Power Control Optimization for Autonomous Hybrid Electric Vehicles with Flexible Driveline Torque Demand

Mohammadali Kargar and Xingyong Song

***Abstract*—In the study of powertrain controls optimization for autonomous vehicles, the vehicle's external dynamics and powertrain dynamics are often treated separately, where one of the unique features is usually neglected. This uniqueness, which is referred to as *flexible power demand*, states that the powertrain control does not need to exactly meet the power requested by the vehicle motion controller at every moment. In this study, a method based on the Approximate Dynamic Programming (ADP) is explored to design the powertrain controller, where the flexibility in power demand is incorporated in the ADP framework. At last, a case study is shown to examine the efficacy of the explored method.**

***Index Terms*—Autonomous vehicles; Hybrid electric vehicles; Energy Management; Approximate Dynamic Programming.**

I. Introduction

Passenger vehicles consume almost 65 percent of U.S transportation fuels [1]. The need for lower fuel consumption has been giving momentum to automotive technologies, particularly in the area of powertrain hybridization and autonomy. While there are important benefits from each of the two technologies individually, combining these two trends together also has significant potential of further improving the vehicle fuel economy.

An autonomous HEV has two levels of control. The upper-level controller plans the external dynamics of the vehicle. The lower-level controller decides how to supply the requested driving power efficiently from the power sources. Recently, some researchers have applied the emerging idea of flexible torque request [2]. In this approach, the lower-level controller does not need to exactly meet the power required by the upper level at every moment. Instead, it may have some deviations. The methodology they have used in [2] is based on Pontryagin's minimum principle (PMP). PMP solutions are dependent on the initial condition. Thus, each time the initial condition changes, a new optimization is required.

In this paper, a customized ADP method is explored for the first time to tackle the powertrain optimization problem with flexible power demand. ADP provides solutions independent of the initial condition of the system. However, due to the non-quadratic nature of the cost function and the non-affine structure of the powertrain dynamics, the use of conventional standard ADP methods can be challenging. Therefore, a customization to the ADP method is proposed to enable its application to non-quadratic cost functions with non-affine dynamics and nonlinear constraints on control inputs.

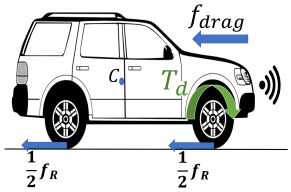

Fig. 1: Free Body Diagram of the Vehicle.

The remainder of this paper is organized as follows. The HEV modeling is discussed in Section II. Section III presents the power management with flexible power demand. ADP framework is introduced in Section IV, followed by a numerical example in Section V. At last, concluding remarks are given in Section VI.

II. HEV Dynamics

A. Upper-Level Dynamics

Fig. 1 shows the free body diagram of a vehicle in a straight path in an inertial frame. Let x be the longitudinal position of the center of mass of the vehicle C. Also, let v be the longitudinal velocity of the vehicle. The external kinematics and dynamics of the vehicle can be written as below:

$$\dot{x}(t) = v(t), \qquad x(0) = x_0 \quad (1)$$

$$\dot{v}(t) = \frac{1}{m}\left[\frac{1}{r}T_d(t) - f_{drag} - f_R\right], \qquad v(0) = v_0$$

$$f_{drag} = \frac{1}{2}\rho C_{drag} A_f v(t)^2, \quad f_R = \mu_R mg \quad (2)$$

where m, r, T_d, A_f, f_{drag}, f_R, and μ_R are the vehicle's mass, wheel's radius, driveline torque, effective frontal area, air drag force, rolling resistance force and rolling resistance coefficient, respectively. Besides, ρ and C_{drag} denote the air density and the coefficient of drag, respectively.

