algorithms

MDPI

Article

A Geometry-Aware Segmented Deep Reinforcement Learning Method for Speed Control in Airport Surface Taxiing

Jiuxia Guo *, Zihao Ren, Yaqian Du, Jingyang Huang and Pengcheng Dan

School of Air Traffic Management, Civil Aviation Flight University of China, Guanghan 618307, China; renzh168@cafuc.edu.cn (Z.R.); duyaqian@cafuc.edu.cn (Y.D.); huangjy@cafuc.edu.cn (J.H.); danpengcheng@cafuc.edu.cn (P.D.)

* Correspondence: jxguo@cafuc.edu.cn

Abstract

Aircraft taxiing speed control along predefined airport surface routes is a constrained single-aircraft longitudinal control problem involving heterogeneous route geometry, action smoothness, and terminal velocity feasibility. Existing learning-based taxiing controllers commonly use a unified policy for the whole route, which may be insufficient for handling straight-segment propulsion, curved-segment speed regulation, and action discontinuities near straight–curve transitions. This paper proposes SegCoord-Taxi, a geometry-aware segmented deep reinforcement learning framework for taxiing speed control. The route is decomposed into straight segments, curved segments, and transition boundary zones. A Straight-Segment Policy (SSP) and a Curved-Segment Policy (CSP) generate geometry-dependent base acceleration commands, a Switch Residual Adapter (SRA) provides local residual correction near transition regions, and a Route-Level Feasibility Projection (RFP) maps the coordinated action into an executable acceleration satisfying route-level feasibility constraints. Experiments on departure taxiing routes at Chengdu Tianfu International Airport (ZUTF) included baseline comparison, ablation analysis, projection diagnostics, sensitivity analysis, and a trajectory-level case study. On the evaluated ZUTF case-study routes, SegCoord-Taxi achieves the lowest final velocity on the test set, 0.336 ± 0.017 m/s, compared with 0.732 ± 0.061 m/s for the unified Proximal Policy Optimization (PPO) controller and 0.586 m/s for the curvature-aware constrained optimizer. The complete framework also reduces switch action jump from 1.022 ± 0.017 m/s^2 to 0.429 ± 0.004 m/s^2 in the ablation study. These results indicate improved terminal feasibility and transition-region smoothness in the evaluated single-airport case-study setting under an explicit efficiency–smoothness–feasibility trade-off. Future work will extend the framework to multi-aircraft and multi-airport settings under operational uncertainty.

Keywords: aircraft taxiing speed control; deep reinforcement learning; segmented policy; geometry-aware control; residual adaptation; feasibility projection; terminal velocity constraint; action smoothness

Check for updates

Academic Editor: Feng-Cheng Lin

Received: 13 May 2026
Revised: 14 June 2026
Accepted: 16 June 2026
Published: 20 June 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Airport surface taxiing connects stands, taxiways, runway entrances, arrival and departure procedures, and constitutes a critical phase in which an aircraft transitions from ground operations to airborne operations. Unlike en-route flight, taxiing is typically characterized by low speed, low thrust, frequent acceleration and deceleration, and strong route geometric constraints. Its operating state is jointly affected by the airport surface

network structure, taxiing route geometry, runway-end capacity, and air traffic control rules. Previous studies have shown that airport surface congestion, taxiing time, stop frequency, and speed variations affect fuel consumption, emissions, and operational efficiency during ground operations [1–7]. Therefore, generating a smooth and executable speed profile that satisfies both local route-geometry constraints and terminal velocity feasibility along a predefined taxiing route is an important continuous-control problem in intelligent airport surface operations.

Existing studies on airport surface operation optimization have evolved from macroscopic scheduling toward route–speed coordination and learning-based control. Pushback control, departure metering, and surface congestion management mainly aim to reduce engine-on waiting time and runway-end queuing at the system level [1–5]. Route planning, scheduling optimization, and speed-profile generation methods use graph modeling, multi-objective optimization, and uncertainty modeling to balance taxiing time, operating cost, and conflict avoidance [8–16]. In recent years, data-driven models, graph neural networks, visual perception, and deep learning methods have also been applied to taxiing time prediction, surface-state modeling, trajectory analysis, and rapid route assignment. In addition, imitation learning and reinforcement learning have begun to be used for taxiing speed modeling and autonomous taxiing control, providing a new technical pathway for continuous-action decision-making. However, most existing learning-based taxiing controllers still adopt a unified policy for the entire route or jointly address route selection and speed regulation. They rarely explicitly distinguish the different longitudinal-control requirements of straight segments, curved segments, and straight–curve transition regions.

1.1. Research Motivation and Gap

Although these studies provide an important foundation, three limitations remain for single-aircraft longitudinal speed control along a predefined taxiing route. First, macroscopic scheduling and route-planning methods mainly optimize traffic flow, routing, or timing decisions, and do not directly generate closed-loop acceleration commands for a single aircraft. Second, unified learning-based policies must handle straight-segment propulsion, curved-segment deceleration, and transition-region action continuity within one policy, which may weaken geometry-specific control. Third, a purely learning-based acceleration policy does not inherently guarantee terminal velocity feasibility in a complete route rollout. Even when the route is completed, the final speed may remain too high without an additional route-level feasibility mechanism.

To address these problems, this paper proposes SegCoord-Taxi, a geometry-aware segmented deep reinforcement learning framework for speed control along predefined airport taxiing routes. Based on route geometric characteristics, the framework divides the taxiing process into straight segments, curved segments, and switching boundary regions. The final acceleration command is generated collaboratively by the Straight-Segment Policy (SSP), Curved-Segment Policy (CSP), Switch Residual Adapter (SRA), and Route-Level Feasibility Projection (RFP). SSP is responsible for efficient propulsion on low-curvature straight segments, CSP uses look-ahead geometric information for curvature-aware control on curved segments, SRA performs residual correction for local action discontinuities near straight–curve transitions, and RFP maps the coordinated learning-based action into an executable acceleration command that satisfies action-rate consistency, non-terminal propulsion, and terminal-speed feasibility constraints.

1.2. Main Contributions

The main contributions of this paper are fourfold. First, a geometry-aware segmented speed-control framework is proposed for predefined airport taxiing routes, explicitly sepa-

rating straight-segment propulsion, curved-segment speed regulation, transition-region residual correction, and route-level feasibility handling. Second, heterogeneous policy structures are designed for different geometric conditions: SSP uses a Proximal Policy Optimization policy with a multilayer perceptron structure (PPO-MLP) for low-curvature straight segments, while CSP uses a Proximal Policy Optimization policy with a temporal convolutional network structure (PPO-TCN) and look-ahead geometric information for curved segments. Third, a Switch Residual Adapter is introduced to reduce action discontinuities caused by hard switching between straight and curved policies without replacing the base actions. Fourth, Route-Level Feasibility Projection is incorporated as a deterministic execution layer to enforce action-rate, non-terminal propulsion, and terminal-speed feasibility constraints during full-route rollout.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces data processing, Markov decision process (MDP) modeling, and the SegCoord-Taxi framework. Section 4 presents the experimental settings, baseline comparison, ablation study, projection diagnostics, sensitivity analysis, and trajectory-level case study. Section 5 concludes the paper and discusses limitations and future work.

2. Related Work

Research on airport surface operations has developed from system-level traffic management to route–speed coordination and learning-based taxiing control. The existing literature can be broadly grouped into four streams: airport surface operation optimization, route planning and speed-profile generation, data-driven surface-state prediction, and learning-based taxiing control.

Early studies on airport surface operation optimization mainly focused on congestion management, pushback control, departure metering, and emission analysis. Simaiakis et al. [1] analyzed the effects of surface congestion on taxiing time, fuel burn, and emissions. Nikoleris et al. [2] estimated fuel consumption and emissions during taxiing operations. Ashok et al. [3] evaluated emission-reduction measures during pushback, taxiing, and takeoff. Later studies further investigated departure metering, airport surface congestion mechanisms, adaptive taxiing rule management, and emission-oriented taxiing schemes [4–7]. These studies demonstrate the operational importance of surface-movement optimization. However, their main focus is system-level traffic flow, queue management, operating rules, or environmental assessment, rather than closed-loop longitudinal acceleration control for a single aircraft along a predefined route.

Route planning, scheduling optimization, and speed-profile generation constitute another important research direction. Tancredi et al. [8] modeled airport surface movement as a route-generation and conflict-resolution problem. Chen et al. [9,10] proposed active-routing methods that integrate speed profiles into airport ground movement optimization. Zhang et al. [11] developed an online speed-profile generation approach, and Weiszer et al. [12] formulated airport ground movement as a multi-objective routing and scheduling problem. Related studies also addressed taxi-time uncertainty, hybrid optimization, real-time route assignment, and four-dimensional ground movement concepts [13–16]. In addition, Cheng and Sweriduk [17] investigated aircraft taxi trajectory design under trajectory-based surface operations. Their work showed that, even when a taxi route and timing constraints are specified, the velocity and acceleration profiles still have design freedom and can be adjusted to improve comfort, fuel burn, emissions, and noise. These studies place taxiing speed-profile generation within a constrained trajectory-design problem along predefined airport surface routes. Most route–speed optimization methods still rely on predefined speed phases, offline optimization, reference trajectories, or externally

specified constraints, and they generally do not learn geometry-dependent closed-loop acceleration policies from route states.

Data-driven methods have also been widely used for airport surface-state prediction, taxiing time estimation, trajectory analysis, and visual surveillance. Maji et al. [18] used large-scale trajectory data for airport ground movement analysis and optimization. Ravizza et al. [19], Wang et al. [20], Park and Kim [21], Yin et al. [22], Qiao et al. [23], Yuan et al. [24], and Okwir et al. [25] investigated taxiing time prediction from different perspectives, including model comparison, feature importance, influencing-factor analysis, similar-scenario matching, CNN-GRU modeling, and machine-learning benchmarking. Graph-based and vision-based studies have also been applied to surface movement prediction, safety assessment, variable taxi-time prediction, visual surveillance, and taxiway analysis datasets [26–29]. These studies improve airport surface perception, prediction, and decision support. However, most of them remain predictive or surveillance-oriented and do not directly output executable longitudinal acceleration commands.

Learning-based methods have gradually been introduced into taxi-speed modeling and autonomous taxiing control. Pham et al. [30] used generative adversarial imitation learning to model realistic aircraft taxi-speed behavior and showed that simplified speed assumptions may not adequately represent real taxiing variability. Tran et al. [31] formulated autonomous taxiing as a reinforcement learning problem and used PPO to learn a unified taxiing speed-control policy. Xiang et al. [32] applied improved Q-learning to aircraft path planning at airports. Yang et al. [33] further considered multi-agent reinforcement learning for autonomous taxiing. These studies show the potential of learning-based methods for nonlinear decision-making in airport taxiing. However, existing reinforcement-learning taxiing controllers usually adopt a unified policy for the whole taxiing process or focus on path selection and multi-aircraft coordination. They rarely explicitly decompose the longitudinal speed-control task into straight-segment propulsion, curved-segment speed regulation, transition-region residual adaptation, and deterministic terminal-feasibility enforcement.

Learning-based speed regulation has also been discussed in adjacent road-traffic control research. Rhanizar and El Akkaoui [34] reviewed learning-based variable speed limit control strategies and summarized the development from single-agent RL and DRL to multi-agent reinforcement learning for speed regulation. Although variable speed limit control in road traffic differs from aircraft taxiing, this literature is relevant because it highlights the use of learning-based speed-control policies to balance mobility, safety, and environmental objectives. It also provides useful methodological context for future extensions of aircraft taxiing speed control from single-aircraft scenarios to multi-aircraft cooperative control.

Table 1 summarizes representative studies related to airport surface optimization, speed-profile generation, data-driven modeling, and learning-based taxiing control, and positions SegCoord-Taxi within these research streams.

Taken together, prior studies have provided foundations for airport surface optimization, route–speed profile generation, data-driven prediction, and learning-based taxiing control. For predefined-route single-aircraft speed control, the remaining gap lies in the integrated treatment of route-geometry segmentation, local acceleration control, switch-region smoothness, and terminal feasibility. SegCoord-Taxi addresses this gap by assigning these functions to SSP/CSP, SRA, and RFP within a unified closed-loop control framework.

Table 1. Comparison with representative related studies.

