Engineering Science and Technology, an International Journal 43 (2023) 101434

HOSTED BY

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: www.elsevier.com/locate/jestch

Extended-deep Q-network: A functional reinforcement learning-based energy management strategy for plug-in hybrid electric vehicles

Amr Mousa [a,b]

[a] Powertrain Engineering Department, AVL List GmbH, Hans-List-Platz 1, 8010 Graz, Austria
[b] Automotive Mechatronics and Management, Fachhochschule Oberösterreich, Stelzhamerstraße 23, 4600 Wels, Austria

ARTICLE INFO

Article history:
Received 19 June 2022
Revised 3 April 2023
Accepted 1 May 2023
Available online 30 May 2023

Keywords:
Action masking
Deep Q-networks
Functional architecture
Reinforcement learning
Energy management strategy
Plug-in hybrid electric vehicles

ABSTRACT

Plug-in Hybrid Electric Vehicles offer a promising solution for the increasing CO_2 emission problem. However, the improved economy strongly depends on the energy management strategy. Traditional rule-based strategies are no more practical considering the increasing complexity in control objectives. In this study, an adaptive online Reinforcement Learning (RL) agent is developed, which learned an energy management strategy with a near-optimal performance. A novel hybrid approach is proposed to integrate the agent into the existing rule-based hybrid control unit architecture with a limited operation domain for more practicality and suitability to series-production control systems. Dynamic Programming (DP) and rule-based strategy are used to benchmark the developed RL agent performance. The objective is to minimize the vehicle's total fuel consumption and the frequent engine on/off switching to improve driver comfort and vehicle drivability. Several RL-based algorithms have been experimented and as a result, an Extended-Deep Q-Network (E-DQN) agent is proposed by this paper, trained on one cycle, and deployed on two other cycles with different onboard energy levels to evaluate the performance. The paper findings showed that E-DQN outperformed the rule-based strategy achieving up to 10.46% improvement in fuel economy closer to the DP performance alongside providing adequate compliance with the vehicle drivability and driver comfort objectives.

1. Introduction

1.1. Motivation

Plug-in Hybrid Electric Vehicles (PHEVs) showed significant improvement in fuel consumption and CO_2 emissions over the last decade. However, the performance superiority depends on the quality of the on-board Energy Management Strategy (EMS). The emerging technologies such as connected vehicles and automated driving increased the complexity of the EMS control objectives. Traditional hand-crafted rule-based approaches are robust, reliable and computationally affordable. However, they do not guarantee optimality with several objectives in a multi-domain, nonlinear and time-varying systems. Accordingly, the need for more intelligent controllers becomes vital for future vehicles.

1.2. Related work

Under the assumption of a complete prior driving cycle's knowledge in a finite horizon, several scholars used Dynamic Programming (DP) as a global optimization-based EMS to solve the

Abbreviations: A3C, Asynchronous Advantage Actor Critic; ADAM, Adaptive Moment Estimation; AB, Additive Boost; A-ECMS, Adaptive Equivalent Consumption Minimization Strategy; AER, All-Electric Range; AI, Artificial Intelligence; BSFC, Brake-Specific Fuel Consumption; CDCS, Charge-Depletion Charge-Sustaining; CO2, Carbon Dioxide; CD, Conventional Drive; DP, Dynamic Programming; DDQN, Double Deep Q-Network; DNN, Deep Neural Network; DQN, Deep Q-Network; DRL, Deep Reinforcement Learning; E-DQN, Extended-Deep Q-Network; EMS, Energy Management Strategy; ED, Electric Drive; EM, Electric Machine; eMPC, Explicit MPC; HCTI, History Cumulative Trip Information; HEV, Hybrid Electric Vehicle; HCU, Hybrid Control Unit; HFM, High Fidelity Model; HV, High Voltage; HWFET, Highway Fuel Economy Driving Schedule; ICE, Internal Combustion Engine; IR, Internal Resistance; ITS, Intelligent Transportation System; MPC, Model Predictive Control; NEDC, New European Driving Cycle; NN, Neural Network; OG, Optimum Generation; OD, Open Drive; OCV, Open-Circuit Voltage; PHEV, Plug-in Hybrid Electric Vehicle; POMDP, Partially Observable Markov Decision Process; PPO, Proximal Policy Optimization; PG, Policy Gradient; PSO, Particle Swarm Optimization; QP, Quadratic Programming; R, Recuperation; ReLU, Rectified Linear Unit; RL, Reinforcement Learning; RMSE, Root Mean Square Error; SARSA, State-Action-Reward-State-Action; SoC, State of Charge; TD, Temporal Difference; TqEM, EM Torque; TqICE, ICE Torque; UDDS, Urban Dynamometer Driving Schedule.

E-mail address: amrmousa.m@gmail.com

https://doi.org/10.1016/j.jestch.2023.101434
2215-0986/

problem for an optimal solution [2–6]. Although the developed strategy is only valid for this cycle and is impractical to generalize for others, DP is used to benchmark other sub-optimal real-time controllers.

There are several contributions such as instantaneous optimization-based EMSs such as Adaptive Equivalent Consumption Minimization Strategy (A-ECMS) introduced by C. Musardo et.al where the equivalent cost λ is estimated in real-time [7] while Guo et al. implemented a Model Predictive Control (MPC)-based EMS, with a novel velocity prediction method adopting Quadratic Programming [8]. Moreover, a Particle Swarm Optimization-based nonlinear MPC strategy (PSO-based MPC) is proposed in [9] saving 10% in the fuel consumption in comparison to the Charge-Depletion Charge-Sustaining (CDCS) strategy. The aforementioned approaches offered a feasible solution to real-time Hybrid Control Units (HCUs), however the trade-off between control objectives, controller performance in real-time, tuning effort, and vehicle hardware capabilities are difficult to balance [10]. Reinforcement Learning (RL)-based energy management approaches in PHEVs took immense attention of the Artificial Intelligence (AI) community due to the ability to learn control policies through interaction without being explicitly programmed on a certain strategy.

According to the RL methods classification introduced by Xiaosong, et.al [11], simplex algorithms such as Q-learning, Dyna and State-Action-Reward-State-Action (SARSA); and hybrid algorithms such as Deep Q-Network (DQN), Proximal Policy Optimization (PPO), Asynchronous Advantage Actor Critic (A3C), were used by several researchers and achieved substantial improvement to the performance in PHEV's EMSs.

A simplex model-free RL algorithm such as TD(λ)- learning was used by both Yue et al. in a super-capacitor Hybrid Electric Vehicle (HEV) [12] and Lin et al. in a thermal HEV [13]. The former achieved 10% less energy dissipation compared to the best baseline management policy while the latter attained improvement in fuel economy by 42%. Qi et al. used DQN, as a hybrid RL algorithm, to control the torque split ratio and it improved the fuel consumption up to 16.3% [14]. Actor-critic algorithm was utilized by He et al. achieving 89% of the fuel economy of DP with unknown driving conditions [15]. Furthermore, Zhu e al. introduced more advanced hybridization in RL algorithms designing the optimal speed and power depletion by the virtue of connectivity look-ahead information and mapping features [16]. The problem was formulated as a Partially Observable Markov Decision Process (POMDP) solved by actor-critic algorithm which saved 17.4% more fuel compared to the baseline controller.

1.3. Contribution

The contribution of this paper is developing an advanced online onboard energy management strategy using Deep Reinforcement Learning (DRL) which provides improvement to the vehicle fuel consumption while maintaining the level of driver comfort and vehicle drivability of the currently used rule-based approach.

More importantly, the novelty of this work and the distinction among other contributions is that the developed strategy does not replace the whole control unit, but only is incorporated on a limited domain. This domain includes certain powertrain operating modes while other modes are controlled by the rule-based approach. The determination of torque distribution between propulsion machines and other component state requests for each mode remains the responsibility of the conventional rule-based controller.

This approach has several advantages:

- The RL action space is narrowed, therefore it needs less computational resources to obtain a good policy and has higher potential for online real-time applications.
- System constraints, safety, diagnosis, and protection topics can be included in the control logic while maintaining verification and validation, requirements engineering, and other parts of the series production process on a high-quality level.

