Model Predictive Eco-Driving Control for Heavy-Duty Trucks Using Branch and Bound Optimization

Bart Wingelaar\(^\text{D}^\text{C}\), Gustavo R. Gonçalves da Silva\(^\text{D}^\text{C}\), and Mircea Lazar\(^\text{D}^\text{C}\), Member, IEEE

Abstract—Eco-driving (ED) can be used for fuel savings in existing vehicles, requiring only a few hardware modifications. For this technology to be successful in a dynamic environment, ED requires an online real-time implementable policy. In this work, a dedicated Branch and Bound (BnB) model predictive control (MPC) algorithm is proposed to solve the optimization part of an ED optimal control problem. The developed MPC solution for ED is based on a prediction model that includes velocity dynamics as a function of distance and a finite number of driving modes and gear positions. The MPC optimization problem minimizes a cost function with two terms: one penalizing the fuel consumption and one penalizing the trip duration. We exploit contextual elements and use a warm-started solution to make the BnB solver run in real-time. The results are evaluated in numerical simulations on two routes in Israel and France and the long haul cycle of the Vehicle Energy consumption Calculation Tool (VECTO). In comparison with a human driver and a Pontryagin's Minimum Principle (PMP) solution, 25.8% and 12.9% fuel savings, respectively, are achieved on average.

Index Terms—Optimal control, control systems, optimization, velocity control, automotive control, nonlinear control systems, predictive control.

I. INTRODUCTION

HEAVY-DUTY trucks (HDTs), also known as Heavy-duty Vehicles (HDVs), are responsible for 5% of the total emissions and a quarter of the road transport emissions in the EU [1]. Intending to reduce these emissions for HDVs, the EU has set goals for a 15% HDV emission reduction in 2025 and a 30% reduction in 2030 [2]. These emission reductions can be obtained as a consequence of reduced fuel consumption, which in turn can be realized by applying eco-driving (ED). Eco-driving is a driving strategy and its goal is to minimize fuel consumption by determining an optimal velocity profile, which can be defined as an optimal control problem (OCP) [3]. Using ED is cost-effective [4] and does not require substantial vehicle hardware modifications [5]. This makes ED a promising fuel-saving method for existing and future HDTs with internal combustion engines (ICEs).

Since HDTs have discrete gear transmissions, the velocity profile optimization is generally a complex mixed-integer nonlinear programming problem. Several approaches are available to solve the ED optimization problem, including analytical solutions to simplified problems [6], [7], numerical application of PMP [3], [8], [9], [10], [11], dynamic programming (DP) [12] and some bi-level optimization methods [13], [14]. The integrality constraint of the gear position can also be relaxed, reducing the problem to a nonlinear programming problem as done in [15] and [16]. This ED optimization can also run in a model predictive control (MPC) fashion (e.g., [17]). In this context, MPC can be particularly useful in cases where spatial and time-varying elements are included in the optimization problem, such as look-ahead information [18] and vehicle-to-everything (V2X) communication [19]. However, online MPC inherently requires real-time optimization, which can be challenging [20].

In recent studies [3], [8], [9], [16], this computational burden was reduced by optimization over driving modes. This novel method converts the difficult-to-solve mixed-integer nonlinear programming problem with continuous decision variables (e.g., engine and brake torques) and discrete decision variables (e.g., gears), into an integer programming problem with only discrete decision variables (driving modes). Thus, this formulation converts the optimization problem into a combinatorial optimization problem. Studies based on this approach, [3], [8], [9], [16], apply a numerical PMP solution to achieve real-time implementation. Although a PMP algorithm is computationally efficient, its use can lead to difficulties with constraints handling and co-state initialization.

Therefore, to achieve a MPC solution for ED that is implementable in real-time, it is more of interest to use solvers that are specifically aimed at solving combinatorial problems, such as Branch and Bound (BnB). However, using a general-purpose BnB re-introduces the high computational complexity issue, compromising the real-time constraint. Hence, in this work we investigate and develop a dedicated BnB MPC solver for the driving modes eco-driving problem including slope and V2X preview information, such that real-time implementability constraints are met. In relation with previous works [3], [16], due to the inclusion of slope preview, we extend the velocity profile generating capabilities with two extra driving modes: one to accelerate the HDT and one for prolonged negative road gradients. Secondly, we introduce three elements to make the solver real-time implementable: 1) application of context-based (such as cost function,
dynamics, constraints, and a look-ahead horizon) node elimination—this element is not present in general-purpose BnB solvers such as BONMIN [21]; 2) the BnB solver is warm-started with a PMP solution [3], providing a locally optimal solution; 3) in contrast to general BnB approaches, the lower bound is not formulated with a relaxation of the integrality constraint, instead a custom heuristic based on the same context information as 1) is used.

Furthermore, we include a realistic model for gearbox transmission efficiency to improve fuel consumption assessment. Finally, we evaluate the fuel savings in numerical simulations on three routes: one in Israel, one in France, and the VECTO long haul cycle, and we show that the BnB solver outperforms a human–driver model and a PMP–MPC-based solution.

The remainder of this paper is organized as follows: In section II we present the longitudinal HDT dynamics. Then in Section III, the ED OCP is formulated for the HDT with a positive velocity and considering the inertia and a continuous service brake. Moreover, the modeling fundamentals are coherent with the modeling as used in [6], and is approximated by a polynomial which is fitted on the data from [25]:

$$T_{loss} = c_1(i)\omega_e + c_2(i)T + c_3(i).$$

In (5), $c_1$–$c_3$ are gear ratio $i$–dependent constants. The used torque loss maps are illustrated for gears 1, 8, and 12 in Fig. 1. Note that gear 12 is in a direct drive configuration.

3) Engine Speed: Engine speed $\omega_e$ [RPM] is a function of the gear position $y$ [-] and velocity. In neutral gear ($y = 0$), the ICE is disengaged from the drivetrain and idles at a low engine speed. $\omega_e$ is defined as:

$$\omega_e = \frac{\omega_{idle}(y)\rho_i}{\pi r_w v}$$

where $r_w$ is the radius of the rear wheels, $i_r$ the final drive ratio and $\omega_{idle}$ the idling speed.

B. Driving Modes

Next, we assume that the velocity profile can be followed by switching between a finite number of subsystems, called driving modes. The resulting switched hybrid system is defined as:

$$\frac{dv}{ds} = f(M(s), v(s)),$$

with $f(\cdot)$ the system dynamics of driving mode $M \in \tilde{M}$ with $|M| = 6$. The system dynamics is now only dependent on the driving mode as control variable.