Study	Research Focus	Speed-Control Approach and Geometry/Feasibility Treatment	Relation to SegCoord-Taxi
Cheng and Sweriduk [17]	Aircraft taxi trajectory design under trajectory-based surface operations	Designs reference velocity and acceleration profiles under timing constraints, acceleration limits, comfort, fuel burn, emissions, and noise considerations	Establishes taxi speed-profile generation as a constrained trajectory-design problem; SegCoord-Taxi further studies closed-loop segmented acceleration control along predefined routes
Chen et al. [9,10]; Zhang et al. [11]	Airport ground movement and route–speed profile optimization	Generates or optimizes speed profiles through graph-based, mathematical, or online optimization models	Provides route–speed optimization foundations; SegCoord-Taxi focuses on geometry-conditioned local acceleration control during closed-loop rollout
Pham et al. [30]	Realistic aircraft taxi-speed modeling	Uses imitation learning to model realistic taxi-speed behavior from data	Supports the need for data-driven taxi-speed modeling; SegCoord-Taxi extends this direction from behavior modeling to executable acceleration control
Tran et al. [31]	Autonomous taxiing speed control	Uses a unified PPO policy for taxiing speed control over the whole route	Provides the most directly comparable learning-based baseline; SegCoord-Taxi separates straight, curved, and transition-region control through SSP, CSP, and SRA
Rhanizar and El Akkaoui [34]	Learning-based variable speed limit control in road traffic	Reviews RL, DRL, and MARL strategies for learning-based speed regulation with mobility, safety, and environmental objectives	Offers adjacent-domain methodological context for learning-based speed control and future multi-agent extensions
Yang et al. [33]	Multi-aircraft autonomous taxiing	Applies multi-agent reinforcement learning to autonomous aircraft taxiing with time, fuel, and emission objectives	Relates to future multi-aircraft extension; the present study focuses on single-aircraft speed control along predefined routes
SegCoord-Taxi	Aircraft taxiing speed control along predefined airport routes	Combines SSP/CSP segmented DRL, SRA residual adaptation, and RFP deterministic feasibility projection	Provides geometry-dependent base control, local switching correction, and route-level terminal feasibility enforcement

3. Materials and Methods

3.1. Study Scenario and Dataset

This study takes the departure taxiing process at Chengdu Tianfu International Airport (ZUTF) as the research scenario and focuses on speed-profile optimization for a single aircraft along a predefined taxiing route. Unlike airport-wide route planning or multi-aircraft cooperative scheduling, this study assumes that the taxiing route from the stand to the runway entrance has already been determined. The task of the model is to generate a longitudinal acceleration control sequence under this route constraint, thereby forming a taxiing speed profile that satisfies operational efficiency, smoothness, curve safety, and terminal velocity feasibility.

In current airport surface operations, taxiing routes and movement clearances are usually determined by airport operating procedures and air traffic control instructions, while the longitudinal speed profile is mainly regulated by the pilot according to taxiway speed restrictions, route geometry, traffic spacing, runway-entry holding requirements, and operational experience. This practice is flexible and safety-oriented, but it does not provide an explicit closed-loop acceleration sequence for the entire predefined route. Therefore, it is difficult to systematically coordinate efficiency, curve-related speed regulation, transition-region smoothness, and terminal stopping feasibility. The proposed method is not intended to replace air traffic control or airport-wide surface management. Instead, it is designed as a route-constrained speed-profile generation and decision-support method that provides a smooth and executable longitudinal acceleration sequence after the taxiing route has been assigned.

The case-study dataset combines a processed OpenStreetMap-based ZUTF surface-network topology with departure routes reconstructed from Feichangzhun website flight information and trajectory-playback images. The surface network was organized as a directed node–edge graph, and each route was matched through stand positions, taxiway connectivity, runway-entry directions, and route consistency checks. This processing produces fixed stand-to-runway taxiing routes for the subsequent closed-loop speed-control experiments.

The airport surface network is represented as a directed topological graph composed of nodes and edges. Nodes correspond to taxiway intersections, stand connection points, points near runway entrances, or other topologically meaningful locations, while edges correspond to traversable taxiway segments between adjacent nodes. The surface-network visualization in Figure 1 was generated from the processed directed node–edge graph. Taxiway centerline edges were reconstructed from the topology data, and a representative stand-to-runway departure route was overlaid according to the matched route sequence. This figure illustrates the processed ZUTF taxiway network, the representative departure taxiing route, and the predefined route constraint considered in this study.



Figure 1. ZUTF surface taxiway network and a representative departure taxiing route. The grey lines denote the surface taxiway network, the red line denotes the selected taxiing route, the blue points denote topology nodes along the route, and the green and purple markers denote the start and end positions, respectively.

After route augmentation and unified processing, this study constructs a dataset containing 62 departure taxiing routes. The data split is performed at the route level rather than by randomly splitting sampling points, in order to avoid information leakage caused by adjacent sampling points from the same route appearing simultaneously in the training and test sets. Table 2 summarizes the route-level split and geometric characteristics of the processed dataset.

Table 2. Dataset split and route-level geometric statistics of the processed ZUTF departure taxiing routes.

Statistic	Training	Validation	Test	Total
No. of routes	43	9	10	62
No. of resampled points	27,011	5456	8121	40,588
Proportion of points (%)	66.55	13.44	20.01	100
Route length, mean ± std (m)	3133.6 ± 1198.4	3023.4 ± 1422.2	4053.3 ± 1453.1	3265.9 ± 1299.3
Route length, min–max (m)	1349.2–5763.0	1332.4–5743.1	1494.7–6173.4	1332.4–6173.4
Resampled points per route, mean ± std	628.2 ± 239.7	606.2 ± 284.4	812.1 ± 290.7	654.6 ± 259.9
Curved segments, mean ± std	4.2 ± 1.1	4.1 ± 0.9	4.5 ± 1.3	4.2 ± 1.1
Switching boundaries, mean ± std	8.1 ± 2.2	8.0 ± 1.9	8.7 ± 2.4	8.2 ± 2.1
Boundary-zone proportion, mean ± std (%)	22.11 ± 7.44	23.32 ± 9.50	18.00 ± 6.95	21.62 ± 7.74

The processed routes cover different route lengths and repeated geometric transitions within the ZUTF surface network. The total route length ranges from 1332.4 m to 6173.4 m, with an average of 3265.9 ± 1299.3 m. Each route contains 4.2 ± 1.1 curved segments and 8.2 ± 2.1 straight–curve switching boundaries on average. The boundary-zone proportion reaches $21.62 \pm 7.74\%$. These statistics indicate that transition-region control is a recurring part of the full-route speed-control task in the processed ZUTF dataset.

Based on the route-level split and geometric labels, the processed data are further organized into training samples for the subsequent segmented control framework. Straight-segment samples are used for training the straight-segment speed policy, curved-segment sequence samples are used for training the curved-segment speed policy, and boundary sequence samples near straight–curve transitions are used for switch residual adaptation. Boundary-zone samples are not treated as an independent base segment category; instead, they are superimposed near straight or curved segments to describe policy-switching regions.

3.2. Route Geometry Processing

To convert raw airport taxiing routes into a unified geometric representation suitable for speed-control modeling, this study first performs topological checking, geometric reconstruction, arc-length resampling, curvature calculation, and segment labeling. Raw routes are usually represented by node sequences in the airport surface network. However, the spacing between nodes is non-uniform, and the actual taxiway geometry between adjacent nodes may not be a simple straight line. Therefore, directly using the raw node sequence as model input may lead to inconsistent geometric scales and affect subsequent curvature estimation, curve identification, and look-ahead window construction.

First, this study checks the topological connectivity of the route node sequence and reconstructs the complete taxiing route using the edge geometry between adjacent nodes. This step preserves the actual taxiway alignment while ensuring that subsequent arc-length resampling and curvature calculation are based on continuous route geometry. Compared with simply connecting adjacent node coordinates, edge-geometry-based route reconstruction can more accurately represent gentle curves, turns, and local directional changes within taxiway segments.

After obtaining the continuous route geometry, this study calculates the cumulative arc length along the route direction. Let a route be represented by a continuous sequence of geometric points $\mathcal{P} = \{p_0, p_1, \ldots, p_N\}$, where $p_i = (x_i, y_i)$ denotes the planar coordinates of the i-th geometric point. The distance between adjacent points is defined as:

$$d_i = \| p_{i+1} - p_i \|_2 \tag{1}$$

where d_i denotes the Euclidean distance between p_i and p_{i+1}, and $\| \cdot \|_2$ denotes the L_2 norm. The Euclidean norm is used because the taxiway geometry is represented in a planar Cartesian coordinate system, and arc-length resampling requires the physical distance between adjacent geometric points. This treatment preserves the geometric scale of taxiway edges before curvature calculation and look-ahead window construction. On this basis, the cumulative arc length of the i-th point is given by:

$$s_i = \sum_{k=0}^{i-1} d_k \tag{2}$$

where s_i represents the cumulative distance traveled from the route origin to the i-th point along the taxiway geometry. Based on the cumulative arc-length coordinate, this study resamples the route at a fixed spatial interval so that different routes have a consistent geometric resolution. A 5 m arc-length interval is used to generate resampled route points. It should be noted that this spatial interval is used only for route geometric representation, feature calculation, and look-ahead window construction; it is not equivalent to the control step in the reinforcement learning environment. The control step is still determined by temporal discretization, and the actual displacement of the aircraft at each step is obtained by integrating the current speed and acceleration.

After resampling, this study calculates the local heading angle and curvature at each sampling point. The heading angle describes the tangential direction of the route at the current position, while curvature characterizes the degree of route bending. Since raw geometry and numerical differentiation may introduce local noise, the curvature sequence is further smoothed to reduce the influence of isolated noisy points on curve identification. The smoothed curvature is used as the main geometric basis for subsequent straight- and curved-segment classification.

The identification of straight and curved segments does not rely solely on the curvature threshold of individual sampling points. Instead, it jointly considers local curvature, heading variation, and continuous segment length. This treatment is necessary because airport taxiing routes contain many gentle bends, short transition regions, and local geometric disturbances. If only pointwise curvature thresholds are used, short straight transition segments near curves may be mislabeled as curved segments, and small curvature fluctuations caused by local noise may be mistaken for real curves. By combining curvature magnitude, local directional variation, and segment-level continuity constraints, this study obtains more stable base segment labels. Finally, each resampled point is labeled as either a straight segment or a curved segment, which serve as the base operating regions for SSP and CSP.

Near the switching locations between straight and curved segments, this study further constructs boundary zones to describe positions where segmented policies may produce action discontinuities. For each straight-to-curve or curve-to-straight switching point, a boundary region is defined within a certain distance on both sides of the switching point, and sampling points within this boundary region are assigned distance-decaying weights. Let the arc-length position of the switching point be s_b, the sampling point position be s, and the boundary half-width be D_b. The boundary weight can be expressed as:

$$w_b(s) = \max\left(0, 1 - \frac{|s - s_b|}{D_b}\right) \tag{3}$$

where $w_b(s)$ denotes the boundary weight at arc-length position s. This study sets $D_b = 40$ m. The weight reaches its maximum at the switching point and gradually

decreases as the sampling point moves away from the switching point. The linear decay function is adopted as a simple residual-gating form near straight–curve switching points. It provides a monotonic decrease in residual influence from the switching point to the boundary edge without introducing additional shape parameters. Alternative Gaussian and exponential boundary weighting functions are further examined in the sensitivity analysis in Section 4.5.2. The subsequent switch residual adaptation module uses this weight to modulate residual actions, so that residual correction is mainly concentrated in the switching regions between the Straight-Segment Policy and the Curved-Segment Policy. Figure 2 presents a representative example of the resulting geometric segmentation and boundary-zone construction, showing how the resampled route, smoothed curvature profile, base segment labels, and transition boundary zones are connected.

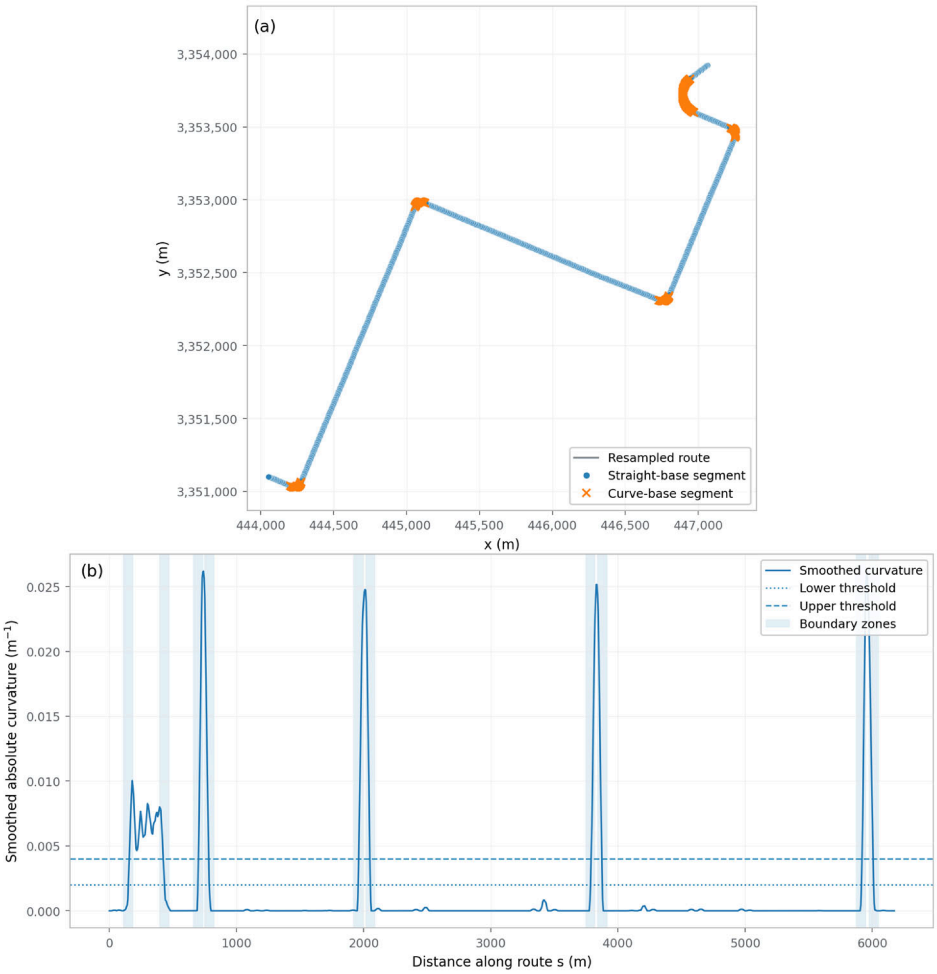


Figure 2. Geometry-based segment and boundary construction for taxi-route decomposition. (**a**) Straight and curved segment labels on a resampled taxiing route. The straight and curved segments are used as the base regions for SSP and CSP, respectively. (**b**) Smoothed absolute curvature profile along route arc length, with curvature thresholds and boundary zones around straight–curved segment transitions. The boundary zones are used to support SRA-based residual adaptation near policy-switching regions.