1.4. Paper outline

The rest of this paper is organized as follows: Section 2 presents the details of the vehicle model in MATLAB environment, based on the P2 powertrain configuration, utilizing the existing HCU logic derived from the Simulink High Fidelity Model (HFM) provided by the industrial partner, AVL List GmbH. Section 3 presents the RL-based EMS which is proposed to enable real-time control applying the Q-learning algorithm. Several techniques such as tabular Q-learning and Deep Neural Networks (DNNs) are experimented. The proposed methodology combined some of these techniques into an optimized E-DQN agent. In Section 4, this agent is tested and verified in a model-in-the-loop environment against the baseline EMS and the global optimization technique that provides the optimal EMS. Finally, Section 5 summarizes the findings, draws the conclusion and recommends the subsequent steps for future research.

2. Technical framework

2.1. Vehicle characteristics

The vehicle considered has a P2 configuration which locates the Electric Machine (EM) on the gearbox input, after the clutch, offering higher efficiency without the drag torque losses of the Internal Combustion Engine (ICE) [17], as shown in Fig. 1. The main vehicle parameters are listed in Table 1.

A quasi-static vehicle model for P2-PHEV is sufficient to maintain the vehicle physical causality and provides a plant model to develop the agent quickly and efficiently compared to using the HFM in training.

Considering the vehicle moves on a road with inclination θ in Fig. 2, the power demand P^{d} depends on the drivetrain internal power loss P^{loss}, and external forces F^{ext} as given by Eq. 1.

$$P_d = P_{loss} + (F_{ext} \cdot V) \tag{1a}$$

$$P_d = (T_{loss} \cdot \omega) + (F_{aero} + F_{tire} + F_{gravity} + F_{inertia}) \cdot V \tag{1b}$$

$$P_d = (T_{loss} \cdot \omega) + \left(\frac{1}{2}\rho A C_d V^2 + mg \cdot \cos(\theta) C_r + mg \cdot \sin(\theta) + ma\right) \cdot V \tag{1c}$$

In Eq. 1, ω is crankshaft rotational speed, ρ is the air density, A is the frontal area, C_d is the aerodynamic drag coefficient, C_r is the rolling

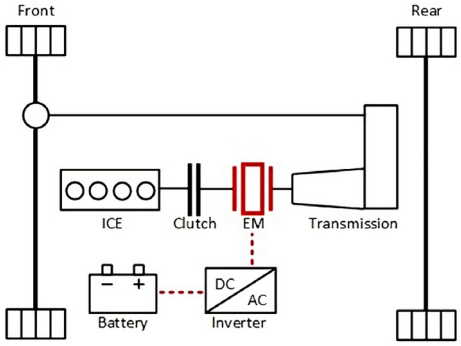


Fig. 1. The configuration of the P2-PHEV used in this research.

Table 1
Component parameters of the P2-PHEV model, source: AVL DSP team.

Component	Parameter	Value
Vehicle	Total mass	1998 kg
	Frontal area	2.349 m^2
ICE	Type	1.2L TGDI Gasoline Engine
	Max power	102 kW @ 5500 rpm
EM	Type	Permanent Magnet Synchronous Motor
	Max power	94 kW
Battery	Capacity	14.71 kWh
	Nominal voltage	350 V
	Max current	450 A
	Usable SoC range	20% - 95%
Transmission	Type	7-speed dual-clutch
	Gear ratios	[16.803 9.454 6.323 4.709 3.497 2.776 2.385]
Others	Electrical auxiliary load	500 W

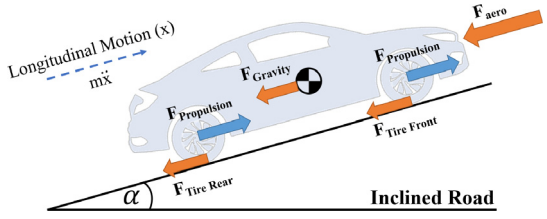


Fig. 2. Vehicle free-body diagram.

resistance coefficient, m is vehicle mass, θ is the inclination angle, and V is the current vehicle speed. The drivetrain T_{loss} results from the internal mechanical friction losses depending on the rotational speed, the torque, and the gear selected.

Vehicle model components are developed based on mathematical models and empirical performance maps. The ICE has a quasi-static fuel consumption model with neglected engine transients which are much faster than the vehicle dynamics. The fuel consumption is described by Eq. 2 which is plotted as the ICE Brake-Specific Fuel Consumption (BSFC) map in Fig. 3 where ω_{ICE} and T_{ICE} are the engine rotational speed and torque respectively.

$$\dot{m}_{fuel_{ICE}} = f(\omega_{ICE}, T_{ICE}) \tag{2}$$

Similarly, the EM model calculates the motor efficiency η_{EM} as a function of the motor rotational speed ω_{EM} and the torque T_{EM} governed by Eq. 3 for both modes, the motor in the positive torque region and the generator in the negative torque region Fig. 4.

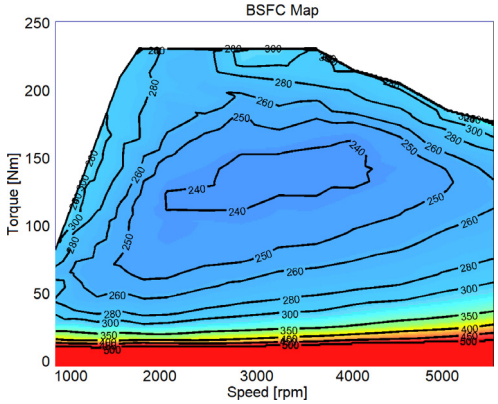


Fig. 3. ICE BSFC map (g/kWh), source: AVL DSP team.

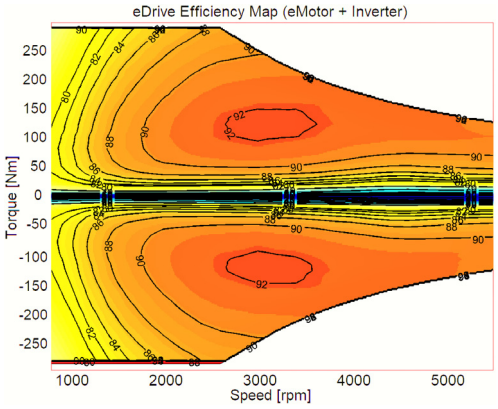


Fig. 4. EM efficiency maps, source: AVL DSP team.

$$\eta_{EM} = f(\omega_{EM}, T_{EM}) \tag{3}$$

The battery is modeled based on the Thevenin model [18] with an equivalent electric circuit configuration shown in Fig. 5 where the battery's State of Charge (SoC) is modeled with Eq. 4.

$$\dot{SoC} = \frac{-1}{Q_{Bat}} \cdot \frac{V_{OC} - \sqrt{V_{OC}^2 - 4P_{Bat} \cdot R_{Bat}}}{2R_{Bat}} \tag{4}$$

The equation parameters are open-circuit voltage V_{OC}, battery internal resistance R_{Bat}, battery terminals consumed power P_{Bat}, and battery capacitance Q_{Bat}. Battery pre-determined maps are used to estimate the Open-Circuit Voltage (OCV) and Internal Resistance (IR) only at 25 °C for model simplicity as shown in Fig. 6. The minimum and maximum battery SoC thresholds are set to be [20%, 95%] for the sake of battery health. Fig. 7

2.2. Hybrid control units

The HCU works coherently with several vehicle subsystems where the input signals combined with the driving situation are processed and the target "optimum" settings of drivetrain components ("optimum" hybrid operation mode) are selected. The 6 operation modes for the P2 configuration are classified as fixed modes and free modes according to Ambühl et al. [19] as follows:

- Fixed modes are restricted in selection by certain fixed rules as follows: **1) Additive Boost (acronym: AB** {mode index: 1}) which is selected when the demanded traction torque exceeds the maximum ICE torque and EM supplies the extra torque. **2) Recuperation (R** {7}) which is a regenerative breaking mode that uses the generator's negative torque to slow down the vehicle and charge the battery. **3) Open Drive (OD** {8}) that turns off the ICE and enables the EM to control the transmission oil pump which controls the clutches, when the vehicle is stationary.

