Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Predictive Hybrid Powertrain Energy Management with Asynchronous Cloud Update

Junpeng Deng * **Luigi del Re** * **Stephen Jones** **

* *Johannes Kepler University Linz, Altenbergerstrae 69, 4040 Linz, Austria (e-mail: junpeng.deng@jku.at, luigi.delre@jku.at).*
** *AVL List GmbH, Graz, Austria, (e-mail: Stephen.Jones@avl.com)*

Abstract: The optimal energy management of a hybrid powertrain has the task to provide the required traction power combining both power sources in the best way. This can be achieved well if the future drive cycle is known/precomputed. However, both speed and traction power requirement may deviate from the expected ones due to many factors, like traffic, weather etc. Against this background, it might be sensible to recompute them whenever needed to keep using the latest future information. Unfortunately, this computation is typically too slow for real time use. In this paper we propose a control structure in which the real time task is solved by a predictive controller which tracks the optimal reference from the cloud, and requests an update of the reference regularly. The update can integrate new information from V2X. This asynchronous operation allows recovering most of the performance of the perfect prediction, while removing tight constraints on the offline computation and copes better with interruptions in communications to the cloud.

Keywords: powertrain control, predictive control, HEV, cloud computing, energy management

1. INTRODUCTION

Higher energy efficiency is a key requirement for new vehicles, and one of the main motivations of the increasing interest in hybrid electrical vehicles (HEV). Driving style is one of the critical factors affecting energy efficiency, and correspondingly, much work has been done to develop algorithms to determine the ideal speed profile if the route is known *a priori*, the so called eco-driving (see e.g. Sciarretta et al. [2015]; Hellström et al. [2009]). For classical vehicles, eco-driving algorithms usually deliver gear shifts as well. They mostly rely on dynamic programming (DP), see e.g. Bertsekas [2016], which provides a global optimum.

In HEVs two kinds of energy sources are used with independent storage units and components. Optimal energy efficiency is not as simple to define as in the conventional vehicle, as both energy kinds cannot be simply added. In the case of autarchic hybrids, i.e. those who produce all energy from fuel and only save it temporarily in electrical form, it is possible to treat all losses equally (Hahn et al. [2015]). In general the so called Equivalent Cost Minimization Strategy (ECMS) is used (Serrao et al. [2009]), which essentially defines an equivalence factor between electrical and fuel consumption. However, this equivalence factor is associated with and sensitive to the specific powertrain setup and driving cycles, and it is very difficult to determine it a priori (Onori et al. [2016]).

⋆ This work has been supported by the LCM K2 Center within the framework of the Austrian COMET-K2 program. The first author thanks Gian Paolo Incremona for his helpful discussion and suggestions during the work.

Of course, DP based eco-driving can be applied to HEVs as well, with the additional inputs (especially State-of-Charge (SOC)). However, the computation burden of DP increases exponentially with the number of states and inputs, thus can offer only a slow reaction to changes in the driving conditions. A tempting alternative would be to use nonlinear model predictive control (NMPC) algorithms which allow to approximate the original problem over a shorter time horizon, with some loss of performance but with the possibility to include constraints on traffic (see e.g. Polterauer et al. [2019]). But the efficiency loss can be significant, for instance if future slopes are outside the prediction horizon.

Against this background, there have been several proposals to do both, i.e. to use a hierarchical structure (Scattolini and Colaneri [2007]) consisting typically of a slow but more precise computation for the whole trip, and of a "safe" but less optimal MPC for the short term operation (see e.g. Buerger et al. [2018]; Tianheng et al. [2014]).

