Stochastic MPC With Learning for Driver-Predictive Vehicle Control and its Application to HEV Energy Management

Stefano Di Cairano, *Member, IEEE*, Daniele Bernardini, Alberto Bemporad, *Fellow, IEEE*, and Ilya V. Kolmanovsky, *Fellow, IEEE*

***Abstract*— This paper develops an approach for driver-aware vehicle control based on stochastic model predictive control with learning (SMPCL). The framework combines the on-board learning of a Markov chain that represents the driver behavior, a scenario-based approach for stochastic optimization, and quadratic programming. By using quadratic programming, SMPCL can handle, in general, larger state dimension models than stochastic dynamic programming, and can reconfigure in real-time for accommodating changes in driver behavior. The SMPCL approach is demonstrated in the energy management of a series hybrid electrical vehicle, aimed at improving fuel efficiency while enforcing constraints on battery state of charge and power. The SMPCL controller allocates the power from the battery and the engine to meet the driver power request. A Markov chain that models the power request dynamics is learned in real-time to improve the prediction capabilities of model predictive control (MPC). Because of exploiting the learned pattern of the driver behavior, the proposed approach outperforms conventional model predictive control and shows performance close to MPC with full knowledge of future driver power request in standard and real-world driving cycles.**

***Index Terms*— Automotive controls, driver-machine interaction, energy management, model predictive control (MPC), optimization, real-time learning, stochastic control.**

I. Introduction

While modern vehicles are complex systems composed of mechanical, electrical, and electronic subsystems, the primary element that affects the vehicle operation is still the driver. Thus, vehicle control strategies that seek highly optimized performance need to optimize the system composed of the vehicle and the driver, hence explicitly accounting for the driver behavior.

Manuscript received October 3, 2012; accepted June 8, 2013. Manuscript received in final form July 3, 2013. Date of publication July 25, 2013; date of current version April 17, 2014. Recommended by Associate Editor L. Del Re.

S. Di Cairano was with Ford Research and Advanced Engineering, Dearborn, MI 48127 USA. He is now with the Department of Mechatronics, Mitsubishi Electric Research Laboratories, Cambridge, MA 02139 USA (e-mail: dicairano@ieee.org).

D. Bernardini and A. Bemporad are with IMT Institute for Advanced Studies, Lucca 55100, Italy (e-mail: daniele.bernardini@imtlucca.it; alberto.bemporad@imtlucca.it).

I. V. Kolmanovsky is with the Department of Aerospace Engineering, the University of Michigan, Ann Arbor, MI 48109 USA (e-mail: ilya@umich.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2013.2272179

Driver information is not easily exploited by classical control strategies, but model predictive control (MPC) appears to be suitable for this purpose. MPC repeatedly optimizes a control sequence over a receding horizon by exploiting a model to predict the future system behavior. Because of the capability of achieving high performance in multivariable systems subject to constraints, MPC has attracted considerable interest in the automotive industry (see, [1]–[9] and references therein).

To optimize the overall system composed of vehicle and driver, the MPC prediction model must capture the driver behavior. Although detailed models are available for the dynamics of the vehicle components [10], [11], suitable frameworks for modeling the driver behavior are less established. In some cases, the driver is modeled as a feedback controller that seeks to achieve a certain control goal, such as tracking a reference [12], [13]. In other cases, the driver is represented by an autonomous system, often driven by a random process. For instance, [14] proposes a linear model with additional nonlinearities such as actuator saturation, slew-rate, and time delays, [15] proposes a hidden Markov model, and [16] proposes nonlinear ARMAX models. In [17], a hybrid driver model is proposed, which consists of discrete modes and continuous control functions.

In this paper,[1] we consider a discrete stochastic model of the driver where the actions are correlated in time. The model takes the form of a Markov chain, similarly to [15]. Markov chains have been previously shown to be effective for capturing certain driver behaviors, see for instance [15], [20], [21], and the discrete state space makes them good candidates for use within numerical control algorithms. Because of the Markov chain, the optimization of the vehicle-driver model requires a stochastic control approach. Stochastic controllers have been used in automotive applications to tackle the uncertainty that arises from the environment around the vehicle, and to generate optimal control solutions taking into account the statistics of the disturbances. For instance, [20]–[23] apply stochastic dynamic programming (SDP) to optimize fuel economy and emissions, and [24] applies linear stochastic optimal control to chassis control. As it is known, curse of dimensionality limits the application of SDP to low-order models. In addition, the large computational

[1] Preliminary studies related to this paper were presented in [18] and [19].

effort required to compute the SDP solution results in the impossibility of rapidly updating the control policy in real-time in reaction, for instance, to changes of the stochastic model. On the other hand, linear stochastic control methods, although computationally simpler, are based on assumptions that usually do not allow to fully capture the driver behavior. In this paper, we define a strategy based on MPC that exploits system theory and numerical algorithms to efficiently achieve the stochastic optimization of the vehicle-driver system.

In recent years, various stochastic model predictive control (SMPC) algorithms have been proposed, based on different prediction models and control problem formulations, see for instance [25]–[30]. In this paper, we build upon the SMPC originally proposed in [29] based on scenario enumeration and quadratic programming. The relevant driver behaviors are modeled by a Markov chain and the stochastic vehicle-driver model is used in a finite horizon optimal control problem, where the average of the performance objective over different scenarios is optimized subject to constraints on state and input variables. The scenarios represent the driver actions that, according to the Markov chain, are most likely to realize. In this paper, we introduce online learning of the Markov chain, which allows to adjust to variations of the driver behavior. By updating the Markov chain in the SMPC, the controller adapts with minimal computational effort to changes in the driver behavior, for instance due to varying traffic conditions, road types, or driver emotional states and objectives.

The proposed control approach is demonstrated in energy management of a hybrid electric vehicle (HEV). Indeed, the driver behavior strongly affects fuel consumption, and hence HEV energy management. The energy management control system [11], [31] selects the power flows from the energy sources and storages to satisfy the driver power request, while accounting for constraints in power flows and energy storages. Indeed, an improved prediction of the driver actions allows for a better prediction of the future power request, and hence for more informed decisions on the power flows. The control design proposed in this paper uses statistical information on the driver that is updated in real-time to adjust to changes in the driver behavior, possibly in response to environment changes. Besides the specific application to the HEV energy management, the framework developed in this paper is useful for addressing general vehicle control problems where the overall vehicle behavior optimization requires real-time estimation of the statistical driver action patterns, as demonstrated, for instance, for adaptive cruise control in [19].

This paper is organized as follows. Section II describes the Markov chains as models for the driver and a Markov chain learning algorithm to adapt to changes in driver behavior. In Section III, we introduce the stochastic model predictive control algorithm and we combine it with the learning algorithm to obtain SMPC with learning (SMPCL). Then, we apply the SMPCL approach to energy management of a series hybrid electric vehicle. In Section IV, we introduce the series HEV (SHEV) architecture and the simulation model used to validate our control algorithm, and in Section V we design the SHEV energy management by SMPCL. In Section VI, we present the simulation results of the control strategy in closed-loop with the SHEV simulation model in standard and real-world driving cycles. The SMPCL performance is compared with a standard MPC and a MPC with perfect driver preview along the entire horizon. The conclusions are summarized in Section VII.

Notation: $\mathbb{R}$, $\mathbb{R}_{0+}$, $\mathbb{R}_{+}$, $\mathbb{Z}$, $\mathbb{Z}_{0+}$, $\mathbb{Z}_{+}$ denote the set of real, nonnegative real, positive real, integer, nonnegative integer, and positive integer numbers, respectively, and $\mathbb{R}_{(a,b)} = \{c \in \mathbb{R} : a < c < b\}$, where a similar interpretation is given also to $\mathbb{R}_{[a,b]}$, $\mathbb{Z}_{(a,b)}$, etc. For a set $\mathcal{A}$, $|\mathcal{A}|$ denotes the cardinality. For a vector a, $[a]_i$ is the ith component. For a matrix A, $[A]^j$ is the j column, $[A]_{ij}$ is the element of the ith row and jth column. We denote a square matrix of size $s \times s$ entirely composed of zeros by 0_s, and the identity matrix by I_s, where subscripts are dropped when clear from the context.

II. Stochastic Model Learning of Driver Behavior

We start by introducing a model of the driver based on Markov chains, where the states capture the possible driver actions and where the transition probabilities are updated in real-time to adapt to changes in driver behavior.

A. Markov Chain-Based Driver Models

Let the driver actions be modeled by a stochastic process $w(\cdot)$ where $w(k) \in \tilde{\mathcal{W}} \subset \mathbb{R}$ for all $k \in \mathbb{Z}_{0+}$. Even though we consider a scalar $w(k)$ for simplicity, the extension to vector-valued process $w(\cdot)$ is straightforward. With a little abuse of notation, we denote by $w(k)$ the realization of the disturbance at $k \in \mathbb{Z}_{0+}$. Depending on the application, $w(k)$ may represent quantities such as power request, acceleration, velocity, steering wheel angular rate, or a combination of the above. All these quantities are actually measured in the vehicle through standard sensors, and hence we assume that $w(k)$ is measured at time k but is unknown for $t > k$.

For prediction purposes, the random process generating w is modeled (with some approximation) by a Markov chain with values in $\mathcal{W} = \{w_1, w_2, \ldots, w_s\} \subset \mathbb{R}$, where $w_i < w_{i+1}$ for all $i \in \{1, \ldots, s-1\}$. The cardinality $|\mathcal{W}|$ defines the tradeoff between the complexity of the stochastic model and its ability to capture the driver behavior. The Markov chain is defined by a transition probability matrix $T \in \mathbb{R}^{s \times s}$, such that

$$[T]_{ij} = \Pr[w(k+1) = w_i \mid w(k) = w_j] \tag{1}$$

for all $i, j \in \{1, \ldots, s\}$. Given $p(k) \in \mathbb{R}^s$ where $[p(k)]_j = \Pr[w(k) = w_j]$, the probability distribution of $w(k+1)$ is described by

$$p(k+1) = Tp(k). \tag{2}$$

B. Driver Model Learning

While first principles vehicle models can be derived from physics and plant parameters, stochastic models of the driver

are identified from data. Identifying a Markov chain requires estimating the transition probabilities $[T]_{ij}$, which amounts to estimating the transition frequencies.

Consider first the case of identification from a batch of L measurements, $\{w_m(k)\}_{k=0}^{L}$. For given $\mathcal{W}$, $\tilde{\mathcal{W}}$, define $w_0 = 2\inf\{w,\ w \in \tilde{\mathcal{W}}\} - w_1$, and $w_{s+1} = 2\sup\{w,\ w \in \tilde{\mathcal{W}}\} - w_s$. Then, let $\mathcal{I}_i = \{w \in \mathbb{R} : (w_{i-1}+w_i)/2 < w \leq (w_i+w_{i+1})/2\}$ denote the interval associated with the state w_i of the Markov chain, for all $i \in \{1, \ldots, s\}$. Define

$$\mathcal{K}_{ij} = \{k \in \mathbb{Z}_{[1,L]} : w_m(k+1) \in \mathcal{I}_i,\ w_m(k) \in \mathcal{I}_j\} \quad (3)$$

$n_{ij} = |\mathcal{K}_{ij}|$, i.e., the number of transitions from w_j to w_i, and $n_j = \sum_{i=1}^{s} n_{ij}$, i.e., the number of transitions from w_j. The transition matrix T is estimated by

$$[T]_{ij} = \frac{n_{ij}}{n_j},\ \forall i, j \in \{1, \ldots, s\}. \quad (4)$$

Proposition 1: Consider $\mathcal{W} = \tilde{\mathcal{W}}$, and the measurements $\{w_m(k)\}_{k=0}^{L}$. Assume $\Pr[w(k) = w_j]$, $j \in \{1, \ldots, s\}$, is defined by the Markov chain (2), and let the transition probability matrix T be estimated by (4). Then, if each state of the Markov chain is positive recurrent, $\lim_{L\to\infty}[T]_{ij} = \Pr[w(k+1) = w_i \mid w(k) = w_j]$. □

The proposition is an immediate consequence of the law of large numbers [32]. Indeed, the correct estimation requires data that span the entire state-space of the Markov chain, according to the positive recurrence assumption.

