strategies can be learned from the steering profiles in Figure 18(c) and (d). At about 48 s, the original tracking strategy changes the steering angle from 20° to −24° at some speed in the beginning of the backward motion tracking as shown in Figure 18(c). Therefore, the process of steering angle change results in the largest lateral error up to 0.434 m in the tracking simulation. On the other hand, the proposed tracking strategy stops and conducts the steering angle initialization for each segment. Starting at corresponding steering angle succeeds to improve the tracking performance.

2) PARKING-ORIENTED COST FUNCTION AND CONSTRAINTS

For comparison, three cases with all straight cost weight, all curve cost weight, and the proposed cost weight design, are presented respectively to show the requirement of the cost function design in this paper.

The simulation results with three kinds of cost function design are presented in Figure 19 and Table 8. As usual, MPC controllers used at high speed should not control with

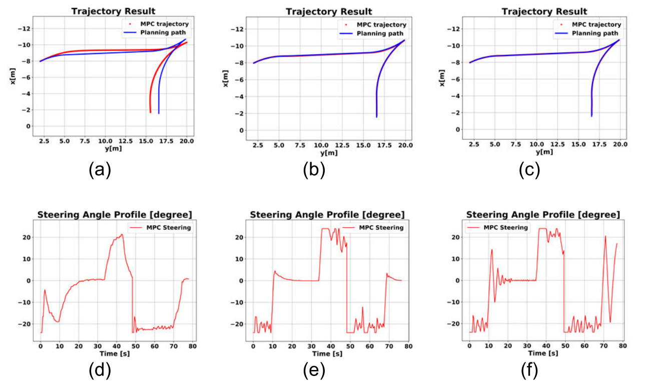

FIGURE 19. The tracking results of comparison for parking-oriented cost function and constraint design. The left presents the results using only cost function at straight. The middle presents the results using the proposed cost function. The right presents the results using only cost function at curve. (a)-(c) The trajectory results. (d)-(f) The steering profiles.

TABLE 8. The simulation results of the comparison for the proposed cost function design.

	Straight cost weight	Proposed cost weight	Curve cost weight
Distance RMSE (m)	0.511	0.038	0.032
Heading RMSE (degree)	6.987	0.523	0.519
Max Distance Error (m)	1.072	0.092	0.076
Max Heading Error (degree)	13.396	1.885	2.094
Final x Error (m)	-0.118	-0.171	-0.176
Final y Error (m)	-1.015	0.018	0.015
Final Yaw Error (degree)	2.933	0.182	0.267

large steering angle for lateral stability. Therefore, the control energy weight is designed to limit the value of steering angle for steering stability. However, the results in parking scenarios may contain huge lateral and heading error as shown in Figure 19(a) because steering angle has difficulty to maintain maximum value as shown in Figure 19(d). On the other hand, if the steering energy weight is set as 0 for the MPC controller, the steering overshoot and damping will occur when dramatic steering changes, such as the steering angle at about 10 s and 70 s in Figure 19(f). The phenomenon may result in the long settling time of steering control. When it comes to practice, the long settling time may cause more lateral error due to the delay of actuator, and even make steering angle diverge more seriously. With above considerations, the proposed vehicle controller applies different control cost weights for straight and curve paths to utilize the advantages of both control cost weights.

3) SIMULTANEOUS COMPUTATION AND CONTROL FRAMEWORK

In this comparison, simulations with longer control periods and the control period with the proposed method were conducted to discuss the importance of the proposed framework.

In actual, the control period of the original framework should be 0.5 s for practical control. However, MPC controller fails to be executed in this control period. Therefore, the control period of the original framework is set as 0.2 s for this comparison. On the other hand, the control period of the proposed framework is 0.05 s. The simulation results are presented in Figure 20 and Table 9. From the tracking results in Figure 20 and Table 9, the much better performance of the proposed framework can be clearly observed. Moreover, the resulting control sequences of the comparing vehicle controller are rougher than the proposed one as shown in Figure 20(c)-(f). For this reason, the tracking performance of the proposed vehicle controller is much better in Table 9.

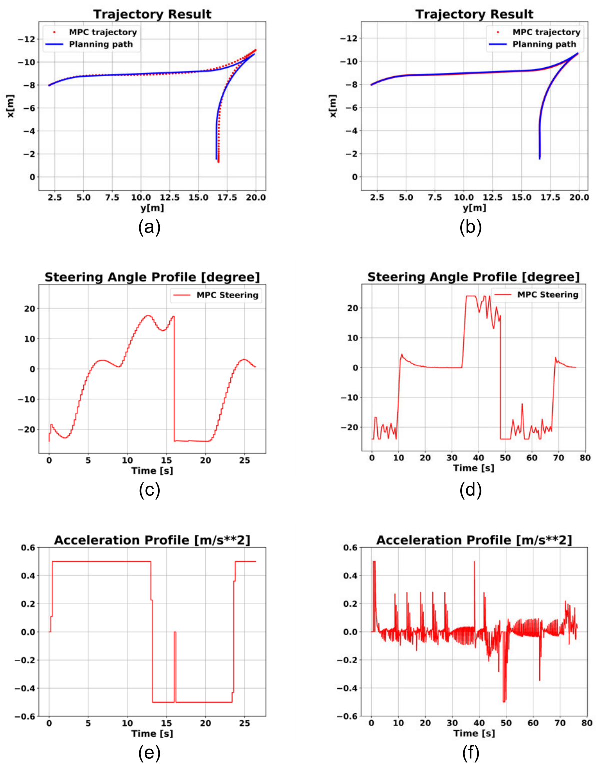

FIGURE 20. **The tracking results of comparison for simultaneous computation and control framework. The left presents the results with the comparing control period = 0.2 s. The right presents the results with the proposed control period = 0.05 s. (a) and (b) The trajectory results. (c) and (d) The steering profiles. (e) and (f) The acceleration profiles.**

TABLE 9. **The simulation results of the comparison for simultaneous computation and control framework.**

	Control Period 0.2 s	Control Period 0.05 s
Distance RMSE (m)	0.165	0.038
Heading RMSE (degree)	2.793	0.523
Max Distance Error (m)	0.363	0.092
Max Heading Error (degree)	7.376	1.885
Final x Error (m)	0.264	-0.171
Final y Error (m)	0.176	0.018
Final Yaw Error (degree)	-1.348	0.182

In aspect of discrete control, the longer controller period leads to the results with huger error. Moreover, in aspect of practice, the longer controller period indicates the larger gap of adjacent control command. This phenomenon results in more delay time of actuators to track control command. Therefore, the resulting error due to long control period in practice must be more serious than simulations. Simultaneous computation and control Framework is able to decrease the control period for the controller with heavy computational load so that the control can be executed for real-time control.

4) SIMULTANEOUS COMPUTATION AND CONTROL FRAMEWORK

In this case, the simulation results with steering command smoother is compared to demonstrate the effects aforementioned.