Having the whole computational power on board may be prohibitive for vehicles productions, and the increasing availability of cloud computational power suggests moving the demanding computations there, with the additional advantage that the cloud can acquire additional information at infrastructure level, like traffic jams, road blocks etc. (Mell et al. [2011]). The use of the cloud to perform heavy task like DP has already been the subject of interest. For example, Ozatay et al. [2014] proposes a cloud-based velocity profile optimization to improve fuel economy, Grubwinkler et al. [2013]; Yang et al. [2017] utilize the cloud to predict possible power demand, with

the latter work also calculating fuel-optimal power split actions over a pre-known driving route. Besides these works, the potential of cloud computing and dual communication is not yet fully exploited. First, most of the works use the cloud only once before departure, there is no update or recalculation to cope with traffic changes, which could be vital in a long trip. Second, the control is usually placed in the vehicle layer, but powertrain actions especially gear shifts, are guided by heuristics.

Eco-driving is usually implemented as an advisory system, e.g. the optimal speed is suggested, but it is left to the driver (human or automated) to follow it keeping in account the actual traffic constraints. In other words, the two activities — the computation of optimal profile and the implementation of these suggestions are to some extent independent control tasks.

In this paper, we propose a dual-layer controller in which an "upper" layer is responsible of producing and updating a reference over longer horizon on the cloud. A "lower" layer uses NMPC to provide in an optimal way the instantaneous requested power while tracking this reference compatibly with the actual environment. As the availability of the cloud cannot be always guaranteed, the lower layer has to work as a fall-back control, and it is also responsible for sending the request to the cloud asking for an updated reference based on latest traffic. Furthermore, a potential analysis of energy savings at the powertrain level resulting from a reference update is conducted.

2. MODELS

2.1 Drive cycle (actual velocity profile)

The drive cycle comes from measurements by our colleagues in a city route in Linz, Austria. It consists of two signals: geographic altitude $h(s)$ and velocity profile $v(s)$ along the distance s as shown in Fig.1.

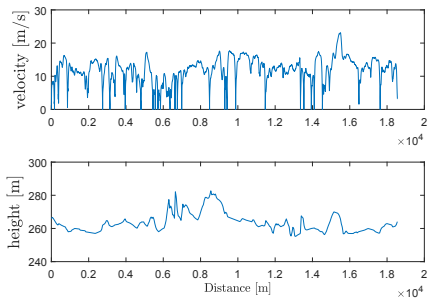

Fig. 1. Real drive cycle

2.2 Vehicle model

It is assumed that the powertrain is stiff and there is no wheel slip. The traction force F_p, depending on the drive cycle, can be computed by

$$\begin{aligned} F_p &= F_r(\phi, v) + \lambda^{(j)} ma \\ F_{\mathrm{r}}(\phi, v) &= mg\left(c_{\mathrm{r}} + \sin\left(\phi(s)\right)\right) + \frac{c_{\mathrm{d}}\rho_{\mathrm{air}}A}{2} v^2 \end{aligned} \tag{1}$$

where F_{r} is the resistance force, m is the vehicle mass, a is the vehicle acceleration, j is the selected gear and $\lambda^{(j)}$ is the factor accounting for the rotational inertia, g is the gravitational constant, c_{r} is the rolling resistance, $\sin\left(\phi(s)\right)$ is the road grade, c_{d} is the drag coefficient, ρ_{air} is the air density, A is the front area of the vehicle. The specific parameters are listed in Appendix A.

The engine speed ω_{e} can be related to the vehicle speed v through

$$\omega_{\mathrm{e}} = \frac{\gamma^{(j)}}{r} v \tag{2}$$

where $\gamma^{(j)}$ is the gear dependent transmission ratio and r is the wheel radius. By using equation Eq.1 and 2, the power demand P can be calculated by

$$P = F_p \cdot v \tag{3}$$

2.3 Powertrain model

Fig. 2 illustrates the structure of the P2 parallel hybrid powertrain studied in this work. The wheel is connected with brake and gearbox. P is provided by either the engine P_e or battery P_b, or both, and this power split is decided by coupler, i.e.

$$P + P_L = P_e + P_b \tag{4}$$

where P_L is the power loss of the whole powertrain.

