Journal of Industrial and Management Optimization

JIMO, 22(12): 6709–6738.
DOI:10.3934/jimo.2026224
Received: 16 September 2026
Revised: 03 November 2026
Accepted: 05 November 2026
Published: 20 November 2026

https://www.aimspress.com/journal/jimo

Research article

Refined distributional soft actor-critic approach for optimal energy management in hybrid mining trucks

Xinxin Zhao[1], Xingrui Wen[1], Menglei Liu[1], Jingliang Duan[1], Lifang Zheng[1,*] and Nasser Lashgarian Azad[2]

[1] School of Mechanical Engineering, University of Science and Technology Beijing, Beijing 100083, China
[2] Department of Systems Design Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada

* **Correspondence:** E-mail: zhenglifang@ustb.edu.cn; Tel: 010-62332557.

Abstract: Energy management of multi-mode power-split hybrid mining trucks requires economical power allocation under coupled component constraints. This study developed a refined distributional soft actor-critic (R-SAC)-based energy management strategy that integrates established distributional learning refinements with state-dependent real-time interval mapping. The policy is trained offline, and its normalized outputs are converted into engine operating commands and powertrain-mode decisions subject to modeled component limits. On an unseen evaluation cycle, starting from a common initial state of charge (SOC) of 55.00%, R-SAC consumes 10.766 L of fuel and ends at SOC 49.83%. Actual fuel consumption is reduced by 8.1%, 4.9%, and 4.1% relative to rule-based energy management, equivalent consumption minimization, and PPO-MASK, respectively. Under a common nominal energy-equivalence convention, its equivalent fuel consumption is 11.255 L, the lowest among the four evaluated strategies. Hardware-in-the-loop implementation on an automotive-grade controller yields final absolute differences of 0.01 percentage points in SOC and 0.028 L in fuel consumption relative to offline simulation. These results support the fuel-saving potential of the integrated strategy and the consistency of its embedded implementation under the tested conditions.

Keywords: hybrid mining trucks; optimal control; reinforcement learning; deep reinforcement learning; energy management strategy; hardware-in-the-loop testing; neural network

Mathematics Subject Classification: 90C40, 93E20, 68T07

1. Introduction

Amid the global energy crisis, electrification, energy diversification, and fuel decarbonization are widely recognized as promising solutions and inevitable trends for the automotive and transportation sectors, including mining trucks [1,2]. Mining trucks are characterized by heavy payloads, challenging road conditions, and extended operational cycles [3]. This trend necessitates a shift toward cleaner and more sustainable hybrid systems, underscoring the need for continuous advancements in energy management technologies to support their evolution. Mining trucks often operate in harsh environments characterized by high temperatures, humidity, and dust. Hybrid systems, supported by electric drive assistance, reduce reliance on the engine, thereby improving system reliability under extreme conditions. Effective energy allocation and utilization directly influence both fuel economy and environmental impact. Consequently, designing efficient energy management strategies is of paramount importance [4].

Energy conservation is the primary objective of energy management strategies (EMS) for hybrid powertrain systems. Additionally, concerns regarding the safety, adaptability, reliability, longevity, and practicality of hybrid powertrains fall within the scope of energy management [5,6]. Given the constraints imposed by power sources and vehicle dynamics, EMS for power-split hybrid powertrain systems is formulated as an optimization problem. With the rapid advancement of machine learning, EMS can be classified into two categories: traditional EMS and learning-based EMS; the former primarily encompasses rule-based and optimization-based EMS. Optimization-based EMS employs solvers such as dynamic programming (DP) [7] and Pontryagin's minimum principle (PMP) [8]. Additionally, real-time optimization-based EMS utilizes algorithms like the equivalent consumption minimization strategy (ECMS) [9] and model predictive control (MPC) [10,11]. As a branch of machine learning, reinforcement learning (RL) excels in control-oriented scenarios, garnering substantial interest in the field of energy management [12].

Deep reinforcement learning combines neural-network function approximation with sequential decision-making and has been demonstrated in high-dimensional decision and control tasks [13,14]. Researchers have reviewed its applications to hybrid-powertrain energy management [15,16]. Performance depends on the training data, constraints, reward design, and evaluation conditions [17].

The Q-learning algorithm, a cornerstone of traditional reinforcement learning, gained significant popularity among researchers in the early stages of the field. Liu et al. [18] proposed a reinforcement learning–based adaptive energy management (RLAEM) strategy for hybrid tracked vehicles. By developing a control model with battery state of charge (SOC) and generator speed as state variables and engine torque as the control variable, this strategy autonomously learns optimal power allocation schemes for various driving schedules. Similarly, Zou et al. [19] introduced a real-time energy management strategy for hybrid tracked vehicles based on reinforcement learning, employing an innovative Markov chain recursive learning algorithm to optimize energy distribution between the engine, generator, and the battery. However, the Q-learning algorithm suffers from the curse of dimensionality due to its grid-based training approach, where higher fuel economy often leads to significantly increased training times. The powerful fitting capabilities of neural networks can replace Q-tables for information storage without causing substantial dimensional expansion, paving the way

for the emergence of DRL. Wu et al. [20] innovatively applied the deep Q-learning (DQL) algorithm to the energy management of hybrid buses, effectively addressing the limitations of traditional Q-learning in state discretization.

DQL approximates an action-value function with a neural network but normally selects among a finite set of actions. Fine discretization of engine operating points can therefore increase the action set substantially. Actor-critic methods such as DDPG instead parameterize continuous actions directly. For power-split hybrid buses, terrain information and DP-based pretraining have been incorporated into continuous-control energy management [21]. These examples motivate continuous-action formulations without implying that feedforward DQL or DDPG intrinsically provides long-term memory.

The energy management strategies for power-split hybrid systems also encompass a range of advanced deep reinforcement learning algorithms, including policy-generalization methods [22], hierarchical energy management [23], online-trainable asynchronous advantage actor-critic (A3C) [24], and dueling deep deterministic policy gradient (dueling DDPG) [25]. Adaptive energy allocation has also been investigated for an electric-hydraulic hybrid vehicle [26]. These methods enhance the optimization of energy distribution and control in complex hybrid powertrains. Furthermore, an improved SD3 algorithm refines energy management strategies for hybrid electric buses [27]. Additionally, battery-health-aware deep reinforcement learning has been investigated under naturalistic driving conditions [28]. Knowledge-guided two-population search [29], dual-Q-learning scheduling [30], and hierarchical dual-cache optimization [31] illustrate how soft-computing methods exploit problem structure. These scheduling applications provide methodological context; their objectives and action spaces differ from vehicle energy management.

Despite the progress of reinforcement learning in hybrid electric vehicle energy management, its practical application still faces challenges related to training stability, operational constraints, and real-time implementation [32]. Recent studies have investigated data-driven real-time dynamic programming for hybrid electric vehicle energy management [33], while action assessment and correction mechanisms have been developed to improve operational safety [34]. Constrained deep reinforcement learning has also been investigated for power-split hybrid electric vehicle energy management [35]. Moreover, existing action correction methods require further adaptation to the coupled operating characteristics of multi-mode power-split systems, where feasible engine speed, torque, and operating modes depend on instantaneous vehicle demands. These limitations motivate the integration of stable policy learning with state-dependent action constraints for practical energy management applications. LLM-assisted reinforcement learning has also been explored for battery-storage market bidding [36]. That framework uses external contextual information to adjust decision risk preferences, whereas the present EMS allocates power within a coupled vehicle drivetrain. The distinction concerns the task, information inputs, and execution interface; market-bidding gains are not treated as vehicle fuel-economy benchmarks. For a multi-mode power-split mining truck, the main challenge is to coordinate continuous engine operating points with discrete powertrain modes under changing power demand. Value-estimation errors can affect the learned allocation, whereas unconstrained network outputs can violate the coupled torque-speed limits. An EMS must therefore combine a suitable learning rule with a physically interpretable action interface. Embedded execution also requires separate verification of command consistency and timing.

This study develops an EMS for a nominal 110-ton-class mining truck, with a modeled total mass of 113,000 kg. The learner adopts the distributional SAC refinements introduced in [37], also known in that reference as DSAC-T or DSACv2; R-SAC denotes their use in the present EMS framework.

The application-specific contribution is the integration of this learner with powertrain-dependent interval mapping and its evaluation through simulation and controller-in-the-loop operation. The contributions are as follows:

Application-specific R-SAC formulation. The multi-mode power-split powertrain is represented as a sequential energy-allocation problem with continuous operating-point commands and a threshold-based mode decision, using the critic refinements adopted from [37].

Coupled feasible-action generation. A state-dependent interval mapper uses demanded torque, speed, gear, and component envelopes to translate normalized policy outputs into operating points within the modeled instantaneous limits when a feasible interval exists.

Embedded implementation assessment. The trained actor is reconstructed in Simulink and deployed on an automotive-grade vehicle control unit. HIL tests compare SOC, fuel consumption, component operation, and mode switching with offline simulation.

2. Preliminary work

2.1. Environment

2.1.1. Vehicle model

The hybrid mining truck investigated in this study integrates two permanent magnet synchronous electric machines (M/G1 and M/G2), a diesel engine, a lithium battery, a multi-planetary gear transmission system with two clutches and two transmission brakes, a main reducer, and a differential system. The transmission system adopts a power-split topology, supporting multiple operating modes, including pure electric and hybrid modes, and transmits power from the power sources to the wheels through torque and speed coupling. The detailed structure of the vehicle's transmission system is illustrated in Figure 1.

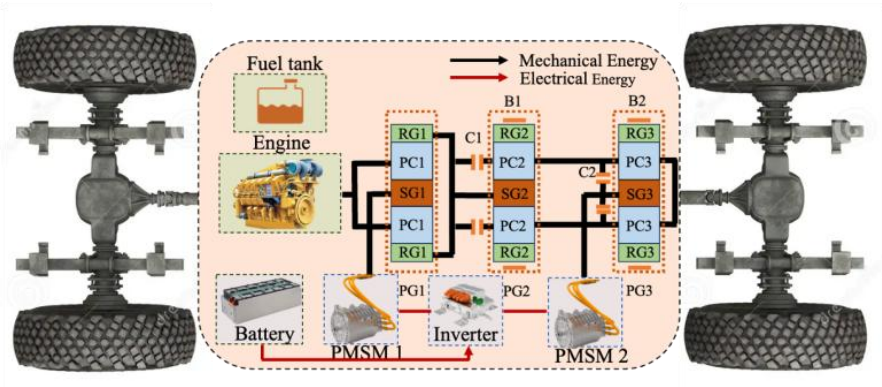


Figure 1. Architecture of the investigated system.

