ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/369359944

Advanced Energy Management Strategies for Plug-In Hybrid Electric Vehicles via Deep Reinforcement Learning

Conference Paper · December 2022

DOI: 10.4271/2022-01-7109

CITATIONS
0

READS
16

2 authors, including:

Amr Mousa
AVL LIST GMBH

4 PUBLICATIONS **1** CITATION

SEE PROFILE

All content following this page was uploaded by Amr Mousa on 22 March 2023.

The user has requested enhancement of the downloaded file.

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

2022-01-7109 Published 22 Dec 2022

Advanced Energy Management Strategies for Plug-In Hybrid Electric Vehicles via Deep Reinforcement Learning

Amr Mousa and Gerhard Benedikt Weiss Virtual Vehicle Research GmbH, Co-Simulation & Software

Citation: Mousa, A. and Weiss, G.B., "Advanced Energy Management Strategies for Plug-In Hybrid Electric Vehicles via Deep Reinforcement Learning," SAE Technical Paper 2022-01-7109, 2022, doi:10.4271/2022-01-7109.

Received: 30 Jun 2022 Revised: 24 Sep 2022 Accepted: 14 Oct 2022

Abstract

Plug-in Hybrid Electric Vehicles (PHEVs) achieve significant fuel economy by utilizing advanced energy management strategies in controlling the power distribution decision in real-time. Traditional heuristic approaches bring no additional benefits, including efficiency and development cost, considering the increasing complexity in control objectives. This paper extends a previous study of the same problem (RL) and vehicle topology to develop a Reinforcement Learning agent by investigating the performance of state-of-the-art algorithms, such as Rainbow-DQN with its variants, PPO and A3C, against the baseline rule-based and Dynamic Programming (DP) strategies. The developed RL agent is optimizing challenging control objectives such as fuel economy, vehicle drivability and driver comfort. The Rainbow-DQN is studied separately to optimize the agent compared to all the algorithm variants and after wards, the best performing variant is compared to tuned PPO and A3C agents. Proper evaluation criteria is defined and the concerned agents are tested with nine different scenarios to examine the generalization capabilities and performance robustness. The results revealed that the A3C agent surplussed both the PPO and the Rainbow-DQN achieving a maximum performance of 98.43% of the DP with a robustness of 97.32% ± 0.78 for the other cycles and an average of 177.7 sec for each engine start compared to 96.3 sec for the rule-based approach. Furthermore, as a future work, the paper investigated and proposed a cloud-based training concept for automated scaled-up training, evaluation and deployment of RL policies for the (P)HEVs of the future.

Keywords

Plug-in Hybrid Electric Vehicles, Energy Management Strategy, Reinforcement Learning, Proximal Policy Optimization (PPO), Asynchronous Advantage Actor Critic (A3C), Deep Q-Network (DQN), Cloud-based Distributed Training.

Introduction

Motivation

In the recent decade, environmental concerns and the increasing global warming awareness led to strong governmental legislations to control the emission levels for the automotive manufacturers. EU passenger cars were restricted to 95 gCO_2/km in 2020, normalized to the New European Driving Cycle (NEDC), compared to 170 g/km in 2000 [1]. Plug-in Hybrid Electric Vehicles (PHEVs) offered a promising solution to minimize the emission level and fuel consumption of the conventional fuel vehicles while maintaining adequate range and less dependency on the electricity infrastructure compared to the electric vehicles. PHEVs are showing less than 2.2 l/100 km in the Worldwide Harmonized Light Vehicles Test Procedure (WLTP). The German market alone has 64% of the registered new electric passenger vehicles as PHEVs while they contributed to 45% of Europe's electrified passenger vehicles in 2020 compared to just 3.3% in 2011 [2].

Nevertheless, PHEVs have two different Energy Storage Sources (ESSs), the High-Voltage (HV) battery and the Internal Combustion Engine (ICE), which requires a sophisticated Energy Management Strategy (EMS) to coordinate and achieve significant improvement in the performance [3]. The automotive market witnessed revealing technologies recently such as autonomous driving, Intelligent Transportation Systems (ITS) and connected vehicles. Such technologies increase the control objectives of the vehicle systems and accordingly, hand-crafted traditional rule-based approaches are no longer bringing benefits due to having enormous

© 2022 A. Mousa, G.B. Weiß. Published by SAE International. This Open Access article is published under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits distribution, and reproduction in any medium, provided that the original author(s) and the source are credited.

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

parameters to calibrate. Additionally, the parameters are calibrated empirically based on expert knowledge which does not mean that the obtained results are optimal. Moreover, the availability of sensors and environmental data cannot be utilized efficiently with the current approaches. Therefore, the heuristic approach nevertheless provides avenues for improving the vehicle's energy efficiency and researching state-of-the-art intelligent control systems is becoming a must-have requirement for the future vehicles.

Literature Review

Dynamic Programing (DP), as an example for the optimization-based control approaches, provides an optimal solution for the EMS [4, 5, 6, 7]. DP has a non-causal property which means it requires a finite horizon with a defined velocity and torque profile in the future in order to obtain global optimality. Moreover, the developed strategy cannot be generalized to other cycles showing limited inferior capabilities and adaptability to complex driving cycles. The aforementioned drawbacks limited the applicability of DP to provide theoretical benchmarks for other real-time 'implementable' control techniques [8] and potentially guide certain other algorithms towards optimal solutions [9] in the PHEV domain.

Several researchers contributed to other optimization-based approaches such as Equivalent Consumption Minimization Strategy (ECMS) [10], Model Predictive Control (MPC) [11], Explicit MPC (eMPC) [12] and Particle Swarm Optimization-based nonlinear MPC strategy (PSO-based MPC) [13]. Such approaches offered feasible solutions to real-time controllers, although a trade-off between accuracy and real-time capability is to be considered. Tuning prediction horizon length and discrete-time sample highly affects the real-time performance of the MPC besides the selected vehicle model fidelity level, the optimization solver, and vehicle hardware capabilities as well [14].

Transformational era has begun with the expanding applications for machine learning that provided state-of-the-art solutions to various problems in many research fields. Reinforcement Learning (RL) is an advanced mathematical formulation to control problems utilizing machine learning techniques and algorithms. RL became a topic of much interest in the AI community after achieving substantial performance with superhuman scores in playing Atari games [15], Chess and Shogi in AlphaGo Zero program by DeepMind [16]. Consequently, extensive attention by the research community is given to RL-based EMS approaches in PHEVs due to their ability to learn control policies through trial and error without explicit programming to follow a certain strategy.

After the rapid development of deep learning and neural networks in the recent years, Deep Reinforcement Learning (DRL)-based EMSs witnessed a significant increasing intelligence. DRL algorithms can have value-based, policy-based or hybrid representation [17]. Value based algorithms tend to learn estimating the state-action pair value while the policy-based algorithms learn the policy directly by interacting with the environment and adjusting the probabilities of good and bad actions. Hybrid algorithms, often called Actor-Critic (AC) algorithms, evaluate the quality of actions using the critic while the actor decides on the action based on a probability distribution and derives the policy. AC algorithms are more stable than value-based algorithms, while they require fewer training samples than policy-based algorithms.

Deep Q-Network (DQN) is a value-based algorithm used by Zhu et al. in developing an EMS for a mild HEV and comparing the performance against DP and Adaptive-ECMS. The results revealed a significant improvement for the fuel consumption with a near optimal solution achieved by the agent [18]. Meanwhile, policy-based algorithms such as Proximal Policy Optimization (PPO) showed an average fuel reduction of 3.1% compared to the reference strategy when utilized by Hofstetter et al. in controlling the torque split of an HEV [19].

Furthermore, Liessner et al. utilized a Deep Deterministic Policy Gradient (DDPG) agent, which is a hybrid AC algorithm, to consider different drivers behavior which showed improvement in the fuel economy up to 96.3% of the DP optimal strategy [20]. Moreover, a continuous action space was employed by He et al. to control the power split ratio, engine speed and torque [21]. The DRL horizon was extended to include optimizing charging/discharging strategies [22], predicting vehicle speed and power demand trajectories, utilizing weather and traffic conditions in a Vehicle-to-everything (V2X) environment [23], and learning the drivers behavior cooperatively in a Vehicle-to-Vehicle (V2V) environment using asynchronous variants of standard RL algorithms [24].

Contribution

This paper extends the author's previous work that: 1) proposed an novel practical approach to incorporate the RL-agent into a real vehicle Hybrid Control Unit (HCU); 2) applied value-based tabular and deep Q-learning algorithms; 3) presented an Extended-DQN (E-DQN) agent that was tested and verified for generalization capabilities in an industrial High Fidelity Model (HFM) achieving up to 10.46% improvement in fuel economy [25, 26]. The same vehicle topology is considered in the present research while the functional architecture is explained in more details and applied to the other algorithms as well.

The main contributions of this research are:

- Developing a tailored training environment for the EMS problem of (P)HEVs mainly based on Ray RLlib and OpenAI Gym libraries with highly scalable and distributed computation capabilities.
- Investigating the applicability, performance and robustness of various state-of-the-art RL algorithms such as Rainbow DQN, PPO and A3C.
- Optimizing several control objectives simultaneously such as fuel economy, vehicle drivability and driver comfort measured in average engine run period.
- Challenging the generalization capabilities of the best performing A3C agent by inferring it into nine different test cases and evaluating against a well-defined evaluation criteria.

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

The experimental results reveal that the A3C agent surpassed the rule-based controller by achieving a maximum performance of 98.43% of the DP with a robustness of 97.32% ± 0.78 for the other test cases, and an average of 177.7 sec for each engine start compared to 96.3 sec for the rule-based approach.

Paper Outline

The remainder of this paper is organized as follows: **section 2** develops the vehicle model in python environment, based on the same P2 powertrain configuration previously used in [25, 26], and an improved basic logic derived from the existing HCU representation of the Simulink HFM. In **section 3**, the problem is formulated mathematically and an OpenAI Gym-based environment is created to enable a compatibility with RL open-source libraries such as Ray RLlib and Stable Baselines. **Section 4** presents the experimentation analysis and results of several state-of-the-art algorithms such as Rainbow-DQN with its variants, PPO and A3C against the baseline rule-based and DP strategies. Finally in **section 5**, the findings are summarized and the conclusion is drawn for this study. Moreover, the research team's future work of incorporating a cloud-based training cycle for automated scaled-up training, evaluation and deployment of RL policies is explained.