B. Lower-Level Dynamics

1) Powertrain Dynamics: The hybrid powertrain considered in this study is shown in (Fig. 2). Neglecting the

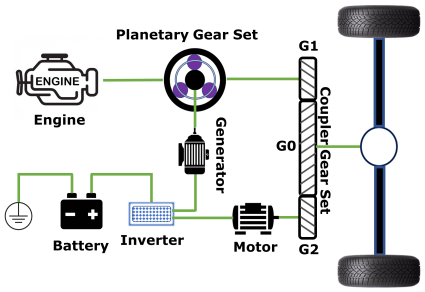

Fig. 2: Power-split Powertrain Schematic.

inertia of the moving parts in the powertrain, the power balance at the inverter reveals battery power $P_{batt}(t)$ as:

$$P_{batt}(t) = \mu_m^{k_m} T_m(t)\omega_m(t) + \mu_g^{k_g} T_g(t)\omega_g(t) \quad (3)$$

where torque and angular velocity of the motor and the generator are depicted by T_m, ω_m, T_g, and ω_g, respectively. The efficiency coefficients of the generator and the motor are shown by μ_g and μ_m when they are functioning as generators, respectively. Also, k_m and k_g are equal to 1 if their corresponding electrical machines operate to generate electricity and equal to -1 otherwise.

2) Battery Dynamics: An equivalent circuit model given in [3] is used to model the dynamics of the battery:

$$\dot{SoC}(t) = -\frac{V_{batt} - \sqrt{V_{batt}^2 - 4R_{batt}P_{batt}(t)}}{2R_{batt}Q_{batt}}, \quad SoC(0) = SoC_0 \quad (4)$$

where SoC is the state of the charge of the battery. Also, V_{batt}, R_{batt}, and Q_{batt} denote the battery's open-circuit voltage, internal resistance, and capacitance, respectively.

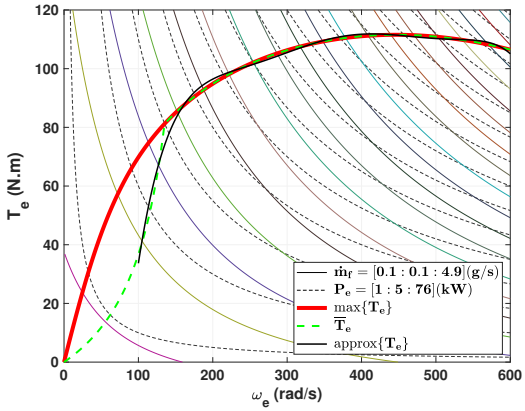

Fig. 3: Engine Map [3].

3) Fuel Consumption Dynamics: The fuel consumption dynamics are generally governed by a function Γ whose inputs are engine's angular velocity, ω_e, and engine's torque, T_e, and quantifies the fuel consumption rate as depicted by Eq. (5).

$$\dot{m}_{fuel} = \Gamma(\omega_e, T_e) \quad (5)$$

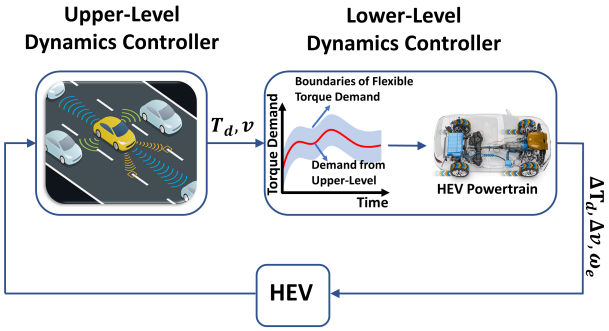

Fig. 4: Energy Management Hierarchy with Flexible Torque Demand.

An engine map generated from the experimental data is usually used to capture this mapping, as shown in Fig. 3. Since we have two degrees of freedom in the powertrain, it is a valid assumption [3], [4] that for any engine power $P_e(t) \triangleq T_e(t)\omega_e(t)$, the solution pair $(T_e(t), \omega_e(t))$ will be such that the engine power lies on its most efficient point on the engine map for which $T_{e,opt}$ is mathematically approximated by the equation below (solid black line):

$$T_{e,opt} = (60 * atan(\frac{\omega_e}{70})) - 0.00018 * {\omega_e}^2 + 0.14 * \omega_e \quad (6)$$

4) Flexible Power Demand Dynamics: Deviation in the supplied power by the powertrain (Fig. 4) will result in deviation in the expected acceleration obtained in the upper-level controller, and thus a deviation in the velocity and displacement expected by the upper-level controller. To make sure the vehicle reaches its destination, the velocity and the position deviations are only allowed during the intermediate time instants. In other words, it means that the longitudinal displacement and velocity plus driveline torque demand might be different from their value expected by the upper-level controller at any intermediate instant, but the deviation in the longitudinal displacement and velocity must vanish as time reaches the time horizon.

