Proceedings of the Seventh International Conference on
Railway Technology: Research, Development and Maintenance
Edited by: J. Pombo
Civil-Comp Conferences, Volume 15, Paper 14.5
Civil-Comp Press, Edinburgh, United Kingdom, 2026
ISSN: 2753-3239, doi: 10.4203/ccc.15.14.5

From Offline Optimization to Real-Time Policy: Multi-Objective Neural Network Controllers for ATO

R. Parise[1] and M. Schenker[2]

[1] Institute of Vehicle Concepts, German Aerospace Center, Berlin, Germany
[2] Institute of Vehicle Concepts, German Aerospace Center, Stuttgart, Germany

Abstract

The transition towards autonomous trains necessitates intelligent automatic train operation systems capable of real-time, multi-objective trajectory optimization for highly efficient operations. While traditional open-loop optimal control methods are computationally demanding for online adaptation, data-driven approaches such as behavioral cloning inherently suffer from compounding errors, and standard reinforcement learning struggles to satisfy safety constraints in sparse-reward settings. To bridge this gap, this paper presents a novel learning framework for robust, closed-loop train control. We propose a two-step methodology: first, a neural policy is warm-started through pre-training on a dataset of constraint-satisfying open-loop trajectories generated via numerical optimization. Subsequently, the policy is fine-tuned in closed-loop via direct policy search within a continuous neural ordinary differential equation architecture. By backpropagating gradients directly through the system dynamics, a curriculum learning strategy is used to optimize multi-objective costs such as energy consumption, wear and punctuality. Simulation results demonstrate that this physics-informed approach effectively eliminates the compounding errors, adheres to nonlinear speed boundaries, and achieves real-time operational robustness. The resulting controller evaluates in

milliseconds and synthesizes energy-efficient driving profiles under varying operational disturbances. This framework enables automated trains to adapt their driving style on the tracks, reducing energy consumption and ensuring a more comfortable, on-time journey.

Keywords: automatic train operation, multi-objective optimization, neural network controller, physics-informed neural network, differentiable physics simulation, real-time control, optimal control.

1 Introduction

The future of modern railway networks demands higher levels of capacity, energy efficiency, and operational resilience. As networks transition toward higher Grades of Automation (GoA) over the European Train Control System (ETCS), the core challenge shifts from simply driving the train to dynamically optimizing trajectories in real-time. Europe's Rail Joint Undertaking explicitly notes that increasing railway line capacity and energy efficiency intrinsically starts with these higher grades of automation (GoA3/4) [1]. In this context, intelligent Automatic Train Operation (ATO) is no longer just about following a static speed curve. Its main function is complex trajectory generation that seamlessly balances energy consumption, passenger comfort and mechanical wear, while respecting timetables and infrastructure constraints [2]. This modernization is further augmented by complementary concepts such as Virtual Coupling Train Sets (VCTS) [3]. By enabling shorter headways, dynamic train convoy formations, and cooperative driving strategies, VCTS directly increases track capacity and facilitates coordinated energy-saving maneuvers, building upon previous foundational work in interspacing control [4, 5].

Despite this industry mandate, deploying optimal train control in real-time operational settings remains challenging. Traditional open-loop optimal control methods and finite horizon approaches such as Model Predictive Control (MPC) can compute highly efficient trajectories, but repeatedly solving these optimization problems online under continuously changing operational conditions, such as delays, dispatch updates, or varying traffic interactions, can become computationally demanding. At the same time, deriving analytically tractable closed-loop feedback policies based on approaches such as Hamilton–Jacobi–Bellman (HJB) equations or Pontryagin's Maximum Principle (PMP) is difficult in practice due to the nonlinearities of train dynamics and infrastructure constraints [6]. As a result, recent research has increasingly explored Machine Learning (ML) methods as a complementary approach for approximating real-time control policies.

While standard Supervised Learning (SL) and Reinforcement Learning (RL) are increasingly proposed to create fast closed-loop policies, they encounter some practical bottlenecks when applied to safety-critical ATO systems. Pure Supervised Learning for closed-loop control (such as behavioral cloning) inherently suffers from *covariate shift*:

as noted by Codevilla et al., end-to-end imitation systems experience a severe domain shift between offline training experience and online behavior, leading to compounding errors and dangerous drift during actual execution [7]. On the other hand, standard model-free RL algorithms rely on stochastic exploration. In heavily constrained railway environments governed by strict speed limits, random exploration frequently violates safety boundaries, leading to a highly sparse and unstable gradient landscape that makes convergence difficult to guarantee. Even existing safe RL methods that attempt to improve constraint handling through mechanisms such as action projection or safety shields often fall short, as they either introduce heavy online computational overhead or struggle to scale within continuous-time horizons.

Consequently, a critical challenge remains: how can we train a computationally efficient, closed-loop neural policy that guarantees strict safety compliance without relying on unstable stochastic exploration or heavy online corrections? To bridge this gap, this paper introduces a structured, physics-informed pipeline. We first initialize the policy through behavioral cloning (BC) pre-training, using a dataset derived from open-loop trajectory generation with hard constraints. We then optimize it through closed-loop fine-tuning via Direct Policy Search (DPS) in a continuous-time Neural ODE setting using soft constraint terms. By backpropagating smooth gradients natively through the system dynamics, our network learns safe, efficient driving profiles across the entire horizon without requiring discrete, computationally heavy online action corrections.

