FROM OFFLINE OPTIMIZATION TO REAL-TIME POLICY: MULTI-OBJECTIVE NEURAL NETWORK CONTROLLERS FOR ATO

R. Parise, M. Schenker

Railways 2026 - Automatic Train Operation and Virtual Coupling

The Goal – Real-Time, State-Based Controller

Key objective

→ Develop an intelligent, real-time controller for Automatic Train Operation (ATO).

→ **Input**: Instantaneous state ***(t, v, a)***

and position ***(s)***. No explicit lookahead features

→ **Output**: Continuous control effort ***(u)*** (traction/braking).

→ **Requirement**: Must evaluate in milliseconds to enable advanced concepts like Virtual Coupling (VCTS).

The challenge

→ Balancing multi-objective goals: Energy efficiency, punctuality, passenger comfort, and mechanical wear.

→ Strictly respecting non-linear track speed limits and infrastructure constraints

The Naive Approach and Pitfalls

The "Classic" ML Solution

→ Build a standard Neural Network (NN).

→ Craft a cost function (energy + time + comfort).

→ Add speed limits as soft constraints (penalties).

→ Train from scratch using gradient descent.

The Reality (Why it fails)

→ **Sparse Gradients**: In heavily constrained railway environments, random initialization leads to constant safety violations.

→ **No Convergence**: The network fails to reach the terminal state, resulting in a sparse, uninformative gradient landscape.

→ **Unsafe:** Pure Reinforcement Learning (RL) relies on stochastic exploration, which is fundamentally unsafe for trains.

Proposed Framework Overview



1. Open Loop Generation

Pre-calculate optimal examples using an interior-point optimizer (CasADi/IPOPT) inspired in DLR's sensor software.



2. Behavioral Cloning

Pre-train the Neural Network to mimic these examples using supervised learning.



3. Closed-Loop Tuning

Train the NN with gradient descent using continuous Neural ODEs and curriculum learning.

Step 1: Open-Loop Optimization

Why offline optimization?

→ Solvers like IPOPT are highly reliable for handling **hard constraints** (trajectories never violate speed limits).

→ **The Catch**: They are computationally expensive online and work best with relatively simple, convex cost functions.

Our Approach

→ Use CasADi to generate a dataset of feasible, mathematically optimal open-loop trajectories.

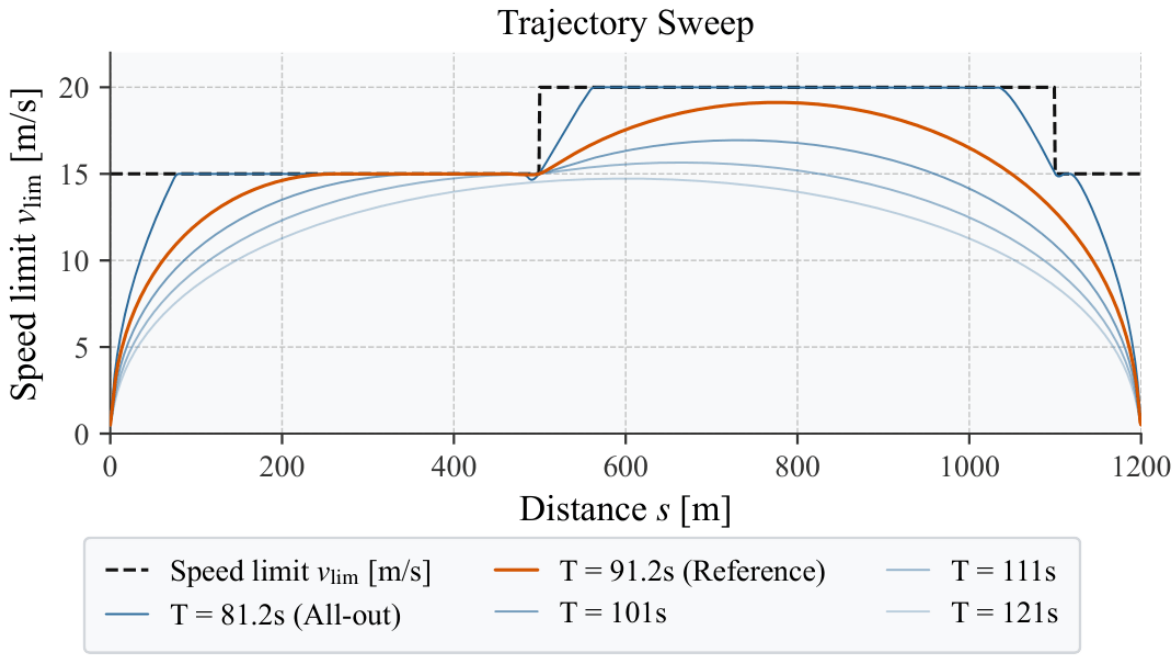
→ Use a simplified proxy cost function (minimizing squared control effort).