While (4) identifies the transition probability matrix from a batch data set, since the driver behavior changes over time, the Markov chain needs to be updated online by a recursive algorithm. Let $\delta_j \in \{0, 1\}^s$, for all $j \in \{1, \ldots, s\}$. At time $k \in \mathbb{Z}_{0+}$, $[\delta_j]_i(k) = 1$ if and only if $w(k) \in \mathcal{I}_i$ and $w(k-1) \in \mathcal{I}_j$. Hence, the vectors δ_j define which transition has occurred, and T is recursively estimated by

$$n_j(k) = n_j(k-1) + \textstyle\sum_{i=1}^{s}[\delta_j(k)]_i \quad (5)$$

$$\lambda_j(k) = \frac{1}{n_j(k)}\textstyle\sum_{i=1}^{s}[\delta_j(k)]_i \quad (6)$$

$$[T(k)]^j = (1 - \lambda_j(k))[T(k-1)]^j + \lambda_j(k)\delta_j(k) \quad (7)$$

for all $j \in \{1, \ldots, s\}$, where the initialization $n_j(0) = \bar{n}_j$ and $T(0) = \bar{T}$ may be obtained, for instance, from (4) based on data available *a priori*. Equation (5) updates the number of transitions from state w_j, (6) stores the total number of transitions from each state observed so far, and (7) updates the transition matrix. Note that only one column of T is actually updated at each time step, since each transition provides new information only on the state from which the transition was observed. Indeed, the estimator (5)–(7) is equivalent to batch estimation (4).

The limitation of (5)–(7) is that the sensitivity to new data decreases with the amount of data, due to (6). This would not be a problem if the driver behavior was stationary. However, in real driving conditions the driver behavior may change significantly over time due to factors such as traffic conditions, road type, time of the day, driver's physical/emotional status, etc. To overcome such limitation we apply an estimator for which the sensitivity to data remains constant [32], by replacing (5) and (6) with

$$\lambda_j(k) = \bar{\lambda}\sum_{i=1}^{s}[\delta_j(k)]_i \quad (8)$$

for all $j \in \{1, \ldots, s\}$, where $\bar{\lambda} \in (0, 1)$ is a constant parameter. Equations (7) and (8) define an exponential averaging where $\bar{\lambda}$ trades off convergence rate for sensitivity to new data.

In Fig. 1 the effect of learning by (7) and (8) is shown on the Markov chain that is used later in Section VI. The Markov chain models the driver's power request while driving a small HEV along the new European driving cycle (NEDC). The Markov chain (2) is initialized by $T(0) = I_s$, $n_j(0) = 0$, for all $j \in \{1, \ldots, s\}$, and $s = 16$. In total, the NEDC cycle was repeated three times with $\bar{\lambda} = 0.01$. After the second execution of the NEDC cycle, the transition probabilities do not change significantly.

III. Stochastic Model Predictive Control With Learning

When the driver model proposed in Section II is considered in an MPC framework, the MPC optimal control problem results in a stochastic finite horizon optimal control problem. We solve such a problem by the scenario enumeration and multistage stochastic optimization originally proposed in [29]. Consider the linear discrete-time system

$$x(k+1) = Ax(k) + B_1u(k) + B_2w(k) \quad (9a)$$

$$y(k) = Cx(k) + D_1u(k) + D_2w(k) \quad (9b)$$

where $x(k) \in \mathbb{R}^{n_x}$ is the state, $u(k) \in \mathbb{R}^{n_u}$ is the input, $y(k) \in \mathbb{R}^{n_y}$ is the output, and $w(k) \in \mathcal{W}$ is a scalar stochastic disturbance, whose distribution $p(k)$ is modeled by the Markov chain (2). By (1)

$$p(k+1) = [T(k)]^j,\ \text{if } w(k) = w_j,\ j \in \{1, 2, \ldots, s\} \quad (10)$$

where it is assumed that $w(k)$ is known at time k. In automotive applications, the assumed knowledge on $w(k)$ is realistic because the driver actions are measured by vehicle sensors.[2] The state, input, and output vectors in (9) are subject to the pointwise-in-time constraints

$$x(k) \in \mathcal{X},\ u(k) \in \mathcal{U},\ y(k) \in \mathcal{Y} \quad \forall k \in \mathbb{Z}_{0+} \quad (11)$$

where $\mathcal{X} \subseteq \mathbb{R}^{n_x}$, $\mathcal{U} \subseteq \mathbb{R}^{n_u}$, and $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$ are polyhedral sets. Because of $w(k)$ in (9), the MPC problem minimizes a risk measure of a given performance index. We consider a quadratic function of the state and the input as the performance index, and the expected value as the risk measure

$$\mathbb{E}_{\{w(j)\}_{j=0}^{N-1}}\Big[\textstyle\sum_{j=1}^{N}(x(k+j) - x_{\text{ref}})'\,Q\,(x(k+j) - x_{\text{ref}}) + \sum_{j=0}^{N-1}u(k+j)'Ru(k+j)\Big] \quad (12)$$

where x_{ref} is a given state reference, N is the prediction horizon, and Q, R are weight matrices of appropriate dimensions. Since $|\mathcal{W}|$ is finite, (12) can be optimized by enumerating

[2] If $w(k)$ is not directly measured, $w(k-1)$ can be estimated from $x(k)$ and $x(k-1)$. Hence, $p(k+1) = T[T(k)]^j$, if $w(k-1) = w_j$, $j \in \{1, 2, \ldots, s\}$, i.e., one additional open-loop prediction step is required.

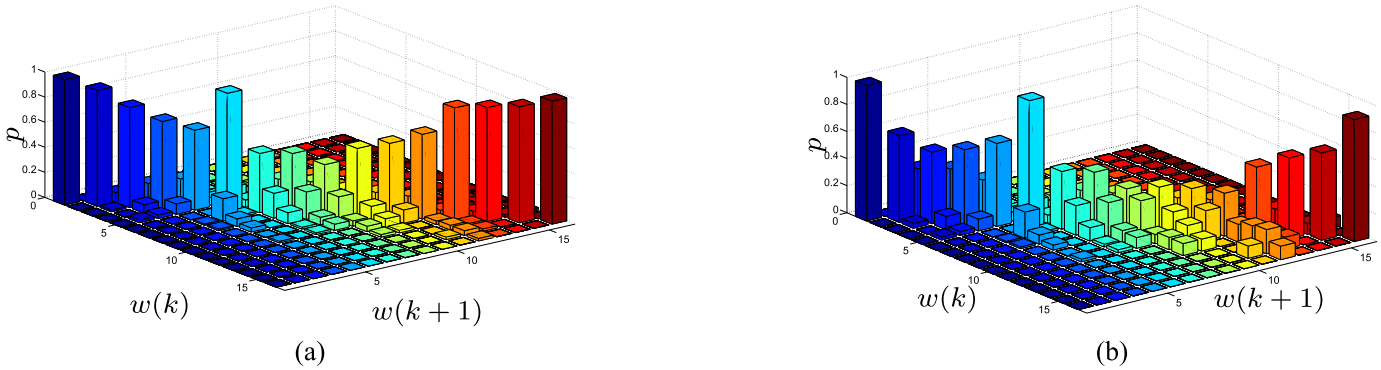

Fig. 1. Effect of Markov chain learning in the application in Section VI along several execution of NEDC cycle for $T(0) = I_s$, $\bar{\lambda} = 0.01$. (a) Transition probabilities after half NEDC cycle. (b) Transition probabilities after two NEDC cycles.

all the admissible realizations (the scenarios) of the stochastic disturbance sequence, and then solving an optimization problem with a control sequence per scenario, and appropriate constraints that enforce causality. However, the optimization problem obtained in this way is large, because it considers even disturbance sequences with arbitrarily small probability.

In our approach, (9)–(11) are used to construct a variable horizon optimization problem where only the disturbance sequences that are more likely to realize are accounted for, and hence the optimization problem is simplified. Instead of considering all possible scenarios by using (10) to compute the disturbance probability, we construct a scenario tree with variable depth. The scenario tree describes the most likely scenarios of future disturbance realizations, and is updated at every time step using newly available measurements of the state $x(k)$, the disturbance $w(k)$, and the updated estimate $T(k)$, according to the receding horizon philosophy of MPC.

The scenario tree is computed from the Markov chain model of the disturbance introduced in Section II. Let us define the following quantities.

- $\mathcal{T} = \{\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_n\}$: the set of the tree nodes. Nodes are indexed progressively as they are added to the tree (i.e., $\mathcal{N}_1$ is the root node and $\mathcal{N}_n$ is the last node added);
- $pre(\mathcal{N}) \in \mathcal{T}$: the predecessor of node $\mathcal{N}$;
- $succ(\mathcal{N}, w) \in \mathcal{T}$: the successor of node $\mathcal{N}$ for $w \in \mathcal{W}$;
- $\pi_{\mathcal{N}} \in [0, 1]$: the probability of reaching $\mathcal{N}$ (from $\mathcal{N}_1$);
- $x_{\mathcal{N}} \in \mathbb{R}^{n_x}$, $u_{\mathcal{N}} \in \mathbb{R}^{n_u}$, $y_{\mathcal{N}} \in \mathbb{R}^{n_y}$, $w_{\mathcal{N}} \in \mathcal{W}$: the state, input, output, and disturbance value, respectively, associated with node $\mathcal{N}$, where $x_{\mathcal{N}_1} = x(k)$, $y_{\mathcal{N}_1} = y(k)$, and $w_{\mathcal{N}_1} = w(k)$;
- $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_c\}$: the set of candidate nodes, defined as $\mathcal{C} = \{\mathcal{N} \notin \mathcal{T} \mid \exists (i, j) : \mathcal{N} = succ(\mathcal{N}_i, w_j)\}$;
- $\mathcal{S} \subset \mathcal{T}$: the set of leaf nodes, $\mathcal{S} = \{\mathcal{N} \in \mathcal{T} \mid succ(\mathcal{N}, w_j) \notin \mathcal{T}, \ \forall j \in \{1, \ldots, s\}\}$, whose cardinality is denoted by $n_{\text{leaf}} = |\mathcal{S}|$.

Every path from the root node to a leaf node represents a disturbance realization scenario that is considered in the optimization problem. The procedure to construct the scenario tree is listed in Algorithm 1 and described next.

Starting from the root node $\mathcal{N}_1$, which is associated with $w(k)$, a list $\mathcal{C}$ of candidate nodes is evaluated considering all the possible s future values of the disturbance in $\mathcal{W}$ and their realization probabilities. The candidate with maximum probability $\mathcal{C}_{i^*}$ is added to the tree and removed from $\mathcal{C}$. The procedure is repeated by generating at every step new candidates as children of the last node added to the tree, until the tree contains $n_{\max}$ nodes. Algorithm 1 expands the tree in the most likely direction, so that the paths with higher probability are extended longer in the future, since they may have more impact on performance. This leads to tree with a flexible structure where the paths from the root to the leaves may have different lengths and hence different prediction horizons (see Fig. 2). Thus, we call the tree a multiple-horizon optimization tree. The reader is referred to [29] for further details on the scenario-based SMPC approach and on the tree construction algorithm.