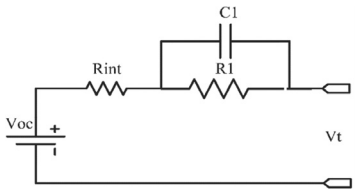


Fig. 5. Battery's equivalent electric circuit based on Thevenin's model [18].

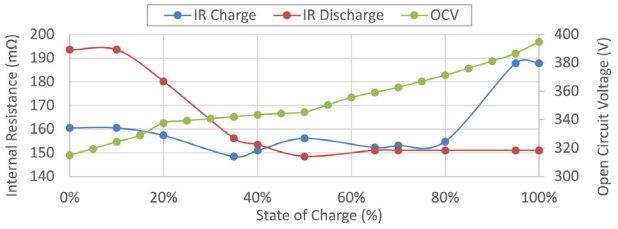


Fig. 6. The HV battery characteristics at 25 °C, source: AVL DSP team.

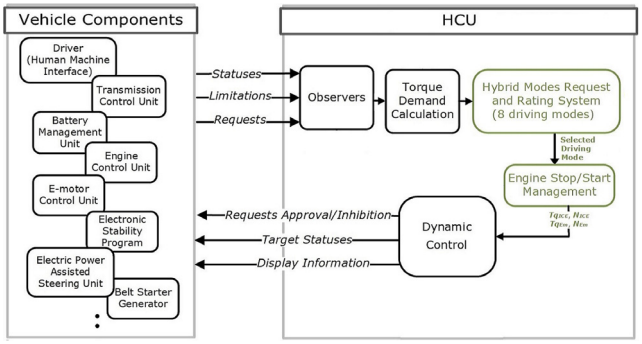


Fig. 7. Hybrid mode requests function in the generic P0-P4 HCU architecture, source: AVL DSP team.

- Free modes are selected in the time steps when several modes are available to select from and the energy distribution decision is crucial for accomplishing the optimization target as follows: **1) Conventional Drive (CD** {0}) where ICE is the only propulsion source and supplying the low voltage auxiliaries. **2) Optimum Generation (OG** {3}) where the demanded torque is low and the ICE load point can be increased to a better fuel-economy location. The EM works as a generator for charging the HV battery with the leftover power from the ICE. **3) Electric Drive (ED** {6}) which enables the EM to propel the vehicle.

3. Methodology

Reinforcement Learning, as a subset of machine learning, is used for solving control problems, allowing the agent to interact with the environment and learn the ideal policy by reinforcing or inhibiting patterns of behavior to maximize the reward. The agent applies sequential decisions and learns through a delayed environment feedback which makes it more suitable for applications in real-time such as the PHEV's EMS. The reader is referred to [20] for the more details about RL elements such as policies, reward functions, value functions, and environment model.

3.1. Problem formulation

The RL agent for P2-PHEV has a continuous state space defined by $s_k = [SoC_k, T_{d_k}, V_k, D_{rem_k}, Eon_k]$ where at time step k, SoC_k is the battery state of charge, T_{d_k} is the driver torque demand, V_k is the vehicle velocity, D_{rem_k} is the trip remaining distance and E_{on_k} is the engine on/off state. D_{rem_k} is included in the state space after several experiments to give the agent insight over proceeding in the cycle and plan the SoC depletion trajectory accordingly. E_{on_k} is included for later use in the RL reward function definition to minimize the engine on/off switching frequency to improve driver comfort and vehicle drivability. The action space is discrete where the control variable, the driving mode selection, is $a_k \in \{0, 1, 3, 6, 7, 8\}$. The P2-PHEV's discrete-time space control optimization problem and system constraints are described in Eqs. 5 to 11.

$$[SoC_{k+1}, T_{d_{k+1}}, V_{k+1}, D_{rem_{k+1}}, Eon_{k+1}] = f([SoC_k, T_{d_k}, V_k, D_{rem_k}, Eon_k], a_k), k = 0, 1, \ldots n-1 \quad (5)$$

$$\min J_\pi(s_0) = \lim_{n\to\infty} E\left\{\sum_{k=0}^{n-1} \gamma^k \cdot r(s_k, a_k)\right\} \quad (6)$$

Subject to

$$a_k \in \{0, 1, 3, 6, 7, 8\} \quad (7)$$

$$SoC_{min_k} \leqslant SoC_k < SoC_{max_k}, SoC_0 = SoC_{init} \quad (8)$$

$$T_{ICE_{min_k}} \leqslant T_{ICE_k} < T_{ICE_{max_k}} \quad (9)$$

$$T_{EM_{min_k}} \leqslant T_{EM_k} < T_{EM_{max_k}} \quad (10)$$

$$I_{Bat_{min_k}} \leqslant I_{Bat_k} < I_{Bat_{max_k}} \quad (11)$$

3.2. Tabular Q-learning based EMS

The tabular Q-Learning algorithm updates the Q-value for each state-action pair using Bellman Eq. 12 until the Q-function converges to the optimal Q-function (Q^*) in an approach called the iteration of values [20].

$$Q(s_k, a_k) \leftarrow Q(s_k, a_k) + \alpha\left[r_k + \gamma \max_a Q(s_{k+1}, a) - Q(s_k, a_k)\right] \quad (12)$$

Q-function is represented in a table called Q-table whose rows represent the states, and the columns represent the actions. Thus, the table dimensions are the number of states multiplied by the number of actions. Continuous state space is not possible as the Q-table number of rows would be infinite, and accordingly, state-space discretization is a necessity. The P2-PHEV's EMS algorithm based on a model-free Q-learning RL agent is represented in Algorithm 1.

Algorithm 1: Model-free Q-learning algorithm for P2-PHEV

```
Set values for learning rate α, discount factor γ, epsilon ϵ, epsilon decay dϵ
Initialize Q(s, a) to zeros
for episode = 1: number of episodes do
    Reset environment with s_0
    for k = 1: number of steps per episode do
        With probability ϵ, select a random action a_k
        Otherwise, select a_k = argmax Q(s, a)
        Execute action a_k, and observe reward r_k and state s_{k+1}
        Update Q with bellman equation:
        Q(s_k, a_k) ← Q(s_k, a_k) + α[r_k + γ max_a Q(s_{k+1}, a) − Q(s_k, a_k)]
        s ← s', ϵ ← ϵ · dϵ
    end for
end for
```

3.3. Deep Q-learning based EMS

The DQN algorithm, first proposed by DeepMind [21], is used to estimate and update the Q-values of each state-action pair in a given environment. The agent developed in this research has the following characteristics:

3.3.1. NN architecture

The algorithm's Neural Network (NN) has an input layer with five neurons fed by the state values, four hidden layers each of 32 neurons and Rectified Linear Unit (ReLU) activation function, and an output layer estimating the Q-value for each one of the six actions. Various studies confirmed that the network architecture highly affects the agent performance [22,23]. After considering other contributions tackling similar problems in RL [5,24–26], a separate optimization study was conducted to select the best NN architecture.

The agent collected 50 k experience tubles, following a random policy, which were postprocessed to accumulate the episodic reward and calculate the Q-value for each tuble. Afterwards, several NN architectures were trained and tested on different datasets in a supervised learning fashion to calculate the Root Mean Square Error (RMSE) as training and validation errors. The aforementioned architecture was selected due to achieving the minimum RMSE among others which indicates the best ability for the NN to learn and approximate the Q-function efficiently.