Let $\tilde{T}_d$, $\tilde{x}$, and $\tilde{v}$ denote the flexible driveline torque, longitudinal displacement, and velocity, respectively. Rewriting the external dynamics of the vehicle considering the flexibilities yields:

$$\dot{\tilde{x}}(t) = \tilde{v}(t), \quad \tilde{x}(0) = x(0) = x_0 \quad (7)$$

$$\dot{\tilde{v}}(t) = \frac{1}{m}\left[\frac{1}{r}\tilde{T}_d(t) - f_{drag} - f_R\right], \tilde{v}(0) = v(0) = v_0$$

$$f_{drag} = \frac{1}{2}\rho C_{drag} A_f \tilde{v}(t)^2. \quad (8)$$

Note that $\tilde{x}(0) = x(0)$ and $\tilde{v}(0) = v(0)$ come from the fact that at the beginning of the driving cycle, the lower-level controller starts with the same longitudinal displacement and velocity resulted from the upper level.

Let define $\Delta x \triangleq \tilde{x} - x$, $\Delta v \triangleq \tilde{v} - v$, and $\Delta T_d \triangleq \tilde{T}_d - T_d$. Therefore, considering (1), (2), (7), (8), one can find:

$$\Delta \dot{x}(t) = \Delta v(t), \qquad \Delta x(0) = 0 \tag{9}$$

$$\begin{aligned} \Delta \dot{v}(t) = \frac{1}{m}[-\frac{1}{2}\rho C_{drag} A_f \Delta v(t)(2v(t) \\ + \Delta v(t)) + \frac{1}{r}\Delta T_d(t)], \qquad \Delta v(0) = 0. \end{aligned} \tag{10}$$

III. Problem Formulation

To keep track of the SoC of the battery and the deviations in the longitudinal displacement and the velocity, the following state vector is defined:

$$\mathbf{X}(t) \triangleq [\Delta x(t), \Delta v(t), SoC(t)]^T. \tag{11}$$

The input vector is also defined as:

$$\mathbf{U}(t) \triangleq [\omega_e(t), T_e(t), \Delta T_d(t)]^T. \tag{12}$$

Using (6), the input vector can be reduced to:

$$\mathbf{U}(t) = [\omega_e(t), \Delta T_d(t)]^T. \tag{13}$$

Considering Eqs. (3), (4), (9), and (10), the system dynamics can be summarized to:

$$\dot{\mathbf{X}}(t) = \mathscr{F}(\mathbf{X}(t), \mathbf{U}(t)), \qquad \mathbf{X}(0) = \mathbf{X}_0 \tag{14}$$

where $\mathscr{F}$ is defined as below:

$$\mathscr{F} \triangleq \begin{bmatrix} \Delta v(t) \\ \frac{1}{m}\left[-\frac{1}{2}\rho C_{drag} A_f \Delta v(t)(2v(t) + \Delta v(t)) + \frac{1}{r}\Delta T_d(t)\right] \\ -\frac{V_{batt} - \sqrt{V_{batt}^2 - 4R_{batt}P_{batt}(t)}}{2R_{batt}Q_{batt}} \end{bmatrix} \tag{15}$$

In order to solve the optimization problem, we first need to define a cost function. Next, we try to find the series of inputs that make it minimum. The cost function J_c in its most general form can be defined as below:

$$J_c(\mathbf{X}(0), \mathbf{U}(0)) = \int_0^{t_f} \lambda(\mathbf{X}(t), \mathbf{U}(t))\ dt + \psi(\mathbf{X}(t_f)) \tag{16}$$

where t_f is the final time, and $\lambda(\mathbf{X}(t), \mathbf{U}(t))$ is the cost of intermediate states and inputs. Note that $\mathbf{X}(t_f)$ represents the final state vector and $\psi : \mathbb{R}^3 \to \mathbb{R}_+$ is called the penalizing function which is a design parameter chosen as a nonnegative function. This function is used to ensure the system will reach the desired terminal point $\mathbf{X}_{des}(t_f)$ by penalizing the state vectors far from $\mathbf{X}_{des}(t_f)$.