The simulation results are presented in Figure 21 and Table 10. When it comes to tracking performance, there are no big difference between both methods as shown in Table 10. In aspect of the steering profiles, some damping steering exists in Figure 21(c) to make steering control more accurate. However, these trembles make vehicle actuator hard to track, and even cause damages to the steering mechanism. On the other hand, the steering profile in Figure 21(d) gets rid of steering trembles and maintains the maximum steering speed. Therefore, Steering Command Smoother is able to cancel steering trembles and maintain the tracking performance as shown in Table 10.

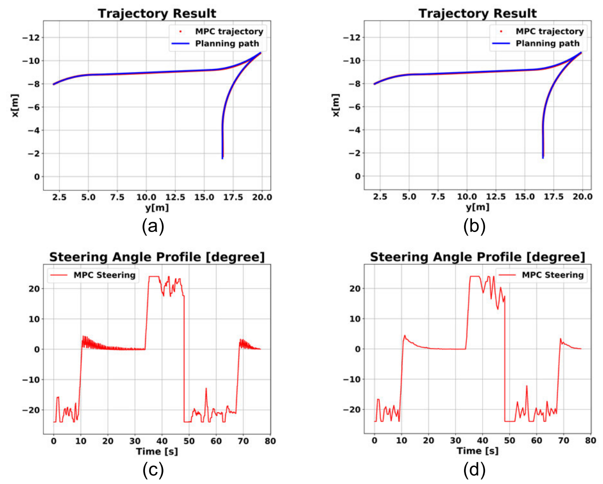

FIGURE 21. **The tracking results of comparison for steering command smoother. The left presents the results without steering command smoother. The right presents the results with steering command smoother. (a) and (b) The trajectory results. (c) and (d) The steering profiles.**

TABLE 10. **The simulation results of the comparison for steering command smoother.**

	Without Smoother	With Smoother
Distance RMSE (m)	0.039	0.038
Heading RMSE (degree)	0.469	0.523
Max Distance Error (m)	0.087	0.092
Max Heading Error (degree)	1.673	1.885
Final x Error (m)	-0.169	-0.171
Final y Error (m)	0.028	0.018
Final Yaw Error (degree)	0.196	0.182

VII. EXPERIMENTAL RESULTS AND ANALYSIS

This section details how to make the proposed motion planner and vehicle controller work in practice. Moreover, the experiments are conducted in various parking scenarios in everyday life to validate that the proposed motion planner and vehicle controller can work well in practice.

The system structure of the autonomous parking system is presented in Figure 22. The proposed motion planner requires map information and start and goal configurations as input. As a result, the proposed motion planner subscribes the car state from the NDT localization of the autonomous vehicle, map information from LiDAR perception, and the desired goal configuration from the default setups. Then, the proposed motion planner results in a parking path and sends it to the proposed vehicle tracking as reference path for vehicle path tracking. In advance, the proposed vehicle controller subscribes the vehicle state (x, y, θ) and vehicle state (δ) as feedbacks and publishes the speed and steering command to control the autonomous vehicle continually. Additionally, the signal delay is set as 3 seconds for the autonomous vehicle.

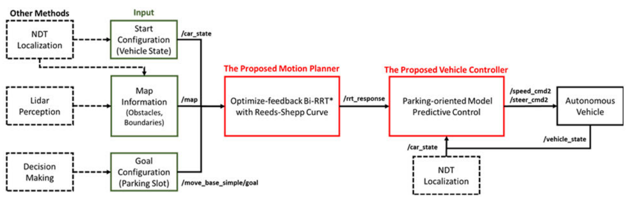
Other Methods
Input
NDT Localization
Start Configuration (Vehicle State)
/car_state
Lidar Perception
Map Information (Obstacles, Boundaries)
/map
Decision Making
Goal Configuration (Parking Slot)
/move_base_simple/goal
The Proposed Motion Planner
Optimize-feedback Bi-RRT* with Reeds-Shepp Curve
/rrt_response
The Proposed Vehicle Controller
Parking-oriented Model Predictive Control
/speed_cmd2
/steer_cmd2
Autonomous Vehicle
/vehicle_state
/car_state
NDT Localization

FIGURE 22. **The system structure of the experiments for autonomous parking.**

The perpendicular parking trials are composed of common perpendicular left trial and strict perpendicular trial, as shown in Figure 23. Threes perpendicular parking trial aims to park in the target slot which is 2.30 m by 5.50 m.

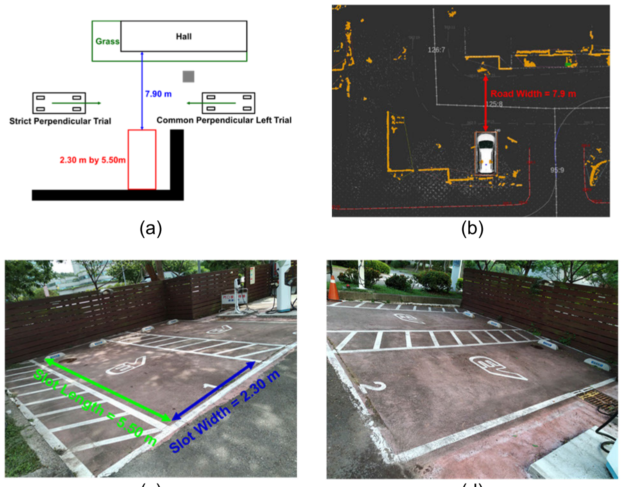

FIGURE 23. **The environment of perpendicular parking trials. (a) The perpendicular trials. (b) The environment shown in system. (c) and (d) The target perpendicular parking slot.**

Common perpendicular left trials contain the freest spaces for the resulting path among the three trials. The resulting path is not limited by the walls of the hall, but the resulting path crosses on the grass. So, the depression of the grass boundaries should be overcome by the proposed vehicle controller.

Strict perpendicular trials contain an obstacle blocking around the cusp location in common perpendicular right trials. Therefore, the proposed motion planner should plan a parking path to avoid the obstacle and succeed to park in the target slot. Moreover, the proposed vehicle controller has to perform pretty accurate tracking to prevent the vehicle falling into collision and accomplish the parking mission.

The parallel parking trials consist of common parallel back-left trials and common parallel front-left trials, as shown in Figure 24. Both of the trials are named from the relative position of the target slot. The size of the target slot is 2.40 m by 5.425 m. In addition, there is 0.92 m distance between the target slot and the front slot. Both trials aim to park in the target slot without any collision to the vehicle and the cone in adjacent slots.