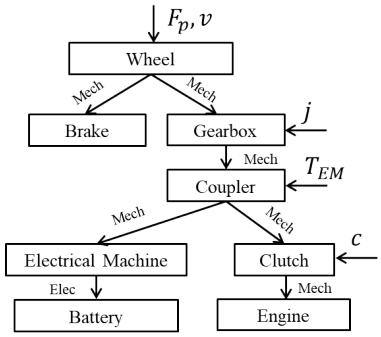

Fig. 2. P2 parallel hybrid structure

Engine model The engine model is given in Eq.5.

$$\begin{aligned} \tau_{\mathrm{eff}} &= \tau + J \cdot \dot{\omega_{\mathrm{e}}} \\ q_{\mathrm{f}} &= q_{\mathrm{f}}(\omega_{\mathrm{e}}, \tau_{\mathrm{eff}}) \end{aligned} \tag{5}$$

where τ is the engine torque, J is the inertia, ω_{e} is the engine speed, q_{f} is the fuel flow rate.

Electric machine The electrical power is determined based on a electrical power map using effective torque and rotational speed. The effective torque is the torque including acceleration of machine inertia.

Battery model The state of charge ξ can be described by (Tremblay et al. [2007]):

$$\begin{aligned}\dot{\xi} &= \frac{1}{Q_n} \cdot I_b(\xi, P_b) \\ I_b(\xi, P_b) &= \frac{-U_{oc}(\xi) + \sqrt{U_{oc}(\xi)^2 - 4P_b R_i}}{2R_i} \\ U_b &= U_{oc} + R_i \cdot I_b \\ U_{oc}(\xi) &= \left(E_0 - \frac{K}{\xi} + Ae^{(B(\xi-1))}Q_n\right)\end{aligned} \tag{6}$$

where Q_n is the nominal charge, I_b is the battery current, U_{oc} is open circuit voltage, E_0 is the battery constant voltage, K is polarisation voltage, A and B are model constants, U_b is the battery clamp voltage, R_i is internal resistance.

3. PROBLEM DESCRIPTION

As already stated, we assume the future velocity profile estimation and thus—for a known topology and vehicle—estimated traction power profiles are given. Thus, our goal is to provide the required traction power profile for a hybrid vehicle at the minimum cost acting on the power split and gear shift. We use backward modelling approach (Hofmann [2014]). The costs include fuel consumption, number of gear shifts and clutch changes, and we require a charge sustaining operation.

3.1 Reference updating strategy

Before vehicle starts a trip, it sends a request to the cloud, the cloud returns the estimated velocity $\hat{v}(s)$ and topology information of this trip to the upper layer controller, which is also in the cloud. The upper layer controller computes the estimated power demand $\hat{P}$ and a general optimal reference $x_r^*(s)$, which includes the reference gear, SOC and clutch position over the whole route. Our powertrain control tracks the powertrain variables using a short-prediction horizon NMPC.

However, due to the disturbances in traffic, following the same reference may not be optimal any longer. In these cases, the local controller can send a request to the cloud asking for recomputation of the reference based on current situation. The request of re-calculation can be triggered by either A fixed distance or time interval, or the deviation between real states of vehicle $x(s)$ and $x_r^*(s)$, or the deviation between $\hat{v}$ and actual speed, etc. Next, the cloud re-estimates and the upper layer controller accordingly re-calculates the reference $x_r^*(s)$ and sends it back to the vehicle. The vehicle starts following the new $x_r^*(s)$ until the end of the trip or next trigger point. This process is illustrated in Fig. 4.

4. OPTIMIZATION PROBLEM FORMULATION

The scheme is depicted in Fig.4, and will be explained in details in the following section.

4.1 Global optimization – upper layer

The upper layer control aims at providing a globally optimal reference over the whole route, therefore DP is used as a control method. In our case, the states $x_k = [j_k, \xi_k, c_k]$, where j_k is the gear position, ξ_k is the SOC of the battery, c_k is the clutch position. The inputs $u_k = [u_{\tau,k}, u_{j,k}, u_{c,k}]$ are the power split between battery and engine, gear shift command, and clutch shift command respectively. For the brake strategy, a heuristic is applied, i.e. all brake torque is used for battery recharging when the battery is not full. When the battery is full or the required brake torque exceeds the maximum power of the battery, the excessive brake torque is "discarded" in the mechanical brakes.