3.3.2. DDQN with τ-soft update

The Double Deep Q-Network (DDQN) algorithm, introduced by Hasselt et al. [27], is used to reduce the observed Q-values overestimation bias and stabilize the training by decoupling the action selection from the target Q-value estimation using two different NNs; policy and target networks. Lillicrap et al. proposed an improvement to the DDQN called 'τ-soft update' which controls the moving target and stabilizes the learning process. The τ value lies in the range of [0,1] where a value in between makes the target network weights slowly track the learning policy network instead of directly copying the weights, while $\tau = 1$ makes it the algorithm a normal DQN [28]. The authors claim that such minor change guarantees convergence to a robust solution by moving the unstable function approximation problem closer to a supervised learning problem.

3.3.3. N-steps bootstrapping

The Bellman equation for estimating the $Q(s,a)$ of each state-action pair, Eq. 13, represents the 1-step Temporal Difference (TD) which is often called TD(0) [20].

$$Q(s_t,a_t) = r_t + \gamma \max_a Q(s_{t+1},a) \tag{13}$$

The equation is recursive and $Q(s_{t+1},a_{t+1})$ can be replaced by its estimate from s_{t+1} assuming a_{t+1} is chosen optimally or near optimally. Unrolling it for n-times is called n-steps bootstrapping technique. It was experimented by Fedus et al. [29] and revealed that the agent's performance relies heavily on the proper selection of the n-value which they suggested to be small, e.g., 3 or 4.

3.3.4. Action masking

Several functionalities are defined with the HCU's internal logic which decides on the available modes and represent them in a logical "mode enabler" vector, Fig. 8. Action masking, proposed by Vinyals et al. [30], uses this vector, after changing the 0 logic to $-\infty$, and multiply it by the network estimated Q-values as shown in Fig. 9. Accordingly, the best mode of only the available modes will be selected according to the RL policy. Other approaches suggested including the agent only in the free modes and bypass it in the fixed modes segment. This approach was tested, and the agent faced difficulties in learning the correct value function that represents the future cumulative rewards, as the agent is not aware of part of such rewards if bypassed, hence it never converged. The advantage of our approach is that the agent will have full-observability/controllability over the environment even with the action masking presence, therefore the value function converges, and a proper policy is built.

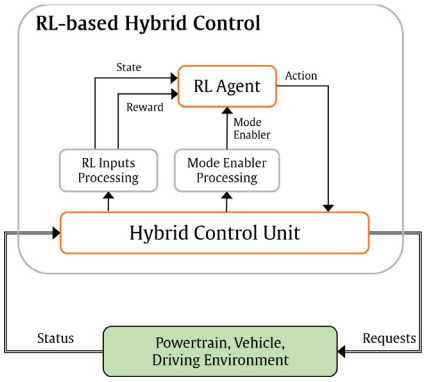


Fig. 8. The proposed RL-based HCU architecture.

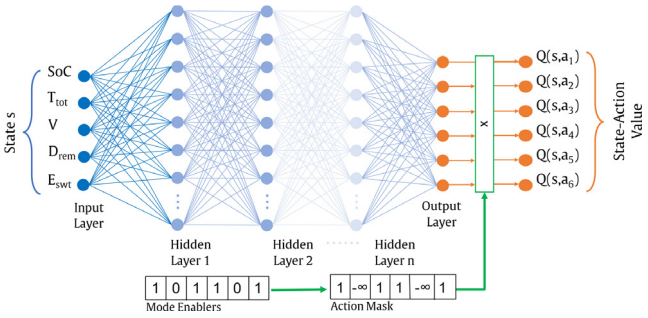


Fig. 9. RL agent incorporating the DQN's neural network and action masking.

3.3.5. Reward function

The reward signal represents the search objective of the RL problem that incentivizes/inhibits the agent behavior in the environment, hence affects the performance. Various scholars noticed that gradual battery SoC depletion, sustaining the battery charge over the entire trip, tends to provide a near-optimal performance for the EMS problem. Accordingly, space-domain indexed SoC reference is proposed to guide the SoC depletion rate gradually in the entire trip taking into account the prior knowledge of the trip distance D [31,32]. The trip distance can be estimated from the navigation systems widely available nowadays. Additionally, Li et.al proposed a model called History Cumulative Trip Information (HCTI) to estimate its value with experimental proven accuracy from historical record [31]. The reward function defined utilizing the traveled distance $d \in D$ and the engine change state indicator E_{switch} (starting-up or shutting-down) is shown in Eq. 14. χ, ϕ and ψ are set to 40, 36 and 0.7 respectively after careful tuning and the hyperbolic tangent function is used to bound the reward in the [-1,1] range for more stability in the NN estimations.

$$Reward = -\tanh\left(\phi \cdot |SoC - SoC_{reference}| + \chi \cdot \dot{m}_{fuel} + \psi \cdot E_{switch}\right) \tag{14}$$

$$SoC_{reference} = SoC_{initial} \cdot (1 - d/D) + d/D \cdot SoC_{final} \tag{15}$$

Combining the previous techniques into an extended version of the DQN agent, the E-DQN model-free RL agent is represented in Algorithm2.

Algorithm 2: Model-free DDQN algorithm for P2-PHEV

1: Set values for RL training parameters and NN hyperparameters
2: Initialize replay memory D with capacity N
3: Initialize policy network Q with random weights θ
4: Initialize target network $\hat{Q}$ with random weights θ^{-}
5:**for** episode = 1: number of episodes **do**
6: Reset environment with s_0
7: **for** k = 1: number of steps per episode **do**
8: With probability ϵ, select a random action a_k
9: Otherwise, select $a_k = argmaxQ(s,a)$
10: Execute action a_k, and observe the n-steps bootstrapped reward $\sum_k^{k+n} r$ and state s_{k+n+1}
11: Store transition $\left(s_k, a_k, \sum_k^{k+n} r, s_{k+n+1}\right)$ in memory D
12: $s \leftarrow s\prime, \epsilon \leftarrow \epsilon \cdot d\epsilon$
13: Sample random minibatch of (s_j, a_j, r_j, s_{j+1}) from memory D
14: Set $y_j = \begin{cases} r_j & \text{if} s_j \text{is terminal} \\ r_j + \gamma \max_a \hat{Q}(s_{j+1}, a; \theta^{-}) & \text{otherwise} \end{cases}$
15: Perform gradient descent step on $\left(y_j - Q(s_j, a_j; \theta)\right)^2$ with respect to the policy network parameters θ
16: Update the target network $\hat{Q}$ parameters $\hat{\theta} = \tau\theta + (1-\tau)\theta^{-}$
17: **end for**
18: **end for**

4. Results and Discussion

4.1. Simulation environment

The driving cycles used in this research are New European Driving Cycle (NEDC), Highway Fuel Economy Driving Schedule (HWFET), Urban Dynamometer Driving Schedule (UDDS) [33], and GRAZ cycle which is an in-house cycle provided by AVL team. The vehicle model's accuracy influences the optimality of the trained RL control strategy; therefore, model validation experiments were conducted to quantify the estimation error for fuel consumption and SoC change compared to the HFM. The experiments used the NEDC cycle repeated six times to cover the vehicle's All-Electric Range (AER) with different initial SoC levels. The results assert that SoC and fuel consumption differences are negligible where the SoC difference oscillates between −0.3: 0.2% while the accumulated fuel consumption difference ranges between 30: 50 ml, which is 1.1% of the total fuel consumption. It is concluded that the vehicle model is accurate enough to be used instead of the HFM for further experimentation.

4.2. Tabular Q-learning based EMS

Algorithm 1 showed how the tabular Q-learning method is incorporated into an RL agent. The environment state space $(SoC, T_{tot}, V, D_{rem}, E_{on})$ is discretized to (40, 20, 20, 20, 2) resulting in a Q-table with dimensions of (640,000 x 6). The training hyperparameters are tuned after several experiments and summarized in Table 2.

Fig. 10 demonstrates the Q-learning performance results on the GRAZ cycle with $SoC_{init} = 75\%$. Fig. 10a shows the episode cumulative return and the Q_0 which represents the estimated Q-value at the beginning of the episode. Q_0 represents the agent's expected cumulative return and the closer it is to the true return, the better the agent can expect future rewards, and accordingly, the best actions can be taken to maximize the cumulative episode reward. Nevertheless, Q_0 never approached the true return value which reveals that the agent is not able to learn the environment correctly as shown in Fig. 10a.