1) Cruising Mode $M^{cr}$: In cruising mode, the truck drives with a constant velocity and $y > 0$. The ICE delivers an engine output torque (as measured by a dynamometer) $T_e$, to overcome $F_{res} > 0$ and transmission losses. The dynamics in this mode are described by:

$$\frac{dv}{ds} = \frac{1}{mv} \left( \frac{\eta_f \eta_t i_r(y)}{r_w} T_e - F_{res} \right) = 0.$$
2) Eco–Roll Mode $M^{ec}$: In eco–roll mode, the ICE is disengaged from the drivetrain with the gear position in neutral ($y = 0$) and it rotates at an idling speed $\omega_{idle}$. The HDT dynamics are now fully dependent on $F_{res}$ and the equivalent rotational inertia of the drivetrain $J_{dt}$. The dynamics in this mode are described by:

$$\frac{dv}{ds} = -\frac{r_w^2}{(mr_w^2 + J_{dt})v} F_{res}. \quad (9)$$

3) Coasting Mode $M^{co}$: In coasting mode, the resistance force, equivalent powertrain rotational inertia, and internal engine friction $T_{in,fr}$ determine the HDT dynamics. The dynamics are described by:

$$\frac{dv}{ds} = -\frac{r_w^2}{(mr_w^2 + J_{fr}(y))v} \left( \frac{\eta_f \eta_l i_j i_j(y)}{r_w} T_{in,fr} + F_{res} \right). \quad (10)$$

4) Engine Brake Mode $M^{eb}$: Besides the internal engine friction, an ICE can also deliver a braking force with an engine brake (a type of continuous service brake). In this mode, we assume that maximum engine brake torque $T_{eb,m}$ is applied to decelerate the vehicle. The dynamics are described by:

$$\frac{dv}{ds} = -\frac{1}{(mr_w^2 + J_{fr}(y))v} \left( \frac{\eta_f \eta_l i_j i_j(y)}{r_w} T_{in,fr} + T_{eb,m} + F_{res} \right), \quad (11)$$

where $T_{eb,m}$ is the maximum braking torque of the continuous service brake. The modeling of this brake is based on an approximation of the data in [26].

5) Acceleration Mode $M^{ac}$: In the acceleration mode, the engine torque is used to accelerate the truck instead of just cruising. The dynamics are described by:

$$\frac{dv}{ds} = \frac{r_w^2}{(mr_w^2 + J_{fr}(y))v} \left( \frac{\eta_f \eta_l i_j i_j(y)}{r_w} T_{e,ef} - F_{res} \right). \quad (12)$$

where $T_{e,ef}$ is a fixed engine output torque profile based on the optimal brake specific fuel consumption (BSFC) line [24]. For a given engine output power $P_e = \frac{\pi}{3} T_e \omega_e$, the corresponding point, $(T_{e,ef}, \omega_e)$, on the optimal BSFC line minimizes the following:

$$T_{e,ef} \omega_e = \min_{T_e} \frac{m_f(T_e, \omega_e)}{P_e} \left( \frac{\eta_f \eta_l i_j i_j(y)}{r_w} T_{e,ef} - F_{res} \right). \quad (13)$$

This line is illustrated in Fig. 2.

6) Downhill Mode $M^{dh}$: In downhill mode, the HDT maintains a constant speed in negative gradient scenarios. Unlike cruising mode, $F_{res} < 0$ due to the negative gradient. Moreover, this mode is only enabled when any gear position in coasting mode cannot provide sufficient braking torque to maintain a constant speed or decelerate the HDT. Therefore, the engine brake provides just enough additional braking torque to maintain a constant speed. The engine brake torque is computed such that:

$$\frac{dv}{ds} = \frac{1}{mv} \left( \frac{\eta_f \eta_l i_j i_j(y)}{r_w} (T_{in,fr} + T_{eb}) + F_{res} \right) = 0. \quad (14)$$

C. Fuel Consumption

Each driving mode has a specific fuel consumption $\frac{dm_f}{ds}$, which is presented in (15):

$$\frac{dm_f}{ds} = \begin{cases} \frac{1}{v} m_f(T_e, \omega_e) & M \in \{M^{cr}, M^{ac} \} \\ \frac{1}{v} m_f(idle) & M \in \{M^{ec} \} \\ 0 & M \in \{M^{co}, M^{eb}, M^{dh} \}, \end{cases} \quad (15)$$

where $m_f(idle)$ is a small amount of fuel consumed by the idling ICE. The fuel consumption $m_f$ is a polynomial fit which maps $T_e$ and $\omega_e$ to the diesel fuel map, given by:

$$m_f = c_4 + c_5 \omega_e + c_6 T_e + c_7 \omega_e^2 + c_8 \omega_e T_e + c_9 T_e^2, \quad (16)$$

where $c_4$-$c_9$ are constants. In Fig. 2, $m_f$ is illustrated, as well as the optimal BSFC line, maximum engine output torque constraint $T_{e,max}$, internal engine friction $T_{in,fr}$, and engine speed constraints.

III. ECO–DRIVING OPTIMAL CONTROL PROBLEM

The ED MPC problem formulation can be described by the following OCP:

$$\min_{M \in M} J \quad (17)$$

$$J = \int_{s_0}^{s_f} \left( w_f \frac{dm_f(M, v)}{ds} + w_i \frac{1}{v} \right) ds + \beta (v(s_f) - v_f)^2, \quad (18)$$

where $J \ [\cdot]$ is the cost to minimize, $s_0$ the current position, $s_f$ the end position, $w_f$ the weight on fuel consumption, $w_i$ the weight on trip duration, $\beta$ a terminal penalty, and $v_f$ the end velocity reference. For appropriate weight scaling, we assume $w_f + w_i = 1$. Later on, we will evaluate these weights with tuning parameter $\phi \ [g/s]$:

$$\phi = \frac{w_i}{w_f}. \quad (19)$$

Next, we discretize (18) in two steps. First, the distance domain $s$ is discretized into $N$ equidistant samples with length $\delta s = \frac{s_f - s_0}{N}$ on which the driving mode–gear combination is
constant. Secondly, the state dynamics \( f(\cdot) \) are discretized with the forward Euler method. This results in the following discrete OCP at time \( k \):

\[
J = \min_{M_{i,k} \in M} \sum_{i=0}^{N-1} L(\tilde{v}, M_{i,k}) \delta s + \beta (v_{N|k} - v_{f,N})^2 \tag{20a}
\]

\[
\tilde{v} = \frac{1}{2} (u_{i|k} + v_{i+1|k}) \tag{20b}
\]

\[
L(\tilde{v}, M_{i,k}) = w_f \frac{df(M_{i,k}, \tilde{v})}{d\tilde{v}} + w_i \frac{1}{\tilde{v}} \tag{20c}
\]

subject to:

\[
v_{0|k} = v_0 \tag{20d}
\]

\[
v_{i+1|k} = v_{i|k} + f(M_{i,k}, v_{i|k}) \delta s \tag{20e}
\]

\[
v_{\min}(i) \leq v_{i|k} \leq v_{\max}(i) \tag{20f}
\]

\[
\omega_{e,\min}(T_e) \leq \omega_e \leq \omega_{e,\max}(T_e) \tag{20g}
\]

\[
T_e \leq T_{e,max}(\omega_e) \tag{20h}
\]

\[
T_{eb} \leq T_{eb,max}(\omega_e) \tag{20i}
\]

where \( v_0 \) is the current velocity of the HDT at time \( k \) and (20e) is the discretized dynamics. (20f)–(20i) represent, respectively, constraints on minimum and maximum vehicle speed, engine speed, engine output torque, and engine brake torque.