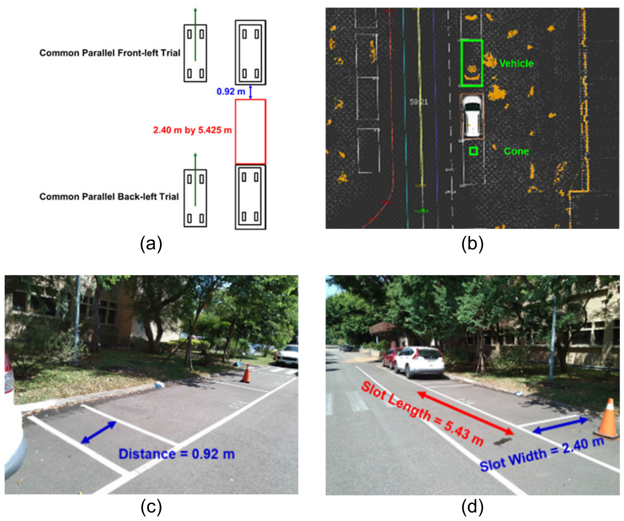

FIGURE 24. **The environment of parallel parking trials. (a) The parallel trials. (b) The environment shown in system. (c) and (d) The target parallel parking slot.**

A. SIGNAL DELAY FOR PRACTICAL CONSIDERARION

To accomplish the parking function in practice, signal delay [22] should be considered in the proposed vehicle controller using (32) with $t_{signal} = 0.3\ s$. The following experiments were conducted to compare the difference in advance.

The experimental results, such as planning paths, tracking trajectories, distance error profiles, and steering profiles, are presented in Figure 25. In advance, the statistical results are recorded in Table 11. From Figure 25(a) and (b), both the planning paths in common perpendicular right trials consist of one forward motion segment and a backward one respectively. The tracking performance can be compared by vehicle

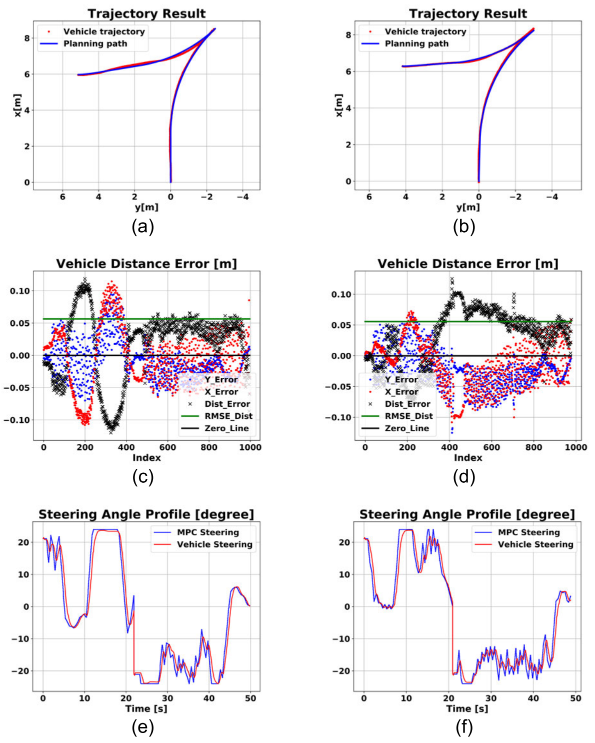

FIGURE 25. The experimental results with signal time delay (Right) or not (Left). (a) and (b) The tracking results. (c) and (d) The distance error profiles. (e) and (f) The steering profiles.

TABLE 11. The statistical results of signal delay experiments.

	Without	With
Distance RMSE (m)	0.056	0.056
Heading RMSE (degree)	2.025	1.107
Max Distance Error (m)	0.120	0.125
Max Heading Error (degree)	6.166	2.716
Final x Error (m)	0.029	-0.059
Final y Error (m)	0.019	0.004
Final Yaw Error (degree)	0.047	-1.448

trajectories. The vehicle trajectory in Figure 25(a) damps around the reference path also corresponding to the distance error information with sign in Figure 25(c). The reason is that there is always signal delay in the steering control. In other words, the response of the steering actuator is always late for the current state. Also, the signal delay has a direct impact on heading tracking described in (3). This impact can be clearly learned from large heading error in Table 11. Compared to the results without signal delay consideration, the proposed vehicle controller takes signal delay into account to deal with the practical problem and improve the performance of heading error.

B. PERPENDICULAR PARKING

For perpendicular parking, there are three kinds of trials corresponding to some difficulties individually for experiments. The simulation and experimental results and analysis in common perpendicular left trials, common perpendicular right trials, and strict perpendicular trials, are going to be discussed.

1) COMMON PERPENDICULAR LEFT TRIALS

The first trial is common perpendicular left trial. In this trial, the vehicle crosses on the grass. Therefore, the proposed methods should overcome the depression and grass texture to complete the parking mission.

The planning path of common perpendicular left trial 1 is presented in Figure 26. Moreover, the parking process is presented in Figure 27. At first, the steering angle was turned to the max right steering and the first segment was tracked as shown in Figure 27(a)-(d). At the location around the only cusp in Figure 27(d), the font rears of the vehicle crossed into the depression of the cusp. For this reason, the vehicle body became oblique so that the localization results were impacted. The most important of all, the proposed vehicle controller could let the vehicle escape the depression and complete the parking mission finally as shown in Figure 27(e)-(h).

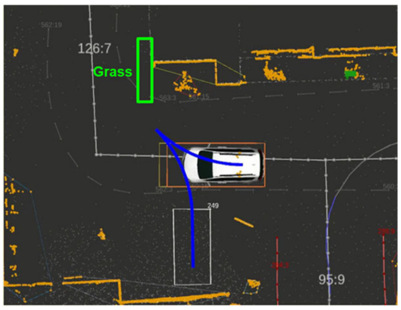

FIGURE 26. Common perpendicular left trial 1 with the planning parking path.

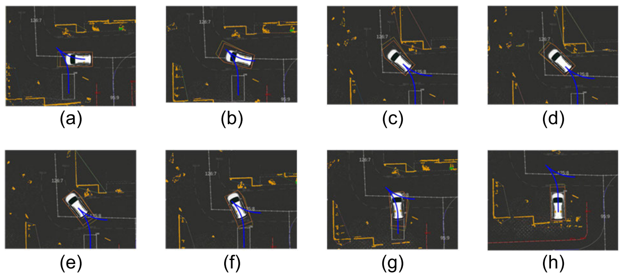

FIGURE 27. The whole process of common perpendicular left trial 1. (a) to (h) The images of the record video with equal time interval in order.

Moreover, the comparison of simulation and experimental results, such as trajectory, distance error, and heading error, are presented in Figure 28. The simulation was conducted in an ideal condition. Therefore, without the impact of depression and practical factors, the tracking performance was quite accurate. The RMSE distance error is 0.031 m and the RMSE heading error is 0.362 degree. When it comes to the real-world experiment trial, the performance is affected by the aforementioned factors. The impact of the grass depression can be observed from the cusp location in Figure 28(b).