A quasi-static method was employed to develop a simulation model for calculating the energy consumption of the vehicle transmission system. The vehicle's power demand model, based on parameters such as demanded speed and acceleration, computes the required driving torque using a longitudinal vehicle dynamics model, as detailed in Eq. (1).

$$\begin{cases} T_{\mathrm{req}} = (F_{\mathrm{f}} + F_{\mathrm{i}} + F_{\mathrm{w}} + F_{\mathrm{j}})R_{\mathrm{w}}, \\ \omega_{\mathrm{req}} = \dfrac{v}{R_{\mathrm{w}}}, \\ F_{\mathrm{f}} = mgf\cos\beta, \\ F_{\mathrm{i}} = mg\sin\beta, \\ F_{\mathrm{w}} = \dfrac{1}{2}C_{\mathrm{D}}A\rho_{\mathrm{a}}v^2, \\ F_{\mathrm{j}} = \delta_{\mathrm{m}}m\dot{v}, \end{cases} \tag{1}$$

where m and R_{w} denote the mass of the vehicle and the radius of the wheel, f and β represent the rolling coefficient and the inclination of the road, C_{D}, A, and ρ_{a} represent the coefficient of air drag, the frontal area of the vehicle body, and the density of air, respectively, and δ_{m} is the conversion factor of the rotating mass of the vehicle. The demand torque T_{req} and speed ω_{req} of the vehicle can be obtained when the driving cycle is known based on the vehicle speed v and acceleration $\dot{v}$.

$$\begin{cases} T = A[T_{\mathrm{E}} \quad T_{\mathrm{M1}} \quad T_{\mathrm{M2}}]^{\mathrm{T}} i_g \\ \omega = B[\omega_{\mathrm{E}} \quad \omega_{\mathrm{M1}} \quad \omega_{\mathrm{M2}}]^{\mathrm{T}} / i_g \end{cases} \tag{2}$$

$$\mathrm{A} = \begin{bmatrix} \dfrac{\mathrm{K}_1(\mathrm{K}_2+1)}{\mathrm{K}_1+1} & 0 & -(\mathrm{K}_3+1) \\ \dfrac{\mathrm{K}_1(\mathrm{K}_2+1)}{\mathrm{K}_1+1} & 0 & -(\mathrm{K}_3+1) \\ \dfrac{\mathrm{K}_1(\mathrm{K}_2+1)}{\mathrm{K}_1+1} & 0 & 1 \\ -\dfrac{\mathrm{K}_1}{\mathrm{K}_1+1} & 0 & 1 \\ -\dfrac{\mathrm{K}_1}{\mathrm{K}_1+1} & 0 & 1 \end{bmatrix},$$

$$\mathrm{B} = \begin{bmatrix} \dfrac{(\mathrm{K}_1+1)}{\mathrm{K}_1(\mathrm{K}_2+1)} & -\dfrac{1}{\mathrm{K}_1(\mathrm{K}_2+1)} & 0 \\ \dfrac{(\mathrm{K}_1+1)}{\mathrm{K}_1(\mathrm{K}_2+1)} & -\dfrac{1}{\mathrm{K}_1(\mathrm{K}_2+1)} & 0 \\ \dfrac{(\mathrm{K}_1+1)}{\mathrm{K}_1(\mathrm{K}_2+1)} & -\dfrac{1}{\mathrm{K}_1(\mathrm{K}_2+1)} & 0 \\ \dfrac{(\mathrm{K}_1+1)}{\mathrm{K}_1} & -\dfrac{1}{\mathrm{K}_1} & 0 \\ \dfrac{(\mathrm{K}_1+1)}{\mathrm{K}_1} & -\dfrac{1}{\mathrm{K}_1} & 0 \end{bmatrix}$$

In Eq. (2), i_g is the fixed final-drive reduction ratio (22.07), not the operating-mode index. The column vectors T and ω contain the wheel-side torque and angular-speed values for the five operating modes. A and B are the corresponding 5-by-3 torque and kinematic coefficient matrices; their rows select the modes, and their columns refer to the engine, M/G1, and M/G2. T_E, T_{M1}, and T_{M2} are the respective shaft torques in N m; ω_E, ω_{M1}, and ω_{M2} are the corresponding angular speeds, expressed in rad/s for power calculations. K_1, K_2, and K_3 are the ring-to-sun tooth-number ratios of the planetary sets (2.14, 2.14, and 3.46). The superscript T denotes transpose. Multiplication by i_g for torque and division by i_g for speed convert between the transmission output and the wheels; subscript i in Eq. (3) selects the active mode.

The zero second column of A describes a reduced output-torque relation, not a zero M/G1 torque. Once planetary equilibrium and the operating mode are imposed, the three shaft torques cannot be specified independently. Eliminating the dependent reaction torque permits output torque to be written explicitly in terms of engine and M/G2 torque. M/G1 still provides reaction torque, supports engine-speed regulation, and exchanges electrical power. Its speed appears in B and its torque-speed product in Eq. (5). Torque and speed signs follow the shaft conventions of the topology; a negative torque alone does not establish the electrical operating state.

$$\{T_{req} = T_i, \omega_{req} = \omega_i\}, i = mode = 1, 2, \dots, 5 \tag{3}$$

Based on the system model, the operating conditions and power demands of the two motors are derived. According to the definition of the state of charge (SOC) for a lithium battery, the energy consumption model of the lithium battery is formulated as follows: P_{B} represents the power of the battery system, η denotes the efficiency of the motor and generator, and U_{oc}, R_{B}, $I(t)$, and Q indicate the open-circuit voltage, internal resistance, current of the battery system, and the capacity, respectively. In Eqs. (6)–(7), Q is in Ah, I(t) is in A, and Δt is in s; 3600 converts Ah to coulombs. The equations use fractional SOC; the results report percent SOC.

$$\begin{cases} P_{\mathrm{req}} = T_{\mathrm{req}}\omega_{\mathrm{req}}, \\ P_{\mathrm{req}} = P_{\mathrm{E}} + P_{\mathrm{B}}, \\ P_{\mathrm{E}} = T_{\mathrm{E}}\omega_{\mathrm{E}}. \end{cases} \tag{4}$$

$$P_{\mathrm{B}} = T_{\mathrm{M2}}\omega_{\mathrm{M2}}\eta_m^k - T_{\mathrm{M1}}\omega_{\mathrm{M1}}\eta_g^k, k = \begin{cases} 1, & \text{motor} \\ -1, & \text{generator} \end{cases}. \tag{5}$$

$$\begin{cases} P_{\mathrm{B}} = U_{\mathrm{oc}}I(t) - R_{\mathrm{B}}I^2(t), \\ I(t) = \dfrac{U_{\mathrm{oc}} - \sqrt{U_{\mathrm{oc}}^2 - 4R_{\mathrm{B}}P_{\mathrm{B}}}}{2R_{\mathrm{B}}}, \\ \mathrm{SOC}(t) = \mathrm{SOC}(t-1) - \dfrac{I(t)\Delta t}{3600Q} \end{cases} \tag{6}$$

The fuel consumption of the diesel engine and the power consumption of the lithium battery can be calculated according to the working state of the power source. When the torque and speed of the diesel engine are determined, the instantaneous fuel consumption $\dot{m}_{\mathrm{f}}$ can be calculated based on the test data.

$$\dot{SOC} \approx \frac{SOC(t) - SOC(t-1)}{\Delta t},$$

$$\dot{\mathrm{SOC}} = -\frac{I(t)}{3600Q} = -\frac{U_{\mathrm{oc}} - \sqrt{U_{\mathrm{oc}}^2 - 4R_{\mathrm{B}}P_{\mathrm{B}}}}{36002R_{\mathrm{B}}Q} \tag{7}$$

$$\dot{m}_{\mathrm{f}} = f_{\mathrm{E}}(T_{\mathrm{E}}, \omega_{\mathrm{E}}). \tag{8}$$

The plant is a quasi-static model for cycle-level energy-management evaluation. Combustion transients, clutch engagement dynamics, and battery thermal and aging dynamics are outside its scope. Nominal vehicle and component parameters are listed in Table A1 of Appendix A.

Engine BSFC and M/G1 and M/G2 efficiency maps were obtained from component bench tests, with linear interpolation between measured engine operating points. Battery open-circuit-voltage versus SOC characteristics were parameterized from calibration tests. These measurements supply the component maps rather than an independent validation dataset.

Integrated-model verification used the rule-based EMS over both prescribed cycles. Reference and simulated speed were compared, followed by checks of gear/mode sequences and engine/motor torque-speed limits. Shaft power was converted to electrical power using the appropriate motoring or generating efficiency; battery current and fuel flow were then integrated on a common time base to obtain SOC and cumulative fuel. These checks establish numerical consistency, not independent whole-vehicle accuracy. HIL testing in Section 4.3 separately assesses controller implementation using the simulated plant.

2.1.2. Scene

According to the multi-mode hybrid engineering vehicle working scenario, the information on the vehicle's speed and hybrid system working mode determines the information on the vehicle's driving status. Hybrid mining dump trucks derive all their energy from fuel and batteries. Since it does not have an external power source, energy-efficient solutions must ensure fuel efficiency and stable battery power. This study incorporates information on the operating condition to consider variations in SOC from a macro perspective and introduces nonparametric rewards to increase generality. The reward function is represented by the following equation:

$$\mathrm{Reward} = -(\dot{m}_{\mathrm{f}}\mathrm{SOC}(t) + e^{(\frac{i}{n})-1}|\mathrm{SOC}(t) - \mathrm{SOC}_r|\dot{\mathrm{SOC}}) \tag{9}$$

where SOC_r is the reference SOC value, which is 0.55 (55%), $\dot{\mathrm{SOC}}$ denotes the SOC change rate, approximated from successive samples over Δt, and $\frac{i}{n}$ denotes the jobs that have been run and, to reflect the increasing importance of the SOC ratio and the effect of selection over time, the percentage of cycles to the total number of work cycles.

Therefore, the EMS problem formulation consists of defining the objectives, constraints, and variables required to effectively control and optimize energy in the system. The physical constraints established are as follows:

$$\begin{cases} \mathrm{SOC}_{\min} \le \mathrm{SOC} \le \mathrm{SOC}_{\max}, \\ T_{\mathrm{E_min}} \le T_{\mathrm{E}} \le T_{\mathrm{E_max}}, \\ \omega_{\mathrm{E_min}} \le \omega_{\mathrm{E}} \le \omega_{\mathrm{E_max}}, \\ T_{\mathrm{M1,2_min}} \le T_{\mathrm{M1,2}} \le T_{\mathrm{M1,2_max}}, \\ \omega_{\mathrm{M1,2_min}} \le \omega_{\mathrm{M1,2}} \le \omega_{\mathrm{M1,2_max}}, \end{cases} \tag{10}$$

2.2. Soft actor-critic

2.2.1. Brief review of SAC

SAC is an off-policy actor-critic method that optimizes an entropy-regularized return [38]. Experience replay allows past transitions to be reused, and the stochastic actor provides a continuous-action exploration mechanism. Entropy regularization does not change SAC into a combination of on-policy and off-policy learning.