Vehicle Modeling

Powertrain Architectures

The architecture of a hybrid vehicle's powertrain is more complex compared to the traditional vehicles and it can be classified into three different powertrain architectures: series, parallel, and power-split (series/parallel) [27]. The power-split architecture considered by this research, shown in figure 1, makes it possible to achieve a synergy based on the advantages of the series and parallel architectures.

The ICE operates at an optimum efficiency and can provide propulsion solely, avoiding the accumulated energy conversion losses. The power-split architecture is more complex with three energy systems, several electrical and mechanical couplings in between, which underlines the need for an adequate EMS to control such complex transmissions efficiently.

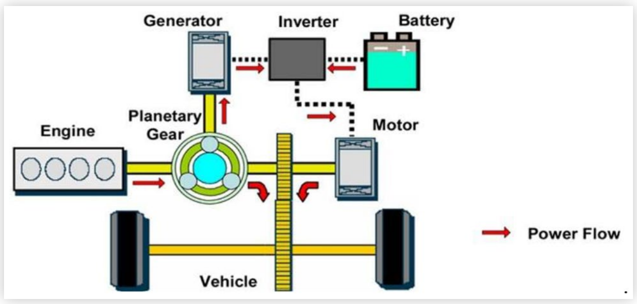
In the power-split configuration, other powertrain architectures are possible as well, as shown in figure 2, such as: 1) **P0** where HEVs feature a Belt Driven Starter/Generator (BSG) that is directly coupled to the ICE. 2) **P1** which locates the EM on the crankshaft, known as Integrated Starter/Generator (ISG). 3) **P2**, the configuration used throughout this research, which locates the EM on the gearbox input, after the clutch offering higher efficiency without ICE drag torque losses by disconnecting it from the EM, which makes pure electric drive possible. 4) **P3** which has the EM at the gearbox output. 5) **P4** on the contrary locates the EM on the driving axle, connected to the wheels all the time [16, 17, 18].

Hybrid Control Units

The fuel improvement in HEVs ranges from 10% in mild hybrids up to more than 30% for highly hybridized vehicles [29]. Providing the HCU with sophisticated strategies is needed to realize such a potential. As a vital component in the powertrain, HCU works coherently with several vehicle subsystems such as Human-Machine Interface (HMI) with the driver's demands, Transmission Control Unit (TCU), Engine Control Unit (ECU), Battery Management System (BMS), etc. The aforesaid systems send status signals, several limitations, and requests to the HCU as shown in figure 3. All the inputs, combined with the driving situation, are processed within the HCU and the target "optimum" settings for the drivetrain components are selected.

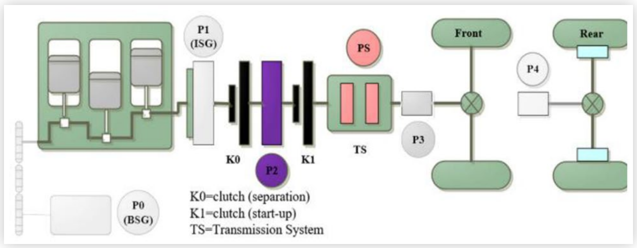
The HCU has a traditional EMS called Charge-Depletion Charge-Sustaining strategy (CDCS) that favors the use of electric propulsion up to the minimum State of Charge (SoC)

FIGURE 1 Hybrid vehicle power-split (parallel/series) configuration [27]



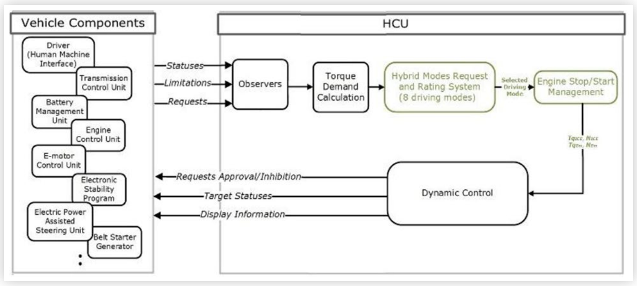
© A. Mousa, G.B. Weiß

FIGURE 2 The configurations of HEVs based on the EM location [28].



© A. Mousa, G.B. Weiß

FIGURE 3 Various input signals from vehicle components and output signals utilized by HCU [25, 26].


Vehicle Components
Driver (Human Machine Interface)
Transmission Control Unit
Battery Management Unit
Engine Control Unit
E-motor Control Unit
Electronic Stability Program
Electric Power Assisted Steering Unit
Belt Starter Generator
HCU
Statuses
Limitations
Requests
Observers
Torque Demand Calculation
Hybrid Modes Request and Rating System (8 driving modes)
Selected Driving Mode
Engine Stop/Start Management
Requests Approval/Inhibition
Target Statuses
Display Information
Dynamic Control

© A. Mousa, G.B. Weiß

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

level of the battery, then switching to ICE propulsion mode. It is preferred for PHEVs to run out of charge after a driving trip, where an external charging source shall be available to offer a battery charge. In this case, more electricity is utilized, which means more fuel is saved [30].

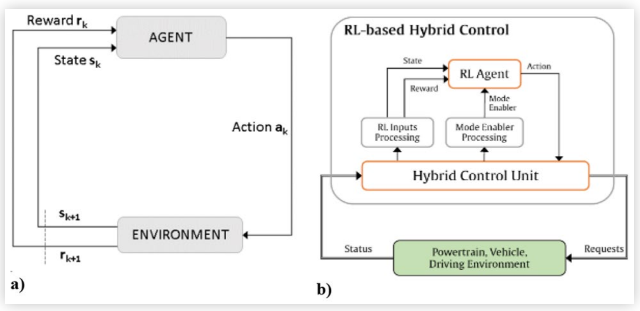
P2-HCU has a model to generate the quasi-stationary requests for the ICE, EM, and drivetrain components which are calculated in parallel for each hybrid mode. The operation modes are classified as fixed modes and free modes according to Ambühl et al. [31].

Fixed modes are selected by certain fixed rules inside the HCU and cannot be overridden by the EMS as follows: 1) **Additive Boost (acronym: AB {mode index: 1}):** this mode is selected when the maximum ICE torque cannot offer the demanded traction torque. The EM is started to supply the extra torque needed by the driver. **2) Recuperation (R {7}):** it is a regenerative breaking mode that slows down the vehicle with negative torque requests by utilizing the generator's negative torque. **3) Open Drive (OD {8}):** this mode is enabled when the vehicle is stationary by turning off the ICE and delivering power from the EM to the transmission oil pump that controls the clutches.

Free modes are selected upon the availability of several modes simultaneously. This means the system can be operated using any of the available modes and the EMS decision is required to achieve the optimization objectives. They are as follows: 1) **Conventional Drive (CD {0}):** the ICE is propelling the vehicle in this mode and supplying the low voltage auxiliaries. **2) Optimum Generation (OG {3}):** the ICE load point can be shifted to a more efficient location when the demanded torque is low, and the leftover power is consumed by the generator for charging the High Voltage (HV) battery. **3) Electric Drive (ED {6}):** this mode enables pure electric drive for the vehicle with a decoupled ICE.

The PHEV has several system constraints, safety, diagnostics, and protection limitations that shall be fulfilled by the EMS. The HCU meets such conditions by a heuristic approach which the developed strategy shall not override. Therefore, the conditions imposed by the system functionalities are defined as a mode enabler vector which is considered by the learning-based strategy to select the appropriate action each time step. In this way, only the modes that are allowed from the system point of view will be chosen and no violations to the system boundaries are possible.

FIGURE 4 a) The conventional and b) the proposed RL architecture [25, 26].



© A. Mousa, G.B. Weiß

The conventional RL architecture is shown in figure 4a where the agent freely selects the action according to its policy and the environment responds with the system state and reward. Figure 4b illustrates the modification proposed to the RL architecture. The main part of the HCU decides on the available modes for the EMS to select from, according to its rule-based control scheme, and constructs the 'mode enabler' logical vector. This vector is utilized by the RL agent, with the action masking technique presented by Vinyals et al. [32], to select the best action from the available actions according to the RL policy.

The selected action is further processed into the HCU remaining control functionalities and sent to the environment to be executed. The reader is referred to [25, 26] for the details of including the action masking into the RL agent decision process.

Vehicle Model

Having a reliable vehicle dynamics model that represents the plant to be controlled is crucial for developing the EMS. The AVL DSP department provided a high-fidelity plant model that brings a huge computational burden. Therefore, a quasi-static control-oriented vehicle model that maintains the vehicle physical causality is developed and used for the RL agent training and testing.

Model Parameters

The vehicle powertrain component parameters are listed in table 1.

Longitudinal Vehicle Model

The longitudinal vehicle model depends on the dynamics of the vehicle to calculate the power and torque demanded. Equation (1) governs the power demand P_d considering the vehicle moves on a road with

TABLE 1 Component parameters of the studied P2 PHEV model [25, 26].

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

© A. Mousa, G.B. Weiß

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

inclination θ where $T_{int_{loss}}$ is the drivetrain internal torque losses, and F_{ext} is external forces.

$$P_d = \left(T_{int_{loss}} \cdot \omega\right) + \left(F_{ext} \cdot V\right) \tag{1a}$$

$$F_{ext} = F_{aerodynamics} + F_{tire} + F_{gravity} + F_{inertia} \tag{1b}$$

$$F_{aerodynamics} = \frac{1}{2}\rho A C_d V^2 \tag{1c}$$

$$F_{tire} = mg\,cos\left(\theta\right)C_r \tag{1d}$$

$$F_{gravity} = mg\,sin\left(\theta\right) \tag{1e}$$

$$F_{inertia} = ma \tag{1f}$$

The 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* rolling resistance coefficient, m and V are vehicle mass and velocity respectively. The drivetrain $T_{int_{loss}}$ results from the internal mechanical friction losses which is modeled in the component models in the following section.