The cost function and the constraints of optimal control problem (OCP) of DP is

$$\begin{aligned} J_{\mathrm{DP}} = &\sum_{k=0}^{N-1} \left[q_{\mathrm{f}}(k)T_s + \beta_1 z_j(k) + \beta_2 z_c(k)\right] \\ \omega_{\mathrm{e,min}} \leq &\omega_{e,k} \leq \omega_{\mathrm{e,max}} \\ j_k \in &\{1, \ldots, N_{\mathrm{gear}}\} \\ \xi_{\mathrm{min}} \in &\xi \leq \xi_{\mathrm{max}} \\ \xi_N \in &[\xi_{\mathrm{min,N}}, \xi_{\mathrm{max,N}}] \\ c_k \in &\{0, 1, 2\} \\ u_{j,k} \in &\{-1, 0, 1\} \\ u_{c,k} \in &\{0, 1, 2\} \end{aligned} \tag{7}$$

where T_s is the sampling time, β_1, β_2 are weighting factors, z_j is the total number of gear shifts, z_c is total number of clutch shifts, $N = T/T_s$ is predicted number of time steps, where T is the total time of the driving cycle. The clutch has three states: closed, slip, and open. The gear shift action can only be shifted up, down, or remain the same. To keep charge sustaining, the final value of SOC is constrained to a range that is more or less equal to initial SOC. The specific discretization and parameters are listed in table A.2.

Given initial conditions, the optimal sequence x_r^* can be computed.

4.2 Local optimization – lower layer

In the lower layer, MPC is deployed for global reference following and disturbances handling over a short horizon. Due to fast-changing traffic, there may be some discrepancies between predicted power demand $\hat{P}$ (by cloud) and actual power demand P_{act} (by driver). Here we use the prediction horizon of 10s, assuming that P_{act} within 10 seconds is known.

The cost function of the lower layer MPC is

$$J_{\mathrm{MPC}} = \sum_{k=t}^{t+N_p} \left[m_1 \Delta\xi_k^2 + m_2 \Delta j_k^2 + m_3 \Delta c_k^2 + m_4 q_{\mathrm{f}}(k) T_s\right] \tag{8}$$

where Δ refers to the difference between current value of states and reference value. m_1, m_2, m_3, m_4 are weighting factors. N_p is the prediction horizon.

Apart from the tracking errors, there is an additional cost term q_f. This is added to robustify the controller in order

