TU Delft

Delft University of Technology

Document Version
Final published version

Licence
CC BY

Citation (APA)
Mallick, S., Battocletti, G., Dong, Q., Dabiri, A., & De Schutter, B. (2025). Learning-Based MPC for Fuel Efficient Control of Autonomous Vehicles With Discrete Gear Selection. *IEEE Control Systems Letters*, *9*, 1117-1122. https://doi.org/10.1109/LCSYS.2025.3575335

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Learning-Based MPC for Fuel Efficient Control of Autonomous Vehicles With Discrete Gear Selection

Samuel Mallick, Gianpietro Battocletti, Qizhang Dong, Azita Dabiri, and Bart De Schutter, *Fellow, IEEE*

***Abstract*—Co-optimization of both vehicle speed and gear position via model predictive control (MPC) has been shown to offer benefits for fuel-efficient autonomous driving. However, optimizing both the vehicle's continuous dynamics and discrete gear positions may be too computationally intensive for a real-time implementation. This letter proposes a learning-based MPC scheme to address this issue. A policy is trained to select and fix the gear positions across the prediction horizon of the MPC controller, leaving a significantly simpler continuous optimization problem to be solved online. In simulation, the proposed approach is shown to have a significantly lower computation burden and a comparable performance, with respect to pure MPC-based co-optimization.**

***Index Terms*—Model predictive control (MPC), autonomous vehicles, learning.**

I. Introduction

FOR OPTIMAL control of autonomous vehicles (AVs), model predictive control (MPC) is a powerful and prevalent method [9], [15]. In this context, co-optimization of an AV's speed and gear-shift schedule is a promising approach to achieve high-performing and fuel-efficient autonomous driving; however, considering the gear-shift schedule requires the online solution of a mixed-integer nonlinear program (MINLP) [1], for which the computational burden can be intensive.

To address this issue the MINLP can be made easier to solve by relaxing the problem or by finding heuristic numerical solutions [5], [11]. However, the relaxed problem can still be difficult to solve, and approximate solutions can be suboptimal. Alternatively, a decoupled approach can be used. In this case, first the speed is optimized, and next a gear-shift schedule is selected for the given speed using, e.g., a learning-based gear controller [14], or dynamic programming [12]. However, decoupling speed control and gear control is suboptimal compared to the joint optimal control of both together. Finally, the computational burden can be alleviated by using a learning-based controller that controls both speed and gear-shift schedule in place of an MPC controller [7]; however, in contrast to an MPC controller, a learning-based controller is not able to guarantee constraint satisfaction.

In light of the above issues, this letter presents a novel learning-based MPC controller for the co-optimization of speed and gear-shift schedule for an AV. Taking inspiration from [4], a learned policy selects and fixes the gear positions across the prediction horizon of the MPC controller, such that optimal control and constraint satisfaction are handled by a nonlinear program (NLP), rather than an MINLP. A neural network (NN)-based policy is proposed where, to address the exponential growth of the policy's action space with the prediction horizon, a recurrent architecture is used. The policy learns to select gears that are optimal for the original optimization problem, rather than decoupling the gear-shift schedule from the speed control, such that, in contrast to [12], [14], the notion of co-optimization is retained. In this way, the MPC controller is able to consider the gear and powertrain dynamics without optimizing explicitly over discrete inputs, in contrast to [5], [11]. Furthermore, unlike pure learning-based controllers [7], the use of the MPC controller gives constraint satisfaction. To this end, we propose a backup gear-shift schedule that can guarantee feasibility of the MPC optimization problem. Finally, due to the recurrent architecture, the policy, once trained for a specific prediction horizon, generalizes over prediction horizons without the need for retraining.

II. Problem Setting

Consider the vehicle and powertrain models [11]

$$\begin{aligned} T(t)n(t) &= g(t) + Cv^2(t) + ma(t) + F_{\mathrm{b}}(t), \\ \omega(t) &= (30/\pi) \cdot n(t)v(t), \end{aligned} \tag{1}$$

with t continuous time, a the acceleration, v the velocity, and m the mass of the vehicle. Furthermore, C is the wind drag

Received 14 March 2025; revised 2 May 2025; accepted 22 May 2025. Date of publication 30 May 2025; date of current version 7 July 2025. This work was supported by the European Research Council (ERC) through the European Union's Horizon 2020 Research and Innovation Programme (ERC Advanced Grant CLariNet) under Grant 101018826. Recommended by Senior Editor L. Menini. *(Corresponding author: Samuel Mallick.)*

The authors are with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: s.h.mallick@tudelft.nl; g.battocletti@tudelft.nl; q.dong@tudelft.nl; a.dabiri@tudelft.nl; b.deschutter@tudelft.nl).

