A Hierarchical NMPC and TD3-Based Framework for Seamless Cruise-to-Park Automated Valet Parking

Dajie Tian and Levent Guvenc *

Posted Date: 29 April 2026

doi: 10.20944/preprints202604.2014.v1

Keywords: automated valet parking; NMPC; reinforcement learning; TD3

 doi:10.20944/preprints202604.2014.v1

Article

A Hierarchical NMPC and TD3-Based Framework for Seamless Cruise-to-Park Automated Valet Parking

Dajie Tian [1,2] and Levent Guvenc [1,2,3,*]

[1] Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH 43210, USA
[2] Automated Driving Lab, Department of Mechanical and Aerospace Engineering, The Ohio State University, Columbus, OH 43212
[3] Department of Mechanical and Aerospace Engineering, The Ohio State University, Columbus, OH 43210, USA
* Correspondence: guvenc.1@osu.edu

Abstract

Automated valet parking requires a difficult balance between reliable long-range empty spot seeking within structured parking lots and precise, low-speed maneuvers into tight terminal poses. Traditional controllers often struggle to bridge these two distinct operational domains. This paper proposes a hierarchical cruise-to-park framework that leverages the strengths of classical and learning-based control by using a Nonlinear Model Predictive Controller (NMPC) for predefined-route cruising and a Twin Delayed Deep Deterministic Policy Gradient (TD3) agent for final parking. The system is implemented in a high-fidelity Simulink environment with Unreal Engine-based sensors. During the cruising phase, a camera-based module detects available slots to trigger a seamless transition to the parking phase. The NMPC utilizes a custom cost function to minimize error on curved approaches, while the TD3 policy is trained with reward shaping and an explicit time penalty to promote efficient and stable low-speed docking using LiDAR feedback. Simulation results demonstrate smooth phase transition, accurate cruising, and effective terminal parking in the training slot. The validation results on six previously unseen target slots within the same structured parking-lot environment show encouraging intra-lot transferability without retraining, supporting the proposed modular architecture as a practical baseline for integrated cruise-to-park automated valet parking studies.

Keywords: automated valet parking; NMPC; reinforcement learning; TD3

1. Introduction

Automated valet parking (AVP) is a key function in intelligent transportation systems because it reduces the burden of low-speed parking, helps prevent minor collisions, and improves the use of limited parking resources in lots and garages [1]. Unlike highway autonomy, AVP operates in confined spaces with frequent curvature changes and tight terminal constraints, requiring both reliable long-range motion execution during cruising and precise low-speed maneuvers into a final target pose during parking. As a result, AVP must address long-horizon tracking and short-horizon high-precision docking simultaneously, while remaining robust to sensing imperfections, model mismatch, and discontinuities caused by curved approaches or mode switching [2,3].

Existing AVP solutions are generally divided into rule-based and learning-based methods [1,4]. Rule-based approaches typically combine geometric or graph-search planners with tracking controllers to achieve predictable behavior and constraint satisfaction in structured maps. Prior studies have shown that such planning and re-planning pipelines can handle narrow lots and perception errors with appropriate sensing configurations [5], and that search-based methods can generate feasible collision-free paths in tight parking scenarios [6]. However, these pipelines often require extensive retuning across layouts and can become fragile when errors accumulate over long

 doi:10.20944/preprints202604.2014.v1

approach segments, especially under high curvature, heading wrap-around, or local-minimum effects caused by limitations of low-speed actuation.

Learning-based methods, particularly deep reinforcement learning (DRL), have been explored to reduce hand-crafted rule complexity and to learn policies directly from interaction. Although reinforcement learning has shown promise in autonomous driving, its real-world deployment remains constrained by poor sample efficiency, training instability, and strong dependence on reward design and exploration strategy [4]. These issues are even more pronounced in parking tasks, where sparse rewards provide weak learning signals, slow convergence, and may induce undesirable behaviors such as oscillation or spinning near the goal [4,7]. To alleviate these problems, data-efficient and reward-shaped reinforcement learning formulations have been proposed to improve convergence and better alignment between planning and control objectives [8]. This has motivated hierarchical designs in which model-based and learning-based components handle long-range cruising and precise terminal docking within their respective operating regimes.

Recent AVP research increasingly adopts hierarchical architectures that combine model-based control with reinforcement learning, allowing long-range motion to remain constraint-aware while the final maneuver benefits from learned nonlinear behavior [7]. On the model-based side, optimization-based planners can explicitly enforce kinematic feasibility and collision-avoidance constraints, and have shown strong performance in cluttered or irregular environments, especially when iterative refinement or warm-start strategies are used to maintain practical computation [9,10]. Online re-planning methods have also been studied to preserve trajectory continuity when newly detected obstacles invalidate an ongoing parking plan [11]. In parallel, reinforcement learning has gained attention for low-speed parking because it can capture nonlinear maneuvers with continuous control and has recently demonstrated improved robustness under non-ideal scenarios with modern algorithms [12]. More broadly, MPC-RL integration has been investigated to reduce trial-and-error while preserving constraint structure, such as by using NMPC to assist policy learning or switching, or by learning decision policies that modulate MPC execution [13,14].

Despite this progress, several gaps remain in achieving a seamless cruise-to-park AVP pipeline in a structured parking-lot setting. First, much of the literature still formulates parking as an isolated start-to-slot task, whereas practical AVP requires long-range cruising, perception-driven slot confirmation, and reliable mode switching before the terminal maneuver. Second, long-horizon cruising in confined lots can suffer from tracking error accumulation and discontinuities caused by tight curvature and heading wrap-around, yet these effects are rarely analyzed together with downstream parking performance in an integrated framework. Third, learning-based parking policies often exhibit inefficient near-goal behaviors, such as oscillation or spinning under sparse or delayed rewards, while prior studies do not consistently report stage-wise error trends or demonstrate transfer performance across multiple target slots under fixed controller settings.

Motivated by these gaps, this paper proposes a cruise-to-park AVP framework that couples constraint-aware nonlinear model predictive control (NMPC) for cruising with a learning-based low-speed parking policy. In the implemented system, the vehicle first follows a predefined cruising route in a structured parking lot, and a camera-based free-slot detection module triggers the transition from slot searching to parking once an available space is detected [15]. During cruising and search, the NMPC formulation employs a customized cost function that balances path-tracking accuracy, speed regulation, and control smoothness, enabling accurate long-horizon execution with small tracking errors, including on curved segments. During parking, a Twin Delayed Deep Deterministic Policy Gradient (TD3) agent is trained for continuous control using LiDAR feedback for collision-aware docking [16]. The reward function includes an explicit time-penalty term to suppress pre-parking spinning and to encourage timely convergence to the terminal pose, consistent with the role of reward shaping in improving learning efficiency under sparse or delayed feedback [17]. The overall system is implemented in a high-fidelity Simulink environment with Unreal Engine-based sensors, enabling consistent evaluation of the perception-triggered phase transition and the end-to-end cruise-to-park process.

 doi:10.20944/preprints202604.2014.v1

The integrated framework is validated through parking-lot simulations in Simulink. Generalization is examined on six previously unseen target slots using identical controller settings and without further retraining, while approach-phase tracking errors and terminal parking errors are recorded for evaluation. The results show stable phase transitions, accurate cruising behavior, and transferable parking performance across the tested slots, suggesting that the proposed NMPC tracker with a customized cost function and the time-penalized TD3 parking policy provide a practical baseline for cruise-to-park AVP studies.

The main contributions of this paper are summarized as follows:

- A hierarchical cruise-to-park AVP framework is developed by integrating camera-triggered slot detection, NMPC-based long-range cruising, and TD3-based low-speed parking within a unified simulation environment.
- A customized NMPC cost design is introduced for the cruising phase to improve long-horizon path-following performance by jointly regulating tracking accuracy, vehicle speed, and control smoothness in structured parking-lot routes.
- A time-penalized TD3 reward formulation is proposed for the parking phase to reduce inefficient oscillation or spinning near the goal and to improve convergence toward the target pose during collision-aware docking.
- The proposed framework is validated on multiple previously unseen parking slots under fixed controller settings to examine phase transition behavior, docking feasibility, and target-slot transfer within the same parking-lot layout.

The remainder of this paper is organized as follows. Section 2 presents the methodology, including the overall system architecture, the perception-triggered transition logic, the NMPC design for cruising, and the TD3 formulation for parking control. Section 3 reports the simulation settings and experimental results, focusing on cruising performance, terminal parking accuracy, and generalization across unseen parking slots. Section 4 discusses the main findings, practical implications, and limitations of the proposed framework. Section 5 concludes the paper and suggests directions for future work.

2. Methodology

2.1. Problem Formulation and System Architecture

This work formulates automated valet parking (AVP) as a cruise-to-park task in a structured parking-lot environment. Starting from an initial pose, the ego vehicle follows a predefined cruising route while searching for an available parking slot, and then performs a low-speed docking maneuver into the target slot under terminal pose constraints. The vehicle pose is defined as $\mathbf{p} = [x, y, \psi]^T$, where x and y denote the planar coordinates in the parking-lot frame and ψ is the yaw angle. The control input is $\mathbf{u} = [v, \delta]^T$, where v is the commanded longitudinal speed and δ is the steering command. The control objective is to reach the target pose with bounded terminal errors while satisfying actuation limits and maintaining collision-free motion.

To address the different requirements of long-horizon cruising and short-horizon terminal docking, a hierarchical two-stage control architecture is adopted. During the cruising and slot-search phase, a nonlinear model predictive controller (NMPC) tracks the reference route and regulates vehicle motion under explicit constraints, aiming to reduce tracking error accumulation before entering the terminal parking region. During the final parking phase, a Twin Delayed Deep Deterministic Policy Gradient (TD3)-based controller generates steering actions for precise low-speed docking. In this stage, the longitudinal speed is fixed at $2\ m/s$, while the TD3 agent determines the steering command.

A phase manager and command selector coordinate the interaction between the two control stages to enable a smooth cruise-to-park transition. During cruising, a camera-based parking-slot perception module monitors slot availability. Once a free slot is confirmed, the perception module provides the corresponding target pose and triggers the switching logic from NMPC cruising to TD3

 doi:10.20944/preprints202604.2014.v1

parking. The phase manager handles the controller handoff, while the command selector routes the selected control commands to the vehicle plant to maintain smooth and well-behaved speed and steering signals. After switching, the TD3 parking controller executes the terminal maneuver using parking-related observations, including relative goal information and LiDAR-derived feedback.