Drivetrain Component Models The models for drivetrain components are developed based on mathematical models and empirical performance maps. The ICE has a quasi-static fuel consumption model with neglected engine transients due to being much faster than the vehicle dynamics. The Brake-Specific Fuel Consumption (BSFC) map is plotted in figure 5 and governed by the function described in equation (2).

$$m_{fuel\,ICE} = f\left(\omega_{ICE}, T_{ICE}\right) \tag{2}$$

Equation (3) calculates the motor efficiency η_{EM} similarly for both modes for the EM, the motor drive in the positive torque region and the generator in the negative torque region, as shown in figure 6.

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

The delta of the HV battery SoC is calculated by equation (4). V_{oc}, R_{bat}, P_{bat}, Q_{bat} are the battery Open-Circuit Voltage (OCV), Internal Resistance (IR), terminals consumed power and capacitance respectively.

$$\Delta SOC = -(V_{oc} - \sqrt{V_{oc}^2 - 4P_{bat}R_{bat}})\,/\left(2Q_{bat}.R_{bat}\right) \tag{4}$$

For the sake of model simplicity, the battery pre-determined maps are used to estimate the OCV and IR only at 25 °C and the battery SoC thresholds are set to be [20%, 95%] for maintaining the battery health.

FIGURE 5 ICE BSFC map (g/kWh) [25, 26].

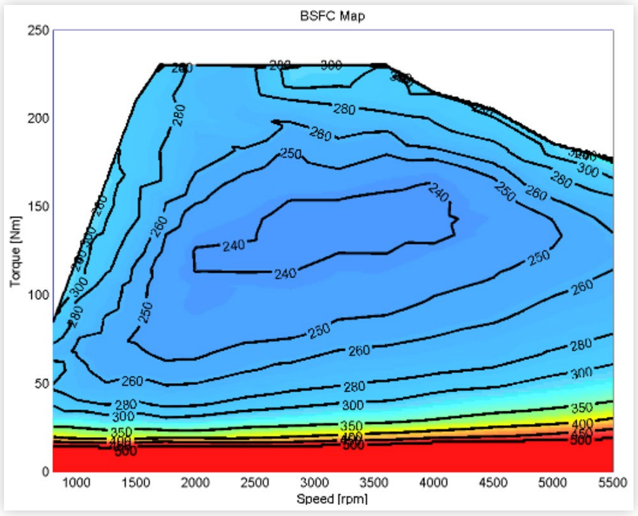
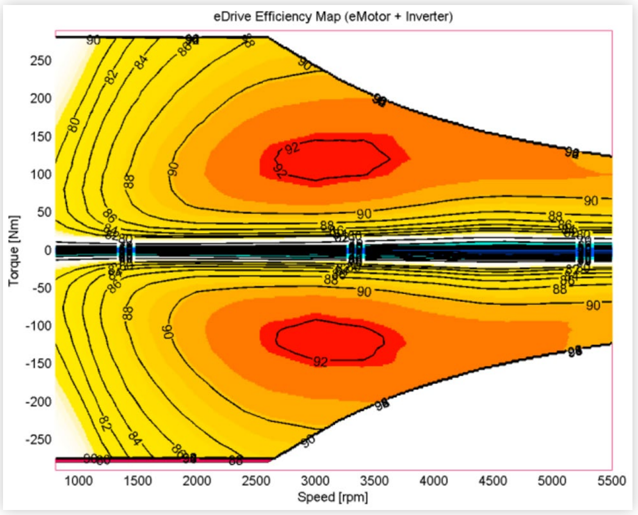


FIGURE 6 EM efficiency maps [25, 26].



Methodology

Reinforcement Learning is a revolutionary machine learning technique to solve complex control problems by rewarding desired behavior patterns and punishing undesired ones [17]. The RL agent is interacting with its environment, decide on actions to take and build knowledge through trial and error based on the environment feedback.

Problem Formulation

The EMS problem is formulated as Markov Decision Process (MDP) defined with the parameters $\left(\mathcal{S},\mathcal{A},\mathcal{P},\mathcal{R},\gamma,s_0\right)$ where $\mathcal{S}$ is the system states, $\mathcal{A}$ is the available actions, $\mathcal{P}$ is the transition probability function defined by $\mathcal{P} == \mathcal{S}\times\mathcal{A}\times\mathcal{S}\rightarrow\mathbb{R}$, the

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

reward function $\mathcal{R} == \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, the discount factor $\gamma \leq 1$ and the system initial state s_0. Learning a stochastic policy $\pi = \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the target of the RL algorithms. The policy shall maximize the cumulative future reward of an episode discounted by the γ factor $\Sigma_{t=0}^{T} \gamma^t r_t$ where r_t is the reward at time step t as a function of the current state s_t and the executed action a_t.

The EMS problem is bounded by a continuous state space defined by $s_k = [SoC_t, T_{d_t}, V_t, D_{rem_t}, E_{on_t}]$ where SoC_t is the current battery state of charge at time step t, T_{d_t} is the driver torque demand, V_t is the vehicle velocity, D_{rem_t} is the trip remaining distance and E_{on_t} is the engine on/off state. D_{rem_t} is included in the state space to express the agent progress in the episode to provide adequate segregation between states which was noticed to improve the agent learning significantly upon inclusion. E_{on_t} is included to give the agent insight over the engine current state because in the reward function definition, frequent engine switching (on and off) is penalized to comply with the driver comfort requirement. Furthermore, the EMS problem has a discrete action space where the control variable is the vehicle driving mode index $a_t \in \{0, 1, 3, 6, 7, 8\}$. The P2-PHEV's discrete-time control optimization problem, system constraints and reward function are described in equations (5) to (11).

$$\left[SoC_{t+1}, T_{d_{t+1}}, V_{t+1}, D_{rem_{t+1}}, E_{on_{t+1}}\right] = f\left(\left[SoC_t, T_{d_t}, V_t, D_{rem_t}, E_{on_t}\right], a_t\right),$$
$$t = 0,1,\ldots \mathrm{T}-1 \tag{5}$$

$$min \quad J_\pi(s_0) = E\left\{\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t)\right\} \tag{5}$$

Subject to:

$$a_t \in \{0,1,3,6,7,8\} \tag{6}$$

$$SoC_{min_t} \leq SoC_t < SoC_{max_t}, \qquad SoC_0 = SoC_{init} \tag{7}$$

$$T_{ICE_{min_t}} \leq T_{ICE_t} < T_{ICE_{max_t}} \tag{8}$$

$$T_{EM_{min_t}} \leq T_{EM_t} < T_{EM_{max_t}} \tag{9}$$

$$T_{Bat_{min_t}} \leq I_{Bat_t} < I_{Bat_{max_t}} \tag{10}$$

The reward function used in this research considers: 1) the fuel consumption, 2) the space-domain indexed SoC reference to guide the SoC depletion rate gradually in the entire trip and 3) the engine frequent switching on and off. The reward function is similar to the author's previous work [25, 26] but it was modified to exclude the hyperbolic tangent function as it showed increased unnecessary non-linearity for the agent to model without bringing additional benefit to the network convergence. The reward function is governed by equation (12) where χ, φ, ψ are set to 48, 172 and 1 respectively after careful tuning.

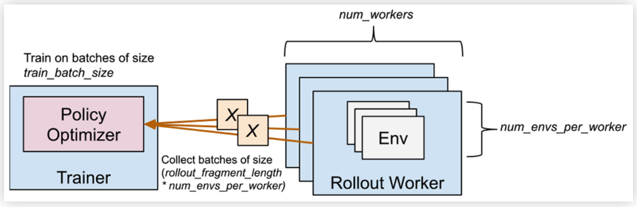
$$\begin{gathered} R_t = -\left(\chi \cdot \dot{m}_{fuel_t} + \varphi \cdot \left|SoC_t - SoC_{reference}\right| + \psi \cdot E_{on_t}\right), \\ SoC_{reference} = SoC_{initial} \cdot (1 - d/D) + d/D \cdot SoC_{final} \end{gathered} \tag{12}$$

The training framework was built based on Ray RLlib which is an open-source library providing scalable and production-level RL algorithms with highly distributed workloads and unified APIs [33]. RLlib provided our application with advanced capabilities such as vectorized environments for parallel rollouts, parallel trainings as shown in figure 7, multi-GPUs support, synchronous/asynchronous sampling capabilities, and RL training in online environment or offline using collected data from the HFM simulations.

The environment model is built using OpenAI gym interface which is a standardized API compatible with several RL libraries such as Stable Baselines, RLlib, Keras, …etc [34]. The environment contains most importantly the vehicle physical model and the basic rule-based HCU logic module as shown in figure 8. The latter handles the mode enablers and provides the vehicle's physical and parametric limitations. RLlib empowered our application with a unified interface to deal with action masking via the parametric action space for all the supporting algorithms. Other libraries were unofficially providing this feature such as Stable Baselines which only considered the PPO among all other algorithms till the date of writing this paper.

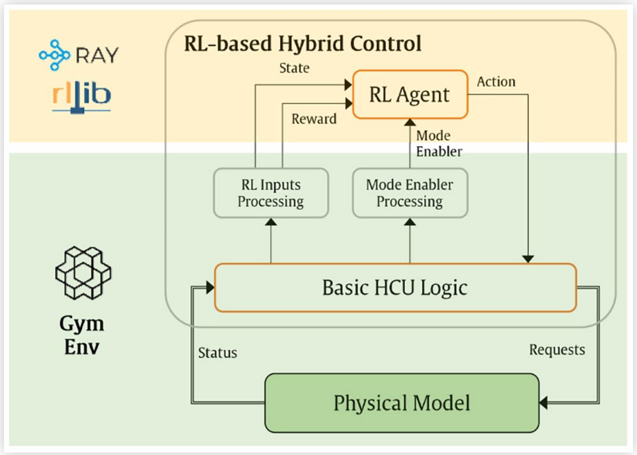
On the HFM implementation level, figure 9 illustrates the functional software architecture in more details. The normal operation of the HCU includes observing the vehicle state and requests in the Observers module. Afterwards, the traction torque is determined, and all quasi-stationary modes are calculated in parallel and the mode-enablers vector is created. The Mode Selection module handles the CDCS strategy and selects the mode accordingly which is processed to determine

FIGURE 7 RLlib training workflow [33].