In contrast to conventional ED OCPs, which optimize over discrete and continuous (input) variables \((T, T_{eb} \text{ in (20)})\), the continuous inputs are now fixed by the driving mode dynamics. Consequently, this new formulation is converted into a combinatorial optimization problem, which is typically solved using BnB methods.

### IV. ECO–Driving Branch and Bound MPC

In this section, we introduce the general idea of using MPC with Branch and Bound and the main elements that compose the solution. In Section V we then formalize the local elements that make it a dedicated solution.

#### A. The MPC Control Part

Next, we formulate a feedback control scheme with an MPC controller in the loop, where the feedback is provided by the automated HDT or human driver. For each iteration, the MPC optimization problem is solved with a BnB algorithm. The optimization is warm–started with an initial PMP solution. A schematic overview is provided in Fig. 3.

In the algorithm, we assume that live route data is provided to the MPC control loop in the form of \( \alpha_r, v_{\min,r}, v_{\max,r} \). Although temporal events influence this data, for now it is assumed to be constant. Inside the control loop, the incoming data is first pre–processed to extract the relevant data \((\alpha, v_{\min}, v_{\max})\), which is then fed to the PMP and BnB solvers. Finally, the first optimal driving mode and gear is provided to the automated HDT or human driver. We summarize the implementation of the ED MPC part in Algorithm 1. Notice that the algorithm already includes some context–based steps.

**Algorithm 1 ED MPC Algorithm**

**Inputs:** \( v_{0|k}, v_{\min,r}, v_{\max,r}, \alpha, \phi. \)

1. \([X_{PMP}, UB_0] = \text{WarmStart}(v_{0|k}, v_{\min}, v_{\max}, \alpha, \phi) \) // run PMP algorithm and return upper bound and set of nodes \( X_{PMP} \) corresponding to local optimal PMP mode sequence

2. \([v_{\min}, v_{\max}] = \text{NodeElim}(v_{\min,r}, v_{\max,r}) \) // update for feasible areas

3. Determine \( h \) // determine heuristic term in LB on a regular sized grid for all stage–velocity combinations

4. \([X_{BnB}] = \text{EDBnB}(X_{PMP}, UB_0, v_{0|k}, v_{\min}, v_{\max}, \alpha, \phi, h) \) // run ED BnB and return set of visited nodes

5. \( v^* = \text{OptSeq}(X_{BnB}) \) // determine optimal Node path from visited nodes

6. \([M^*, y^*] = \text{DetOptPath}(v^*) \) // deduce optimal mode–gear sequence

7. Apply \( M[0], y[0] \) // apply first input of optimal sequence

#### B. The BnB Algorithm

Branch and Bound algorithms are generally considered computationally intensive [27], but can converge to a global optimum [28]. BnB consists of several fundamental elements: branching, bounding, a search strategy and elimination. The branching term refers to the exploration of nodes in a search tree and the search strategy is the order in which nodes are branched (explored or visited). The branching operation partitions the problem into smaller sub–problems (nodes) [29]. The search strategy can be categorized in informed search (e.g., best–first search and heuristics) or uninformed search (e.g., depth–first search, breadth–first search) [30].

The bounding term refers to exploring the lower and upper bounds of the nodes. Upper bounds (of which only the smallest is of interest in a minimization problem) indicate a maximum of the domain on which the optimal solution of \( J \) can be found and the lower bound is a minimum in this domain. In a typical optimization problem, each node has a lower bound, and the best solution thus far is represented by an upper bound. The inclusion of these bounds makes BnB inherently an informed search strategy by itself. As the parent node is iteratively branched into child nodes, the lower bounds and upper bounds are evaluated (bounding) for each child node. All child nodes with a lower bound lower than the upper bound have the potential to yield a better solution (a better upper bound). However, if the lower bound of a (child) node is higher than the upper bound, then branching this node further has no
potential to yield a better upper bound. Thus the node is not further explored (elimination or pruning).

In the ED context, the BnB elements have the following meaning. A node corresponds to the HDT velocity at a certain distance stage (or depth level) after having applied a driving mode–gear combination for a certain distance (a multiple of $\delta s$). The stage costs of reaching a node represent the sum of fuel consumption and time duration. Nodes are eliminated when constraints (for instance maximum speed) are exceeded or when the final stage is reached with a lower total cost.

A better BnB performance can be achieved by initializing an initial upper bound that is close to the optimal solution. This initial upper bound can be provided by a warm-started feasible solution, such as PMP. Having a feasible solution also means that the BnB algorithm can be terminated at any time.

Likewise, better lower bounds can also yield better BnB performance and even speed–up computations. These lower bounds are usually composed by a stage cost at stage $j$ and some heuristic term for the cost to go from stage $j$ to the final stage $N$. Its main implication is that it limits the generation of child nodes. In Section V-C we formalize the computation of this bound.

In Algorithm 2, the general idea of the BnB optimization implementation is presented.

Algorithm 2 ED Branch and Bound Algorithm

Inputs: $X_{PM}$, $UB_0$, $v_{0k}$, $v_{min}$, $v_{max}$, $h$

1. Set upper bound $UB = UB_0$
2. Set $X_{BnB} = X_{PM}$
3. Set $B = v_{0k}$ and $k = 0$
4. Explore nodes until queue is empty or until time limit is exceeded:

   while $k < N$ AND time < time_limit do
      \[ C = \{ \text{ienen in } B \} = \text{singleLevelBBSSearch}(C, node, h, k, N, UB) \]
      \[ C = \text{ContextNodeElim2}(C, v_{min}, v_{max}, \varepsilon) \]
      \[ B = C \]
      \[ k = k + 1 \]
   end

Output: $X_{BnB}$

Remark 1: In this paper, we develop a dedicated context–based BnB solver that is based on the general BnB principles and search strategies. Compared to dynamic programming, which explores all sub–problems and determines the optimal solution using backtracking. The used BnB solver uses lower and upper bounds to only explore (solve) the sub–problems that contain the optimal solution. Spatial BnB on the other hand, is more focused on (convex) relaxation and convergence. This makes spatial BnB a less suitable candidate for context–based strategies. Observe that other existing BnB solvers fall into these categories, i.e., either general purpose BnB (BONMIN, which uses NLP and branch–and–cut strategies) or spatial BnB (COUENNE) [21, 31]. Because such general purpose BnB solvers do not exploit context–based elimination and heuristics, they are typically much slower (e.g., BONMIN is a factor 100 slower for a simpler eco–driving problem). As such, it is not useful to implement them in the complex scenarios presented in the simulation section.

V. IMPLEMENTATION OF THE ED BnB MPC ALGORITHM

In this section we describe the elements in MPC and BnB pseudo–codes that make this a dedicated context–based solver. First, we describe node elimination methods based on exploiting the nature of the problem (this is used in Step 2 of MPC and Step 4.6 of BnB). We then introduce the idea of warm–starting the BnB solver with a fast PMP solver and further, we develop a heuristic for the lower–bound computation. Finally, we present the two search strategies used with their respective modifications and a sorting algorithm for the lower–bound.