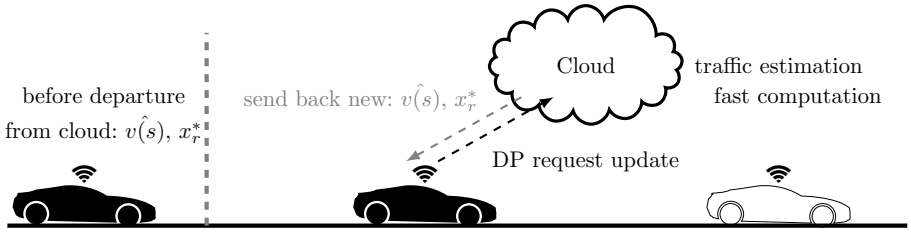

Fig. 3. Graphic rendering of interaction between vehicle and cloud

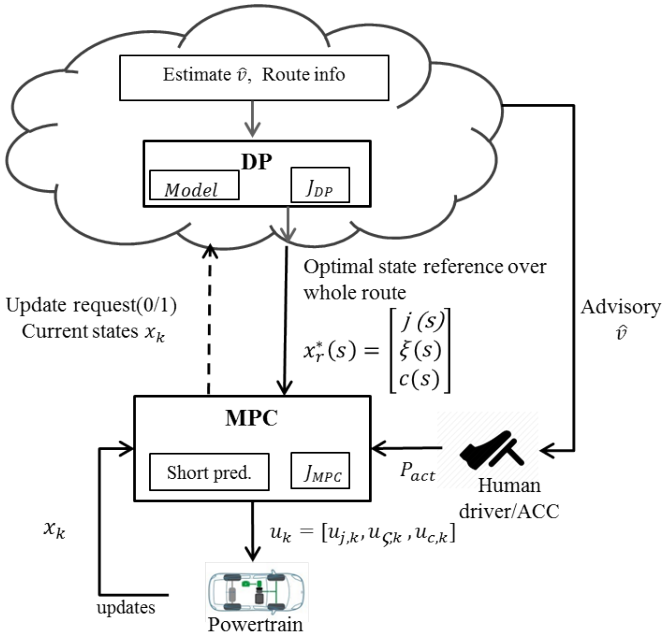

Fig. 4. Control scheme with DP at cloud level and MPC at vehicle level

to cope with uncertainties and communication failure—in uncertainties free case ($v = \hat{v}$), the output of the MPC is the solution with minimum tracking error, which is also the solution with minimum fuel—because they are the same. If the current reference is not optimal ($v \neq \hat{v}$), or when there is interruption in communication, these two cost terms have conflicts with each other. The controller has to balance between the general trend (tracking error) and local optimum (fuel). This is to prevent unreliable results from MPC due to simply following the inaccurate reference.

Since the vehicle and powertrain models are the same as that of DP, the constraints are the same too, except that the terminal constraint on SOC is no longer required.

5. CASE STUDY AND RESULTS

The case study aims at testing above updating strategy and exploring the potential of performance improvement by updates. Here we update the reference based on a fixed distance interval over the route.

5.1 Predicted speed profiles generation

For the first proof of concept, we consider a simplified problem as follows. We split the route into several sections with equal distance, according to the number of updates n. The section number is then $n + 1$. For example, if an update happens twice as shown in Fig.5, the route is divided into three parts. The actual velocity, taken from a measurement, is plotted in black dash. The first prediction is in blue, the second prediction (red line) starts at one third of the route, and third prediction starts at two thirds of the route. All predictions are assumed to be accurate at the beginning, and getting more and more inaccurate as the distance increases. Here we processed the attenuation of prediction quality by a scaling factor 1.3, which means the predicted speed proportionally increases from 1 to 1.3 times of nominal values.

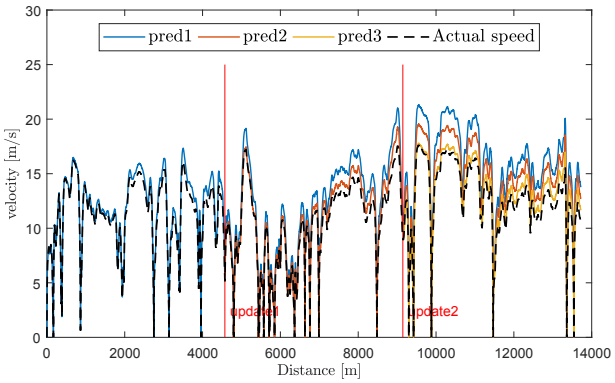

Fig. 5. Velocity profiles and prediction (update twice)

According to Fig.5, the corresponding power demand is calculated as well. Then the optimal references of the real profile $x_{r,real}^*$ and each prediction can be derived by solving DP. The case study considers two approaches:

- Approach 1: follow non-updated reference, i.e. $n = 0$. The powertrain follows the reference based on the first prediction along the whole route.
- Approach 2: follow updated reference. If $n = 2$, the powertrain follows the reference based on the first prediction for first one third of the road, then switch to the reference based on second prediction until two thirds of the route, and then switch to reference based on third prediction until the end of the route. $x_{r,upn}^*$ is used for the actual reference followed. Here we consider the update times up to four times, i.e. $n \in \{0, 1, 2, 3, 4\}$.
- To set a baseline, we run a simulation by tracking $x_{r,real}^*$. This is the "best" that can be achieved by tracking—when first prediction is perfect.