© A. Mousa, G.B. Weiß

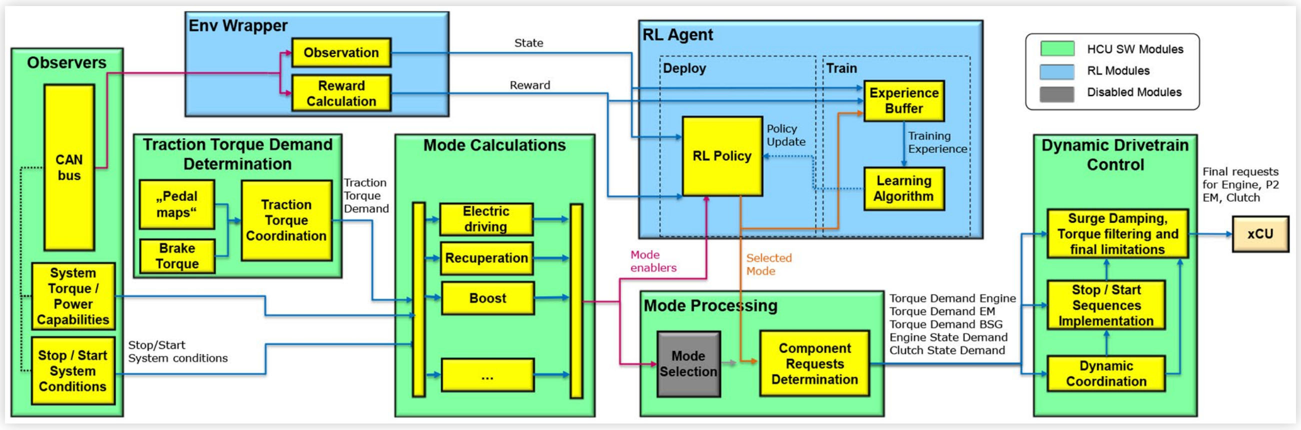
FIGURE 8 RL training framework for the PHEV problem [33].



© A. Mousa, G.B. Weiß

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

FIGURE 9 The functional architecture of RL-based HCU software.



© A. Mousa, G.B. Weiß

the powertrain component requests and they are coordinated with the corresponding Control Units (xCU).

On the other hand, the RL modules have its own Environment Wrapper which processes the vehicle observers and send the state and reward signals to the agent. The RL Agent module comprises the 'Deploy' unit which includes the policy and the 'Train' unit which has a storage memory for collecting experiences and the training algorithm. RL policy processes the mode enablers vector from the HCU's Mode Calculations module and decides on the action only within the available modes. Furthermore, RL policy obtains an updated policy from time to time from the learning algorithm to be executed for future decisions. Nevertheless, the 'Train' unit in the RL Agent is not involved in the real-time control decision, hence it can be decoupled from the HCU hardware unit and included on a separate xCU or hosted on the cloud for extended resources.

Deep Q-Network (DQN) Based EMS

DQN is a Q-Learning value-based algorithm that utilizes deep learning to empower the RL agent to deal with high-dimensional continuous spaces. The Q-learning algorithm comprises the action-value function (s,) which describes the expected discounted cumulative rewards till the end of the episode given the current state s, taking the action a and following the optimal policy π^* after. This policy is implicitly included in the optimal Q function $Q^*(s,a)$ by taking the action that has the highest Q-value in a certain state. $Q^*(s,a)$ shall satisfies the Bellman equation $Q^*(s,a) = r + \gamma \mathbb{E}_{(s'|,s|,a)}\left[\max_{a'} Q^*(s',a')\right]$ where s', a' are the next state and action respectively while γ is the discount factor and r is the reward.

DQN uses a neural network to approximate the $Q^*(s,a)$ function by training it to minimize the loss $\mathcal{L}(\theta) = \mathbb{E}_{(s,a,s',r)}\left[\left(r + \gamma max_{a'} Q(s',a';\theta) - Q(s,a;\theta)\right)^2\right]$. This algorithm witnessed developing different variants recently such as Double-DQN [35], Dueling-DQN [36], Prioritized Experience Replay (PER) [37] and n-steps bootstrapping [17]. Double-DQN uses two networks, the policy network with parameters θ and target network with parameters θ^-, to reduce the observed Q-values overestimation bias by decoupling the action selection from the target Q-value estimation, leading to a more stable training and an improved policy. The loss function is modified to estimate the Temporal Difference (TD) with the next Q-value from the target network instead, as described in equation (13).

Dueling-DQN improves the learning by allowing the network to better

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,s',r)}\left[TD^2\right] \tag{11}$$

$$TD = \left(r + \gamma max_{a'} Q\left(s',a';\theta^-\right) - Q\left(s,a;\theta\right)\right)$$

differentiate the action values. The Q-value is split into (s) which is the value of being at state s and the advantage $A(s,a)$ calculated by a separate stream in the network such that $Q(s,a)=V(s)+A(s,a)$. PER-based DQN prioritizes the experience sampling from the memory buffer according to their importance and significance instead or random sampling. The loss function for the training batch of instances m is modified according to equation (12) where $\beta \in (0 \leq \beta \leq 1)$ is the Boltzmann distribution that controls the amount of prioritization, ε is a small number to avoid the zero division, and α determines the priority sampling degree.

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m}(TD)^2 \cdot \left(\frac{\Sigma_k^N \left(TD_k + \varepsilon\right)^\alpha}{N.\left(TD_i + \varepsilon\right)^\alpha}\right)^\beta \tag{12}$$

N-steps bootstrapping unrolls the recursive Bellman equation for n times with replacing $Q(s',a')$ by the next reward and the discounted estimate at state s'' assuming a'' is chosen optimally or near optimally from the agent's current policy. All of these variants are combined in an outperforming state-of-the-art algorithm called Rainbow-DQN [38] which is experimented for the EMS problem and tuned to improve the performance.

Proximal Policy Optimization (PPO) Based EMS

In Policy Gradient (PG) methods, the policy $\pi(a|s;\theta)$ is directly parameterized as a distribution over actions. PPO is a

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

critic-based policy gradient method, which trades-off between sampling data from the rollouts and optimizing the surrogate objective function with stochastic gradient ascent. PPO has better sample complexity over the Trust Region Policy Optimization (TRPO) by removing the KL penalty and the need to make adaptive updates while maintaining the compatibility with stochastic gradient descent [39]. The objective function with adaptive KL-penalty is described in equation (13) where β is the KL coefficient and A_t is an estimate of the advantage function at timestep t.

$$\mathcal{L}(\theta) = \mathbb{E}_{(s_t, a_t, r_t)}\left[\frac{\pi(a_t|s_t;\theta)}{\pi(a_t|s_t;\theta_{old})} A_t - \beta KL[\pi_{old}, \pi]\right] \quad (13)$$

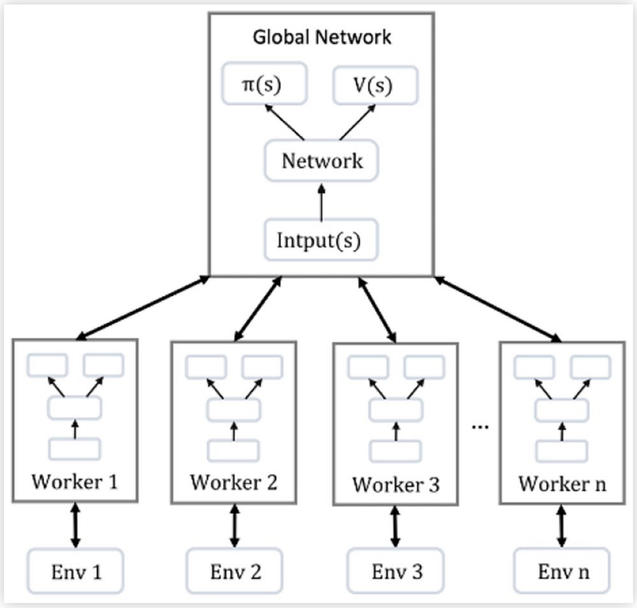
β is annealed in the RLlib implementation while the clipping can be used instead as well. The value function loss is defined in equation (14) where c_1, c_2 are the value function and the entropy coefficients respectively, while S denotes the entropy bonus and $L_t^{VF}(\theta)$ is the squared loss $\left(V_\theta(s_t) - V_t^{target}\right)^2$.

$$\mathcal{L}(\theta) = \hat{\mathbb{E}}_t\left[L_t^{CLIP} - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)\right] \quad (14)$$

Asynchronous Advantage Actor Critic (A3C) Based EMS

A3C is a policy gradient algorithm that combines learning an explicit policy $(a_t | s; \theta)$ to select the best action, and estimating the state-value function $V(s_t; \theta_v)$ to determine how good each state is. The algorithm uses a mix of n-step returns to update both the policy and the value-function. A3C utilizes a critic network to learn the value function while several actors are trained in parallel and sync the accumulated gradients with the global parameters for training stability. In figure 10, the algorithm launches several workers asynchronously where they interact with their own instance(s) of the environment, train their own copy of the network and share the results at the end of the simulation. This results in a more diverse in the experience collected from each worker, better learning efficiency and faster training.

FIGURE 10 A3C high-level architecture [40].



The network parameters (θ, θ_v) are updated with the loss function $\mathcal{L}(\theta, \theta_v)$ and the advantage function $A_t\ (s_t, a_t; \theta, \theta_v)$ with the hyperparameter k as given by equations (15) and (16) [24].

$$\mathcal{L}(\theta, \theta_v) = \mathbb{E}_{(s,a,s',r)}\left[A_t^2 - A_t \log\pi(a_t) - \beta\mathcal{H}(\pi)\right], \quad (15)$$

$$A_t(s_t, a_t; \theta, \theta_v) = \sum_{i=1}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{t+k}; \theta_v) - V(s_t; \theta_v) \quad (16)$$

The first term in the loss function optimizes the value function while the second optimizes the policy function. The exploration is encouraged by the third term via the entropy of the policy H(π) and the scaling parameter β.