In this study, to minimize the fuel consumption during the drive cycle, the intermediate cost is set to be:

$$\lambda(\mathbf{X}(t), \mathbf{U}(t)) = \dot{m}_{fuel}(t). \tag{17}$$

Denoting the discretization sample time by δt and discrete time index by k, one can discretize Eq. (14) by using the Euler method:

$$\mathbf{X}(k+1) = \mathbf{X}(k) + \delta t \mathscr{F}(\mathbf{X}(k), \mathbf{U}(k)), k = 0, 1, 2, ..., N-1 \tag{18}$$

Similarly, Eq. (16) can be discretized as:

$$\begin{aligned} J(\mathbf{X}(0), \mathbf{U}(0)) &= \sum_{k=0}^{N-1} \delta t\, (\lambda(\mathbf{X}(k), \mathbf{U}(k))) + \psi(\mathbf{X}(N)) \\ &= \sum_{k=0}^{N-1} \delta t\, (\dot{m}_{fuel}(k)) + \psi(\mathbf{X}(N)) \end{aligned} \tag{19}$$

where $N \triangleq \frac{t_f}{\delta t}$.

Based on the definition of the cost function in Eq. (19), the cost-to-go $V_k(\mathbf{X}(k))$ is defined as the cost from the state $\mathbf{X}(k)$ at time index k to the end of the time horizon and is equal to:

$$\begin{aligned} V_k(\mathbf{X}(k)) &= \sum_{\tau=k}^{N-1} \delta t\, (\dot{m}_{fuel}(\tau)) + \psi(\mathbf{X}(N)) \\ &= \delta t\, (\dot{m}_{fuel}(k)) + \sum_{\tau=k+1}^{N-1} \delta t\, (\dot{m}_{fuel}(\tau)) \\ &= \delta t\, (\dot{m}_{fuel}(k)) + V_{k+1}(\mathbf{X}(k+1)), \\ &\qquad k = 0, 1, 2, ..., N-1. \end{aligned} \tag{20}$$

This equation simply states that the cost of going from $\mathbf{X}(k)$ at time index k to $\mathbf{X}(N)$ is equal to the cost of going from $\mathbf{X}(k)$ to some $\mathbf{X}(k+1)$ plus the cost of going from $\mathbf{X}(k+1)$ to $\mathbf{X}(N)$. Also, note that:

$$V_N(\mathbf{X}(N)) = \psi(\mathbf{X}(N)). \tag{21}$$

Let optimal cost-to-go $V_k^*(\mathbf{X}(k))$ be the minimum cost of going from $\mathbf{X}(k)$ to $\mathbf{X}(N)$. Based on Bellman's principle of optimality [5], one can define $V_k^*(\mathbf{X}(k))$ in a recursive manner:

$$V_k^*(\mathbf{X}(k)) = \min_{\mathbf{U}(k) \in \mathbb{U}(k)} \left(\delta t\, (\dot{m}_{fuel}(k)) + V_{k+1}^*(\mathbf{X}(k+1))\right)$$

$$\text{subject to} \begin{cases} \mathbf{X}(k) \in \mathbb{X}(k) \\ P_{batt}(k) \leqslant \dfrac{V_{batt}^2}{4R_{batt}} \quad \text{for } k \neq N \end{cases} \tag{22}$$

$$V_N^*(\mathbf{X}(N)) = \psi(\mathbf{X}(N)) \tag{23}$$

The constraints mentioned in Eq. (22) are to make sure that the states and the inputs of the system remain in the feasible region. For example, in the application of the HEVs, the SoC cannot be less than zero or more than 1 or ω_e cannot be less than zero or more than its maximum (around 450 (rad/s)). In addition, the last constraint in Eq. (22) is introduced to make sure the SoC remains a real number.

Let $(\mathbf{X}_{min}(k),\ \mathbf{X}_{max}(k))$, and $(\mathbf{U}_{min}(k),\ \mathbf{U}_{max}(k))$ denote the minimum and the maximum limits for $\mathbf{X}(k)$ and $\mathbf{U}(k)$, respectively. Then, one can define the time-variant state grid $\mathbb{X}(k)$ and input grid $\mathbb{U}(k)$ as below:

$$\mathbb{X}(k) = \{\mathbf{X}(k) \mid \mathbf{X}_{min}(k) \leqslant \mathbf{X}(k) \leqslant \mathbf{X}_{max}(k)\} \tag{24}$$

$$\mathbb{U}(k) = \{\mathbf{U}(k) \mid \mathbf{U}_{min}(k) \leqslant \mathbf{U}(k) \leqslant \mathbf{U}_{max}(k)\} \tag{25}$$

Besides, the control input by which $V_k^*(\mathbf{X}(k))$ is attained is called the optimal control input, $\mathbf{U}^*(\mathbf{X}(k))$.

