Progress and summary of reinforcement learning on energy management of MPS-EV

Yang Lin, Liang Chu, Jincheng Hu, Zhuoran Hou, Jihao Li, Jingjing Jiang, Yuanjian Zhang

ARTICLE INFO

Keywords:
Reinforcement learning
Multi-power source electric vehicles
Energy management strategy

ABSTRACT

The escalating environmental concerns and energy crisis caused by internal combustion engines (ICE) have become unacceptable under environmental regulations and the energy crisis. As a promising alternative solution, multi-power source electric vehicles (MPS-EVs) integrate various clean energy systems to enhance the powertrain efficiency. The energy management strategy (EMS) plays a pivotal role for MPS-EVs to maximize efficiency, fuel economy, and range. Reinforcement Learning (RL) has emerged as an effective methodology for EMS development, attracting continuous attention and research. However, a systematic analysis of the design elements of RL-based EMS is currently lacking. This paper addresses this gap by presenting a comprehensive analysis of current research on RL-based EMS (RL-EMS) and summarizing its design elements. This paper first summarizes the previous applications of RL in EMS from five aspects: algorithm, perception scheme, decision scheme, reward function, and innovative training method. It highlights the contributions of advanced algorithms to training effectiveness, provides a detailed analysis of perception and control schemes, classifies different reward function settings, and elucidates the roles of innovative training methods. Finally, by comparing the development routes of RL and RL-EMS, this paper identifies the gap between advanced RL solutions and existing RL-EMS. Potential development directions are suggested for implementing advanced artificial intelligence (AI) solutions in EMS.

1. Overview

The restriction of environmental regulations and the rise of clean energy technologies have accelerated electric vehicles replacing conventional vehicles [1]. Compared with ICE vehicles, battery-powered electric vehicles (BEVs) have been proven to have significant advantages in emission, economy, and efficiency. However, limited battery capacity causes mileage anxiety [2]. Multi-power source electric vehicles (MPS-EVs) provide two or more power sources that can work together. The engine and motor are used for driving. The fuel tank, hydrogen tank, and other devices are used for storage. By regulating the output power from multiple sources, their advantages can be better exploited. It is necessary for MPS-EVs to adjust the working mode according to the changeable working conditions. This requires the vehicle to have a comprehensive EMS to determine the power distribution. An appropriate and effective EMS
can significantly improve energy efficiency, and achieve lower emissions and higher economy [3].

The early EMS can be broadly categorized into rule-based and optimization-based methods [4]. Currently, rule-based methods dominate real-world implementations. The reason why they can be applied to practice is their robustness. Created by engineers with a wealth of expert experience, rule-based can adapt to various driving cycles without the risk of battery wearing out. They are easy to develop and safe to apply [5]. Simplified operational rules facilitate the controller’s ease of operation. Representative algorithms include fuzzy logic (FL) mode [6], charge consumption charge maintenance (CD-CS) mode [7], and others. While these rule-based methods find practical applicability, their potential for enhancing fuel economy in real-world scenarios is typically lower than that achieved by optimization-based methods. By setting the cost function in advance, the researchers trained the EMS to make control signals matching with the working condition to optimize some indexes and reduce the corresponding cost, such as improving fuel economy, prolonging the service life of components, and reducing emissions. This enables the optimization-based methods to perform better on specific driving cycles. Optimization-based algorithms can be categorized into global optimization and real-time optimization [8]. Global optimization requires obtaining the entire driving cycle in advance and calculating the global optimal solution based on predefined speed profiles. One of the representatives of global optimization is dynamic programming (DP). As a mathematical optimization method and computer programming method, DP divides the complex calculation process into solving subproblems [9] and gradually obtains the global optimal solution in a recursive way. Some researchers used DP to develop EMS and significantly improved the control effect [10,11]. However, it is impossible to obtain accurate speed information in advance and bear the high computational expense for driving cycles lasting thousands of seconds. DP cannot be applied in real time. In many cases, it serves as a benchmark for assessing the performance of other methods. Some researchers also incorporated the principles of DP to inspire the development of rule-based or real-time optimization techniques [4,12].