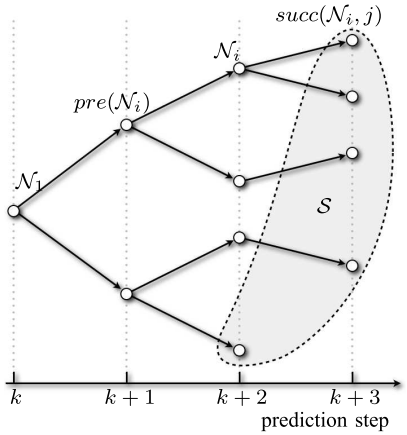

Fig. 2. Graphical representation of a multiple-horizon optimization tree. Some roof-to-leaves paths have length 2; others have length 3. Hence, different scenarios may have different prediction horizons.

For the sake of shortness, in what follows we use x_i, u_i, y_i, w_i, π_i, and pre(i) to denote $x_{\mathcal{N}_i}$, $u_{\mathcal{N}_i}$, $y_{\mathcal{N}_i}$, $w_{\mathcal{N}_i}$, $\pi_{\mathcal{N}_i}$, and pre$(\mathcal{N}_i)$, respectively. At time k, based on the tree constructed from $w(k)$ and $T(k)$, the following stochastic MPC problem

Algorithm 1 SMPC Tree Generation Procedure

At any step k:
set $\mathcal{T} = \{\mathcal{N}_1\}$, $\pi_{\mathcal{N}_1} = 1$, $n = 1$, $c = s$;
set $\mathcal{C} = \bigcup_{j=1}^{s} \{\text{succ}(\mathcal{N}_1, \text{w}_j)\}$
while $n < n_{\max}$ **do**
  for all $i \in \{1, 2, \ldots, c\}$, **do**
    compute $\pi_{\mathcal{C}_i}$ according to (10);
  end for
  set $i^* = \arg\max_{i \in \{1,2,\ldots,c\}} \pi_{\mathcal{C}_i}$;
  set $\mathcal{N}_{n+1} = \mathcal{C}_{i^*}$;
  set $\mathcal{T} = \mathcal{T} \cup \{\mathcal{N}_{n+1}\}$;
  set $\mathcal{C} = \bigcup_{j=1}^{s} \{\text{succ}(\mathcal{C}_{i^*}, w_j)\} \cup (\mathcal{C} \setminus \mathcal{C}_{i^*})$;
  set $c = c + s - 1$, $n = n + 1$;
end while

is solved:

$$\min_{\mathbf{u}} \sum_{i \in \mathcal{T} \setminus \{\mathcal{N}_1\}} \pi_i (x_i - x_{\text{ref}})' Q (x_i - x_{\text{ref}}) + \sum_{i \in \mathcal{T} \setminus \mathcal{S}} \pi_i u_i' R u_i \tag{13a}$$

$$\text{s.t.} \quad x_1 = x(k) \tag{13b}$$

$$x_i = A x_{\text{pre}(i)} + B_1 u_{\text{pre(i)}} + B_2 w_i,\ i \in \mathcal{T} \setminus \{\mathcal{N}_1\} \tag{13c}$$

$$y_i = C x_{\text{pre(i)}} + D_1 u_{\text{pre(i)}} + D_2 w_i,\ i \in \mathcal{T} \setminus \{\mathcal{N}_1\} \tag{13d}$$

$$x_i \in \mathcal{X},\ y_i \in \mathcal{Y},\ i \in \mathcal{T} \setminus \{\mathcal{N}_1\} \tag{13e}$$

$$u_i \in \mathcal{U},\ i \in \mathcal{T} \setminus \mathcal{S} \tag{13f}$$

where $\mathbf{u} = \{u_i : \mathcal{N}_i \in \mathcal{T} \setminus \mathcal{S}\}$ is the multiple-horizon input sequence. Eq. (13) is a quadratic program (QP) with $n_u(n_{\max} - n_{\text{leaf}})$ optimization variables. Once the problem is solved, the decision vector u_1 associated with the root node $\mathcal{N}_1$ is used as the control input $u(k)$. Causality in prediction is enforced by allowing only one control action for every node, except for leaf nodes where there are no control actions.

Equation (13) is an approximation of the optimization of the expected value (12). If the scenario tree $\mathcal{T}$ is fully expanded, i.e., all the leaf nodes are at depth N and all parent nodes have s successors, the objective function (13a) is equivalent to (12). Otherwise, (13a) is an approximation of (12) based on the largest probability scenarios. The representativeness-complexity tradeoff of the approximation is defined by the number of nodes in the tree $n_{\max}$, and possibly by a maximum depth for the tree. The stability of the closed-loop system can be addressed by including a stochastic control Lyapunov function in the form of constraints in (13), as discussed in [29]. The complete SMPCL strategy is summarized in Algorithm 2.

Remark 1: In (9), the disturbance with statistics defined by Markov chain (2) is additive. This is motivated by the application considered next. However, it is straightforward to apply the same approach with other types of disturbances, as long as the system dynamics for an assigned disturbance value is linear in the state and in the input, such as in the case of parametric uncertainties in (9). □

In the next section, we demonstrate the effectiveness of Algorithm 2 in automotive applications, by showing its capabilities in energy management of HEVs.

Algorithm 2 Stochastic MPC With Learning

for all $k \in \mathbb{Z}_{0+}$ **do**
  get measurements $x(k)$, $w(k)$;
  update $T(k)$ from $T(k-1)$ and $w(k)$ by (7), (8);
  construct the scenario tree by Algorithm 1;
  solve the SMPC problem (13) and obtain u_1;
  apply $u(k) = u_1$;
end for

IV. SHEV Energy Management

In recent years, HEVs have been increasingly introduced in the market because of their improved fuel efficiency, which is obtained by coupling the internal combustion engine with an electric drivetrain usually composed of an electric motor, a generator, and a battery. The electric and internal combustion drivetrains produce mechanical energy, and the electrical drivetrain can also convert mechanical energy into chemical energy stored in the battery. HEV energy management addresses the decision on how much power is generated/drained/stored in the different components to maximize fuel efficiency while providing the power that the driver requests and enforcing operating constraints on the powertrain. Indeed, the power request depends on the vehicle speed and the acceleration that the driver wants to achieve, and thus it is in general an expression of the driver behavior, also in reaction to the surrounding environment.

HEV energy management has been addressed in several ways, see for instance [11], [20], [23], [31], [33]–[38], and the references therein. Optimal solutions are based on dynamic programming (DP) [11], [33], [34] and assume full knowledge of the future power request, which ultimately defines the vehicle speed. These techniques provide the fuel economy ceiling (i.e., the upper bound) on an *a priori* known driving cycle, but they are unsuitable for normal real world driving, when the driving cycle is not known. DP also results in time-varying control laws that are memory-expensive to implement.

More recently, SDP has been proposed to enable the implementation of DP-based energy management in real driving [20], [23], [39]. In SDP, the knowledge of the future power request is substituted by its statistics obtained from data sets of potential driving cycles. SDP results in time-invariant control laws that depend on data statistics. However, the SDP computation is still time and resource expensive,[3] and hence the control law cannot be adjusted directly on the vehicle in response to changes to the power request statistics.

In this paper, we propose the approach developed in Section III that allows for the statistics of the power request to be updated in real-time, and hence adapts to the different styles of the driver (relaxed, performance, economical, etc.), to the driver's standard routes (city, highway, mixed, etc.), and to the traffic patterns that the driver commonly encounters (light

[3]The precise calculation of the DP policy for a standard driving cycle may take several days even in large-scale computing environments. This is due to the need for simulating high-fidelity inverse models on fine state space grids. The computation of the SDP policy may take (significantly) longer due to the need for performing multiple (value or policy) iterations.

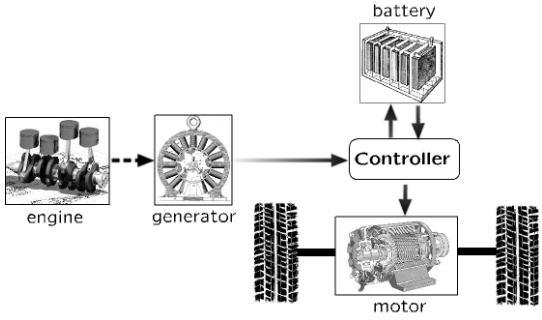

Fig. 3. Schematics of a series hybrid electric powertrain.

traffic, high-speed traffic, traffic jams, etc.). By learning the power request statistics and optimizing the energy efficiency in a SMPCL framework, we expect to achieve benefits similar to SDP, with the additional capability of adjusting the control strategy to the specific conditions with significantly lower computational effort. In particular, we expect the proposed approach to provide fuel economy improvements in everyday driving. Everyday driving performance is becoming significantly important for the automotive industry. In fact, the upcoming corporate average fuel economy (CAFE) standards [40] include provisions for "off-cycle driving," referred to as features that provide improvements of fuel efficiency and reduction of emissions that are not measurable on standard environmental protection agency (EPA) test cycles, but have effects in everyday driving. This is the case for optimizing the fuel economy in the commonly driven routes for the primary driver of the vehicle. Next, we discuss the physical architecture of the HEV considered in this paper, and the simulation model used for validating the SMPCL energy management strategy.

A. SHEV Powertrain Architecture

We consider the series hybrid electric vehicle (SHEV) [11], [20], [38] whose powertrain is shown schematically in Fig. 3. In SHEV, the electric motor is the unique source of traction at the wheels. The motor receives electric power from a DC bus to which a battery and a generator are connected. The generator converts the mechanical power from the engine into electrical power in the DC bus. Compared with the powersplit configuration [37] where the power flow coupling involves mechanical powers and is obtained by a planetary gear set, the electrical bus of the series configuration has a higher efficiency and fewer constraints [20], [41]. On the other hand, the mechanical power is always converted to electrical power, with power losses as a consequence. Series hybrid electric powertrain have been in marketed HEV passenger cars, are currently in marketed extended range electric vehicles (or plug-in HEV), are of interest for fuel-cell and diesel hybrid vehicles, and are used in military and commercial trucks and buses, also because of the more flexible packaging since the power can be transferred through the electrical bus instead of through the drivetrain.

According to Fig. 3, in the SHEV configuration the electric motor is the unique source of traction at the wheels

$$P_{\text{wh}}(t) = \eta_{\text{wh}}(t) P_{\text{mot}}(t) \tag{14}$$

where P_{wh}[W] is the power at the wheels, P_{mot}[W] is the power output of the electric motor, and $\eta_{\text{wh}} \in \mathbb{R}_+$ is the (time-varying) drivetrain efficiency.

Remark 2: In (14) and in all the subsequent power flow equations, we follow the convention that for generators and storages the power is positive when provided and negative when acquired, while for consumers (the wheels) the power is positive when acquired and negative when provided. Because of the bidirectionality of the power flows, the efficiency variables can assume values larger than 1. In particular, the efficiency variables in the equations are smaller than 1 if the power is positive, and greater than 1 otherwise.