IV. Solving Flexible Power Demand Optimization Using a Customized ADP

A. Optimal Control Formulation

Consider a general optimal control problem for a discrete dynamical system with p equality constraints and q inequality constraints:

$$\begin{aligned} \underset{\mathbf{U}(0),\mathbf{U}(1),...,\mathbf{U}(N-1)}{\text{minimize}} \quad & \sum_{k=0}^{N-1} \delta t\left(\lambda(\mathbf{X}(k),\mathbf{U}(k))\right)+\psi(\mathbf{X}(N)) \\ \text{subject to} \quad & \mathbf{X}(k+1)=\mathbf{X}(k)+\delta t\mathscr{F}(\mathbf{X}(k),\mathbf{U}(k)) \\ & \mathscr{f}_i(\mathbf{X}(k),\mathbf{U}(k))=0 \qquad i=1,...,p \\ & \mathscr{g}_j(\mathbf{X}(k),\mathbf{U}(k))\leqslant 0 \qquad j=1,...,q \\ & \mathbf{X}(k)\in\mathbb{X}(k), \\ & \mathbf{U}(k)\in\mathbb{U}(k). \end{aligned} \tag{26}$$

DP is an optimization method that provides globally optimal solutions to this optimization problem by breaking it into N subproblems [6]. DP solves the problem backward from the last step toward the first step. At each step, the state grid and input grid are discretized. Then for each point in the state grid, the optimal decision is found by solving the optimization problem below:

$$\begin{aligned} \underset{\mathbf{U}(k)}{\text{minimize}} \quad & \delta t\left(\lambda(\mathbf{X}(k),\mathbf{U}(k))\right)+V_{k+1}^*(\mathbf{X}(k+1)) \\ \text{subject to} \quad & \mathbf{X}(k+1)=\mathbf{X}(k)+\delta t\mathscr{F}(\mathbf{X}(k),\mathbf{U}(k)) \\ & \mathscr{f}_i(\mathbf{X}(k),\mathbf{U}(k))=0 \qquad i=1,...,p \\ & \mathscr{g}_j(\mathbf{X}(k),\mathbf{U}(k))\leqslant 0 \qquad j=1,...,q \\ & \mathbf{X}(k)\in\mathbb{X}(k), \\ & \mathbf{U}(k)\in\mathbb{U}(k). \end{aligned}$$

Note that $V_{k+1}^*(\mathbf{X}(k+1))$ is already known for all the states at time index $k+1$. Thus, at each step, for all the points in the state grid, the optimal input $\mathbf{U}^*(\mathbf{X}(k))$ and the corresponding optimal cost-to-go $V_k^*(\mathbf{X}(k))$ need to be cached in tables. Storing $\mathbf{U}^*(\mathbf{X}(k))$ and $V_k^*(\mathbf{X}(k))$ for every point in the state grid and for each step requires a vast amount of memory, which is one of the significant drawbacks of DP.

However, in the problem of minimizing fuel consumption, first, the system in Eq. (18) is highly non-affine with a non-quadratic cost function. Secondly, the inputs need to be constrained to meet the physical limitations in the engine operation, and comfortableness of the drive. Therefore, the conventional ADP methods [7], [8] in which the system is considered to be input-affine with quadratic cost function and unconstrained inputs will not work here.