Following Fig.6 shows the results of case $n = 0$ where the reference is not updated. The powertrain tracks the

reference well in general, despite some small variations on SOC and clutch during distance range 6000 to 7000. This is because due to the disturbances, the reference is no longer optimal—the minimum tracking error solution is different from the minimum fuel consumption solution. The controller has to pursue a balance between following the general trend and local optimization. The tracking result is able to get closer to the reference, when an update happens as shown in Fig.7. For all cases, MPC tracks the reference well generally, but due to limited space only example plots are shown here.

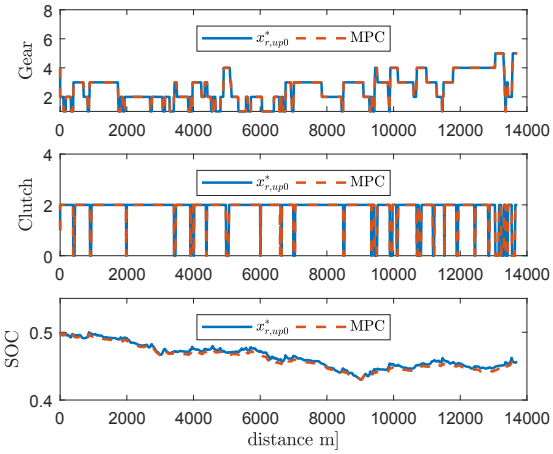

Fig. 6. Results of Approach 2: tracking non-updating reference

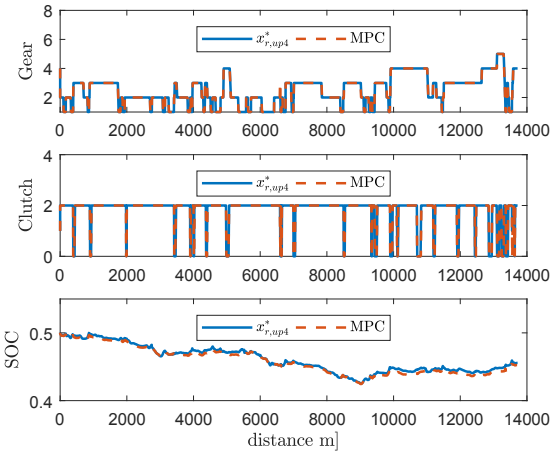

Fig. 7. Results of Approach 2: tracking updated four times reference

Fig. 8 compares the performances of above approaches. The upper figure shows the additional fuel consumption compared with DP offline solution. The lower figure shows the total cost of the MPC results. To have a fair comparison, the cost of MPC is calculated based on DP cost function J_{DP} using eq.7. It is worth mentioning that, the final SOC values of all MPC results are within 1% difference range compared to DP, therefore we regard them all as charge-sustaining, i.e. no need to consider additional penalty on final SOC values.

As shown in the upper figure in Fig.8, the baseline case has more or less the same performance with DP, but small variation (less than 2%) due to numerical issues (most likely due to the discretization approximations). When tracking the non-updated reference, i.e. $n = 0$, the additional fuel consumption is up to 10%. Although the controller tried to minimize the fuel consumption when the reference is not optimal, it still gives compromised results due to short prediction horizon. The additional fuel consumption is 8% when the reference is updated once, and 6.5% when the reference is updated two or three times. When $n = 4$, the additional fuel consumption is only about 5% more than DP, which is already quite close to the baseline case. In general, with more updates, the additional fuel consumption becomes lower. It is noticed that fuel consumption of $n = 3$ is slightly higher than $n = 4$, but please bear in mind that performance does not only include fuel consumption, but also gear shifts and clutch changes. So if we see the lower figure which depicts the overall cost, it is clear that the more frequent updates of the reference, the better the overall performance. The numerical results are listed in table A.3 in Appendix A.