The motor power results from the generator power and the power provided from the battery as

$$P_{\text{mot}}(t) = \eta_{\text{mot}}(t)(P_{\text{gen}}(t) + P_{\text{bat}}(t)) \tag{15}$$

where P_{gen}[W] is the generator power, $\eta_{\text{mot}} \in \mathbb{R}_+$ is the (time varying) motor efficiency, and P_{bat}[W] is the power flow from the battery to the electrical bus, and then to the motor.

The electrical generator is powered by the internal combustion engine

$$P_{\text{gen}}(t) = \eta_{\text{gen}}(t) P_{\text{eng}}(t) \tag{16}$$

where $\eta_{\text{gen}} \in \mathbb{R}_{(0,1)}$ is the (time varying) generator efficiency, and P_{eng}[W] is the engine brake power. Finally, the engine power determines the fuel consumption through the relation

$$P_{\text{fuel}}(t) = \frac{P_{\text{eng}}(t)}{\eta_{\text{eng}}(t)} \tag{17}$$

where P_{fuel}[W] is the amount of net power that can be extracted from the fuel burnt in the cylinders, $\eta_{\text{eng}}(t) \in (0, 1)$ is the engine efficiency, and from (17) the fuel mass flow w_f[kg/s] is $w_f = P_{\text{fuel}}/H_f$, where H_f[J/kg] is the specific lower heating value of the fuel [11].

The battery power flow in (15) changes the amount of charge stored in the battery. Given the battery power flow to the bus $P_{\text{bat}} = i_{\text{bus}} V_{\text{bus}}$, where i_{bus}[A] is the current in the bus and V_{bus}[V] is the (controlled) DC bus voltage, the battery charge Q_{bat}[C] evolves according to

$$\frac{d}{dt} Q_{\text{bat}}(t) = -i_{\text{bus}}(t) = -\frac{P_{\text{bat}}(t)}{\eta_{\text{bat}}(t) V_{\text{bus}}(t)} \tag{18}$$

where $\eta_{\text{bat}} \in \mathbb{R}_+$ is the (time varying) battery efficiency, which accounts for power losses in power electronics and battery.

While (14)–(17) are formulated in the power domain, the time varying efficiencies in (14)–(17) depend on the rotational speed and torque at which the components are operating. The efficiencies are usually described by static maps of the corresponding rotational speed and torque, mainly obtained from experimental data. In the SHEV, the generator and engine speeds are coupled, possibly through a reduction gear, and so are the electric motor and wheel speeds, usually through a gearbox or continuously variable transmission (CVT).

Thus, the electric motor speed is assigned by the wheel speed (i.e., the current vehicle speed) and the CVT/gearbox reduction ratio, which is usually not under direct control of the energy management strategy in the powertrain software

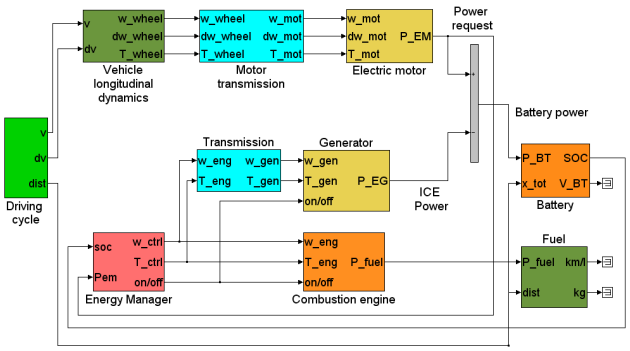

Fig. 4. Quasi-static SHEV model for closed-loop simulations.

architecture. On the other hand, the engine and generator speeds are decoupled from the electric motor and wheel speeds by the electrical bus. The optimal engine speed ω_{eng}[rad/s] and torque τ_{eng}[Nm] can be selected as functions of the generator power output, independently from the electric motor and wheel speeds, by a map $[\omega_{\text{eng}}\ \tau_{\text{eng}}] = \gamma(P_{\text{gen}})$. A proper selection of such a map, together with an appropriate control of the powertrain and battery dynamics, are the key for improving SHEV fuel efficiency.

B. Quasi-Static Simulation Model of SHEV

For closed-loop simulations of the SHEV, we use a quasi-static simulation (QSS) model in the QSS Toolbox [42], which implements a reversed causality quasi-static approach.

The simulation is quasi-static in the sense that the dynamic evolution is broken into a sequence of stationary states at discrete-time instants. Reversed causality means that the simulation is executed by reversing the classical causality relations. In causal simulations, torques and forces are causes that generate rotational speeds and velocities as effects. Hence, given the current speed and a selected force, the acceleration and the updated velocity are computed. In the reversed causality approach, from the current velocity and (desired) acceleration, the needed force and the updated velocity are computed. For instance, in the SHEV model, from the (desired) acceleration and current vehicle velocity, the vehicle longitudinal force is computed as

$$F(k) = \frac{m(v(k+1) - v(k))}{T_m} + c_2 v(k)^2 + c_1 v(k) + c_0$$

where c_0, c_1, c_2 are coefficients of the load model representing the rolling resistance, bearings friction, and airdrag, and T_m[s] is the simulation stepsize.

The major advantage of QSS, when compared with causal high-fidelity industrial simulation models (see [38]), is computational. Also, the model used here is open source and freely available [42].

The SHEV simulation model implemented in the QSS toolbox is a small fuel-efficient vehicle described in [11, Ch. 3] and augmented with an electrical motor and a battery. The efficiency maps of the components are obtained from experimental data and validated, see [11, Ch. 3, 4] and [42].

As shown in Fig. 4, in the SHEV simulation model, the rotational dynamics of the drivetrain and electrical motor are obtained, from the vehicle speed $v(k)$ and acceleration at the current step $(a(k) = (v(k+1) - v(k))/T_m)$ by reversed causality applied at each component. The mechanical couplings that impose kinematics and torque relations resolve the signal values in the powertrain components. Thus, from the current and next vehicle velocity, the required motor power $P_{\text{mot}}(k)$ is computed. Because of the quasi-static approach, the efficiencies in (14)–(17) are considered constant during each step, i.e., computed for the current torque and speed at the beginning of the step itself, and applied as multiplicative gains to the components' torques.

The free variable for the controller to manipulate in (15) is the generator power $P_{\text{gen}}(k)$. Let the generator power $P_{\text{gen}}(k)$ and the generator setpoint be assigned, from $P_{\text{mot}}(k)$ and $P_{\text{gen}}(k)$ the battery power $P_{\text{bat}}(k)$ is computed by (15), and the battery charge is updated by (22) integrated for the simulation stepsize T_m. Thus, the controller selects the engine operating point $(\omega_{\text{eng}}(k), \tau_{\text{eng}}(k))$, which determines the generator speed and torque, and hence the generator power $P_{\text{gen}}(k)$. At the same time, $(\omega_{\text{eng}}(k), \tau_{\text{eng}}(k))$ determines the engine efficiency $\eta_{\text{eng}}(k) = \eta_{\text{eng}}(\omega_{\text{eng}}(k), \tau_{\text{eng}}(k))$, and by (17) the fuel consumption during the simulation step, $w_f(k)T_m$. To better capture the impact of the engine dynamics on the efficiency, a triangular approximation of the engine efficiency is used, i.e., $\eta_{\text{eng}}(k) = 1/2\left(\eta_{\text{eng}}(\omega_{\text{eng}}(k), \tau_{\text{eng}}(k)) + \eta_{\text{eng}}(\omega_{\text{eng}}(k-1), \tau_{\text{eng}}(k))\right)$, which is based on the rationale that first the torque is changed, then the engine speed changes, see [38, Fig. 3].

V. SHEV Energy Management by SMPCL

Next we apply the SMPCL approach developed in Section III to the SHEV energy management described in Section IV. Deterministic MPC has been previously applied to energy management of hybrid electric vehicles for different powertrain configurations; see, for instance, [36]–[38].

For the SHEV whose powertrain schematics is shown in Fig. 3 and that was described in Section IV, the energy management strategy can be structured as composed of two parts. An algorithm that, given the current state of the hybrid powertrain and the power request, selects the generator power P_{gen}, and a map that given P_{gen} selects the engine operating point that maximizes the combined engine-generator efficiency and provides the desired generator power

$$[\omega_{\text{eng}}\ \tau_{\text{eng}}] = \gamma^*(P_{\text{gen}}).$$

For the engine to actually operate along γ^*, the generator power transitions need to be "smoothed" by using the battery as a constrained energy buffer, as experimentally demonstrated in [38]. MPC is a natural candidate for such a control strategy since it is capable of enforcing the constraints on battery charge, battery power, and the tradeoff between power smoothing and charge regulation. However, in [38], it was also remarked that with deterministic MPC a short horizon needs to be used because an increase in the prediction horizon resulted in increased computational load without performance gain, the latter due to the absence of reliable information on the

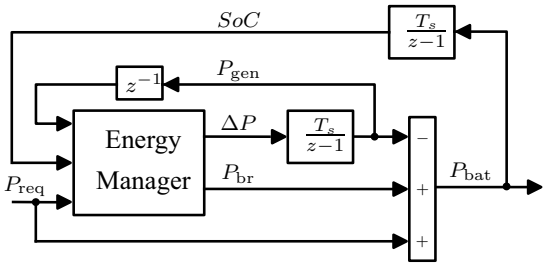

Fig. 5. Control-oriented model of SHEV for energy management.

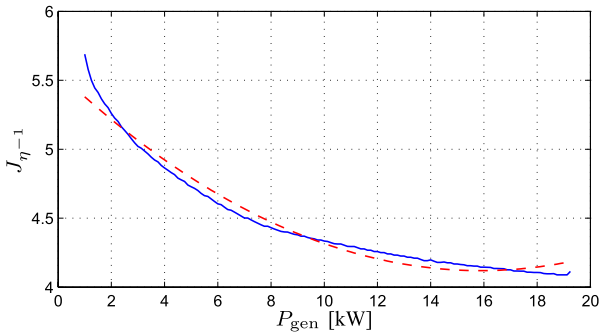

Fig. 6. Quadratic approximation $J_{\eta^{-1}}$ (red dashed line) of the inverted optimal efficiency curve (blue solid line) as function of generator power.

future driver power request. Here we show that, by learning the driver behavior in terms of power request and by using such information in the MPC strategy, the SMPCL approach developed in this paper can obtain further improvements in fuel economy.

To design the SMPCL controller, we obtain a control-oriented prediction model from the SHEV powertrain model described in Section IV, according to schematics in Fig. 5, where

$$P_{\text{req}}(k) = \frac{1}{\eta_{\text{wh}}(k)\eta_{\text{mot}}(k)} P_{\text{wh}}(k) \tag{19}$$

is the power request at time k as seen from the DC bus, and

$$\Delta P(k) = P_{\text{gen}}(k) - P_{\text{gen}}(k-1) \tag{20}$$

is the step-to-step generator power variation. The power balance at the DC bus requires that

$$P_{\text{bat}}(k) = P_{\text{req}}(k) - P_{\text{gen}}(k) + P_{\text{br}}(k), \ \forall k \in \mathbb{Z}_{0+} \tag{21}$$

where $P_{\text{br}}(k) \geq 0$ is the power drained by conventional friction brakes (in case regenerative braking is not sufficient).