After the above processing steps, each route is converted into a unified arc-length sequence with local geometric features, base segment labels, and boundary-zone weights. These processed geometric features constitute the data basis of SegCoord-Taxi: straight-segment samples are used to train SSP, curved-segment sequences are used to train CSP, and boundary-zone sequences are used to train SRA.

3.3. Taxiing Speed-Control MDP

After route geometry processing, this study formulates aircraft taxiing speed control along a predefined route as a continuous-action Markov decision process. For each given taxiing route, the aircraft starts from the route origin and moves along the route direction within discrete control intervals. At each time step, the control policy outputs a longitudinal acceleration according to the current motion state and route geometric information. The environment then updates the aircraft speed and arc-length position based on kinematic equations. This process continues until the aircraft completes the route, stalls, exceeds the maximum number of steps, or triggers a safety-related termination condition.

Let the geometrically processed taxiing route be represented as:

$$\prod = \{q_0, q_1, \ldots, q_N\} \tag{4}$$

where Π denotes the geometrically processed taxiing route and q_i denotes the i-th arc-length-resampled point on the route. Each sampling point contains information such as arc-length position, planar coordinates, local heading angle, curvature, straight/curved segment label, and boundary-zone weight. The aircraft state at time step t can be represented as:

$$o_t = \left[x_t^{dyn}, x_t^{geo}, x_t^{seg}, G_t\right] \tag{5}$$

Here, x_t^{dyn} describes the current longitudinal motion state, including speed, acceleration history, jerk, progress, and terminal proximity; x_t^{geo} describes the geometric characteristics near the current position; x_t^{seg} represents segment semantic information such as straight/curved labels and boundary-zone indicators; and G_t provides the look-ahead geometric structure along the arc-length direction. The main components of the MDP observation state are summarized in Table 3.

Table 3. Components of the MDP observation state.

Component	Description	Typical Variables	Role in Control
x_t^{dyn}	Dynamic state	Current speed v_t (m/s), previous acceleration a_{t-1} (m/s^2), jerk j_t (m/s^3), route progress s_t/L (-), distance to route end d_t^{end} (m)	Describes longitudinal motion and terminal proximity
x_t^{geo}	Local geometric state	Heading angle ψ_t (rad), curvature κ_t (m^{-1}), absolute curvature $\lvert\kappa_t\rvert$ (m^{-1}), turning direction (-)	Provides local curve and heading information
x_t^{seg}	Segment semantic state	Straight/curved label z_t (-), boundary-zone indicator I_t^b (-), boundary weight $w_b(s_t)$ (-)	Identifies segment type and switching region
G_t	Look-ahead geometry window	Future curvature sequence (m^{-1}), segment labels (-), boundary information along arc length (m)	Supports anticipation of upcoming route geometry

The action space is a one-dimensional continuous longitudinal acceleration:

$$a_t \in [a_{\min}, a_{\max}] \tag{6}$$

where a_t denotes the longitudinal acceleration action at time step t, and $a_{\min}$ and $a_{\max}$ denote the minimum and maximum allowable acceleration bounds, respectively. Since the lateral path is fixed by the predefined taxiing route, the control variable is limited to one-dimensional longitudinal acceleration, and the geometric effect of lateral motion enters the MDP through route curvature and the speed-dependent lateral-acceleration proxy.

Let the control interval be Δt. The environment updates the speed at the next time step according to the current speed and action:

$$v_{t+1} = \max(0, v_t + a_t \Delta t) \tag{7}$$

The arc-length advancement of the aircraft within this control interval is approximated using trapezoidal integration:

$$\Delta s_t = \frac{1}{2}(v_t + v_{t+1})\Delta t \tag{8}$$

Therefore, the arc-length position at the next time step is:

$$s_{t+1} = s_t + \Delta s_t \tag{9}$$

where v_t and v_{t+1} denote the current and next speeds, respectively, Δs_t denotes the arc-length advancement within one control interval, and s_t and s_{t+1} denote the current and next arc-length positions. According to the updated arc-length position, the environment queries the corresponding curvature, heading angle, segment label, and boundary-zone information from the resampled route table. It should be emphasized that the route resampling interval is only the spatial resolution for geometric queries and look-ahead window construction; it is not equivalent to the aircraft displacement within one control interval. At each control step, the aircraft may pass over multiple resampled route points or remain within the same local segment, depending on its current speed and acceleration.

Curve safety is characterized by a curvature-related lateral-acceleration proxy. Let the curvature at the current position be κ_t. The lateral-acceleration proxy is defined as:

$$a_t^{lat} = v_t^2 |\kappa_t| \tag{10}$$

where a_t^{lat} denotes the lateral-acceleration proxy at time step t, v_t denotes the current speed, and κ_t denotes the curvature at the current position. When the aircraft travels at a relatively high speed in a high-curvature region, this value increases substantially, indicating that the speed profile may be insufficiently safe or insufficiently smooth in the curved region. This term is jointly determined by route geometry and current speed, making it more suitable for integration into a curvature-aware control framework than fixed rule-based speed limits. This quantity is a kinematic proxy derived from the relation between speed and curvature, where $|\kappa_t|$ corresponds to the reciprocal of the local turning radius. This proxy characterizes curvature-induced lateral motion pressure along the predefined taxiing path and is not intended as a certified passenger-comfort limit or a high-fidelity aircraft ground-dynamics model. In the learning environment, this proxy is introduced as a soft constraint, encouraging CSP to reduce speed in high-curvature regions while avoiding hard termination caused by local curvature fluctuations.

The environment termination conditions include smooth completion, high-speed completion, stalling, timeout, and unsafe termination. If the aircraft reaches the end of the route and its terminal speed is below the smooth-stop threshold v_{stop}, the episode is regarded as a smooth completion. If the aircraft reaches the route end with $v_T > v_{stop}$, the episode is recorded as a high-speed completion. This case indicates that the route has been completed but the terminal velocity is not feasible for smooth stopping. The threshold v_{stop} is used to evaluate terminal completion quality, while the terminal speed envelope used by RFP is defined separately in Section 3.4.5. Based on this MDP formulation, the following section introduces how SegCoord-Taxi assigns the longitudinal acceleration decision to SSP, CSP, SRA, and RFP according to route geometry and feasibility requirements.

3.4. SegCoord-Taxi Framework

This study proposes SegCoord-Taxi, a segmented cooperative speed-control framework for predefined airport taxiing routes. The framework consists of a Straight-Segment

Policy (SSP), a Curved-Segment Policy (CSP), a Switch Residual Adapter (SRA), and a Route-Level Feasibility Projection module (RFP).

SSP and CSP handle longitudinal speed control for low-curvature straight segments and high-curvature curved segments, respectively. SRA is used to correct action discontinuities produced by the two policies in switching regions, while route-level feasibility applies route-level constraints to the combined action. As shown in Figure 3, the overall process is as follows: the environment reads the route geometry and segment label according to the current arc-length position, and SSP or CSP outputs a base acceleration. Near straight–curve transitions, SRA predicts a residual acceleration and corrects the base action. The final action is then passed through route-level feasibility constraints before being input into the environment to update speed and position. In this framework, route-level feasibility is defined as a combination of deterministic constraints, including action-rate limits, terminal smooth stopping, and non-terminal low-speed feasibility, without introducing a new trainable policy.

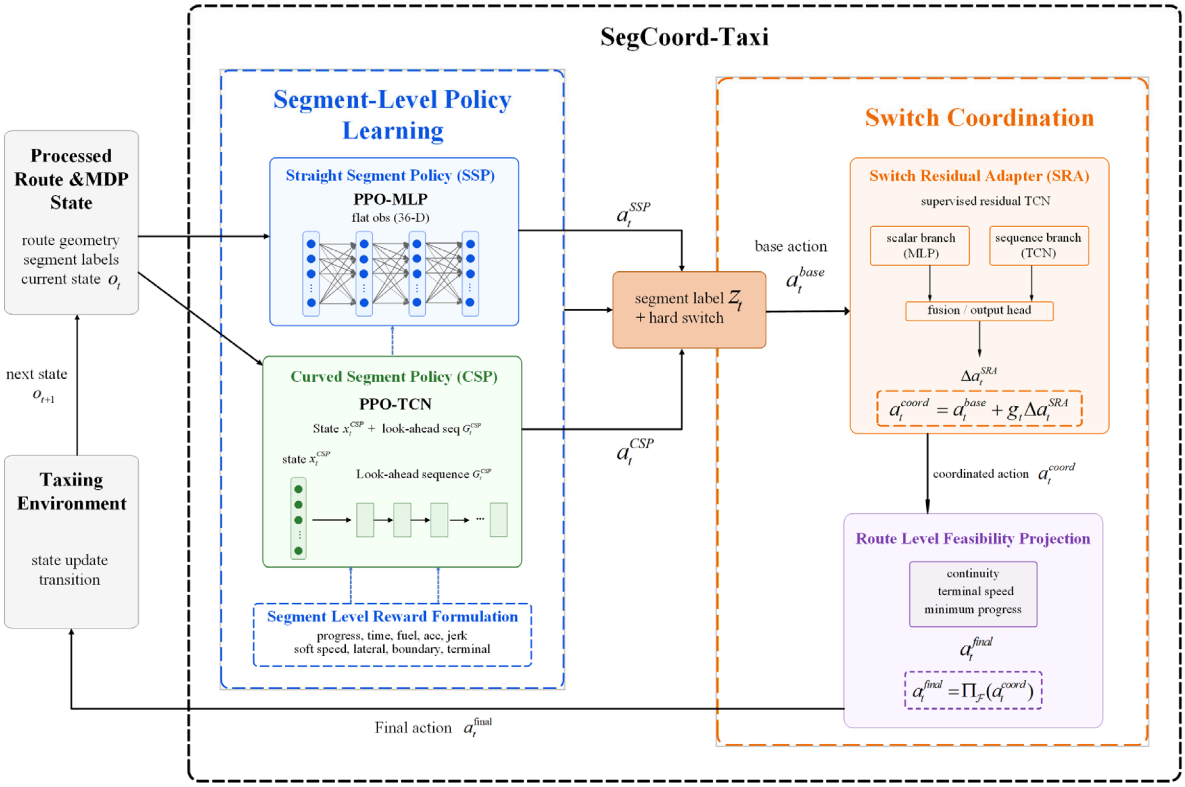


Figure 3. Overall architecture of the SegCoord-Taxi framework.

The framework uses route geometric structure to decompose the full-route speed-control process into base segmented policy learning, switch-region residual adaptation, and route-level feasibility handling. SSP and CSP generate geometry-dependent base actions, SRA is activated near straight–curve transition regions, and RFP is applied before environment execution. The module-level input–output relationship and full-route inference procedure are further summarized in Section 3.4.6.

3.4.1. Straight-Segment Policy

SSP is used for longitudinal speed control on low-curvature straight segments and adopts a PPO-MLP structure. On straight segments, route curvature is relatively small and lateral safety pressure is comparatively limited. The control objective is therefore to maintain stable propulsion, reduce unnecessary acceleration and deceleration fluctuations,

and provide a smooth base speed state for subsequent curved or switching regions. Its observation is obtained by applying a straight-segment encoding function to the full MDP state:

$$o_t^{SSP} = \phi_S(o_t), \;\; o_t^{SSP} \in \mathbb{R}^{36} \tag{11}$$

where o_t^{SSP} denotes the observation input of SSP, $\phi_S(\cdot)$ denotes the straight-segment encoding function, and $\mathbb{R}^{36}$ indicates that the observation is a 36-dimensional real-valued vector. This encoding organizes the motion state, local geometry, within-segment progress, remaining distance, boundary information, and sparse look-ahead features into a one-dimensional flat vector. Since geometric variation on straight segments is relatively mild, the model does not need to preserve the complete look-ahead curvature sequence. The MLP can directly extract the information required for speed control from the flattened features.

The policy mapping of SSP can be expressed as:

$$a_t^{SSP} = \pi_{\theta_S}^{SSP}(o_t^{SSP}) \tag{12}$$

where $\pi_\theta^{SSP}(\cdot)$ denotes the SSP policy network parameterized by θ and a_t^{SSP} is the base acceleration for straight segments. This action serves as the candidate control action in non-boundary straight segments. In straight–curve transition regions, it is further passed to the subsequent SRA and route-level feasibility modules for coordination and constraint enforcement. Therefore, SSP is functionally positioned to generate efficiency-oriented and smooth base speed control on straight segments, rather than independently completing terminal stopping over the entire route.