Figure 1 illustrates the overall system architecture, including the simulation environment and vehicle plant, the perception modules, the phase manager and command selector, the NMPC cruising controller, and the TD3 parking controller.

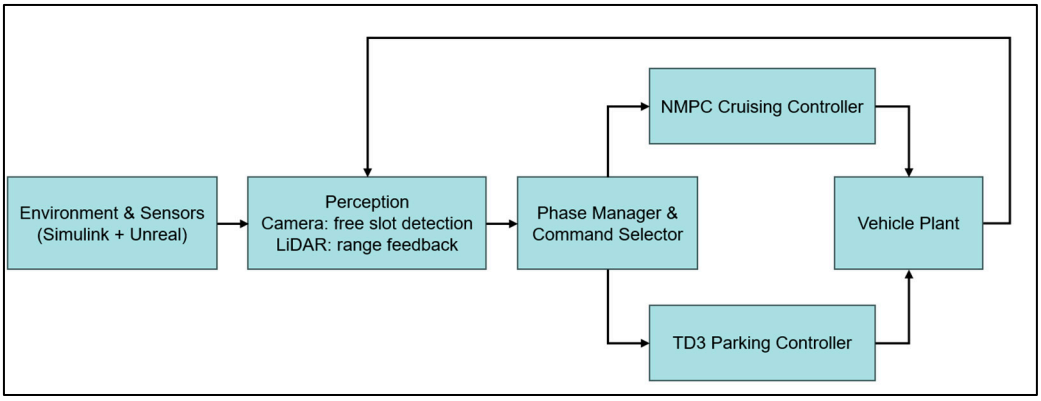

Figure 1. System block diagram of the cruise-to-park AVP framework.

2.2. Simulation Environment

All experiments are conducted in MATLAB/Simulink R2025a. The reinforcement learning agent is trained and evaluated using the Reinforcement Learning Toolbox. The closed-loop simulation model integrates the parking-lot environment, the ego-vehicle plant, the perception modules, the phase manager and command selector, the NMPC cruising controller, and the TD3 parking controller. The NMPC module is implemented using MATLAB's nonlinear MPC functionality in Simulink.

The structured parking-lot layout and slot indexing are adapted from the MathWorks automatic parking valet example [18]. In that baseline, the parking process is organized into a search phase and a parking phase. Building on this layout, the present work implements the proposed NMPC-TD3 cruise-to-park framework, including the controller design, phase-switching logic, and the training and generalization protocol. Figure 2 shows the parking-lot map with indexed parking slots and marked aisles. Each parking slot is assigned a unique index for target selection and experiment organization. In this study, the TD3 policy is trained on a single slot (slot 7) and evaluated on six previously unseen slots (14, 15, 23, 39, 47, and 64) using identical controller settings and without retraining. This indexed layout also enables consistent comparison between training and test scenarios.

All vehicle states and reference quantities are represented in a parking-lot-fixed planar frame, following the pose and control definitions introduced in Section 2.1. To avoid discontinuities in heading-related computations, yaw angles are wrapped to a principal interval using modulo-based mapping, which ensures stable behavior near the $\pm\pi$ boundary. The closed-loop system operates with a discrete controller sample time of $T_s = 0.1\,\mathrm{s}$, and both the nonlinear model predictive controller and the TD3 parking controller are executed at this rate. The simulation stop time is set to $T_f = 50\mathrm{s}$, consistent with the training and evaluation configuration. Sensor outputs are updated at the same effective rate as the controller cycle, enabling synchronized perception and control throughout all experiments.

 doi:10.20944/preprints202604.2014.v1

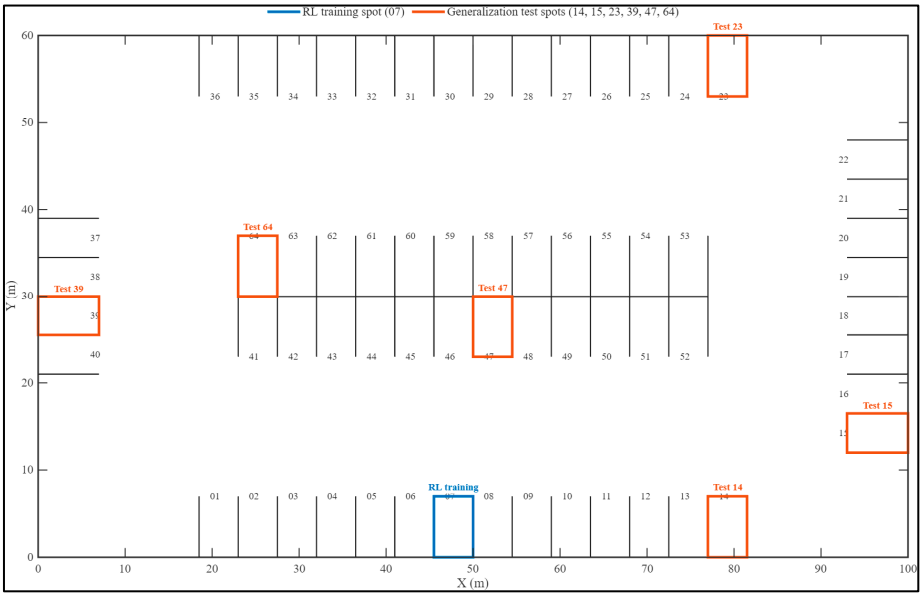

Figure 2. Layout of the parking lot with training spot and validation test spots.

2.3. Vehicle Model and Constraints

To describe low-speed vehicle motion in the structured parking-lot environment, the ego vehicle is modeled using a planar kinematic bicycle model [19], as shown in Figure 3. The vehicle state is defined by its planar position (X, Y) and yaw angle ψ in the parking-lot-fixed frame, while the control input consists of the longitudinal speed v and front steering angle δ. The vehicle geometry is characterized by the distances from the center of gravity (CG) to the front and rear axles, denoted by l_f and l_r, respectively. In this work, the kinematic bicycle model is used both as the closed-loop vehicle model and as the prediction model for the NMPC cruising controller.

The continuous-time kinematic bicycle model is written as

$$\dot{X} = v\cos\,\psi, \tag{1}$$

$$\dot{Y} = v\sin\,\psi, \tag{2}$$

$$\dot{\psi} = \frac{v}{L}\tan\,\delta, \tag{3}$$

where $L = l_f + l_r$ is the wheelbase. For digital implementation in Simulink, the continuous-time model is discretized with the controller sampling time T_s, yielding

$$X_{k+1} = X_k + T_s v_k \cos\,\psi_k, \tag{4}$$

$$Y_{k+1} = Y_k + T_s v_k \sin\,\psi_k, \tag{5}$$

$$\psi_{k+1} = \psi_k + T_s \frac{v_k}{L}\tan\,\delta_k, \tag{6}$$

This discrete-time model is used for closed-loop simulation and NMPC prediction.

To ensure feasible and consistent closed-loop execution, constraints are imposed on both the control inputs and the vehicle motion. The longitudinal speed is bounded by $v \in [0,2]\,m/s$, restricting the vehicle to forward low-speed operation throughout the maneuver. The steering command is limited to $\delta \in [-\pi/4, \pi/4]$ rad, consistent with the steering range used during both training and evaluation. In addition, the vehicle motion is constrained within the map-defined workspace limits of the parking lot. Any command exceeding the prescribed bounds is saturated before being applied to the vehicle plant, ensuring that identical actuator limits are enforced for both

 doi:10.20944/preprints202604.2014.v1

the NMPC cruising controller and the TD3 parking controller. The timing and actuator constraints used in the experiments are summarized in Table 1.

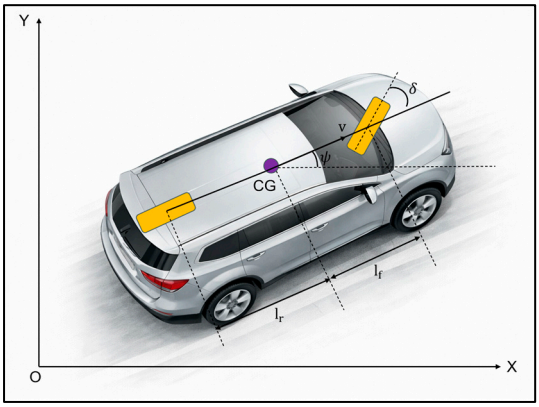

Figure 3. Simplified kinematic bicycle model.

Table 1. Timing parameters and actuator constraints used in the experiments

Item	Symbol	Value
Controller sample time	T_s	0.1 s
Longitudinal speed bound	v	$[0,2]\ m/s$
Steering angle bound	δ	$[-\pi/4, \pi/4]\ rad$
Workspace limits	(x, y)	defined by parking-lot map bounds

2.4. Perception and Switching Logic

2.4.1. Camera-Based Free Slot Detection

During the cruising stage, a simulated camera module is used to detect an available parking slot ahead of the ego vehicle and to provide a switching trigger for cruise-to-park operation. Unlike pixel-level vision pipelines, this camera is implemented as a geometry-based sensor in the Unreal/Simulink environment. It does not output images, but instead evaluates the visibility of candidate slots using the known parking-lot geometry and returns a binary detection signal together with the target pose of the selected free slot.