In [38], it was shown that with the state of charge (SoC) maintained in a 40%–60% range, an integrator model for the battery dynamics is appropriate for use as a prediction model. Thus, the battery state of charge is normalized with respect to the battery capacity (SoC$(k) = 1$ fully charged, SoC$(k) = 0$ fully discharged), and its dynamics are modeled as

$$\text{SoC}(k+1) = \text{SoC}(k) - \kappa T_s P_{\text{bat}}(k) \tag{22}$$

where $T_s = 1$s is the sampling period and $\kappa > 0$ is a scalar parameter identified from simulation data of the model in Section IV-B. By collecting (20)–(22), the powertrain dynamics for SHEV energy management is formulated as the linear system (9), where $x(k) = [\text{SoC}(k) \ P_{\text{gen}}(k-1)]'$, $u(k) = [\Delta P(k) \ P_{\text{br}}(k)]'$, $w(k) = P_{\text{req}}(k)$, $y(k) = P_{\text{bat}}(k)$, and

$$\begin{aligned} A &= \begin{bmatrix} 1 & \kappa T_s \\ 0 & 1 \end{bmatrix}, \ B_1 = \begin{bmatrix} \kappa T_s & -\kappa T_s \\ 1 & 0 \end{bmatrix}, \ B_2 = \begin{bmatrix} -\kappa T_s \\ 0 \end{bmatrix} \\ C &= [\,0 \ -1\,], \quad D_1 = [\,-1 \ 1\,], \quad D_2 = 1. \end{aligned} \tag{23}$$

To guarantee a prolonged battery life and to enforce the operating ranges of powertrain components and electromechanical limitations, the state, input and output vectors of system (9), (23) are subject to constraints (11), where

$$\mathcal{X} \triangleq \{x : \underline{\text{SoC}} \leq [x]_1 \leq \overline{\text{SoC}}, \ 0 \leq [x]_2 \leq \overline{P}_{\text{mec}}\} \tag{24a}$$

$$\mathcal{U} \triangleq \{u : \underline{\Delta P} \leq [u]_1 \leq \overline{\Delta P}, \ [u]_2 \geq 0\} \tag{24b}$$

$$\mathcal{Y} \triangleq \{y : \underline{P}_{\text{bat}} \leq y \leq \overline{P}_{\text{bat}}\} \tag{24c}$$

where $\underline{\text{SoC}} = 0.4$, $\overline{\text{SoC}} = 0.6$, $\overline{P}_{\text{mec}} = 20$ kW, $\overline{\Delta P} = -\underline{\Delta P} = 1$ kW, and $\overline{P}_{\text{bat}} = -\underline{P}_{\text{bat}} = 40$ kW.

In (23), ΔP and P_{br} are commanded by the energy management system, while $w = P_{\text{req}}$ is commanded by the driver and thus modeled as the Markov chain (2), according to what discussed in Section II-A. The use of Markov chains to model the driver power request has been applied also in [20] and [21], where real-time learning was, however, not considered.

The SMPCL cost function needs to account for three terms: 1) the power smoothing effect; 2) the battery state of charge regulation; and 3) the steady-state efficiency for the chosen engine-generator power. To account for 3), we construct a quadratic approximation of the inverse of the engine-generator efficiency on the optimal efficiency curve

$$J_{\eta^{-1}}(P_{\text{gen}}) = \phi(P_{\text{gen}} - P^*_{\text{gen}})^2 + \gamma \tag{25}$$

obtaining a sufficient approximation, as shown in Fig. 6.

The SMPCL cost function is implemented by (13a), where

$$x_{\text{ref}} = \begin{bmatrix} \text{SoC}_{\text{ref}} \\ P_{\text{ref}} \end{bmatrix}, \ Q = \begin{bmatrix} Q_{\text{SoC}} & 0 \\ 0 & Q_J \phi \end{bmatrix}, \ R = \begin{bmatrix} R_{\Delta P} & 0 \\ 0 & R_{\text{br}} \end{bmatrix} \tag{26}$$

where $\text{SoC}_{\text{ref}} = 0.5$ is the reference state of charge, $P_{\text{ref}} = P^*_{\text{gen}}$ is the engine-generator (absolute) maximum efficiency power, $Q_{\text{SoC}} > 0$ penalizes deviations from battery state of charge setpoint, $Q_J > 0$ pushes the engine to operate close to maximum efficiency power, $R_{\Delta P} > 0$ enforces smooth mechanical power variations, and $R_{\text{br}} > 0$ penalizes the use of friction brakes. After calibration through multiple simulations, the values of the weights are set to $Q_{\text{SoC}} = 10^2$, $Q_J = 10\phi$, $R_{\Delta P} = 1$, and $R_{\text{br}} = 10^3$. The constraints on ΔP are softened [8], so that problem (13) for the SHEV energy management is always feasible.

In addition, we define an engine shutdown threshold P_{dn}, so that if $P_{\text{gen}}(k) < P_{\text{dn}}$, the powertrain operates in purely electric mode. To avoid conflict with the objective of maximizing the engine efficiency, we vary the setpoint on the generator power P_{ref} by defining a threshold P_{th} and imposing that if $P_{\text{req}}(k) < P_{\text{th}}$, $P_{\text{ref}}(k) = 0$, while $P_{\text{ref}}(k) = P^*_{\text{gen}}$ otherwise. For the designed controller, we implemented $P_{\text{th}} = 5$ kW and $P_{\text{dn}} = 0.5$ kW.

The optimization tree defining the optimal control problem is generated with $n_{\max} = 100$ nodes. The value for $n_{\max}$ has been chosen as a tradeoff between computational complexity and prediction capability. For predicting P_{req}, a Markov chain

with $s = 16$ states is used. The Markov Chain transition probabilities are initialized by (4), using power request profiles (P_{req}) from standard driving cycles (NEDC, FTP-75, FTP-Highway, Mode 10-15), and online learning (7), (8) is executed with $\bar{\lambda} = 0.01$, which implies that 99.2% of the memory vanishes in approximately 8 min. In the SMPCL problem, the prediction of P_{req} implies the prediction of P_{ref} as well, which then varies along the prediction horizon, so that the disturbance modeled by the Markov chain is actually vector valued.

VI. Simulation Results on Standard and Real-World Driving Cycles

The SMPCL controller for energy management designed in Section V is connected to the SHEV QSS simulation model described in Section IV for closed loop simulations on several driving cycles. Indeed, the simulation model and the MPC prediction model are not the same. For instance, the simulation model of the battery is nonlinear and the inverse efficiency function in (25) is only an approximation of the actual inverse efficiency. Thus, the closed-loop simulations also assess the SMPCL robustness to modeling errors and uncertainties.

According to what we described in Section IV, the power request, that is the main disturbance for the energy management controller, is obtained from the velocity profile of the cycles. We have used standard driving cycles where the velocity profile is specified, and real-word driving cycles where velocity data have been recorded by an acquisition system during regular driving. In what follows we compare the SMPCL controller with a prescient MPC (PMPC) that knows the future power request along the entire prediction horizon, and with a frozen-time MPC (FTMPC) where the power request is assumed constant over the prediction horizon. A FTMPC solution has been tested experimentally on a fully functional vehicle in [38], and it has shown significant fuel economy improvement with respect to baseline strategies. The cost functions of PMPC and FTMPC are the same as the one of SMPCL, and their predictions horizons are set to $n_{\max}$.

A. Simulations on Standard Driving Cycles

We report simulations on three standard driving cycles, NEDC, FTP 75, and FTP-Highway. Even though fuel consumption is not explicitly minimized, the cost function (13a), with weights as in (26), forces the engine to operate close to its optimal operation point by using the battery power for smoothing the aggressive engine power transients that are inefficient. This results in improved fuel economy.

The results of SMPCL, FTMPC, and PMPC are shown in Table I, in terms of norm of variations of generator power (i.e., engine operation smoothness), fuel consumption, battery charge difference (ΔSoC) between the end and the beginning of the driving cycle, equivalent fuel consumption, and equivalent fuel consumption improvement with respect to FTMPC. The equivalent fuel consumption is computed by converting ΔSoC into fuel and adding it to the fuel consumption. Specifically the equivalent fuel consumption $E_{\text{D,C}}$ is

$$E_{\text{D,C}} = F_{\text{D,C}} - \alpha_{\text{D}} \Delta \text{SoC}_{\text{D,C}} \tag{27}$$

TABLE I
SHEV Energy Management Simulation Results on Standard Driving Cycles

	$\|\Delta P\|$	Fuel cons.	ΔSoC gain/loss	Equiv. fuel cons.	impr. wrt FTMPC
NEDC					
FTMPC	37.57kW	204g	0.35%	197g	–
→ SMPCL	16.28kW	166g	-0.82%	184g	6.45%
PMPC	15.25kW	196g	0.84%	177g	9.97%
FTP-75					
FTMPC	89.28kW	348g	0.64%	334g	–
→ SMPCL	26.07kW	292g	0.08%	290g	13.10%
PMPC	32.30kW	307g	0.89%	286g	14.20%
FTP-Highway					
FTMPC	39.33kW	267g	0.64%	253g	–
→ SMPCL	16.84kW	281g	2.12%	235g	7.26%
PMPC	16.33kW	254g	0.91%	234g	7.32%

where $F_{\text{D,C}}$ and $\Delta\text{SoC}_{\text{D,C}}$ are the fuel consumption and the difference of SoC from initial condition at the end of the cycle D obtained with controller C, respectively. In (27), $\alpha_{\text{D}} \in \mathbb{R}_+$ is the cycle-dependent coefficient that maps battery charge into fuel, computed as

$$\alpha_{\text{D}} = \frac{F_{\text{D,PMPC}}}{\beta_{\text{D}} + \Delta\text{SoC}_{\text{D,PMPC}}} \tag{28}$$

where β_{D} is the battery consumption obtained on cycle D when no mechanical power is provided by the ICE, i.e., $P_{\text{req}}(k) = P_{\text{bat}}(k)$, for all $k \in \mathbb{Z}_{0+}$. For testing the SMPCL algorithm, the Markov chain is initialized by batch estimation (4) using data from four standard driving cycles (FTP-75, FTP-Highway, NEDC, Mode10-15), then each cycle is run twice before measuring the performance, so that the controller has the possibility of learning the pattern of the cycle. Plots related to NEDC, FTP 75, and FTP-Highway are reported in Figs. 7–9, respectively.

The results show that SMPCL improves fuel economy with respect to FTMPC by taking advantage of the learned power request patterns to perform more accurate predictions. The advantage of the SMPCL strategy over FTMPC is smaller for the NEDC cycle. This is due to the "piecewise linear" nature of the NEDC velocity profile, which makes the prediction of the power request often straightforward or alternatively extremely difficult.

Larger fuel economy improvements with SMPCL over FTMPC are noticeable in FTP-75 and FTP-Highway cycles. In these cases, the vehicle velocity, and as a consequence the power request, has a more varied pattern that cannot be predicted by FTMPC, while its statistics is learned and exploited by SMPCL.