Table 2
The training hyperparameters of the Q-learning algorithm.

Learning rate α	0.01	Epsilon decay rate $d\epsilon$	0.01
Discount rate γ	0.995	Epsilon minimum $d\epsilon_{min}$	0.1
Epsilon ϵ_{start}	1		

The Q-learning based agent suffers from the curse of dimensionality due to discretization [32]. Although the discretization level used causes the Q-table to be huge (640, 000x6) and requires large memory in the HCU to handle, it is not sufficient for proper segregation and differentiation between adjacent states. The histogram in Fig. 11 shows how frequently each state is visited during a single episode. The results reveal that the majority of the states are visited more than once, up to 658 times. In discrete state-space environments such as the grid world, each state-action pair holds a Q-value that fully represents the discounted cumulative return following the optimal policy till the episode end according to the Q-value definition. However, in the discretized continuous state space environments such as PHEV's, the discretized state-action pairs are visited more frequently with different Q-values which causes the Q-function to diverge and never approximate its true representation. Fig. 10b shows the true cumulative return and the Q-value for each time-step/state in the current episode. Proceeding with the agent training, the trajectory of the estimated Q-values shall follow the true return especially with small values of the exploration term ϵ. However, due to the coarse discretization and the curse of dimensionality problem, both curves never followed each other with different hyperparameters in several experiments. The tabular Q-learning approach is concluded not to be suitable for future expanded applications with large state space environments such as PHEVs.

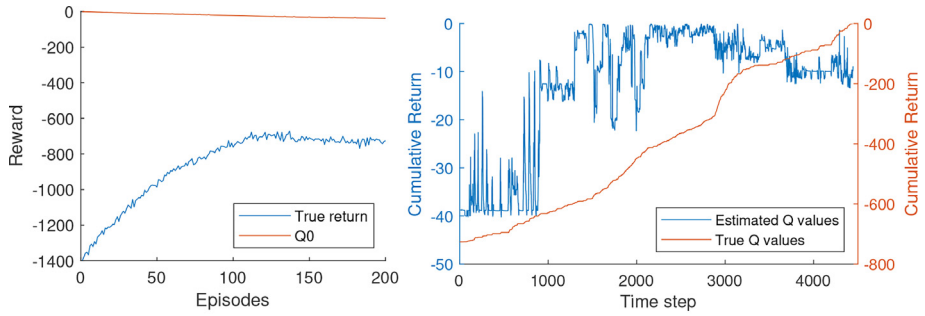
4.3. Deep Q-learning based EMS

4.3.1. Training Data

Machine learning researchers recommend preprocessing the input data to the NN to be normalized in a range of [0–1] or standardized with a zero mean and a standard deviation of one [34]. Input data normalization was applied instead of standardization for more practicality regarding operating the RL agent with new unseen cycles which have unknown means and standard deviations but known ranges instead.

4.3.2. N-steps bootstrapping

Four agents were tested using the same hyperparameters but with varying the n-steps for each. Fig. 12a, representing the TD (0) agent with $n = 1$, had a diverging Q-function, however, the agent started to converge very slowly using $n = 3$ as the case in Fig. 12b. Better performance is achieved by setting $n = 16$ as shown in Fig. 12c in the contrary to the recommendation of setting the n-value to be small, around 4, for better off-policy behavior [29]. However, it is noticed that decaying the n-value from 16 to 6, by reducing 1 each 10 episodes, showed the best performance as demonstrated in Fig. 12d. After episode 100, the n-value was 6 which helped the agent to utilize the experience available in the replay buffer to learn more efficiently for 80 more episodes closer to the off-policy behavior. The Q0 value started to stabilize after episode 100 even with oscillations due to the randomness in the



(a) Training rewards (b) Expected and true return per episode

Fig. 10. Q-learning agent results on the GRAZ cycle.

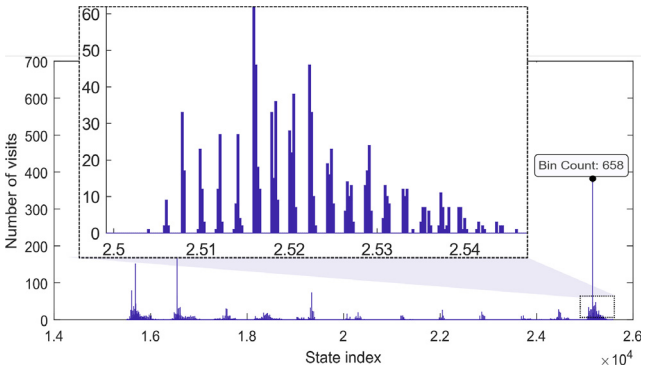
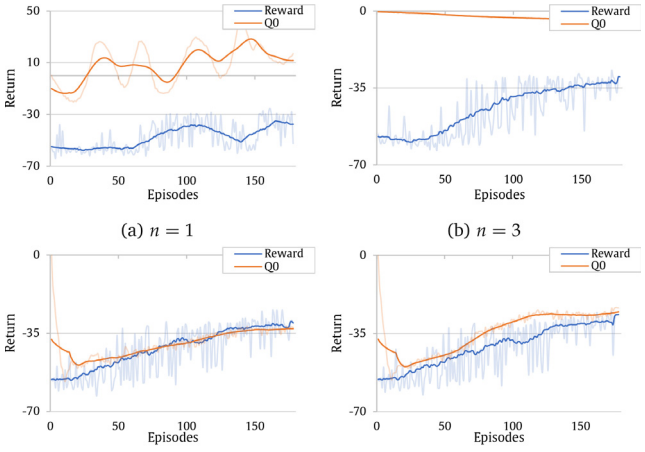


Fig. 11. Discretized states visit frequency per episode.



(a) $n = 1$ (b) $n = 3$

(c) $n = 16$ (d) $n = 16 - 6$

Fig. 12. n-steps bootstrapping results in DRL training.

action selection and the reward tried to come closer and closer by proceeding in the training.

4.3.3. E-DQN agent for the P2-PHEV

The previous results are combined into an extended version of the DQN agent (E-DQN) which is a model-free RL agent with decaying n-steps bootstrapping, and τ-soft update. The training set of optimal hyperparameters are defined in Table 3 after several tuning experiments.

Fig. 13a shows the agent training results where after limited number of episodes, the Q0 approached the cumulative reward which reflects building proper knowledge about the environment while the training loss was decreasing to convergence in Fig. 13b.

The trained agent behavior was tested on the same cycle following the policy completely with no exploration. The agent achieved 98.47% of the DP optimal results while the CDCS achieved 96.59%. The difference between the CDCS and RL is not significant in the HWFET cycle because the whole cycle is in a high-speed range which means the optimum engine operation zone in addition to a little amount of energy available onboard to be utilized with $SoC_{init} = 30\%$. Accordingly, the trained agent is simulated on 6-UDDS (six adjoining UDDS cycles to exceed the AER) and GRAZ cycle to test its generalization capability and robustness with different SoC_{init} levels. The numerical results are summarized in Table 4 while the performance is shown in Fig. 14 and Fig. 15.

Different SoC_{init} levels highly affected the performance of the E-DQN agent compared to the CDCS and DP. With $SoC_{init} = 25\%$ shown in Fig. 14a, the agent almost followed the CDCS strategy due to the lack of available electric energy onboard, therefore proper energy utilization is not possible. However, at higher SoC_{init} levels such as 50% and 75% shown in Fig. 14b and Fig. 14c respectively, the RL agent was closer to the DP behaviour rather than the CDCS by depleting the electric energy wisely throughout the whole trip. This enabled the RL agent to operate the ICE on optimal load points as shown in engine BSFC plot, Fig. 14d. The RL agent was closer to the ICE optimum operation line and the DP which interprets the 10.46% improvement in the fuel economy.