Instead of treating the vehicle dynamics as an unmodeled black-box environment, as is typical in model-free RL, the solution proposed in this paper embeds the mathematical system dynamics directly into the neural network training loop. By utilizing adjoint sensitivity methods, specifically Neural ODEs, which use a backward-in-time (in our case, backward-in-space) adjoint ODE to compute gradients through continuous dynamics [6]: gradients are propagated directly through the physics engine itself. This differentiable physics architecture allows the network to efficiently learn a closed-loop policy. Once trained, the policy evaluates in milliseconds, respects complex, nonlinear speed limits by staying anchored within the safe operational envelope provided by the initialization, and dynamically optimizes for multi-objective costs.

The goal of this paper is to introduce an ATO learning framework that combines open-loop trajectory generation, BC pre-training, and closed-loop fine-tuning via DPS with a Neural ODE policy. Moving beyond proof-of-concept, we aim to demonstrate the following:

- **A Novel Two-Step Training Framework:** A methodology that warm-starts a neural policy via BC pre-training before transitioning to physics-informed, closed-loop fine-tuning via DPS. This aims to eliminate the covariate shift problem and increases robustness to disturbances, while effectively preserving adherence to safety constraints by anchoring the optimization nearby feasible baseline open-loop trajectories.

- **Real-Time Operational Robustness:** The deployment of a real-time, closed-

loop neural controller that exhibits operational robustness to temporal disturbances (e.g., delays) and physical model uncertainties (e.g., varying train mass). By utilizing physical domain randomization within the differentiable simulator, the policy natively compensates for inertial imprecisions, circumventing the need for online optimization recalculations.

The remainder of this paper is structured as follows: Section 2 formulates the system modeling and the optimal control problem. Section 3 details the two-step methodology and training framework. Section 4 presents the closed-loop results and evaluates the policy's operational robustness. Finally, Section 5 concludes the paper and discusses future directions.

2 System Modeling and Problem Formulation

2.1 Problem Formulation

We consider a nonlinear optimal control problem formulated in the spatial domain, where the independent variable is the longitudinal position $s \in [0, S]$. Let $\mathcal{X} \subset \mathbb{R}^n$ denote the admissible state space and $\mathcal{U} \subset \mathbb{R}^m$ denote the admissible control space. The system dynamics are described by the nonlinear ordinary differential equation

$$\frac{dx}{ds}(s) = f(x(s), u(s), s), \qquad x(0) = x_0, \tag{1}$$

where $x(s) \in \mathcal{X}$ is the state vector, $u(s) \in \mathcal{U}$ is the control input, and $f : \mathcal{X} \times \mathcal{U} \times \mathbb{R} \to \mathbb{R}^n$ is a continuously differentiable dynamics function.

The optimal control problem seeks a bounded control trajectory $u \in L^\infty([0, S]; \mathcal{U})$ and a corresponding continuous state trajectory $x : [0, S] \to \mathcal{X}$ that minimize the cost functional

$$\mathcal{J}(x(\cdot), u(\cdot)) = \Phi(x(S)) + \int_0^S L(x(s), u(s), s)\, ds, \tag{2}$$

subject to the dynamics in (1) and the path constraints

$$g(x(s), u(s), s) \leq 0, \qquad \forall s \in [0, S]. \tag{3}$$

Here, $\Phi : \mathcal{X} \to \mathbb{R}$ denotes the terminal cost, while $L : \mathcal{X} \times \mathcal{U} \times \mathbb{R} \to \mathbb{R}$ denotes the running cost.

2.2 Open-Loop vs. Closed-Loop Control

In optimal control, solutions can be broadly classified into open-loop and closed-loop strategies. Solving the open-loop problem yields an optimal control trajectory $u^*(s)$ that is strictly a function of the independent spatial variable s. While mathematically

rigorous for nominal conditions, open-loop controls are inherently fragile to disturbances or unmodeled dynamics. Conversely, a closed-loop (or feedback) control policy seeks a mapping $u^*(x(s), s)$, formulating the control action as a direct function of the instantaneous state. Deriving a globally optimal closed-loop policy is significantly more mathematically demanding, yet it is desirable due to its inherent robustness to disturbances and model imprecision, allowing the system to self-correct upon deviations from the nominal path.

2.3 Dynamic Model

To model the vehicular dynamics accurately, we utilize a 3rd-order spatial-domain differential model. Let the state vector be defined as $x(s) = [t(s), v(s), a(s)]^T$, representing the elapsed time, velocity, and longitudinal acceleration at position s. The control variable $u(s) \in [-u_{\text{lim}}, u_{\text{lim}}]$ acts as a proxy for the target acceleration, subjected to a first-order lag to model the inertia of the powertrain and braking systems. The vehicle dynamics formulated over the space domain are defined as:

$$\begin{aligned} \frac{dt}{ds}(s) &= \frac{1}{v(s)} \\ \frac{dv}{ds}(s) &= \frac{a(s) - r(v(s))}{m \cdot v(s)} \\ \frac{da}{ds}(s) &= \frac{u(s) - a(s)}{\tau \cdot v(s)} \end{aligned} \tag{4}$$

where m denotes the vehicle mass and τ is the system time constant. The term $r(v(s))$ denotes the specific resistance (constant, rolling, and aerodynamic resistance terms) acting on the vehicle, modeled as $r(v) = c_0 + c_1 v + c_2 v^2$, where c_0, c_1, c_2 are the Davis coefficients.