B. Simulations on Real-World (Off-Cycle) Driving

SMPCL appears capable of outperforming standard deterministic MPC and gets close to PMPC when tested on standard driving cycles. However, we want to verify the same capabilities in off-cycle driving, that is, in real-world

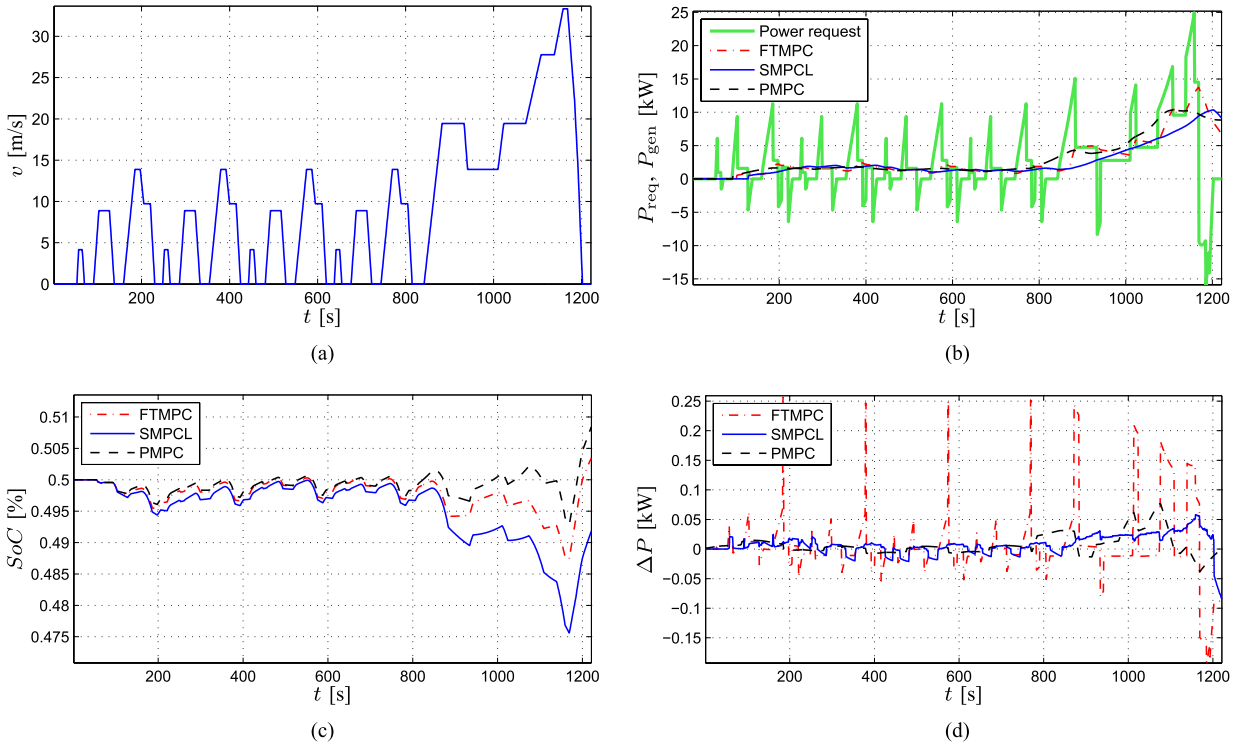

Fig. 7. SMCL for HEV energy management: results on NEDC driving cycle. (a) Vehicle velocity. (b) Driver power request and generator power. (c) Battery state of charge. (d) Generator power variation.

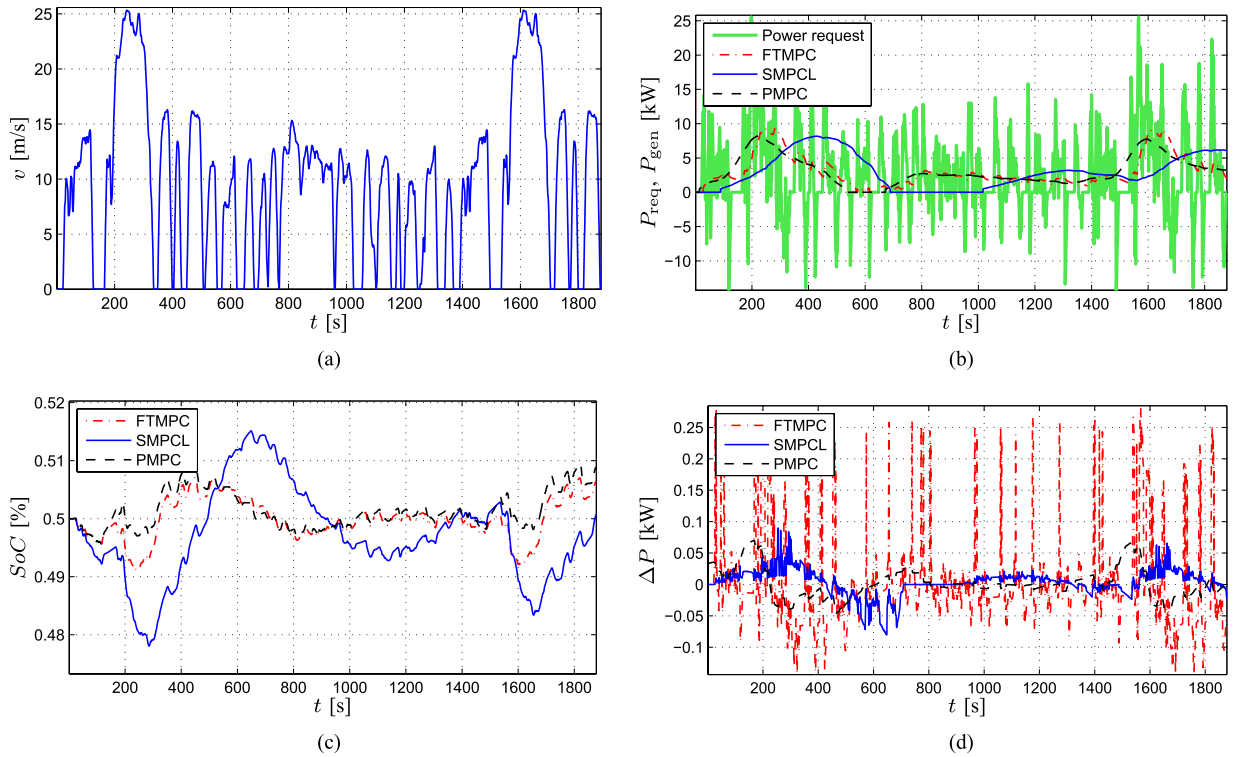

Fig. 8. SMCL for HEV energy management: results on FTP-75 driving cycle. (a) Vehicle velocity. (b) Driver power request and generator power. (c) Battery state of charge. (d) Generator power variation.

driving conditions, where the decision on the driving style is strongly dependent on the driver. We consider two data sets of regular urban driving with different driving styles obtained by recording data from GPS: the first (Trace 1) shows smooth accelerations and the second (Trace 2) shows steep accelerations. The acquired velocity profiles are fed into the simulation model that generates the power request, which is the stochastic disturbance for SMPCL, according to what

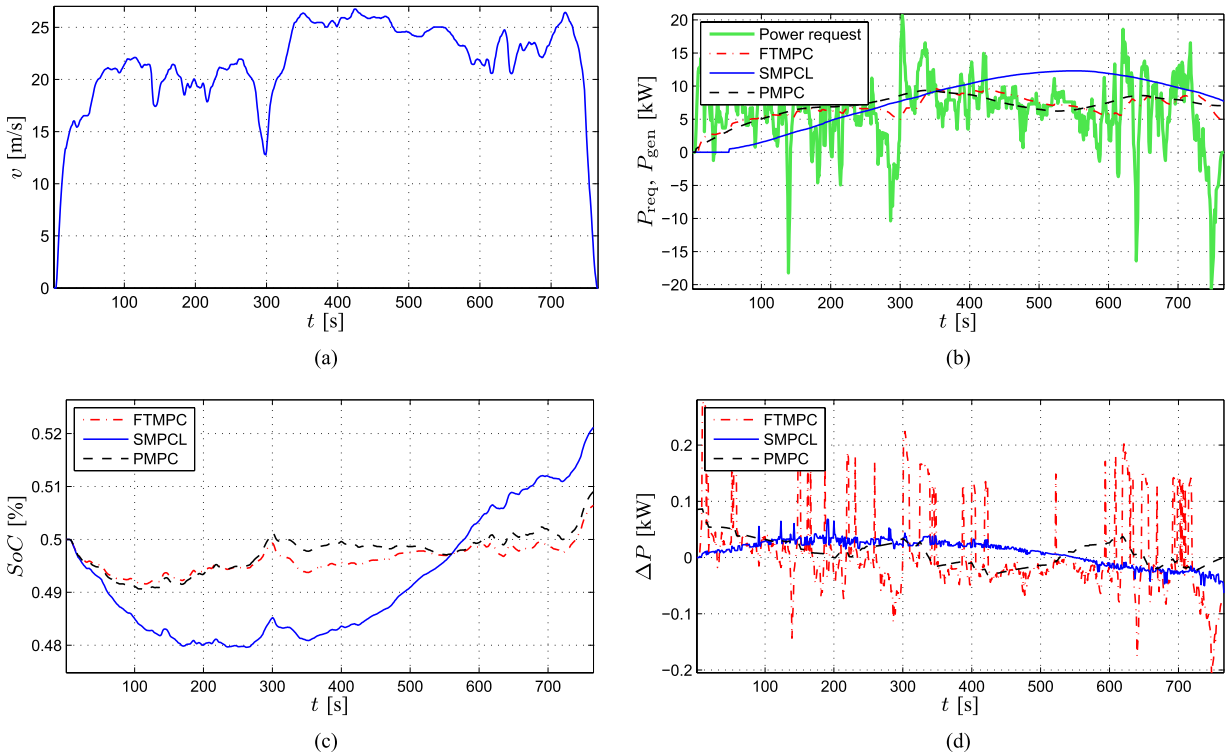

Fig. 9. SMCL for HEV energy management: results on FTP-Highway driving cycle. (a) Vehicle velocity. (b) Driver power request and generator power. (c) Battery state of charge. (d) Generator power variation.

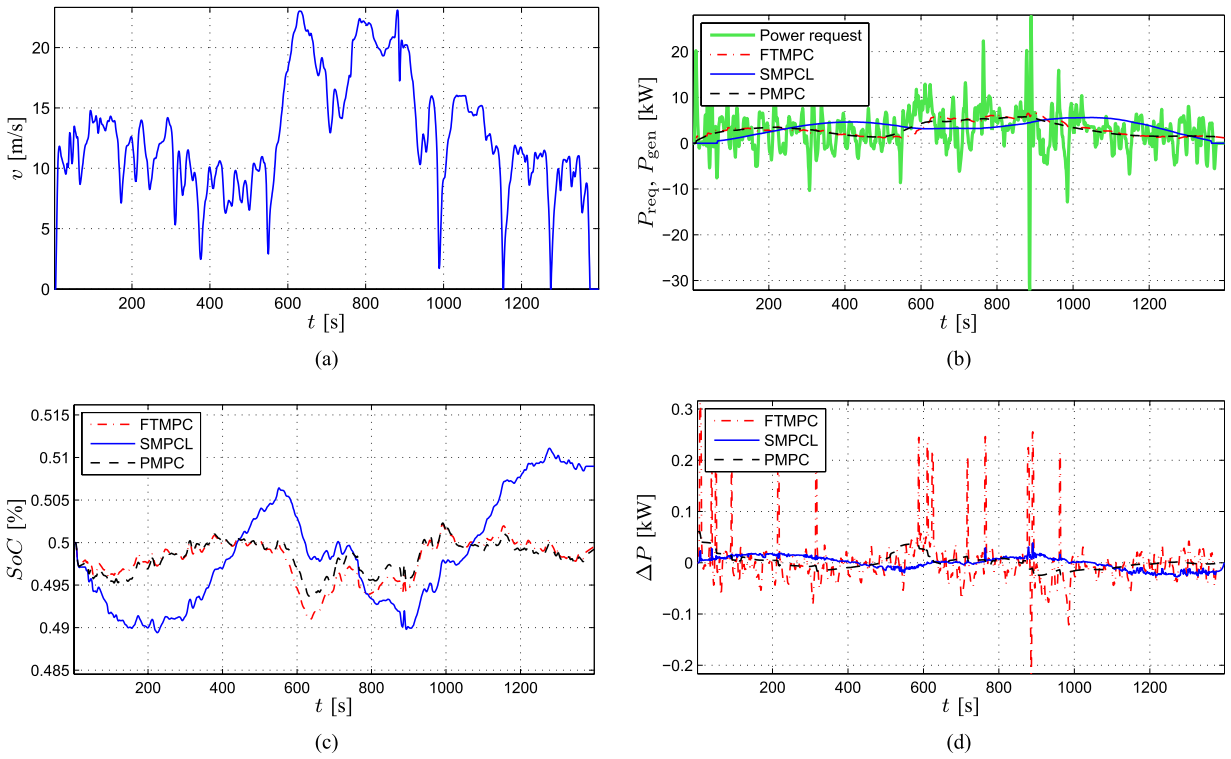

Fig. 10. SMCL for HEV energy management: results on real-world Trace 1. (a) Vehicle velocity. (b) Driver power request and generator power. (c) Battery state of charge. (d) Generator power variation.