EMS based on a real-time optimization algorithm emphasizes real-time performance and outputs a control signal to minimize the cost with the cost at the current time as the objective function [13]. This method reduces the computational cost of the control strategy and improves the real-time performance, making it possible to apply the optimization-based method to the real road. Typical representatives are equivalent consumption minimization strategy [14] (ECMS) and model predictive control [15] (MPC). ECMS constructs the final cost by superimposing the consumption of different types of energy at the current moment by setting an equivalence factor. It takes minimizing the equal fuel consumption cost at the current moment as the objective of the control strategy [16,17]. However, the equivalence factor needs to be specified artificially, an inappropriate value may reduce the fuel economy and bring the risk of state of charge (SOC) for the battery exceeding the constraints, which results in the reliance on expertise. Similar to ECMS, the MPC controller calculates a future control sequence in which all control signals meet the objective of minimizing energy costs. It takes the first action of the calculated control sequence as the executed control signal. This process is repeated at the next time step by moving the prediction range forward one step [15]. Some researchers used MPC to develop EMS based on accurate models [18–20]. Although the real-time optimization algorithm solves the problem of real-time application in the driving process, there is still a large gap between the optimization performance of the whole driving cycle and global optimization because of its local optimization characteristics.

Reinforcement learning as an artificial intelligence method with excellent real-time performance and global optimization-like capabilities is expected to solve the MPS-EV energy management optimization problem [21]. In recent years, reinforcement learning has achieved beyond human-level results on a range of domains such as Go [22] and Google Research Football [23], which demonstrated its potential for application to complex control problems. As a highly concerned artificial intelligence method, RL brings a new solution to the development of MPS-EV EMS. RL models the control problem. During the training process of the algorithm, the agent interacts with the environment and improves the control strategy based on the results of the interaction, through which the RL algorithm can achieve self-learning and gradually approach the global optimal solution [24]. In contrast to rule-based methods, RL, based in optimization theory, can attain superior control outcomes. Compared to global optimization algorithms like DP, RL can be employed in real-time with reduced computational costs, enhancing its potential for practical applications; compared with real-time optimization algorithms such as ECMS and MPC, reinforcement learning has self-learning characteristics [25], requires less prior knowledge and model construction, and the control effects is closer to global optimization [21]. These features have attracted wide attention from researchers in academia and industry, and more and more studies are applying reinforcement learning to MPS-EV energy management problems.

This paper provides a summary of how the RL method has been applied to MPS-EV energy management. Unlike other reviews, this paper delves into the specific technical routes utilized by academia in recent years, detailing their unique characteristics and advantages in relation to the development of RL-EMS. Additionally, a thorough explanation of all RL-EMS research settings is included for comprehensive analysis:

1) Algorithmic applications: Reinforcement learning algorithms are a family of artificial intelligence algorithms that originated from behaviorism and have been further extended by the rise of deep learning in the last decade. Through an extensive literature study and survey, this paper summarizes the types of RL algorithms applied in MPS-EV energy management problems in general and summarizes the applications of the algorithms based on chronological order.

2) State Sensing: In the RL-EMS, the state describes the sensing capability of the EMS. It specifies the ability and scope of the EMS to acquire information and is the sole basis for algorithmic decisions. However, the definition of state variables varies in different EMS studies. Therefore, this paper summarizes the definition and design of the state of reinforcement learning solution variables and classifies them into two categories: in-vehicle state variables and out-of-vehicle state variables for a comprehensive introduction.
3) Decision schemes. The introduction of reinforcement learning has overturned the traditional EMS research approach and enriched the design of control schemes for vehicle power signals. In this paper, we list three control frameworks for final energy distribution and classify them into end-to-end control, hierarchical control, and parallel control for in-depth discussion.

4) Reward setting. As a self-learning method, the reward function of reinforcement learning shows the iterative goal of the algorithm. This paper summarizes a series of metrics commonly found in RL-EMS such as equivalent fuel consumption and component maintenance and their expressions in the function.

5) Training methods: The choice of training data and the tuning of hyperparameters significantly influence the ultimate control effectiveness of reinforcement learning solutions. In this paper, we analyze RL-EMS studies in terms of training data diversity, parameter initialization, and other improvements in training methods.

The rest of this paper is organized as follows: Section 2 introduces the popular types of multi-energy source vehicles, the general solution process and optimization objectives of RL-EMS research. Section 3 comprehensively surveys recent applications of reinforcement learning algorithms in energy management strategies across five perspectives. It presents advancements in each study and enhancements made to traditional RL-EMS. Section 4 provides an overview of RL-EMS and an outlook on the current challenges and future research directions of RL-EMS.