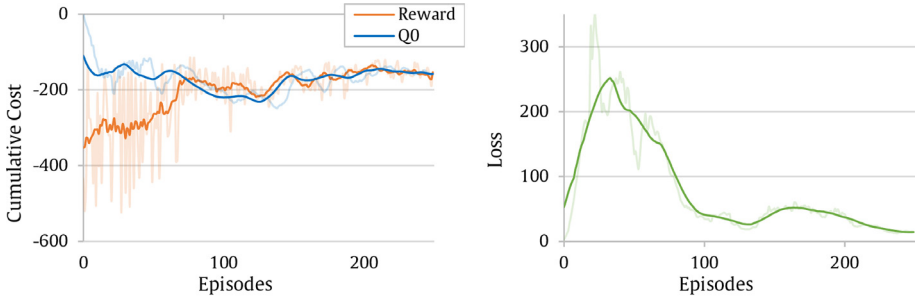
Moreover, E-DQN agent was tested on GRAZ cycle where the performance of the three SoC_{init} levels is demonstrated in Fig. 15. For a better understanding of the results, the mode selection for the three strategies with 75% SoC_{init} is plotted in Fig. 15e for GRAZ cycle. The CDCS strategy started depending on the ICE where SoC begins to be sustained only after reaching the minimum threshold. On the contrary, the DP selected the Conventional Drive (CD) mode once starting the highway region close to time step 1600s where the velocity is relatively high. Moreover, the Electric Drive (ED) mode was used instead for the other cycle segments where the velocity is relatively low.

The main unique strategy of the DP is to use the ICE more in the high-speed segments and depends completely on the ED mode for all the other low-speed segments. Such a strategy strongly concurs with the ICE system characteristics of working more efficiently at high velocities as illustrated in Fig. 15d showing the ICE load point locations on the BSFC map. The DP located the load points mostly in the central region within 240 g/kWh BSFC while the ICE load points in the CDCS strategy are located on the left region, which is much less optimal. In addition to the ICE utilization, DP tended to deplete the SoC gradually through the entire trip which coincides with the conclusion derived previously in [31,32].

On the other side, the RL used a different strategy of prioritizing the ED mode always till the 1850s where a quick drop in the SoC

Table 3
The training hyperparameters of the E-DQN agent.

Learning rate μ	0.001	Epsilon decay rate $d\epsilon$	0.005
Learning rate decay $d\mu$	0.0005	Epsilon minimum $d\epsilon_{min}$	0.1
Learning rate update frequency	30 eps.	n-steps	16–6
Learning rate minimum μ_{min}	1e-05	Experience buffer size	10,000
Epsilon ϵ_{start}	1	NN optimizer	ADAM



(a) Training rewards (b) Training loss

Fig. 13. E-DQN agent training results on HWFET cycle with 30%SoC_{init}.

Table 4
E-DQN generalization performance results on 6-UDDS and GRAZ cycles (Values in **bold** represent the best result).

		6-UDDS Cycle			GRAZ Cycle		
		25%	**50%**	**75%**	**25%**	**50%**	**75%**
Final SoC (%)	**CDCS**	20.6%	20.5%	20.5%	20.5%	20.5%	20.5%
	DP	20.8%	20.6%	20.6%	20.4%	20.5%	20.4%
	RL	20.8%	20.8%	20.8%	20.5%	20.4%	20.5%
Fuel Consumption (ml)	**CDCS**	3962.8	2907.3	1547.6	3804.0	2676.2	1595.3
	DP	**3873.5**	**2656.1**	**1326.4**	**3788.5**	**2592.6**	**1404.2**
	RL	3924.2	2721.7	1409.7	3797.3	2681.3	1483.8
Fuel Economy(%)	**CDCS**	97.70%	90.54%	83.32%	99.59%	**96.78%**	86.39%
	DP	100%	100%	100%	100%	100%	100%
	RL	**98.69%**	**97.53%**	**93.72%**	**99.77%**	96.58%	**94.33%**
Number of engine starts	**CDCS**	**34**	72	**60**	**66**	72	**32**
	DP	308	320	210	103	76	53
	RL	62	**66**	106	68	**33**	50

takes place as shown in Fig. 15c. The CD mode is used instead till the end of the highway segment where the ED mode is prioritized again after the 2400s. The CD mode is used for some segments later to maintain the SoC depletion trajectory within a proper rate while trying to minimize the number of engine starts to maximize the total episode reward. Fig. 15d shows that the adjacency between the DP and the RL is closer than between the RL and the CDCS confirming the cognition of better engine utilization and higher fuel economy.

The results in Table 4 validate the surplus of the E-DQN agent compared to the CDCS strategy particularly in the high SoC_{init} levels where much electric energy is available and can be utilized properly. The RL agent outperformed the CDCS with more than 10% and around 8% closer to DP performance in the 6-UDDS and the GRAZ cycles respectively with 75% SoC_{init}. However, its performance was very close to the CDCS by less than 1% improvement with 25% SoC_{init} in both cycles.

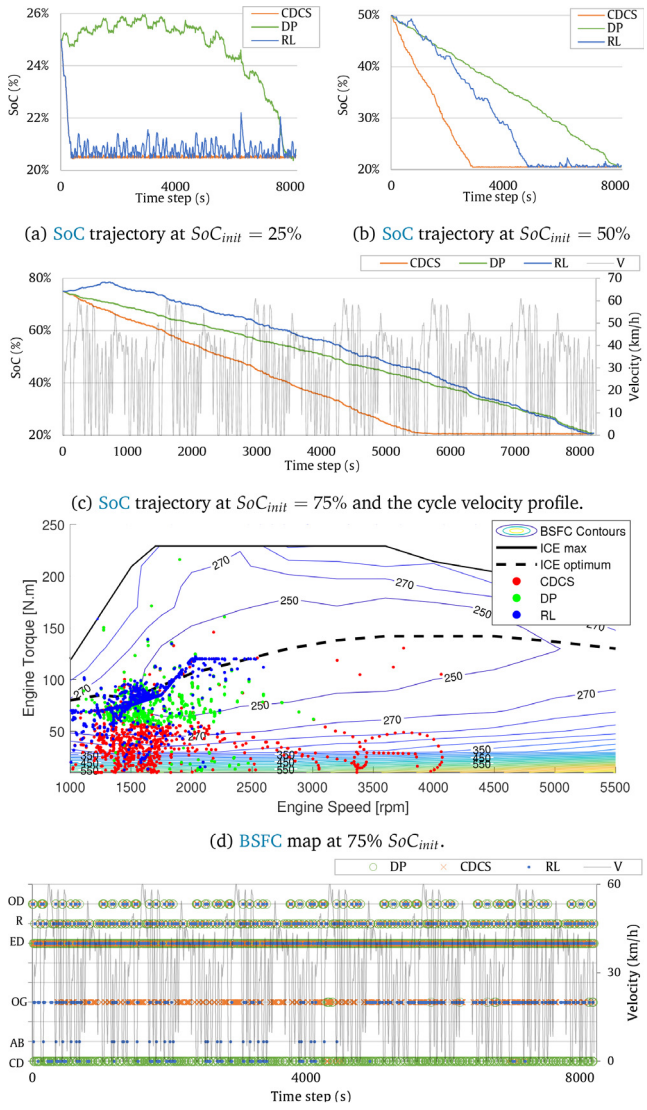
Furthermore, the number of engine-starts for the RL agent was kept in an acceptable range with an average of one engine start each 96.3s for the GRAZ cycle compared to one start each 61.8s and 89.3s in the DP and CDCS respectively. However, the CDCS achieved longer time for each engine start with an average of 155.4s in the 6-UDDS compared to 30.5s and 105.5s for the DP and the RL respectively.

5. Concluding Remarks

5.1. Summary and conclusion

As one of the main categories in the automotive industry electrification process, PHEVs offer a promising solution for the increasing CO_2 emission problem. Their improved economy strongly depends on the HCU's control strategy. Navigation, communication devices, and sensors inspired a growing development of the advanced energy management strategies especially that the CDCS strategy is neither simple nor advantageous anymore with the increasing control objectives.