2.4 Cost Functions and Constraints

We consider a fixed route of length S and a desired journey time T. The optimal control objective in the spatial domain is defined as the sum of a running cost and a terminal cost:

$$\mathcal{J} = \int_0^S L(x(s), u(s), s)\, ds + \Phi(x(S)). \tag{5}$$

The running cost L is constructed from weighted soft penalties. Let $v_{\text{lim}}(s)$ denote the spatial speed limit profile, and let $[x]^+ = \max(0, x)$. We use the Huber loss parametrized by δ

$$\rho_\delta(x) = \begin{cases} \frac{1}{2}x^2, & |x| \le \delta, \\ \delta\left(|x| - \frac{1}{2}\delta\right), & |x| > \delta. \end{cases} \tag{6}$$

The running cost includes terms for speed limits, coasting, energy efficiency, and component wear and is given by

$$L(x(s), u(s), s) = w_v \left[v(s) - v_{\lim}(s) - \delta_v\right]^+ + w_{\text{coast}} \left|u(s)\right| + w_{\text{energy}} \left[u(s)\right]^+ + w_{\text{wear}} \rho_{\delta_{\text{wear}}}\left(\frac{du}{ds}(s)\right). \quad (7)$$

The terminal cost aggregates end-point constraints (zero velocity and journey time close to timetable):

$$\Phi(x(S)) = w_{\text{term},v} \rho_{\delta_v}(v(S)) + w_{\text{term},t} \rho_{\delta_t}(t(S) - T). \quad (8)$$

2.5 Neural Network Policy Representation

We approximate the closed-loop optimal control law utilizing a fully connected feed-forward neural network parameterized by weights θ. The network policy π_θ maps the normalized instantaneous state and spatial coordinate directly to the control effort:

$$u(s) = \pi_\theta\big(s, t(s), v(s), a(s)\big) \in [-u_{\lim}, u_{\lim}] \quad (9)$$

The input state vector does not include explicit look-ahead features, such as distance to upcoming speed limit changes or gradient profiles. This design choice forces the network to implicitly learn and memorize the topological and speed-limit profile of the specific track it is being trained on. Consequently, a single, specialized neural network is trained exclusively per track.

3 Methodology

3.1 Two-Step Optimization Framework Overview

As established in the problem formulation, classical optimal control approaches such as HJB theory and PMP provide a principled foundation for optimal trajectory generation. However, their direct application to high-dimensional nonlinear train dynamics with state-dependent constraints is computationally intractable in practice. This gap motivates our data-driven, physics-informed approach, which avoids solving the full optimal control problem online while still leveraging the underlying system dynamics.

Attempting to directly learn the optimal policy π_θ via DPS using gradient descent from a randomly initialized neural network generally fails. In highly constrained railway environments, random initialization leads to trajectories that consistently violate safety constraints or fail to reach the terminal state, resulting in a sparse, uninformative gradient landscape. To circumvent this, we propose a structured two-phase framework: first, the network undergoes BC pre-training using a robust dataset of trajectories derived from open-loop trajectory generation. Once the network is safely anchored

within a feasible region of parameters, it transitions to closed-loop fine-tuning via DPS to systematically minimize the comprehensive multi-objective cost function defined in (5).

3.2 Open-Loop Trajectory Generation and Boundary Conditions

To generate the open-loop trajectory dataset for pre-training, we formulate a direct optimization problem using the CasADi library [8]. While our baseline database is computed using a customized, simplified cost formulation tailored for this problem, the numerical framework and multi-objective optimization structure are mathematically modeled after the principles of DLR's state-of-the-art SEnSOR (Smart Energy Speed Optimizer Rail) software [9]. This ensures our training trajectories are grounded in validated railway optimization methodologies.

For the generation of this dataset, the solver utilizes strict boundary conditions: the initial state is fixed at the origin $x(0) = [0, v_{\min}, 0]^T$ and the final state is strictly enforced at $x(S) = [T_{\text{target}}, v_{\min}, 0]^T$. Rather than the complex objective of Eq. 5, the CasADi solver minimizes a proxy for energy consumption $\int_0^S u(s)^2 \frac{1}{v(s)} ds$, strictly subject to the hard path constraint $v(s) \leq v_{\lim}(s)$. This creates a mathematically well-posed problem that yields highly efficient, rule-compliant baseline trajectories.

3.3 Target Journey Time Selection

To showcase the efficacy of the proposed framework, we define our target operational schedule by calculating the absolute minimum "all-out" travel time required to traverse the route, and subsequently appending a strictly defined temporal buffer ($t_{\text{buffer}} = 10\text{s}$). As illustrated in Fig. 1, this precise temporal relaxation provides the optimizer with the necessary degrees of freedom to strategically coast and minimize wear, while remaining heavily constricted by the physical speed limitations of the track.

3.4 Sampling Strategy for Initial Conditions

To ensure the closed-loop policy is robust to disturbances, it must be exposed to states outside the optimal path during training. However, because a well-controlled vehicle will naturally remain in close proximity to the reference trajectory, we want to sample with higher density in the vicinity of the nominal path.