Entropy, a fundamental concept in information theory, quantifies the uncertainty of a random variable. For a continuous random variable with a probability density function, its differential entropy is derived from the functional expression

$$\mathcal{H}(X) = -\int p(x) \ln p(x) dx. \tag{11}$$

In reinforcement learning, the randomness of a policy π under a given state s is typically quantified by its entropy. The maximum entropy reinforcement learning framework augments the traditional reward maximization objective by incorporating a requirement to maximize policy entropy, thereby enhancing exploration capabilities. The optimization objective is formulated as the sum of the standard cumulative reward and an entropy regularization term, expressed as:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}(r(\tau) + \alpha \mathcal{H}(\pi(\cdot\,|s))). \tag{12}$$

where α represents the temperature coefficient, which balances the relative importance of entropy maximization and reward maximization. A larger α increases the policy's exploratory behavior, reducing the likelihood of the objective function converging to a local optimum, and vice versa. The same adaptive α is retained in the soft-value, policy-improvement, and actor-loss expressions. For the objective function augmented with an entropy regularization term, the critic network is enhanced to account for entropy, with its Bellman equation formulated as:

$$\Gamma^\pi Q(s_t, a_t) \triangleq r(s_t, a_t) + \gamma \mathbb{E}[V(s_{t+1})], \tag{13}$$

where

$$V(s_t) = \mathbb{E}_{a_t \sim \pi}[Q(s_t, a_t) - \alpha \log(\pi(a_t|s_t))]. \tag{14}$$

In the ideal soft-policy-improvement formulation, the policy is moved toward the exponential distribution induced by the soft action values by minimizing a KL-divergence objective. With finite

neural networks, sampled gradients, and approximate critics, an individual update is not guaranteed to improve the deployed policy.

$$\pi_{new} = arg\ min_{\pi'} D_{KL}(\pi'(\cdot\,|s)\ ||\frac{exp(\frac{Q^{\pi_{old}}(s,\cdot)}{\alpha})}{Z^{\pi_{old}}(s)}). \tag{15}$$

The updates for the state-value function and the action-value function are performed using the mean squared error method, as shown in Eqs. (16) and (17), with their respective derivatives given in Eqs. (18) and (19):

$$J_V(\psi) = \mathbb{E}[\frac{1}{2}(V_\psi(s_t) - \mathbb{E}_{a_t\sim\pi}(Q(s_t,a_t) - \alpha\log(\pi(a_t|s_t))))^2] \tag{16}$$

$$J_Q(\theta) = \mathbb{E}[\frac{1}{2}(Q_\theta(s_t,a_t) - \widehat{Q}(s_t,a_t))^2] \tag{17}$$

$$\nabla_\psi J_V(\psi) = \nabla_\psi V_\psi(s_t)(V_\psi(s_t) - Q_\theta(s_t,a_t) + \alpha\log(\pi_\phi(a_t|s_t))) \tag{18}$$

$$\begin{aligned}\nabla_\theta J_Q(\theta) = \nabla_\theta Q_\theta(s_t,a_t)(Q_\theta(s_t,a_t) - \\ r(s_t,a_t) - \gamma\overline{V}_\psi(s_{t+1}))\end{aligned} \tag{19}$$

where $\overline{V}_\psi(s)$ represents the target value function network.

SAC uses a reparameterized action of the form in Eq. (20) to differentiate the actor objective through a sampled action. Independent Gaussian noise is transformed using the state-dependent policy mean and standard deviation, so the resulting action still depends on the policy parameters. Reparameterization supplies a differentiable sampling path; off-policy learning follows from the use of replayed transitions, rather than independence between the action and the policy.

$$a = \tanh(\mu_\phi + \zeta\sigma_\phi), \tag{20}$$

where ζ represents the noise, $\zeta \sim N(0,1)$, and μ and σ denote the mean and the standard deviation. The hyperbolic tangent function, tanh, is employed to constrain the actions within a specific range.

The policy network after reparameterization is shown in Eq. (21), with its objective function and gradient function given by Eqs. (22) and (23), respectively.

$$a_t = f_\phi(\zeta; s_t). \tag{21}$$

$$J_\pi(\phi) = \mathbb{E}[\alpha\log\pi_\phi(f_\phi(\zeta;s_t)|s_t) - Q_\theta(s_t, f_\phi(\zeta;s_t))]. \tag{22}$$

$$\nabla_{\phi} J_{\pi}(\phi) = \nabla_{\phi} \alpha \log \pi_{\phi}(a_t|s_t) + (\nabla_{a_t} \alpha \log \pi_{\phi}(a_t|s_t) - \nabla_{a_t} Q(a_t|s_t)) \nabla_{\phi} f_{\phi}(\zeta; s_t). \tag{23}$$

The temperature coefficient can be determined automatically by using Eq. (24),

$$L(\alpha) = \mathbb{E}[-\alpha \log \pi(a_t|s_t) - \alpha \mathcal{H}_0], \tag{24}$$

where $\mathcal{H}_0$ represents the target entropy, which dynamically balances exploration and exploitation in the policy. Specifically, when the entropy of the policy, $\mathcal{H}(\pi(\cdot\,|s_t))$, falls below the target entropy, the algorithm automatically increases the temperature coefficient α, thereby enhancing the weight of the entropy regularization term in the optimization objective, which encourages greater exploration behavior. Conversely, when the policy entropy exceeds $\mathcal{H}_0$, the value of the temperature coefficient is reduced, shifting the optimization focus toward maximizing cumulative rewards. This adaptive mechanism intelligently maintains the policy entropy near an ideal level, ensuring sufficient exploration while preserving the policy's performance.

2.2.2. Distributional SAC

A distributional critic models the soft return distribution rather than only its mean. In the Gaussian parameterization adopted here, the critic predicts both a mean Q-value and a standard deviation. Duan et al. [37] introduced refinements to address the additional gradient variability introduced by distributional learning and to reduce sensitivity to the return scale in the reported benchmark tasks. These properties motivate the present application, but do not guarantee convergence for every vehicle model or reward formulation.

$$Q^{\pi}(s,a) = \mathbb{E}[Z^{\pi}(s,a)] \tag{25}$$

The Bellman equation for the soft value function is expressed as follows:

$$\Gamma^{\pi} Z^{\pi}(s_t,a_t) = r(s_t,a_t) + \gamma[Z^{\pi}(s_{t+1},a_{t+1}) - \alpha \log \pi(a_{t+1}|s_{t+1})]. \tag{26}$$

Similar to the policy function, the soft value function is modeled as a Gaussian distribution, as shown in Eq. (27), and is trained using the reparameterization technique. Its objective function and gradient functions are given by Eqs. (28) and (29), respectively.

$$Z_{\theta}(\cdot\,|s,a) = N(Q_{\theta}(s,a), \sigma_{\theta}(s,a)^2) \tag{27}$$

$$J_Z(\theta) = \mathbb{E}[D_{KL}(\Gamma^{\pi_{\theta'}} Z_{\theta'}(\cdot\,|s,a), Z_{\theta}(\cdot\,|s,a))] \tag{28}$$

$$\nabla_\theta J_Z(\theta) = \mathbb{E}[-\frac{(\Gamma^{\pi_\theta{}'} Z(s,a) - Q_\theta(s,a))\nabla_\theta Q_\theta(s,a)}{\sigma_\theta(s,a)^2} - \frac{((\Gamma^{\pi_\theta{}'} Z(s,a) - Q_\theta(s,a))^2 - \sigma_\theta(s,a)^2)\nabla_\theta \sigma_\theta(s,a)}{\sigma_\theta(s,a)^3}] \tag{29}$$

where $Q_\theta(s,a)$ represents the mean, and $\sigma_\theta(s,a)$ represents the standard deviation.

3. Refined-SAC optimization-based EMS framework

The EMS requires continuous selection of engine operating points and accounting for their effect on subsequent battery SOC. A coarse discrete-action learner would require a prescribed speed-torque grid, whereas a continuous actor can select operating points within state-dependent bounds. SAC additionally allows experience reuse during offline training and stochastic exploration of alternative allocations [38]. These features motivate its selection, rather than an assumption that it universally outperforms PPO, TD3, or ECMS. Figure 2 shows the offline training framework.

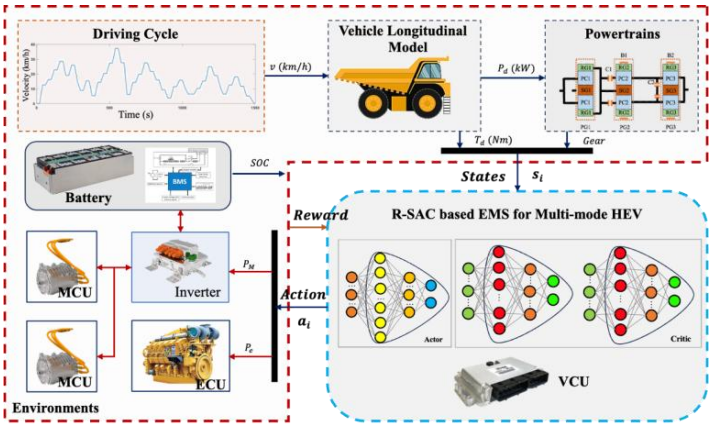


Figure 2. Schematic framework of the offline training of the R-SAC-based EMS.

R-SAC is used because expectation substitution, twin return-distribution critics, and variance-related gradient scaling target value-estimation difficulties in the selected learning framework [37]. The interval mapper addresses a different requirement: translating the learned decision into a feasible operating point of the coupled drivetrain. Their combination is therefore motivated by complementary learning and execution roles. The component selection follows [37]; the present vehicle experiments evaluate the complete EMS and do not isolate the contribution of each mechanism.

3.1. Principle of refined-SAC

The learner adopts the refinements proposed by Duan et al. [37]. The reasons for retaining each component are described below, before the corresponding gradient expression in Eq. (30).

1) Expected-value substitution replaces the sampled return in the mean-related critic update with its expectation conditional on the sampled next action. This removes the additional next-return sampling noise from that term; sampling remains in the stochastic policy and in the distribution-scale update.

2) Twin value-distribution learning uses two separately parameterized critics and selects the smaller mean value for the relevant target and actor objective. This follows distributional clipped double-Q learning [37], reducing the incentive to exploit overestimated values while potentially introducing underestimation.

3) Variance-based critic gradient adjustment scales the mean update and adapts the target-return clipping range to the predicted return dispersion [37]. The purpose is to control the gradient scale and reward-scale sensitivity. It is not a proof that gradient explosion, vanishing, or poor local solutions are eliminated in this application.