2. Reinforcement learning and energy management of MPS-EV

2.1. Energy management of MPS-EV

2.1.1. Representations of MPS-EV configuration

With the maturity of vehicle electrification, electrified powertrains have allowed for high-power electrical transmission, making it possible for electric motors to do work instead of engines. While the power performance is improved, the development of battery technology provided more adequate energy reserves with far less pollution than fossil fuels [26]. Relevant technologies have helped to develop a series of multi-power source electric vehicle configurations. Different types of MPS-EVs are equipped with various energy storage devices to make use of clean energy by utilizing electricity or hydrogen to keep driving. Hybrid electric vehicles (HEVs) and plug-in hybrid electric vehicles (PHEVs) take batteries and fuel tanks as energy storage devices, while fuel cell vehicles (FCEVs) are equipped with fuel cells, batteries, or supercapacitors.

2.1.1.1. Hybrid electric vehicle & plug-in hybrid electric vehicle. The vehicle moves forward using the power from both the motor and engine. The mechanical power required to drive HEVs and PHEVs comes directly or indirectly from the engine to some extent. Including the traditional internal combustion engine in an electrified powertrain leads to the release of fossil fuel emissions and a complicated structure [27]. Unlike traditional vehicles with combustion engines, hybrid powertrains use a combination of electrical and mechanical energy to adjust the engine’s working points. The hybrid drive system can utilize the motor instead of the engine in low engine efficiency situations like idling and low load. This conserves the battery’s electric power. Alternatively, it can boost the engine output power to operate in the high-efficiency zone. The part beyond the demanded power is stored in the battery [28] to provide power for possible discharges in the future. As shown in Fig. 1, the commonly used configurations in HEV & PHEV can be divided into series hybrid, parallel hybrid, and series-parallel hybrid according to the connection mode of different drive components [29].

PHEVs and HEVs have different charging methods. PHEVs have a larger battery capacity and can connect to external power sources, while HEVs rely on consuming fuel to maintain battery power. PHEVs consume battery power in pure electric mode and

![Fig. 1. Powertrain architectures of HEV&PHEV, (a) series hybrid, (b) parallel hybrid, (c) series-parallel hybrid.](image_url)
activate the engine when the battery is low. This range that only relies on electric driving is called All Electric Range (AER) [145]. The EMS for HEVs must maintain the battery level within a specific range. The EMS for PHEVs should allocate energy consumption between power sources when driving distance exceeds AER for economic and environmental friendliness.

Due to the different topology of the powertrain system, different configurations usually correspond to different degrees of mixing, which refers to the proportion of motor output power in the overall system output power. The series hybrid is often seen as more similar to the pure electric vehicle than other configurations. There exists no mechanical linkage between the engine and the drive axle as shown in Fig. 1(a), the power output is converted into electrical energy, which is turned into mechanical energy to drive the vehicle. Because the engine and flywheel speeds can be decoupled, the engine can operate at high efficiency zones. However, this configuration requires higher motor output power, leading to size and weight issues. Energy can flow through two paths in a parallel hybrid configuration as shown in Fig. 1(b), providing maximum torque output when the engine and motor work together for optimal dynamic performance. Parallel hybrid configuration has a more flexible layout for their engine and motor, the engine’s power needs to pass through the drive axle to charge the energy storage system, and this charging method cannot function when the vehicle is stationary. Parallel hybrid configuration may not be the best option for urban driving. The series-parallel hybrid configuration usually uses the planetary gear to transfer power from the engine to the axles. The power split system is currently the most commonly used in series-parallel hybrid configuration, which allows for decoupling between engine speed and vehicle speed. There are two ways in which power can be transferred from the engine to the wheels as shown in Fig. 1(c). Firstly, it can flow mechanically through the rings of the planetary system and reach the axles. Secondly, the engine’s power can be used to generate electricity, which then powers the motor and moves the wheels. Therefore, the powertrain allows for a combination of parallel and series operation, solving size and charging issues. The series-parallel hybrid configuration has been preferred by numerous vehicle manufacturers in past few years [146]. The advantages and disadvantages of various configurations are summarized in Table 1.