To achieve this, exactly 50% of the initial training states are drawn from a Gaussian distribution centered tightly on the nominal open-loop reference trajectory ($\mu = x_{\text{ref}}(s), \sigma = \Sigma_{\text{ref}}$). The remaining 50% are sampled uniformly across the entirety of the feasible state space, bounded by $v \in [v_{\min}, v_{\lim}(s)]$, $a \in [-u_{\lim}, u_{\lim}]$, and t bounded between the all-out schedule and the maximum allowable delayed schedule. As shown in Fig. 2 below, this hybrid sampling strategy guarantees local precision near the optimum while ensuring global recoverability when subjected to unexpected operational

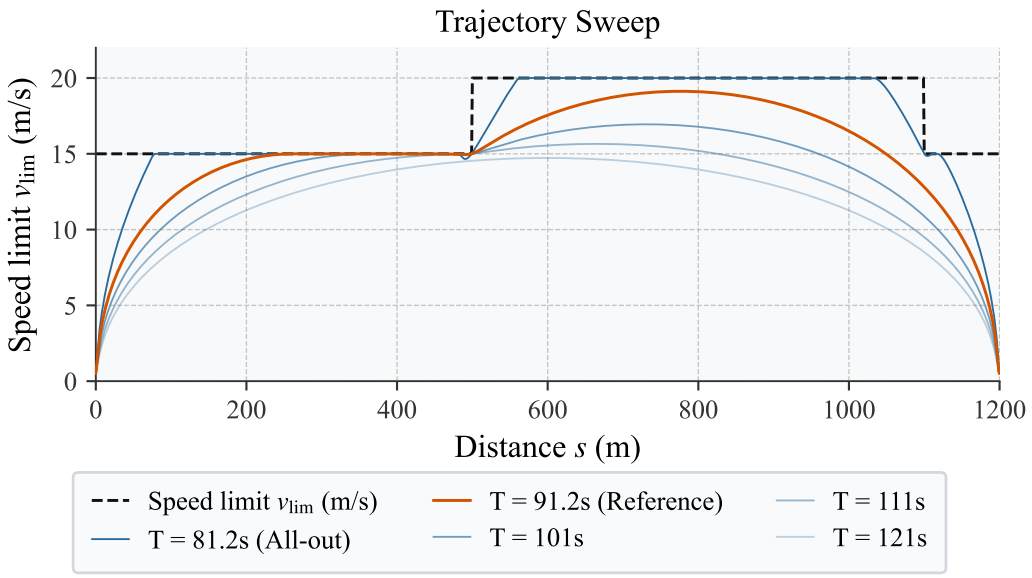

Figure 1: Open-loop trajectories mapping the reachable $t - s$ envelope. Blue lines illustrate optimal solutions for increasing journey times, bounded by the absolute fastest ("all-out") route in the darkest shade. The target operational schedule (the reference trajectory highlighted in red), is defined as a 10-second temporal buffer.

delays or state perturbations.

3.5 Algorithmic Implementation of the Training Framework

The first phases which includes data generation and pre-training are consolidated in Algorithm 1. Phase 1 builds a stable, constraint-satisfying foundation utilizing open-loop trajectories generated using CasADi alongside our targeted hybrid sampling strategy. Crucially, during the supervised learning loop, dynamic data augmentation (Gaussian noise injection on the temporal and kinematic states) is applied to simulate sensor disturbances and spatial deviations. This anchors the policy and builds local robustness before closed-loop training begins.

Once the network successfully mimics these baseline maneuvers, the continuous closed-loop fine-tuning (Phase 2) initializes the fine tuning DPS (Algorithm 2). We employ a curriculum learning approach for the multi-objective loss weights: The policy is set to first fine-tuned on terminal conditions and strict speed limit adherence, before progressively introducing complex soft penalties for coasting and mechanical wear. To prevent gradient shock during phase shifts, loss weights are smoothly interpolated over a warm-start transition period. The closed-loop trajectories are unrolled via a continuous Neural ODE, allowing exact gradients to backpropagate directly through the physical system dynamics, ensuring steady convergence toward a robust, multi-objective closed-loop policy. To ensure the resulting closed-loop policy is resilient to