Candidate slot locations are represented by a set of predefined slot reference points $\{(x_i, y_i)\}$ on the map, where i denotes the slot index. For each slot, the line-of-sight bearing angle and Euclidean distance are computed as

$$\phi_i = atan2(y_i - y, x_i - x), \tag{7}$$

$$d_i = \sqrt{(x_i - x)^2 + (y_i - y)^2}, \tag{8}$$

The camera applies forward field-of-view (FoV) and depth gating to determine whether a slot is visible. To ensure robust gating near angle wrap-around boundaries, the bearing can be evaluated either as an absolute line-of-sight angle in the parking-lot frame or as a relative angle with respect to the ego heading. Both representations are equivalent for the FoV test. Using the relative form, the bearing difference is defined as

$$\Delta\phi_i = wrap_\pi(\phi_i - \psi), \tag{9}$$

where $\text{wrap}_\pi(\cdot)$ maps an angle to $(-\pi, \pi]$. A slot is considered visible if

$$|\Delta\phi_i| \le \frac{FoV}{2}, \tag{10}$$

 doi:10.20944/preprints202604.2014.v1

$$d_i \leq MaxDepth, \tag{11}$$

Figure 4 illustrates the field-of-view and range-gating geometry used by the camera module. Among the candidate slots that satisfy the FoV and depth constraints and are labeled as unoccupied by the map, an available slot is selected according to the slot indexing order:

$$i^* = min\left\{i: |\Delta\phi_i| \leq \frac{FoV}{2}, d_i \leq MaxDepth, slot\ i\ is\ unoccupied\right\}, \tag{12}$$

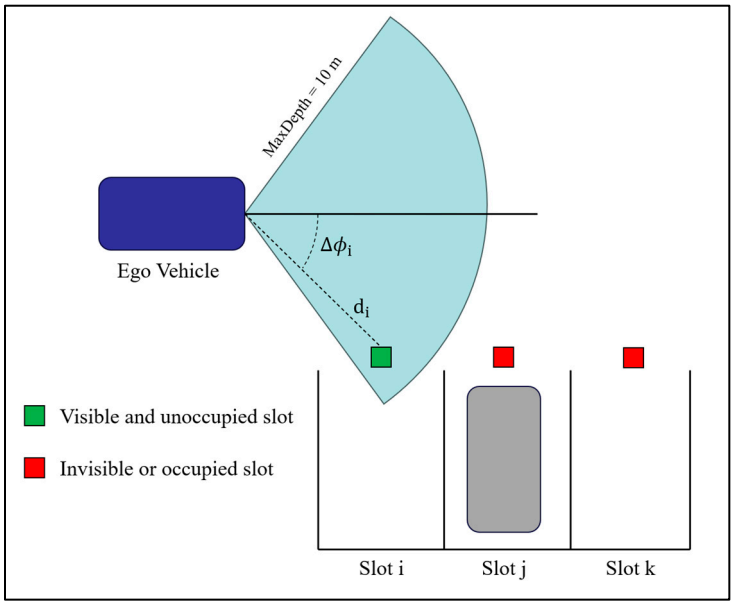

Figure 4. Camera FoV and range gating for free-slot detection.

The detection signal is set to 1 if the feasible set in (12) is nonempty and 0 otherwise, and the target pose is defined as the docking pose associated with slot i^*. To prevent intermittent detections from causing oscillatory switching behavior, the target pose is latched once a free slot is detected for the first time, and the latched target pose is held constant thereafter. This latched target pose is then used by the mode-switching logic to trigger the transition from NMPC cruising to TD3 parking. In the reported experiments, $MaxDepth = 10\ m$, $\text{FoV} = 120°$, and the camera update rate is synchronized with the controller sampling time of $T_s = 0.1\ s$.

2.4.2. LiDAR Feedback for Final Parking

During the final parking stage, a simulated LiDAR module provides range-based feedback to support collision-aware docking in tight terminal regions. The LiDAR is implemented as an Unreal/Simulink sensor based on ray casting on the parking-lot geometry. Instead of generating point clouds, it returns a compact range vector that summarizes the nearest obstacle distance along a set of fixed beam directions around the ego vehicle.

Let N denote the number of LiDAR beams. The beam directions are uniformly distributed over $360°$ around the ego vehicle, yielding a set of relative beam angles

$$\alpha_k = \frac{2\pi(k-1)}{N}, k = 1, \dots, N, \tag{13}$$

For each beam k, the LiDAR reports the distance to the nearest obstacle along the corresponding ray direction, forming the range vector

$$\mathbf{d} = [d_1, d_2, \dots, d_N]^\top \in R^N, \tag{14}$$

 doi:10.20944/preprints202604.2014.v1

Figure 5 illustrates the simulated LiDAR sensing geometry used during final parking. To ensure numerical stability and consistent sensing behavior, range measurements are limited to a valid interval,

$$d_k \in [d_{min}, d_{max}], k = 1, \dots, N, \quad (15)$$

where d_{max} is the maximum sensing range and d_{min} is the minimum valid range. In the reported experiments, $N = 12$ beams are used with $d_{min} = 0.5\ m$ and $d_{max} = 6\ m$. The LiDAR update rate is synchronized with the controller sampling time of $T_s = 0.1\ s$, providing range feedback at each control step during parking.

The LiDAR range vector is embedded directly into the TD3 observation used for the parking policy. Specifically, the observation combines relative goal information, ego motion state, and LiDAR ranges. The LiDAR contributes the N-dimensional vector $\mathbf{d}$ as a compact representation of nearby free space and surrounding obstacles. This range-based representation provides an effective perception signal for learning collision-free low-speed parking maneuvers while remaining lightweight and fully synchronized with the closed-loop controller execution.

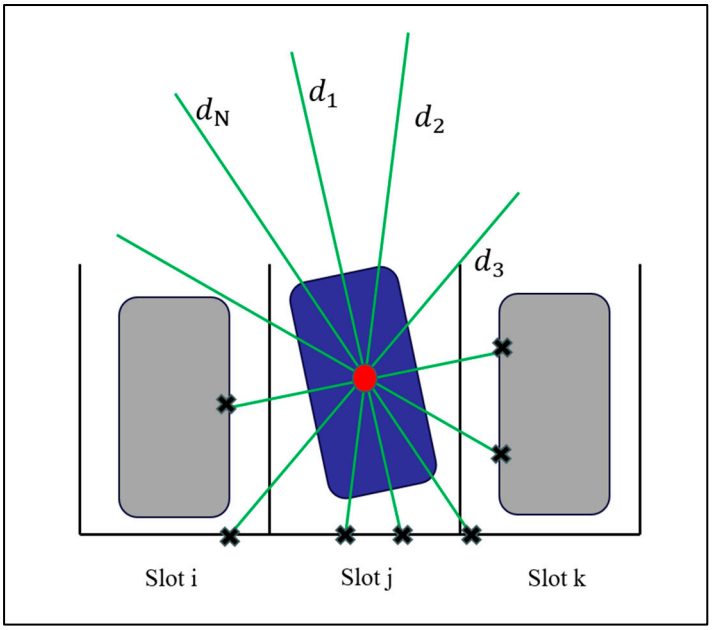

Figure 5. Simulated LiDAR range feedback for final parking.

2.4.3. Seamless Cruise-to-Park Transition

The integrated phase manager and command selector coordinate the cruising and parking stages to enable a seamless cruise-to-park transition. The switching logic is implemented as a finite-state machine with two operational modes, namely Cruise/Slot Search and Park, followed by two terminal outcomes, Done and Fail, as shown in Figure 6. In the Cruise/Slot Search mode, the NMPC controller tracks the predefined route while the perception modules continuously monitor slot availability. In the Park mode, control authority is transferred to the TD3 parking controller for the terminal maneuver, and the process terminates upon either successful docking or failure caused by collision or timeout.

The transition from Cruise/Slot Search to Park is event-triggered by the camera-based free-slot detection described in Section 2.4.1. Once a free slot is confirmed, the corresponding target pose is latched and used to trigger the switching logic from NMPC cruising to TD3 parking. This event-driven design ensures that the terminal parking phase is activated only after a valid and stable target slot has been identified.

 doi:10.20944/preprints202604.2014.v1

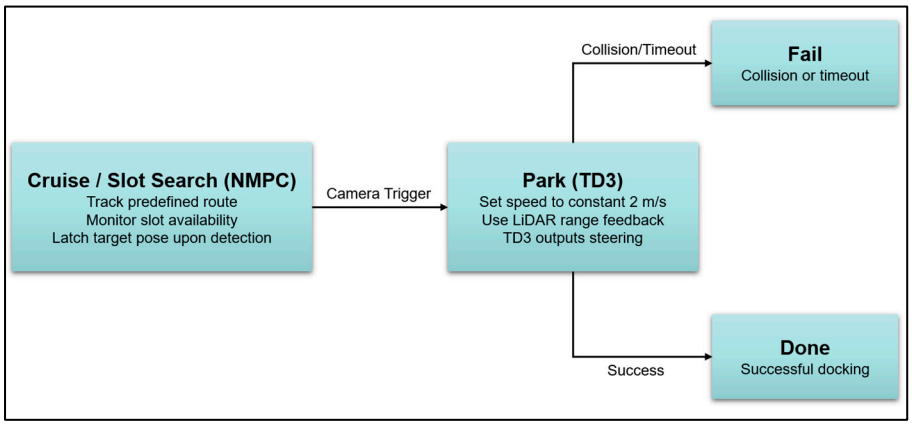

Figure 6. Finite-state machine for seamless cruise-to-park operation.

To ensure stable behavior at the switching instant, the integrated phase manager and command selector perform a controlled handover between the two controllers. First, latching the target pose prevents oscillatory switching caused by intermittent detections. Second, consistent actuator limits are enforced across both stages through command saturation, ensuring that the commanded inputs remain within the same admissible bounds before and after the transition. Third, after switching to the park mode, the commanded longitudinal speed is fixed at $v = 2\,m/s$, while the TD3 policy provides the steering command for terminal docking. This design yields a stable handover from long-horizon route tracking to short-horizon, collision-aware parking without reinitializing the vehicle state.

2.5. NMPC Cruising Controller

2.5.1. NMPC Optimization Formulation

During the Cruise/Slot Search stage, a nonlinear model predictive controller (NMPC) is used to track a predefined reference route in the structured parking lot while respecting actuator limits. At each control update, the NMPC solves a constrained finite-horizon optimization problem using the current vehicle state and a preview of the upcoming reference trajectory [20]. The controller state is defined as $\mathbf{x} = [x, y, \psi]^T$, and the control input is $\mathbf{u} = [v, \delta]^T$, where v is the commanded longitudinal speed and δ is the steering command.