### Table 1
Comparison between different configurations of HEV& PHEV.

| Configuration          | Advantage                     | Disadvantage         | Reference       |
|------------------------|-------------------------------|----------------------|-----------------|
| Series hybrid          | High engine efficiency        | Size issues          | [27,145]        |
| Parallel hybrid        | Dynamic performance           | Inconvenient charging| [29,146]        |
| Series-parallel hybrid | Flexible mode                 | Complex structure    | [28,29]         |

2.1.1.2. Fuel Cell Electric Vehicle. FCEVs rely on electricity the onboard fuel cell generates to keep moving. The fuel cell (FC) is a chemical device that converts the chemical energy possessed by the fuel directly into electrical energy, and the commonly utilized fuel is high-purity hydrogen [148,149]. The fuel cell has a high internal energy conversion efficiency as it directly converts chemical energy into electrical energy without combustion. Therefore, it generates low noise and heat without any mechanical vibration. Other types of energy storage devices have lower energy density compared to fuel cells, which have a 45%–60% fuel to electricity conversion efficiency that exceeds thermal power generation efficiency. Fig. 2 (a) and Fig. 2 (b) show two main configurations of the FCEV powertrain: the fuel cell/battery (FC/B) configuration and the fuel cell/battery/supercapacitor (FC/SC/B) configuration. The current fuel cell widely used in FCEV is the polymer electrolyte membrane fuel cell (PEMFC), which has a high conversion efficiency in the electrochemical process [30,150]. Hydrogen reacts with oxygen in the fuel cell to produce electrical energy to drive the motor, which is mechanically connected to the drive wheel. The powertrain requires an unstable power output because of the complex and variable driving cycles. While the fuel cell is suitable for operation in a steady state to prolong its service life [31], the battery is employed to perform the corresponding charge and discharge operations to adjust the fuel cell output power. FCEVs that take hydrogen as fuel are faster to refuel and have better economy and efficiency [32]. In addition to batteries, some FCEVs incorporate supercapacitors (SCs) as energy storage devices. Supercapacitor (SC) is a type of energy storage device that has a high energy storage capacity and is capable of providing powerful pulse power because of its high power density and fast transient response [33]. Integrating the rapid transient response of SC with the gradual transient response of fuel cells presents a viable alternative for enhancing the efficiency and performance of Fuel Cell Electric Vehicle.

2.1.2. Modeling of energy management problem of MPS-EV

In the real-world driving process, the driver adjusts the pedals to control the speed of the vehicle. In the simulation, the acceleration and deceleration process of the vehicle model is simulated based on the speed profile provided by the predefined training driving cycles, and the power demand of the multi-energy source vehicle is then determined. The MPS-EV energy management problem is dedicated to the rational distribution of energy flow under different working conditions to improve fuel economy. The consumption of electricity and fuel is the most intuitive reflection of the economy and is also a standard index to evaluate the control effect of the EMS. Some research takes into account the effect of energy distribution on the health condition of components, such as the adverse effects of frequent start-stop on engine and battery life [34,35]; fuel consumption and battery SOC are directly affected by energy distribution, and a good EMS can finish the testing driving cycle with less power and fuel consumption by balancing the distribution of multi-power sources.

Taking the backward model [36–38] as an example, Fig. 3 illustrates the general process of the energy management problem: 1) working condition requirements: the vehicle’s driving cycle is derived from training data. The speed profile of the driving cycle and vehicle parameters such as vehicle mass and windward area determine the vehicle output power during the simulation; 2) control actions: the EMS allocates energy flow, and the vehicle’s operating mode is determined by the control signals received from the EMS.
The engine, motor, and generator work in the given state; 3) performance analysis: the energy sources output power, with the engine, battery, and other models to calculate and record the fuel and power consumption. Upon completion of the driving cycle, pertinent metrics are employed to assess the efficacy of the EMS. Meanwhile, during the solving process, relevant physical quantities should be constrained: SOC, torque, and speed should all be within a certain range.