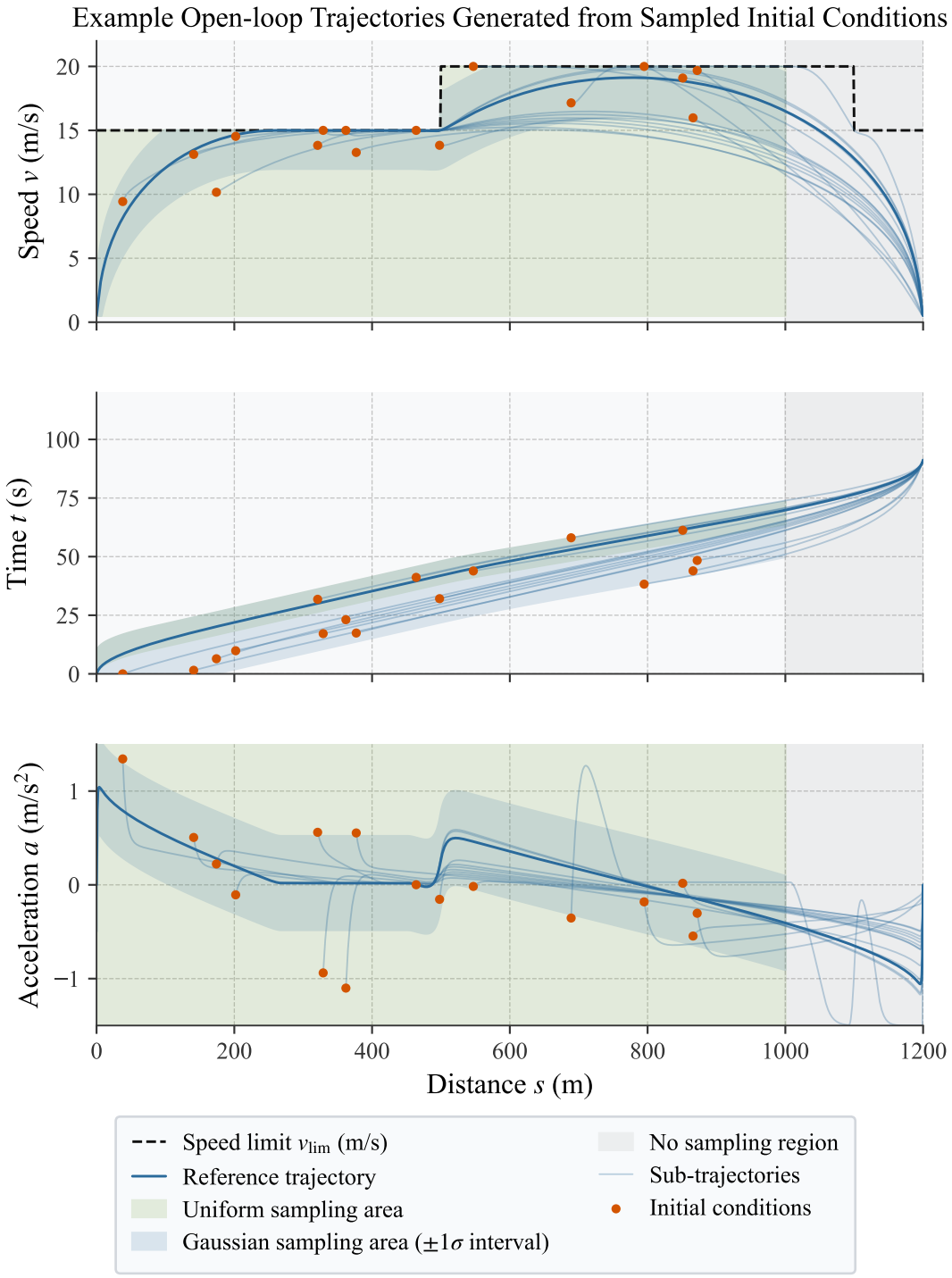

Figure 2: Example of some open-loop trajectories (blue lines) generated from varying initial conditions (red dots) (e.g., varying starting speeds and delays), forming the baseline dataset for BC. The sampling regions are the colored areas.

Algorithm 1 Open-Loop Trajectory Generation & Behavioral Cloning Pre-Training

Input: Speed limits $v_{\text{lim}}(s)$, final distance S, target journey time T_{target}.

Compute minimum journey time and reference schedule (Section 3.3).
Sample initial states $\mathcal{X}_0$ using the Gaussian/Uniform strategy (Section 3.4).
For each $x_0 \in \mathcal{X}_0$, compute the open-loop optimal trajectory $u^*(s)$ via CasADi (Section 3.2).
Compile state-action pairs into a supervised dataset $\mathcal{D}$.
for epoch $e = 1, \ldots, E_{\max}^{(1)}$ **do**
    Sample a mini-batch of states (t, v, a, s) and optimal controls u^* from $\mathcal{D}$.
    Apply Gaussian noise disturbances to input states (t, v, a).
    Train policy π_θ via MSE min. between predicted u and optimal u^*.
    if validation loss does not improve for $N_{\text{pat}}^{(1)}$ epochs **then** break
end for

unmodeled parameter imprecisions such as varying passenger loads, in every training iteration, the vehicle mass evaluated within the Neural ODE is sampled randomly from a uniform distribution around the nominal mass. The network backpropagates gradients through these varying physical realizations, forcing the policy to learn control strategies that are inherently robust to inertial deviations, without requiring an explicit mass estimator.