The optimized gradient expression for the soft value function is given as follows. Hyperparameters include maximum training episodes K, temperature parameter α, learning rate l_α, learning rate of the soft value network l_Z, learning rate of the policy network l_π, target network update factor τ, maximum capacity of the experience replay buffer M, and discount factor γ.

$$\nabla_{\theta_i} J_Z^{\text{scale}}(\theta_i) = (\omega_i + \zeta_\omega)\mathbb{E}[-\frac{(y_q^{\min} - Q_{\theta_i}(s,a))\nabla_{\theta_i} Q_{\theta_i}(s,a)}{\sigma_{\theta_i}(s,a)^2 + \zeta} - \frac{((C(y_z^{\min}, b_i) - Q_{\theta_i}(s,a))^2 - \sigma_{\theta_i}(s,a)^2)\nabla_{\theta_i}\sigma_{\theta_i}(s,a)}{\sigma_{\theta_i}(s,a)^3 + \zeta}]. \tag{30}$$

where $i = 1,2$ indexes the two value-distribution networks, $y_Z = \Gamma^{\pi_{\theta'}} Z(s,a)$, $y_q = \mathbb{E}[y_Z]$, $C(y_Z; b) = \text{clip}(y_Z, Q_\theta(s,a) - b, Q_\theta(s,a) + b)$, b is the clipping half-width. The update rules for b and ω are given below.

In Eq. (30), the mean target and sampled return target are both obtained from the target critic with the lower predicted mean. The two additive stabilizers are positive: one prevents division by zero at small predicted scales, and the other prevents the gradient multiplier from vanishing.

$$b_i \leftarrow \tau\xi\ \mathbb{E}[\sigma_{\theta_i}(s,a)] + (1-\tau)b_i, \tag{31}$$

$$\omega_i \leftarrow \tau\ \mathbb{E}[\sigma_{\theta_i}(s,a)^2] + (1-\tau)\omega_i, \tag{32}$$

where ξ is a constant for adjusting the range of the clipping boundary, $\xi = 3$.

The objective function and gradient function of the policy function are given by Eqs. (33) and (34), respectively. The pseudocode for the R-SAC algorithm is presented in Table 1.

The objective in Eq. (33) is maximized by gradient ascent. In Eq. (34), the first term differentiates the log policy density with respect to the actor parameters while holding the sampled action fixed. The second term accounts for the dependence of the sampled action on those parameters through the reparameterization.

$$J_\pi(\phi) = \mathbb{E}[\min_{i=1,2} Q_{\theta_i}(s,a) - \alpha\log(\pi_\phi(a|s))] \tag{33}$$

$$\begin{aligned}\nabla_{\phi}J_{\pi}(\phi) = \mathbb{E}[-\alpha\partial_{\phi}log\pi_{\phi}(a|s)\\ +(\nabla_{a}min_{i=1,2}Q_{\theta_i}(s,a) - \alpha\nabla_{a}log\pi_{\phi}(a|s))\\ \times\nabla_{\phi}f_{\phi}(\zeta;s)], \quad a = f_{\phi}(\zeta;s).\end{aligned} \tag{34}$$

Table 1. Pseudocode of the R-SAC algorithm.

Algorithm: R-SAC

Input: K, α, l_Z, l_α, l_π, τ, M, N, γ

Initialize critics, actor, and their target copies; Episode = 0.

While Episode $<$ K **do**
  For each sampling step do
    Calculate a raw action a_t based on current states s_t
    Corrective action $\tilde{a}_t$ is obtained from a_t through a real-time specific interval mapping algorithm
    Get Rewards r_t, next state s_{t+1}
    Store (s_t, a_t, r_t, s_{t+1}) in Experience buffer
  End for
  If Current experience pool data volume $\geq$ M **do**
    Randomly select N sets of experiences from an experience playback library of size M (s_t, a_t, r_t, s_{t+1})
    Update Soft Value Networks based on

$$\theta \leftarrow \theta - l_Z\nabla_{\theta_i}J_Z^{\text{scale}}(\theta_i)$$

    Update the policy network based on

$$\phi \leftarrow \phi + l_\pi\nabla_{\phi}J_{\pi}(\phi)$$

    Update the temperature parameter based on

$$\alpha \leftarrow \alpha - l_\alpha\nabla_{\alpha}L_{\alpha}(\alpha)$$

    Update the target network

$$\overline{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\overline{\theta}_i, \overline{\phi} \leftarrow \tau\phi + (1-\tau)\overline{\phi}$$

  End if
Episode = Episode + 1
End While

3.2. Optimization target with constraints

The action and state definitions are given in Eqs. (35) and (36). The actor produces three normalized continuous outputs: two operating-point coordinates and one mode-selection coordinate. The first two are mapped to engine speed and torque; the third is thresholded according to Eq. (37). Thus, the network has a continuous output representation, while the executed powertrain mode remains discrete.

$$A = \{a_i, i = 1,2,3, -1 \leq a_i \leq 1\} \tag{35}$$

where a_1 and a_2 represent the speed and torque of the engine, and a_3 denotes the mode of the powertrain system.

The state includes vehicle speed, SOC, demanded torque, gear, and normalized progress within the driving cycle. Cycle progress is a phase indicator, not a preview of future speed demand. Evaluation on an unseen cycle therefore remains necessary.

$$S = \{speed, \mathrm{SOC}, T_{\mathrm{req}}, gear, \frac{i}{n}\} \tag{36}$$

Equation (37) selects an EV or HEV mode group from the third output. Gear selection and the drivetrain topology determine the corresponding member of that group. Thresholding this output does not by itself verify the torque-speed feasibility of the selected operating point.

$$mode = \begin{cases} 1,4, & -1 \le a_3 \le 0 \\ 2,3,5, & 0 < a_3 \le 1 \end{cases} \tag{37}$$

The mapper uses speed, torque demand, gear, and component limits. For u in [-1,1] and a nonempty interval $[x_{min}, x_{max}]$, the command is $x_{min} + \frac{(u+1)(x_{max}-x_{min})}{2}$. Coupled coordinates are mapped sequentially using the preceding selections. Empty intervals or invalid inputs require a supervisory fallback. Instantaneous feasibility does not guarantee SOC invariance, recursive feasibility, or safety under arbitrary disturbances or model errors.

Within a fixed feasible branch, the continuous interval transform is affine in its normalized coordinate. The complete interface can nevertheless be nonsmooth because it also includes mode thresholds and changes of active bounds. The implementation stores raw normalized actions in replay and uses that same representation in the critic and actor loss. The mapper is part of the environment interface, outside the differentiation graph: the critic approximates the return obtained after the raw action has been mapped and executed, rather than evaluating a mismatched physical action [39].

The actor update differentiates the reparameterized policy and the learned raw-action critic, not the Simulink mapper. No mapper Jacobian is required, but critic approximation, bootstrapping, and smoothing near mode thresholds can bias the return gradient [40]. The entropy term applies to the raw-action policy, not necessarily to executed commands after thresholding or many-to-one mapping.

Figure 3 summarizes the real-time interval mapping procedure.

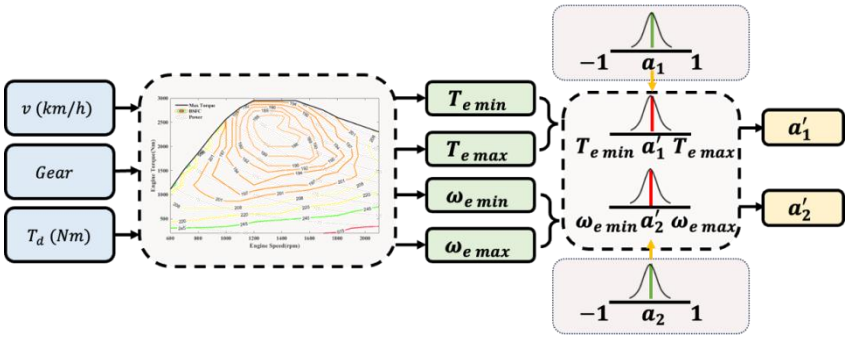


Figure 3. Architecture diagram of the real-time specific interval mapping algorithm.

3.3. R-SAC-based EMS setting

The actor uses five state inputs, two hidden layers with 128 and 64 units, and three mean-action outputs for deployment. Each distributional critic receives the five states and three normalized actions, and outputs the mean and scale of a soft-return distribution. Table 2 specifies the learning parameters. The actor, two critics, temperature coefficient, and replay buffer are used in offline learning; online deployment uses the trained actor and action mapper with fixed weights.

Table 2. Hyperparameters of the R-SAC algorithm.

Parameters	Value
Experience replay pool M	2^{14}
Experience batches N	512
Discount factor γ	0.95
Learning rate l	3×10^{-4}
Target network update factor τ	0.005
Target entropy H_0	-3
Initial temperature coefficient	0.2

Figure 4 shows how the R-SAC policy and interval mapper are integrated into the EMS.

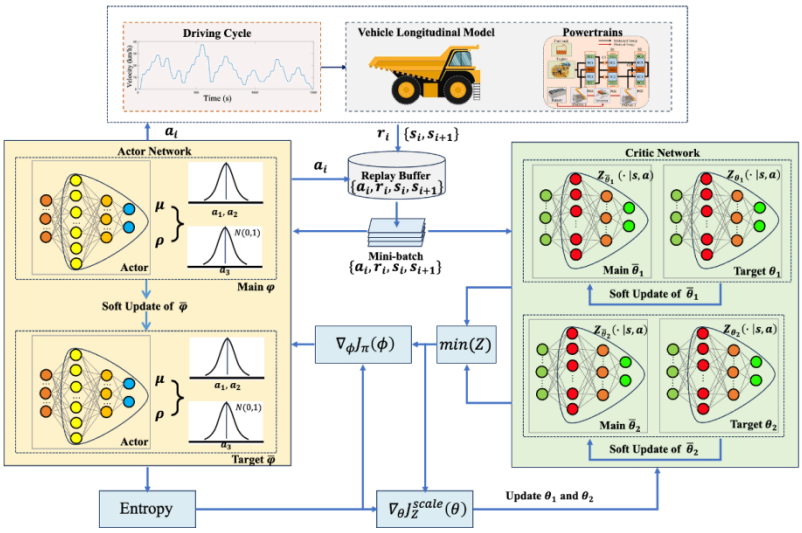


Figure 4. Schematic of the proposed R-SAC-based EMS framework.

At each policy decision instant, the normalized state is supplied to the actor. Its raw action is held until the next decision; the mode coordinate is thresholded, and the remaining coordinates are mapped to feasible commands using the current drivetrain state. Offline learning stores $[s_t, a_t, r_t, s_{t+1}]$ with the raw action and samples replay transitions once the buffer condition is met. HIL execution uses the fixed mean actor with the same normalization and mapping, without replay or parameter updates. The reported 1000-Hz plant rate does not specify the actor invocation rate.

Training is an offline development cost, separate from online latency and the fuel use measured over an evaluation cycle. Training wall-time accounting includes initialization, data collection, learner

updates, evaluation, and checkpointing; MATLAB startup and initial compilation are separate setup costs. The deployed 5-128-64-3 mean network requires 9024 dense-layer multiply-accumulate operations and stores 9219 parameters (36,876 bytes at 32-bit precision). Mapping adds component lookups, drivetrain algebra, and bound calculations, but no iterative trajectory optimization. These mean-network counts exclude the stochastic actor's scale head and do not establish measured training time or target-controller latency.

4. Results and discussion

4.1. Performance of the optimal EMS

Figure 5 shows the accumulated reward during training. The accumulated reward is a numerical score from Eq. (9), not a measure of fuel use or energy.