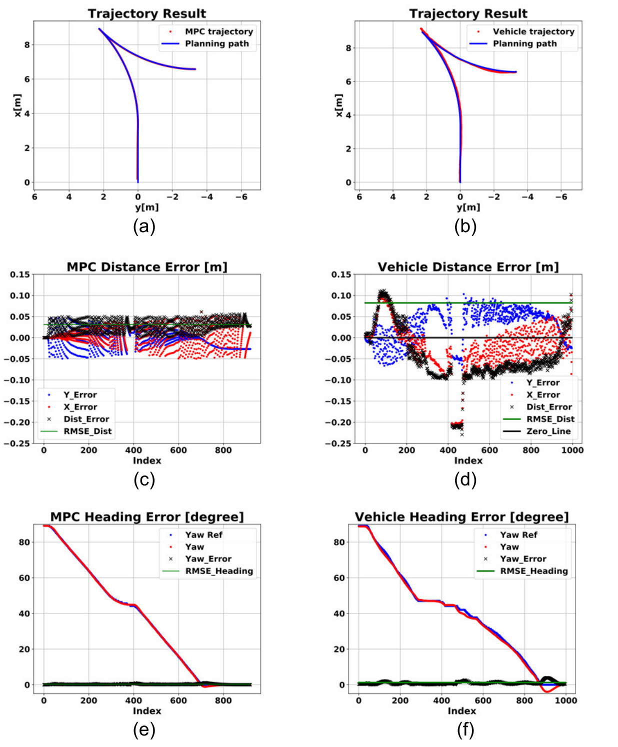

FIGURE 28. The simulation (left) and experimental (right) tracking results of common perpendicular left trial 1. (a) and (b) The trajectory results. (c) and (d) The distance error profiles. (e) and (f) The heading error profiles.

Furthermore, the phenomenon can correspond to the 400th to 450th index in Figure 28(d) which the distance error suddenly jumps to about -0.020 m. As the vehicle distance error shown, the following indexes indicate that the proposed vehicle control is able to deal with the impact of the grass depression and finish the parking mission.

Also, the steering and speed control profiles are recorded in Figure 29. Because of the signal delay consideration,

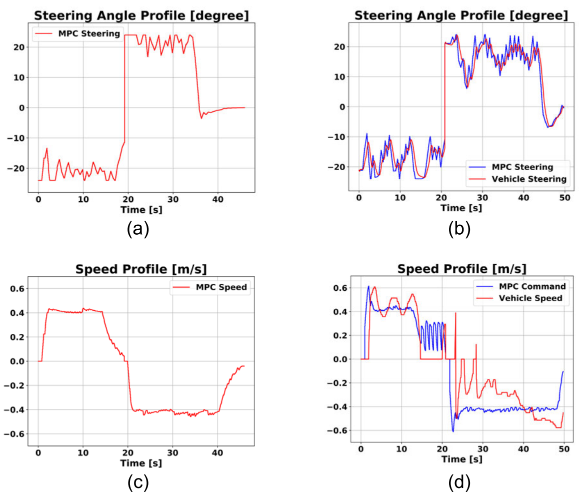

FIGURE 29. The simulation (left) and experimental (right) tracking profiles of common perpendicular left trial 1. (a) and (b) The steering angle profiles. (c) and (d) The speed profiles.

the steering control command in Figure 29(b) contains some damping to make the actual steering control get rid of signal delay. In compared to both steering profile in Figure 29(a) and (b), both control patterns are similar. However, the vehicle control with steering feedback may be affected by other system factors in practice, such as actuator properties and controller. Therefore, the vehicle steering in Figure 29(b) still has some difference between Figure 29(a). On the other hand, the vehicle system is not sensitive to low speed. For this reason, the proposed vehicle controller just publishes the speed commands without speed feedback to the vehicle.

Because different start configurations lead to different planning paths, two additional trials were conducted to demonstrate the ability of the proposed parking system for different parking condition in this trial as shown in Figure 30.

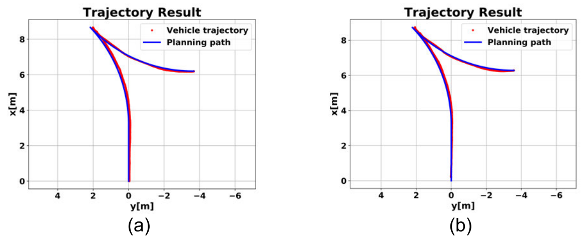

FIGURE 30. The additional tracking trials in common perpendicular left trial. (a) Trial 2. (b) Trial 3.

Finally, the statistical results of simulation and experimental trials in common perpendicular left trials are recorded in Table 12. Regarding to the results of the three trials, it denotes that the proposed parking system has a stable performance in common perpendicular left trials.

TABLE 12. The statistical results of common perpendicular left trials.

	Simulation	Trial 1	Trial 2	Trial 3
Distance RMSE (m)	0.031	0.082	0.084	0.072
Heading RMSE (degree)	0.362	1.262	1.193	1.180
Max Distance Error (m)	0.061	0.229	0.131	0.118
Max Heading Error (degree)	1.125	3.957	2.486	3.373
Final x Error (m)	0.188	0.034	-0.002	0.189
Final y Error (m)	0.027	0.025	-0.045	0.009
Final Yaw Error (degree)	-0.033	0.440	-0.048	-0.141

2) STRICT PERPENDICULAR TRIALS

In strict perpendicular trials, an obstacle is placed at the cusp location in common perpendicular right trials to correspond to the condition of strict perpendicular scenario.

The planning path in this trial is presented in Figure 31. The whole parking process is recorded in Figure 32. The location of the obstacle can be learned from Figure 31. To avoid this obstacle, the proposed motion planner generated a path containing three motion directional changes to overcome this trial. When it comes to the whole parking process, the vehicle approached the obstacle closely two times as shown in

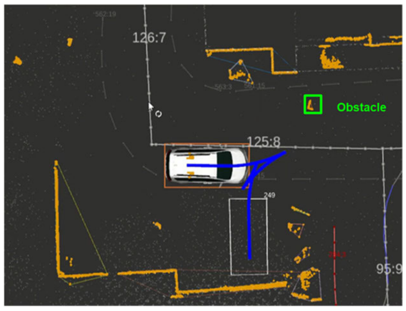

FIGURE 31. Strict perpendicular trial 1 with the planning parking path.

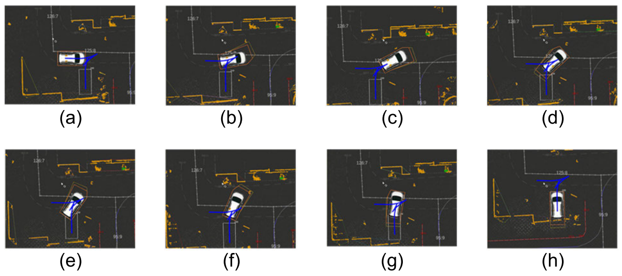

FIGURE 32. The whole process of strict perpendicular trial 1. (a) to (h) The images of the record video with equal time interval in order.