A. Warm Start With PMP

A BnB algorithm can be warm–started with a feasible initial solution: a feasible sequence of driving modes and the cost thereof as initial upper bound $UB_0 = J_{PM}$. In this work, the fast PMP algorithm as developed in [3] provides an initial solution in case a feasible solution is found ($v_{min} \leq v_{PM} \leq v_{max}$). If such a solution is found, then the BnB algorithm can be stopped at any time and the final result is feasible.

However, the PMP algorithm is not guaranteed to yield a feasible solution due to constraint handling difficulties. If no feasible PMP solution is found, $UB_0 = \infty$. To minimize constraint handling issues, the implementation of the used PMP algorithm is described as follows. First, $v_{max}$ is subdivided into sections where $v_{max}$ is constant. Then the final stage and velocity of each of these sections form a velocity event. Then, the velocity events are solved sequentially with the PMP algorithm. Finally, a terminal penalty is added to the cumulative cost of the individual simulations to formulate the initial upper bound ($J_{PM} = J_{PM} + \beta(v_{PM}, v_{f.N})^2$), since the PMP solution solves a two–point boundary value problem which has no terminal penalty.

B. Update Maximum and Minimum Velocities

With the BnB search tree exploring the distance–velocity domain, it is of interest to exclude infeasible node regions which leads to a faster solver. To be more specific, real–world $v_{min}$ and $v_{max}$ are discontinuous and create such regions. The HDT dynamics are limited by the maximum engine and deceleration power, which thereby can violate $v_{min}$ and $v_{max}$ constraints if such discontinuities occur. The predictive element takes care of this problem and ensures finding a
feasible sequence of driving modes. However, regions of nodes whose child nodes violate these constraints are still explored.

To overcome this problem, \( v_{\text{min}} \) and \( v_{\text{max}} \) were updated such that their derivatives match the maximum acceleration and minimum deceleration of the driving modes. This means that at the beginning of an acceleration event (e.g., after a red light), \( v_{\text{min}} \) is updated forwards using the acceleration mode. For a deceleration (e.g., when reaching a red light), then \( v_{\text{max}} \) is updated backwards using the engine brake mode.

To update \( v_{\text{min}} \), \( v_{\text{max}} \) efficiently, we assume a maximum ideal acceleration which is defined as applying maximum engine power \( P_{e,\text{max}} \) to the HDT, defined as:

\[
P_{e,\text{max}} = \max \frac{\pi}{30} T_e \omega_e, \tag{21}
\]

subjected to constraints (20g) and (20h). Furthermore, the transmission efficiency of the gearbox and equivalent rotational inertia is neglected. The dynamics are described by:

\[
\frac{dv}{ds}_{\text{max}} = \eta_l P_{e,\text{max}} - F_{\text{res}} v. \tag{22}
\]

Similarly, the maximum braking power \( P_{b,\text{max}} \) is defined as:

\[
P_{b,\text{max}} = \max \frac{\pi \omega_e}{30} (T_{\text{in},fr} + T_{\text{eb,m}}). \tag{23}
\]

Similarly to (22), the minimum ideal deceleration is:

\[
\frac{dv}{ds}_{\text{min}} = -(\eta_l P_{b,\text{max}} + F_{\text{res}} v). \tag{24}
\]

C. Lower Bound and Admissible Heuristic

A lower bound is a minimum on the domain on which the optimal solution of \( J \) can be found. There are usually no formal closed–form definition of the lower bounds (as they would imply knowledge of the optimal solution), so they are usually composed by a sum of stage costs and some heuristic term. We formalize this definition next. Let us first define the total cost \( g \) to reach a node at stage \( j \) form stage 0:

\[
g = \sum_{k=0}^{j-1} L_k \tag{25}
\]

where \( L_k \) is the stage cost given in (20). Next, the lower bound \( LB \) is formulated as:

\[
LB = g + h, \tag{26}
\]

where \( h \) is a heuristic term that is an estimate on the cost from node \( j \) to the final stage \( N \). Typically, this heuristic term is formulated by relaxation of the integrality constraint [29]—which turns (20) into a nonlinear programming (NLP) problem—, but it can also be simply 0. Furthermore, \( h \) must be admissible to retrieve the global optimal solution of \( J \).

In [16], a single case of such an NLP was solved, but this process is not real–time implementable if applied to individual nodes. Moreover, if the resulting NLP solution yields a local optimum, the heuristic is not necessarily admissible and the globally optimal solution is no longer guaranteed to be retrieved. To find a real–time implementable and admissible heuristic, we again use context–specific elements by applying an energy and time–based heuristic similar to [32] and [33]. The added novelty is that for each lower bound, a 2-parameter optimization problem is solved.

Consider node \( i \) at stage \( j \) and \( \bar{v}_h \) as the average velocity from stage \( j \) to stage \( N \). The minimum work done by the air drag can then be described by:

\[
W_{\text{air}}(\bar{v}_h) = (N - j) \frac{1}{2} \rho_a c_d A_f \bar{v}_h^2 \tag{27}
\]

This minimum work does not require the velocity profile to be known. The work of the other resistance forces \( W_d \) from stage \( j \) to stage \( N \) is described by:

\[
W_d = \sum_{k=j}^{N-1} mg(c_r \cos(\alpha_k) + \sin(\alpha_k)), \tag{28}
\]

which can be determined when slope preview is given. Next, we consider the kinetic energy (excluding rotating parts) of the HDT at the node \( i \) with velocity \( v_j \) as:

\[
E_{\text{kin},i} = \frac{1}{2} m v_j^2, \tag{29}
\]

By determining the energy difference \( \Delta E_{\text{kin}} \) for going from \( j \) to \( N \), the energy consumption can be estimated as:

\[
\Delta E_{\text{kin}} = E_{\text{kin},N} - E_{\text{kin},j} + W_d + W_{\text{air}}(\bar{v}_h). \tag{30}
\]

If \( \Delta E_{\text{kin}} > 0 \), then energy is added to the HDT by means of burning fuel. If not, energy is subtracted due to braking forces. With both the energy consumption and time duration defined, we now formulate the total cost–to–go \( J_{i,N} \) from node \( i \) to the final stage:

\[
J_{i,N} = \frac{w_f \max(0, \Delta E_{\text{kin}})}{Q \eta_{\text{opt}}} + \frac{w_f \delta (N - j)}{\bar{v}_h} + \beta (v_N - v_j)^2, \tag{31}
\]

where \( Q \) [J/g] is the specific energy of diesel fuel and \( \eta_{\text{opt}} \) an estimated maximum combined efficiency of all conversion efficiencies from fuel to work. Notice that the actual value of \( J_{i,N} \) is not known beforehand, as it depends on the two free variables \( \bar{v}_h \) and \( v_N \). However, for each node, we can solve a small optimization problem with these two variables (all other variables are fixed) in order to provide a minimal admissible value, which is defined as the heuristic \( h \):

\[
h(j, v_i) = \min_{\bar{v}_h, v_N} J_{i,N}(\bar{v}_h, v_N, j, v_i). \tag{32}
\]

This simple two–variable optimization is solved easily in a (partially) analytic way and is provided to the BnB solver in a 2D lookup table (LUT) with dimensions: node stage and node velocity on a regular–sized grid. For each point in the LUT, (32) is calculated and subsequently \( h \) can be sampled from the LUT during the BnB optimization.