Results and Discussion

Environment Model

In this research, three standard driving cycles are used: the New European Driving Cycle (NEDC), the Highway Fuel Economy Driving Schedule (HWFET) and the Urban Dynamometer Driving Schedule (UDDS) [41]. Moreover, an in-house cycle provided by the simulation team in AVL, presenting a commuter route in Graz city in Austria, was used for evaluating the agent performance as well. The developed vehicle model was validated by running Graz cycle and 6-NEDC (six adjoining NEDC cycles to exceed the vehicle's all electric range) on the HFM utilizing the CDCS strategy and recording the inputs to test the vehicle model with. The fuel consumption and SoC trajectory were determined and compared to the HFM results. The vehicle model showed negligible discrepancy compared to the HFM by maximum absolute error in fuel consumption of 74.1, 51.7, 26.6 ml and in SoC trajectory of 1.7, 2, 0.9% for NEDC cycle with 95%, 20% SoC_{init} and Graz cycle with 75% SoC_{init} respectively. Contrariwise, the vehicle model showed 2032x faster performance by achieving a step time of 0.33378 ms for one real time step compared to 678.36 ms of the HFM model.

Evaluation Criteria

To evaluate the performance of the developed RL agents, evaluation metrics have been defined as follows:

1. Fuel economy f_{eco} (%): the driving cycle fuel consumption (l/100 km) related to the DP performance on the same cycle and SoC_{init}. The fuel economy shall be improved over the CDCS performance on average.
2. Average engine run period (sec/E_{on}): the trip duration divided by the number of engine starts shall not

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

be lower than the CDCS performance on the same cycle.

3. Performance robustness (%): the strategies performance robustness is measured for N cycles by the mean μ and standard deviation σ of the fuel economoy f_{eco} as shown in equation (17).

$$\mu = \sum_{i=0}^{N-1} f_{eco_i} / N, \qquad \sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(f_{eco_i} - \mu\right)^2} \tag{17}$$

Rainbow DQN Study

A separate study is conducted in this section similar to the study carried out by Hessel et al. who introduced the Rainbow DQN agent based on studying the effect of each and every variant of the DQN algorithms such as Double-DQN, Dueling-DQN, Multi-step learning and PER [38]. All such variants were shown to be largely complementary and the Rainbow-DQN agent that combines all of them showed superior performance in terms of data efficiency and final performance.

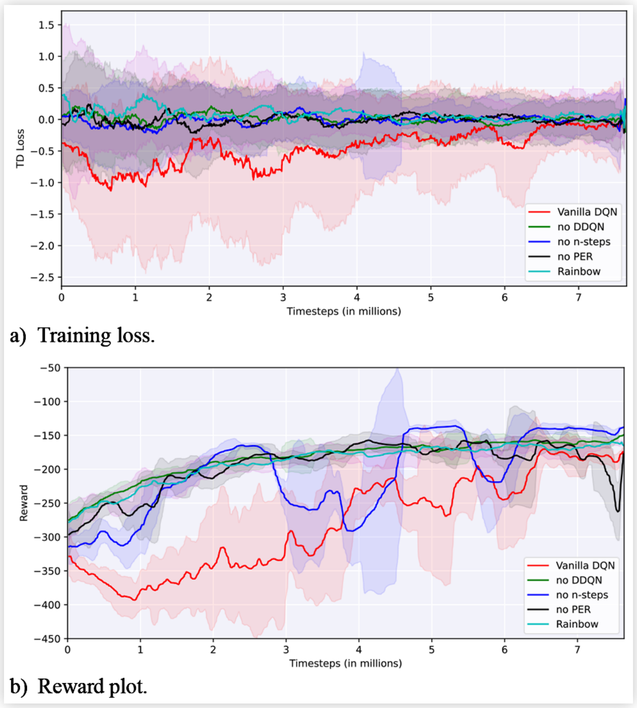
Similarly, the Rainbow-DQN agent is trained with our environment and compared to four other ablated variants to determine the effect of each component to the performance in the EMS problem. The four agents were 1) vanilla DQN, 2) no double, 3) no PER and 4) no n-step. Other components investigated by Hessel et al. such as Dueling-, Distributional and NoisyNet-DQN were not compatible with our environment utilizing the parametric action space, therefore were not included in the experimentation.

The agents were trained on HWFET cycle with 30% SoC_{init} for 7.75 million timesteps. The machine, which has a 4.00 GHz Core i7-6700K with 32 GB of RAM and a NVIDIA GTX 1080 Ti GPU, achieved an average time of 83 mins for each agent which has 7 parallel workers and 2 vector environments. The training hyperparameters used for all agents are listed in table A1 in appendix A. It is worthwhile to mention that the learning rate was halved for the Vanilla DQN agent after noticing a diverging performance.

The results in figure 11 revealed that all agents converged, and the policies were improving by time, however, the vanilla DQN agent had the least stable performance with higher oscillating loss trajectory.

On the contrary, 'no Double' agent achieved the best results followed by the 'Rainbow' with minor differences which shows that prioritized replay and n-step learning are the most two crucial components, and this agrees with the findings of Hessel et al. The removal of these components, especially the n-step learning, hurt the training stability and the final performance. Accordingly, the 'no Double' agent is recommended out of the DQN variants for the subsequent experimentations.

FIGURE 11 DQN study results.



© A. Mousa, G.B. Weiß

State-of-the-art RL Algorithms Comparison

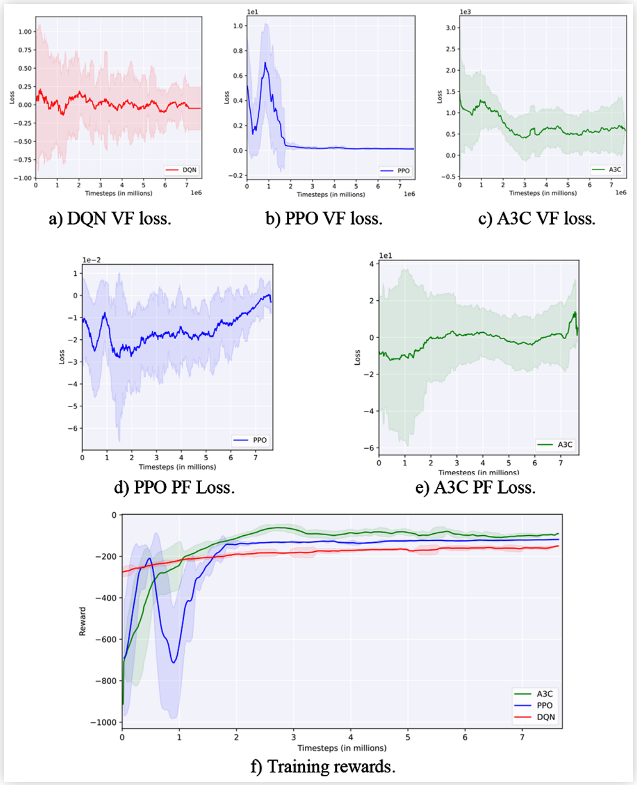
A comparative study is conducted to compare three model-free approaches: the selected DQN, PPO and A3C agents. Similar to the DQN study, the agents were trained on HWFET cycle with 30% SoC_{init} for 7.75 million timesteps utilizing the same architecture for the training.

The DQN agent utilized the Rainbow configuration but with single neural network for both policy update and target estimation. PPO agent used a shared network to represent the policy and value function which allows useful features to be shared while sacrificing interference between objectives a bit. On the other hand, A3C agent used a Generalized Advantage Estimator (GAE) with the value function, clipped gradients and entropy-based regularization. Employing 7 parallel workers and 2 vector environments for each agent, PPO achieved the longest training time of 138 mins compared to 86 mins for the DQN while A3C achieved the fastest training time of only 40.1 mins. The training hyperparameters used for all agents are listed in tables A1–A3 in appendix A.

Figure 12 shows the Value Function (VF) loss, the Policy Function (PF) loss and the training rewards as well. The three agents were developing a proper approximation to the environment return with a decreasing VF loss while the PPO and A3C (the PG-based algorithms) were improving the policy by time as shown in PF loss curves. Results in figure 12f revealed that both A3C and PPO agents started with a highly randomized policy achieving poor rewards at the beginning of the training while DQN started with a relatively better policy. A clear-cut outcome is that A3C quickly surpassed both DQN and PPO agents only after one million time steps while PPO achieved the second-best performance.

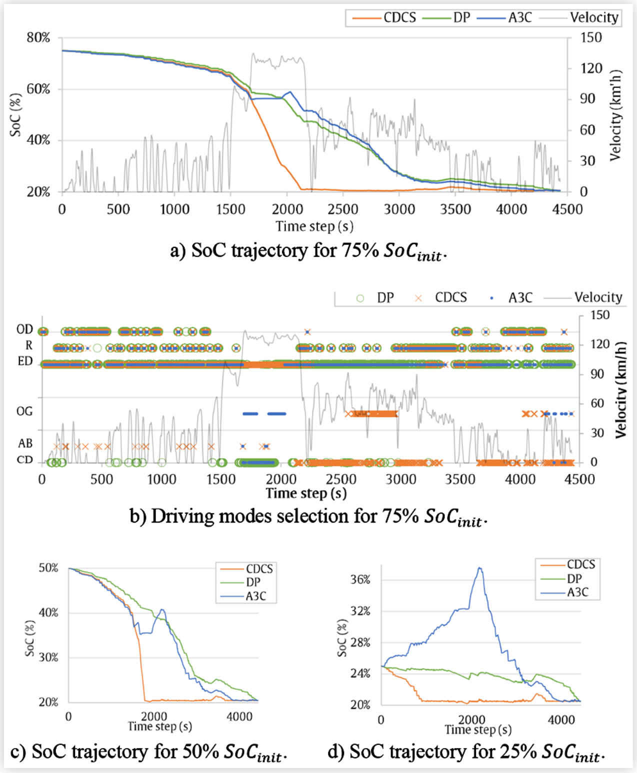
Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

FIGURE 12 The three agents comparative study results for the Value Function (VF) loss, Policy Function (PF) loss and the training rewards.