Figure 32(c) and (f). Apparently, the requirement of control accuracy is stricter than common perpendicular right trials. The proposed parking system has enough ability to deal with the challenges of the trials.

The tracking results of the simulation and experimental trial are presented in Figure 33. Both trajectory results in Figure 33(a) and (b) are highly similar to the planning path. It indicates the good tracking performance which can be learned from Figure 33(c)-(f) in advance.

The trajectory results of another two trial are presented in Figure 34. Also, the statistical results of the simulation and three trials are recorded in Table 13. The stable performance of the proposed parking system can be clearly from the statistical results.

TABLE 13. The statistical results of strict perpendicular trials.

	Simulation	Trial 1	Trial 2	Trial 3
Distance RMSE (m)	0.040	0.057	0.063	0.053
Heading RMSE (degree)	0.467	1.026	1.127	1.229
Max Distance Error (m)	0.077	0.155	0.165	0.118
Max Heading Error (degree)	1.725	2.785	3.118	3.198
Final x Error (m)	0.177	0.040	0.028	0.013
Final y Error (m)	-0.021	0.061	0.082	0.064
Final Yaw Error (degree)	0.194	0.300	0.780	0.463

C. PARALLEL PARKING

For parallel parking, there are two common trials in everyday life for experiments, common parallel back-left trials and common parallel front-left trials. Moreover, both the adjacent parking slots were occupied by a vehicle and a cone so that the autonomous vehicle should move in narrow space.

1) COMMON PARALLEL BACK-LEFT TRIALS

In common parallel back-left trials, the autonomous vehicle started from the back-left side of the parking slot to accomplish the parallel parking.

The panning path of the common parallel back-left trial 1 is presented in Figure 35. The whole parking process is presented in Figure 36. At first, the vehicle went forward to the first cusp as shown in Figure 36(a)-(d). Then, the vehicle moved backward to the parking slot as shown in Figure 36(d)-(f). Finally, the vehicle adjusted the position to the slot between the adjacent vehicle and the cone as shown in Figure 36(f)-(h).

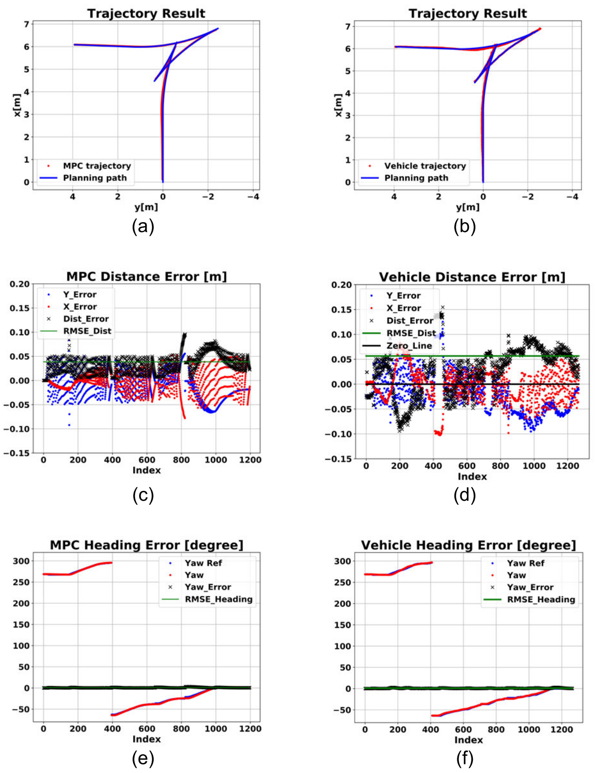

FIGURE 33. The simulation (left) and experimental (right) tracking results of strict perpendicular trial 1. (a) and (b) The trajectory results. (c) and (d) The distance error profiles. (e) and (f) The heading error profiles.

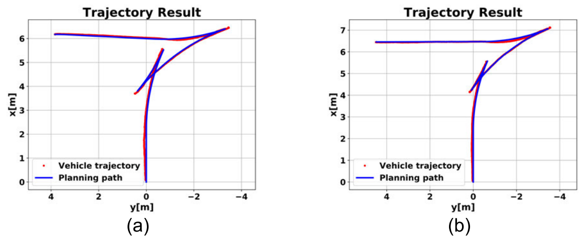

FIGURE 34. The additional tracking trials in strict perpendicular trial. (a) Trial 2. (b) Trial 3.

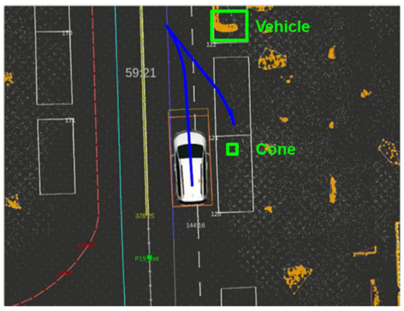

FIGURE 35. Common Parallel Back-left Trial 1 with the planning parking path.

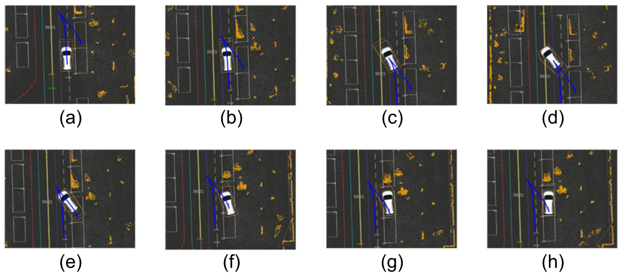

FIGURE 36. The whole process of common parallel back-left trial 1. (a) to (h) The images of the record video with equal time interval in order.

In advance, the tracking results of the simulation and the experiments are presented in Figure 37. The control profiles are presented in Figure 38. For simulation, the accurate tracking performance can be learned from the

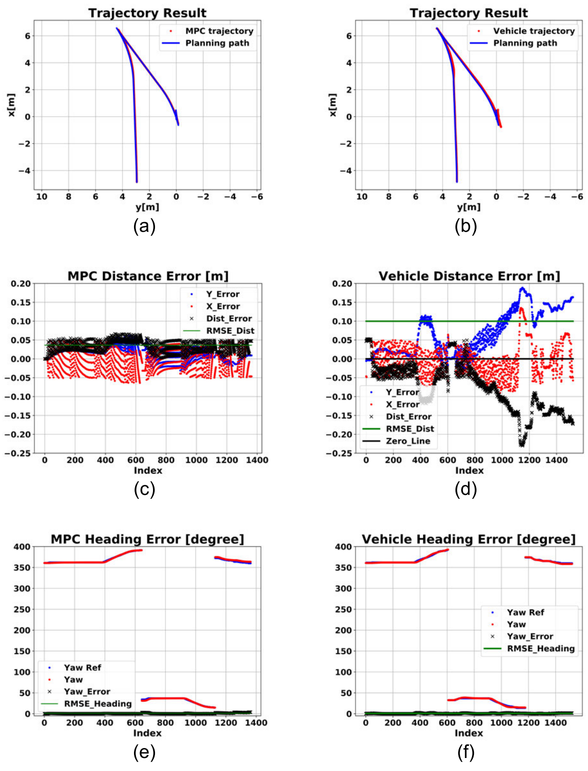

FIGURE 37. The simulation (left) and experimental (right) tracking results of common parallel back-left trial 1. (a) and (b) The trajectory results. (c) and (d) The distance error profiles. (e) and (f) The heading error profiles.