→ This creates our "ground truth" dataset of safe driving profiles.

Step 1 (cont.): Target Schedule & Sampling

Defining the Baseline

→ **Nominal Trajectory:** Calculated as the "all-out" (maximum speed) trajectory + a 10-second temporal buffer.



Step 1 (cont.): Target Schedule & Sampling

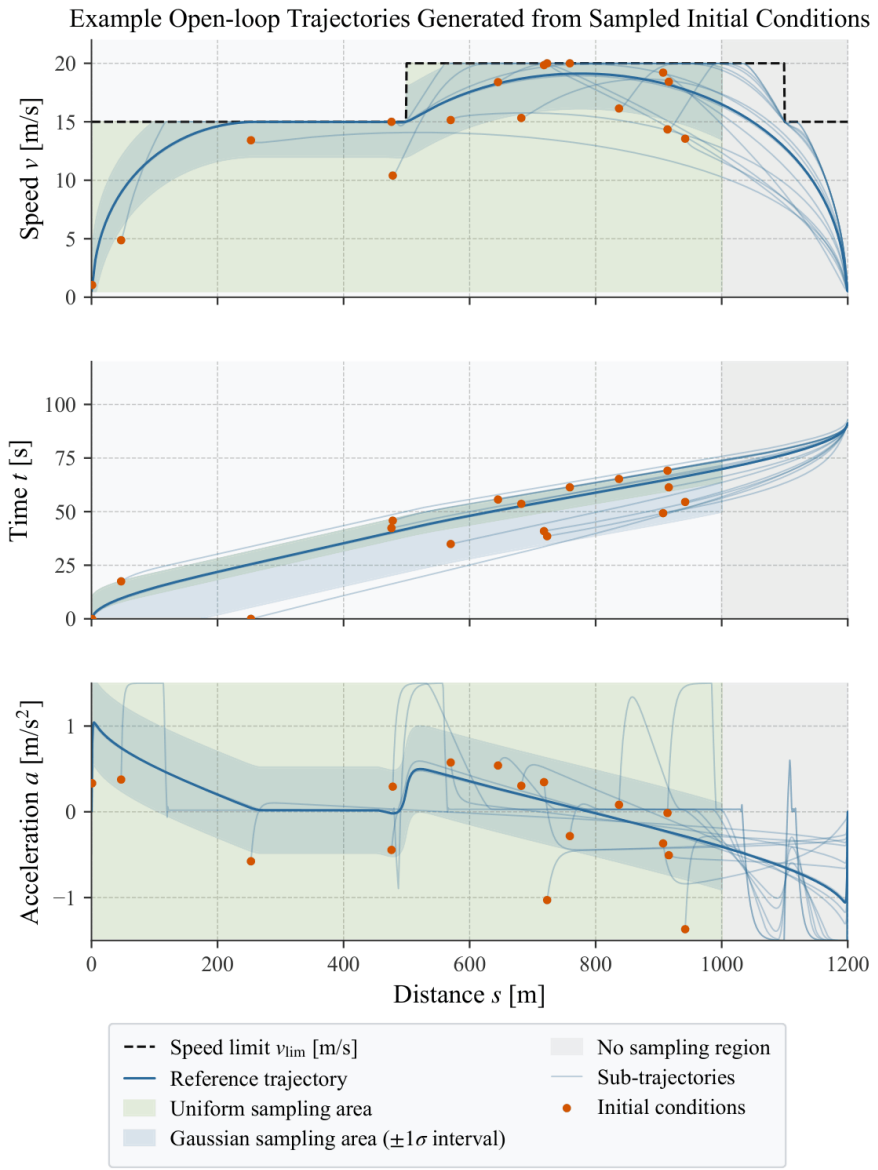
Defining the Baseline

→ **Nominal Trajectory:** Calculated as the "all-out" (maximum speed) trajectory + a 10-second temporal buffer.

Sampling Strategy

→ We generate solutions for many initial conditions around this nominal path (varying starting speeds and delays).

→ Ensures a robust dataset for the Neural Network to learn from.