Digital Object Identifier 10.1109/LCSYS.2025.3575335

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

coefficient, F_b is the brake force, T is the engine torque, and ω is the engine speed. The friction function

$$g(t) = \mu m g \cos\big(\alpha(t)\big) + mg \sin\big(\alpha(t)\big), \tag{2}$$

with μ the rolling friction constant and g the gravitational acceleration, defines the road friction for road angle α, which, for simplicity of presentation, is assumed to be constant, i.e., $\alpha(t) = \alpha$ and $g(t) = G$. The lumped gear ratio $n(t) = z(j(t))z_f/r$ is determined by the final drive ratio z_f, the wheel radius r, and the transmission gear ratio z, a discrete variable selected by the gear position $j \in \{1, \dots, 6\}$. To model engine dynamics the torque rate of change is constrained as $|\dot{T}(t)| \leq \Delta T_{\max}$. Furthermore, consider the fuel model [11]

$$\dot{m}_f(t) = c_0 + c_1\omega(t) + c_2\omega(t)T(t), \tag{3}$$

with c_0, c_1, and c_2 constants. The variables F_b, T, and ω are physically bounded above and below, e.g., $T_{\min} \leq T \leq T_{\max}$. Note that the bounds on ω implicitly bound v between

$$v_{\min} = \frac{\pi \cdot \omega_{\min} r}{30 \cdot z(1) z_f} \quad \text{and} \quad v_{\max} = \frac{\pi \cdot \omega_{\max} r}{30 \cdot z(6) z_f}. \tag{4}$$

For convenience, in the following, we define a function that maps a speed and gear choice to the engine speed

$$\omega(v, j) = \frac{30 \cdot v \cdot z(j) z_f}{r\pi}. \tag{5}$$

We consider the control of an AV to track a reference trajectory in a fuel efficient manner. Denote the vehicle position, reference position, and reference velocity at time t as $p(t)$, $p_{\text{ref}}(t)$, and $v_{\text{ref}}(t)$, respectively. The performance metric is

$$P = \sum_{k=0}^{K_{\text{sim}}} \beta L_t\Big(\big[p(k\Delta t), v(k\Delta t)\big]^\top, \big[p_{\text{ref}}(k\Delta t), v_{\text{ref}}(k\Delta t)\big]^\top\Big) + L_f\big(v(k\Delta t), T(k\Delta t), j(k\Delta t)\big), \tag{6}$$

where $\beta > 0$ expresses the importance of tracking against fuel efficiency, and k is a discrete-time counter for time steps of Δt seconds. The tracking cost

$$L_t(x, y) = (x - y)^\top Q (x - y) \tag{7}$$

quadratically penalizes deviations from the reference trajectory, with $Q \in \mathbb{R}^{2\times 2}$ a positive-definite weighting matrix. The fuel cost $L_f(v, T, j) = \Delta t(c_0 + c_1\omega(v, j) + c_2\omega(v, j)\, T)$ penalizes the fuel consumption over a time step.

III. Learning-Based MPC

In this section we introduce the proposed controller for the task. Defining the state, reference state, and control input as

$$\begin{aligned} x(k) &= \big[p(k\Delta t), v(k\Delta t)\big]^\top, \\ x_{\text{ref}}(k) &= \big[p_{\text{ref}}(k\Delta t), v_{\text{ref}}(k\Delta t)\big]^\top, \\ u(k) &= \big[T(k\Delta t), F_b(k\Delta t), j(k\Delta t)\big]^\top, \end{aligned} \tag{8}$$

(1) can be approximated with the discrete-time dynamics $x(k+1) = f(x(k), u(k))$, with

$$f(x, u) = \begin{bmatrix} x_1 + x_2 \Delta t \\ x_2 + \frac{\Delta t}{m}\Big(\frac{u_1 z(u_3) z_f}{r} - C x_2^2 - u_2 - G\Big) \end{bmatrix}, \tag{9}$$

where subscripts select an element of a vector, e.g., x_i is the ith element of the vector x.

A. Mixed-Integer Nonlinear MPC

Consider an MPC scheme with prediction horizon $N > 1$ defined by the following MINLP:

$$\begin{aligned} J\big(x(k),\ \mathbf{x}_{\text{ref}}(k)\big) = \min_{\mathbf{x}(k), \mathbf{u}(k)} & \sum_{i=0}^{N} \beta L_t\big(x(i|k), x_{\text{ref}}(i+k)\big) \\ & + \sum_{i=0}^{N-1} L_f\big(x_2(i|k), u_1(i|k), u_3(i|k)\big) && \text{(10a)} \\ \text{s.t.} \quad & x(0|k) = x(k) && \text{(10b)} \\ & \text{for} \quad i = 0, \dots, N-1: \\ & \quad x(i+1|k) = f\big(x(i|k), u(i|k)\big) && \text{(10c)} \\ & \quad |x_2(i+1|k) - x_2(i|k)| \leq a_{\max} \Delta t && \text{(10d)} \\ & \text{for} \quad i = 0, \dots, N-2: \\ & \quad |u_1(i+1|k) - u_1(i|k)| \leq \Delta T_{\max} \Delta t && \text{(10e)} \\ & \quad |u_3(i+1|k) - u_3(i|k)| \leq 1 && \text{(10f)} \\ & \big(\mathbf{x}(k), \mathbf{u}(k)\big) \in \mathcal{C} && \text{(10g)} \end{aligned}$$

where $x(i|k)$ and $u(i|k)$ are the predicted states and inputs, respectively, i steps into the prediction horizon of the MPC controller at time step k. Furthermore, bold variables gather a variable over the prediction horizon, e.g.,

$$\begin{aligned} \mathbf{x}(k) &= \big[x^\top(0|k), \dots, x^\top(N|k)\big]^\top, \\ \mathbf{x}_{\text{ref}}(k) &= \big[x_{\text{ref}}^\top(k), \dots, x_{\text{ref}}^\top(k+N)\big]^\top. \end{aligned} \tag{11}$$