Algorithm 2 Closed-Loop Fine-Tuning via DPS using Curriculum Learning

Initialize π_θ with pre-trained Phase 1 weights (Section 2.5).
for each curriculum phase $P_i \in$ Phases **do**
    Initialize $EMA \leftarrow \infty$, $\mathcal{J}^* \leftarrow \infty$, $patience \leftarrow 0$.
    Re-initialize Adam optimizer and Cosine Annealing scheduler.
    for epoch $e = 1, \ldots, E^{(2)}_{\max}$ **do**
        Sample a batch of initial conditions x_0.
        If $e \leq E_{\text{warm}}$ **then** interpolate weights $\boldsymbol{w} \leftarrow (1-\alpha)\boldsymbol{w}_{i-1} + \alpha \boldsymbol{w}_i$ where $\alpha = \frac{e}{E_{\text{warm}}}$.
        Sample train mass $m \sim \mathcal{U}(m_{\min}, m_{\max})$ for passenger load uncertainty.
        Unroll trajectories by solving $\frac{dx}{ds} = f(x, \pi_\theta(x), s)$
        Compute active cost $\mathcal{J}^{(k)}$ and update weights $\theta \leftarrow \theta - \lambda \nabla_\theta \mathcal{J}^{(k)}$.
        Update $EMA \leftarrow (1 - \alpha_{ema}) \cdot EMA + \alpha_{ema} \cdot \mathcal{J}^{(k)}$.
        if $e > E_{\text{warm}}$ or $P_i = P_1$ **then**
            if $EMA < \mathcal{J}^* \cdot (1 - \tau)$ **then**
                Set $\mathcal{J}^* \leftarrow EMA$, $patience \leftarrow 0$, and save checkpoint $\theta^* \leftarrow \theta$.
            else
                Increment $patience \leftarrow patience + 1$.
            end if
            if $patience \geq N^{(2)}_{\text{pat}}$ **then break** (Early stop phase P_i)
            end if
        else
            Reset $\mathcal{J}^* \leftarrow \infty$, $patience \leftarrow 0$ ▷ Skip early stopping during transition
        end if
    end for
    Restore best weights for phase P_i: $\theta \leftarrow \theta^*$.
end for

4 Results and Evaluation

To evaluate the efficacy of our proposed approach, we compare the closed-loop performance of a policy trained strictly via Behavioral Cloning (BC) pre-training against our final policy fine-tuned via Direct Policy Search (DPS). As illustrated in the left column of Fig. 3, deploying the purely supervised pre-trained policy in a closed-loop environment exposes the classic vulnerabilities of BC. Small deviations from the open-loop trajectories due to disturbances compound over time (the aforementioned covariate shift phenomenon). This causes the train state to drift outside the training distribution, leading to speed limit violations. Consequently, while BC provides a useful and necessary initialization, it is fundamentally insufficient for safety-critical ATO where strict adherence to speed profiles is mandatory. In contrast, the right column of Fig. 3 demonstrates how DPS fine-tuning overcomes these limitations. By leveraging the closed-loop gradients provided by the differentiable simulator during training, the DPS policy learns to anticipate and actively correct spatial and temporal deviations, successfully eliminating the covariate shift and ensuring robust adherence to safety constraints.

Having established the safety and robustness of the DPS framework, we further evaluate the final policy's ability to balance multi-objective trade-offs. Fig. 4 demonstrates the closed-loop evaluation of the policy starting from $s = 0$ under varying initial temporal conditions (ranging from a generous time buffer of 15 seconds to an "all-out" zero-buffer schedule). The fully trained network autonomously adapts its driving strategy on the fly.

When a large time buffer (e.g., 15.0s) is available, the policy natively synthesizes a highly smooth profile maximizing passenger comfort, characterized by very gradual acceleration and deceleration with minimal control effort. As the schedule becomes more restricted (e.g., 10.0s and 5.0s buffers), the policy strategically trades smoothness for energy conservation; it applies sharper initial traction to reach operational speeds quickly, which allows it to utilize distinct, prolonged coasting phases. As the buffer approaches zero, the policy aggressively maximizes acceleration to prioritize strict punctuality. Across all temporal conditions, the policy safely adheres to the strict non-linear speed boundaries. This emergent behavior closely aligns with the energy-optimized speed profiles targeted in modern offline smart railway trajectory optimizers [9].