In this study, a customized ADP algorithm is explored for the first time, which solves the abovementioned issues. First of all, this ADP algorithm can be applied to both control-affine and non-affine systems. Secondly, it can take care of the non-quadratic nonlinear cost function of the states and the inputs. In this algorithm, the approximated optimal cost-to-go at each step is found using deep neural networks (DNNs):

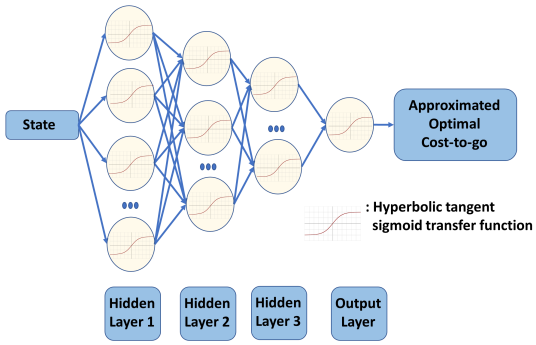

Fig. 5: Structure of the Deep Neural Network.

$$\bar{V}^*(\mathbf{X}(k))=\phi_k(\mathbf{X}(k)). \tag{27}$$

The input in this DNN is the current state $\mathbf{X}(k)$ and the output is the approximated optimal cost-to-go $\bar{V}^*(\mathbf{X}(k))$. The structure of the DNN ($\phi_k(.)$) is shown in Fig. 5. The size of hidden layers in this study by extensive research is found to be [15, 10, 5].

B. The training procedure

Similar to DP, ADP proceeds backward in time. To start the training procedure at $k=N$, one can define:

$$\bar{V}^*(\mathbf{X}(N))=\psi(\mathbf{X}(N) \tag{28}$$

and train the network accordingly. This algorithm first learns the approximated optimal cost-to-go at the final step using Eq. (23). Then, it iterates backward in time, and at each step k and for every point $\mathbf{X}(k)$ inside $\mathbb{X}(k)$, it applies the set of the discretized constrained inputs $\mathbf{U}(k)\in\mathbb{U}(k)$. Then, it finds the minimum approximated cost-to-go $\bar{V}^*(\mathbf{X}(k))$ corresponding to $\mathbf{X}(k)$, and feed the pair $(\mathbf{X}(k),\bar{V}^*(\mathbf{X}(k)))$ to the DNN to train the controller (Eq. (29)). Therefore, rather than learning the optimal control input directly using the methods in [7], [8] which necessitates the issues mentioned, in this approach, the approximated optimal cost-to-go is learned for the points in $\mathbb{X}(k)$. Then, the optimal control input is selected by comparing the approximated cost-to-go for all possible paths from the point $\mathbf{X}(k)$ using the constrained discretized inputs in $\mathbb{U}(k)$, and choosing the path with the minimum approximated cost-to-go.

$$\begin{aligned} &\bar{V}^*(\mathbf{X}(k))=\phi_k(\mathbf{X}(k)) \\ &=\begin{cases} \psi(\mathbf{X}(N), & for \quad k=N \\ \min\limits_{\mathbf{U}(k)\in\mathbb{U}(k)} \left(\delta t\left(\dot{m}_{fuel}(k)\right)+\phi_{k+1}(\mathbf{X}(k+1))\right), & \\ \qquad for \quad k\neq N. \end{cases} \end{aligned} \tag{29}$$

The cost function used in Eq. (29) for the intermediate steps can be any non-quadratic or quadratic function with respect to the states and inputs. The training algorithm is detailed in Algorithm 1. Once the learning procedure is done, it can be used in the forward simulation to find the sub-optimal control input sequence given the initial condition $\mathbf{X}(0)$. The implementation algorithm is explained in Algorithm 2.

Algorithm 1: Training Neural Network

Select a small positive number α, and a big enough integer $IterMax$;
for $k = N : \text{-}1 : 0$ **do**
  Choose h different random training samples $\mathbf{X}^l(k)$ in $\mathbb{X}(k)$ where $l \in \{1, 2, ..., h\}$;
  For each training sample $\mathbf{X}^l(k)$ find $\bar{V}_k^*(\mathbf{X}^l(k))$ using Eq. (29) ;
  Initialize $\phi_k^0(.)$ with random parameters;
  for i $= 1 : IterMax$ **do**
    Update the neural network $\phi_k^i(.)$ and find the parameters to approximate $\bar{V}_k^*(\mathbf{X}^l(k))$ using backpropagation on the entire training samples;
    if $\|\phi_k^i(.) - \phi_k^{i-1}(.)\| \leq \alpha$ **then**
      Break;
    end
  end
  $\phi_k(.) \leftarrow \phi_k^i(.)$;
end