Let p denote the prediction horizon. Given the current state $\mathbf{x}_0$ and a reference preview $\{\mathbf{r}_k\}_{k=1}^{p}$ along the cruising route, where $\mathbf{r}_k = [x_k^{\text{ref}}, y_k^{\text{ref}}, \psi_k^{\text{ref}}]^T$, the NMPC computes an optimal control sequence $\{\mathbf{u}_k\}_{k=0}^{p-1}$ by minimizing a composite objective that balances route-tracking accuracy, speed regulation, and control smoothness. The optimization problem is formulated as

$$min_{\{\mathbf{u}_k\}_{k=0}^{p-1}} \sum_{k=1}^{p} \ell(\mathbf{x}_k, \mathbf{u}_k, \Delta\mathbf{u}_k; \mathbf{r}_k) + \ell_T(\mathbf{x}_p, \mathbf{r}_p), \tag{16}$$

subject to the discrete-time prediction model

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k), k = 0, .., p-1, \tag{17}$$

and input constraints

$$v_{\min} \le v_k \le v_{\max}, \tag{18}$$

$$\delta_{\min} \le \delta_k \le \delta_{\max}, \tag{19}$$

Here, $\Delta\mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_{k-1}$ denotes the control increment used to encourage smooth actuation. In the implemented controller, the bounds are set to $v_{\min} = 0\,m/s$, $v_{\max} = 2\,m/s, \delta_{\min} = -\frac{\pi}{4}\,rad, \delta_{\max} = \frac{\pi}{4}\,rad$, consistent with the operating constraints defined in Section 2.3.

 doi:10.20944/preprints202604.2014.v1

The NMPC is implemented in Simulink using a nonlinear MPC controller block parameterized by a workspace-defined nonlinear MPC object. At each sampling instant, the finite-horizon optimization problem is solved by a built-in nonlinear programming solver based on sequential quadratic programming [21]. The controller uses the same sampling time as the prediction model, and the reference preview is provided as an external input generated by the route planner as a function of simulation time. To improve numerical stability and reduce step-to-step control discontinuities, the previous control action is fed back to the controller through the dedicated last manipulated variable input with a one-step delay, enabling warm-starting and consistent handling of control increments. The resulting NMPC output is reshaped into a 2×1 column vector to match the downstream command interface $[v, \delta]^T$. In this work, the controller sampling time is $T_s = 0.1\ s$, and both the prediction horizon and the control horizon are set to 10 steps.

2.5.2. Custom NMPC Cost Function

A customized NMPC objective is adopted to improve tracking performance on curved cruising segments and reduce error accumulation prior to the cruise-to-park transition. Let the predicted state at step k be $\mathbf{x}_k = [x_k, y_k, \psi_k]^T$ and the reference preview be $\mathbf{r}_k = [x_k^{\text{ref}}, y_k^{\text{ref}}, \psi_k^{\text{ref}}]^T$. Define the position differences $\Delta x_k = x_k - x_k^{\text{ref}}$ and $\Delta y_k = y_k - y_k^{\text{ref}}$. The reference-frame tracking errors are computed as

$$e_{lat,k} = -\sin\left(\psi_k^{ref}\right)\Delta x_k + \cos\left(\psi_k^{ref}\right)\Delta y_k, \tag{20}$$

$$e_{lon,k} = \cos\left(\psi_k^{ref}\right)\Delta x_k + si\,n\left(\psi_k^{ref}\right)\Delta y_k, \tag{21}$$

The heading error is defined with angle wrapping to avoid discontinuities near the $\pm\pi$ boundary:

$$e_{\psi,k} = \text{wrap}\left(\psi_k - \psi_k^{ref}\right) \in (-\pi, \pi], \tag{22}$$

where $\text{wrap}_\pi(\cdot)$ maps an angle to the principal interval $(-\pi, \pi]$. To encourage smooth actuation, control increments are penalized:

$$\Delta v_k = v_k - v_{k-1}, \tag{23}$$

$$\Delta \delta_k = \delta_k - \delta_{k-1}, \tag{24}$$

A terminal error vector is formed from the last-step reference-frame errors:

$$\boldsymbol{e}_T = [e_{lat,p}, e_{lon,p}, e_{\psi,p}]^T, \tag{25}$$

Over the prediction horizon p, the customized NMPC cost is defined as

$$J = \sum_{k=1}^{p}\left(w_{lat}e_{lat,k}^2 + w_{lon}e_{lon,k}^2 + w_\psi e_{\psi,k}^2 + w_v v_k^2 + w_\delta \delta_k^2 + w_{\Delta v}\Delta v_k^2 + w_{\Delta\delta}\Delta\delta_k^2\right) + w_T \parallel \boldsymbol{e}_T \parallel_2^2, \tag{26}$$

This formulation emphasizes reference-frame path tracking through $(e_{lat}, e_{lon}, e_\psi)$, penalizes excessive control effort and rapid command changes via $(v_k, \delta_k, \Delta v_k, \Delta\delta_k)$, and explicitly penalizes end-of-horizon alignment through the terminal penalty $\parallel \boldsymbol{e}_T \parallel_2^2$. Using reference-frame errors reduces coupling between global (x, y) deviations and heading changes on curved segments, while the wrapped heading error avoids artificial jumps caused by angle wrap-around. The increment penalties suppress oscillatory behavior that can otherwise amplify tracking errors over long cruising horizons. Since the commanded speed is constrained to be nonnegative in the NMPC formulation, no additional reverse-motion penalty is included in the cost.

The weight values used in this paper are summarized in Table 2. These parameters were selected empirically through trial-and-error tuning in closed-loop simulation. Starting from feasible initial values, the weights were iteratively adjusted to balance reference tracking accuracy, heading alignment, control effort, and control smoothness on representative cruising scenarios, especially curved route segments. The final parameter set reported in Table 2 corresponds to the configuration that provided stable tracking performance with small accumulated error and without excessive

 doi:10.20944/preprints202604.2014.v1

oscillation or command chattering. Once selected, the same weight set was fixed for all reported experiments.

Table 2. NMPC cost weights used in the cruising controller

Category	Parameter	Value
Tracking	w_{lat}	40
Tracking	w_{lon}	8
Tracking	w_{ψ}	25
Effort	w_{v}	0.6
Effort	w_{δ}	2.0
Smoothness	$w_{\Delta v}$	6.0
Smoothness	$w_{\Delta \delta}$	10.0
Terminal	w_{T}	120

2.6. TD3 Parking Controller

2.6.1. MDP Formulation

The final parking task is formulated as a continuous-state, continuous-action Markov decision process (MDP), in which the TD3 agent learns a steering policy for low-speed collision-aware docking [22,23]. At each decision step t, the parking environment provides an observation vector s_t, the agent outputs an action a_t, and the simulator propagates the vehicle state to the next step according to the closed-loop parking dynamics described in Sections 2.3 and 2.4.

The observation is a 16-dimensional vector composed of target-relative pose features and LiDAR-based environment feedback. Specifically, the current vehicle pose and the target parking pose are transformed into a target-slot-aligned local frame. Let $e_{x,t}$ and $e_{y,t}$ denote the relative planar errors with respect to the target pose, and let $e_{\psi,t}$ denote the relative heading error in the target-aligned frame. To improve numerical conditioning, the position errors are normalized by a factor of 10, while the heading information is represented by its sine and cosine values to avoid discontinuities associated with angular wrap-around. In addition, the LiDAR measurements are normalized by the maximum sensing range $d_{\max}$. The state vector is therefore defined as

$$s_t = \left[\frac{e_{x,t}}{10}, \frac{e_{y,t}}{10}, \sin(e_{\psi,t}), \cos(e_{\psi,t}), \tilde{d}_{1,t}, \tilde{d}_{2,t}, \dots, \tilde{d}_{12,t}\right]^T \in \mathbb{R}^{16}, \tag{27}$$

where $\tilde{d}_{i,t} = d_{i,t}/d_{\max}$, $i = 1, \dots, 12$, and $d_{max} = 6\ m$. In the implemented sensing setup, 12 LiDAR beams are used, consistent with the LiDAR model described in Section 2.4.2.

The action is defined as a one-dimensional continuous steering command,

$$a_t = \delta_t, \delta_t \in \left[-\frac{\pi}{4}, \frac{\pi}{4}\right], \tag{28}$$

thus, during the parking stage, the TD3 policy controls only the steering angle, while the commanded longitudinal speed is maintained at a constant value of $v = 2m/s$, consistent with the cruise-to-park transition logic described in Section 2.4.3.

An episode terminates under one of three conditions: successful parking, invalid operation, or timeout. Successful parking is declared when the Euclidean position error with respect to the target pose is smaller than $0.75\ m$ and the absolute heading error is smaller than 10°. Invalid operation corresponds to failure cases such as leaving the admissible parking region or approaching an obstacle closer than the minimum valid LiDAR distance threshold, which is set to $0.5\ m$. During training, an additional invalid condition is triggered when the vehicle heading exceeds 2π, which is used to suppress persistent spinning behavior. A timeout occurs when the episode reaches the maximum allowed duration. In the reported experiments, the controller sampling time is $T_s = 0.1\ s$, and the maximum episode length is 500 steps, corresponding to a 50 s horizon.

 doi:10.20944/preprints202604.2014.v1

During training, the initial vehicle pose was randomized over several predefined regions of the parking-lot map to improve robustness to different approach conditions. The initial position was sampled either from two fixed entry points or from three subregions spanning the training area, with the yaw angle randomly drawn from the corresponding heading intervals. The initial longitudinal speed was fixed at 2 m/s.

2.6.2. Network Architecture and Training Configuration

The parking policy is learned using the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm [16]. In the implemented framework, the actor network maps the 16-dimensional observation vector to a one-dimensional continuous steering command, while two critic networks estimate the action value for the current state-action pair. Following the TD3 design, the two critics are trained in parallel, the minimum of their target Q-values is used to reduce overestimation bias, the actor is updated less frequently than the critics, and target policy smoothing is applied when forming the bootstrapped target. These mechanisms jointly improve the stability of continuous-control learning.

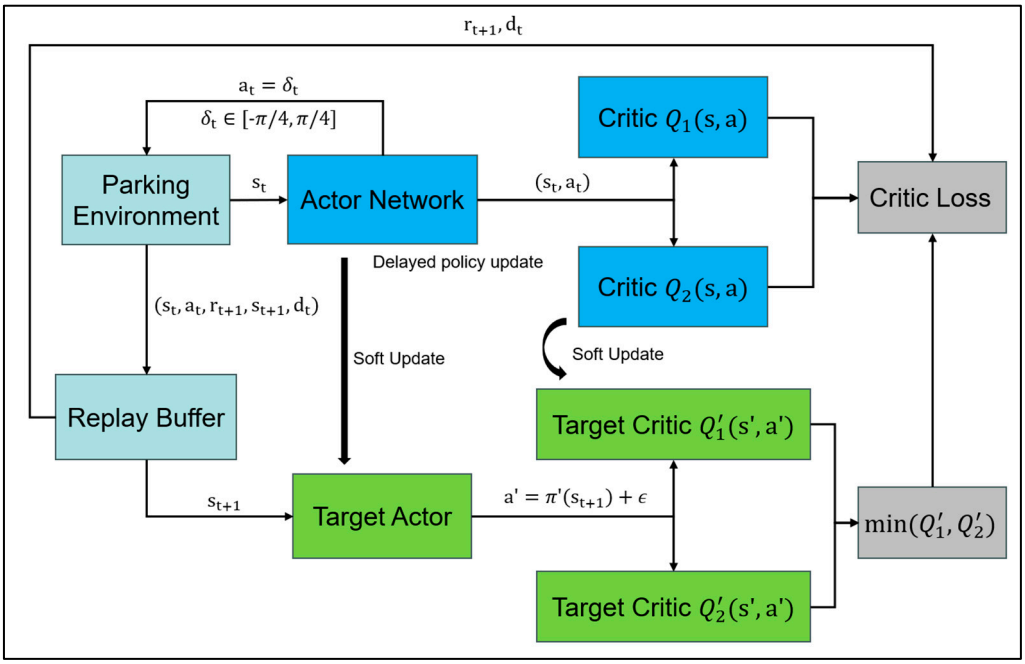

Figure 7. TD3 parking controller framework.