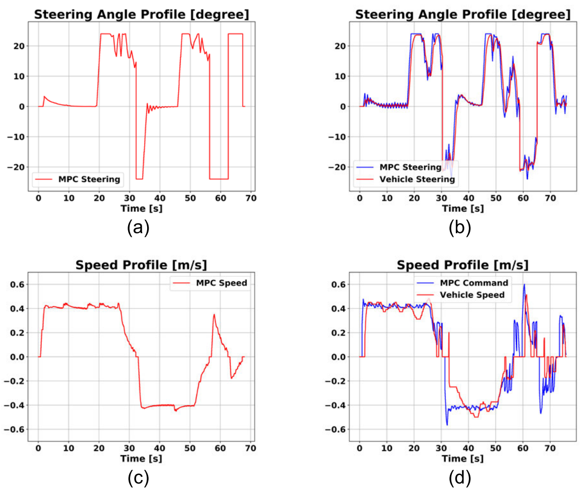

FIGURE 38. The simulation (left) and experimental (right) tracking profiles of common parallel back-left trial 1. (a) and (b) The steering angle profiles. (c) and (d) The speed profiles.

Figure 37(a), (c), and (e). Moreover, the three motion directional changes occurred around 32, 56, and 62 s which can be observed from Figure 38(a) and (c). On the other hand, four motion directional changes occurred in this experimental trial as shown in Figure 38(d). This phenomenon indicates that the vehicle went over the destination when tracking the third segment. As a result, the proposed vehicle controller adjusted the position by additional forward motion at about 73 s as shown in Figure 38(d). Due to lack of speed feedback, the distance error may be accumulated by short movements in the narrow parking space after about the 1200th index in Figure 37(d).

Moreover, the two additional trials are presented in Figure 39. Also, the statistical results of the simulation and trials are recorded in Table 14. The tracking performance of the proposed parking system are still good enough to work around this kind of parallel parking trial.

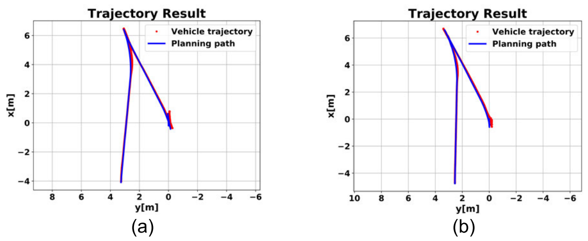

FIGURE 39. The additional tracking trials in common parallel back-left trial. (a) Trial 2. (b) Trial 3.

2) COMMON PARALLEL FRONT-LEFT TRIALS

In common parallel front-left trials, the autonomous vehicle started from the front-left side of the parking slot to achieve the parallel parking mission.

The planning path of common parallel front-left trial 1 is presented in Figure 40. The whole parking process us presented in Figure 41. The autonomous vehicle started with

TABLE 14. The statistical results of common parallel back-left trials.

	Simulation	Trial 1	Trial 2	Trial 3
Distance RMSE (m)	0.036	0.100	0.088	0.102
Heading RMSE (degree)	1.221	0.977	1.275	0.959
Max Distance Error (m)	0.066	0.230	0.196	0.219
Max Heading Error (degree)	3.744	2.632	3.833	3.591
Final x Error (m)	-0.053	0.057	-0.088	-0.056
Final y Error (m)	-0.009	-0.163	0.130	-0.191
Final Yaw Error (degree)	3.744	-1.456	1.648	0.956

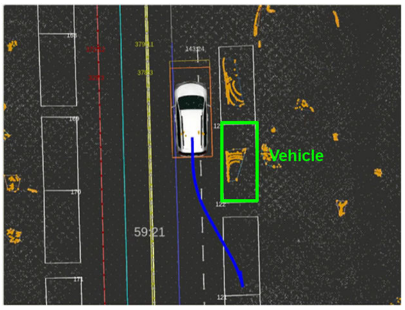

FIGURE 40. Common parallel front-left trial 1 with the planning parking path.

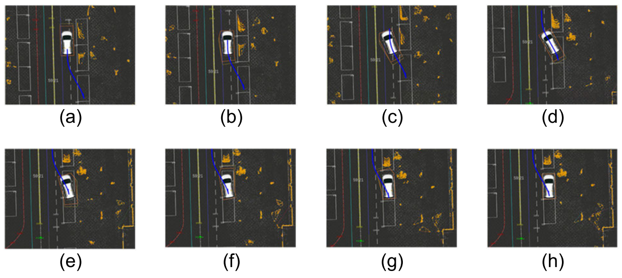

FIGURE 41. The whole process of common parallel front-left trial 1. (a) to (h) The images of the record video with equal time interval in order.

backward motion to the parking slot in this trial as shown in Figure 41(a)-(e). Finally, the vehicle adjusted the position and heading in the narrow spaces around the slot as shown in Figure 41(f)-(h).

Moreover, the simulation and experimental tracking results are presented in Figure 42. The tracking path consists of three path segments, two backward motion and one forward motion. In this trial, the vehicle conducted the same motion directional changes as simulation. However, as shown in Figure 42(d), the distance error is still accumulated in the narrow spaces around the parking slot as common parallel front-left trial 1.