It would be also interesting to look at the performance of the controller if MPC only tracks the reference without doing local optimization on fuel. To this end, we set m_4 to zero in Eq.8 and run all simulations again. The results shows 13% additional fuel consumption in average more than the case where fuel is considered. It illustrates that, adding the cost term q_f helps robustify the controller—even if the upper layer reference is no longer optimal, the lower layer controller can still gain back some performance.

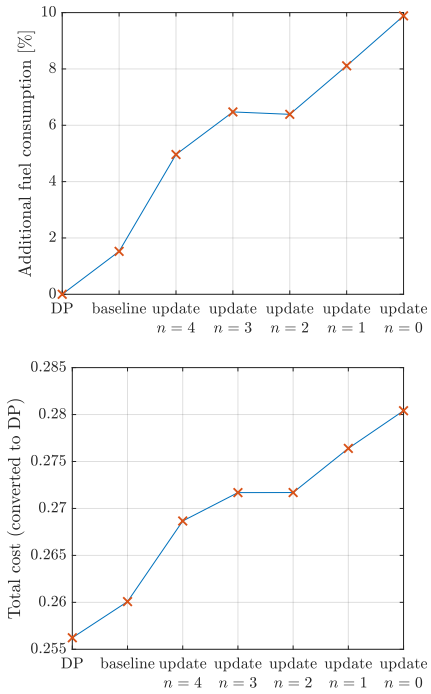

Fig. 8. Additional fuel consumption and total cost compared with DP solution

In a nutshell, updating the reference improves the performance of the controller. More frequent updates mean more in-time correction and therefore better results. It is worth noticing that, proportional scaling is a relatively simplified way to modify the prediction. In a congested city area,

the "shape" of real traffic can be much more varied, and we assumed more potential can be exploited by updating reference.

6. CONCLUSION AND OUTLOOK

The paper proposed a dual-layer powertrain energy management strategy for hybrid vehicles, with an upper layer DP-based optimizer providing an updatable general solution, and a lower layer MPC tracking the reference while handling the actual situations. Under the structure of communication between envisioned remote cloud and local controller, the reference can be more responsive and closer to the global optimum, without the concerns of computational burden.

The future work could consider more flexible updating strategies, for example a trigger condition can be the detection of significant deviation between actual velocity and reference one. Adaptive distance/time update strategies can also be practical to accommodate the different scenarios. Criteria of how to select the strategies could be built. In addition, a real-world validation using a cloud can be very interesting.

Appendix A. PARAMETERS

Table A.1. Vehicle model parameters

variable	g	ρ_{air}	m	r	A	c_b	c_r
value	9.81	1.2	1585	0.315	2.2	4500	0.0108
units	m/s^2	kg/m^3	kg	m	m^2	Nm	-
variable	c_d	η					
value	0.2578	0.92					
units	-	-					
$\gamma^{(j)}$	[12.06, 8.05, 5.39, 4.27, 3.29, 2.56, 2.15, 1.71]						
$\lambda^{(j)}$	[1.3, 1.24, 1.16, 1.1, 1.06, 1.04, 1.02, 1]						

Table A.2. Optimization parameters

variable	$\omega_{\text{e,min}}$	$\omega_{\text{e,min}}$	T_s	β_1	β_2	$\xi_{\min}$	$\xi_{\max}$
value	104.72	366.52	1	2e-4	2e-4	0.1	0.9
unit	rad/s	rad/s	s	-	-	-	-
variable	$\xi_{min,N}$	$\xi_{max,N}$	m_1	m_2	m_3	m_4	
value	0.45	0.55	2	2e-3	2e-3	2e-2	
discre.	dis_ξ	dis_{u_τ}					
	100	31					

Table A.3. Case study results

Case	DP	Baseline	$n=4$	$n=3$	$n=2$
Fuel cons. [kg]	0.2264	0.2299	0.2377	0.2411	0.2409
Total cost	0.2562	0.2601	0.2687	0.2716	0.2717
Case	$n=1$	$n=0$			
Fuel cons. [kg]	0.2448	0.2488			
Total cost	0.2764	0.2804			

REFERENCES

Bertsekas, D. (2016). *Dynamic programming and optimal control*, volume 13. Athena Scientific, Belmont, MA, USA.