2.2. Reinforcement learning and Markov decision process

Reinforcement learning sets an agent to interact with its environment, and the agent spontaneously updates its policy for the interaction through the rewards it receives [39]. In the RL-based EMS research, the agent is responsible for controlling the distribution of energy flow. The environment encompasses the vehicle model and driving cycles. The vehicle’s performance is measured by the rewards earned by the agent, which are calculated by reward function. An essential feature of reinforcement learning is the self-learning ability of the agent. As depicted in Fig. 4, the agent observes the environmental state and chooses an action based on its current policy. The action selection results from the balance between exploration and exploitation. The agent has to try different control strategies to enrich its experience. At the same time, it also has to ensure the efficiency of exploration and avoid useless and dangerous exploration. The environment is influenced by the action, returning a reward and presenting a new state [40], providing experiences for the agent to conduct policy iteration.

The interaction dynamics between the agent and the environment are commonly formulated as a Markov decision process (MDP), where the state at the next step is solely dependent on the current state and unrelated to the preceding state [41]. At each time step, the agent receives observations indicating the vehicle’s condition from the environment and chooses an action to dictate the energy distribution.

Fig. 2. Powertrain architectures of FCEV, (a) FC + B, (b) FC + SC + B.

Fig. 3. General process of MPS-EV energy management under the control of RL.
distribution among multiple power sources. After one time step, the vehicle is affected by the action of the agent and enters a new state. The reward is calculated and returned to the agent. This process continues to cycle until the end of the driving cycles and constitutes a typical Markov decision process [42]. It is worth noting that in the current research, the Markov process of RL-EMS is completely observable. The state transition probability matrix of vehicle speed in the Markov chain of one iteration is basically unchanged and known. There is no hidden variable influence. The Markov decision process includes five elements: \((X, A, P_{XA}, \gamma, R)\) [43]. \(X\) represents the state space of the Markov process, which contains all possible values of the state variables. \(A\) represents the action space of the agent. \(P_{XA}\) represents the state transition probability for each \(X\) and action \(A\), \(P_{XA}\) gives the distribution of the next state that the environment will transition to after agent take action \(A\) in state \(X\). \(\gamma\) represents the discount rate, and the value is between 0 and 1. By adjusting this value, the agent can balance the importance of the current reward and subsequent rewards. \(R\) represents the reward obtained by the agent after making the action. These five elements are essential factors in the reinforcement learning training process.

**Fig. 4.** Interaction between agent and environment. (a) Interaction process of RL, (b) Interaction process of RL-EMS.

**Fig. 5.** Driving cycles that often occur in RL-EMS studies.

Abbreviation. UDDS: Urban Dynamometer Driving Schedule. WLTC: World Light Vehicle Test Cycle. NEDC: New European Driving Cycle. CTUDC: China Typical Urban Driving Cycle. NYCC: New York City Cycle. FTP-75: Federal Test Procedure. JC08: Japanese JC08. JN1015: Japanese 10–15. WVUCITY: West Virginia University City Driving Cycle. HEFET: Highway Fuel Economy Test. LA92: Los Angeles 92 Driving Cycle. MANHATIAN: Manhattan Driving Cycle. WVUSUB: West Virginia Suburban Driving Cycle. CCDC: China Car Driving Cycle. WVUINTER: West Virginia Interstate Driving Cycle.
3. RL-EMS studies for MPS-EV

The recent research in multi-power source vehicle energy management is classified and discussed in this section. From the RL framework and MDP elements introduced in the previous section, it is clear that after selecting the algorithms used, the RL-based EMS can be trained by completing the modeling of the key factors in the MDP. This section first introduces the algorithms used in the recent research according to the order of development from reinforcement learning to deep reinforcement learning, listing the advances made by various algorithms in the EMS field; secondly, the state variables perceived by the agent in the MDP are enumerated in detail; then the selection of actions for the agent and the corresponding control framework are described; the settings of the reward functions are shown categorically; finally, the novel training method based on the practical characteristics of the energy management problem is presented. Fig. 5 and Table 2 show the basic settings of the different types of algorithms in MPS-EV energy management and the recurrent driving cycles in the training data.