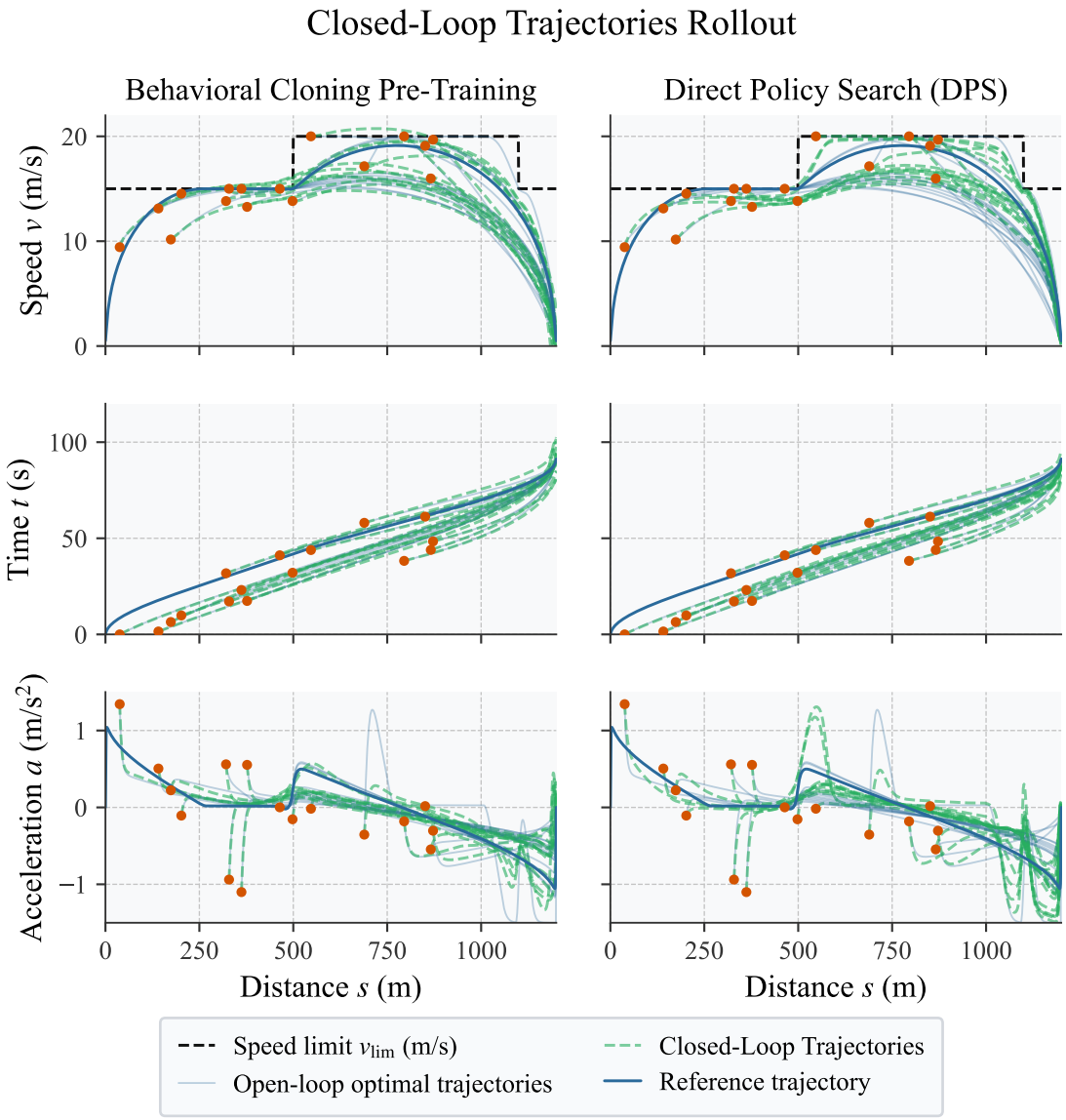

Figure 3: Side-by-side comparison of closed-loop trajectory rollouts under varying initial conditions. **(Left)** The purely pre-trained Behavioral Cloning (BC) policy suffers from compounding errors (covariate shift), causing the network to drift and violate speed limits. **(Right)** The Direct Policy Search (DPS) fine-tuned policy remains completely robust to perturbations, proactively correcting deviations to safely track reference profiles without violating operational constraints.