Algorithm 2: Implementation

for $k = 0 : N - 1$ **do**
  $\mathbf{U}^*(\mathbf{X}(k)) =$

$$\underset{\mathbf{U}(k)\in\mathbf{U}(k)}{\operatorname{argmin}} \big(\delta t(\dot{m}_{fuel}(k)) + \phi_{k+1}(\mathbf{X}(k+1))\big);$$

  $\mathbf{X}(k+1) = \mathbf{X}(k) + \delta t \mathcal{F}(\mathbf{X}(k), \mathbf{U}(k))$
end

V. Simulation Results

In this section, the proposed controller is investigated over a real data set [9]. This data set has captured the external kinematics of the vehicles including their velocity, acceleration, and the headway with respect to the leading and rear vehicle. The data has been collected for a 150 meter-long section of a highway (I-35 Corridor) in the city of Austin, USA. To investigate the efficacy of the controller, a random vehicle is considered. Fig. 6 shows the velocity profile of the target vehicle. Specifications of the environment and the vehicle are presented in Table I.

Parameter	Value	Parameter	Value	Parameter	Value
m	1350 kg	r	0.28 m	μ_R	0.007
ρ	1.225 kgs/m^3	C_d	0.3	A_f	2.2 m^2
μ_g	0.9	μ_m	0.9	V_{batt}	202 V
Q_{batt}	23400 A.s	R_{batt}	0.45 Ω	Δx_{min}	-3.5 m
Δx_{max}	3.5 m	Δv_{min}	-2.5 m/s	Δv_{max}	2.5 m/s
$\Delta T_{d,min}$	-150 N.m	$\Delta T_{d,max}$	150 N.m	t_f	26.56 s
δt	0.08 s	α	0.05	$IterMax$	15

TABLE I: Environment Specifications & Vehicle System Parameters

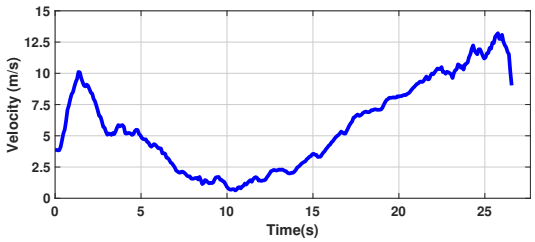

Fig. 6: Target Vehicle Velocity Profile.

The corresponding constraint Eqs. (30) to (32) are imposed on Δx, Δv, and ΔT_d to satisfy the passengers' safety and comfortableness of the drive. The maximum range of the flexibility in x & v can be limited by the relative distances & velocities with the front and rear vehicles at each step. Also, the measure of the flexibility for T_d can be constrained by the maximum torque of the engine and the motor at the current speed. For simplicity, the limitations of the flexibilities remain constant [4].

$$\Delta x_{min} \leqslant \Delta x(t) \leqslant \Delta x_{max} \tag{30}$$
$$\Delta v_{min} \leqslant \Delta v(t) \leqslant \Delta v_{max} \tag{31}$$
$$\Delta T_{d,min} \leqslant \Delta T_d(t) \leqslant \Delta T_{d,max} \tag{32}$$

The control performance of the system using the proposed control method for $\mathbf{X}(0) = [0(m), 0(m/s), 60\%]^T$ is shown in Figs. (7) - (10). For comparison, the performance of the system under the baseline approach is also presented in Figs. (8) - (10). In the baseline optimization, the flexibility in the power demand is not allowed, and the controller is responsible for supplying the exact amount of power requested by the upper-level dynamics. Thus, in the baseline optimization, the state space reduces to $[SoC]$ and the input space is $[\omega_e]$. An additional 27% fuel economy improvement was achieved by the ADP algorithm as compared with the value achieved by the baseline strategy. The summary of the results is shown in Table II.

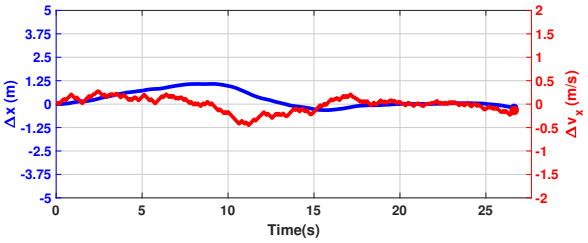

Fig. 7: Position & Velocity Deviation History of the Vehicle.