If no solution exists for (10) $J(x(k), \mathbf{x}_{\text{ref}}(k)) = \infty$ by convention. The constraint (10e) enforces the engine torque dynamic behavior, (10f) prevents skipping gears when shifting, and (10d) limits acceleration to $a_{\max}$. The bounds on engine torque, engine speed, and brake force are grouped in

$$\begin{aligned} \mathcal{C} = \Big\{ (\mathbf{x}, \mathbf{u}) \Big| \ & T_{\min} \leq u_1(i|k) \leq T_{\max}, \\ & F_{b,\,\min} \leq u_2(i|k) \leq F_{b,\,\max}, \\ & w_{\min} \leq \omega\big(x_2(i|k), u_3(i|k)\big) \leq w_{\max}, \\ & w_{\min} \leq \omega\big(x_2(i+1|k), u_3(i|k)\big) \leq w_{\max}, \\ & i = 0, \dots, N-1 \Big\}. \end{aligned} \tag{12}$$

The last condition in $\mathcal{C}$, relating $x_2(i+1|k)$ to $u_3(i|k)$, ensures that the gear at time step $k+i$ maintains the engine speed within its bounds for all $t \in [(k+i)\Delta t, (k+i+1)\Delta t]$.

The MINLP (10) provides a state feedback controller via solving (10) at each time step k and applying the first element $u^*(0|k)$ of the optimal sequence $\mathbf{u}^*(k)$ to the system. However, the computation required to solve (10) online renders it unsuitable for a real-time implementation, i.e., the computation time is larger than the available real time for the MPC to make a decision. In the following, we introduce an alternative MPC controller that can be executed efficiently online.

B. Learning-Based Nonlinear MPC

Let us define a reduced control action that does not include the gear choice as $u'(k) = [T(k\Delta t), F_b(k\Delta t)]^\top$. We then introduce the following MPC controller, parameterized by a gear-shift schedule $\mathbf{j}(k) = [j(0|k), \ldots, j(N-1|k)]^\top$:

$$J\left(x(k), \mathbf{x}_{\text{ref}}(k), \mathbf{j}(k)\right) = \min_{\substack{\mathbf{x}(k),\\ \mathbf{u}'(k)}} \sum_{i=0}^{N} \beta L_t\left(x(i|k), x_{\text{ref}}(i+k)\right) + \sum_{i=0}^{N-1} L_f\left(x_2(i|k), u_1'(i|k), j(i|k)\right) \tag{13a}$$

$$\text{s.t.} \quad \text{(10b), (10d), (10e)} \tag{13b}$$

$$\left(\mathbf{x}(k), \left[u'^{,\top}(0|k), j(0|k), \ldots, u'^{,\top}(N-1|k), j(N-1|k)\right]^\top\right) \in \mathcal{C} \tag{13c}$$

$$\text{for } i = 0, \ldots, N-1 \quad x(i+1|k) = f\left(x(i|k), \left[u'^{,\top}(i|k), j(i|k)\right]^\top\right) \tag{13d}$$

$$\text{for } i = 0, \ldots, N-2: \quad |j(i+1|k) - j(i|k)| \le 1. \tag{13e}$$

With $\mathbf{j}$ prespecified, no discrete variables are optimized in the problem (13), which can now be solved efficiently using numerical nonlinear solvers. Note that if $\mathbf{j} = \mathbf{u}_3^*$, the optimal gear-shift schedule from (10), then $J(x, \mathbf{x}_{\text{ref}}, \mathbf{j}) = J(x, \mathbf{x}_{\text{ref}})$.

We propose the use of a learned policy that selects and fixes the gears over the prediction horizon based on the optimal solution to the MPC problem from the previous time step. Define the shifted solutions to (13) at time step k, i.e., the optimal control and state trajectories from time step $k-1$ advanced by one time step, as

$$\bar{\mathbf{x}}(k) = \left[x^\top(k), x^{*,\top}(2|k-1), \ldots, x^{*,\top}(N|k-1)\right]^\top \in \mathbb{R}^{2N},$$
$$\bar{\mathbf{u}}'(k) = \left[u'^{,*,\top}(1|k-1), \ldots, u'^{,*,\top}(N-1|k-1), u'^{,*,\top}(N-1|k-1)\right]^\top \in \mathbb{R}^{2N}.$$

Note that the first element of $\bar{\mathbf{x}}(k)$ is replaced with the state $x(k)$, such that in the case of modeling errors the real state is present. Furthermore, define the shifted gear-shift schedule

$$\bar{\mathbf{j}}(k) = \left[j(1|k-1), \ldots, j(N-1|k-1), j(N-1|k-1)\right]^\top. \tag{14}$$

Consider the selection of $\mathbf{j}$ by a policy, parameterized by θ,

$$\mathbf{j} = \pi_\theta\left(\bar{\mathbf{x}}, \bar{\mathbf{u}}', \mathbf{x}_{\text{ref}}, \bar{\mathbf{j}}\right), \tag{15}$$

as a function of the reference trajectory and the shifted solutions from the previous time step.[1] Note that for simplicity the time index (k) is dropped. In Section IV the architecture and training of π_θ are described.