The actor network takes the normalized parking observation defined in Section 2.6.1 as input. It consists of two fully connected hidden layers with 256 units each, followed by ReLU activations. The output layer is followed by a hyperbolic tangent activation and an output-scaling stage so that the final action remains within the steering bound $[-\pi/4, \pi/4]$. The critic component adopts the twin-critic structure of TD3. Each critic receives both the observation and the steering action as inputs. In each critic, the state pathway and action pathway are first processed separately and then fused to estimate the scalar Q-value.

During training, transitions $(s_t, a_t, r_{t+1}, s_{t+1})$ are stored in a replay buffer and sampled in mini-batches for off-policy learning. Target networks are maintained for both the actor and the critics, and soft target updates are used after learning iterations. The policy update is delayed with respect to the critic updates, and smoothed target actions are generated by adding clipped noise to the target actor output when computing the critic target. This training setup is consistent with the standard TD3 framework while being specialized to the steering-only parking task considered in this work [16].

The main training hyperparameters used in this work are summarized in Table 3.

 doi:10.20944/preprints202604.2014.v1

Table 3. TD3 training hyperparameters used for parking policy learning

Hyperparameter	Value	Explanation
Observation dimension (n_o)	16	Target-relative pose features and normalized LiDAR ranges
Action dimension (n_a)	1	Continuous steering command
Actor hidden layers	[256,256]	Two fully connected hidden layers
Critic hidden layers	[256,256]	Two fully connected hidden layers for each critic
Actor learning rate (α_π)	3×10^{-4}	Learning rate of the policy network
Critic learning rate (α_Q)	1×10^{-3}	Learning rate of each Q-network
Discount factor (γ)	0.99	Discount factor for future rewards
Experience buffer length (N_{buf})	1×10^{6}	Capacity of the replay buffer
Mini-batch size (B)	256	Number of sampled transitions per update
Sample time (T_s)	0.1 s	Agent interaction and update interval
Delayed policy update frequency (d)	2	Delayed policy update interval
Target smooth factor (τ)	5×10^{-3}	Soft target update factor
Target policy smoothing variance $({\sigma_{tgt}}^2)$	0.1	Variance of smoothing noise for target actions
Target policy smoothing bounds	$[-0.25,0.25]$	Clipping range for target smoothing noise
Exploration noise standard deviation	0.15	Initial exploration noise level
Exploration noise minimum	0.02	Minimum exploration noise level
Exploration noise decay rate	2×10^{-4}	Decay rate of exploration noise
Target policy smoothing decay rate	1×10^{-5}	Decay rate of target smoothing variance
Gradient threshold	1	Gradient clipping threshold
Actor L_2 regularization	1×10^{-4}	Regularization for the actor optimizer
Maximum episodes (N_{epi})	15000	Training budget
Maximum steps per episode (N_{step})	500	Episode horizon

Although the maximum training budget was set to 15,000 episodes, an early stopping rule was used in the implemented training process. Specifically, an evaluation statistic was computed every 20 episodes as the mean episode reward over the evaluation window, and training was terminated once this value exceeded 140. Under this criterion, the reported parking policy converged and stopped after 850 episodes.

2.6.3. Reward Shaping with an Explicit Time Penalty

To improve parking efficiency and suppress oscillatory behavior near the target pose, the TD3 parking policy is trained with a shaped reward that combines distance-based incentives, orientation alignment, control regularization, terminal bonuses and penalties, and an explicit per-step time penalty [24,25]. The total reward at step t is defined as

$$r_t = r_{\text{dist},t} + r_{\text{prog},t} + r_{\text{ori},t} + r_{\text{ctrl},t} + r_{\text{park},t} + r_{\text{invalid},t} + r_{\text{time},t}, \quad (29)$$

The distance reward encourages the vehicle to approach the target parking pose in the planar position space. Let $e_{x,t}$ and $e_{y,t}$ denote the unnormalized target-relative position errors. In the

 doi:10.20944/preprints202604.2014.v1

implementation, these quantities are recovered from the normalized observation entries by multiplying the first two state components by 10. The distance reward is defined as

$$r_{\text{dist},t} = 2\exp\left(-0.05{e_{x,t}}^2 - 0.04{e_{y,t}}^2\right), \tag{30}$$

To additionally reward step-to-step improvement toward the goal, a progress reward is introduced using the increase in the distance reward relative to the previous control step. The reward increment is clipped to the interval $[0,0.1]$ before scaling, yielding

$$r_{\text{prog},t} = 3sat_{[0,0.1]}\left(r_{\text{dist},t} - r_{\text{dist},t-1}\right), \tag{31}$$

where $sat_{[0,0.1]}(\cdot)$ denotes saturation to the interval $[0,0.1]$. This design rewards positive progress while preventing excessively large stepwise reward jumps.

The orientation reward promotes heading alignment with the target parking pose. Let $e_{\psi,t} = \psi_t - \psi_g$ denote the heading mismatch between the current and target poses. The orientation term is defined as

$$r_{\text{ori},t} = 0.1\exp\left(-20{e_{\psi,t}}^2\right), \tag{32}$$

To discourage unnecessarily large steering commands and rapid steering oscillations, a control penalty is applied to both the steering action and its step-to-step variation:

$$r_{\text{ctrl},t} = -0.05{\delta_t}^2 - 0.1(\delta_t - \delta_{t-1})^2, \tag{33}$$

In addition to these shaping terms, terminal rewards are used to distinguish successful parking and invalid outcomes. A successful parking event produces a positive terminal bonus,

$$r_{\text{park},t} = 100\mathbb{I}_{\text{parked},t}, \tag{34}$$

while invalid operation, including leaving the admissible region or approaching obstacles closer than the minimum LiDAR safety threshold, incurs a penalty,

$$r_{\text{invalid},t} = -50\mathbb{I}_{\text{invalid},t}, \tag{35}$$

where $\mathbb{I}_{\text{parked},t}$ and $\mathbb{I}_{\text{invalid},t}$ are binary indicators for successful parking and invalid termination, respectively.

To explicitly discourage lingering, repeated spinning, or oscillatory maneuvers that do not lead to task completion, a per-step time penalty is applied whenever the episode is still active:

$$r_{\text{time},t} = -0.02(1 - \mathbb{I}_{\text{parked},t})(1 - \mathbb{I}_{\text{invalid},t}), \tag{36}$$

This term contributes a constant negative reward at every nonterminal step. As a result, behaviors that waste time without improving the parking state accumulate additional penalty. In particular, spinning or oscillatory corrections near the target become less favorable unless they lead to measurable progress or immediate task completion. The explicit time penalty therefore biases the learned policy toward shorter and more decisive parking maneuvers, reducing time-to-park while complementing the distance, orientation, and progress rewards [25].

The reward coefficients used in this work are summarized in Table 4.

Table 4. Reward coefficients used for TD3 parking policy learning.

Component	Value	Role
Distance reward scale (c_{dist})	2	Scales the position-based reward
Longitudinal position weight (w_x)	0.05	Penalizes target-relative x-error in distance reward
Lateral position weight (w_y)	0.04	Penalizes target-relative y-error in distance reward
Progress reward scale (c_{prog})	3	Scales the stepwise progress term

 doi:10.20944/preprints202604.2014.v1

Progress saturation lower bound (p_{min})	0	Prevents negative progress reward
Progress saturation upper bound (p_{max})	0.1	Limits excessively large progress increments
Orientation reward scale (c_{ori})	0.1	Scales the heading-alignment reward
Orientation error weight (w_{ψ})	20	Penalizes heading mismatch in orientation reward
Steering penalty weight (w_{δ})	0.05	Penalizes large steering commands
Steering increment penalty weight ($w_{\Delta\delta}$)	0.1	Penalizes rapid steering variation
Parking bonus (R_{park})	100	Reward for successful parking
Invalid-operation penalty ($R_{invalid}$)	-50	Penalty for collision or invalid termination
Time penalty (R_{time})	-0.02	Per-step penalty during active episode

2.7. Metrics and Evaluation Setup

2.7.1. Training–Validation Split and Generalization Setup

To evaluate the transfer capability of the proposed cruise-to-park framework, the parking policy is trained on a single target slot and then validated on multiple previously unseen slots without retraining. In the reported implementation, the TD3 parking controller is trained using slot 7 as the only training target, while the NMPC cruising controller, perception modules, and phase-switching logic remain fixed throughout both training and validation.

The single-slot training setup is adopted intentionally to examine whether the learned low-speed parking policy captures transferable terminal maneuvering behavior rather than memorizing slot-specific trajectories. Slot 7 is selected as the training target because it provides a representative parking configuration within the structured lot while allowing repeated closed-loop interaction under consistent sensing, actuation, and transition conditions. By restricting training to one slot, the subsequent validation can more clearly reveal the generalization capability of the learned parking controller across different target locations.

After training, the learned policy is validated on six previously unseen target slots, namely slots 14, 15, 23, 39, 47, and 64. During this evaluation stage, all controller settings are kept unchanged, including the NMPC configuration, the TD3 network parameters, the reward function, the parking success criteria, and the perception-triggered switching logic. No additional fine-tuning, retraining, or slot-specific parameter adjustment is performed. Therefore, performance differences across the validation slots reflect the transferability of the learned parking behavior under fixed controller settings rather than adaptation to each individual target.

This train-on-one-slot, validate-on-multiple-unseen-slots protocol is used to assess whether the proposed hierarchical framework can maintain reliable cruise-to-park execution when the terminal parking target changes within the same parking-lot environment. In this sense, the generalization study focuses on target-slot transfer under a fixed map and fixed controller design, providing a practical measure of robustness for the learned parking phase.

2.7.2. Metrics and Logging

To evaluate the performance of the proposed framework, both cruising-stage tracking quality and terminal parking accuracy were recorded during closed-loop simulation. Since the overall system follows a hierarchical cruise-to-park structure, the logged metrics were organized according to the two operational stages.