Moreover, both \( \bar{v}_h \), \( v_N \) are constrained by \( v_{\text{max}} \) and \( v_{\text{min}} \). The \( \bar{v}_h \)–domain can be further reduced since acceleration to \( v_{\text{max}} \) or deceleration to \( v_{\text{min}} \) is not instantaneous. Therefore, we update the \( v_{\text{max}} \) and \( v_{\text{min}} \) profiles in (32), by constraining their derivatives with maximum acceleration and deceleration, as described in V–B. This can then lead to larger values of \( h \).
D. Search Strategy

Several options exist for the search strategy. For instance, breadth-first search explores all the nodes of a depth level (distance stage) first, before advancing to a deeper level. Consequently, the upper bound is only updated at the final depth level of the search tree, resulting in many visited nodes which is memory intensive. On the other hand, depth-first search updates the upper bound in the earliest search iterations. However, depth-first search can explore subtrees that do not hold an optimal solution. In this regard, informed search strategies could be a better alternative, although informed search strategies could still end up in an exhaustive search with a large number of visited nodes (e.g., when many nodes have a similar lower bound).

Algorithm 3 Single Level Bounded Breadth-First Search

Inputs: \( C, node, h, k, N, UB \)
1. Branch \( node \) in \( ChildNodes \) // applying driving mode–gear combinations
2. for ( \( childNode \) in \( ChildNodes \) ) {
   \[ LB = \text{DetermineLB}(childNode, h, k) \] //determine \( g \) and \( LB \) using (29) and (30).
   2.1 if \( LB < UB \) AND \( childNode \) is feasible then
      Add \( childNode \) to \( C \)
      if \( k == N - 1 \) then
         \( UB = LB \)
      end
   end
}

Output: \( C, UB \)

Algorithm 4 Greedy Probing Algorithm

Inputs: \( UB, D, k, N \)
1. while \( k < N \) do
   1.1 branch \( D \) into \( chilNodes \)
   1.2 \( \mathcal{E} \) is \( childNode \) with lowest \( LB \)
   if \( \mathcal{E} == \emptyset \) or \( LB \) of \( \mathcal{E} > UB \) then
      1.3 return // no \( UB \) update
   else
      1.4 \( D = \mathcal{E} \)
   end
   1.5 \( k = k + 1 \)
end
2. \( UB = \min(UB, LB \ of \ \mathcal{E}) \)

Output: \( UB \)

It is of interest to utilize the strengths of both informed and uninform ed search strategies. To this extent, we formulate the search strategy for the ED BnB algorithm as follows. We apply a custom breadth–first search strategy (Algorithm 3) which, after each branched depth level, probes the best child node (node with the lowest lower bound) with a greedy depth policy (Algorithm 4). This combination has shown to perform well with context–based node elimination (Section V-E).

The late upper bound update that inherently comes with pure breadth-first search means that few nodes are eliminated based on their lower bounds. This problem is magnified in case of a too low \( h \) (see (26)). To overcome this, an additional depth probing strategy in combination with a greedy policy was used. A probing strategy keeps a certain branching policy constant and explores the consequences thereof [34]. Only the best node (node with lowest \( LB \)) is depth probed at each stage after breadth-first branching. This probing is then done with a greedy policy, where only the child node with the lowest \( LB \) is branched further. This way, the upper bound can be efficiently improved at the earliest stages yielding less node exploration. Finally, it is important to mention that the heuristic term in the lower bound plays an important role in the performance of these search strategies.

E. ED Context-Based Node Elimination

Finally, the BnB problem formulation tends to have an exponential growth in the amount of explored node regardless of the constraints and tight lower bounds. Given the nature of the ED problem, this exponential growth can be avoided. For example, consider two nodes at the same distance stage (or tree depth level) with an almost identical velocity. The cost of reaching these nodes from the root node can significantly deviate. However, the minimum cost of reaching the final distance stage from those two nodes is approximately the same. Therefore it makes no sense to branch both nodes into subtrees as it has negligible benefit. Instead, only the node with the lowest \( g \) (see (25)) can be branched.

We applied this procedure as follows. For a given stage, the velocity range \( v_{\text{max}} - v_{\text{min}} \) is discretized into small domains of width \( \varepsilon \) (\( \varepsilon = 0.01 \) was used). Only the node with the lowest \( g \) in that domain is kept while other nodes with a higher \( g \) are eliminated. This process is illustrated in Fig. 4 and presented in Algorithm 5. This node elimination is an important factor for making the ED MPC algorithm real–time implementable. It effectively assumes that different nodes in the search tree have the same physical meaning. Although this method is problem specific, it can be applied to general problems where the nodes have a contextual relation.

VI. RESULTS AND EVALUATION

In this section we assess the results obtained with the proposed ED MPC BnB algorithm against a human driver behavior control baseline and an MPC PMP solution for three different routes. We evaluate the results based on: total fuel...
Algorith 5 Context Based Node Elimination 2/2

Inputs: $A$, $v_{\min,k}$, $v_{\min,k}$, $\varepsilon$

1. Discretize $v$ into $n$ sections, yielding:

   $v_{\min,k}$, $v_{\min,k}$, $v_{\min,k}$, ..., $v_{\max,k}$, $v_{\max,k}$

2. Define nodes set $C$

   for $(i = 0; i < n; i = i + 1)$ {
   
   3.1 $B = \{ \text{nodes} \} \in (v_{\min,k} + \varepsilon < v_{\min,k} + (i + 1)\varepsilon) \subseteq A$
   
   3.2 add node with lowest LB of $B$ to $C$
   
   }

Output: $C$

TABLE I

| Route          | Start coordinate | End coordinate | Length [m] |
|---------------|------------------|----------------|-----------|
| Israel route  | 3°38'59.6"N     | 31°17'25.8"N  | 45900     |
| France route  | 48°51'25.7"N    | 49°26'35.8"N  | 132940    |
| VECTO cycle   | N/A              | N/A            | 100180    |

consumption, trip duration, and the real–time computation constraint. We emphasize a comparison with the PMP solution since a comparison with a human driver has already been presented in [3].

Three routes, presented in Table I, are assessed to avoid making route–specific conclusions. The Israel route is near Neve Zohar, and the France route is from Paris to Rouen. Lastly, the third route is the standardized VECTO long hauling cycle [35]. The $v_{\max}$ profile of the routes is shaped to include stop signs, curvatures, and traffic lights events, while $v_{\min} = 10$ [km/h].

For standstill events, we also set $v_{\max} = 10$ [km/h] for the following reasons: 1) the dynamics discretization (node spacing) is too coarse for the lowest velocities, yielding less feasible nodes; 2) the lowest velocities are incompatible with the $\omega_{e,min}$ constraint; and 3) the dynamics are infeasible at $v = 0$. The results in this section are focused on the Israel route due to space limitations.