© A. Mousa, G.B. Weiß

FIGURE 13 Graz cycle evaluation results for the A3C agent compared to CDCS and DP.



© A. Mousa, G.B. Weiß

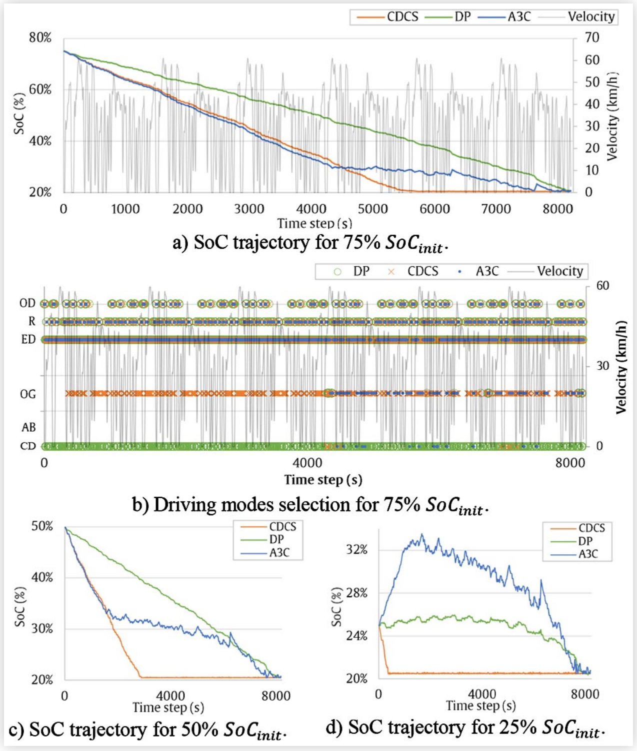
The Learned Policy Evaluation

This section is devoted to the learned policy characteristics of the trained agents. The generalization capabilities are evaluated by inferring the agent into new unseen cycles such as Graz, 6-UDDS and 4-HWFET, each with 25%, 50% and 75% of SoC_{init}. The numerical results are summarized in table A4 in appendix A for the three agents while the performance figures are shown in figures 13–15, referring only for the A3C as the best performing agent compared to CDCS and DP.

The performance figures shed the light on the developed strategy of utilizing the electric drive in the low speed regions and using the engine more on the high-speed segments. Moreover, the strategy tries to deplete the battery energy wisely through the whole trip compared to the CDCS which depletes the battery first, then sustains the SoC by using the engine for the rest of the trip. This agrees with the results of our previous work that developed the E-DQN agent for the same vehicle [25, 26], however the three agents noticeably performed better than the E-DQN.

The results in table A4 showed that the A3C agent outperformed the PPO, DQN and the CDCS relative to the DP. It showed the best performance improvement in the 6-UDDS cycle with 75% SoC_{init} by achieving 96.14% of the DP compared to 83.81% in the CDCS with a 12.33% improvement, while the second-best improvement was 10.43% in Graz cycle after achieving 98.43% of the DP performance for the same SoC_{init} level. The performance robustness of the A3C agent was a maximum of 97.32% ± 0.78 in the high SoC level compared

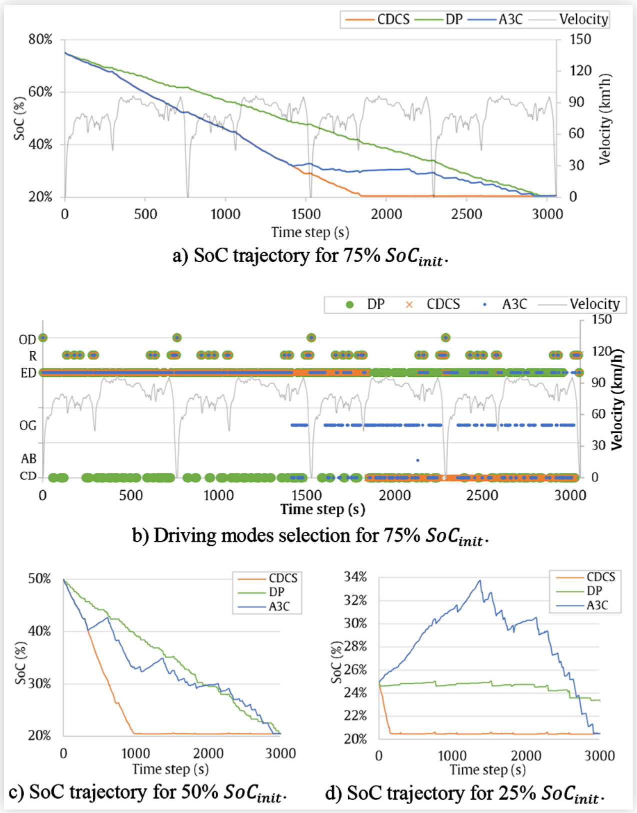
FIGURE 14 6-UDDS cycle evaluation results for the A3C agent compared to CDCS and DP.



© A. Mousa, G.B. Weiß

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

FIGURE 15 4-HWFET cycle evaluation results for the A3C agent compared to CDCS and DP.

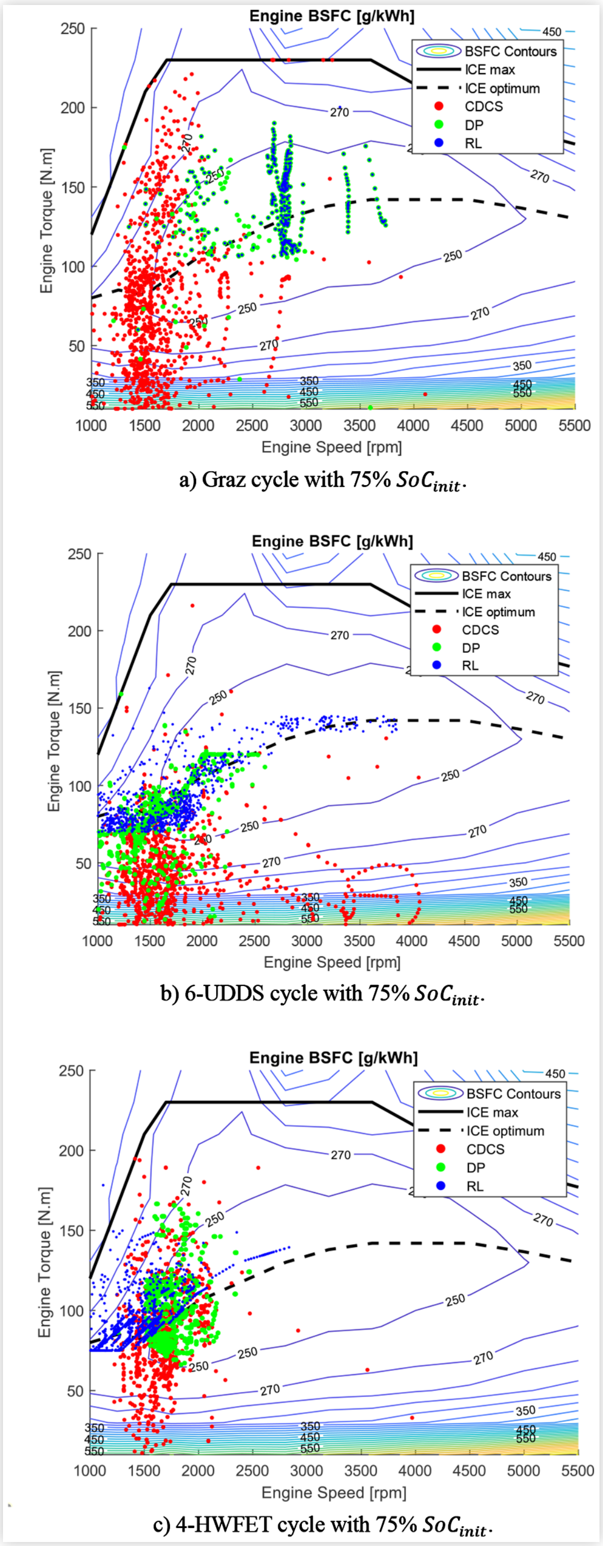


© A. Mousa, G.B. Weiß

to 96.74% ± 1.13 for the low SoC level. Moreover, it achieved a maximum engine run period of 494 sec/E_{on} compared to a minimum of 91 sec/E_{on} while the CDCS achieved 191 sec/E_{on} and 38 sec/E_{on} respectively.

The improvement in the fuel economy was significant, therefore the performance BSFC map for the three cycles with 75% SoC_{init} are plotted in figure 16 to further understand the strategy behind. The A3C agent located the load points closer to the DP and in more efficient region than the CDCS. The cognition of better engine utilization and higher fuel economy is confirmed by the DP and the RL adjacency, which is closer than between the DP and the CDCS.

FIGURE 16 BSFC maps for the A3C agent compared to DP and CDCS.