For the Cruise/Slot Search stage, the main evaluation quantities were the lateral tracking error and heading error defined in the reference frame of the cruising route, as introduced in Section 2.5.2. These quantities were used to assess how accurately the NMPC controller followed the predefined

 doi:10.20944/preprints202604.2014.v1

route before the parking transition was triggered. In addition to the full-time histories of the tracking errors, the approach-state error at the switching instant was also recorded to characterize the vehicle state passing from NMPC cruising to TD3 parking.

For the final parking stage, the main metrics were the terminal position error, terminal heading error, parking outcome, and time-to-park. The terminal position error was computed as the Euclidean distance between the final vehicle position and the target parking pose, while the terminal heading error was defined as the absolute yaw difference with respect to the target orientation. The time-to-park was measured over the parking phase after the cruise-to-park transition was activated. Invalid termination events, including collision-related failure or leaving the admissible region, were also logged.

In the reported experiments, the results are presented as slot-wise closed-loop evaluation outcomes under fixed controller settings. For each target slot, the recorded metrics include parking duration, terminal position error, lateral and longitudinal terminal errors, and terminal heading error. Aggregate mean values across the six unseen-slot tests are additionally reported to summarize overall transfer performance. This evaluation protocol enables both stage-specific analysis and overall assessment of the end-to-end cruise-to-park behavior under a consistent testing configuration.

All evaluations were conducted using the same trained TD3 policy and the same fixed NMPC, perception, and switching parameters described in the methodology section. No retraining or slot-specific retuning was introduced during testing, so the reported metrics directly reflect the transferability of the integrated framework under a fixed parking-lot setup.

It should be noted that the reported slot-wise evaluation was conducted in a deterministic Simulink closed-loop environment. For a given target slot, the trained TD3 policy, NMPC controller, perception-triggered switching logic, actuator constraints, and initial test configuration were all kept fixed. In addition, the target pose was latched once the free slot was detected, which ensured a consistent cruise-to-park transition point for the same test case. Under this deterministic setup, repeated simulations of the same slot produced identical closed-loop outcomes. Therefore, the reported result for each slot is representative of that fixed evaluation condition rather than a stochastic average over multiple randomized trials.

3. Results

3.1. Overall Performance of the Proposed AVP Framework

To evaluate the end-to-end capability of the proposed AVP framework, a representative cruise-to-park trial for slot 64 was examined in the structured parking-lot environment. As shown in Figure 8, the ego vehicle successfully completed the full task from the initial position to the target slot. The vehicle first followed the predefined cruising route under the NMPC cruising controller and then switched to the TD3 parking controller for the terminal docking maneuver after the target slot was confirmed by the perception module. The task was completed successfully without collision or invalid termination. Slot 64 was selected as the representative end-to-end case because it illustrates the complete cruise-to-park process in one of the unseen target-slot scenarios.

The full trajectory demonstrates that the proposed framework successfully integrates long-range constrained cruising and low-speed parking within a unified closed-loop architecture. During the cruising phase, the NMPC controller maintained stable path-following behavior along the predefined route and guided the vehicle smoothly toward the parking region. After the cruise-to-park switching point, the TD3 controller generated the steering actions required for precise low-speed docking into slot 64. No obvious trajectory discontinuity was observed around the controller handoff, indicating that the phase manager and command selector achieved the intended smooth transition between the two control stages.

Overall, this representative result confirms the feasibility of the proposed cruise-to-park AVP pipeline in a structured parking-lot scenario. The successful completion of the full maneuver

 doi:10.20944/preprints202604.2014.v1

indicates that the perception-triggered switching logic, the NMPC cruising controller, and the TD3 parking controller can operate coherently within the proposed end-to-end AVP framework.

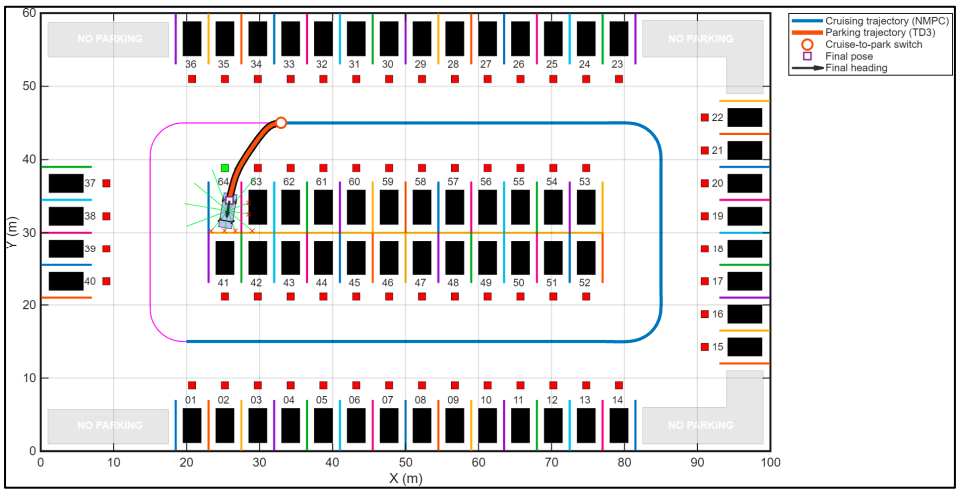

Figure 8. Representative cruise-to-park trajectory for slot 64.

3.2. NMPC Cruising Results

The cruising-stage tracking performance of the proposed framework was evaluated using the representative test for slot 64. During the cruising stage, the NMPC controller guided the vehicle along the predefined route before the parking maneuver was triggered. The lateral tracking error and heading error remained bounded throughout the approach phase, indicating that the vehicle entered the parking stage with a well-aligned pose for the subsequent TD3 parking controller.

Figure 9 shows the lateral tracking error during the NMPC cruising stage. Since the cruising controller primarily aims to keep the vehicle aligned with the reference path, the evaluation focuses on lateral deviation and heading alignment rather than longitudinal tracking error. The lateral error remained bounded within a small range throughout the approach phase, with larger deviations mainly appearing during curved segments and local geometric transitions of the route. Quantitatively, the cruising-stage tracking remained accurate, with a maximum lateral deviation of 0.0369 m and a mean absolute lateral error of 0.0051 m. These transient peaks were limited in magnitude and decayed quickly after the vehicle completed the corresponding turning adjustments, demonstrating effective path regulation under the customized NMPC formulation.

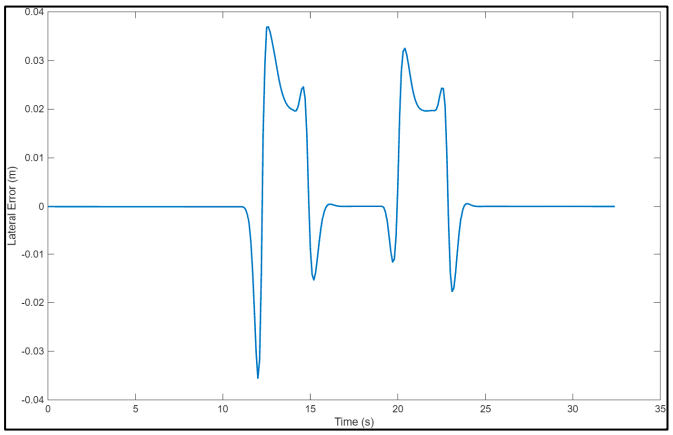

Figure 9. Lateral tracking error during NMPC cruising for slot 64.

Figure 10 presents the heading error of the vehicle relative to the local path tangent during cruising. The heading error remained well controlled overall and was mainly concentrated around

 doi:10.20944/preprints202604.2014.v1

the same turning regions where lateral correction was required. Although several short-duration peaks appeared during curvature changes, the error quickly returned toward zero after each transition, indicating that the controller regulated the vehicle orientation without persistent oscillation. The final yaw error at the end of the cruising phase was approximately zero, further confirming that the vehicle entered the parking stage with a well-aligned pose.

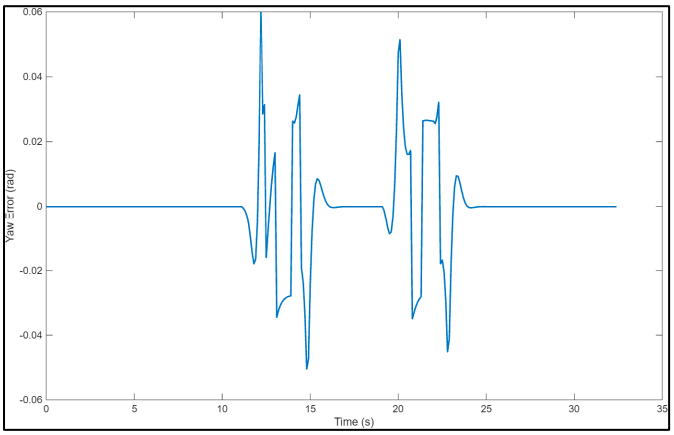

Figure 10. Heading error during NMPC cruising for slot 64.

Overall, the NMPC controller achieved accurate and well-behaved cruising motion before the phase switch to parking. The bounded lateral deviation and rapidly recovering heading response indicate that the controller provided a stable pre-parking trajectory, which is important for ensuring a smooth handoff to the TD3 parking controller in the integrated AVP framework.

3.3. TD3 Parking Results

The parking-stage performance of the proposed framework was evaluated using the TD3 training process and the terminal parking behavior in the training slot. Since the parking policy was trained in slot 7, this section focuses on the learned parking behavior in that slot before the generalization results on unseen slots are presented in Section 3.4.

Figure 11 shows the TD3 training reward curve for parking policy learning. As training progressed, the episode reward exhibited an overall increasing trend, while the average reward and evaluation statistics also improved gradually. Training was terminated after 850 episodes once the stopping criterion was satisfied. In the implemented training setup, the evaluation statistic was computed every 20 episodes as the mean episode reward over the evaluation window, and training stopped when this value exceeded 140. At termination, the final average reward was 89.0592 and the evaluation statistic reached 143.201. These results indicate that the agent progressively learned more effective parking behaviors during training. The stabilization of the reward trend in the later training stage suggests that the proposed reward formulation provided informative learning signals and supported convergence of the parking policy.

The learned policy was further examined in the training slot to evaluate its terminal docking performance. As shown in Figure 12, the vehicle successfully executed the parking maneuver in slot 7 and reached the target parking region with stable low-speed behavior. The terminal trajectory remained smooth, and no obvious oscillatory correction or unstable spinning behavior was observed near the goal, indicating that the TD3 controller was able to generate effective steering actions for the final docking stage.