Observe that there are many choices for $\mathbf{j}$ for which (13) has no solution. While we can expect π_θ to almost always provide at least a feasible $\mathbf{j}$ (if not optimal), here we propose a *backup solution* that will be useful for guaranteeing feasibility at deployment. We then prove that this backup solution, while potentially suboptimal, always provides a feasible $\mathbf{j}$. Define the set of feasible gears for a given velocity v as:

$$\Phi(v) = \left\{j \in \{1, \ldots, 6\} \,\middle|\, w_{\min} \le \omega(v, j) \le w_{\max}\right\}, \tag{16}$$

and ϕ as any mapping from v to one of the gears $j \in \Phi(v)$, e.g., $\phi(x_2) = \max_{j' \in \Phi(x_2)} j'$. The backup solution, defined by the function σ, is then

$$\mathbf{j} = \sigma(x_2) = \left[\phi(x_2), \ldots, \phi(x_2)\right]^\top. \tag{17}$$

Furthermore, define a map from gears to velocities that satisfy the engine speed constraints as $\Omega(j) = \{v | j \in \Phi(v)\}$.

Proposition 1: Assume that, for $j \in \{1, \ldots, 6\}$ and for all $x_2 \in \Omega(j)$, there exist u_1 and u_2 such that $T_{\min} \le u_1 \le T_{\max}$, $F_{b,\min} \le u_2 \le F_{b,\max}$, and

$$\frac{u_1 z(j) z_f}{r} - C x_2^2 - u_2 - G = 0. \tag{18}$$

Then, for a state $x(k)$ such that $v_{\min} \le x_2(k) \le v_{\max}$, and a gear-shift sequence $\mathbf{j}(k) = \sigma(x_2(k))$, problem (13) has a solution, i.e., $J(x(k), \mathbf{x}_{\text{ref}}(k), \mathbf{j}(k)) < \infty$.

Proof: See the Appendix. ∎

As x_2^2 is positive and monotonic, verifying condition (18) for a given vehicle involves checking only the endpoints of the range $\Omega(j)$ for $j = 1, \ldots, 6$, yielding 6×2 conditions to be verified. The condition is satisfied for reasonable vehicle parameters, including those used in Section VI. Note that Proposition 1 guarantees instantaneous feasibility of (13). Recursive feasibility follows trivially if the true underlying system is (9). In the case of modeling errors, e.g., when the dynamics (9) are a discrete-time approximation of the continuous-time system (1), a robust MPC formulation would be required for recursive feasibility and constraint satisfaction. This is left for future work.

The proposed control algorithm is given in Algorithm 1. If the gear-shift schedule proposed by π_θ is infeasible, the control input is computed using the backup gear-shift schedule (17), which is feasible by Proposition 1.

Algorithm 1 Control Algorithm at Time Step k

Inputs: $x(k)$, $\mathbf{x}_{\text{ref}}(k)$, $\bar{\mathbf{x}}(k)$, $\bar{\mathbf{u}}'(k)$, and $\bar{\mathbf{j}}(k)$
$\mathbf{j}(k) \leftarrow \pi_\theta\left(\bar{\mathbf{x}}(k), \bar{\mathbf{u}}'(k), \mathbf{x}_{\text{ref}}(k), \bar{\mathbf{j}}(k)\right)$
Solve (13) for $J\left(x(k), \mathbf{x}_{\text{ref}}(k), \mathbf{j}(k)\right)$ and $\mathbf{u}'^{,*}(k), \mathbf{x}^*(k)$
  If $J = \infty$ **then** $\mathbf{j}(k) \leftarrow \sigma\left(x_2(k)\right)$
    Solve (13) for $J\left(x(k), \mathbf{x}_{\text{ref}}(k), \mathbf{j}(k)\right)$ and $\mathbf{u}'^{,*}(k), \mathbf{x}^*(k)$
Apply $\left[u'^{,*,\top}(0|k), j(0|k)\right]^\top$ to the system

IV. Gear-Shift Schedule Policy

For convenience in the following, define $q(i)$ to stack the ith elements from $\bar{\mathbf{x}}$, $\bar{\mathbf{u}}'$, $\mathbf{x}_{\text{ref}}$ and $\bar{\mathbf{j}}$, e.g.,

$$q(0) = \left[x^\top(k), u'^{,*,\top}(1|k-1), x_{\text{ref}}^\top(k), j(1|k-1)\right]^\top$$
$$q(N-1) = \left[x^{*,\top}(N|k-1), u'^{,*,\top}(N-1|k-1), x_{\text{ref}}^\top(k+N-1), j(N-1|k-1)\right]^\top. \tag{19}$$

We propose to model the policy π_θ with an NN, with the parameter θ the model weights. Representing π_θ with a

[1] If α would be time-varying then α would be an additional input argument.