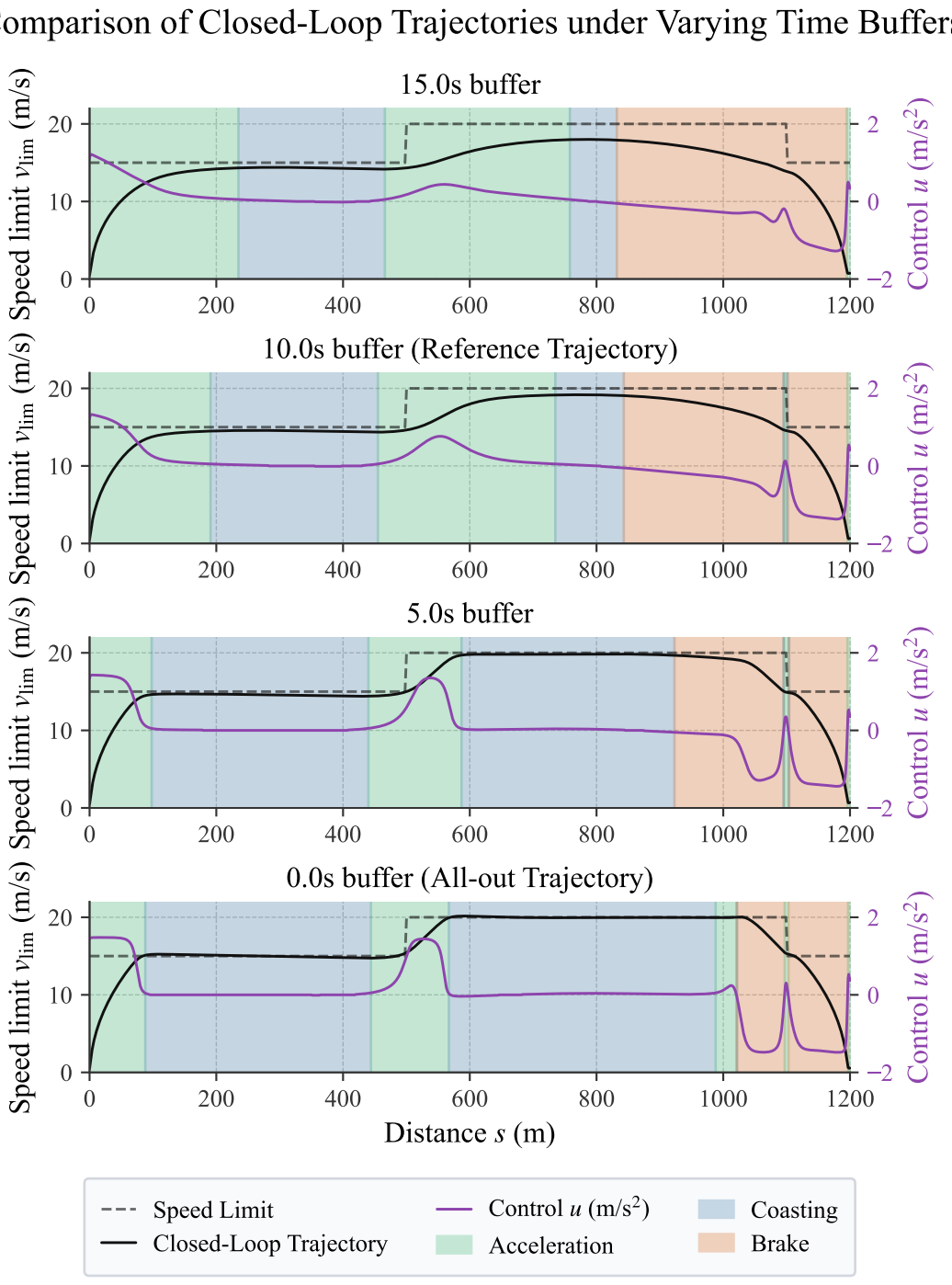

Figure 4: Final DPS closed-loop policy rollouts evaluated from the origin ($s = 0$) under varying schedule time buffers. The neural controller autonomously adapts its continuous multi-objective driving strategy (dynamically modulating traction, coasting, and braking phases) to safely respect track speed limits while satisfying strict arrival schedules.

5 Conclusions and Future Directions

In this paper, we presented a novel framework for learning robust and energy-efficient control policies for ATO. We demonstrated that relying solely on BC pre-training based on feasible open-loop trajectories yields policies susceptible to covariate shift and compounding errors, making them nonviable for safety-critical railway applications. By transitioning to closed-loop fine-tuning via DPS within a differentiable simulation framework, and employing a multi-phase curriculum learning approach, we successfully trained a policy that enforces strict adherence to speed boundaries. Furthermore, our approach naturally yields optimal driving strategies, seamlessly interpolating between traction, coasting, and braking phases to maximize energy efficiency without compromising punctuality.

Moving forward, several key paths remain for future research. First, a thorough stochastic analysis is required to mathematically validate the policy's robustness against a wider range of operational disturbances, such as dynamic drag variations, track adhesion loss, and sensor noise. Second, while the current continuous-time formulation heavily penalizes speed limit violations via soft constraints, safety-critical railway operations demand formal guarantees. Investigating the integration of hard constraints remains a potential next step to ensure absolute rule compliance.

Finally, scaling this framework to accommodate much longer track trajectories presents a unique challenge. Integrating the system dynamics via a Neural ODE over extended spatial horizons is susceptible to vanishing or exploding gradients during adjoint backpropagation. To mitigate this computational bottleneck, future iterations of this framework could transition from DPS to an Actor-Critic learning paradigm. By utilizing a value function approximation to estimate the cost-to-go, drawing upon the methodologies explored in [10], the framework can alleviate the reliance on end-to-end gradient backpropagation over the entire trajectory, ensuring stable and scalable training across extensive railway networks.

References

[1] Europe's Rail Joint Undertaking. (2024, Sep.) Increasing railway line capacity starts with increased automation. Accessed: 2026-05-27. [Online]. Available: https://rail-research.europa.eu/latest-news/increasing-railway-line-capacity-starts-with-increased-automation/

[2] Z. Wang, E. Quaglietta, M. G. Bartholomeus, and R. M. Goverde, "Assessment of architectures for automatic train operation driving functions," *Journal of Rail Transport Planning & Management*, vol. 24, p. 100352, 2022.

[3] Deutsches Zentrum für Luft- und Raumfahrt (DLR). (2026) Virtual coupling and cooperative train operations: Simulation and analysis. Accessed: 2026-05-27.

[Online]. Available: https://www.dlr.de/en/fk/research-and-transfer/research-services/simulation/virtuelles-kuppeln

[4] R. Parise, M. Schenker, and H. Dittus, "Energy impact of different intra-platoon spacing policies for virtually-coupled trains sets (vcts)," in *The Fifth International Conference on Railway Technology*, August 2022. [Online]. Available: https://elib.dlr.de/188151/

[5] R. Parise and K. Mullankuzhy, "Virtually coupled trains dynamics and energy efficiency: A simulation-based analysis," in *The Sixth International Conference on Railway Technology: Research, Development and Maintenance*, ser. Proceedings of the Sixth International Conference on Railway Technology: Research, Development and Maintenance. Civil-Comp Press, September 2024. [Online]. Available: https://elib.dlr.de/207224/

[6] L. Cheng, J. Cao, X. Yang, W. Wang, and Z. Zhou, "A train trajectory optimization method based on the safety reinforcement learning with a relaxed dynamic reward," *Discover Applied Sciences*, vol. 6, no. 9, p. 469, 2024.

[7] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9329–9338.

[8] J. Andersson, J. Gillis, G. Horn, J. Rawlings, and M. Diehl, "Casadi—a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2018.

[9] R. Radhakrishnan and M. Schenker, "Optimization algorithm for minimizing railway energy consumption in hybrid powertrain architectures: A direct method approach using a novel two-dimensional efficiency map approximation," *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, June 2024. [Online]. Available: https://elib.dlr.de/204962/

[10] R. Parise, "Optimal polynomial control for finite horizon problems in virtual train coupling," Master's thesis, Technische Universität Berlin, February 2025. [Online]. Available: https://elib.dlr.de/215067/