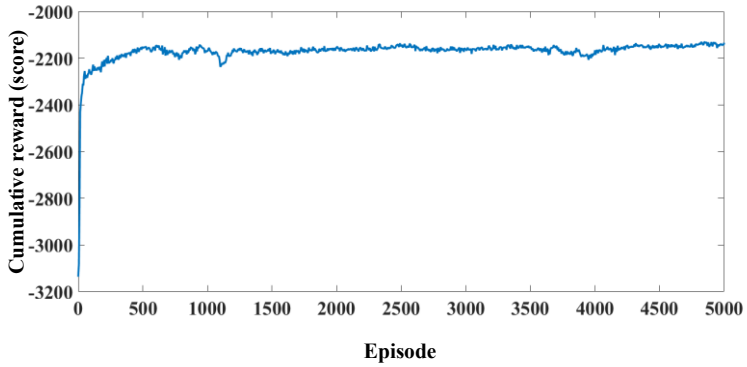
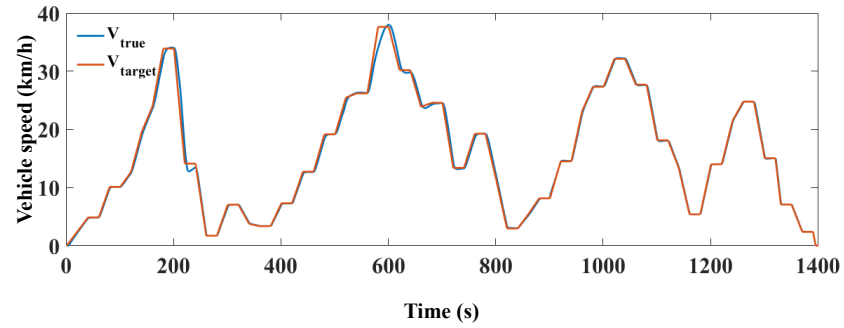
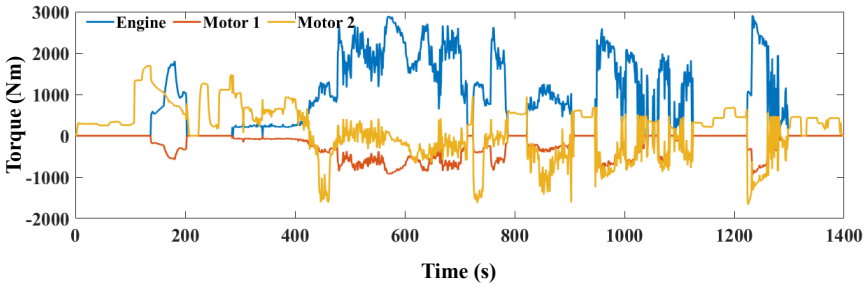


Figure 5. Accumulated training reward of R-SAC.

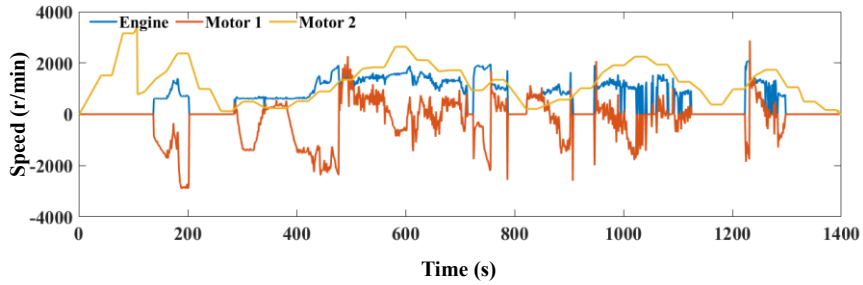
Figure 6 shows the vehicle speed and component torque and speed trajectories under Cycle 1. The M/G1 traces show its participation in power splitting.



(a)



(b)



(c)

Figure 6. Results of R-SAC under driving Cycle 1. (a) Vehicle speed. (b) Torque of power devices. (c) Speed of power devices.

Figures 7(a) and 7(b) show the cumulative fuel consumption and battery SOC, respectively, for R-SAC under Cycle 1. The SOC falls from 55% to approximately 45% at around 450 s and then generally recovers, with fluctuations, to 49.3% at the end of the cycle. Cumulative fuel consumption reaches 9.155 L.

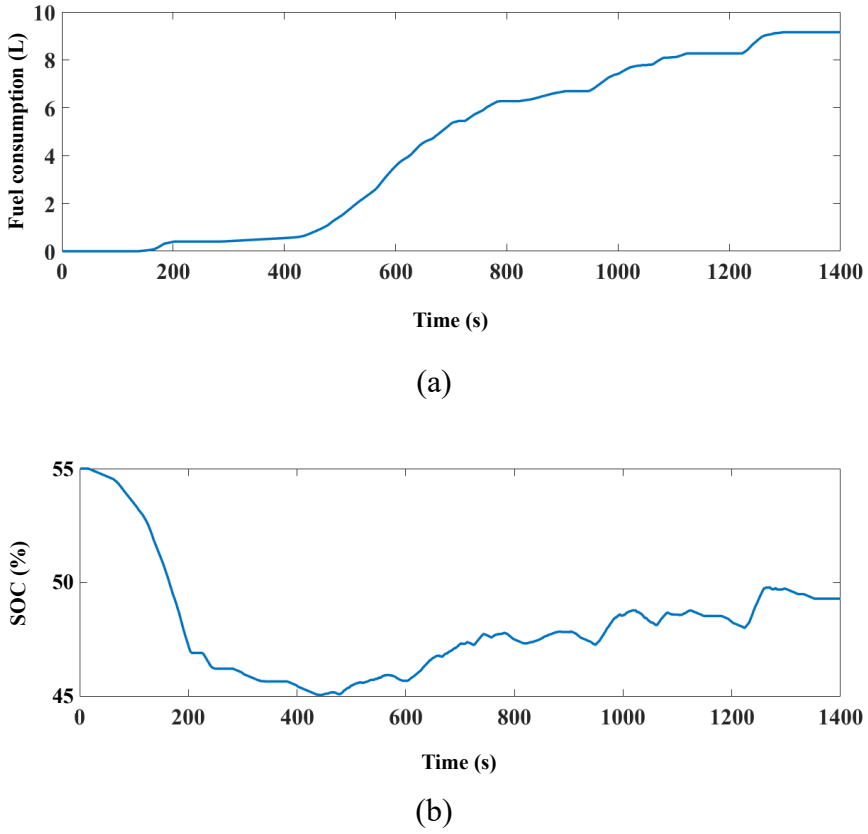
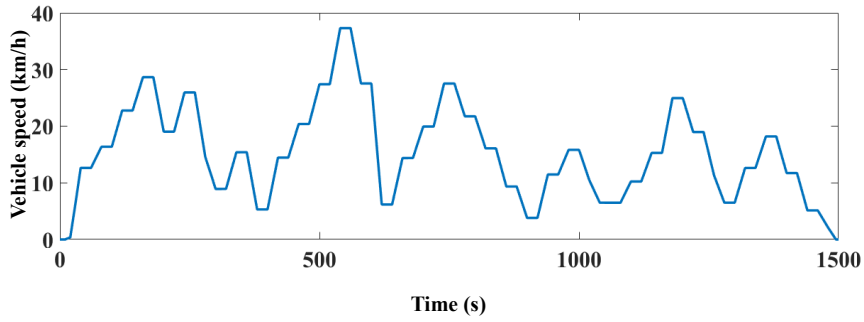


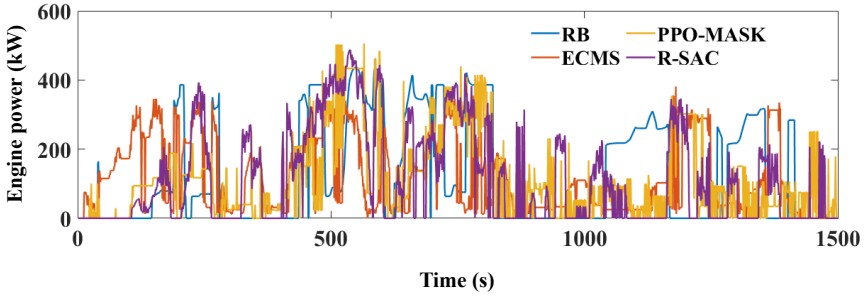
Figure 7. Fuel consumption and SOC under R-SAC for Cycle 1. (a) Cumulative fuel consumption. (b) SOC (%).

4.2. Performance under other driving cycles

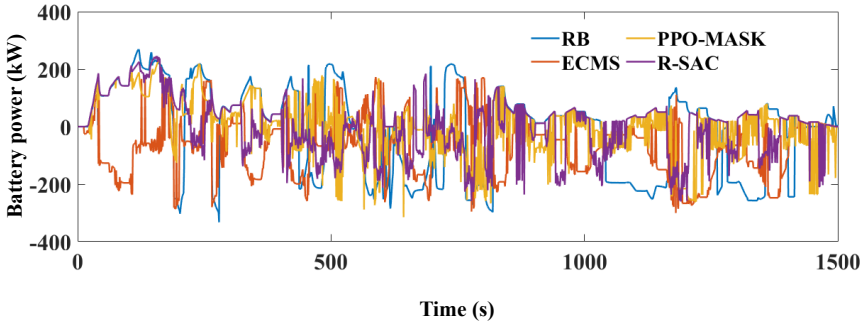
Cycle 2 is used as an unseen evaluation cycle to compare R-SAC with RB-EMS, ECMS, and PPO-MASK. Figure 8 shows the engine-battery allocation for the four strategies. Under changing demand, R-SAC selects a combination of engine operation, motor power exchange, and mode decisions. A fuel-economy comparison must also account for terminal SOC, because a lower fuel total obtained by greater battery depletion is not an equivalent energy outcome.



(a)



(b)



(c)

Figure 8. Energy allocation comparison of four algorithms under driving Cycle 2. (a) Vehicle speed. (b) Power of engine. (c) Power of battery.

The generator regulates the engine speed, while the traction motor shares wheel demand. Motor power exchange can therefore shift engine operation while meeting the same driving task. Figure 8 shows this power redistribution, and Figure 9 shows the associated mode transitions.

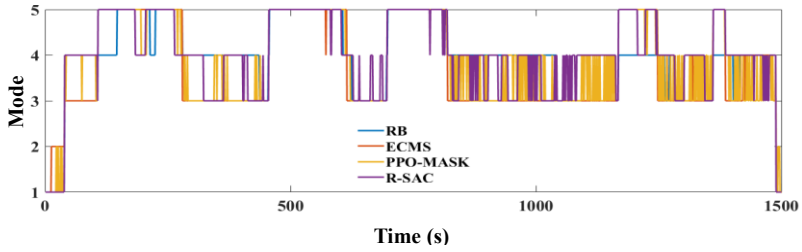


Figure 9. Mode transition process comparison among four algorithms under driving Cycle 2.

Figure 10 compares the engine operating points on the brake-specific fuel consumption (BSFC) map.