© A. Mousa, G.B. Weiß

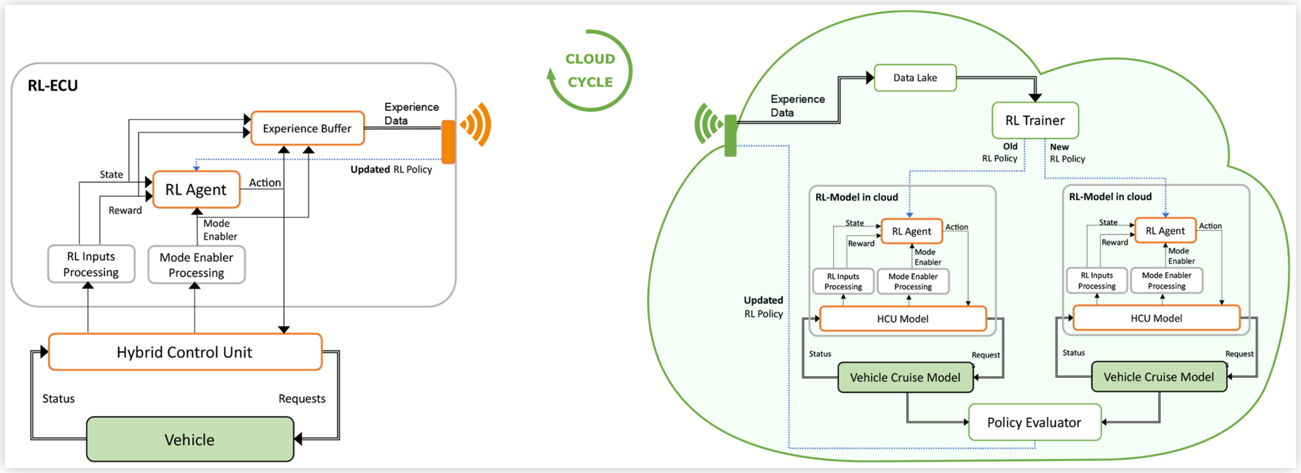
Conclusion

Concluding Remarks

Energy management strategy is an essential and vital component of the (P)HEVs to realize high level of efficiency and improvement in fuel economy. Recently, RL showed an increasing capability in solving complex problems alongside providing proven efficient strategies for (P)HEVs. In this paper, we extended our previous work in [25, 26] to develop a python-based framework and include three state-of-the-art RL algorithms, namely Rainbow DQN, PPO, and A3C, in a comparative study. A detailed experimental evaluation is presented to estimate the generalization capabilities and robustness. The results revealed that A3C is outperforming both Rainbow-DQN and PPO by achieving a maximum performance of 98.43% of the DP for the Graz cycle with high level of energy onboard to utilize, while the agent performance reached a robustness of 97.32% ± 0.78 in other cycles with the

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

FIGURE 17 Cloud environment cycle concept proposed by the research team.



© A. Mousa, G.B. Weiß

same initial SoC level. Moreover, the A3C was able to extend the average engine run period to 177.7 sec/E_{on} while CDCS achieved only 96.3 sec/E_{on}. This concludes that A3C agent maintained a satisfactory balance between the fuel economy and the driver comfort/vehicle drivability objectives.

Future Work

Connected vehicles in a networked environment is a research hotspot in the automotive industry. In the near future, the road vehicles would be capable of communicating with each other and possibly optimizing their performance in a collaborative approach. Accordingly, the future vehicle controller shall not only consider the onboard energy efficiency but also more objectives including other vehicles objectives in a coordinated and harmonized way. Advanced learning methods such as the asynchronous variants of RL algorithms [24] open the door for shorter computational time and realizing parallel distributed calculations.

Figure 10 showed the architecture of the A3C algorithm with several workers and global network. This workflow inspires developing a cloud-based training environment for a fleet of (P)HEVs equipped with RL agents. Our team is proposing a concept of deploying an intelligent RL agent into a test vehicle and connect it to the cloud for training purposes in a so called "cloud-environment cycle". As shown in figure 17 in the prototype vehicle, the RL-ECU is communicating with the vehicle's HCU for observing the states and deciding on an action for the next time step. The functionality is extended beyond the previous architecture, shown in figure 4b, to incorporate a memory for storing the collected experience and a communication module for cloud connectivity. The experience data is sent to the cloud from time to time by the vehicle and by others in the future scaled-up model. The received data by the cloud is stored in a 'Data Lake' that filters, categorizes, archives the data, and prepares for feeding the offline RL trainer.

Similar to the global network in A3C architecture, the RL trainer further trains the current policy with new diverse experience from all the agents connected to the cloud, and the trained 'new' policy and the old policy are deployed to two HFM model instances for testing and evaluation. The policy evaluator handles evaluating and comparing the results from both policies and decides on the updated RL policy accordingly. Such an updated RL policy is communicated to the fleet RL-ECUs to deploy and utilize for future decisions. This approach is embraced by our team for future research focusing on the development aspect besides prototyping it to examine the validity in real vehicle's environment.

References

1. International Council on Clean Transportation, "2020–2030 CO 2 Standards for New Cars and Light-Commercial Vehicles in the European Union," Icct, 2016.
2. Sandra Wappelhorst, G.B., "The Uptake of Plug-In Hybrid Electric Vehicles in Europe's Company Car Fleets: Trends and Policies | International Council on Clean Transportation," accessed May 5, 2021, https://theicct.org/blog/staff/phev-europe-company-cars-apr2021
3. Liu, W., *Introduction to Hybrid Vehicle System Modeling and Control* (John Wiley and Sons, 2013), doi:10.1002/9781118407400
4. O'Keefe, M.P. and Markel, T., "Dynamic Programming Applied to Investigate Energy Management Strategies for a Plug-In HEV1," in *22nd International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium and Exposition, EVS 2006*, 1035-1046, 2006.
5. Yang, Y., Hu, X., Pei, H., and Peng, Z., "Comparison of Power-Split and Parallel Hybrid Powertrain Architectures with a Single Electric Machine: Dynamic Programming Approach," *Appl. Energy* 168 (2016): 683-690, doi:10.1016/j.apenergy.2016.02.023.
6. Sundström, O. and Guzzella, L., "A Generic Dynamic Programming Matlab Function," in *Proceedings of the IEEE International Conference on Control Applications*, 2009, 1625-1630. doi: 10.1109/CCA.2009.5281131.
7. Wang, R. and Lukic, S.M., "Dynamic Programming Technique in Hybrid Electric Vehicle Optimization," in *2012*

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

IEEE International Electric Vehicle Conference, 2012. doi: 10.1109/IEVC.2012.6183284.

8. Li, Y., He, H., Peng, J., and Wang, H., "Deep Reinforcement Learning-Based Energy Management for a Series Hybrid Electric Vehicle Enabled by History Cumulative Trip Information," *IEEE Trans. Veh. Technol.* 68, no. 8 (2019): 7416-7430, doi:10.1109/TVT.2019.2926472.
9. Liu, T., Tang, X., Hu, X., Tan, W. et al., "Human-like Energy Management Based on Deep Reinforcement Learning and Historical Driving Experiences," July 2020, doi: 10.48550/arxiv.2007.10126.
10. Rezaei, A., Burl, J.B., and Zhou, B., "Estimation of the ECMS Equivalent Factor Bounds for Hybrid Electric Vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 6, pp. 2198-2205, Nov. 2018, doi: 10.1109/TCST.2017.2740836.
11. Borhan, H.A., Zhang, C., Vahidi, A., Phillips, A.M. et al., "Nonlinear Model Predictive Control for Power-Split Hybrid Electric Vehicles," *Proc. IEEE Conf. Decis. Control* (2010): 4890-4895, doi:10.1109/CDC.2010.5718075.
12. Taghavipour, A., Azad, N.L., and McPhee, J., "Real-Time Predictive Control Strategy for a Plug-In Hybrid Electric Powertrain," *Mechatronics* 29 (2015): 13-27, doi:10.1016/J.MECHATRONICS.2015.04.020.
13. Wang, Y., Wang, X., Sun, Y., and You, S., "Model Predictive Control Strategy for Energy Optimization of Series-Parallel Hybrid Electric Vehicle," *J. Clean. Prod.* 199 (2018): 348-358, doi:10.1016/J.JCLEPRO.2018.07.191.
14. Chen, H., "*Predictive Control Strategies of Plug-in HEVs*," 2019.
15. Mnih, V. et al., "Human-level control through deep reinforcement learning," *Nature* 518, no. 7540 (2015): 529-533, doi:10.1038/nature14236.
16. Silver, D. et al., "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," December 2017, doi: 10.48550/arxiv.1712.01815
17. Sutton, R.S. and Barto, A.G., *Reinforcement Learning, Second Edition: An Introduction - Complete Draft* (MIT Press, 2018), 1-3.
18. Zhu, Z., Liu, Y., and Canova, M., "Energy Management of Hybrid Electric Vehicles via Deep Q-Networks," in *2020 American Control Conference (ACC)*, vol. 2020, 3077-3082, 2020, doi: 10.23919/ACC45564.2020.9147479.
19. Hofstetter, J., Bauer, H., Li, W., and Wachtmeister, G., "Energy and Emission Management of Hybrid Electric Vehicles using Reinforcement Learning," *IFAC-PapersOnLine* 52, no. 29 (2019): 19-24, doi:10.1016/J.IFACOL.2019.12.615.
20. Liessner, R., Schroer, C., Dietermann, A., and Bäker, B., "Deep Reinforcement Learning for Advanced Energy Management of Hybrid Electric Vehicles," in *ICAART 2018 - Proceeding 10th International Conference on Agents and Artificial Intelligence*, vol. 2, 61-72, 2018, doi: 10.5220/0006573000610072.
21. Li, Y., He, H., Peng, J., and Zhang, H., "Power Management for a Plug-in Hybrid Electric Vehicle Based on Reinforcement Learning with Continuous State and Action Spaces," *Energy Procedia* 142 (2017): 2270-2275, doi:10.1016/J.EGYPRO.2017.12.629.
22. Hoang, D.T., Wang, P., Niyato, D., and Hossain, E., "Charging and Discharging of Plug-In Electric Vehicles (PEVs) in Vehicle-to-Grid (V2G) Systems: A Cyber Insurance-Based Model," *IEEE Access* 5 (2017): 732-754, doi:10.1109/ACCESS.2017.2649042.
23. Sun, C., Moura, S.J., Hu, X., Hedrick, J.K. et al., "Dynamic Traffic Feedback Data Enabled Energy Management in Plug-in Hybrid Electric Vehicles," *IEEE Trans. Control Syst. Technol.* 23, no. 3 (2015): 1075-1086, doi:10.1109/TCST.2014.2361294.
24. Mnih, V. et al., "Asynchronous Methods for Deep Reinforcement Learning," in *33rd International Conference on Machine Learning (ICML 2016)*, vol. 4, 2850-2869, February 2016, doi: 10.48550/arxiv.1602.01783.
25. Mousa, A., "Extended-Deep Q-Network : A Functional Reinforcement Learning-Based Energy Management Strategy for Plug-in Hybrid Electric Vehicles," *Eng. Sci. Technol. an Int. J.* (2022).
26. Mousa, A., "AI-based Energy Management Strategies for P2 Plug-in Hybrid Electric Vehicles," *M.S. thesis, AMM, FH OOE*, (2021), doi:10.5281/ZENODO.7684683, URL: https://doi.org/10.5281/zenodo.7684682.
27. Liu, J. and Peng, H., "Modeling and Control of a Power-Split Hybrid Vehicle," *IEEE Trans. Control Syst. Technol.* 16, no. 6 (2008): 1242-1251, doi:10.1109/TCST.2008.919447.
28. Xue, Q., Zhang, X., Teng, T., Zhang, J. et al., "Energy Management Strategy, and Control," 2020.
29. Sciarretta, A. and Guzzella, L., "Optimal Energy-Management Strategies," *IEEE Control Syst.*, vol. 27, no. 2, pp. 60-70, Apr. 2007, doi: 10.1109/MCS.2007.338280.
30. Hou, C., Xu, L., Wang, H., Ouyang, M. et al., "Energy Management of Plug-In Hybrid Electric Vehicles with Unknown Trip Length," *J. Franklin Inst.* 352, no. 2 (2015): 500-518, doi:10.1016/j.jfranklin.2014.07.009.
31. Ambuhl, D. and Guzzella, L., "Predictive Reference Signal Generator for Hybrid Electric Vehicles," *IEEE Trans. Veh. Technol.* 58, no. 9 (2009): 4730-4740, doi:10.1109/TVT.2009.2027709.
32. Vinyals, O. et al., "StarCraft II: A New Challenge for Reinforcement Learning," August 2017, accessed September 18, 2021, https://arxiv.org/abs/1708.04782v1
33. Liang, E. et al., "RLlib: Abstractions for Distributed Reinforcement Learning," in *37th International Conference on Machine Learning ICML 2018*, vol. 7, 4768-4780, December 2017, doi: 10.48550/arxiv.1712.09381.
34. Brockman, G. et al., "OpenAI Gym," June 2016, doi: 10.48550/arxiv.1606.01540.
35. van Hasselt, H., Guez, A., and Silver, D., "Deep Reinforcement Learning with Double Q-Learning," in *30th AAAI Conference on Artificial Intelligence AAAI 2016*, 2094-2100, September 2015, accessed July 21, 2021, https://arxiv.org/abs/1509.06461v3
36. Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H. et al., "Dueling Network Architectures for Deep Reinforcement Learning," in *33rd International Conference on Machine Learning (ICML 2016)*, vol. 4, 2939-2947, November 2015, doi: 10.48550/arxiv.1511.06581.