Step 2: Behavioral Cloning (BC)

Training the Network

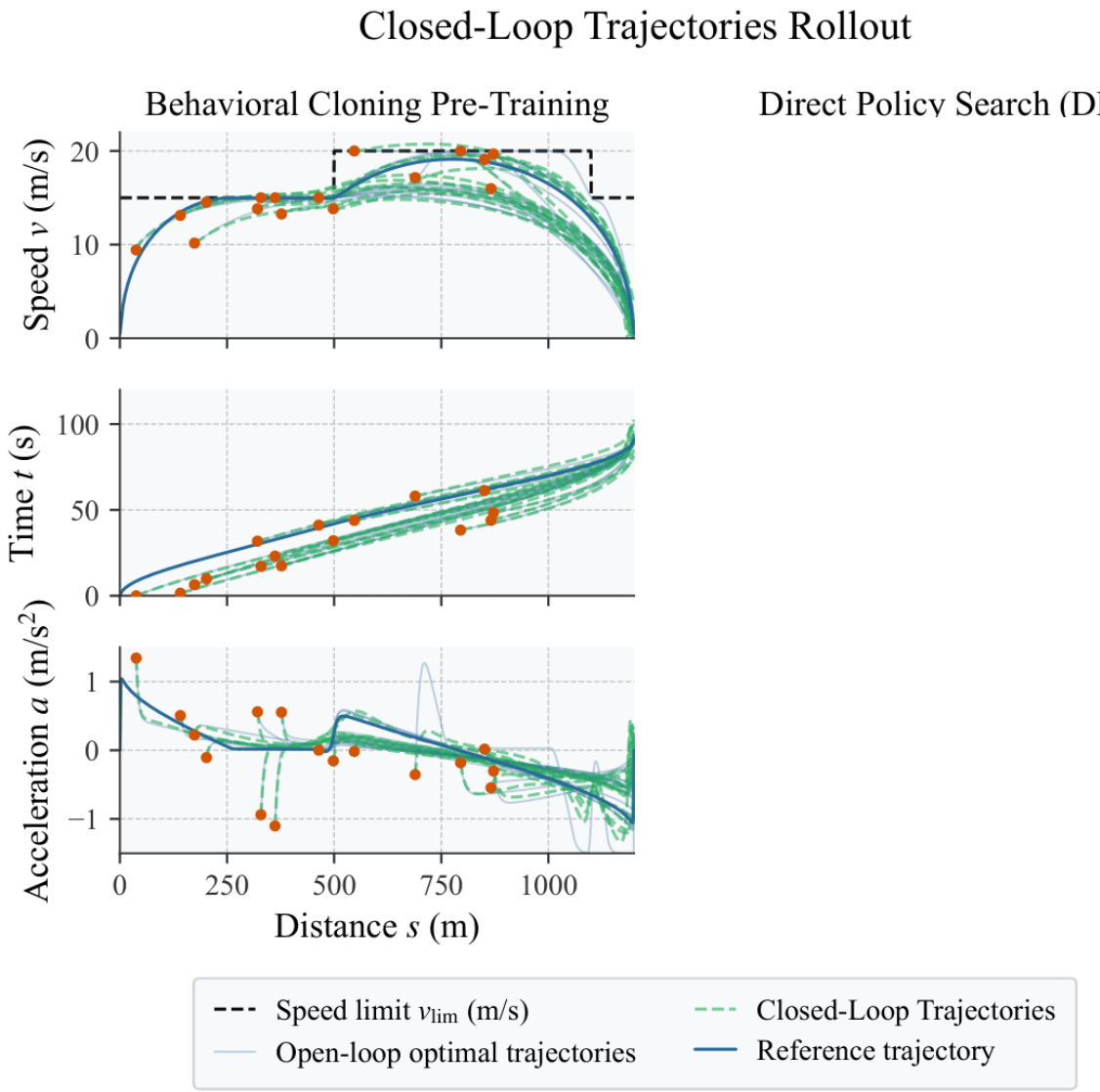
→ Train the NN to mimic the IPOPT trajectories.

→ Inject stochastic noise at each step to improve robustness.

The Limitation: Covariate Shift

→ Policy is still limited to the simplified cost function

→ Compounding Errors: Small deviations push the train into unvisited states



Step 3: Closed-Loop Fine-Tuning

Unlocking Multi-Objective

→ Start with the warm-started weights from Step 2 (already safe and feasible).