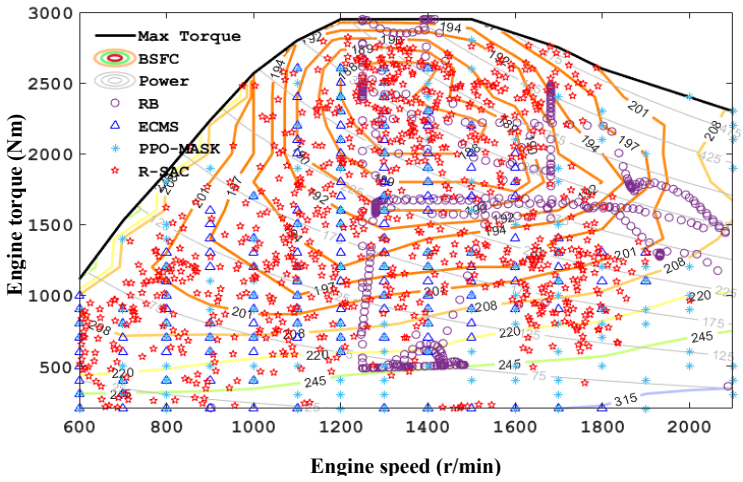


Figure 10. Engine operating points on the brake-specific fuel consumption map for the four strategies under Cycle 2.

Figure 11 groups the engine operating points by BSFC and reports engine-off observations separately.

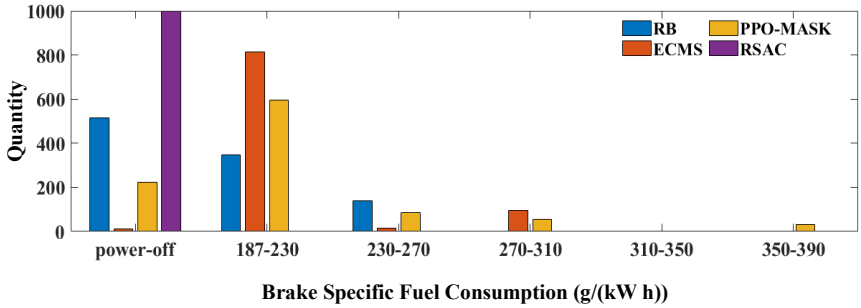


Figure 11. Distribution of engine operating points by brake-specific fuel consumption under Cycle 2.

The counts in Figure 11 describe where the engine operated on the map, rather than how much fuel it consumed over the cycle.

4.3. Performance in online EMS application

HIL testing evaluates the execution of the trained EMS on the controller while it interacts with a real-time plant model, exposing communication, numerical, and implementation effects. Figure 12 shows the HIL platform architecture.

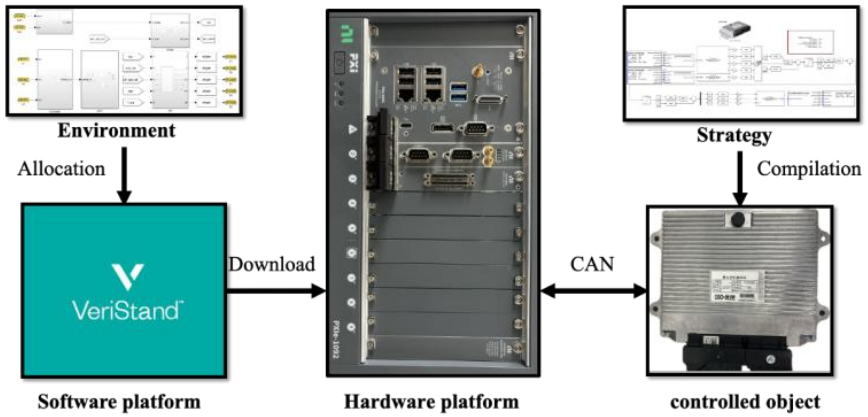


Figure 12. Overall architecture of the hardware-in-the-loop real-time simulation platform.

In hardware-in-the-loop (HIL) testing of energy management strategies, the VCU is typically selected as the controlled object. In this experiment, the EV22314B01 automotive-grade controller from Youkong was chosen as the controlled object. This controller was developed based on the NXP S32K314 main chip, with its specifications detailed in Table 3.

Table 3. Parameters of the NXP S32K314 chip.

Performance	Parameter
Microcontroller	32-bit NXP S32K314
Static random access memory (SRAM)	512 KB
Maximum frequency	160 MHz
Floating point capability	Yes
Flash memory	4 MB
Single-board microprocessor	MC33CFS6500

The trained mean-action actor is reconstructed in Simulink using the exported weights, retaining the training normalization, hidden-layer activations, and output transformation. Equation (38) gives the forward pass, where ρ1 and ρ2 denote the trained hidden-layer activations. The controller is compiled and flashed to the EV22314B01; states and commands are exchanged through the CAN DBC interface.

During HIL execution, the real-time plant supplies vehicle states to the VCU, which evaluates the fixed actor, applies mode selection and interval mapping, and returns physical commands. The setup

uses CAN at 500 kbps and a plant execution frequency of 1000 Hz; neither specifies the actor task period or its measured latency. Figure 13 shows the HIL test bench.

$$\begin{aligned} h_1 &= \rho_1(s_t W_1 + b_1), \\ h_2 &= \rho_2(h_1 W_2 + b_2), \\ a_t &= tanh(h_2 W_3 + b_3). \end{aligned} \tag{38}$$

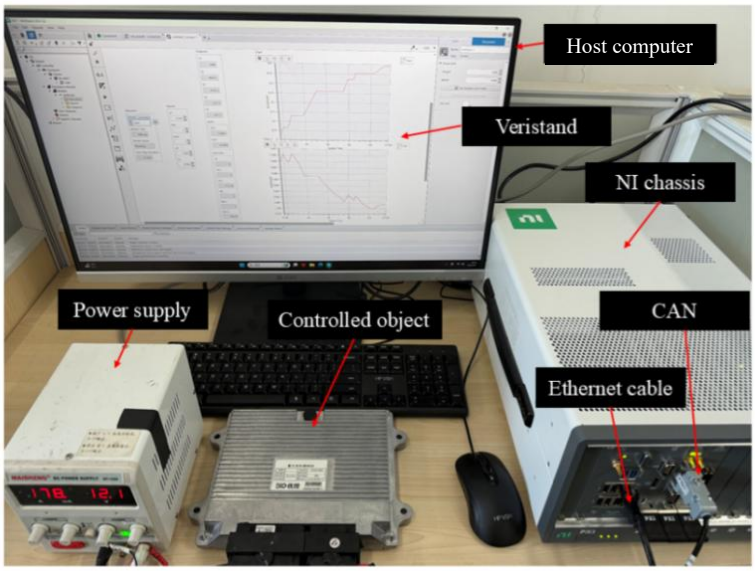


Figure 13. Hardware-in-the-loop test bench for the R-SAC energy management strategy.

HIL experiments under Cycle 2 compare the deployed policy with its offline counterpart. Figure 14 presents the resulting engine and motor operation and mode switching. The M/G1 speed and torque traces appear in Figure 14(b) and (e), respectively.

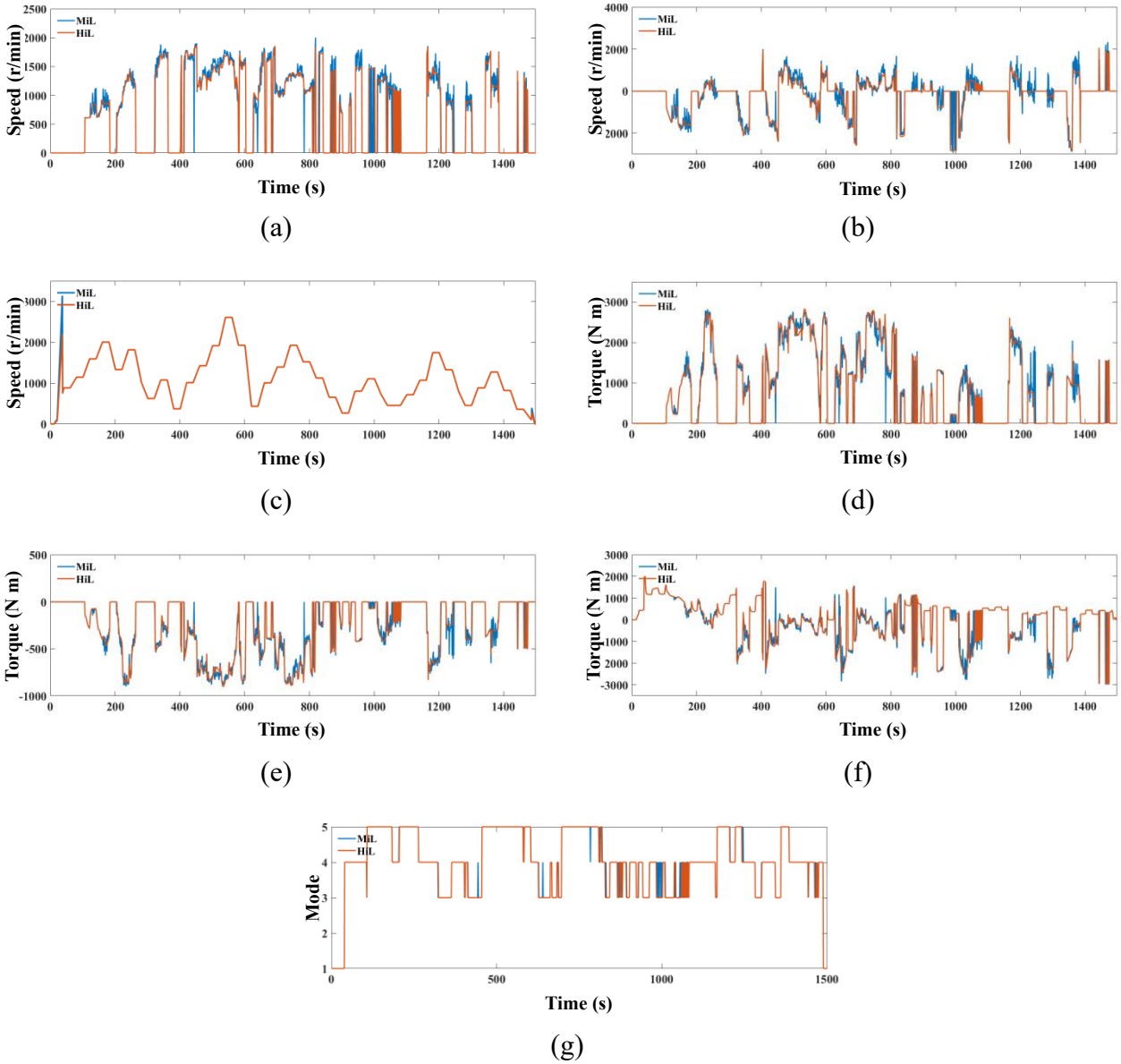


Figure 14. Comparison curves of mode switching and power operation states under offline simulation and hardware-in-the-loop testing. (a) Engine speed. (b) Motor 1 speed. (c) Motor 2 speed. (d) Engine torque. (e) Motor 1 torque. (f) Motor 2 torque. (g) Mode.