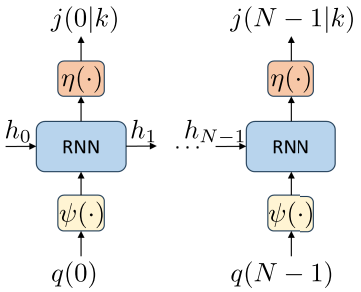


Fig. 1. Recurrent NN, with hidden states h_i, showing how the chain of inputs sequentially generates the gear-shift schedule. The maps ψ and η are input and output transformations.

standard feed-forward NN has the key issue that the action space grows exponentially with the prediction horizon N, while the input space grows linearly. Indeed, for a given N there are 6^N possible gear-shift schedules and $4(N+1)+2N$ inputs (from $\bar{\mathbf{x}}$, $\mathbf{x}_{\text{ref}}$, $\bar{\mathbf{u}}'$, and $\bar{\mathbf{j}}$). An NN capable of representing the input-output mapping as N increases may need to be very large and highly complex. Furthermore, there is an explicit temporal relationship between gear-shifts, which is not structurally enforced in a feed-forward NN. In light of these points, inspired by [4] we propose a sequence-to-sequence recurrent architecture using a recurrent NN (RNN), as shown in Figure 1. The inputs $\bar{\mathbf{x}}, \bar{\mathbf{u}}', \mathbf{x}_{\text{ref}}$ and $\bar{\mathbf{j}}$ are considered as N different inputs in a chain, i.e., the vectors $q(i) \in \mathbb{R}^7$ for $i = 0, \ldots, N-1$. A single RNN is trained with input space $\mathbb{R}^7$, where the output is a single gear position. The output sequence of N gear positions is then generated by sequentially evaluating the RNN on the chain of inputs $q(0), \ldots, q(N-1)$. In this way the recurrent structure results in a constant number of inputs and outputs for the network for any prediction horizons, with only the number of sequential evaluations changing with the horizon. Furthermore, the temporal relationship is structurally enforced, i.e., the gear at time step $i+k$ considers the prior gears and inputs via the hidden state h_i.

More formally the policy is defined as

$$\pi_\theta\big(\bar{\mathbf{x}}, \bar{\mathbf{u}}', \mathbf{x}_{\text{ref}}, \bar{\mathbf{j}}\big) = \Big[\eta\Big(y\Big(\psi\big(q(0)\big), h_0\Big)\Big), \ldots, \eta\Big(y\Big(\psi\big(q(N-1)\big), h_{N-1}\Big)\Big)\Big]^\top, \tag{20}$$

where the input mapping ψ, defined by

$$\psi\big(\ q = \big[x^\top, u'^{,\top}, x_{\text{ref}}^\top, j\big]^\top\big) = \Big[(x - x_{\text{ref}})^\top, \frac{x_2 - v_{\min}}{v_{\max} - v_{\min}}, \frac{x_{2,\text{ref}} - v_{\min}}{v_{\max} - v_{\min}}, u'^{,\top}, \omega(x_2, j), j\Big]^\top, \tag{21}$$

transforms the inputs into a representation that contains the tracking error, the vehicle and reference velocities, and the predicted inputs, including the engine speed. This representation is chosen to give the RNN the most relevant information for selecting $\mathbf{j}$. The model function of the RNN is

$$\delta(i) = \big[\delta_1(i), \ldots, \delta_6(i)\big]^\top = y\Big(\psi\big(q(i)\big), h_i\Big), \tag{22}$$

where $\delta_j(i)$ is the probability of choosing gear j at the ith output in the sequence. The output mapping η selects the gear with the largest probability and applies a clipping: $\eta(\delta(0)) = \lambda_0$, and for $i > 0$

$$\eta\big(\delta(i)\big) = \begin{cases} \lambda_i & \text{if } -1 \le \lambda_i - j(i-1|k) \le 1 \\ j(i-1|k) - 1 & \text{if } \lambda_i - j(i-1|k) < -1 \\ j(i-1|k) + 1 & \text{if } \lambda_i - j(i-1|k) > 1 \end{cases} \tag{23}$$

with $\lambda_i = arg\,max_j \delta_j(i)$. The clipping ensures that $\mathbf{j}$ can be feasible by satisfying constraint (13e).

The policy π_θ can be trained in a supervised manner using a dataset of input-output pairs

$$\mathcal{T} = \big\{\big((\bar{\mathbf{x}}_l, \bar{\mathbf{u}}'_l, \mathbf{x}_{\text{ref},l}, \bar{\mathbf{j}}_l), \mathbf{j}_l\big)\big\}_{l=1}^{N_{\text{data}}}. \tag{24}$$

In Section VI the collection of $\mathcal{T}$ is detailed. With the number of inputs and outputs of the model function y independent of the prediction horizon N, an added benefit of the RNN architecture is that, once trained, the policy can be applied to an MPC controller with larger N by applying longer input sequences. Furthermore, the policy can be trained with data generated by different controllers with different horizons N.

V. Comparison Controllers

In this section we outline three controllers against which the proposed method will be evaluated.

MINLP-Based MPC: This MPC controller solves the MINLP (10) at each time step k, applying $u^*(0|k)$ to the system. This controller provides the baseline performance for all other controllers, but is highly computationally intensive.

Mixed-Integer Quadratic Program-Based (MIQP) MPC: This controller follows the approach from [11], where the MINLP (10) is relaxed such that the remaining optimization problem is an MIQP. In particular, a McCormick relaxation is applied to the bi-linear term in L_{f}, the quadratic term in the dynamics is replaced by a piecewise-linear approximation, and all bi-linear terms in the dynamics, e.g., $u_1 z(u_3)$, are replaced by mixed-integer inequalities (see [11] for details).