A. Baseline Reference of Human Driver

To assess the relative fuel savings compared to existing methods, a comparison is made with the driving behavior of a (simulated) human driver. This human driving behavior forms a baseline reference and consists of a speed controller to track the maximum velocity as velocity reference and a previewing time, in which the HDT can decelerate earlier in case the velocity reference decreases in the previewed horizon. For the longitudinal speed controller we use a PI controller with parameters $K_p = 10000$ and $K_i = 1$. The preview time is the equivalent of the anticipating behavior of a real human driver and, based on [36], is set to

$$t_{prev} = 2.8 + 0.25(v_0 - \min(v_{m,prev})),$$

with $v_{m,prev}$ the maximum velocity profile of the previewed horizon. Next, we model the gear shift heuristic as in [37] with a gear upshift if $\omega_e > 2000$ and a downshift if $\omega_e < 1000$. In addition, a low gear is kept in downhill situations to increase the braking torque [38]. The baseline reference velocity profile for the Israel route is illustrated in Fig. 10. For this scenario, the total fuel consumption was 2621.5 [g] with a trip duration of 2886 [s].

B. Simulation Results: Influence of the Tuning Parameters

In this section we analyze the influence of the tuning parameters $\phi$, $\beta$ and $N$ on the fuel consumption and time duration of the optimized profiles. For that, the routes are simulated for combinations of $\phi \in \{4, 6, 8, 10, 20, 30, 40, 50, 60, 70, 80, 100\}$, $\beta \in \{1, 5, 10, 20, 50, 100, 200\}$ and $N \in \{100, 200, 300, 400\}$ with $\delta s = 20$.

The influence of the parameter $\phi$ on fuel consumption and trip duration illustrated in Fig. 5. One can observe that an increase in $\phi$ correlates with an increased fuel consumption and shorter trip duration; this behavior was consistent for the three routes. This correlation trend is also stronger for the BnB solver. Additionally, a larger spread in data points is visible for low $\phi$ values, suggesting that the influence of $N$ and $\beta$ increases as $\phi$ decreases. However, no trend in the influence of $\beta$ was observed.

Next we examined the influence of the prediction horizon $N$, as it significantly affects computation times (see Section VI-C). From Fig. 6 we notice that for a larger $\phi$, $N$ has a relatively small influence on fuel consumption. Therefore, using larger $\phi$-values enables the BnB MPC solver to also run on less powerful hardware.

Fig. 5. Total fuel consumption and trip duration for various simulations of the Israel route for the ED MPC Bnb and PMP solver.

Fig. 6. Influence of prediction horizon $N$ on the Israel route for different $\phi$. Observe that a larger $N$ does not always yield a lower fuel consumption.
Fig. 7. Israel route. Respectively from top to bottom: velocity–trajectories; driving mode; gear position. Simulation settings: \( N = 200, \phi = 10, \beta = 10 \).

Results of this particular simulation (total fuel consumed [g], trip duration [s]):
- PMP ED MPC: 2901 [s], 9722 [g];
- BnB ED MPC: 2927 [s], 8601 [g];

Performance:
- BnB vs. PMP: 0.90% time increase, 11.53% fuel saved;
- BnB vs. baseline reference: 1.43% time increase, 31.85% fuel saved.

**Table II**

| Route          | Average Estimation of Optimization Intervals in Matlab and C++ | Average Computation Time of Optimization Intervals in Matlab and C++ |
|---------------|----------------------------------------------------------------|---------------------------------------------------------------------|
| N=100         | 0.1729 0.0066 0.2590 0.0130 0.2399 0.0120                         | 0.5202 0.0260 0.7364 0.0368 0.6598 0.0330                           |
| N=200         | 1.0387 0.0519 1.3930 0.0697 1.2316 0.0616                         | 1.7609 0.0880 2.2655 0.1133 2.0182 0.1009                           |
| N=300         |                                                                |                                                                     |
| N=400         |                                                                |                                                                     |

On the other hand, a lower \( \phi \) increases the influence of \( N \) on fuel consumption and a larger horizon does not necessarily result in lower fuel savings \((N = 200 \text{ is optimal})\). The main cause for this is that the average \( v_{\text{max}} \)-constraint (which is approximately followed) and the relative potential energy of the HDT are prediction horizon–dependent, such that the fuel consumption and trip duration do not increase proportionally with the horizon increase.

**C. Real–Time Constraint**

The algorithms were simulated in Matlab, which has no explicit code compilation step. Once compiled in C++, for instance, a 20x speedup can be achieved [8]. In [9] the real–time constraint is quantified with a policy update of 1 [Hz] for a driver advisory system, resulting in a time constraint of 20 [s] on the optimization time in Matlab. The average optimization computation times of the routes are presented in Table II, along with the estimated computation time in C++. One can observe that the constraint is met and that the computation time increases disproportionately with a horizon increase.

**D. Simulation Results: Final Route Simulation**

We now evaluate the results on a final route simulation in which we include the observations made in the previous sections. Our choice of parameters is as follows. A maximum trip duration increase of 5% with respect to the baseline reference trip duration is allowed, leading to a choice of \( \phi = 10 \). Based on the trend in Fig. 6 and a trade–off with computational time, we also selected \( N = 200 \). Finally, we use \( \beta = 10 \).

The total fuel consumption, trip duration, and relative fuel savings for the three routes are presented in Table III. One can observe that the BnB solver consistently outperformed the PMP solution and the human driver baseline reference. In addition, it can also be observed in Table III that the highest fuel savings occurred on the Israel route, while this is also the shortest route (Table I). It is therefore likely that extending the Israel route will yield similar fuel savings as the other two. The resulting velocity profile with corresponding driving mode and gear position for the Israel route is illustrated in Fig. 7.

One can observe that the BnB and PMP velocity profiles are similar and relatively close to \( v_{\text{max}} \). The main differences are mostly observed at road gradient changes and \( v_{\text{max}} \) transients. Consider for instance the velocities around 11000 [m]: around this point, the BnB velocity is lower than the PMP velocity and the altitude profile indicates a local maximum. This means that part of potential energy is temporarily converted into kinetic energy and that BnB is able to capture this advantage more efficiently.

This performance difference can be explained by two main differences: while PMP is only guaranteed to be locally...
Fig. 8. France route. Respectively from top to bottom: velocity-trajectories; driving mode; gear position. Simulation settings: $N = 200$, $\phi = 10$, $\beta = 10$. Results of this particular simulation (total fuel consumed [g], trip duration [s]): PMP ED MPC: 7621 [s], 38909 [g]; BnB ED MPC: 7706 [s], 35976 [g]; Performance: BnB vs. PMP: 1.12% time increase, 7.54% fuel saved; BnB vs. baseline reference: 2.13% time increase, 16.15% fuel saved.