Figure 15 compares the offline and HIL SOC trajectories.

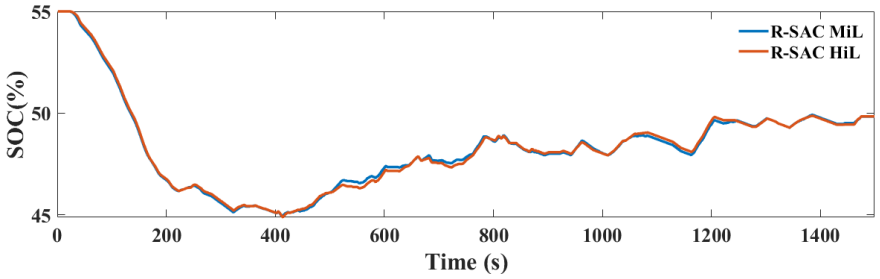


Figure 15. SOC (%) under offline simulation and HIL testing.

Figure 16 shows the signed SOC difference between the two implementations.

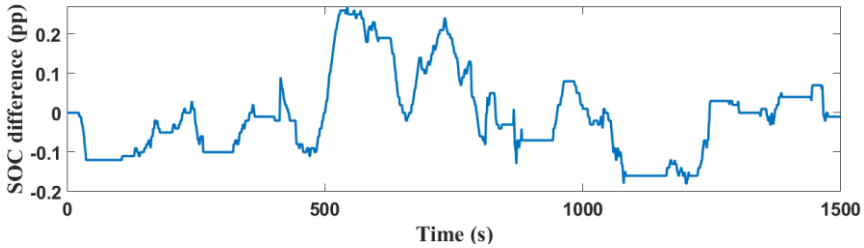


Figure 16. Signed SOC difference between offline simulation and HIL testing.

Figure 17 compares their cumulative fuel consumption.

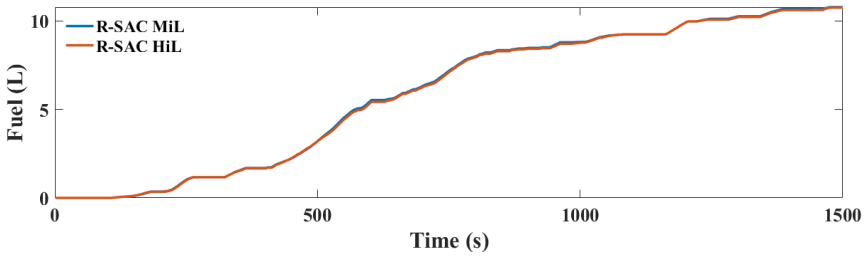


Figure 17. Fuel consumption under offline (MiL) and HIL (HiL) execution of the R-SAC policy.

Figure 18 shows the signed difference in cumulative fuel consumption.

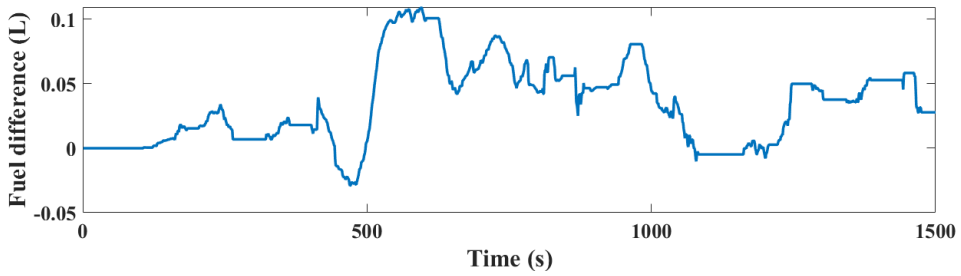


Figure 18. Signed cumulative-fuel difference between offline simulation and HIL testing (L).

Table 4 reports final absolute differences of 0.01 percentage points in SOC and 0.028 L in cumulative fuel consumption. Relative to the offline final values, these correspond to approximately 0.020% and 0.260%, respectively. This agreement supports the reconstruction and closed-loop execution of the fixed policy under Cycle 2.

Table 4. Comparison of fuel consumption and SOC under offline simulation and hardware-in-the-loop testing for the R-SAC algorithm.

	SOC (%)	Fuel consumption (L)
Offline simulation	49.83	10.766
HIL test	49.84	10.738
Absolute final difference	0.01	0.028
Relative final difference (%)	0.020	0.260

Table 5 reports actual fuel consumption and initial/final SOC under Cycle 2. Terminal SOC differs by at most 0.08 percentage points. R-SAC consumes less fuel while ending at a slightly higher SOC than each baseline, so its fuel advantage is not explained by greater battery depletion. All runs end below the initial SOC of 55.00% and are therefore not strictly charge-sustaining.

Table 5. Cycle 2 fuel consumption and initial/final SOC used for comparative calculation.

Strategy	Initial SOC (%)	Final SOC (%)	Fuel (L)	Fuel reduction relative to this baseline
RB-EMS	55.00	49.75	11.710	8.06%
ECMS	55.00	49.76	11.324	4.93%
PPO-MASK	55.00	49.80	11.231	4.14%
R-SAC	55.00	49.83	10.766	Reference

Each reduction is calculated as (baseline fuel - R-SAC fuel)/baseline fuel × 100%. Accordingly, rounding the tabulated values to one decimal place gives 8.1%, 4.9%, and 4.1%.

Table 6 supplements actual fuel consumption with a nominal energy-equivalent metric that includes net battery depletion. Because SOC compensation depends on the conversion convention [41], Eq. (39) combines fuel chemical energy and nominal battery-energy depletion with equal energy weights.

$$E_{\text{net,nom}} = \frac{Q_{\text{nom}} U_{\text{nom}}}{1000} \cdot \frac{(SOC_{\text{i}} - SOC_{\text{f}})}{100},$$
$$F_{\text{eq,nom}} = F_{\text{act}} + \frac{3.6 E_{\text{net,nom}}}{\rho_{\text{f}} H_{\text{u}}}. \tag{39}$$

In Eq. (39), Q_{nom} = 158 Ah and U_{nom} = 608 V are the nominal battery ratings in Table A1; SOC_i and SOC_f are the initial and final SOC in percentage. $E_{net,nom}$ is the estimated net battery-energy depletion in kWh, F_{act} is the actual fuel consumption in L, and $F_{eq,nom}$ is the nominal energy-equivalent fuel consumption in L. The fuel-density value is ρf = 0.85 kg/L, consistent with the model mass-to-volume conversion. The diesel lower heating value Hu = 43.0 MJ/kg is the IPCC default for gas/diesel oil [42], not a measured property of the test fuel. The factors 1000, 100, and 3.6 convert Wh to kWh, SOC percent to a fraction, and kWh to MJ, respectively.

This convention gives a battery-energy conversion of 0.0984952 L/kWh. It uses nominal voltage and the rounded results in Table 5, not a calibrated ECMS factor or an engine-recharging efficiency. The results are nominal energy-accounting estimates, not measured charge-sustaining fuel. Net SOC change already includes recovered braking energy, which is not credited twice.

Table 6. Nominal energy-equivalent fuel comparison under Cycle 2.

Strategy	Net battery energy (kWh)	Fuel equivalent of battery use (L)	Nominal equivalent fuel (L)	R-SAC reduction (%)
RB-EMS	5.043	0.497	12.207	7.80
ECMS	5.034	0.496	11.820	4.78
PPO-MASK	4.995	0.492	11.723	3.99
R-SAC	4.967	0.489	11.255	Reference

Under this convention, R-SAC reduces nominal equivalent fuel by 7.80%, 4.78%, and 3.99% relative to RB-EMS, ECMS, and PPO-MASK (Table 6). It uses both less actual fuel and less nominal battery energy than these baselines; its ranking therefore remains unchanged under any common nonnegative battery-energy weight, although the numerical savings depend on that weight.

Taken together, Figures 8–11 show how engine-speed regulation, motor power exchange, and mode selection redistribute demanded power between the engine and battery. The learner selects the allocation, while the mapper enforces the modeled instantaneous operating bounds. The lower fuel consumption at slightly higher terminal SOC (Table 5), together with the nominal equivalent-fuel comparison (Table 6), supports an energy-management benefit for the complete EMS on Cycle 2.

The comparisons evaluate the integrated EMS, not the independent contribution of each refinement. Matched comparisons with standard SAC, DSAC, or LLM-assisted SAC and controlled component ablations were not performed. The single training history, two driving cycles, and HIL agreement do not establish across-seed convergence or robustness to untested conditions.

5. Conclusions

This study integrates a distributional SAC learner with a powertrain-dependent action interface for a multi-mode hybrid mining truck. The critic refinements are adopted from [37], while the

application formulation, real-time interval mapping, and embedded implementation form the focus of the present EMS study.

For the reported unseen Cycle 2, the strategy uses 10.766 L of fuel and ends at SOC 49.83%. Relative to the tabulated RB-EMS, ECMS, and PPO-MASK totals, fuel consumption decreases by 8.1%, 4.9%, and 4.1%, respectively. The comparable terminal SOC values and the engine operating-point distributions support the interpretation that coordinated allocation improves the complete-cycle result.

HIL implementation yields final SOC and fuel differences of 0.01 percentage points and 0.028 L from offline simulation, supporting the consistency of the deployed network and command interface for the tested cycle.

Further evaluation should include independent vehicle validation, target-controller latency measurements, multiple-seed training, component ablations, and additional duty cycles with payload and grade variations.

Acknowledgments

This work was supported by the Natural Science Foundation of China (Grant No. 52275082), the State Key Laboratory of Mechanical Transmission for Advanced Equipment, Chongqing University (SKLMT-MSKFKT-202332), and China Scholarship Council (No. 202406460052). Any opinions expressed in this paper are solely those of the authors and do not represent those of the sponsors.

Author contributions

All authors contributed to the study conception, methodology, analysis, and manuscript preparation. All authors reviewed and approved the final manuscript.

Conflict of interest

The authors declare that there is no conflict of interest.

Use of generative-AI tools declaration

The authors used OpenAI generative AI tools solely for language editing and proofreading, including the final publication proof. The authors take responsibility for the content of the manuscript.