In addition, the trajectory results of the two more trials are presented in Figure 43. The statistical results of the simulation and three trials are presented in Table 15. In this kind of trials, there may be two possible planning path patterns. Because the start configuration of the trials 2 and 3 are in back of trial 1, the first segments of trials 2 and 3 are forward segments to adjust heading angle. The following process are similar to trial 1. The tracking performance of the proposed

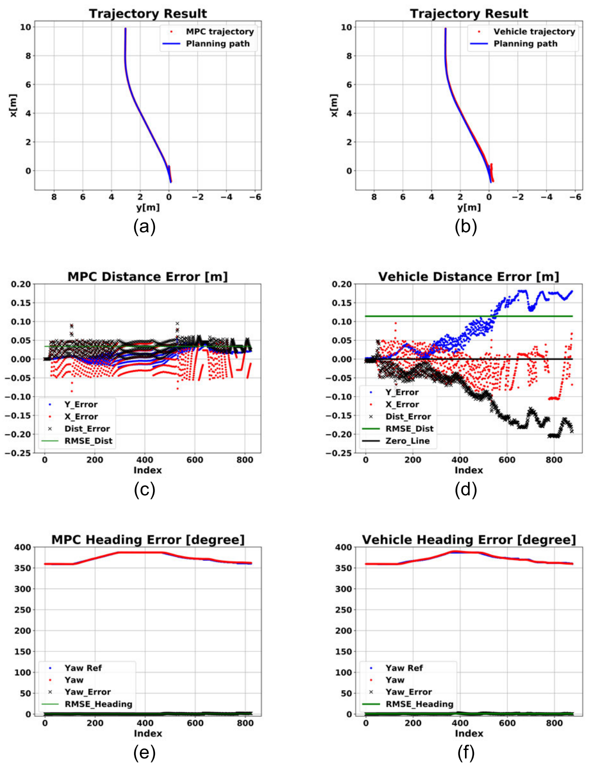

FIGURE 42. The simulation (left) and experimental (right) tracking results of common parallel front-left trial 1. (a) and (b) The trajectory results. (c) and (d) The distance error profiles. (e) and (f) The heading error profiles.

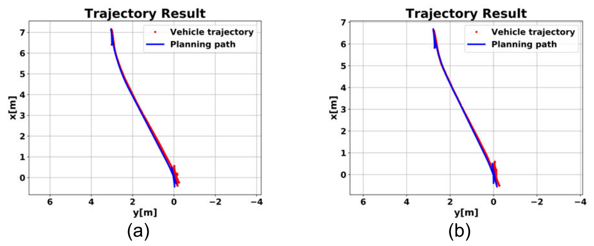

FIGURE 43. The additional tracking trials in common parallel front-left trial. (a) Trial 2. (b) Trial 3.

TABLE 15. The statistical results of common parallel front-left trials.

	Simulation	Trial 1	Trial 2	Trial 3
Distance RMSE (m)	0.037	0.114	0.106	0.113
Heading RMSE (degree)	2.364	0.918	1.201	0.930
Max Distance Error (m)	0.100	0.207	0.165	0.263
Max Heading Error (degree)	6.452	2.822	3.760	2.473
Final x Error (m)	0.029	0.068	0.120	0.155
Final y Error (m)	-0.018	-0.181	-0.157	-0.122
Final Yaw Error (degree)	6.452	0.527	1.011	0.231

vehicle controller is still good enough to overcome this kind of parallel parking mission.

VIII. CONCLUSION

Recently, the market attention of autonomous vehicles arises and autonomous parking has the potential to be the first fully autonomous driving function. Therefore, the paper proposes an autonomous parking system consisting of a sampling-based motion planner and a parking-oriented vehicle controller to work around the autonomous parking issues.

On one hand, the proposed motion planner aims to generate a human-like and collision-free path efficiently with any feasible start and goal configurations in parking scenarios, such as perpendicular parking, parallel parking. Also, the proposed motion planner can improve the common disadvantages of RRT-related methods, such as uncertainties of path quality and inefficient exploration in narrow spaces. Without any template setups, the usage of the proposed motion planner is not limited to well-known environments but scenarios in everyday life.

On the other hand, the proposed vehicle controller is able to track accurately and control smoothly for autonomous parking. Not only the steering control but also speed control are implemented simultaneously to deal with motion direction changes. In advance, the MPC computation and vehicle control are conducted simultaneously in the proposed controller to deal with the side effects of combining both steering and speed controls. Moreover, to implement the proposed method in practice, some practical knowledge about vehicle driving should be considered. Most importantly, the proposed controller is general and friendly to various vehicles because of the simple linear state space model with vehicle kinematics.

Additionally, various and strict simulations have been extensively conducted to fully examine not only the effects of each technique in the methodology but also the abilities for common and strict parking scenarios. The excellent performance in simulations indicates that the proposed parking system has potential to deal with autonomous parking issues. In advance, the performance from experimental tests indicates that the proposed methods can be implemented on real autonomous vehicle to provide autonomous parking services in everyday life.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their detailed comments which helped to improve the quality of the paper. The work was supported by the Ministry of Science and Technology, Taiwan, under grants: MOST 105-2221-E-002-135-MY3 and MOST 108-2634-F-002-0024, and by Industrial Technology Research Institute, Hsinchu, Taiwan.

REFERENCES

[1] R. Hussain and S. Zeadally, “Autonomous cars: Research results, issues, and future challenges,” *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1275–1313, 2nd Quart., 2019.

[2] J. Ziegler *et al.*, “Making bertha drive—An autonomous journey on a historic route,” *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, Apr. 2014.

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

[4] R. Langari, “Autonomous vehicles a tutorial on research and development issues,” in *Proc. IEEE Amer. Control Conf. (ACC)*, Seattle, WA, USA, May 2017, pp. 4018–4022

[5] Z. Feng, S. Chen, Y. Chen, and N. Zheng, “Model-based decision making with imagination for autonomous parking,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 2216–2223.

[6] K. Zheng and S. Liu, “RRT based path planning for autonomous parking of vehicle,” in *Proc. IEEE 7th Data Driven Control Learn. Syst. Conf. (DDCLS)*, May 2018, pp. 627–632.

[7] H. Banzhaf, L. Palmieri, D. Nienhuser, T. Schamm, S. Knoop, and J. M. Zollner, “Hybrid curvature steer: A novel extend function for sampling-based nonholonomic motion planning in tight environments,” in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–8.

[8] S. Shin, J. Ahn, and J. Park, “Desired orientation RRT (DO-RRT) for autonomous vehicle in narrow cluttered spaces,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 4736–4741.

[9] Y. Wang, D. K. Jha, and Y. Akemi, “A two-stage RRT path planner for automated parking,” in *Proc. 13th IEEE Conf. Autom. Sci. Eng. (CASE)*, Aug. 2017, Art. no. 496502.

[10] J.-H. Jhang, F.-L. Lian, and Y.-H. Hao, “Human-like motion planning for autonomous parking based on revised bidirectional rapidly-exploring random tree* with reeds-Shepp curve,” *Asian J. Control*, to be published, doi: 10.1002/asjc.2439.

[11] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, “A survey of motion planning and control techniques for self-driving urban vehicles,” *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[12] J. Wit, C. D. Crane, and D. Armstrong, “Autonomous ground vehicle path tracking,” *J. Robotic Syst.*, vol. 21, no. 8, pp. 439–449, 2004.

[13] C. Samson, “Control of chained systems application to path following and time-varying point-stabilization of mobile robots,” *IEEE Trans. Autom. Control*, vol. 40, no. 1, pp. 64–77, Jan. 1995.