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

37. Schaul, T., Quan, J., Antonoglou, I., and Silver, D., "Prioritized Experience Replay," in *4th International Conference on Learning Representations ICLR 2016 - Conference Track Proceedings*, November 2015, accessed September 17, 2021, https://arxiv.org/abs/1511.05952v4

38. Hessel, M. et al., "Rainbow: Combining Improvements in Deep Reinforcement Learning," in *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, 3215-3222, October 2017, doi: 10.48550/arxiv.1710.02298.

39. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. et al., "Proximal Policy Optimization Algorithms," July 2017, doi: 10.48550/arxiv.1707.06347.

40. Zeng, F., Wang, C., and Ge, S.S., "A Survey on Visual Navigation for Artificial Agents with Deep Reinforcement Learning," *IEEE Access* 8 (2020): 135426-135442, doi:10.1109/ACCESS.2020.3011438.

41. EPA, "Dynamometer Drive Schedules | US EPA," accessed October 16, 2021, https://www.epa.gov/vehicle-and-fuel-emissions-testing/dynamometer-drive-schedules

Acknowledgments

The paper was written at Virtual Vehicle Research GmbH in Graz, Austria. The authors would like to acknowledge the financial support within the COMET K2 Competence Centers for Excellent Technologies from the Austrian Federal Ministry for Climate Action (BMK), the Austrian Federal Ministry for Digital and Economic Affairs (BMDW), the Province of Styria (Dept. 12) and the Styrian Business Promotion Agency (SFG). The Austrian Research Promotion Agency (FFG) has been authorized for the program management. They would furthermore like to express their thanks to their supporting industrial project partner, namely AVL List GmbH, with a sincere gratitude for Patrick Teufelberger, Enzo Harrer (E.H) and Simon Waltenberger for their collaboration and insightful feedback.

Contact Information

Amr Mousa, M.Sc, M.Eng
ORCID:0000-0002-4998-5336
amr.mousa@v2c2.at

Gerhard Benedikt Weiß, M.Sc
gerhard-benedikt.weiss@v2c2.at

Appendix A

Tables A1–A3 list the training hyperparameters used for the agent in the Rainbow-DQN study, the PPO and the A3C agents respectively. Furthermore, table A4 summarizes the evaluation results of the trained RL agents (the selected DQN, PPO and A3C) and the CDCS, with the DP as a benchmark for all the agents.

TABLE A1 DQN agents training hyperparameters.

Learning rate schedule [episode, μ]	[[0, 0.01], [10k, 0.0033]]	Train batch size	32
Epsilon schedule [episode, ε]	[[0, 1.0], [7.5k, 0.1]]	Buffer size	100,000
Target net update frequency	20,000	n-steps	3
Rollout fragment length	8	FCnet hiddens	[4x64]
Discount factor γ	0.99		

© A. Mousa, G.B. Weiß

TABLE A2 PPO agent training hyperparameters.

Learning rate schedule [episode, μ]	[[0, 0.003], [10k, 1e-5]]	Train batch size	4096
Rollout fragment length	200	Buffer size	100,000
Discount factor γ	0.99	Clip param	100
VF loss coefficient	1	FCnet hiddens	[2x256]
Entropy coefficient	0.01	Lambda λ	0.9
KL coefficient β	0.1	Horizon	5000
KL target	0.03		

© A. Mousa, G.B. Weiß

TABLE A3 A3C agent training hyperparameters.

Learning rate schedule [episode, μ]	[[0, 0.003], [10k, 1e-5]]	Train batch size	32
Rollout fragment length	20	Buffer size	100,000
Discount factor γ	0.99	Gradient clip	40
VF loss coefficient	0.5	FCnet hiddens	[2x256]
Entropy coefficient	0.01	Lambda λ	0.9

© A. Mousa, G.B. Weiß

Downloaded from SAE International by Amr Mousa, Wednesday, March 22, 2023

TABLE A4 Evaluation results of the trained RL agents and the CDCS compared to the DP. The best values achieved between the agents for each cycle is formatted in **bold.**

Algorithm	Initial SoC (%)	Driving Cycle	Terminal SoC (%)	Engine Run Period (sec/E_{on})	Fuel Consumption (l/100 Km)	DP Fuel Consumption (l/100 Km)	Fuel Economy (%)	Performance Robustness (%)		
A3C		Graz	20.54	**494**	**3.470**	3.402	98.00%	**97.32**	**±**	**0.78**
	75	6-UDDS	20.81	**249**	**3.252**	3.131	96.14%			
		4-HWFET	20.65	**102**	**3.833**	3.751	97.81%			
		Graz	20.54	**222**	6.155	5.964	96.80%	**97.02**	**±**	**0.66**
	50	6-UDDS	20.63	**114**	**6.208**	5.984	96.26%			
		4-HWFET	20.64	**113**	**6.490**	6.364	98.02%			
		Graz	20.53	**106**	8.904	8.612	96.61%	96.74	±	1.13
	25	6-UDDS	20.77	91	**9.067**	8.649	95.17%			
		4-HWFET	20.66	**109**	**9.119**	8.978	98.43%			
PPO		Graz	20.60	159	3.595	3.402	94.32%	94.66	±	1.73
	75	6-UDDS	20.75	90	3.368	3.131	92.42%			
		4-HWFET	20.86	39	3.854	3.751	97.26%			
		Graz	20.78	131	6.177	5.964	96.42%	95.19	±	1.87
	50	6-UDDS	20.77	64	6.440	5.984	92.38%			
		4-HWFET	21.15	35	6.580	6.374	96.77%			
		Graz	20.56	82	8.788	8.612	97.96%	**97.09**	**±**	**1.42**
	25	6-UDDS	20.74	57	9.085	8.649	94.96%			
		4-HWFET	20.64	48	9.127	8.978	98.34%			
DQN		Graz	20.50	114	3.602	3.402	94.11%	91.98	±	2.39
	75	6-UDDS	20.77	103	3.494	3.131	88.40%			
		4-HWFET	21.21	95	3.997	3.751	93.44%			
		Graz	20.57	139	**6.141**	5.964	97.03%	94.34	±	3.53
	50	6-UDDS	20.64	85	6.640	5.984	89.04%			
		4-HWFET	20.69	69	6.559	6.364	96.94%			
		Graz	20.51	79	8.879	8.612	96.90%	95.08	±	3.09
	25	6-UDDS	20.55	80	9.475	8.649	90.45%			
		4-HWFET	21.33	102	9.167	8.978	97.89%			
CDCS		Graz	20.54	139	3.825	3.402	87.57%	88.52	±	3.78
	75	6-UDDS	20.64	167	3.638	3.131	83.81%			
		4-HWFET	20.64	38	3.969	3.751	94.19%			
		Graz	20.54	62	6.414	5.964	92.45%	90.51	±	3.75
	50	6-UDDS	20.74	114	6.889	5.984	84.88%			
		4-HWFET	20.64	47	6.733	6.364	94.20%			
		Graz	20.54	67	**8.702**	8.612	98.95%	96.26	±	2.27
	25	6-UDDS	20.74	**191**	9.267	8.649	92.85%			
		4-HWFET	20.64	42	9.250	8.978	96.97%			

© A. Mousa, G.B. Weiß

© 2022 A. Mousa, G.B. Weiß. Published by SAE International. This Open Access article is published under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits distribution, and reproduction in any medium, provided that the original author(s) and the source are credited.

Positions and opinions advanced in this work are those of the author(s) and not necessarily those of SAE International. Responsibility for the content of the work lies solely with the author(s).

This paper is based upon a presentation at the *SAE 2022 Intelligent and Connected Vehicles Symposium*.

ISSN 0148-7191

View publication stats