Several scholars proved that advanced AI-based strategies such as reinforcement learning EMSs can significantly improve the fuel economy of PHEVs. This study introduced an adaptive online learning RL agent into the existing HCU architecture. DP results are used to benchmark the developed RL algorithms which solved the EMS for near-optimal solutions. The development process began with formulating the control problem mathematically as an infinite-horizon optimal-control problem. The mathematical formulation is a function of the battery SoC, driver torque demand, vehicle speed, remaining trip distance, and the engine on/off status. The objective is to minimize the total fuel consumption and the frequent engine switch. Accordingly, the design process reasonably



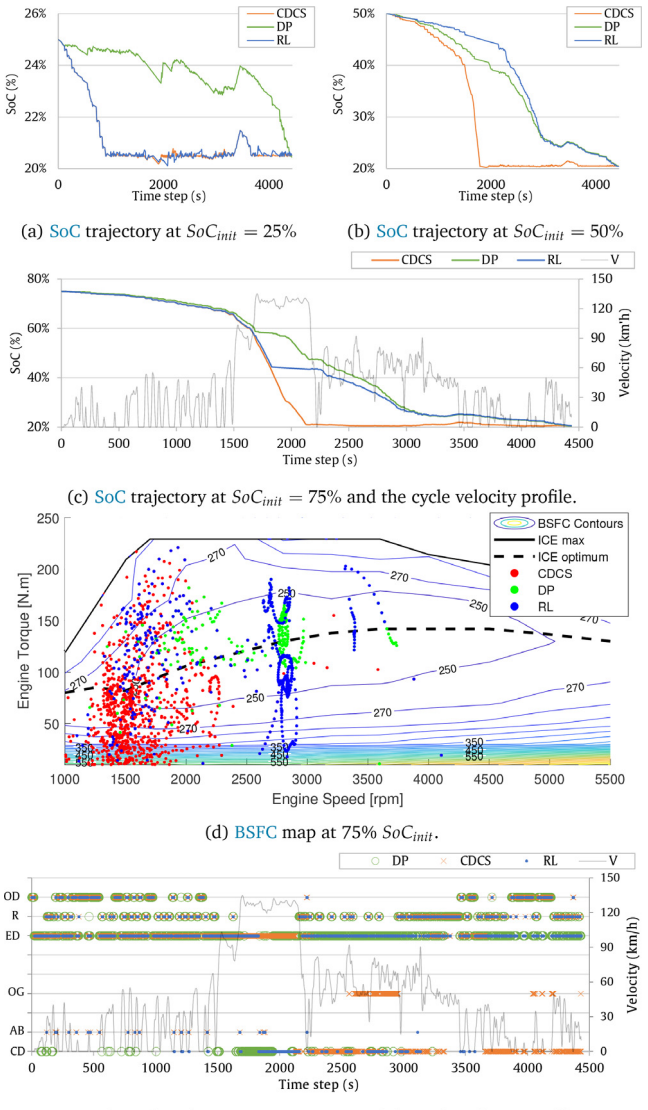
(a) SoC trajectory at $SoC_{init} = 25\%$ (b) SoC trajectory at $SoC_{init} = 50\%$

(c) SoC trajectory at $SoC_{init} = 75\%$ and the cycle velocity profile.

(d) BSFC map at 75% SoC_{init}.

(e) Selected modes at $SoC_{init} = 75\%$ and the cycle velocity profile.

Fig. 14. E-DQN performance on the 6-UDDS cycle with different SoC_{init}.



(a) SoC trajectory at $SoC_{init} = 25\%$ (b) SoC trajectory at $SoC_{init} = 50\%$

(c) SoC trajectory at $SoC_{init} = 75\%$ and the cycle velocity profile.

(d) BSFC map at 75% SoC_{init}.

(e) Selected modes at $SoC_{init} = 75\%$ and the cycle velocity profile.

Fig. 15. E-DQN performance on the GRAZ cycle with different SoC_{init}.

considered the drivability and comfort requirements instead of sacrificing them too much for the sake of fuel economy.

As a result, an optimized E-DQN agent is proposed by the research, trained on the HWFET cycle, and deployed on other two cycles to evaluate the performance. The E-DQN's control strategy outperformed the CDCS strategy in terms of fuel economy, up to 10.46% improvement, alongside providing adequate compliance with the control objectives. The research findings strongly accord to the necessity of enabling AI-based control strategies into the next generation of automobiles particularly in the era of autonomous driving and connected vehicles. The control objectives are becoming more and more complex for traditional methods to handle, thus AI-enabled technologies shall be prioritized by OEMs for further research and development.

5.2. Future prospects and recommendations

The accelerating development of computational resources in recent decades enabled novel complex and intelligent algorithms to be involved in modern control systems. Advancements in the RL field incorporating deep learning and neural networks form the next trend for the RL-based PHEVs energy management strategies. Integration with the Intelligent Transportation Systems (ITSs) to construct a smart city or a smart grid is coming soon with more comprehensive and complicated optimization control objectives. In the future connected environment, distributed and multi-agent DRL systems are necessities for cooperative learning between vehicles on the road. The sooner the transition towards intelligent control systems by automotive OEMs, the better they are prepared and qualified for the upcoming challenges. The research presented an initial step overlooking the way towards realizing an intelligent adaptive energy management system for PHEVs. The upcoming research efforts shall investigate more the following spots:

- DRL algorithms: the DQN algorithm is solely considered by the research in DRL algorithms. However, other types, such as the Policy Gradient (PG) family, are promising to be examined [35] including PPO [36], and A3C [37].
- Multiple objectives: for improved performance, including other objectives to be optimized such as the battery's state of health [38], safety, comfort, user convenience [39], and powertrain mobility [40], bring additional benefits and are advantageous.
- RL agents testing and validation: besides the theoretical feasibility that is validated by simulation, practical implementation is necessary to be achieved through real vehicle evaluations.

Few studies proceeded with their proposed methodologies to the hardware-in-the-Loop and vehicle-in-the-loop testing. Therefore, more research efforts shall further examine the validity of such approaches in real environments.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The author would like to thank Prof. Thomas Schlechter and Prof. Stefan Winkler for their support during the master thesis research. Moreover, sincere gratitude to the AVL DSP team: Patrick Teufelberger, Huan Chen, Evgeny Korsunsky and Bhargav Adabala, for their collaboration and valuable feedback. Furthermore, deep thanks to the editors and anonymous reviewers for providing insightful suggestions and comments to improve the quality of research paper.