Hierarchical MPC: This controller follows the principle of decoupling the optimization of the vehicle speed from the gear-shift schedule. To this end, the simplified dynamics $x(k+1) = \tilde{f}(x(k), F(k))$ are considered, where

$$\tilde{f}(x, F) = \begin{bmatrix} x_1 + \Delta t x_2 \\ x_2 + \frac{\Delta t}{m}\big(F - Cx_2^2 - G\big) \end{bmatrix}. \tag{25}$$

The input F replaces $Tz(j)z_{\text{f}}/r - F_{\text{b}}$, the desired braking force and the applied force from the engine torque combined with the gear. The following NLP is solved:

$$J\big(x(k),\ \mathbf{x}_{\text{ref}}(k)\big) = \min_{\substack{\mathbf{x}(k),\\ \mathbf{F}(k)}} \sum_{i=0}^{N} L_{\text{t}}\big(x(i|k), x_{\text{ref}}(i+k)\big) \tag{26a}$$

$$\text{s.t.} \quad (10b),\ (10d) \tag{26b}$$

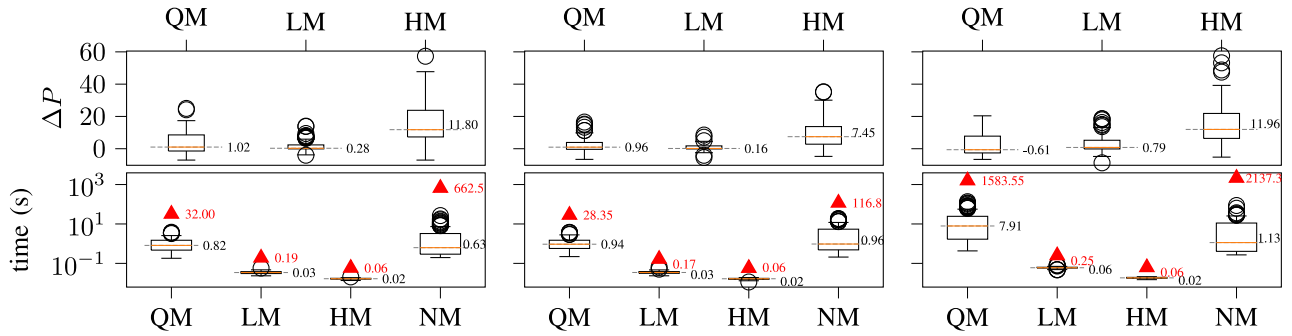
$$\text{for} \quad i = 0, \ldots, N-1:$$

$$x(i+1|k) = \tilde{f}\big(x(i|k), F(i|k)\big) \tag{26c}$$

$$T_{\min}\frac{z(0)z_{\text{f}}}{r} - F_{\text{b,max}} \le F(i|k) \le F_{\max}(k) \tag{26d}$$

$$v_{\min} \le x_2(i|k) \le v_{\max} \quad i = 0, \ldots, N \tag{26e}$$

where the fuel cost cannot be considered as the powertrain dynamics are not modeled. The bound $F_{\max}(k)$ is determined



(a) $N = 15$. (b) $N = 15$ with headwind. (c) $N = 20$.

Fig. 2. Top: distribution of (29) across 100 episodes with median marked. Bottom: distribution of average MPC solve time for each episode across 100 episodes with median marked. The maximal time of all steps is marked with a red triangle.

TABLE I
VARIABLE BOUNDS FOR THE VEHICLE

Symbol	a	v	$F_\mathrm{b} \cdot 10^{-3}$	T	$\omega \cdot 10^{-3}$
Bounds	$[-3, 3]$	$[2.2, 44.4]$	$[0, 9]$	$[15, 300]$	$[0.9, 3]$

at each time step k by considering the gear that provides the most traction for the current velocity

$$F_{\max}(k) = T_{\max} \cdot \max_{j \in \Phi(x_2(k))} \frac{z(j) z_\mathrm{f}}{r}. \tag{27}$$

The gear is then selected as $j(k\Delta t) = \phi(x_2(k))$ and clipped such that (10f) is respected. In our simulations we found $j(k\Delta t) = \phi(x_2(k)) = \max_{j' \in \Phi(x_2(k))} j'$ to perform the best. Finally, $T(k\Delta t)$ and $F_\mathrm{b}(k\Delta t)$ are decided as

$$\begin{aligned} T(k\Delta t) &= \begin{cases} T_{\min} & F^*(0|k) < 0 \\ \frac{F^*(0|k) r}{z(j(k\Delta t)) z_\mathrm{f}} & F^*(0|k) \geq 0 \end{cases}, \\ F_\mathrm{b}(k\Delta t) &= \begin{cases} -F^*(0|k) + \frac{T_{\min} z(j(k\Delta t)) z_\mathrm{f}}{r} & F^*(0|k) < 0 \\ 0 & F^*(0|k) \geq 0 \end{cases}, \end{aligned} \tag{28}$$

with the torque rate constraint (10e) applied with clipping.