3.1. Algorithms used in RL-EMS

From reinforcement algorithms to deep reinforcement learning which has emerged in recent years. Various RL algorithms and their variants are presented with their applications in the MPS-EV EMS field.

| Table 2 | Solutions in RL-EMS for MPS-EVs. |
|---------|----------------------------------|
| **Algorithm** | **Configuration** | **State** | **Action** | **Reward** | **Reference** |
| Q-learning | HEV (parallel) | v, Tmean | Tm | fuel, elec | [47] |
| HEV&PHEV (parallel) | v, SOC, Pedem | Tm | fuel, elec | [50] |
| HETV | v, SO, Pedem | I | fuel | [51] |
| SARSA | PHEB | v, SOC, Pedem | Peng, G | fuel, elec | [56] |
| HETV | SOC, ng | Teng | fuel, ΔSOC | [48] |
| FCEV (PC + Battery) | SOC, ped | DOH | ΔSOC, deg | [59] |
| Dyna | HEV (series) | SOC, Pedem | Peng | fuel, ΔSOC | [65] |
| PHEV (series) | SOC, ng | Teng | fuel, ΔSOC | [62] |
| HETV | SOC, neng | Teng | fuel, ΔSOC | [61] |
| DQN | HEV (series) | SOC, Pedem, ng | Thref | fuel, ΔSOC | [63] |
| HEV (parallel) | SOC, Pedem, dis | Teng | fuel | [71] |
| PHEV (series-parallel) | SOC, Pedem, dis | Peng | fuel | [78] |
| PHEB | v, acc, SOC, Peng | ΔPeng | fuel, ΔSOC | [67] |
| HETV | SOC, Pedem, ng | Threng | fuel, ΔSOC | [69] |
| FCEV (PC + Battery) | SOC, Pedem, ng | Threng | fuel, ΔSOC | [72] |
| DDPG | PHEV (series-parallel) | SOC, Pedem, dis | Threng | fuel, ΔSOC | [73] |
| HEV (parallel) | SOC, Pedem, ng | Threng | fuel | [71] |
| HETV | v, SOC, SOC, Peng | ΔPeng | fuel, ΔSOC | [68] |
| TD3 | v, acc, SOC | SOC, Pedem, ng | Threng | fuel, ΔSOC | [81] |
| PHEV (series-parallel) | SOC, Pedem, ng | Threng | fuel, ΔSOC, deg | [82] |
| FCEV (PC + Battery) | SOC, Pedem, ng | Threng | fuel, ΔSOC | [82] |
| A2C | v, acc, SOC, Pedem | ΔPeng | fuel, ΔSOC | [89] |
| HEV (series-parallel) | v, acc, SOC, Pedem | ΔPeng | fuel, ΔSOC | [89] |
| PHEV (series-parallel) | v, acc, SOC, Pedem | ΔPeng | fuel, ΔSOC | [89] |
| SAC | v, acc, SOC | SOC, Pedem, ng | Threng | fuel, ΔSOC | [90] |
| HEB | SOC, Pedem, ng | Threng | fuel, ΔSOC | [90] |

Abbreviation. v Vehicle speed. Tmean Torque demand. Teng Torque of engine. Tdem Torque of motor. PEng Power of engine. Tdeg Degree of hybridization. Tem Battery temperature. vol Battery voltage. np Number of passengers. dist Distance from destination. mod Work mode. θ Road gradient. fuel Fuel consumption. elec Electric energy consumption. deg FC degradation.
3.1.1. Early reinforcement learning

3.1.1.1. Q-learning. As an early reinforcement learning algorithm, Q-learning is one of the temporal difference (TD) algorithms. Serving as the foundation for numerous reinforcement learning algorithms, the iterative process of the TD method necessitates solely the interaction between the agent and the environment, devoid of modeling and supervision. The learning task is usually to generate the highest valuation in the current state or randomly selects the action to explore the impact of the new action on the interaction, and the decision of the agent obeys the $\varepsilon$-greedy strategy. The TD method uses the value-based approach to represent the optimal degree of different state-action under the current policy. The magnitude of the value represents the expectation of subsequent cumulative rewards under the current state-action. To avoid falling into local optimum, the TD algorithms use the Bellman equation to unify the current value with the value of subsequent time steps. The numerical relationship between the current value and the subsequent time-step value is estimated without basing a complex model. In this way, the TD-based algorithms achieve a combination of real-time application and global optimization.