→ Perform Direct Policy Search (DPS) by backpropagating gradients directly through the physical system.

→ Utilizes Neural ODEs (adjoint method) for continuous physics simulation.

Curriculum Learning

→ **Phase 1**: Start with the simple CasADi cost + soft constraints for speed limits.

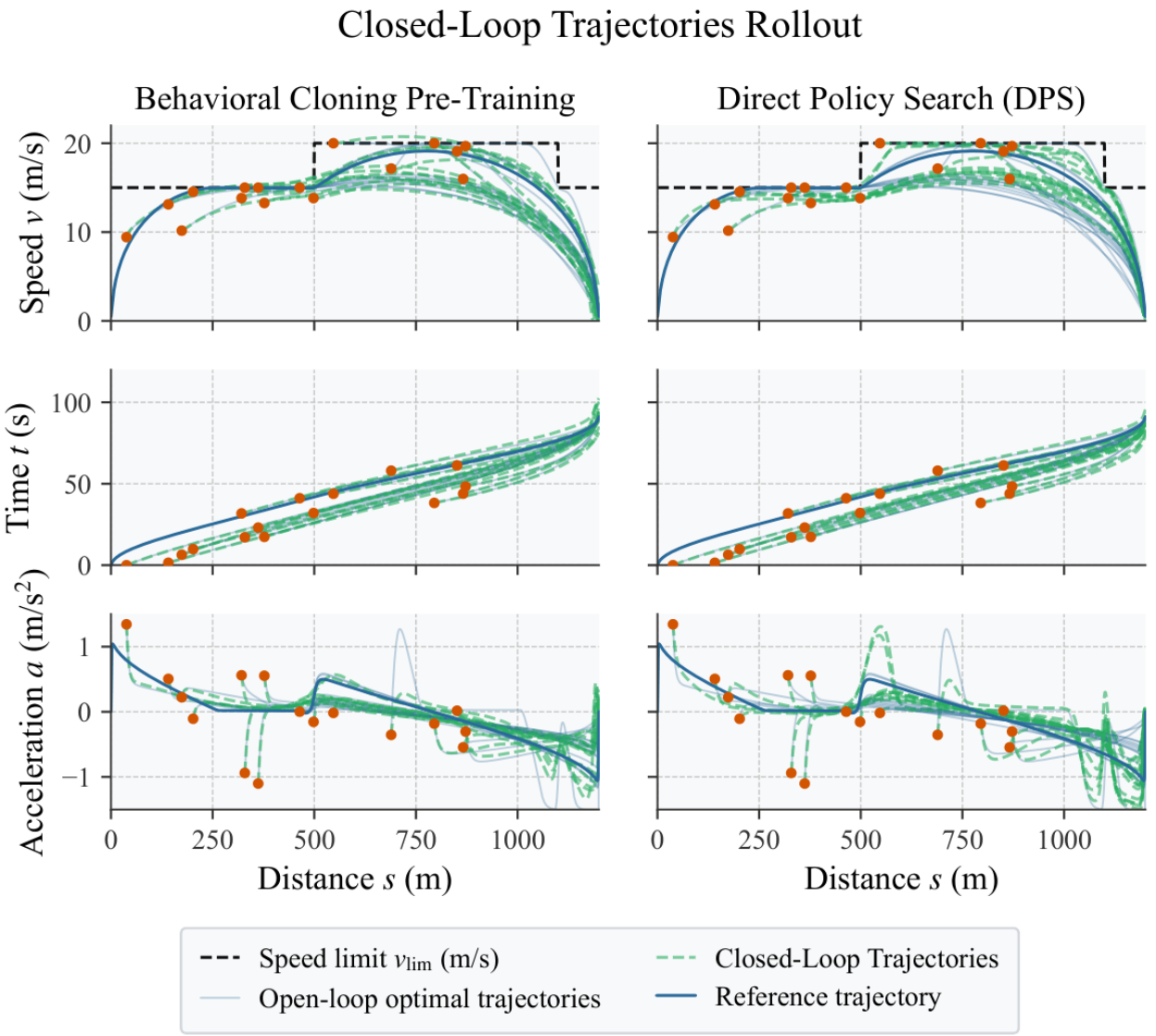
→ **Phase 2**: Progressively introduce complex costs to promote coasting.

→ **Phase 3**: Maximize passenger comfort and minimize mechanical wear.