3.4.2. Curved-Segment Policy

CSP is used for longitudinal speed control on curved segments and adopts a PPO-TCN structure. In curved regions, speed and curvature jointly determine lateral acceleration. The current action depends not only on the current position but also on upcoming curvature variations, curve exits, and switching regions. Therefore, CSP preserves the sequential structure of look-ahead geometry, and its observation is represented as:

$$o_t^{CSP} = \phi_C(o_t) = \left\{ x_t^{CSP}, G_t^{CSP} \right\} \tag{13}$$

where

$$x_t^{CSP} \in \mathbb{R}^{18}, \;\; G_t^{CSP} \in \mathbb{R}^{25\times 15} \tag{14}$$

Here, x_t^{CSP} represents the current dynamics, local geometry, and boundary state, while G_t^{CSP} represents the look-ahead geometric sequence constructed along the arc-length direction. This sequence preserves future curvature variations, straight/curved segment distributions, and boundary-zone information, enabling the policy to distinguish different control situations such as curve entry, curve core, and curve exit. CSP uses an observation representation composed of the current state and the look-ahead sequence, and encodes the look-ahead geometric sequence through a TCN feature extractor.

The policy mapping of CSP is given by:

$$a_t^{CSP} = \pi_{\theta_C}^{CSP}(x_t^{CSP}, G_t^{CSP}) \tag{15}$$

where $\pi_\theta^{CSP}(\cdot)$ denotes the CSP policy network parameterized by θ and a_t^{CSP} denotes the base acceleration for curved segments. The TCN branch extracts local geometric patterns from the look-ahead sequence, while the state branch represents the current speed, acceleration, and within-segment position. The two types of features are fused to output the base acceleration for curved segments. CSP is functionally positioned to regulate the speed

profile along the route under given curved-segment geometric constraints, suppressing excessive speed, the lateral-acceleration proxy, and jerk in high-curvature regions without changing the aircraft's lateral taxiing route.

3.4.3. Segment-Level Reward Formulation

SSP and CSP are trained using segment-level reward functions. The purpose of these rewards is to shape local longitudinal motion behavior under different geometric conditions, rather than to directly optimize all evaluation metrics reported in Section 4. In particular, the segment-level rewards encourage forward progress, suppress excessive acceleration and jerk, penalize impractically low or excessive speed, and introduce curvature-related safety penalties in curved segments. Full-route terminal feasibility is not learned solely through a terminal reward. Instead, it is enforced during closed-loop execution by the Route-Level Feasibility Projection described in Section 3.4.5.

For a control step t, the segment-level reward can be written in a grouped form as:

$$\begin{aligned} R_t^{seg} = \; & w_p r_t^{progress} - w_T c_t^{time} - w_{low} c_t^{low} - w_a c_t^{acc} \\ & - w_j c_t^{jerk} - w_s c_t^{soft} - w_{lat} c_t^{lat} - w_b c_t^{boundary}. \end{aligned} \tag{16}$$

where $r_t^{progress}$ denotes the advancement reward along the route direction, c_t^{time} denotes the time cost, c_t^{low} denotes the low-speed penalty, c_t^{acc} and c_t^{jerk} denote the acceleration and jerk penalties, respectively, c_t^{soft} denotes the soft speed-boundary penalty, c_t^{lat} denotes the curvature-related lateral-acceleration penalty and $c_t^{boundary}$ denotes the boundary-zone smoothness penalty. The coefficients $w_p, w_T, w_{low}, w_a, w_j, w_s, w_{lat}$, and w_b are the corresponding weights.

The default coefficients in Table 4 were determined through multiple rounds of preliminary training and validation-based selection. In the preliminary stage, the reward terms were grouped according to their control functions, including route progress, low-speed avoidance, acceleration moderation, jerk suppression, speed-excess control, curvature-related lateral-acceleration regulation, and boundary-region smoothness. Several candidate coefficient settings were tested through short SSP and CSP training runs and validation rollouts. The coefficient magnitudes were adjusted to keep the major reward groups comparable in scale, so that the progress incentive remained effective while no single penalty term dominated the segment-level return. SSP and CSP were calibrated separately according to their geometric roles. SSP coefficients were selected to maintain stable propulsion and avoid low-speed stagnation on low-curvature segments, whereas CSP coefficients assigned stronger emphasis to lateral-acceleration and jerk-related penalties to suppress excessive speed in curved regions. A candidate setting was retained as the baseline configuration when the validation rollouts showed stable completion, no persistent low-speed stagnation, no excessive speed in high-curvature regions, and no abnormal acceleration or jerk peaks.

The lateral-acceleration penalty is applied only to CSP because curved segments require explicit curvature-aware speed regulation. In contrast, SSP focuses more on forward progress, stable propulsion, and avoidance of low-speed stagnation on low-curvature segments. Boundary-related terms are used to shape the behavior near straight–curve transition regions, where hard switching between SSP and CSP may otherwise lead to abrupt action changes.

The fuel proxy reported in Section 4 is computed from the resulting full-route motion profile and used as an external operational-cost metric. Terminal feasibility is handled during full-route execution by RFP, which maps the coordinated learning-based action into an executable acceleration satisfying action-rate, terminal-speed, and non-terminal

low-speed feasibility constraints. SSP and CSP therefore focus on local geometry-dependent speed-control behavior, while full-route feasibility is enforced during closed-loop rollout.

Table 4. Main segment-level reward groups and coefficient settings for SSP and CSP training.

Reward Group	SSP Coefficient	CSP Coefficient	Main Role
Progress reward	0.100 + 0.025 distance-progress proxy	0.12	Encourage forward movement within the current segment
Time/low-speed penalty	0.030/0.060	-/0.080	Discourage unnecessarily slow or stalled motion
Acceleration penalty	0.035	0.04	Suppress aggressive longitudinal acceleration
Jerk penalty	0.025	0.04	Improve acceleration smoothness
Soft speed-excess penalty	0.18	0.2	Penalize excessive speed
Lateral-acceleration penalty	0	1.8	Enforce curvature-aware behavior in curved segments
Boundary smoothness terms	speed: 0.015; jerk: 0.000	speed: 0.025; jerk: 0.050	Improve behavior near straight–curve transition regions

3.4.4. Switch Residual Adapter

SSP and CSP are trained for different geometric segments, and their action distributions and control preferences may therefore differ. If hard switching is performed only according to straight/curved segment labels, abrupt acceleration-command changes may occur near switching points. SRA is used to perform residual correction in these regions, improving switching continuity while preserving the base control capabilities of SSP and CSP.

SRA predicts local residuals on top of the SSP/CSP base action and mainly serves to correct action continuity in switching regions. Let the base action be:

$$a_t^{base} = \begin{cases} a_t^{SSP}, & z_t = \text{straight} \\ a_t^{CSP}, & z_t = \text{curve} \end{cases} \tag{17}$$

where a_t^{base} denotes the base action and z_t is the base segment label at the current position. When $z_t = \text{straight}$, the base action is a_t^{SSP} output by SSP; when $z_t = \text{curve}$, the base action is a_t^{CSP} output by CSP.

The input of SRA is jointly composed of the environment state, the base action, historical actions, and switching-region information:

$$o_t^{SRA} = \phi_R(o_t, a_t^{base}, a_{t-1}^{base}, a_{t-1}^{final}) \tag{18}$$

where o_t^{SRA} denotes the input observation of SRA, $\phi_R(\cdot)$ denotes the input encoding function of the residual adapter, a_{t-1}^{base} denotes the base action at the previous time step, and a_{t-1}^{final} denotes the final executed action at the previous time step. In terms of network structure, SRA adopts a supervised residual TCN. The scalar branch encodes the current control state and action context, while the sequence branch encodes look-ahead geometric information. The two types of features are fused to output the residual acceleration:

$$\Delta a_t^{SRA} = f_{\theta_R}^{SRA}(o_t^{SRA}) \tag{19}$$

where $f_\theta^{SRA}(\cdot)$ denotes the SRA network parameterized by θ and Δa_t^{SRA} denotes the residual acceleration output by SRA.

The residual acts on the base action through a switching-region gating weight:

$$a_t^{coord} = a_t^{base} + g_t \Delta a_t^{SRA} \tag{20}$$

where a_t^{coord} denotes the coordinated action after residual correction and g_t is determined by the boundary weight and the distance to the nearest switching point. This gating mechanism concentrates residual correction near straight–curve transitions and reduces interference with base-policy outputs in non-boundary regions.

The supervised labels for SRA are constructed from SSP + CSP hard-switching trajectories. Given a base action sequence, a smoothed action target a_t^{smooth} is generated through action-rate constraints, and the residual label is then obtained as:

$$\Delta a_t^{target} = a_t^{smooth} - a_t^{base} \tag{21}$$

where Δa_t^{target} denotes the target residual used for supervised learning. The training loss is:

$$\mathcal{L}_{SRA} = \frac{1}{N}\sum_{t=1}^{N}\left(\Delta a_t^{SRA} - \Delta a_t^{target}\right)^2 \tag{22}$$

where $\mathcal{L}_{SRA}$ denotes the training loss of SRA and N denotes the number of time steps in the training sequence or samples. SRA training uses the above supervised objective, meaning that it learns the difference between the smoothed action target and the hard-switching base action from full-route trajectories generated by SSP + CSP hard switching. Thus, the role of SRA in the framework is learning-based residual adaptation in switching regions, rather than serving as a full-route control policy.

3.4.5. Route-Level Feasibility Projection

Route-level feasibility is used to convert the coordinated action corrected by SRA into a final control action that satisfies full-route constraints. Let a_t^{coord} denote the coordinated action after SRA correction. The final executed action is:

$$a_t^{final} = \Pi_{\mathcal{F}}(a_t^{coord}) \tag{23}$$

where a_t^{final} denotes the final executed action, Π_Φ denotes the Route-Level Feasibility Projection, and Φ denotes the feasible set jointly defined by action continuity, terminal velocity, and non-terminal propulsion constraints. This module contains no trainable parameters. Its role is to impose deterministic constraints on the output of the learning-based policy, so that the final speed profile is executable at the full-route level.

The action-continuity constraint limits the magnitude of acceleration variation between adjacent control intervals:

$$\left|a_t^{final} - a_{t-1}^{final}\right| \le \Delta a_{\max} \tag{24}$$

where a_{t-1}^{final} denotes the final executed action at the previous time step, and $\Delta a_{\max}$ denotes the maximum allowable change between adjacent actions. In straight–curve switching regions, this constraint can adopt a stricter rate-of-change upper bound to suppress acceleration jumps and jerk peaks caused by policy switching. This treatment complements the residual correction performed by SRA: SRA alleviates base-action differences at switching points in a learning-based manner, while the action-continuity constraint limits final action variation at the route-level execution stage.

The terminal-velocity constraint is used to ensure that the aircraft can gradually decelerate to an acceptable range when approaching the end of the route. Let d_t^{end} denote the remaining distance from the current position to the route end. The reference speed in the terminal region can be expressed as

$$v_t \le v^{ref}(d_t^{end}) \tag{25}$$

where v_t denotes the current speed and $v^{ref}(d_t^{end})$ denotes the terminal reference speed envelope determined by the remaining distance d_t^{end}, decreasing as the remaining distance decreases. When the current speed exceeds the reference speed envelope, the projection layer limits positive acceleration or applies braking actions so that the aircraft tends toward a smooth stopping state within the limited remaining distance. This constraint is used to avoid infeasible terminal states in whichthe route is completed but the terminal speed is too high.

The non-terminal low-speed constraint is used to address the risk of persistent low-speed operation or stalling far from the route end. When the terminal-velocity constraint is inactive and the aircraft remains at low speed for an extended period in a non-terminal region, the projection layer applies a minimum propulsion action to maintain effective movement along the route direction. This constraint does not aim to increase cruising speed; rather, it prevents the policy from falling into an unrecoverable low-speed state far from the terminal region.

Through route-level feasibility, the final action of SegCoord-Taxi simultaneously inherits the segmented control capability of SSP/CSP, the switch residual correction capability of SRA, and route-level feasibility constraints. Its output serves as the longitudinal acceleration actually executed by the environment:

$$a_t^{final} = \Pi_{\mathcal{F}}\left(a_t^{base} + g_t \Delta a_t^{SRA}\right) \tag{26}$$

This formulation also summarizes the overall control chain of SegCoord-Taxi: SSP and CSP generate base segmented actions; SRA performs learning-based residual correction in switching regions; the route-level feasibility module constrains action continuity, terminal velocity, and non-terminal propulsion state; and the final output is an executable longitudinal acceleration control sequence.

In implementation, RFP constructs the final acceleration by applying the active action bounds imposed by action-rate, terminal-speed, and non-terminal propulsion requirements. When the feasible acceleration interval is non-empty, the coordinated action is clipped into this interval, so the executed action satisfies the active route-level constraints by construction. If conflicting constraints occur, terminal-speed feasibility and safety-related braking are given priority. This mechanism provides an execution-level feasibility safeguard rather than an additional trainable policy.