Fig. 9. VECTO long haul cycle. Respectively from top to bottom: velocity-trajectories; driving mode; gear position. Simulation settings: $N = 200$, $\phi = 10$, $\beta = 10$. Results of this particular simulation (total fuel consumed [g], trip duration [s]): PMP ED MPC: 4686 [s], 24547 [g]; BnB ED MPC: 4719 [s], 23671 [g]; Performance: BnB vs. PMP: 0.70% time increase, 3.57% fuel saved; BnB vs. baseline reference: 0.96% time increase, 15.15% fuel saved.

optimal [39] (by the initial co-state guess) and focuses on matching the boundary constraints, the BnB algorithm is approximately globally optimal (from a driving modes perspective) and therefore finds a lower cost.

Finally, one can observe that the driving mode-gear switching occurs rather frequently since it is not penalized in the cost function, and the ICE dynamic behavior is omitted. Including these two elements in the optimization problem results in a more realistic outcome. However, this increases the computation time.

VII. CONCLUSION

In this work, we proposed a dedicated MPC BnB solver for a driving modes ED problem including slope and V2X preview information to meet real–time implementability constraints. We introduced two new driving modes for the inclusion of look–ahead slope information in the OCP. To make the solver real–time implementable, context–based node elimination was applied, a context–based heuristic in the lower bound was used and the solver was warm–started with a PMP solution. Then, we evaluated the results of the BnB solver, as well as a human driver model and a MPC PMP–based solution in a numerical simulation of three routes. In comparison with a human driver and a PMP solution, respectively, up to 25.8% and 12.9% fuel savings are achieved on average. The BnB solver also consistently outperforms the human driver and the MPC PMP–based solution. These fuel savings were consistently achieved at the lowest time/fuel weight ratios in the cost function, but not necessarily at the largest prediction horizon.

In future work, it can be investigated whether the developed driving modes method approached the optimal solution that is based on continuous control variables. Second, a driving mode-gear switching penalty and ICE dynamic model can be
Fig. 10. Simulated human driver velocity profiles. Respectively from top to bottom: Israel route, France route, VECTO long haul cycle.

Fig. 11. Total fuel consumption and trip duration for various simulations of the France route (top) and VECTO cycle (bottom). This figure complements Fig. 5.

Fig. 12. Influence of horizon $N$ on the France route (left) and VECTO cycle (right). This figure complements Fig. 6.

added to the optimization problem to study their effect on fuel consumption and trip duration. Moreover, three things are expected to improve performance: 1) the heuristic can be improved using a feedforward artificial neural network; 2) the nodes can be branched in parallel; 3) the warm-started initial solution can be improved by also using the optimal driving mode sequence of the previous optimization interval.

APPENDIX

In this section we present the results of the France route and VECTO cycle. The results of the Israel route are presented in Section VI.

REFERENCES

[1] EE Agency. (2018). Carbon Dioxide Emissions From Europe’s Heavy-Duty Vehicles. Accessed: Mar. 30, 2021. [Online]. Available: https://www.eea.europa.eu/themes/transport/heavy-duty-vehicles/carbon-dioxide-emissions-europe

[2] European Commission. (2020). Reducing CO$_2$ Emissions From Heavy-Duty Vehicles. Accessed: Apr. 2, 2021. [Online]. Available: https://ec.europa.eu/clima/policies/transport/vehicles/heavy_en

[3] B. Wingelaar, G. R. Gonçalves Da Silva, M. Lazar, Y. Chen, and J. T. B. A. Kessels, “Design and assessment of an eco-driving PMP algorithm for optimal deceleration and gear shifting in trucks,” in Proc. IEEE Conf. Control Technol. Appl. (CCTA), Aug. 2021, pp. 8–13.

[4] Y. Xu, H. Li, H. Liu, M. O. Rodgers, and R. L. Guensler, “Eco-driving for transit: An effective strategy to conserve fuel and emissions,” Appl. Energy, vol. 194, pp. 784–797, May 2017.

[5] D. Shen, D. Karbowski, and A. Rousseau, “A minimum principle-based algorithm for energy-efficient eco-driving of electric vehicles in various traffic and road conditions,” IEEE Trans. Intell. Vehicles, vol. 5, no. 4, pp. 725–737, Dec. 2020.

[6] A. Sciarretta, G. De Nunzio, and L. L. Ojeda, “Optimal ecodriving control: Energy-efficient driving of road vehicles as an optimal control problem,” IEEE Control Syst. Mag., vol. 35, no. 5, pp. 71–90, Oct. 2015.

[7] V. K. Lakshmanan, A. Sciarretta, and O. E. Ganaoui-Mourlan, “Cooperative control in eco-driving of electric connected and autonomous vehicles in an un-signalized urban intersection,” IFAC-PapersOnLine, vol. 55, no. 24, pp. 64–71, 2022.

[8] N. Rozkvas, “Real-time velocity and driving mode optimization for hybrid trucks using previewed road and traffic data,” M.S. thesis, Dept. Elect. Eng., Eindhoven Univ. Technol., Eindhoven, The Netherlands, 2019.

[9] E. C. H. Thomassen, “Anticipative velocity optimization for heavy duty trucks using road geometry information,” M.S. thesis, Dept. Elect. Eng., Eindhoven Univ. Technol., Eindhoven, The Netherlands, 2017.

[10] J. Zhu, C. Ngo, and A. Sciarretta, “Real-time optimal eco-driving for hybrid-electric vehicles,” IFAC-PapersOnLine, vol. 52, no. 5, pp. 562–567, 2019.

[11] J. Hu, Y. Shao, Z. Sun, M. Wang, J. Bared, and P. Huang, “Integrated optimal eco-driving on rolling terrain for hybrid electric vehicle with vehicle-infrastructure communication,” Transp. Res. C, Emerg. Technol., vol. 68, pp. 228–244, Jul. 2016.
WINGELAAR et al.: MODEL PREDICTIVE ED CONTROL FOR HDTs USING BnB OPTIMIZATION 15189

[12] E. Hellström, M. Ivanson, J. Aslund, and L. Nielsen, “Look-ahead control for heavy trucks to minimize trip time and fuel consumption,” Control Eng. Pract., vol. 17, no. 2, pp. 245–254, Feb. 2009.

[13] A. Hamednia, N. K. Sharma, N. Murgovski, and J. Fredriksson, “Computationally efficient algorithm for eco-driving over long look-ahead horizons,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 7, pp. 6556–6570, Jul. 2022.

[14] L. Guo, B. Gao, Y. Guo, and H. Chen, “Optimal energy management for HEVs in eco-driving applications using bi-level MPC,” IEEE Trans. Intell. Transp. Syst., vol. 18, no. 8, pp. 2153–2162, Aug. 2017.

[15] N. Robuschi, C. Zeile, S. Sager, and F. Braghin, “Multiphase mixed-integer nonlinear optimal control of hybrid electric vehicles,” Automatica, vol. 123, Jan. 2021, Art. no. 109325.

[16] Y. Chen and M. Lazar, “Real-time driving mode advice for eco-driving using MPC,” in Proc. IFAC World Congr., 2020, pp. 13830–13835.