Buerger, J., East, S., and Cannon, M. (2018). Fast dual-loop nonlinear receding horizon control for energy management in hybrid electric vehicles. *IEEE Transactions on Control Systems Technology*, 27(3), 1060–1070.

Grubwinkler, S., Kugler, M., and Lienkamp, M. (2013). A system for cloud-based deviation prediction of propulsion energy consumption for evs. In *Proceedings of 2013 IEEE International Conference on Vehicular Electronics and Safety*, 99–104. IEEE.

Hahn, S., Waschl, H., Steinmaurer, G., and del Re, L. (2015). Extension of a linear optimal control strategy for hev. In *2015 European Control Conference (ECC)*, 154–159. IEEE.

Hellström, E., Ivarsson, M., Åslund, J., and Nielsen, L. (2009). Look-ahead control for heavy trucks to minimize trip time and fuel consumption. *Control Engineering Practice*, 17(2), 245–254.

Hofmann, P. (2014). Definitionen und klassifizierung der hybridkonzepte. In *Hybridfahrzeuge*. Springer. doi:10.1007/978-3-7091-1780-4 2. URL `http://dx.doi.org/10.1007/978-3-7091-1780-4_2`.

Mell, P., Grance, T., et al. (2011). The NIST definition of cloud computing. *Computer Security Division, Information Technology Laboratory*.

Moser, D., Waschl, H., Schmied, R., Efendic, H., and del Re, L. (2015). Short term prediction of a vehicle's velocity trajectory using its. *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, 8(2015-01-0295), 364–370.

Onori, S., Serrao, L., and Rizzoni, G. (2016). *Hybrid electric vehicles: energy management strategies*, volume 13. Springer.

Ozatay, E., Onori, S., Wollaeger, J., Ozguner, U., Rizzoni, G., Filev, D., Michelini, J., and Di Cairano, S. (2014). Cloud-based velocity profile optimization for everyday driving: A dynamic-programming-based solution. *IEEE Transactions on Intelligent Transportation Systems*, 15(6), 2491–2505.

Polterauer, P., Incremona, G.P., Colancri, P., and del Re, L. (2019). A switching nonlinear mpc approach for ecodriving. In *2019 American Control Conference (ACC)*, 4608–4613. Philadelphia, PA, USA.

Scattolini, R. and Colaneri, P. (2007). Hierarchical model predictive control. In *Proc. 46th IEEE Conference on Decision and Control*, 4803–4808.

Sciarretta, A., De Nunzio, G., and Ojeda, L.L. (2015). Optimal ecodriving control: Energy-efficient driving of road vehicles as an optimal control problem. *IEEE Control Systems Magazine*, 35(5), 71–90.

Serrao, L., Onori, S., and Rizzoni, G. (2009). ECMS as a realization of pontryagin's minimum principle for HEV control. In *Proc. American Control Conference*, 3964–3969. St. Louis, Missouri, USA.

Tianheng, F., Lin, Y., Qing, G., Yanqing, H., Ting, Y., and Bin, Y. (2014). A supervisory control strategy for plug-in hybrid electric vehicles based on energy demand prediction and route preview. *IEEE Transactions on Vehicular Technology*, 64(5), 1691–1700.

Tremblay, O., Dessaint, L.A., and Dekkiche, A.I. (2007). A generic battery model for the dynamic simulation of hybrid electric vehicles. In *2007 IEEE Vehicle Power and Propulsion Conference*, 284–289. Ieee.

Yang, C., Li, L., You, S., Yan, B., and Du, X. (2017). Cloud computing-based energy optimization control framework for plug-in hybrid electric bus. *Energy*, 125, 11–26.