3.4.6. Overall Training and Inference Procedure

The complete SegCoord-Taxi procedure consists of an offline training stage and an online full-route inference stage. In the offline stage, the processed route dataset is first divided into straight-segment samples, curved-segment sequence samples, and boundary-region samples. SSP and CSP are trained separately on straight and curved base segments, respectively. After the two base policies are obtained, hard-switching full-route rollouts are generated to construct supervised residual labels for SRA. Route-Level Feasibility Projection is not trained; it is a deterministic execution module applied during full-route rollout.

Table 5 summarizes the input, output, and functional role of each module in the complete framework.

The full-route inference procedure is summarized in Algorithm 1. At each control step, the environment queries the current geometric state and segment label according to the aircraft arc-length position. SSP or CSP first generates a base acceleration according to the current segment type. SRA then provides residual correction through the boundary gate. Finally, RFP maps the coordinated action into the final executable acceleration, which is used by the environment to update speed and position.

Table 5. Input–output summary of SegCoord-Taxi modules.

Module	Input	Output	Training Mode	Functional Role
SSP	Flat straight-segment observation o_t^{SSP}	Straight-segment base acceleration a_t^{SSP}	PPO training on straight-segment samples	Generate base acceleration for low-curvature straight segments
CSP	Current curved-segment state x_t^{CSP}, and look-ahead geometry sequence G_t^{CSP}	Curved-segment base acceleration a_t^{CSP}	PPO training on curved-segment samples	Generate curvature-aware base acceleration for curved segments
SRA	Current state, base action, previous actions, and boundary-region features	Residual acceleration Δa_t^{SRA}	Supervised residual learning	Reduce local action discontinuities near straight–curve transitions
RFP	Coordinated action a_t^{coord}, previous final action, speed, remaining distance, feasibility parameters	Final executed acceleration a_t^{final}	Deterministic projection	Enforce action-rate, non-terminal propulsion, and terminal-speed feasibility
Environment	Final acceleration a_t^{final} and current motion state	Updated speed, arc-length position, and next observation	Kinematic state update	Simulate full-route longitudinal taxiing motion

Algorithm 1. Full-route inference procedure of SegCoord-Taxi.

Input: Processed taxiing route Π; trained SSP π_θ^{SSP}; trained CSP π_θ^{CSP}; trained SRA f_θ^{SRA}; feasibility parameter set Φ; control interval Δt.

Output: Full-route trajectory $\mathcal{T} = \{s_t, v_t, a_t^{final}\}_{t=0}^{T}$.

1. Initialize the aircraft state: $s_0 = 0$, $v_0 = 0$, $a_{-1}^{final} = 0$, and $t = 0$
2. While the route is not completed and no termination condition is triggered:
3. Query the processed route table at the current arc-length position s_t, including local curvature, heading angle, segment label z_t, boundary weight, remaining distance, and look-ahead geometry.
4. If z_t = straight, construct the SSP observation o_t^{SSP} and compute the straight-segment base action: $a_t^{base} = a_t^{SSP} = \pi_\theta^{SSP}(o_t^{SSP})$.
5. If z_t = curve, construct the CSP current state x_t^{CSP} and look-ahead geometry sequence G_t^{CSP}, and compute the curved-segment base action: $a_t^{base} = a_t^{CSP} = \pi_\theta^{CSP}(x_t^{CSP}, G_t^{CSP})$.
6. Construct the SRA input o_t^{SRA} from the current state, base action, previous actions, and boundary-region features.
7. Predict the residual acceleration: $\Delta a_t^{SRA} = f_\theta^{SRA}(o_t^{SRA})$.
8. Apply the boundary gate g_t to obtain the coordinated action: $a_t^{coord} = a_t^{base} + g_t \Delta a_t^{SRA}$.
9. Apply Route-Level Feasibility Projection: $a_t^{final} = \Pi_\Phi(a_t^{coord})$.
10. Update the aircraft speed: $v_{t+1} = \max(0, v_t + a_t^{final} \Delta t)$.
11. Update the arc-length position: $s_{t+1} = s_t + \frac{1}{2}(v_t + v_{t+1})\Delta t$.
12. Store (s_t, v_t, a_t^{final}) in $\mathcal{T}$, set $t = t + 1$, and continue the rollout.
13. Return the full-route trajectory $\mathcal{T}$.

Algorithm 1 clarifies the sequential control chain of SegCoord-Taxi. SSP and CSP provide geometry-dependent base actions, SRA modifies the base action locally through the boundary gate, and RFP generates the final executable acceleration under route-level feasibility constraints. The final control command is therefore obtained through base-policy selection, residual adaptation, and deterministic feasibility projection within one closed-loop rollout.

3.5. Evaluation Metrics

Model performance is evaluated from five perspectives: task completion, terminal feasibility, action smoothness, curve safety, and operational proxy cost. Task completion measures whether the aircraft can reach the end of the predefined route. Let N denote the number of evaluation episodes. The completion rate is defined as:

$$CR = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_i^{completed} \tag{27}$$

where $\mathbb{I}_i^{completed}$ indicates whether the i-th episode completes the route.

Terminal feasibility is measured by the smooth-stop rate and final velocity. The smooth-stop rate is defined as:

$$SSR = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_i^{smooth} \tag{28}$$

where $\mathbb{I}_i^{smooth}$ indicates whether the i-th episode satisfies the terminal velocity threshold upon reaching the route end. Final velocity reflects the residual speed level at route completion and can distinguish between smooth completion and high-speed completion.

Action smoothness is evaluated using mean absolute jerk, boundary action variation, and switch action jump. Mean absolute jerk measures the global smoothness of acceleration changes along the complete route. Jerk is defined as the rate of change in acceleration between adjacent control intervals:

$$j_t = \frac{a_t - a_{t-1}}{\Delta t} \tag{29}$$

where a_t and a_{t-1} denote the accelerations at the current and previous time steps, respectively, and Δt denotes the control interval. Boundary action variation is used in the baseline comparison because some baseline methods do not contain explicit SSP/CSP policy-switching points. It is calculated as the mean absolute acceleration difference within straight–curve boundary windows:

$$BAV = \frac{1}{|\mathcal{T}_b|}\sum_{t\in\mathcal{T}_b} |a_t - a_{t-1}| \tag{30}$$

where $\mathcal{T}_b$ denotes the set of time steps located in straight–curve boundary windows. Switch action jump is used in the ablation study to evaluate action discontinuity at SSP/CSP switching positions:

$$J_{switch} = \frac{1}{|\mathcal{T}_{sw}|}\sum_{t\in\mathcal{T}_{sw}} |a_t - a_{t-1}| \tag{31}$$

where $\mathcal{T}_{sw}$ denotes the set of time steps at which switching occurs between straight and curved segments.

Curve safety is measured using the lateral-acceleration proxy. Let v_t denote the current speed and κ_t denote the route curvature. The lateral-acceleration proxy is given by:

$$a_t^{lat} = v_t^2|\kappa_t| \tag{32}$$

This metric reflects the safety pressure jointly induced by speed and curvature in curved regions. The mean and maximum values of this proxy are recorded during evaluation to compare curve-related speed-control behavior among different strategies.

Operational proxy cost is evaluated using taxiing time and a fuel proxy metric. Taxiing time reflects route-completion efficiency. The fuel proxy is computed from the resulting full-route motion profile and is used as a relative operational-cost indicator for comparing different control strategies. It is not interpreted as a high-fidelity engine fuel-consumption or emission estimate.

Although multiple metrics are recorded during evaluation, the main result tables focus on taxiing time, fuel proxy, mean absolute jerk, boundary or switching action variation, and final velocity. These metrics directly correspond to the trade-off among efficiency, operational proxy cost, action smoothness, local transition behavior, and terminal velocity feasibility emphasized in this study.

4. Results

4.1. Experimental Settings

This section presents the computational environment, model parameter configuration, and experimental design. All experiments are based on the ZUTF departure taxiing route dataset described in Section 3.1 and use the same route-level train/validation/test split. All methods are evaluated in a full-route closed-loop environment, where the input consists of a given taxiing route and its geometric features, and the control output is a one-dimensional continuous acceleration along the route direction. The lateral path of the aircraft remains the predefined taxiing route, while the longitudinal speed and arc-length position are updated according to the kinematic equations.

4.1.1. Computational Environment and Software Configuration

All experiments are conducted on the same computing platform. The hardware platform uses an AMD Ryzen 7 9700X 8-Core Processor (Advanced Micro Devices, Inc., Santa Clara, CA, USA), equipped with 32 GB RAM and an NVIDIA GeForce RTX 5070 GPU (NVIDIA Corporation, Santa Clara, CA, USA). The operating system is Microsoft Windows 11 Professional (Microsoft Corporation, Redmond, WA, USA). Model training and evaluation are implemented in Python 3.9.25. The deep learning framework is PyTorch 2.7.1, and the reinforcement learning algorithms are implemented based on stable-baselines3 2.7.1 and Gymnasium 1.1.1. GPU acceleration is supported by the PyTorch CUDA runtime 12.8.

4.1.2. Hyperparameter Configuration and Experimental Design

The full-route environment uses a fixed-time control interval, and the control variable is a one-dimensional continuous acceleration along the taxiing route. The control interval is 1.0 s, and the acceleration range is $[-0.98, 0.98]$ m/s^2. Route geometry is resampled at a 5 m arc-length interval for curvature query, segment label identification, and look-ahead window construction. The actual displacement during control is obtained by integrating the current speed and acceleration. The environment does not use target speed curves, rule-based node turning speed limits, or speed-tracking rewards; curvature is used only as a source of geometric state information and a soft lateral-acceleration constraint.

SSP adopts a PPO-MLP structure, with a 36-dimensional flat observation vector as input and the straight-segment base acceleration as output. CSP adopts a PPO-TCN structure, with an input composed of an 18-dimensional current state vector and a 25×15 look-ahead geometry sequence, and outputs the curved-segment base acceleration. The CSP training environment is designed for curved base segments, with a control interval of 1.0 s, an acceleration range of $[-0.98, 0.98]$ m/s^2, a look-ahead sequence length of 25, and a look-ahead step of 5 m. Its PPO training parameters include a learning rate of 2.5×10^{-4}, n_steps = 1024, batch size 256, n_epochs = 10, $\gamma = 0.995$, GAE $\lambda = 0.95$, and clip range 0.20.

Unless otherwise stated, the reported closed-loop results are computed on the route-level test split using deterministic policy outputs. For learning-based configurations, three random seeds are used, and the results are summarized as mean ± standard deviation. Deterministic rule-based and optimization-based baselines are evaluated on the same ten test routes under the same control interval, acceleration bounds, terminal-speed criterion, and evaluation metrics. The training and validation splits are used for policy learning, adapter training, hyperparameter selection, and early stopping.

The Route-Level Feasibility Projection consists of an action-rate constraint, a terminal speed-envelope constraint, and a non-terminal low-speed propulsion constraint. The default upper bound of the action-rate constraint is 0.55 m/s^2, and it is tightened to 0.45 m/s^2 in policy-switching regions. The terminal speed envelope is activated within $D_{term} = 950$ m from the route end and uses a terminal target speed of $v_{target} = 0.30$ m/s. The smooth-stop threshold $v_{stop} = 1.00$ m/s is used only to evaluate whether terminal completion satisfies the smooth-stop criterion. The non-terminal low-speed constraint provides a minimum propulsion action when the aircraft remains at low speed far from the route end.

To evaluate the proposed method, five groups of experiments are conducted. First, the baseline comparison evaluates SegCoord-Taxi against Rule-Based Speed Profile, Heuristic Speed-Profile Optimization, Curvature-Aware Constrained Optimizer, and Single PPO Controller. The Curvature-Aware Constrained Optimizer is included as a non-learning geometry-aware baseline, while Single PPO Controller represents a unified learning-based policy. Second, the ablation study compares SSP + CSP, SSP + CSP + SRA, SSP + CSP + RFP, and the complete SegCoord-Taxi framework to identify the respective roles of SRA and RFP. Third, projection diagnostics are used to isolate the contribution of RFP, including the Single PPO + RFP diagnostic comparison and step-level projection activation statistics. Fourth, sensitivity analyses examine terminal-related parameters, boundary-zone settings, and grouped reward-coefficient perturbations. Fifth, a trajectory-level case study visualizes the speed, action, residual, and projection behavior on a representative route. These experiments jointly evaluate overall performance, module contribution, projection behavior, parameter sensitivity, and local control behavior.

4.2. Baseline Comparison

To evaluate the full-route speed-control performance of SegCoord-Taxi, this study compares the proposed framework with four baseline methods: Rule-Based Speed Profile, Heuristic Speed-Profile Optimization, Curvature-Aware Constrained Optimizer, and Single PPO Controller. Rule-Based Speed Profile represents simplified constant-speed or turning-related speed settings. Heuristic Speed-Profile Optimization represents traditional speed-profile search or segmented speed optimization methods that balance taxiing time, operational proxy cost, timing constraints, and speed smoothness [9,11]. The Curvature-Aware Constrained Optimizer is included as a stronger non-learning geometry-aware baseline that explicitly considers route curvature, forward-looking speed envelopes, acceleration constraints, and terminal-distance constraints during speed-profile generation. Single PPO Controller represents a unified reinforcement learning policy for the whole route, following previous learning-based autonomous taxiing studies [31]. SegCoord-Taxi is the complete framework proposed in this study, consisting of SSP, CSP, SRA, and RFP.