[17] H. Chen, L. Guo, H. Ding, Y. Li, and B. Gao, “Real-time predictive cruise control for eco-driving taking into account traffic constraints,” IEEE Trans. Intell. Transp. Syst., vol. 20, no. 8, pp. 2858–2868, Aug. 2019.

[18] D. P. Waltzing, R. D. Connors, and H. Chen, “Fuel-optimal truck path and speed profile in dynamic conditions: An exact algorithm,” Eur. J. Oper. Res., vol. 306, no. 3, pp. 1456–1472, May 2023.

[19] M. Held, “Fuel-efficient look-ahead control for heavy-duty vehicles with varying velocity demands,” Ph.D. dissertation, KTH Royal Inst. Technol., Stockholm, Sweden, 2020.

[20] J. Zhang and T. Shen, “Real-time fuel economy optimization with nonlinear MPC for PHEVs,” IEEE Trans. Control Syst. Technol., vol. 24, no. 6, pp. 2166–2171, Nov. 2016.

[21] P. Bonami and J. Lee, “BONMIN user’s manual,” IBM Res. Division, Thomas J. Watson Research Center, Yorktown Heights, NY, USA, Tech. Rep., 2008.

[22] J. Holmberg, “Heavy duty vehicle simulation tool for the calculation of fuel consumption and development of a new vecto gearshift algorithm,” M.S. thesis, KTH Royal Inst. Technol., Stockholm, Sweden, 2018.

[23] B. Saaerens, “Optimal control based eco-driving,” Ph.D. dissertation, Dept. Eng., Katholieke Universiteit Leuven, Leuven, Belgium, 2012.

[24] D. Forsberg and M. Hall, “Reduced fuel consumption of heavy-duty vehicles using pulse and glide,” M.S. thesis, Dept. Elect. Eng., Linköping Univ., Linköping, Sweden, 2019.

[25] F. Rodríguez, “Fuel consumption simulation of HDVs in the EU: Comparisons and limitations,” in Proc. Int. Clean Transp. Clean (ICCT), 2018, pp. 1–27.

[26] DAF. (2021). Retarder Systems. Accessed: Oct. 15, 2021. [Online]. Available: https://www.daft.com/-/media/files/document-library/retarders/guide/daf retarder systems 71053-hqpdf.pdf

[27] P. Somol, P. Pudil, F. Ferri, and J. Kittler, “Fast branch and bound algorithm in feature selection,” in Proc. World Multiconf. Systemics, Cybernetics Inform., vol. 7, 2000, pp. 1646–1651.

[28] L. F. Mendonça, J. M. Sousa, and J. M. G. Sá da Costa, “Optimization problems in multivariable fuzzy predictive control,” Int. J. Approx. Reasoning, vol. 36, no. 3, pp. 199–221, Jul. 2004.

[29] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, “Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning,” Discrete Optim., vol. 19, pp. 79–102, Feb. 2016.

[30] Baeldung. (2021). The Informed vs. Uninformed Search Algorithms. Accessed: Nov. 2, 2021. [Online]. Available: https://www.baeldung.com/cs/informed-vs-uninformed-search

[31] P. Belotti, “Couenne: A user’s manual,” IBM Res. Division, Thomas J. Watson Research Center, Yorktown Heights, NY, USA, Tech. Rep., 2020.

[32] O. Chevrant-Breton, T. Guan, and C. W. Frey, “Search space reduction in dynamic programming using monotonic heuristics in the context of model predictive optimization,” in Proc. 17th Int. IEEE Conf. Intell. Transp. Syst. (ITSC), Oct. 2014, pp. 2113–2118.

[33] Z. Ajanović, M. Stolz, and M. Horn, “A novel model-based heuristic for energy-optimal motion planning for automated driving,” IFAC-PapersOnLine, vol. 51, no. 9, pp. 255–260, 2018.

[34] D. P. Waltzing, “Preprocessing and probing techniques for mixed integer programming problems,” ORSA J. Comput., vol. 6, no. 4, pp. 445–454, Nov. 1994.

[35] European Commission. (2021). Vehicle Energy Consumption Calculation Tool—Vecto. accessed: Nov. 2, 2021. [Online]. Available: https://ec.europa.eu/energy/en/transport/energy-consumption-vehicles/vehicle-energy

[36] D.-W. Xing, X.-S. Li, X.-L. Zheng, Y.-Y. Ren, and Y. Ishiwatari, “Study on driver’s preview time based on field tests,” in Proc. 4th Int. Conf. Transp. Inf. Syst. (ICTIS), Aug. 2017, pp. 575–580.

[37] W. Enang, C. Bannister, C. Brace, and C. Vagg, “Modelling and heuristic control of a parallel hybrid electric vehicle,” Proc. Inst. Mech. Eng. D, J. Automobile Eng., vol. 229, no. 11, pp. 1494–1513, 2015.

[38] B. Song, J. Lv, Y. Liu, and F. Kong, “The simulation and analysis on engine and hydraulic retarder continual braking performance of the tracked vehicle on long downhill,” in Proc. 9th Int. Conf. Electron. Meas. Instrum., Aug. 2009, p. 928.

[39] M. Kelly, “An introduction to trajectory optimization: How to do your own direct collocation,” SIAM Rev., vol. 59, no. 4, pp. 849–904, Jan. 2017.

Bart Wingelaar was born in Gorinchem, The Netherlands, in 1996. He received the B.Sc. degree from the Eindhoven University of Technology, The Netherlands, in 2019, and the M.Sc. degree in systems and control from the Electrical Engineering Department, Eindhoven University of Technology, in 2021. Since 2022, he has been a research engineer in the field of marine engineering. His current research interests include optimal control, model predictive control, and complex algorithm design.

Gustavo R. Gonçalves da Silva received the bachelor’s degree in mechatronics engineering and the Ph.D. degree in electrical engineering from the Federal University of Rio Grande do Sul, Porto Alegre, Brazil, in 2014 and 2019, respectively. He held a postdoctoral position with the Eindhoven University of Technology, where he is working with algorithms for heavy-duty trucks real-time velocity profile optimization. His current research interests include model reference and data-driven control methods for MIMO systems, process control, and system identification.

Mircea Lazar (Member, IEEE) is currently an Associate Professor of constrained control of complex systems with the Electrical Engineering Department, Eindhoven University of Technology, The Netherlands. He received the VENI Personal Grant from the Dutch Research Council (NWO) in 2008. He has supervised ten Ph.D. researchers [two received (cum laude)]. His current research interests include stability theory, Lyapunov functions, distributed control, constrained control of nonlinear and hybrid systems, model predictive control, control problems in power systems, power electronics, high-precision mechatronics, and automotive and biological systems. He is an active member of the IFAC Technical Committees 1.3 Discrete Event and Hybrid Systems and 2.3 Nonlinear Control Systems. He received the European Embedded Control Institute Ph.D. Award for his Ph.D. dissertation in 2007. He has chaired the Fourth IFAC Conference on Nonlinear Model Predictive Control in 2012. He is an Associate Editor of IEEE TRANSACTIONS ON AUTOMATIC CONTROL.