References

1. Y. Li, H. He, A. Khajepour, Y. Chen, W. Huo, H. Wang, Deep reinforcement learning for intelligent energy management systems of hybrid-electric powertrains: Recent advances, Open Issues, and Prospects, *IEEE Trans. Transp. Electrific.,* **10** (2024), 9877–9903. https://doi.org/10.1109/TTE.2024.3377809
2. X. Tang, J. Chen, Y. Qin, T. Liu, K. Yang, A. Khajepour, et al., Reinforcement learning-based energy management for hybrid power systems: State-of-the-art survey, review, and perspectives, *Chin. J. Mech. Eng.,* **37** (2024), 43. https://doi.org/10.1186/s10033-024-01026-4

3. A. S. Mohammed, S. M. Atnaw, A. O. Salau, J. N. Eneh, Review of optimal sizing and power management strategies for fuel cell/battery/super capacitor hybrid electric vehicles, *Energy Rep.,* **9** (2023), 2213–2228. https://doi.org/10.1016/j.egyr.2023.01.042
4. D. Shi, S. Wang, Y. Cai, L. Chen, Stochastic predictive energy management of power split hybrid electric bus for real-world driving cycles, *IEEE Access,* **6** (2018), 61700–61713. https://doi.org/10.1109/ACCESS.2018.2876147
5. B. HomChaudhuri, R. Lin, P. Pisu, Hierarchical control strategies for energy management of connected hybrid electric vehicles in Urban roads, *Transp. Res. Part C Emerg. Technol.,* **62** (2016), 70–86. https://doi.org/10.1016/j.trc.2015.11.013
6. M. A. Hannan, F. Azidin, A. Mohamed, Hybrid electric vehicles and their challenges: A review, *Renew. Sustain. Energy Rev.,* **29** (2014), 135–150. https://doi.org/10.1016/j.rser.2013.08.097
7. J. Peng, H. He, R. Xiong, Rule based energy management strategy for a series–parallel plug-in hybrid electric bus optimized by dynamic programming, *Appl. Energy,* **185** (2017), 1633–1643. https://doi.org/10.1016/j.apenergy.2015.12.031
8. S. Zhang, X. Hu, S. Xie, Z. Song, L. Hu, C. Hou, Adaptively coordinated optimization of battery aging and energy management in plug-in hybrid electric buses, *Appl. Energy,* **256** (2019), 113891. https://doi.org/10.1016/j.apenergy.2019.113891
9. H. Li, A. Ravey, A. N'Diaye, A. Djerdir, Online adaptive equivalent consumption minimization strategy for fuel cell hybrid electric vehicle considering power sources degradation, *Energy Convers. Manage.,* **192** (2019), 133–149. https://doi.org/10.1016/j.enconman.2019.03.090
10. F. Zhang, X. Hu, R. Langari, D. Cao, Energy management strategies of connected HEVs and PHEVs: Recent progress and outlook, *Prog. Energy Combust. Sci.,* **73** (2019), 235–256. https://doi.org/10.1016/j.pecs.2019.04.002
11. C. M. Martinez, X. Hu, D. Cao, E. Velenis, B. Gao, M. Wellers, Energy management in plug-in hybrid electric vehicles: Recent progress and a connected vehicles perspective, *IEEE Trans. Veh. Technol.,* **66** (2017), 4534–4549. https://doi.org/10.1109/TVT.2016.2582721
12. R. C. Hsu, C. Liu, D. Chan, A reinforcement-learning-based assisted power management with QoR provisioning for human–electric hybrid bicycle, *IEEE Trans. Ind. Electron.,* **59** (2011), 3350–3359. https://doi.org/10.1109/TIE.2011.2141092
13. V. Mnih, K. Kavukcuoglu, D. Silver, Human-level control through deep reinforcement learning, *Nature,* **518** (2015), 529–533. https://doi.org/10.1038/nature14236
14. J. Hwangbo, J. Lee, A. Dosovitskiy, Learning agile and dynamic motor skills for legged robots, *Sci. Robot.,* **4** (2019), 5872. https://doi.org/10.1126/scirobotics.aau5872
15. Y. Li, H. He, Deep reinforcement learning-based energy management for hybrid electric vehicles, morgan & claypool publishers, 2022. https://doi.org/10.1007/978-3-031-79206-9
16. A. Perera, P. Kamalaruban, Applications of reinforcement learning in energy systems, *Renew. Sustain. Energy Rev.,* **137** (2021), 110618. https://doi.org/10.1016/j.rser.2020.110618
17. B. Xu, F. Malmir, Z. Filipi, Real-time reinforcement learning optimized energy management for a 48V mild hybrid electric vehicle, 2019. https://doi.org/10.4271/2019-01-1208
18. T. Liu, Y. Zou, D. Liu, F. Sun, Reinforcement learning of adaptive energy management with transition probability for a hybrid electric tracked vehicle, *IEEE Trans. Ind. Electron.,* **62** (2015), 7837–7846. https://doi.org/10.1109/TIE.2015.2475419

19. Y. Zou, T. Liu, D. Liu, F. Sun, Reinforcement learning-based real-time energy management for a hybrid tracked vehicle, *Appl. Energy,* **171** (2016), 372–382. https://doi.org/10.1016/j.apenergy.2016.03.082
20. 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, *Appl. Energy,* **222** (2018), 799–811. https://doi.org/10.1016/j.apenergy.2018.03.104
21. Y. Li, H. He, A. Khajepour, H. Wang, J. Peng, Energy management for a power-split hybrid electric bus via deep reinforcement learning with terrain information, *Appl. Energy,* **255** (2019), 113762. https://doi.org/10.1016/j.apenergy.2019.113762
22. C. Qi, C. Song, F. Xiao, S. Song, Generalization ability of hybrid electric vehicle energy management strategy based on reinforcement learning method, *Energy,* **250** (2022), 123826. https://doi.org/10.1016/j.energy.2022.123826
23. C. Qi, Y. Zhu, C. Song, Hierarchical reinforcement learning based energy management strategy for hybrid electric vehicle, *Energy,* **238** (2022), 121703. https://doi.org/10.1016/j.energy.2021.121703
24. A. Biswas, P. G. Anselma, A. Emadi, Real-time optimal energy management of multimode hybrid Eelectric powertrain with online trainable asynchronous advantage actor–critic algorithm, *IEEE Trans. Transp. Electrific.,* **8** (2022), 2676–2694. https://doi.org/10.1109/TTE.2021.3138330
25. J. Xu, Z. Li, G. Du, Q. Liu, L. Gao, Y. Zhao, A transferable energy management strategy for hybrid electric vehicles via dueling deep deterministic policy gradient, *Green Energy Intell. Transp.,* **1** (2022), 100018. https://doi.org/10.1016/j.geits.2022.100018
26. J. Hong, T. Zhang, Z. Zhang, H. Zhang, Investigation of energy management strategy for a novel electric-hydraulic hybrid vehicle: Self-adaptive electric-hydraulic ratio, *Energy,* **278** (2023), 127582. https://doi.org/10.1016/j.energy.2023.127582
27. K. Wang, R. Yang, Y. Zhou, W. Huang, S. Zhang, Design and improvement of SD3-based energy management strategy for a hybrid electric urban bus, *Energies,* **15** (2022), 5878. https://doi.org/10.3390/en15165878
28. X. Tang, J. Zhang, D. Pi, X. Lin, L. M. Grzesiak, X. Hu, Battery health-aware and deep reinforcement learning-based energy management for naturalistic data-driven driving scenarios, *IEEE Trans. Transp. Electrific.,* **8** (2022), 948–964. https://doi.org/10.1109/TTE.2021.3107143
29. Z. Pan, D. Lei, L. Wang, A knowledge-based two-population optimization algorithm for distributed energy-efficient parallel machines scheduling, *IEEE Trans. Cybern.,* **52** (2022), 5051–5063. https://doi.org/10.1109/TCYB.2020.3026571
30. H. Wang, B. R. Sarker, J. Li, J. Li, Adaptive scheduling for assembly job shop with uncertain assembly times based on dual Q-learning, *Int. J. Prod. Res.,* **59** (2021), 5867–5883. https://doi.org/10.1080/00207543.2020.1794075
31. F. Zhao, J. Zhou, L. Wang, H. Sang, A hierarchical optimization algorithm with dual-cache synced tuning mechanism for distributed flexible job shop scheduling problem, *IEEE Trans. Cybern.,* **56** (2026), 5477–5490. https://doi.org/10.1109/TCYB.2025.3644904
32. Q. Zhou, J. Li, B. Shuai, Multi-step reinforcement learning for model-free predictive energy management of an electrified off-highway vehicle, *Appl. Energy,* **255** (2019), 113755. https://doi.org/10.1016/j.apenergy.2019.113755

33. X. Ma, H. Liu, L. Han, N. Yang, Q. Fan, C. Xiang, Intelligent energy management for hybrid electric vehicles based on data-driven real-time dynamic programming, *Energy,* **354** (2026), 140973. https://doi.org/10.1016/j.energy.2026.140973
34. F. Li, M. Li, Y. Wu, H. Li, Y. Song, Z. Huang, Safe reinforcement learning energy management for hybrid electric vehicles: A supervisory action assessment and correction approach, *J. Clean. Prod.,* **544** (2026), 147720. https://doi.org/10.1016/j.jclepro.2026.147720
35. Z. E. Liu, Q. Zhou, Y. Li, S. Shuai, H. Xu, Safe deep reinforcement learning-based constrained optimal control scheme for HEV energy management, *IEEE Trans. Transp. Electrific.,* **9** (2023), 4278–4293. https://doi.org/10.1109/TTE.2023.3240430
36. C. Li, B. Zhang, G. Chen, J. Huang, Z. Dong, Large language model-assisted autonomous bidding for battery energy storage systems: An agentic approach for smart power industrial decision-making, *IEEE Trans. Autom. Sci. Eng.,* 2026. https://doi.org/10.1109/TASE.2026.3740512
37. J. Duan, W. Wang, L. Xiao, J. Gao, S. E. Li, C. Liu, et al., Distributional soft actor-critic with three refinements, *IEEE Trans. Pattern Anal. Mach. Intell.,* **47** (2025), 3935–3946. https://doi.org/10.1109/TPAMI.2025.3537087
38. T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, *Proc. Mach. Learn. Res.,* **80** (2018), 1861–1870. https://proceedings.mlr.press/v80/haarnoja18b.html
39. H. Markgraf, S. Sawant, H. Krasowski, L. Schäfer, S. Gros, M. Althoff, Safe reinforcement learning using action projection: Safeguard the policy or the environment? *Trans. Mach. Learn. Res.,* 2026. https://openreview.net/forum?id=DDrGSEYxGU
40. J. Wen, S. Kumar, R. Gummadi, D. Schuurmans, Characterizing the gap between actor-critic and policy gradient, *Proc. Mach. Learn. Res.,* **139** (2021), 11101–11111. https://proceedings.mlr.press/v139/wen21b.html
41. R. Sauermann, F. Kirschbaum, O. Nelles, Optimization of fuel consumption for rule-based energy management strategies of hybrid electric vehicles: SOC compensation methods, *IEEE Access,* **12** (2024), 112594–112604. https://doi.org/10.1109/ACCESS.2024.3443190
42. A. Garg, K. Kazunari, T. Pulles, *Energy,* 2006 IPCC Guidelines for National Greenhouse Gas Inventories, 2006. Available from: https://www.ipcc-nggip.iges.or.jp/public/2006gl/pdf/2_Volume2/V2_1_Ch1_Introduction.pdf.



AIMS Press

© 2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0)