Quantitatively, the parking maneuver in slot 7 was completed in 6.50 s. The final position error was 0.6270 m, including a lateral error of -0.5615 m and a longitudinal error of -0.2791 m, while the final yaw error was 0.0534 rad (3.06 deg). In addition, the trajectory-wide RMS distance-to-goal during parking was 7.8054 m and the maximum distance-to-goal was 12.7944 m, reflecting the large initial offset before convergence to the target pose. These results indicate that the learned policy

 doi:10.20944/preprints202604.2014.v1

achieved successful and stable terminal parking in the training slot, with good final heading alignment and reasonable terminal accuracy.

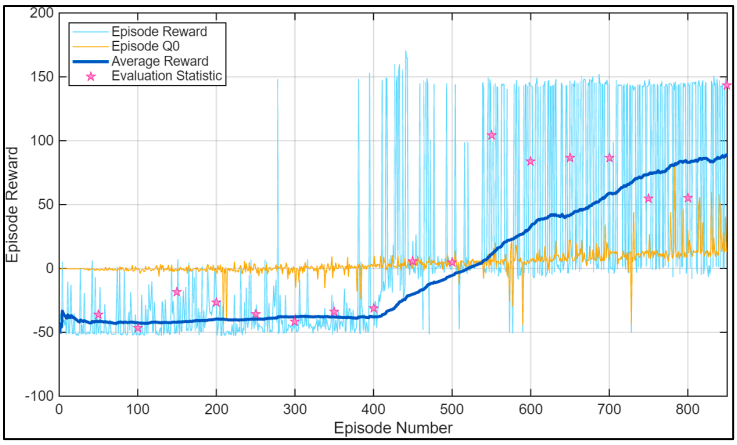

Figure 11. TD3 training reward curve for parking policy learning in slot 7.

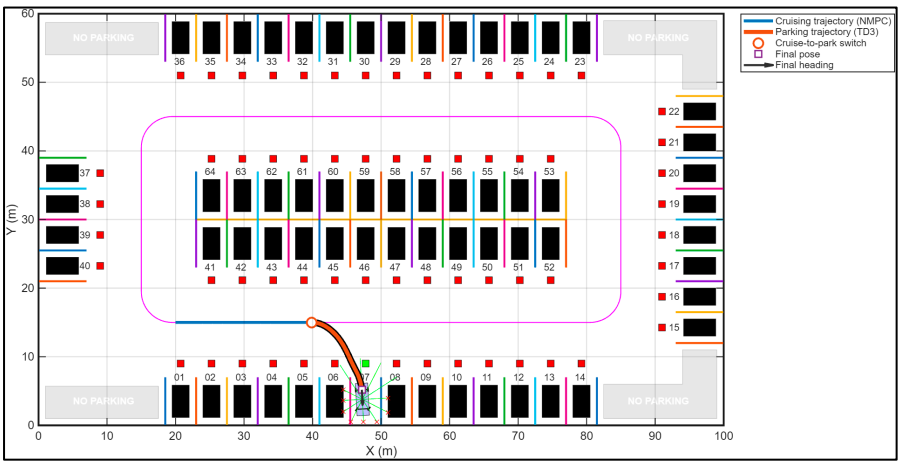

Figure 12. Parking trajectory and final pose in training slot 7.

Overall, the TD3 training and parking results confirm that the learned parking policy can provide a reliable low-speed docking controller within the proposed cruise-to-park AVP framework. Together with the designed reward shaping strategy, the controller enables stable parking behavior in the trained scenario and establishes the basis for the unseen-slot validation study presented in the next section.

3.4. Validation in Unseen Slots

The generalization capability of the learned parking policy was validated on six previously unseen parking slots, namely Slots 14, 15, 23, 39, 47, and 64. All tests were conducted using the same controller settings as in the training slot, and no additional retraining or parameter adjustment was performed. This evaluation was intended to examine whether the learned TD3 parking policy could be transferred to new terminal parking scenarios within the same structured parking-lot environment.

As summarized in Table 5, the learned policy completed feasible docking maneuvers in all six unseen slots under the same fixed controller settings. The parking duration ranged from 6.4 s to 7.4 s. The final position error remained below 0.75 m in Slots 14, 15, 23, and 39, with corresponding final yaw errors of 0.1164 rad, 0.0362 rad, 0.0803 rad, and 0.0880 rad, respectively. More challenging but still successful cases were observed in Slots 47 and 64, where the final position errors increased to

 doi:10.20944/preprints202604.2014.v1

1.3380 m and 1.1849 m, respectively. These two cases did not correspond to parking failure. Instead, the controller still guided the vehicle into the designated parking region with stable terminal behavior and acceptable final orientation, while exhibiting larger terminal offsets than the other validation-slot tests. This indicates that the main limitation in these cases was reduced terminal accuracy rather than loss of parking feasibility. Over all six validation-slot tests, the mean final position error was 0.8748 m and the mean final yaw error was 0.0818 rad (4.69 deg), indicating that the learned policy retained encouraging transfer performance for the validation parking conditions that were not part of the training.

Table 5. Parking performance in validation slots under fixed controller settings

Slot	Duration (s)	Position Error (m)	Lateral Error (m)	Longitudinal Error (m)	Yaw Error (rad)
14	6.4	0.7403	-0.5601	-0.4841	0.1164
15	6.8	0.6573	-0.3459	-0.5589	0.0362
23	6.5	0.6626	-0.5036	-0.4305	0.0803
39	6.4	0.6655	-0.5504	-0.3740	0.0880
47	7.4	1.3380	1.0361	-0.8466	0.0077
64	6.7	1.1849	-0.7824	-0.8898	0.1623

Figure 13 shows the local parking trajectories and final poses in the six validation slots. Although the target locations and approach geometries differed across cases, the learned TD3 controller generated feasible docking trajectories without retraining. The larger terminal offsets in Slots 47 and 64 suggest that terminal accuracy remains sensitive to local approach geometry, but these cases still achieved stable docking within the designated parking region. Overall, the validation results indicate that the policy learned transferable parking behavior within the structured parking-lot environment rather than memorizing the training slot.

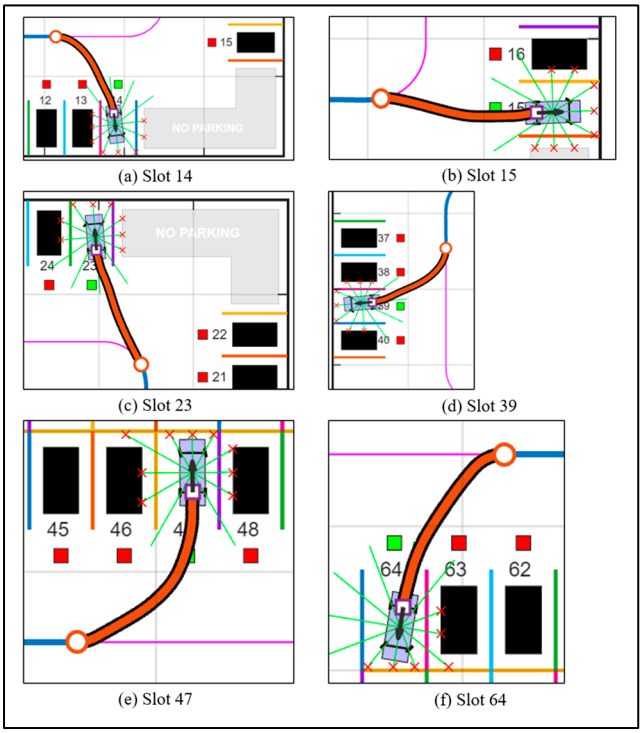

Figure 13. Parking trajectories and final poses in unseen slots 14, 15, 23, 39, 47, and 64.

4. Discussion

The results support the effectiveness of separating the AVP task into long-horizon cruising and terminal parking. In this structure, NMPC provides a constraint-aware tracking layer before the

 doi:10.20944/preprints202604.2014.v1

switching point, while TD3 focuses on the nonlinear low-speed maneuver near the target slot. This division reduces the burden on a single controller and allows each component to operate within a more suitable regime. The observed cruise-to-park behavior suggests that such a hierarchical design can provide a practical baseline for integrated AVP studies in structured parking-lot environments.

The reward design also played an important role in shaping the parking behavior. By adding a per-step time penalty, the TD3 agent was encouraged to complete the maneuver more decisively rather than remaining near the target with inefficient corrections. In the reported tests, no obvious repeated spinning behavior was observed near the goal, suggesting that the time-penalized reward helped suppress locally inefficient behaviors while preserving stable convergence. This finding is consistent with prior studies showing that reinforcement-learning-based parking performance is highly sensitive to reward formulation and task structure [27,28].

The generalization results further suggest that the learned policy did not merely memorize the training slot. Under fixed controller settings and without retraining, the controller maintained feasible docking behavior across all validation slots. However, the variation in terminal position error indicates that parking accuracy remains sensitive to target-slot geometry and relative pose at the switching point. This suggests that the learned docking strategy captured transferable behavior within the structured parking-lot environment, while further improvements are still needed to reduce slot-dependent terminal offsets.

5. Conclusions

This study presented a hierarchical cruise-to-park automated valet parking framework that combines an NMPC-based controller for the cruising phase with a TD3-based controller for the terminal parking phase in a unified Simulink environment. The objective was to achieve a smooth transition from route following to final parking while maintaining stable control performance throughout the maneuver.

The simulation results showed that the proposed framework was able to complete the full parking task under the designed parking-lot setting. During the cruising phase, the NMPC controller kept the vehicle close to the reference path and provided suitable initial conditions for the parking stage. After the switching condition was satisfied, the TD3 controller completed the terminal maneuver without obvious discontinuity at the transition point. In addition to the training slot, the same controller configuration was validated on six previously unseen target slots within the same parking-lot layout, where feasible parking behavior was also obtained without retraining. These results indicate that the framework provides a degree of intra-lot transferability within the tested environment.

Although the current study was limited to simulation and a fixed parking-lot layout, it provides a structured basis for further development of end-to-end AVP systems. Future work can extend the framework by introducing sensor noise, actuator delay, dynamic obstacles, and more diverse parking scenarios, as well as by evaluating its performance in higher-fidelity simulation [29,30], or real-vehicle experiments [31]. It is also planned to use robust parameter space control design for the parking space seeking path tracking in the future [32–36].