References

[1] A. Mousa, AI-based Energy Management Strategies for P2 Plug-in Hybrid Electric Vehicles (2021). doi: 10.5281/ZENODO.7684683. URL: doi: 10.5281/zenodo.7684682.
[2] M.P. O'Keefe, T. Markel, Dynamic programming applied to investigate energy management strategies for a plug-in hev, in: 22nd International Battery Hybrid and Fuel Cell Electric Vehicle Symposium, EVS 2006, 2006, pp. 1035–1046.
[3] J. Liu, H. Peng, Modeling and control of a power-split hybrid vehicle, IEEE Transactions on Control Systems Technology 16 (6) (2008) 1242–1251, https://doi.org/10.1109/TCST.2008.919447.
[4] Y. Yang, X. Hu, H. Pei, Z. Peng, Comparison of power-split and parallel hybrid powertrain architectures with a single electric machine: Dynamic programming approach, Applied Energy 168 (2016) 683–690, https://doi.org/10.1016/j.apenergy.2016.02.023.
[5] C. Hou, L. Xu, H. Wang, M. Ouyang, H. Peng, Energy management of plug-in hybrid electric vehicles with unknown trip length, Journal of the Franklin Institute 352 (2) (2015) 500–518, https://doi.org/10.1016/j.jfranklin.2014.07.009.
[6] O. Sundström, L. Guzzella, A generic dynamic programming matlab function, 2009, pp. 1625–1630. doi: 10.1109/CCA.2009.5281131.
[7] C. Musardo, G. Rizzoni, B. Staccia, A-ecms: An adaptive algorithm for hybrid electric vehicle energy management, Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, CDC-ECC '05 2005 (2005) 1816–1823. doi: 10.1109/CDC.2005.1582424.
[8] N. Guo, Z. Chen, Y. Wu, J. Shen, R. Xiao, A novel velocity forecast method for improving predictive energy management of plug-in hybrid electric vehicles, in: 2017 IEEE Vehicle Power and Propulsion Conference VPPC 2017 - Proceedings 2018-January, 2018, pp. 1–6, https://doi.org/10.1109/VPPC.2017.8330915.
[9] Y. Wang, X. Wang, Y. Sun, S. You, Model predictive control strategy for energy optimization of series-parallel hybrid electric vehicle, Journal of Cleaner Production 199 (2018) 348–358, https://doi.org/10.1016/J.JCLEPRO.2018.07.191.
[10] H. Chen, Predictive Control Strategies of Plug-in HEVs, Ph.D. Thesis (2019).
[11] X. Hu, T. Liu, X. Qi, M. Barth, Reinforcement learning for hybrid and plug-in hybrid electric vehicle energy management: Recent advances and prospects, IEEE Industrial Electronics Magazine 13 (3) (2019) 16–25, https://doi.org/10.1109/MIE.2019.2913015.
[12] S. Yue, Y. Wang, Q. Xie, D. Zhu, M. Pedram, N. Chang, Model-free learning-based online management of hybrid electrical energy storage systems in electric vehicles, IECON Proceedings (Industrial Electronics Conference) (2014) 3142–3148, https://doi.org/10.1109/IECON.2014.7048959.
[13] X. Lin, Y. Wang, P. Bogdan, N. Chang, M. Pedram, Reinforcement learning based power management for hybrid electric vehicles, IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD 2015-January (January) (2015) 32–38. doi: 10.1109/ICCAD.2014.7001326.
[14] X. Qi, Y. Luo, G. Wu, K. Boriboonsomsin, M.J. Barth, Deep reinforcement learning-based vehicle energy efficiency autonomous learning system, IEEE Intelligent Vehicles Symposium, Proceedings (2017) 1228–1233, https://doi.org/10.1109/IVS.2017.7995880.
[15] J. Wu, H. He, J. Peng, Y. Li, Z. Li, Continuous reinforcement learning of energy management with deep q network for a power split hybrid electric bus, Applied Energy 222 (2018) 799–811, https://doi.org/10.1016/J.APENERGY.2018.03.104.
[16] Z. Zhu, S. Gupta, A. Gupta, M. Canova, A deep reinforcement learning framework for eco-driving in connected and automated hybrid electric vehicles (2021) 1–14.
[17] Q. Xue, X. Zhang, T. Teng, J. Zhang, Z. Feng, Q. Lv, A comprehensive review on classification, energy management strategy, and control algorithm for hybrid electric vehicles, Energies 13 (20) (2020) 5355, https://doi.org/10.3390/en13205355.
[18] S.M. Mousavi, G.M. Nikdel, Various battery models for various simulation studies and applications, Renewable and Sustainable Energy Reviews 32 (2014) 477–485, https://doi.org/10.1016/j.rser.2014.01.048.
[19] D. Ambuhl, L. Guzzella, Predictive reference signal generator for hybrid electric vehicles, IEEE Transactions on Vehicular Technology 58 (9) (2009) 4730–4740, https://doi.org/10.1109/TVT.2009.2027709.
[20] R.S. Sutton, A.G. Barto, Reinforcement learning, second edition: An introduction - complete draft, The MIT Press, 2018, pp. 1–3.
[21] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning, Nature 2015 518:7540 518 (7540) (2015) 529–533. doi: 10.1038/nature14236.
[22] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, 32nd International Conference on Machine Learning, ICML 2015 1 (2015) 448–456.
[23] R. Luo, F. Tian, T. Qin, E. Chen, T.Y. Liu, Neural architecture optimization, Advances in Neural Information Processing Systems (2018) 7816–7827.
[24] Z. Zhu, Y. Liu, M. Canova, Energy management of hybrid electric vehicles via deep q-networks, Proceedings of the American Control Conference (2020) 3077–3082, https://doi.org/10.23919/ACC45564.2020.9147479.
[25] C.Z. Chengzhao Yang, An energy management strategy of hybrid electric vehicl..., [Online; accessed 2021–09-09] (0). URL: https://www.ijeart.com/an-energy-management-strategy-of-hybrid-electric-vehicles-based-on-deep-reinforcement-learning.
[26] R. Lian, H. Tan, J. Peng, Q. Li, Y. Wu, Cross-type transfer for deep reinforcement learning based hybrid electric vehicle energy management, IEEE Transactions on Vehicular Technology 69 (8) (2020) 8367–8380, https://doi.org/10.1109/TVT.2020.2999263.
[27] H. van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, 30th AAAI Conference on Artificial Intelligence, AAAI 2016 (2015) 2094–2100.
[28] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings.
[29] W. Fedus, P. Ramachandran, R. Agarwal, Y. Bengio, H. Larochelle, M. Rowland, W. Dabney, Revisiting fundamentals of experience replay, 37th International Conference on Machine Learning, ICML 2020 PartF168147-4 (2020) 3042–3052.
[30] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A.S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser, J. Quan, S. Gaffney, S. Petersen, K. Simonyan, T. Schaul, H. van Hasselt, D. Silver, T. Lillicrap, K. Calderone, P. Keet, A. Brunasso, D. Lawrence, A. Ekermo, J. Repp, R. Tsing, Starcraft ii: A new challenge for reinforcement learning.
[31] Y. Li, H. He, J. Peng, H. Wang, Deep reinforcement learning-based energy management for a series hybrid electric vehicle enabled by history cumulative trip information, IEEE Transactions on Vehicular Technology 68 (8) (2019) 7416–7430, https://doi.org/10.1109/TVT.2019.2926472.
[32] C. Liu, Y.L. Murphey, Power management for plug-in hybrid electric vehicles using reinforcement learning with trip information, 2014 IEEE Transportation Electrification Conference and Expo: Components, Systems, and Power Electronics - From Technology to Business and Public Policy, ITEC 2014doi: 10.1109/ITEC.2014.6861862.
[33] EPA, Dynamometer drive schedules — us epa, [Online; accessed 2021–10-16] (0). URL: https://www.epa.gov/vehicle-and-fuel-emissions-testing/dynamometer-drive-schedules.
[34] C.M. Bishop, Neural networks for pattern recognition (1995) 482.
[35] V.B. Mbuwir, L. Vanmunster, K. Thoelen, G. Deconinck, A hybrid policy gradient and rule-based control framework for electric vehicle charging, Energy and AI 4 (2021), https://doi.org/10.1016/J.EGYAI.2021.100059 100059.
[36] T. Liu, B. Wang, W. Tan, S. Lu, Y. Yang, Data-driven transferred energy management strategy for hybrid electric vehicles via deep reinforcement learning.
[37] V. Mnih, A.P. Badia, L. Mirza, A. Graves, T. Harley, T.P. Lillicrap, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, 33rd International Conference on Machine Learning, 2016, 2016, pp. 2850–2869.
[38] X. Wang, C. Yuen, N.U. Hassan, N. An, W. Wu, Electric vehicle charging station placement for urban public bus systems, IEEE Transactions on Intelligent Transportation Systems 18 (1) (2017) 128–139, https://doi.org/10.1109/TITS.2016.2563166.
[39] H.M. Chung, W.T. Li, C. Yuen, C.K. Wen, N. Crespi, Electric vehicle charge scheduling mechanism to maximize cost efficiency and user convenience, IEEE Transactions on Smart Grid 10 (3) (2019) 3020–3030, https://doi.org/10.1109/TSG.2018.2817067.
[40] R. Yu, W. Zhong, S. Xie, C. Yuen, S. Gjessing, Y. Zhang, Balancing power demand through ev mobility in vehicle-to-grid mobile energy networks, IEEE Transactions on Industrial Informatics 12 (1) (2016) 79–90, https://doi.org/10.1109/TII.2015.2494884.