Evaluating the Final Policy

Robustness and Adaptability

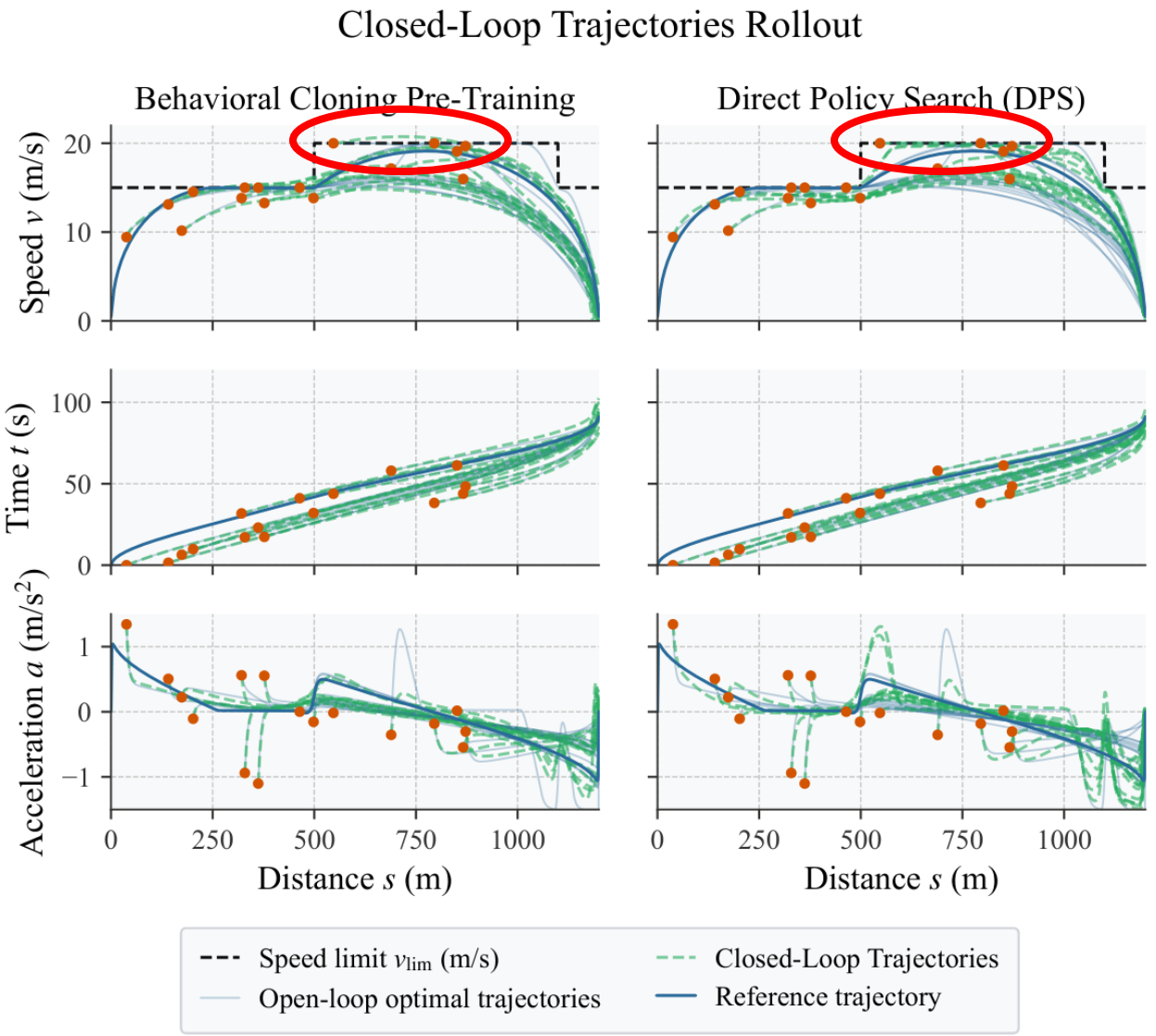
→ **No Covariate Shift**: The DPS policy actively corrects spatial and temporal deviations.



Evaluating the Final Policy

Robustness and Adaptability

→ **No Covariate Shift**: The DPS policy actively corrects spatial and temporal deviations.