discussed in Section V. The obtained results are reported in Table II, and shown in Figs. 10 and 11 for Trace 1 and Trace 2, respectively. The results demonstrate the capability of SMPCL to adapt to different driving styles, by learning the stochastic model of the driver and exploiting it in the construction of the scenario tree. On the considered driving routes, the fuel economy yielded by SMPCL is notably improved with respect to FTMPC and it is almost equivalent to the one obtained with

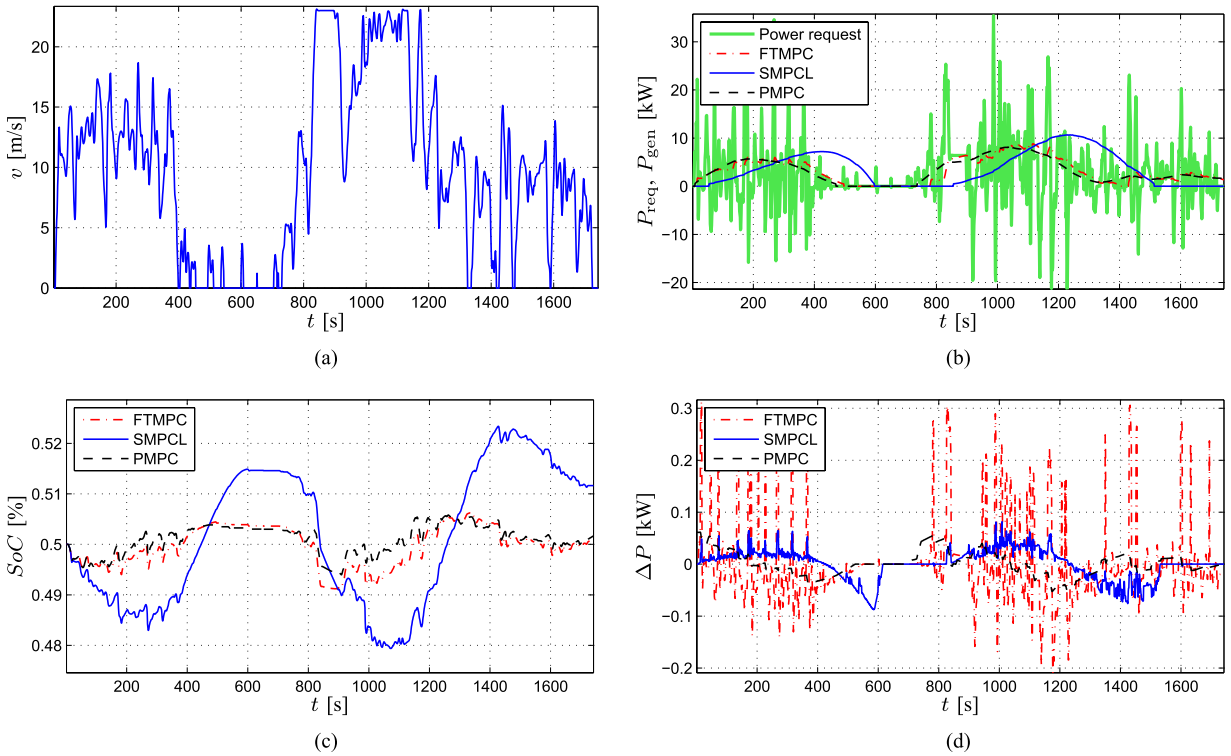

Fig. 11. SMCL for HEV energy management: results on real-world Trace 2. (a) Vehicle velocity. (b) Driver power request and generator power. (c) Battery state of charge. (d) Generator power variation.

TABLE II
SIMULATION RESULTS ON REAL-WORLD DRIVING CYCLES

	$\|\Delta P\|$	Fuel cons.	SoC gain/loss	Equiv. fuel cons.	impr. wrt FTMPC
Trace #1 - smooth accelerations					
FTMPC	37.84kW	243g	-0.05%	244g	–
→ SMPCL	14.32kW	244g	0.90%	225g	8.04%
PMPC	14.08kW	223g	-0.08%	224g	8.19%
Trace #2 - steep accelerations					
FTMPC	80.61kW	327g	0.11%	323g	–
→ SMPCL	35.74kW	320g	1.16%	287g	11.34%
PMPC	30.67kW	287g	0.17%	282g	12.73%

TABLE III
PERCENTAGE IMPROVEMENT OF SMPCL STRATEGY DUE TO ONLINE LEARNING OF THE MARKOV CHAIN

Standard cycle	Learning Improvement	Real-word Driving	Learning Improvement
NEDC	12.7%	Trace #1	1.3%
FTP-75	16.5%	Trace #2	13.4%
FTP-H.	1.1%		

PMPC that exploits full knowledge of the future power request. Also in this case, the advantages of PMPC and SMPCL are more evident in the driving profile with steeper accelerations (Trace 2), which is expected according to the power smoothing objective of the control strategy.

Finally, in Table III we provide an indication of the component of the SMPCL improvement that is exclusively due to online driver model learning. The reported percentage is the ratio of the difference between the equivalent fuel consumption of SMPCL and FTMPC and the difference between equivalent fuel consumption of SMPCL with ($\bar{\lambda} = 0.01$) and without ($\bar{\lambda} = 0$) online learning. In some cases, the benefits exclusively due to learning are small, because the initial Markov chain is already representative of the driving pattern, whereas in the case of more varied driving cycles the benefits of the learning algorithm are significant, indicating that overall learning is useful in driving conditions with complex patterns.

TABLE IV
COMPUTATION TIME OF SMPCL IN THE SHEV ENERGY MANAGEMENT SIMULATIONS

	NEDC	FTP-75	FTP-H.	Trace #1	Trace #2
Average	124 ms	56 ms	32 ms	33 ms	54 ms
Max	357 ms	339 ms	320 ms	346 ms	337 ms

C. Complexity and Computational Issues

Algorithm 2 requires the solution of (13), which is a QP with $n_u(n_{\max} - n_{\text{leaf}})$ variables and $n_{\max}(3n_x + 3n_y + 2n_u) - 2n_u n_{\text{leaf}} - 3n_y - 2$ constraints. Thus, the computational load of (13) depends also on the transition matrix $T(k)$, which determines the structure of the scenario tree at each time step k. In a case where there are few transitions with high probability, the tree will include few scenarios with long prediction horizons, and a small number of leaf nodes, which results in more variables and constraints. On the other hand, if the transitions are almost equiprobable, the tree has a large

average branching factor with more leaf nodes and fewer variables and constraints.

In Table IV, the average and maximum computational times needed to solve an instance of problem (13) are reported, as obtained from simulations on a MacBook Pro 2.7 GHz with MATLAB 7.9 and BPMPD [43] as QP solver. The NEDC cycle, having a piecewise-linear profile, results in highly diagonally dominant transition matrix $T(k)$, thus yielding a smaller set of leaf nodes in the related scenario tree and requiring more computational effort than other cycles, as expected. The requested CPU time observed in simulation (always sufficiently smaller than the sampling period $T_s = 1$ s) indicates that with relatively simple code optimizations, SMPCL execution in ECU is not impossible, especially when considering recent developments on low complexity fast QP solver MPC [44]–[46].

VII. Conclusion

We have proposed a stochastic MPC with learning approach for automotive controls that explicitly considers the driver behavior. In the proposed approach, the pattern of driver behavior is learned online in the form of Markov chains, that are subsequently used in scenario-based stochastic model predictive control. Thus, the closed-loop system adjusts to changes in driving style and different traffic conditions.

We have applied the SMPCL approach to energy management of a SHEV, where the driver model predicts the future power request that relates to the driving cycle and to the driving style. We have evaluated the SMPCL controller in simulations on standard and real-world driving profiles, and we have shown that SMPCL improves the performance of classical MPC (FTMPC), and is often close to MPC with full anticipative action (PMPC). Future research will focus on improving the flexibility of the stochastic models of the driver, and on devising quadratic programming algorithms optimized for the structure of SMPCL problems.

Acknowledgment

The authors would like to thank Dr. G. Ripaccioli for his help in collecting the experimental data used in Section VI-B.

References

[1] S. Di Cairano, A. Bemporad, I. Kolmanovsky, and D. Hrovat, "Model predictive control of magnetically actuated mass spring dampers for automotive applications," *Int. J. Control*, vol. 80, no. 11, pp. 1701–1716, 2007.

[2] P. Ortner and L. del Re, "Predictive control of a diesel engine air path," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 449–456, May 2007.

[3] P. Falcone, F. Borrelli, J. Asgari, H. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 566–580, May 2007.

[4] G. Stewart and F. Borrelli, "A model predictive control framework for industrial turbodiesel engine control," in *Proc. 47th IEEE Conf. Decision Control*, Dec. 2008, pp. 5704–5711.

[5] R. Amari, M. Alamir, and P. Tona, "Unified MPC strategy for idle-speed control, vehicle start-up and gearing applied to an automated manual transmission," in *Proc. 17th IFAC World Congr.*, 2008, pp. 7079–7085.

[6] T. Hatanaka, T. Yamada, M. Fujita, S. Morimoto, and M. Okamoto, "Explicit receding horizon control of automobiles with continuously variable transmissions," *Nonlinear Model Predict. Control*, vol. 384, pp. 561–569, Jan. 2009.

[7] S. Di Cairano and H. Tseng, "Driver-assist steering by active front steering and differential braking: Design, implementation and experimental evaluation of a switched model predictive control approach," in *Proc. 49th IEEE Conf. Decision Control*, Dec. 2010, pp. 2886–2891.

[8] S. Di Cairano, D. Yanakiev, A. Bemporad, I. V. Kolmanovsky, and D. Hrovat, "Model predictive idle speed control: Design, analysis, and experimental evaluation," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 1, pp. 84–97, Jan. 2012.

[9] S. Di Cairano, H. Tseng, D. Bernardini, and A. Bemporad, "Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1236–1248, Jul. 2013.

[10] U. Kiencke and L. Nielsen, *Automotive Control Systems for Engine, Driveline, and Vehicle*. New York, NY, USA: Springer-Verlag, 2000.

[11] L. Guzzella and A. Sciarretta, *Vehicle Propulsion Systems Introduction to Modeling and Optimization*. New York, NY, USA: Springer-Verlag, 2005.

[12] G. Burnham, J. Seo, and G. Bekey, "Identification of human driver models in car following," *IEEE Trans. Autom. Control*, vol. 19, no. 6, pp. 911–915, Dec. 1974.