All methods are evaluated in the same full-route closed-loop environment on the test set. Since some baselines do not involve explicit SSP/CSP policy-switching points, boundary action variation is used in this comparison to measure the mean absolute difference between consecutively applied accelerations within straight–curve boundary windows. Table 6 presents the baseline comparison results.

Table 6. Baseline comparison on the test set.

Method	Taxiing Time (s) ↓	Fuel Proxy ↓	Mean |Jerk| (m/s^3) ↓	Boundary Action Variation (m/s^2) ↓	Final Velocity (m/s) ↓
Rule-Based Speed Profile	602.6	91.266	0.062	0.072	1.319
Heuristic Speed-Profile Optimization	540.25	95.685	0.089	0.101	2.721
Curvature-Aware Constrained Optimizer	565.7	93.648	0.072	0.149	0.586
Single PPO Controller	474.6 ± 7.6	114.864 ± 1.792	0.129 ± 0.009	0.178 ± 0.004	0.732 ± 0.061
SegCoord-Taxi (Proposed)	506.0 ± 4.9	97.831 ± 1.115	0.092 ± 0.005	0.128 ± 0.007	0.336 ± 0.017

Table 6 shows a clear trade-off among efficiency, fuel proxy, smoothness, boundary-region action variation, and terminal feasibility. Rule-Based Speed Profile produces the smoothest acceleration sequence but has the longest taxiing time and a relatively high final velocity. Heuristic Speed-Profile Optimization improves taxiing efficiency compared with the rule-based method, but its final velocity remains high, indicating insufficient terminal deceleration. The Curvature-Aware Constrained Optimizer achieves a low fuel proxy and a relatively low mean |jerk|, showing that a geometry-aware non-learning controller can generate smooth and conservative speed profiles. However, its taxiing time is longer than that of SegCoord-Taxi, and its boundary action variation and final velocity are also higher.

Single PPO Controller achieves the shortest taxiing time, but it has the highest fuel proxy, mean |jerk|, and boundary action variation, suggesting that a unified policy tends to favor aggressive propulsion at the cost of smoothness and local geometric coordination. SegCoord-Taxi achieves a taxiing time of 506.0 s and the lowest final velocity of 0.336 m/s. Compared with the Curvature-Aware Constrained Optimizer, it reduces taxiing time by approximately 10.6%, boundary action variation by approximately 14.1%, and final velocity by approximately 42.7%, although its fuel proxy and mean |jerk| are higher. These results indicate that, on the evaluated ZUTF test routes, SegCoord-Taxi does not dominate all metrics, but provides a more balanced trade-off between operational efficiency, boundary-region coordination, and terminal velocity feasibility within the present predefined-route single-aircraft setting.

To further illustrate the speed evolution of different methods on the same route, a representative test route is selected for visualization in Figure 4. The Rule-Based method maintains a relatively low and stable speed profile, reflecting its conservative control behavior. The Heuristic Speed-Profile Optimization method produces smoother speed variations than Single PPO, but its terminal deceleration remains insufficient. The Curvature-Aware Constrained Optimizer follows a more conservative geometry-aware speed profile and reduces speed before curved or constrained regions, but its overall speed level is lower than that of SegCoord-Taxi on several long straight segments. Single PPO maintains high speeds over long straight sections, but it exhibits more abrupt local speed changes and weaker terminal-speed control. SegCoord-Taxi maintains relatively high speeds on straight segments while actively decelerating near complex geometric regions and the terminal region. The comparison suggests that SegCoord-Taxi provides a trade-off among operational efficiency, boundary-region coordination, and terminal velocity feasibility. For this representative ZUTF test route, the observed behavior illustrates how geometry-dependent policy specialization, switch-region adaptation, and route-level feasibility handling jointly shape the final speed profile.

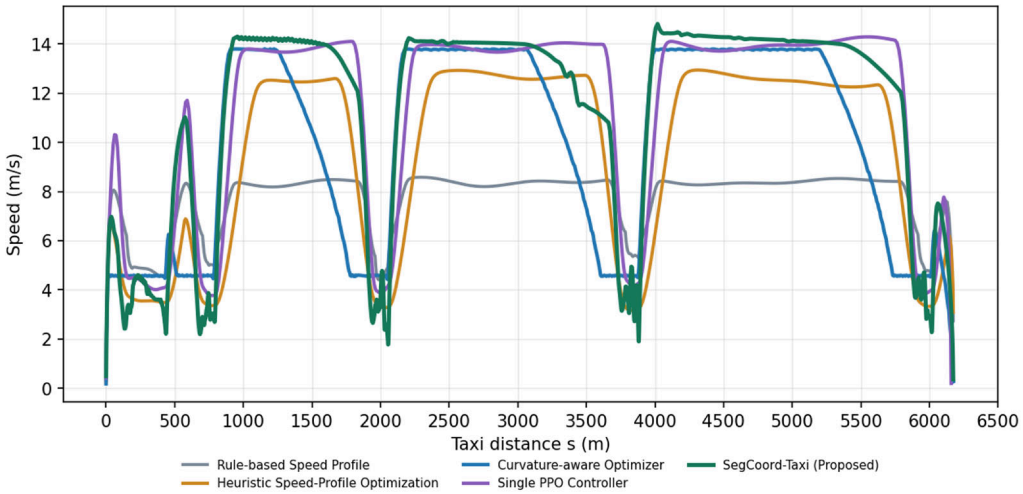


Figure 4. Speed-distance profiles of different baseline methods and SegCoord-Taxi on a representative test route.

4.3. Ablation Study

To analyze the roles of the components in SegCoord-Taxi, this study uses SSP + CSP as the base combination and separately adds SRA and RFP for ablation comparison. Table 7 presents the results of four methods on the test set. SSP + CSP denotes the hard-switching combination of the Straight-Segment Policy and the Curved-Segment Policy. SSP + CSP + SRA denotes the addition of the Switch Residual Adapter to the hard-switching baseline. SSP + CSP + RFP denotes the addition of Route-Level Feasibility Projection. SSP + CSP + SRA + RFP denotes the complete SegCoord-Taxi framework.

Table 7. Ablation study results on the test set across three random seeds.

Method	Taxiing Time (s) ↓	Fuel Proxy ↓	Mean |Jerk| (m/s^3) ↓	Switch Action Jump (m/s^2) ↓	Final Velocity (m/s) ↓
SSP + CSP	492.6 ± 7.7	99.459 ± 1.660	0.164 ± 0.015	1.022 ± 0.017	11.311 ± 0.049
SSP + CSP + SRA	492.2 ± 6.5	98.377 ± 1.388	0.093 ± 0.007	0.433 ± 0.034	11.283 ± 0.034
SSP + CSP + RFP	505.4 ± 6.7	97.704 ± 1.481	0.126 ± 0.005	0.717 ± 0.004	0.353 ± 0.023
SSP + CSP + SRA + RFP	506.0 ± 4.9	97.831 ± 1.115	0.092 ± 0.005	0.429 ± 0.004	0.336 ± 0.017

As shown in Table 7, the ablation results are consistent across the three random seeds and demonstrate different functional roles of SRA and Route-Level Feasibility Projection. Compared with SSP + CSP, adding SRA reduces mean |jerk| from 0.164 ± 0.015 m/s^3 to 0.093 ± 0.007 m/s^3 and reduces switch action jump from 1.022 ± 0.017 m/s^2 to 0.433 ± 0.034 m/s^2. This indicates that SRA effectively alleviates action discontinuities caused by hard switching between the Straight-Segment Policy and the Curved-Segment Policy. However, the final velocity remains high, at 11.283 ± 0.034 m/s, indicating that SRA mainly acts on switching regions and cannot independently solve terminal velocity infeasibility.

After adding Route-Level Feasibility Projection, final velocity decreases substantially from 11.311 ± 0.049 m/s to 0.353 ± 0.023 m/s, indicating that Route-Level Feasibility Projection is the key module for improving terminal feasibility. Meanwhile, switch action jump decreases from 1.022 ± 0.017 m/s^2 to 0.717 ± 0.004 m/s^2, suggesting that projection also improves final action continuity to some extent through action-rate constraints. However, its mean |jerk| is 0.126 ± 0.005 m/s^3, higher than that of the SRA-only configuration, indicating that projection is more oriented toward execution-level feasibility constraints rather than specialized residual correction in switching regions.

The complete SegCoord-Taxi framework, which includes both SRA and Route-Level Feasibility Projection, achieves the lowest mean |jerk|, switch action jump, and final velocity among the four ablation configurations, at 0.092 ± 0.005 m/s^3, 0.429 ± 0.004 m/s^2, and 0.336 ± 0.017 m/s, respectively. Compared with the projection-only configuration, the complete model further improves action smoothness and terminal velocity, while the taxiing time increases to 506.0 ± 4.9 s. These results show that SRA and RFP contribute to different parts of the full-route control process, with SRA affecting switch-region continuity and RFP affecting execution-level terminal feasibility.

4.4. Projection Diagnostics and Contribution Isolation

To further isolate the role of Route-Level Feasibility Projection (RFP), an additional diagnostic comparison was conducted. Let a_t^{coord} denote the coordinated acceleration after SSP/CSP selection and SRA correction, and let a_t^{final} denote the final acceleration after RFP. The projection correction is defined as:

$$\Delta a_t^{RFP} = a_t^{final} - a_t^{coord} \tag{33}$$

The same RFP module was additionally applied to the Single PPO controller, resulting in the variant denoted Single PPO + RFP. This diagnostic variant helps distinguish the contribution of RFP from the effects of segmented policy learning and residual switch adaptation. The diagnostic comparison results are summarized in Table 8.

Table 8. Diagnostic comparison for isolating the effect of Route-Level Feasibility Projection.

Method	Taxiing Time (s) ↓	Fuel Proxy ↓	Mean |Jerk| (m/s^3) ↓	Boundary Action Variation (m/s^2) ↓	Final Velocity (m/s) ↓
Single PPO Controller	474.6 ± 7.6	114.864 ± 1.792	0.129 ± 0.009	0.178 ± 0.004	0.732 ± 0.061
Single PPO + RFP	487.2 ± 5.2	119.628 ± 1.247	0.114 ± 0.012	0.153 ± 0.014	0.571 ± 0.037
SegCoord-Taxi (Proposed)	506.0 ± 4.9	97.831 ± 1.115	0.092 ± 0.005	0.128 ± 0.007	0.336 ± 0.017

Adding RFP to Single PPO reduces the final velocity from 0.732 ± 0.061 m/s to 0.571 ± 0.037 m/s. It also decreases mean $|\ jerk\ |$ and boundary action variation, indicating that the projection layer improves terminal-speed regulation and action-rate consistency. However, Single PPO + RFP still has higher fuel proxy, mean $|\ jerk\ |$, boundary action variation, and final velocity than SegCoord-Taxi. This result indicates that RFP contributes to route-level feasibility, while the segmented policies and SRA remain necessary for geometry-dependent base control and transition-region smoothness.

Projection-level diagnostics were further computed over the test routes to quantify how frequently and how strongly RFP modifies the coordinated actions. The results are summarized in Table 9.

Table 9. Route-Level Feasibility Projection diagnostics on the test set.

Diagnostic Item	Value
Overall RFP activation rate (%)	28.8
Action-rate correction activation rate (%)	16.4
Terminal correction activation rate (%)	12.8
Non-terminal low-speed correction activation rate (%)	0
Mean $\left\|\Delta a_t^{RFP}\right\|$ (m/s^2)	0.104
95th percentile $\left\|\Delta a_t^{RFP}\right\|$ (m/s^2)	0.766
Terminal-region activation rate (%)	60.9

RFP modifies the coordinated action in 28.8% of all time steps, with a terminal-region activation rate of 60.9%. This indicates that its intervention is concentrated in feasibility-critical regions rather than uniformly applied throughout the route. The non-terminal low-speed correction is not triggered in the present test set, suggesting that persistent low-speed stalling does not occur under the evaluated rollouts. Together with the ablation results, these diagnostics indicate that RFP primarily affects terminal-speed regulation and action-rate feasibility, while switch-region continuity is more directly reflected in the SRA-related results.

4.5. Sensitivity Analysis

Three sensitivity analyses are conducted to evaluate the influence of key manually specified settings. The first analysis varies terminal-related RFP parameters. The second analysis varies boundary-zone settings used for switch-region modulation. The third analysis applies grouped reward-coefficient perturbations to SSP and CSP training. All sensitivity experiments are evaluated on the same ten test routes, and the grouped reward-coefficient sensitivity is conducted over three random seeds.

4.5.1. Sensitivity to Terminal Feasibility Parameters

A one-factor-at-a-time sensitivity analysis was performed for the terminal activation distance D_{term}, the terminal target speed v_{target}, and the smooth-stop threshold v_{stop}. When one parameter was varied, the other parameters were kept at their default values. The results are summarized in Table 10.

Table 10. Sensitivity analysis of terminal feasibility parameters.