Evaluating the Final Policy

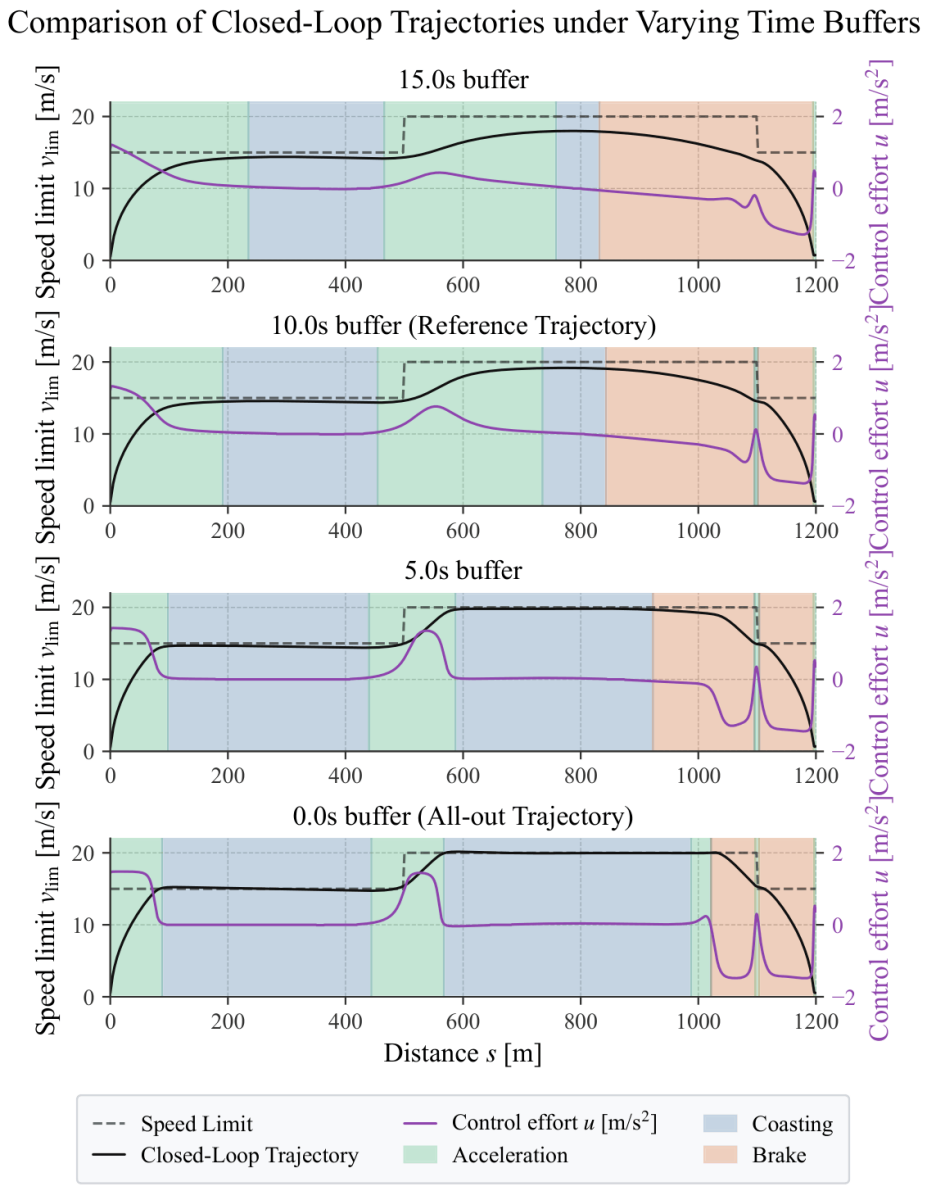
Robustness and Adaptability

→ **No Covariate Shift**: The DPS policy actively corrects spatial and temporal deviations.

→ Dynamic Driving Styles: The NN autonomously adapts to available time buffers.

→ Large buffer (15s): Smooth profile, some coasting.

→ Zero buffer (0s): Aggressive traction, mimics the all-out trajectory



Conclusion & Future Directions

Summary

- Created a fast, computationally cheap neural controller.
- Overcame pure ML convergence issues via open-loop dataset generation.
- Overcame pure BC drift issues via differentiable physics (Neural ODEs) and curriculum learning.
- Policy is natively robust to parameter uncertainties (e.g., varying passenger mass).

Future Work

- Explore Actor-Critic RL paradigms to estimate "cost-to-go" and scale the framework.
- Adapt to much longer track trajectories.
- Integration of hard-constraint shielding layers for absolute safety guarantees.

Imprint

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

Date: 2026-08-26

Authors: Riccardo Parise, Moritz Schenker

Institute: German Aerospace Center - Vehicle Concepts

Image Credits: All images „DLR (CC BY-NC-ND 3.0)“, if not specified