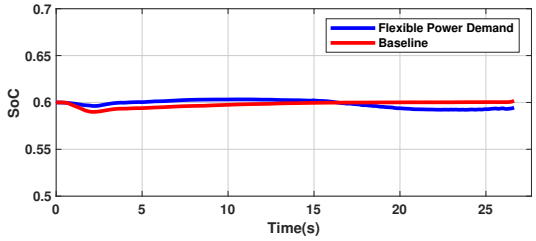

Fig. 8: Battery State of Charge History of the Vehicle.

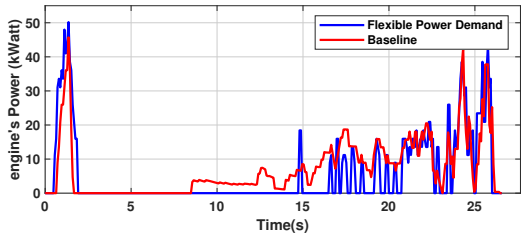

Fig. 9: Engine Power History of the Vehicle.

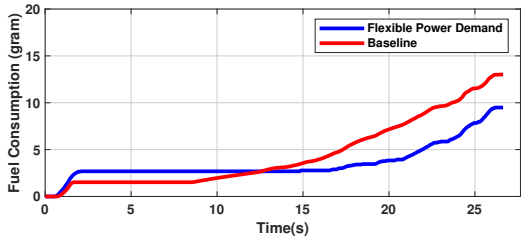

Fig. 10: Fuel Consumption History of the Vehicle.

Method	Terminal Δx (m)	Terminal Δv (m/s)	Terminal SoC (%)	Fuel (g)
ADP	-0.20	-0.14	59.45	9.49
Baseline	–	–	60.17	13.00

TABLE II: Summary of Results for the Drive Cycle

VI. Conclusion

Conventionally, in the study of hybrid electric vehicles, vehicle coordination optimization and powertrain energy management are studied separately. This paper explores a method to optimize these two levels jointly through a unique feature in the autonomous Hybrid Electric Vehicles (HEVs). This feature, which is referred to as flexible power demand, expresses that the power required by the external dynamics need not be met by the powertrain in an autonomous HEV at each step. This method of powertrain energy management can result in fuel economy improvement. Then, the optimization problem is formulated under the framework of a customized approximate dynamic programming method. Finally, to examine the efficacy of the proposed method, a case study from the Austin Data set is studied, which shows a 27% improvement in fuel consumption compared to the fixed power demand optimization method.

Acknowledgment

This research was partially supported by the National Science Foundation under Grant No. 1826410.

References

[1] X. Qu, Y. Yu, M. Zhou, C.-T. Lin, and X. Wang, "Jointly dampening traffic oscillations and improving energy consumption with electric, connected and automated vehicles: a reinforcement learning based approach," *Applied Energy*, vol. 257, p. 114030, 2020.

[2] M. Ghasemi and X. Song, "Powertrain energy management for autonomous hybrid electric vehicles with flexible driveline power demand," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 5, pp. 2229–2236, 2018.

[3] G. Ma, M. Ghasemi, and X. Song, "Integrated powertrain energy management and vehicle coordination for multiple connected hybrid electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 4, pp. 2893–2899, 2017.

[4] F. Zhang, X. Hu, R. Langari, L. Wang, Y. Cui, and H. Pang, "Adaptive energy management in automated hybrid electric vehicles with flexible torque request," *Energy*, vol. 214, p. 118873, 2021.

[5] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[6] D. P. Bertsekas *et al.*, *Dynamic programming and optimal control: Vol. 1*. Athena scientific Belmont, 2000.

[7] A. Heydari and S. N. Balakrishnan, "Finite-horizon control-constrained nonlinear optimal control using single network adaptive critics," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 1, pp. 145–157, 2012.

[8] T. Sardarmehni and A. Heydari, "Suboptimal scheduling in switched systems with continuous-time dynamics: A least squares approach," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2167–2178, 2017.

[9] M. Khajeh-Hosseini and A. Talebpour, "A novel clustering approach to identify vehicles equipped with adaptive cruise control in a vehicle trajectory data," in *100th Annual Meeting of the Transportation Research Board of National Academies*, 2021.