VI. SIMULATIONS

In the following, MINLP problems are solved with Knitro [3], MIQP problems are solved with Gurobi [6], and NLP problems are solved with Ipopt [13]. These solvers are each state-of-the-art for the respective type of optimization problem. All coefficients defining the vehicle model can be found in [11], with bounds given in Table I. Source code is available at https://github.com/SamuelMallick/mpcrl-vehicle-gears.

For training the policy π_θ and for evaluation of the controllers, we consider episodic highway-driving scenarios with $\alpha = 0$. Each episode requires a vehicle, initialized with a velocity in the range $[v_{\min} + 5, v_{\max} - 5]$ ms^{-1}, to track a random reference trajectory for 100s. Randomized reference trajectories are constructed as follows. Beginning with velocity $x_{2,\mathrm{ref}}(0) \sim \mathcal{U}(15, 25)$, the acceleration of the reference trajectory changes over five randomly spaced intervals. For the first and last interval the acceleration is zero, with random values in $[-0.6, 0.6]$ ms^{-2} for the other intervals. Additionally, the reference velocity is clipped to the range $[5, 28]$ ms^{-1} (18–100 kmh^{-1}). To train π_θ with supervised learning the dataset $\mathcal{T}$ is generated using the MIQP-based MPC controller. While the solution provided by MIQP is an approximation of the MINLP solution, we found the quality sufficient to train the policy π_θ, and the computation time required to generate the data less. Data is generated from 300 episodes, with $N = 15$, and used to train an RNN with 4 layers of 256 features in the hidden state, followed by a fully connected linear layer.

A. Evaluation

To evaluate the performance of the controllers we compare the performance metric P, defined in (6), over 100 episodes (not present in the training of π_θ). We select $\beta = 0.01$, tuned to balance the relative importance of the fuel consumption and the quadratic tracking error, and $Q = \mathrm{diag}(1, 0.1)$. All MPC controllers use a horizon of $N = 15$, and both the MPC controllers and the underlying simulation use a time step of $\Delta t = 1$s. The backup solution is defined via $\phi(x_2) = \max_{j' \in \Phi(x_2)} j'$. For the first time step of each episode there are no shifted solutions available. Hence, for the proposed approach the policy π_θ is not used for the fist time step, with instead the MIQP-based MPC problem providing $\mathbf{j}$.

Using MINLP-based MPC (denoted NM) as a baseline, define the cost increase introduced by each controller as

$$\Delta P_\mathrm{type} = 100 \cdot \frac{P_\mathrm{type} - P_\mathrm{NM}}{P_\mathrm{NM}}, \tag{29}$$

with type $\in \{\mathrm{QM}, \mathrm{LM}, \mathrm{HM}\}$ representing the MIQP-based MPC, the proposed approach, and the hierarchical MPC, respectively. Figure 2(a) shows a box-and-whiskers plot of ΔP and the solve time required for each controller[2] It can be seen that LM requires significantly less computation time than QM and NM, with QM and NM unable to always find a solution within the MPC sample time Δt. Furthermore, the performance drop is negligible, with the median even improving over QM, likely due to the use of the exact fuel and friction models in the prediction model. In contrast, while HM requires even less computation time than LM, as an even simpler NLP is solved, the performance drop

[2]Knitro experienced occasional numerical issues when solving (10). In these cases, the MINLP solver Bonmin [2] is used as a backup solver.

is significant. For LM the backup solution was used for 0.27% of time steps. For these time steps the average 1-norm difference between the policy and backup gear-shift schedules $|\pi_\theta(\bar{\mathbf{x}}, \bar{\mathbf{u}}', \mathbf{x}_{\text{ref}}, \bar{\mathbf{j}}) - \sigma(x_2)|_1$ was equal to 12. Representative state and input trajectories are shown in the longer online version of this letter [8].

It is well known that MPC controllers have a level of inherent robustness thanks to re-optimization at each time step [10]. To explore the preservation of this robustness in the proposed learning-based MPC controller, a further evaluation is conducted for 100 episodes where a strong disturbance in the form of a time-varying headwind $v_{\text{w}}(t) \in [8, 14]$ ms^{-1} is present. This disturbance is unmodeled for the MPC controllers, but affects the true dynamics of the system. The headwind changes the relative velocity of the vehicle when calculating wind drag, i.e., the drag term in (1) becomes $C(v+v_{\text{w}})^2$. Figure 2(b) shows the cost increase and the solve time over 100 episodes under headwind disturbance. LM retains a performance that is comparable to that of the mixed-integer approaches, with a superior computational burden. The backup solution was used for 2.15% of time steps.

Finally, we explore the scaling of the approaches with the prediction horizon N, and the generalization of the proposed approach to different horizon lengths. An evaluation is conducted for 100 episodes (without headwind) with $N = 20$. The proposed approach LM uses the policy π_θ trained with $N = 15$, i.e., no retraining. Figure 2(c) shows the cost increase and the solve times. Again, LM retains a comparable performance to the mixed-integer controllers with superior computation time, demonstrating how the learned policy can generalize to horizons longer than that on which it was trained. For LM the backup solution was used for 0.43% of time steps.