[14] Z. P. Jiang and H. Nijmeijer, “Tracking control of mobile robots: A case study in backstepping,” *Automatica*, vol. 33, no. 7, pp. 1393–1399, Jul. 1997.

[15] B. d’Andréa-Novel, G. Campion, and G. Bastin, “Control of nonholonomic wheeled mobile robots by state feedback linearization,” *Int. J. Robot. Res.*, vol. 14, no. 6, pp. 543–559, Dec. 1995.

[16] G. V. Raffo, G. K. Gomes, J. E. Normey-Rico, C. R. Kelber, and L. B. Becker, “A predictive controller for autonomous vehicle path tracking,” *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 1, pp. 92–102, Mar. 2009.

[17] E. Kim, J. Kim, and M. Sunwoo, “Model predictive control strategy for smooth path tracking of autonomous vehicles with steering actuator dynamics,” *Int. J. Automot. Technol.*, vol. 15, no. 7, pp. 1155–1164, Dec. 2014.

[18] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, “Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints,” *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 952–964, Feb. 2017.

[19] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, “Predictive active steering control for autonomous vehicle systems,” *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 566–580, May 2007.

[20] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat, “Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation,” *Int. J. Robust Nonlinear Control*, vol. 18, no. 8, pp. 862–875, 2008.

[21] C. Sun, X. Zhang, Q. Zhou, and Y. Tian, “A model predictive controller with switched tracking error for autonomous vehicle path tracking,” *IEEE Access*, vol. 7, pp. 53103–53114, 2019.

[22] J. Yu, X. Guo, X. Pei, Z. Chen, W. Zhou, M. Zhu, and C. Wu, “Path tracking control based on tube MPC and time delay motion prediction,” *IET Intell. Transp. Syst.*, vol. 14, no. 1, pp. 1–12, Jan. 2020.

[23] J. Ciganek, “Automatic parking control using fuzzy logic,” in *Proc. 6th Int. Conf. Adv. Control Circuits Syst. (ACCS) 5th Int. Conf. New Paradigms Electron. Inf. Technol. (PEIT)*, Nov. 2019, pp. 203–208.

[24] M. D. Filatov, V. E. Serykh, M. M. Kopichev, and V. A. Weinmeister, “Autonomous parking control system of four wheeled vehicle,” in *Proc. IEEE V Forum Strategic Partnership Universities Enterprises Hi-Tech Branches (Sci. Educ. Innov.)*, St. Petersburg, Russia, Nov. 2016, pp. 102–107.

[25] T.-h. Hsu, J.-F. Liu, P.-N. Yu, W.-S. Lee, and J.-S. Hsu, “Development of an automatic parking system for vehicle,” in *Proc. IEEE Vehicle Power Propuls. Conf.*, Sep. 2008, pp. 1–6.

[26] M. Ikhwan, Mardlijah, and D. K. Arif, “Model predictive parking control on four wheel vehicle with optimum parking space,” in *Proc. Int. Conf. Electr. Eng. Informat. (ICELTICs)*, Oct. 2017, pp. 35–39.

[27] M. Josevski, A. Katriniok, V. Neisen, A. Riek, and D. Abel, "Model predictive control for road disturbance rejection in on-curb parking scenarios," in *Proc. IEEE 56th Annu. Conf. Decis. Control (CDC)*, Dec. 2017, pp. 855–860.

[28] P. Petrov and F. Nashashibi, "Automatic vehicle perpendicular parking design using saturated control," in *Proc. IEEE Jordan Conf. Appl. Electr. Eng. Comput. Technol. (AEECT)*, Nov. 2015, pp. 1–6.

[29] Y. Jeong, S. Kim, K. Yi, S. Lee, and B. Jo, "Design and implementation of parking control algorithm for autonomous valet parking," in *Proc. SAE Tech. Paper Ser.*, Apr. 2016, pp. 956–959.

[30] K. Oyama and K. Nonaka, "Model predictive parking control for nonholonomic vehicles using time-state control form," in *Proc. Eur. Control Conf. (ECC)*, Jul. 2013, pp. 458–465.

[31] S. Ma, H. Jiang, M. Han, J. Xie, and C. Li, "Research on automatic parking systems based on parking scene recognition," *IEEE Access*, vol. 5, pp. 21901–21917, Oct. 18, 2017.

[32] N. R. Jazar, *Vehicle Dynamics: Theory and Application*. Boston, MA, USA: Springer, 2008, pp. 379–395.

[33] R. Rajamani, "Vehicle dynamics and control," in *Mechanical Engineering Series*, 2nd ed, F. F. Ling, Ed. New York, NY, USA: Springer, 2012, pp. 15–46.

[34] A. Sakai. (Mar. 2019). *Model Predictive Speed and Steer Control*. [Online]. Available: https://github.com/AtsushiSakai/PythonRobotics.

[35] Trevor Henderson. (Mar. 2019). *Motion Planning*. [Online]. Available: https://github.com/sportdeath/motion_planning

[36] *CVXPY.org*. Accessed: Jul. 28, 2020. [Online]. Available: https://www.cvxpy.org/

[37] Autoware.ai. (Mar. 2019). *Mpc_Follower*. [Online]. Available: https://github.com/autowarefoundation/autoware/tree/feature/mpc_trajectory_follower

JYUN-HAO JHANG was born in Tainan, Taiwan, in 1996. He received the B.S. degree from the Department of Electrical and Computer Engineering, National Chiao Tung University, in 2018, and the M.S. degree from the Department of Electrical Engineering, National Taiwan University, in 2020.

From 2019 to 2020, he worked as an Autonomous Vehicle Research and Development Intern with the Industrial Technology Research Institute (ITRI), Hsinchu, Taiwan. His research interests include motion planning and vehicle control of autonomous parking.

FENG-LI LIAN (Senior Member, IEEE) received the B.S. and M.S. degrees from National Taiwan University, in 1992 and 1994, respectively, and the Ph.D. degree from the University of Michigan, in 2001. From 2001 to 2002, he was a Postdoctoral Scholar with the California Institute of Technology. Since 2002, he has been with the Department of Electrical Engineering, NTU. From 2009 to 2013, he was also the Division Director of the Information Management, Computer and Information Networking Center, NTU. Since 2015, he has been holding a Joint Appointment with the Industrial Technology Research Institute, Hsinchu, Taiwan. His current research interests include distributed and networked control systems, multiple dynamical agent systems, trajectory generation, and path planning for autonomous vehicles. He was a recipient of the Youth Automatic Control Engineering Award from the Chinese Automatic Control Society, Taiwan, in 2007, the Outstanding Youth Award from the Taiwan Association of System Science and Engineering, in 2012, the Dr. Wu, Da-You Memorial Research Award, National Science Council, Taiwan, in 2012, the Excellent Young Scholar Research Grant, National Science Council, Taiwan, from 2012 to 2014, and the NTU Excellent Teaching Award, in 2007, 2008, 2010–2013, 2018, and 2019.