[13] G. Prokop, "Modeling human vehicle driving by model predictive online optimization," *Vehicle Syst. Dyn.*, vol. 35, no. 1, pp. 19–53, 2001.

[14] C. Macadam, "Understanding and modeling the human driver," *Vehicle Syst. Dyn.*, vol. 40, nos. 1–3, pp. 101–134, 2003.

[15] A. Liu and A. Pentland, "Towards real-time recognition of driver intentions," in *Proc. IEEE Conf. Intell. Transp. Syst.*, Nov. 1997, pp. 236–241.

[16] R. Cooper, "System identification of human performance models," *IEEE Trans. Syst., Man Cybern.*, vol. 21, no. 1, pp. 244–252, Jan. 1991.

[17] U. Kiencke, R. Majjad, and S. Kramer, "Modeling and performance analysis of a hybrid driver model," *Control Eng. Pract.*, vol. 7, no. 8, pp. 985–991, 1999.

[18] G. Ripaccioli, D. Bernardini, S. Di Cairano, A. Bemporad, and I. Kolmanovsky, "A stochastic model predictive control approach for series hybrid electric vehicle power management," in *Proc. Amer. Control Conf.*, 2010, pp. 5844–5849.

[19] M. Bichi, G. Ripaccioli, S. Di Cairano, D. Bernardini, A. Bemporad, and I. Kolmanovsky, "Stochastic model predictive control with driver behavior learning for improved powertrain control," in *Proc. 49th IEEE Conf. Decision Control*, Dec. 2010, pp. 6077–6082.

[20] C. Lin, H. Peng, and J. Grizzle, "A stochastic control strategy for hybrid electric vehicles," in *Proc. Amer. Contr. Conf.*, Jul. 2004, pp. 4710–4715.

[21] I. Kolmanovsky and D. Filev, "Stochastic optimal control of systems with soft constraints and opportunities for automotive applications," in *Proc. IEEE Multiconf. Syst. Control*, Jul. 2009, pp. 1265–1270.

[22] I. Kolmanovsky, I. Siverguina, and B. Lygoe, "Optimization of powertrain operating policy for feasibility assessment and calibration: Stochastic dynamic programming approach," in *Proc. Amer. Contr. Conf.*, vol. 2. 2002, pp. 1425–1430.

[23] L. Johannesson, M. Asbogard, and B. Egardt, "Assessing the potential of predictive control for hybrid vehicle powertrains using stochastic dynamic programming," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 1, pp. 71–83, Mar. 2007.

[24] D. Wilson, R. Sharp, and S. Hassan, "The application of linear optimal control theory to the design of active automotive suspensions," *Vehicle Syst. Dyn.*, vol. 15, no. 2, pp. 105–118, 1986.

[25] D. van Hessem and O. Bosgra, "A conic reformulation of model predictive control including bounded and stochastic disturbances under state and input constraints," in *Proc. 41st IEEE Conf. Decision Control*, Dec. 2002, pp. 4643–4648.

[26] A. Bemporad and S. Di Cairano, "Optimal control of discrete hybrid stochastic automata," in *Hybrid Systems: Computation and Control*. New York, NY, USA: Springer-Verlag, 2005, pp. 151–167.

[27] P. Couchman, M. Cannon, and B. Kouvaritakis, "Stochastic MPC with inequality stability constraints," *Automatica*, vol. 42, no. 12, pp. 2169–2174, 2006.

[28] J. Primbs and C. Sung, "Stochastic receding horizon control of constrained linear systems with state and control multiplicative noise," *IEEE Trans. Autom. Control*, vol. 54, no. 2, pp. 221–230, Feb. 2009.

[29] D. Bernardini and A. Bemporad, "Stabilizing model predictive control of stochastic constrained linear systems," *IEEE Trans. Autom. Control*, vol. 57, no. 6, pp. 1468–1480, Jun. 2012.

[30] A. Bemporad and S. Di Cairano, "Model predictive control of discrete hybrid stochastic automata," *IEEE Trans. Autom. Control*, vol. 56, no. 6, pp. 1307–1321, Jun. 2011.

[31] A. Sciarretta and L. Guzzella, "Control of hybrid electric vehicles," *IEEE Control Syst. Mag.*, vol. 27, no. 2, pp. 60–67, Apr. 2007.

[32] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[33] A. Brahma, Y. Guezennec, and G. Rizzoni, "Optimal energy management in series hybrid electric vehicles," in *Proc. Amer. Control Conf.*, Sep. 2000, pp. 60–64.
[34] M. ÒKeefe and T. Markel, "Dynamic programming applied to investigate energy management strategies for a plug-in HEV," in *Proc. 22nd Int. Battery, Hybrid Fuel Cell EVS Exposit.*, 2006, pp. 1–3.
[35] C. Musardo, G. Rizzoni, and B. Staccia, "A-ECMS: An adaptive algorithm for hybrid electric vehicle energy management," in *Proc. 44th IEEE Conf. Decision Control*, Dec. 2005, pp. 1816–1823.
[36] G. Ripaccioli, A. Bemporad, F. Assadian, C. Dextreit, S. Di Cairano, and I. Kolmanovsky, "Hybrid modeling, identification, and predictive control: An application to hybrid electric vehicle energy management," in *Proc. Hybrid Syst., Comput. Control*, 2009, pp. 321–335.
[37] H. Borhan, A. Vahidi, A. Phillips, M. Kuang, I. Kolmanovsky, and S. Di Cairano, "MPC-based energy management of a power-split hybrid electric vehicle," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 3, pp. 593–603, May 2012.
[38] S. Di Cairano, W. Liang, I. V. Kolmanovsky, M. L. Kuang, and A. M. Phillips, "Power smoothing energy management and its application to a series hybrid powertrain," *IEEE Trans. Control Syst. Technol.* [Online]. Available: http://dx.doi.org/10.1109/TCST.2012.2218656
[39] S. J. Moura, H. K. Fathy, D. S. Callaway, and J. L. Stein, "A stochastic optimal control approach for power management in plug-in hybrid electric vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 3, pp. 545–555, May 2011.
[40] "2017 and later model year light-duty vehicle greenhouse gas emissions and corporate average fuel economy," *Federal Registrar*, vol. 77, no. 199, pp. 62623–63200, Oct. 2012.
[41] S. Di Cairano, W. Liang, I. Kolmanovsky, M. Kuang, and A. Phillips, "Engine power smoothing energy management strategy for a series hybrid electric vehicle," in *Proc. Amer. Control Conf.*, 2011, pp. 2101–2106.
[42] L. Guzzella and A. Amstutz, "QSS-toolbox manual," in *Proc. ETH-IMRT*, Jun. 2005.
[43] C. Mészáros, "The BPMPD interior point solver for convex quadratic problems," *Optim. Methods Softw.*, vol. 11, nos. 1–4, pp. 431–449, 1999.
[44] S. Richter, C. N. Jones, and M. Morari, "Computational complexity certification for real-time MPC with input constraints based on the fast gradient method," *IEEE Trans. Autom. Control*, vol. 57, no. 6, pp. 1391–1403, Jun. 2012.
[45] P. Patrinos and A. Bemporad, "An accelerated dual gradient-projection algorithm for linear model predictive control," in *Proc. 51st IEEE Conf. Decision Control*, Dec. 2012, pp. 662–667.
[46] S. Di Cairano, M. Brand, and S. Bortoff, "Projection-free parallel quadratic programming for linear model predictive control," *Int. J. Control*, in press, available at www.tandfonline.com.

Stefano Di Cairano (M'08) received the master's (Laurea) degree in computer engineering and the Ph.D. degree in information engineering from the University of Siena, Siena, Italy, in 2004 and 2008, respectively.

He was granted the International Current Optics of Doctoral Studies in Hybrid Control for Complex Distributed and Heterogeneous Embedded Systems. He was a Visiting Student with the Technical University of Denmark, Lyngby, Denmark, from 2002 to 2003, and the California Institute of Technology, Pasadena, CA, USA, from 2006 to 2007. From 2008 to 2011, he was a Senior Researcher and Technical Expert with Powertrain Control R&A, Ford Research and Adv. Engineering, Dearborn, MI, USA. Since 2011, he has been a Principal Member of the Research Staff in mechatronics with Mitsubishi Electric Research Laboratory, Cambridge, MA, USA. His research is on advanced control strategies for complex mechatronic systems, in automotive, factory automation, and aerospace. His current research interests include predictive control, constrained control, networked control systems, hybrid systems, optimization, automotive, aerospace, and factory automation.

Dr. Di Cairano has been a Chair of the IEEE CSS Technical Committee on Automotive Controls since 2011 and a member of the IEEE CSS Conference Editorial Board. Since 2013, he has been an Associate Editor of the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY.

Daniele Bernardini was born in 1982. He received the master's degree in computer engineering and the Ph.D. degree in information engineering from the University of Siena, Siena, Italy, in 2007 and 2011, respectively.

He was with the Department of Electrical Engineering, Stanford University, Stanford, CA, USA, in 2010. In 2011, he was with the Department of Mechanical and Structural Engineering, University of Trento, Trento, Italy. In October 2011, he joined the IMT Institute for Advanced Studies Lucca, Lucca, Italy, where he is a Post-Doctoral Research Fellow. His current research interests include model predictive control, stochastic control, networked control systems, hybrid systems, and their application to problems in automotive, aerospace, and energy domains.

Alberto Bemporad (F'10) received the master's degree in electrical engineering and the Ph.D. degree in control engineering from the University of Florence, Florence, Italy, in 1993 and 1997, respectively.

He was with the Center for Robotics and Automation, Department of Systems Science and Mathematics, Washington University, St. Louis, DC, USA, from 1996 to 1997, as a Visiting Researcher. From 1997 to 1999, he held a postdoctoral position with the Automatic Control Laboratory, ETH Zurich, Zurich, Switzerland, where he collaborated as a Senior Researcher from 2000 to 2002. From 1999 to 2009, he was with the Department of Information Engineering, University of Siena, Siena, Italy, as an Associate Professor, in 2005. From 2010 to 2011, he was with the Department of Mechanical and Structural Engineering, University of Trento, Trento, Italy. In 2011, he joined as a Full Professor with the IMT Institute for Advanced Studies, Lucca, Italy, where he became the Director in 2012. He has co-founded ODYS S.r.l., Lucca, a spinoff company of IMT Lucca. He is the author or co-author of various MATLAB toolboxes for model predictive control design, including the Model Predictive Control Toolbox (The Mathworks, Inc.). He has published more than 250 papers in the areas of model predictive control, hybrid systems, automotive control, multiparametric optimization, computational geometry, robotics, and finance.

Dr. Bemporad was an Associate Editor of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL from 2001 to 2004 and a Chair of the Technical Committee on Hybrid Systems of the IEEE Control Systems Society from 2002 to 2010.

Ilya V. Kolmanovsky (F'08) received the M.S. and Ph.D. degrees in aerospace engineering and the M.A. degree in mathematics from the University of Michigan, Ann Arbor, MI, USA, in 1993, 1995, and 1995, respectively.

He is currently a Professor with the Department of Aerospace Engineering, University of Michigan. Prior to joining the University of Michigan, he was with Ford Research and Advanced Engineering, Dearborn, MI, USA. His current research interests include control theory for systems with state and control constraints, control of automotive and aerospace propulsion systems, and spacecraft control applications.

Dr. Kolmanovsky is a past recipient of the Donald P. Eckman Award of American Automatic Control Council, the IEEE Transactions on Control Systems Technology Outstanding Paper Award, and several Ford Research and Advanced Engineering Technical Achievement, Innovation and Publication Awards.