Parameter	Value	Time (s) ↓	Fuel Proxy ↓	Mean |Jerk| (m/s^3) ↓	Boundary Variation (m/s^2) ↓	Final Velocity (m/s) ↓
D_{term}	700 m	503.3	97.413	0.097	0.124	0.347
D_{term}	950 m	506.03	97.831	0.096	0.123	0.336
D_{term}	1200 m	507.37	98.102	0.096	0.123	0.33
v_{target}	0.20 m/s	506.27	97.857	0.096	0.123	0.283
v_{target}	0.30 m/s	506.03	97.831	0.096	0.123	0.336
v_{target}	0.50 m/s	505.67	97.79	0.096	0.124	0.458
v_{stop}	0.50 m/s	506.03	97.831	0.096	0.123	0.336
v_{stop}	1.00 m/s	506.03	97.831	0.096	0.123	0.336
v_{stop}	1.50 m/s	506.03	97.831	0.096	0.123	0.336

The terminal-parameter results show an interpretable efficiency–feasibility trade-off. Increasing D_{term} from 700 m to 1200 m reduces the final velocity from 0.347 m/s to 0.330 m/s, while increasing the taxiing time from 503.30 s to 507.37 s. Increasing v_{target} from 0.20 m/s to 0.50 m/s raises the final velocity from 0.283 m/s to 0.458 m/s, while all tested settings remain below the smooth-stop threshold. Varying v_{stop} does not change the rollout metrics because it is used as an evaluation threshold for smooth-stop success, not as an acceleration-generation parameter. The completion rate and smooth-stop rate remain 100% for all tested settings.

4.5.2. Sensitivity to Boundary-Zone Parameters

The boundary-zone mechanism affects the modulation of SRA residual corrections near straight–curve transitions. To test whether the results depend on a specific boundary setting, the boundary half-width D_b and boundary weighting function were varied. This analysis was conducted as an inference-level sensitivity test with fixed trained SSP, CSP, and SRA models. The results are summarized in Table 11.

Table 11. Sensitivity analysis of boundary-zone parameters.

Parameter	Value	Time (s) ↓	Fuel Proxy ↓	Mean |Jerk| (m/s^3) ↓	Boundary Variation (m/s^2) ↓	Switch Action Jump (m/s^2) ↓	Final Velocity (m/s) ↓
D_b	20 m	505.97	97.84	0.095	0.12	0.43	0.334
D_b	40 m	505.17	97.758	0.096	0.123	0.426	0.334
D_b	80 m	505.77	97.771	0.095	0.122	0.425	0.335
Weight function	Linear	505.17	97.758	0.096	0.123	0.426	0.334
Weight function	Gaussian	505.97	97.803	0.095	0.121	0.43	0.343
Weight function	Exponential	506.33	97.853	0.095	0.121	0.43	0.337

The boundary-zone results vary only within a narrow range across the tested settings. For $D_b = 20$, 40, and 80 m, the final velocity remains around 0.334 m/s, and both boundary variation and switch action jump change only slightly. Similar patterns are observed for the linear, Gaussian, and exponential weighting functions. These results support the use of $D_b = 40$ m with linear weighting as a simple and interpretable default configuration. Since SSP, CSP, and SRA are fixed in this test, the results characterize inference-level boundary sensitivity under trained policies.

4.5.3. Grouped Reward-Coefficient Sensitivity

Grouped reward perturbations are applied to the SSP/CSP training rewards. The tested settings include the default configuration, an efficiency-emphasized configuration, a smoothness-emphasized configuration, a curve-safety-emphasized configuration, and a weak-smoothness configuration. Fuel proxy and full-route terminal stopping are excluded from the perturbation settings because they are evaluated or enforced during full-route rollout, whereas SSP and CSP are trained using segment-level reward terms. The grouped reward-coefficient sensitivity results are summarized in Table 12.

Table 12. Grouped reward-coefficient sensitivity analysis.

Reward Setting	Time (s) ↓	Fuel Proxy ↓	Mean |Jerk| (m/s^3) ↓	Boundary Variation (m/s^2) ↓	Switch Action Jump (m/s^2) ↓	Final Velocity (m/s) ↓
Default	511.2	99.117	0.1	0.127	0.431	0.353
Efficiency-emphasized	496.3	97.098	0.106	0.137	0.438	0.371
Smoothness-emphasized	529.2	102.287	0.101	0.128	0.43	0.34
Curve-safety-emphasized	524.5	101.379	0.096	0.124	0.433	0.345
Weak-smoothness	490.6	96.763	0.125	0.138	0.421	0.326

The reward sensitivity results show that reward perturbations mainly shift the trade-off among efficiency, smoothness, and boundary behavior. The efficiency-emphasized and weak-smoothness settings reduce taxiing time and fuel proxy, but they increase mean jerk and boundary variation, indicating more aggressive control behavior. The curve-safety-emphasized setting produces a more conservative profile, with longer taxiing time but lower mean jerk and boundary variation. The smoothness-emphasized setting also leads to more conservative behavior, although the improvement in smoothness-related metrics is not strictly monotonic. Across all reward settings, the completion rate and smooth-stop rate remain 100%, indicating that full-route terminal feasibility is maintained by RFP rather than by a single reward-coefficient configuration. Overall, the sensitivity results show that reasonable grouped coefficient perturbations mainly shift the efficiency–smoothness trade-off. Therefore, this analysis serves as a robustness check around the selected baseline coefficients rather than as a search for globally optimal reward weights.

4.6. Trajectory-Level Case Study

To illustrate the closed-loop behavior of SegCoord-Taxi at the trajectory level, a representative departure taxiing route from the test set is selected. The route contains multiple straight–curve transition regions and a complete terminal deceleration process. Figure 5 shows the speed profile, SRA residual, RFP correction, and terminal-speed zoom, while Figure 6 presents the evolution from the SSP/CSP base action to the final executed acceleration.

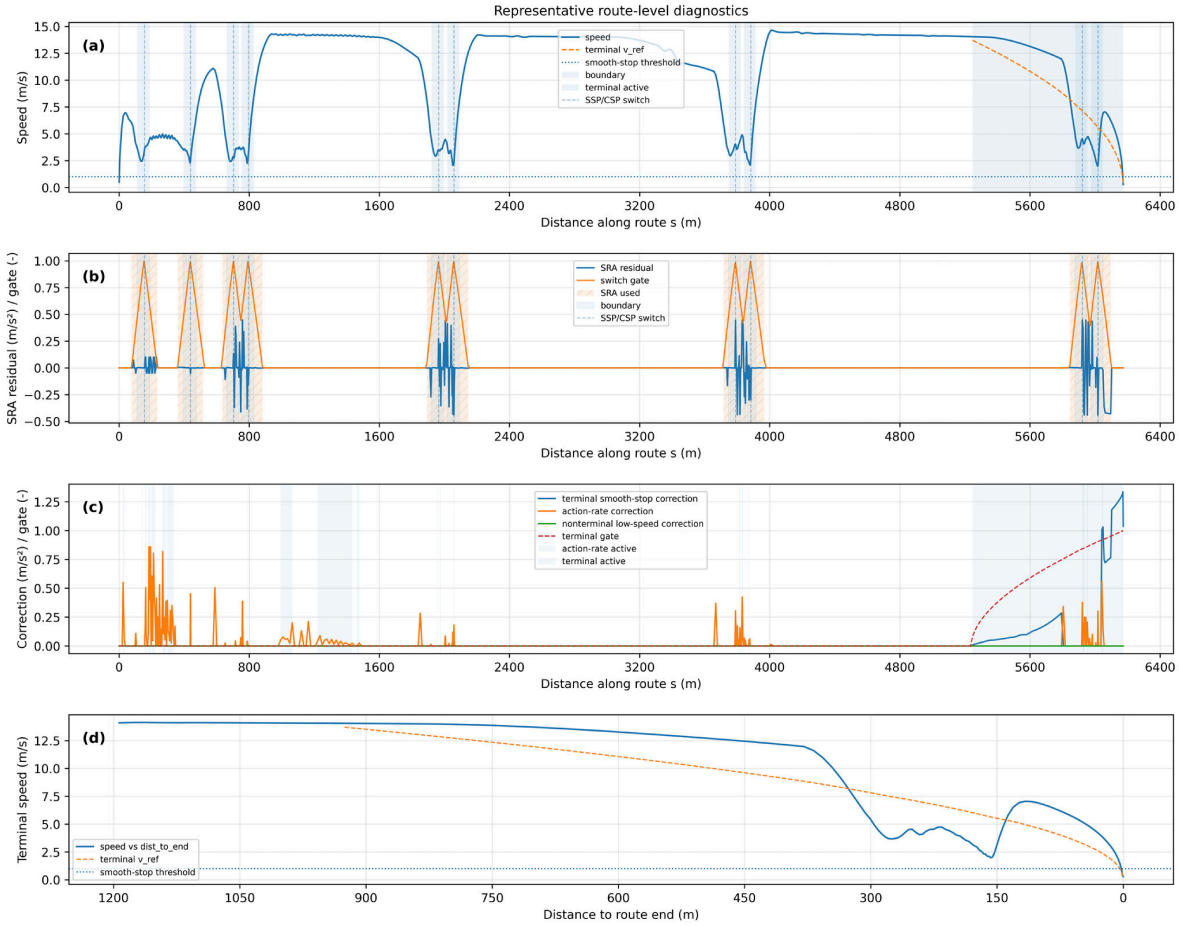


Figure 5. Representative trajectory-level diagnostics of SegCoord-Taxi. (**a**) Speed profile and terminal reference speed. (**b**) SRA residual and switch gate. (**c**) RFP correction components. (**d**) Terminal-speed zoom near the route end.

Figure 5a shows that the aircraft maintains a relatively high speed on long straight sections and reduces speed near geometrically complex regions and the terminal region. In the terminal active region, the reference speed decreases with the remaining distance, and the actual speed follows the decreasing trend before reaching the route end. Figure 5b shows that the SRA residual is mainly concentrated near positions with a high switch-gate value. This pattern is consistent with the design of SRA as a local residual correction module for straight–curve transition regions.

Figure 5c shows the correction components introduced by RFP. Action-rate correction appears near positions with rapid acceleration changes, while terminal correction is concentrated close to the route end. The non-terminal low-speed correction is not evident in this representative route, which is consistent with the projection diagnostics in Section 4.4. Figure 5d provides a magnified view of the terminal region. The final speed falls below the smooth-stop threshold, matching the aggregate terminal-feasibility results reported in the ablation and diagnostic experiments.

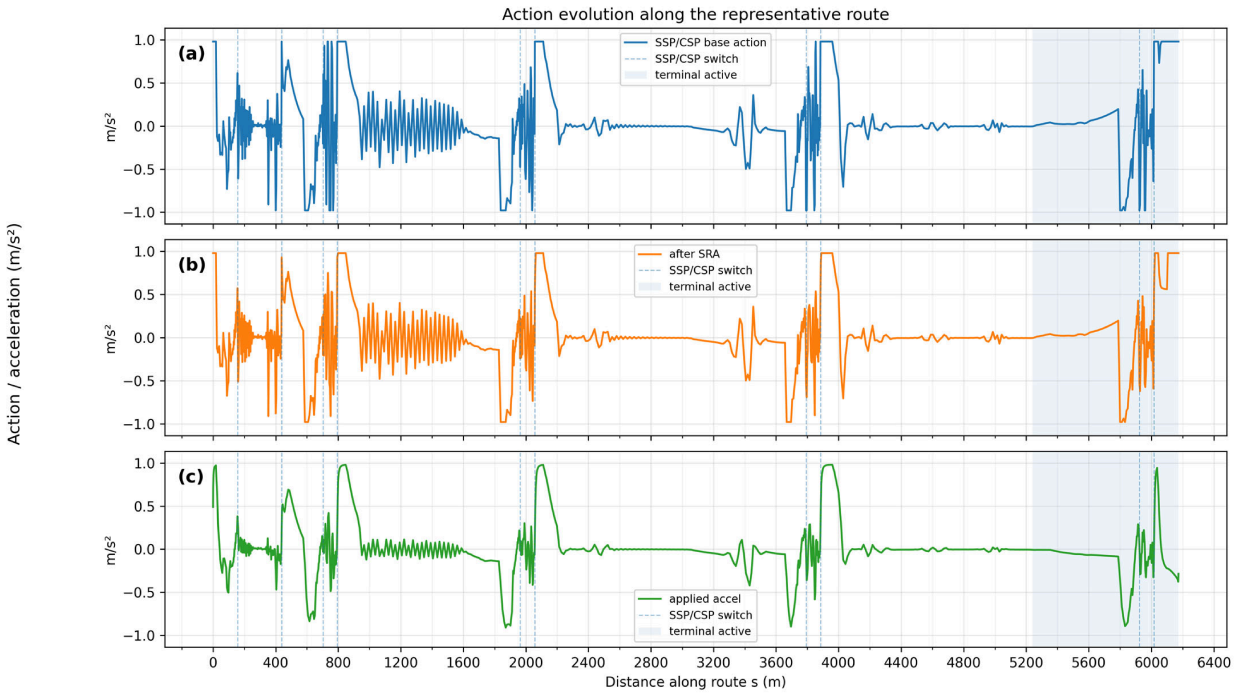


Figure 6. Action evolution along the representative route. (**a**) SSP/CSP base action. (**b**) Coordinated action after SRA. (**c**) Final executed acceleration after RFP.

Figure 6 shows how the acceleration command changes along the same route. The SSP/CSP base action exhibits noticeable local fluctuations around several transition regions. After SRA correction, part of the local action variation is reduced while the main structure of the base action is preserved. After RFP, the final executed acceleration is further constrained by action-rate and terminal feasibility requirements. This trajectory-level case provides a visual example consistent with the aggregate ablation and diagnostic results.