VII. Conclusion

In this letter we have proposed a novel learning-based MPC controller for fuel efficient autonomous driving. By learning a policy that selects the gear-shift schedule over the MPC prediction horizon, the benefits of speed and gear co-optimization, i.e., fuel efficient tracking, are retained without the computational burden of solving a mixed-integer program. The result is a controller that achieves a performance comparable to approaches that solve mixed-integer programs, and that has a computational burden comparable to sub-optimal approaches that decouple speed and gear optimization. Future work will look at extending the approach to vehicle platoons and addressing model mismatch with robust MPC.

Appendix

Proof of Proposition 1

Proof: We prove Proposition 1 by showing the existence of a specific feasible solution to (13), namely a constant-velocity trajectory. We show that $J(x(k), \mathbf{x}_{\text{ref}}(k), \mathbf{j}(k)) < \infty$ with $\mathbf{j}(k) = \sigma(x_2(k))$ for the solution $\mathbf{x}(k), \mathbf{u}'(k)$ with

$$x(i|k) = \left[x_1(k) + \tau x_2(k)\Delta t, x_2(k)\right]^\top,\ i = 0, \dots, N \quad (30)$$

and constant input $u'(i|k) = [u_1, u_2]^\top$ for $i = 0, \dots, N-1$, with $T_{\min} \le u_1 \le T_{\max}$, $F_{\text{b, min}} \le u_2 \le F_{\text{b, max}}$ such that

$$\frac{u_1 z\Big(\phi\big(x_2(k)\big)\Big) z_{\text{f}}}{r} - C x_2^2(k) - u_2 - G = 0. \quad (31)$$

By the assumption in Proposition 1 this control input exists for any $x_2(k)$ with $v_{\min} \le x_2(k) \le v_{\max}$ and for gear $j = \phi(x_2(k))$. The dynamics constraints (13d) are satisfied by this solution as (30) and (31) satisfy the equality constraint, i.e., for $i = 0, \dots, N-1$ we have

$$f\Big(x(i|k), \left[u_1, u_2, \phi\big(x_2(k)\big)\right]^\top\Big) = x(i+1|k). \quad (32)$$

Furthermore, the constraints (13b) and (13e) are trivially satisfied by this solution with constant velocity $x_2(k)$, engine torque u_1, and gear $\phi(x_2(k))$. Finally, constraint (13c) is satisfied as $T_{\min} \le u_1 \le T_{\max}$, $F_{\text{b, max}} \le u_2 \le F_{\text{b, max}}$, and $w_{\min} \le \omega(x_2(k), \phi(x_2(k))) \le w_{\max}$ by the definition of ϕ. Hence, all constraints are satisfied, with $\mathbf{x}(k), \mathbf{u}'(k)$ one of potentially many feasible solutions. Indeed, this solution could be used as a feasible initial guess for a numerical solver. ■

References

[1] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[2] P. Bonami and J. Lee, "Bonmin user's manual," *Numer. Math.*, vol. 4, no. 1, pp. 1–32, 2007.

[3] R. H. Byrd, J. Nocedal, and R. A. Waltz, "Knitro: An integrated package for nonlinear optimization," in *Large-Scale Nonlinear Optimization. Nonconvex Optimization and Its Applications*. Boston, MA, USA: Springer, 2006, pp. 35–59.

[4] C. F. Oliveira da Silva, A. Dabiri, and B. De Schutter, "Integrating reinforcement learning and model predictive control with applications to microgrids," 2024, *arXiv:2409.11267*.

[5] A. Ganesan, S. Gros, and N. Murgovski, "Numerical strategies for mixed-integer optimization of power-split and gear selection in hybrid electric vehicles,"" *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 3194–3210, Mar. 2023.

[6] *Gurobi Optimizer Reference Manual*, Gurobi Optim., LLC, Beaverton, OR, USA, 2024.

[7] G. Li and D. Gorges, "Ecological adaptive cruise control for vehicles with step-gear transmission based on reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4895–4905, Nov. 2020.

[8] S. Mallick, G. Battocletti, Q. Dong, A. Dabiri, and B. De Schutter, "Learning-based MPC for fuel efficient control of autonomous vehicles with discrete gear selection," 2025, *arXiv:2503.11359*.

[9] S. Mallick, A. Dabiri, and B. De Schutter, "A comparison benchmark for distributed hybrid MPC control methods: Distributed vehicle platooning," 2024, *arXiv:2401.09878*.

[10] J. B. Rawlings, D. Q. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Madison, WI, USA: Nob Hill Publ., 2017.

[11] Y. Shao and Z. Sun, "Vehicle speed and gear position co-optimization for energy-efficient connected and autonomous vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 4, pp. 1721–1732, Jul. 2021.

[12] V. Turri, B. Besselink, and K. H. Johansson, "Gear management for fuel-efficient heavy-duty vehicle platooning," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, 2016, pp. 1687–1694.

[13] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, pp. 25–57, Mar. 2006.

[14] Y. Yin, X. Huang, S. Zhan, X. Zhang, and F. Wang, "Hierarchical model predictive control strategy based on Q-learning algorithm for hybrid electric vehicle platoon," *Proc. Inst. Mech. Eng. D, J. Automobile Eng.*, vol. 238, nos. 2–3, pp. 385–402, 2022.

[15] Y. Zheng, S. E. Li, K. Li, F. Borrelli, and J. K. Hedrick, "Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 3, pp. 899–910, May 2017.