Author Contributions: Conceptualization, all authors; methodology, D.T.; software, D.T.; validation, D.T.; formal analysis, all authors; investigation, all authors; resources, all authors; data curation, D.T.; writing—original draft preparation, D.T.; writing—review and editing, L.G.; visualization, all authors; supervision, L.G.; project administration, L.G.; funding acquisition, all authors. All authors have read and agreed to the published version of the manuscript.

Funding: No funding was received for this study.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: The authors thank the Automated Driving Lab at the Ohio State University.

 doi:10.20944/preprints202604.2014.v1

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Khalid, M.; Wang, K.; Aslam, N.; Cao, Y.; Ahmad, N.; Khan, M.K. From Smart Parking towards Autonomous Valet Parking: A Survey, Challenges and Future Works. *J. Netw. Comput. Appl.* **2021**, *175*, 102935. https://doi.org/10.1016/j.jnca.2020.102935
2. Paden, B.; Čáp, M.; Yong, S.Z.; Yershov, D.; Frazzoli, E. A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Trans. Intell. Veh.* **2016**, *1*, 33–55. https://doi.org/10.1109/TIV.2016.2578706
3. Liu, W.; Li, Z.; Li, L.; Wang, F.-Y. Parking Like a Human: A Direct Trajectory Planning Solution. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 3388–3397. https://doi.org/10.1109/TITS.2017.2687047
4. Kiran, B.R.; Sobh, I.; Talpaert, V.; Mannion, P.; Sallab, A.A.A.; Yogamani, S.; Pérez, P. Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 4909–4926. https://doi.org/10.1109/TITS.2021.3054625
5. Jang, C.; Kim, C.; Lee, S.; Kim, S.; Lee, S.; Sunwoo, M. Re-Plannable Automated Parking System with a Standalone Around View Monitor for Narrow Parking Lots. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 777–790. https://doi.org/10.1109/TITS.2019.2891665
6. Qin, Z.; Chen, X.; Hu, M.; Chen, L.; Fan, J. A Novel Path Planning Methodology for Automated Valet Parking Based on Directional Graph Search and Geometry Curve. *Robot. Auton. Syst.* **2020**, *132*, 103606. https://doi.org/10.1016/j.robot.2020.103606
7. Shi, J.; Li, K.; Piao, C.; Gao, J.; Chen, L. Model-Based Predictive Control and Reinforcement Learning for Planning Vehicle-Parking Trajectories for Vertical Parking Spaces. *Sensors* **2023**, *23*, 7124. https://doi.org/10.3390/s23167124
8. Song, S.; Chen, H.; Sun, H.; Liu, M. Data Efficient Reinforcement Learning for Integrated Lateral Planning and Control in Automated Parking System. *Sensors* **2020**, *20*, 7297. https://doi.org/10.3390/s20247297
9. Li, B.; Acarman, T.; Zhang, Y.; Ouyang, Y.; Yaman, C.; Kong, Q.; Zhong, X.; Peng, X. Optimization-Based Trajectory Planning for Autonomous Parking with Irregularly Placed Obstacles: A Lightweight Iterative Framework. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 11970–11981. https://doi.org/10.1109/TITS.2021.3109011
10. Ren, H.; Niu, Y.; Li, Y.; Yang, L.; Gao, H. Automatic Parking Trajectory Planning Based on Warm Start Nonlinear Dynamic Optimization. *Sensors* **2025**, *25*, 112. https://doi.org/10.3390/s25010112
11. Li, B.; Yin, Z.; Ouyang, Y.; Zhang, Y.; Zhong, X.; Tang, S. Online Trajectory Replanning for Sudden Environmental Changes During Automated Parking: A Parallel Stitching Method. *IEEE Trans. Intell. Veh.* **2022**, *7*, 748–757. https://doi.org/10.1109/TIV.2022.3156429
12. Tang, X.; Yang, Y.; Liu, T.; Lin, X.; Yang, K.; Li, S. Path Planning and Tracking Control for Parking via Soft Actor-Critic Under Non-Ideal Scenarios. *IEEE/CAA J. Autom. Sin.* **2024**, *11*, 181–195. https://doi.org/10.1109/JAS.2023.123975
13. Alighanbari, S.; Azad, N. L. Deep Reinforcement Learning with NMPC Assistance Nash Switching for Urban Autonomous Driving. *IEEE Trans. Intell. Veh.* **2023**, *8*, 2604–2615. https://doi.org/10.1109/TIV.2022.3167616
14. Dang, F.; Chen, D.; Chen, J.; Li, Z. Event-Triggered Model Predictive Control with Deep Reinforcement Learning for Autonomous Driving. *IEEE Trans. Intell. Veh.* **2024**, *9*, 463–468. https://doi.org/10.1109/TIV.2023.3329785
15. Suhr, J.K.; Jung, H.G. End-to-End Trainable One-Stage Parking Slot Detection Integrating Global and Local Information. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 4570–4582. https://doi.org/10.1109/TITS.2020.3046039
16. Fujimoto, S.; van Hoof, H.; Meger, D. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, Stockholm, Sweden, 10–15 July 2018; *Proceedings of Machine Learning Research*; Volume 80, pp. 1587–1596. Available online: https://proceedings.mlr.press/v80/fujimoto18a.html
17. Jeng, S.-L.; Chiang, C. End-to-End Autonomous Navigation Based on Deep Reinforcement Learning with a Survival Penalty Function. *Sensors* **2023**, *23*, 8651. https://doi.org/10.3390/s23208651

 doi:10.20944/preprints202604.2014.v1

18. MathWorks. *Train PPO Agent for Automatic Parking Valet*; MATLAB & Simulink Documentation. Available online: https://www.mathworks.com/help/reinforcement-learning/ug/train-ppo-agent-for-automatic-parking-valet.html (accessed on 27 March 2026)
19. Kong, J.; Pfeiffer, M.; Schildbach, G.; Borrelli, F. Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design. *In Proceedings of the 2015 IEEE Intelligent Vehicles Symposium (IV)*, Seoul, Republic of Korea, 28 June–1 July **2015**; pp. 1094–1099. https://doi.org/10.1109/IVS.2015.7225830
20. Mayne, D.Q.; Rawlings, J.B.; Rao, C.V.; Scokaert, P.O.M. Constrained Model Predictive Control: Stability and Optimality. *Automatica* **2000**, *36*, 789–814. https://doi.org/10.1016/S0005-1098(99)00214-9
21. MathWorks. *Configure Optimization Solver for Nonlinear MPC*; MATLAB & Simulink Documentation. Available online: https://www.mathworks.com/help/mpc/ug/configure-optimization-solver-for-nonlinear-mpc.html (accessed on 27 March 2026).
22. Al-Mousa, A.; Arrabi, A.; Daoud, H. *A Reinforcement Learning-Based Reverse-Parking System for Autonomous Vehicles.* IET Intell. Transp. Syst. **2025**, *19*, e12614. https://doi.org/10.1049/itr2.12614
23. Leng, B.; Yu, Y.; Liu, M.; Cao, L.; Yang, X.; Xiong, L. *Deep Reinforcement Learning-Based Drift Parking Control of Automated Vehicles.* Sci. China Technol. Sci. **2023**, *66*, 1152–1165. https://doi.org/10.1007/s11431-022-2273-5
24. Ng, A.Y.; Harada, D.; Russell, S.J. *Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping*. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999)*; **1999**; pp. 278–287. https://dl.acm.org/doi/10.5555/645528.657613
25. Knox, W.B.; Allievi, A.; Banzhaf, H.; Schmitt, F.; Stone, P. Reward (Mis)design for Autonomous Driving. *Artif. Intell.* **2023**, *316*, 103829. https://doi.org/10.1016/j.artint.2022.103829
26. Lim, W.; Lee, S.; Sunwoo, M.; Jo, K. Hierarchical Trajectory Planning of an Autonomous Car Based on the Integration of a Sampling and an Optimization Method. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 613–626. https://doi.org/10.1109/TITS.2017.2756099
27. Chai, R.; Liu, D.; Liu, T.; Tsourdos, A.; Xia, Y.; Chai, S. Deep Learning-Based Trajectory Planning and Control for Autonomous Ground Vehicle Parking Maneuver. *IEEE Trans. Autom. Sci. Eng.* **2023**, *20*, 1633–1647. https://doi.org/10.1109/TASE.2022.3183610
28. Zhang, J.; Chen, H.; Song, S.; Hu, F. Reinforcement Learning-Based Motion Planning for Automatic Parking System. *IEEE Access* **2020**, *8*, 154485–154501. https://doi.org/10.1109/ACCESS.2020.3017770
29. Guvenc, L.; Aksun-Guvenc, B.; Zhu, S.; Gelbal, S.Y. Autonomous Road Vehicle Path Planning and Tracking Control. Wiley, IEEE Press. **2022**.
30. Cao, X.; Chen, H.; Gelbal, S.Y.; Aksun-Guvenc, B.; Guvenc, L. Vehicle-in-Virtual-Environment (VVE) Method for Autonomous Driving System Development, Evaluation and Demonstration. *Sensors* **2023**, *23*, 5088. https://doi.org/10.3390/s23115088
31. Wen, B.; Gelbal, S.Y.; Guvenc, B.A.; Guvenc, L. Localization and Perception for Control and Decision-Making of a Low-Speed Autonomous Shuttle in a Campus Pilot Deployment. *SAE Int. J. Connect. Autom. Veh.* **2018**, *1*, 53–66.
32. Necipoglu, S.; Cebeci, S.A.; Has, Y.E.; Guvenc, L.; Basdogan, C. Robust repetitive controller for fast AFM imaging. *IEEE Trans. Nanotechnol.* **2011**, *10*, 1074–1082.
33. Guvenc, L.; Guvenc, B.A.; Demirel, B.; Emirler, M.T. *Control of Mechatronic Systems*; The Institution of Engineering and Technology: London, UK, 2017.
34. Wang, H.; Gelbal, S.Y.; Guvenc, L. Multi-objective digital PID controller design in parameter space and its application to automated path following. *IEEE Access* 2021, *9*, 46874–46885.
35. Zhu, S.; Wang, J.; Yang, Y.; Aksun-Guvenc, B. Stability of Local Trajectory Planning for Level-2+ Semi-Autonomous Driving without Absolute Localization. *Electronics* 2024, *13*, 3808.
36. Zhu, S.; Aksun-Guvenc, B. Trajectory planning of autonomous vehicles based on parameterized control optimization in dynamic on-road environments. *J. Intell. Robot. Syst.* 2020, *100*, 1055–1067.