5. Conclusions

This study proposed SegCoord-Taxi, a geometry-aware segmented speed-control framework for aircraft taxiing along predefined airport surface routes. The framework decomposes full-route longitudinal speed control into four functional components: a Straight-Segment Policy, a Curved-Segment Policy, a Switch Residual Adapter, and Route-Level Feasibility Projection. These components are designed to handle straight-segment propulsion, curved-segment speed regulation, transition-region action continuity, and terminal feasibility, respectively.

Experiments on departure taxiing routes at Chengdu Tianfu International Airport show that SegCoord-Taxi improves terminal velocity feasibility and local transition smoothness, but does not dominate all efficiency and smoothness metrics. Compared with the unified Single PPO controller, the proposed framework achieves lower fuel proxy, mean jerk, boundary action variation, and final velocity, although it requires longer taxiing time. Compared with the Curvature-Aware Constrained Optimizer, SegCoord-Taxi achieves shorter taxiing time, lower boundary action variation, and lower final velocity, while the non-learning optimizer remains competitive in fuel proxy and mean jerk. These results indicate that SegCoord-Taxi should be interpreted as a trade-off-oriented control framework rather than as a uniformly superior controller across all metrics.

The ablation and diagnostic results further clarify how the complete framework achieves these effects. SRA reduces local discontinuities introduced by hard switching between straight and curved policies, and RFP improves terminal-speed regulation and action-rate consistency during full-route execution. The Single PPO + RFP diagnostic

comparison shows that adding feasibility projection to a unified policy improves terminal behavior, but it does not reproduce the full performance of SegCoord-Taxi. This indicates that the reported results arise from the combined control chain of segmented policy learning, residual adaptation, and deterministic feasibility handling.

This study has several limitations. First, it focuses on single-aircraft longitudinal speed control along predefined routes and does not consider multi-aircraft conflicts, dynamic route adjustment, pushback control, or real-time air traffic control instructions. Second, the experiments are based on a single airport surface network and a fixed set of predefined departure routes. Therefore, the results should be regarded as case-study evidence for the evaluated ZUTF scenario rather than as evidence of cross-airport generalization. Broader applicability will require validation on additional airport surface networks and operating scenarios. Third, the fuel-related metric is a proxy indicator and cannot replace high-fidelity engine fuel-consumption or emission models. Future work will extend the framework to multiple airport surface networks, incorporate higher-fidelity operational models, and investigate multi-aircraft cooperative taxiing under uncertainty and operational constraints. Future deployment-oriented studies will further require human-in-the-loop validation, real-time compatibility with air traffic control (ATC) clearances, robustness under uncertain traffic and route changes, and safety verification before operational use.

Overall, within the evaluated predefined-route single-aircraft setting, SegCoord-Taxi provides a structured approach for full-route taxiing speed control under predefined route constraints. By combining geometry-dependent segmented policies, local residual adaptation, and deterministic Route-Level Feasibility Projection, the framework improves terminal velocity feasibility and reduces switching-induced action discontinuities while maintaining an explicit trade-off among efficiency, smoothness, and feasibility.

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

Funding: This research was funded by the National Key R&D Program of China, grant number 2024YFB2605201; the Fundamental Research Funds for the Central Universities, grant number 26CAFUC04045; and the Civil Aviation Education Talent Project, grant numbers MHJY2025002 and MHJY2025003. The APC was funded by the authors.

Data Availability Statement: The data supporting the findings of this study are not publicly available due to restrictions related to airport surface network information and simulation scenario construction, but they may be made available from the corresponding author upon reasonable request.

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

Abbreviations

The following abbreviations are used in this manuscript:

SSP	Straight-Segment Policy
CSP	Curved-Segment Policy
PPO	Proximal Policy Optimization
MLP	Multilayer Perceptron
TCN	Temporal Convolutional Network
RFP	Route-Level Feasibility Projection
SRA	Switch Residual Adapter
MDP	Markov Decision Process
ATC	Air Traffic Control

References

1. Simaiakis, I.; Balakrishnan, H. Impact of Congestion on Taxi Times, Fuel Burn, and Emissions at Major Airports. *Transp. Res. Rec.* **2010**, *2184*, 22–30. [CrossRef]
2. Nikoleris, T.; Gupta, G.; Kistler, M. Detailed Estimation of Fuel Consumption and Emissions during Aircraft Taxi Operations at Dallas/Fort Worth International Airport. *Transp. Res. Part D Transp. Environ.* **2011**, *16*, 302–308. [CrossRef]
3. Ashok, A.; Balakrishnan, H.; Barrett, S.R.H. Reducing the Air Quality and CO2 Climate Impacts of Taxi and Takeoff Operations at Airports. *Transp. Environ.* **2017**, *54*, 287–303. [CrossRef]
4. Chen, H.; Solak, S. Lower Cost Departures for Airlines: Optimal Policies under Departure Metering. *Transp. Res. Part C Emerg. Technol.* **2020**, *111*, 531–546. [CrossRef]
5. Jiang, Y.; Xue, Q.; Wang, Y.; Cai, M.; Zhang, H.; Li, Y. Traffic Congestion Mechanism in Mega-Airport Surface. *Phys. A Stat. Mech. Its Appl.* **2021**, *577*, 125966. [CrossRef]
6. Liu, Y.; Hu, M.; Yin, J.; Su, J.; Qiao, P. Adaptive Airport Taxiing Rule Management: Design, Assessment, and Configuration. *Transp. Res. Part C Emerg. Technol.* **2024**, *163*, 104652. [CrossRef]
7. Maciejewska, M.; Kurzawska-Pietrowicz, P. Towards Sustainable Airport Operations: Emission Analysis of Taxiing Solutions. *Sustainability* **2025**, *17*, 8242. [CrossRef]
8. Tancredi, U.; Accardo, D.; Fasano, G.; Renga, A.; Rufino, G.; Maresca, G. An Algorithm for Managing Aircraft Movement on an Airport Surface. *Algorithms* **2013**, *6*, 494–511. [CrossRef]
9. Chen, J.; Weiszer, M.; Stewart, P.; Shabani, M. Toward a More Realistic, Cost-Effective, and Greener Ground Movement Through Active Routing—Part I: Optimal Speed Profile Generation. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 1196–1209. [CrossRef]
10. Chen, J.; Weiszer, M.; Locatelli, G.; Ravizza, S.; Atkin, J.A.; Stewart, P.; Burke, E.K. Toward a More Realistic, Cost-Effective, and Greener Ground Movement Through Active Routing: A Multiobjective Shortest Path Approach. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 3524–3540. [CrossRef]
11. Zhang, T.; Ding, M.; Zuo, H.; Chen, J.; Weiszer, M.; Qian, X.; Burke, E.K. An Online Speed Profile Generation Approach for Efficient Airport Ground Movement. *Transp. Res. Part C Emerg. Technol.* **2018**, *93*, 256–272. [CrossRef]
12. Weiszer, M.; Burke, E.K.; Chen, J. Multi-Objective Routing and Scheduling for Airport Ground Movement. *Transp. Res. Part C Emerg. Technol.* **2020**, *119*, 102734. [CrossRef]
13. Wang, X.; Brownlee, A.E.I.; Weiszer, M.; Woodward, J.R.; Mahfouf, M.; Chen, J. A Chance-Constrained Programming Model for Airport Ground Movement Optimisation with Taxi Time Uncertainties. *Transp. Res. Part C Emerg. Technol.* **2021**, *132*, 103382. [CrossRef]
14. Deng, W.; Zhang, L.; Zhou, X.; Zhou, Y.; Sun, Y.; Zhu, W.; Chen, H.; Deng, W.; Chen, H.; Zhao, H. Multi-Strategy Particle Swarm and Ant Colony Hybrid Optimization for Airport Taxiway Planning Problem. *Inf. Sci.* **2022**, *612*, 576–593. [CrossRef]
15. Wang, F.; Bi, J.; Xie, D.; Zhao, X. Quick Taxi Route Assignment via Real-Time Intersection State Prediction with a Spatial-Temporal Graph Neural Network. *Transp. Res. Part C Emerg. Technol.* **2024**, *158*, 104414. [CrossRef]
16. Zhang, T.; Weiszer, M.; Chen, J. The Feasibility of Follow-the-Greens for 4-Dimensional Trajectory Based Airport Ground Movements. *Transp. Res. Part C Emerg. Technol.* **2020**, *116*, 102632. [CrossRef]
17. Cheng, V.H.L.; Sweriduk, G.D. Trajectory Design for Aircraft Taxi Automation to Benefit Trajectory-Based Operations. In Proceedings of the 2009 7th Asian Control Conference, Hong Kong, China, 27–29 August 2009; pp. 99–104.
18. Ma, J.; Zhou, J.; Liang, M.; Delahaye, D. Data-Driven Trajectory-Based Analysis and Optimization of Airport Surface Movement. *Transp. Res. Part C Emerg. Technol.* **2022**, *145*, 103902. [CrossRef]
19. Ravizza, S.; Atkin, J.; Stewart, P.; Burke, E. Aircraft Taxi Time Prediction: Comparisons and Insights. *Appl. Soft Comput.* **2014**, *14*, 397–406. [CrossRef]
20. Wang, X.; Brownlee, A.E.I.; Woodward, J.R.; Weiszer, M.; Mahfouf, M.; Chen, J. Aircraft Taxi Time Prediction: Feature Importance and Their Implications. *Transp. Res. Part C Emerg. Technol.* **2021**, *124*, 102892. [CrossRef]
21. Park, D.K.; Kim, J.K. Influential Factors to Aircraft Taxi Time in Airport. *J. Air Transp. Manag.* **2023**, *106*, 102321. [CrossRef]
22. Yin, J.; Zhang, M.; Ma, Y.; Wu, W.; Li, H.; Chen, P. Prediction and Analysis of Airport Surface Taxi Time: Classification, Features, and Methodology. *Appl. Sci.* **2024**, *14*, 1306. [CrossRef]
23. Qiao, P.; Hu, M.; Yin, J.; Su, J.; Chen, Y.; Yin, M. An Adaptive Similar Scenario Matching Method for Predicting Aircraft Taxiing Time. *Aerospace* **2024**, *11*, 461. [CrossRef]
24. Yuan, L.; Liu, J.; Chen, H.; Fang, D.; Chen, W. A CNN-GRU Hybrid Model for Predicting Airport Departure Taxiing Time. *Aerospace* **2024**, *11*, 261. [CrossRef]
25. Okwir, S.; Amouzgar, K.; Ng, A.H. Exploring Prediction Accuracy for Optimal Taxi Times in Airport Operations Using Various Machine Learning Models. *J. Air Transp. Manag.* **2025**, *122*, 102684. [CrossRef]
26. Zhang, X.; Zhong, S.; Mahadevan, S. Airport Surface Movement Prediction and Safety Assessment with Spatial–Temporal Graph Convolutional Neural Network. *Transp. Res. Part C Emerg. Technol.* **2022**, *144*, 103873. [CrossRef]

27. Yang, X.; Yang, H.; Mao, Y.; Wang, Q.; Yin, S. Multi-Task Dynamic Spatio-Temporal Graph Attention Network: A Variable Taxi Time Prediction Model for Airport Surface Operation. *Aerospace* **2024**, *11*, 371. [CrossRef]
28. Thai, P.; Alam, S.; Lilith, N.; Nguyen, B.T. A Computer Vision Framework Using Convolutional Neural Networks for Airport-Airside Surveillance. *Transp. Res. Part C Emerg. Technol.* **2022**, *137*, 103590. [CrossRef]
29. Ganeriwala, P.; Bhattacharyya, S.; Gunther, S.; Kish, B.; Khan, M.; Dhadoti, A.; Neogi, N. AssistTaxi: A Comprehensive Dataset for Taxiway Analysis and Autonomous Operations. In *2023 International Conference on Machine Learning and Applications (ICMLA)*; IEEE: New York, NY, USA, 2024.
30. Pham, D.-T.; Tran, T.-N.; Alam, S.; Duong, V.N. A Generative Adversarial Imitation Learning Approach for Realistic Aircraft Taxi-Speed Modeling. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 2509–2522. [CrossRef]
31. Tran, T.-N.; Pham, D.-T.; Alam, S. Towards Greener Airport Surface Operations: A Reinforcement Learning Approach for Autonomous Taxiing. *Trans. Jpn. Soc. Aeronaut. Space Sci.* **2024**, *67*, 101–108. [CrossRef]
32. Xiang, Z.; Sun, H.; Zhang, J. Application of Improved Q-Learning Algorithm in Dynamic Path Planning for Aircraft at Airports. *IEEE Access* **2023**, *11*, 107892–107905. [CrossRef]
33. Yang, G.; Cheng, H.; Zhang, Y.; Zhang, W.; Fu, C.; Lian, G. A Multi-Agent Reinforcement Learning Approach to Autonomous Aircraft Taxiing with Taxiing Time, Fuel Consumption, and Emission Optimization. *Energy* **2026**, *346*, 140138. [CrossRef]
34. Rhanizar, A.; El Akkaoui, Z. A Survey About Learning-Based Variable Speed Limit Control Strategies: RL, DRL and MARL. In *Modern Artificial Intelligence and Data Science 2024: Tools, Techniques and Systems*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 565–580.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.