TD can be further divided into single-step temporal difference TD (0) and multi-step temporal difference TD ($\lambda$). In TD (0)-based algorithms, as shown in equation (1), the estimate of this time step requires only the immediate reward and the value of the state-action at the next step, which greatly improves the efficiency:

$$Q(s, a_t) = Q(s, a_t) + \alpha[r_t + \gamma * Q(s_{t+1}, a_{t+1}) - Q(s, a_t)]$$

where $s$, $a$, $r$ denotes the current state, action and reward in the time step $t$, $Q(s, a_t)$ denotes the value of the state-action, $\gamma$ is the discount factor, which reflects the importance of future reward to the current estimate, $\alpha$ denotes the learning rate, which usually decreases as the number of iterations rises.

TD ($\lambda$) needs to obtain reward information and state-action values for the subsequent $n$ time steps to estimate the state-action value of the current time step, as shown in equation (2):

$$G_t^{(n)} = \sum_{i=1}^{n+1} \gamma^{i-1} r_{t+i} + \gamma^{n} Q(s_{t+n}, a_{t+n})$$

$$Q(s, a_t) = Q(s, a_t) + \alpha \left[ (1 - \lambda) \sum_{i=1}^{n} \gamma^{n-i} G_t^{(n)} - Q(s, a_t) \right]$$

TD (0) makes faster use of data to meet the needs of real-time applications, and TD ($\lambda$) iterations are based on more data and are more accurate for value estimation.

The early TD-based algorithms represented by Q-learning stored the estimates of various state-action values in tabular form [45], as shown in Fig. 6. Such a table is called Q-table, and the estimate of the state-action stored in the Q-table is called Q-value. The discretization number is limited because of the size limitation of the Q-table and the impossibility of infinite subdivision of action and state variables, which also causes the defect of control accuracy and perceptual accuracy [46].

The update of the Q-value is the core of the Q-learning, which is a typical Off-policy algorithm. During the iterative process, the decision of the agent obeys the $\varepsilon$-greedy strategy as shown in equation (3), the agent has a certain probability of making the action with the highest valuation in the current state or randomly selects the action to explore the impact of the new action on the interaction, and the value of $\varepsilon$ can be adjusted to balance the exploration and exploitation:

$$\alpha = \begin{cases} \arg\max_s Q(s, a) & \text{if random} > \varepsilon \\ \text{random choice} & \text{if random} < \varepsilon \end{cases}$$

However, the equation used to update the Q-value obeys the greedy strategy. As shown in equation (4), the action used for valuation directly takes the action with the highest value in the next state:

$$Q(s, a_t) = Q(s, a_t) + \alpha \left[ r(s, a_t) + \gamma * \arg\max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s, a_t) \right]$$

As a well-established algorithm, Q-learning and its variants are widely used in the research of MPS-EV.

| Action | State 1 | State 2 | ... | State n |
|--------|---------|---------|-----|---------|
| Action 1 | -1      | 0       | ... | 1       |
| Action 2 | 0       | 2       | ... | 0       |
| ...     | ...     | ...     | ... | 2       |
| Action n | 1       | 3       | -2  | 0       |

Fig. 6. Q-table.
a) Q-learning: Q-learning is one of the TD (0)-based algorithms frequently appearing in RL-EMS research. Xu et al. utilized Q-learning to design an EMS for a parallel HEV [47], resulting in an 8.89% improvement in fuel economy compared to the rule-based approach and a 0.88% improvement compared to ECMS. Liu et al. implemented the Q-learning algorithm in the energy management of a Hybrid Electric Tracked Vehicle (HETV), comparing its performance with the stochastic dynamic programming (SDP)-based energy management method. Simulation results indicated that the Q-learning algorithm achieved superior economic performance compared to SDP, with significantly reduced training time. Under testing conditions, the control performance of the employed method closely approximates that of SDP, indicating the robustness of the applied approach [48]. Ye et al. employed Q-learning to develop an EMS that considered battery life and fuel savings, resulting in reduced fuel consumption while mitigating battery degradation [49]. Xu et al. combined multiple Q-learning agents with the same action and reward function settings for forming an ensemble RL framework. Multiple Q-learning agents made action decisions together by weighted averaging. The agents could be integrated during the learning process by sharing rewards. They could also be integrated to output actions jointly after the training process was completed. The results indicated that the fuel economy achieved by multiple Q-learning agents was 3.2% higher compared to the optimal fuel economy attained by the best single agent [50].

b) Q-learning (λ): Q-learning (λ) is one of the algorithms based on TD (λ), which is one of the first RL algorithms employed in MPS-EV EMS applications. There was research applying Q-learning (λ) to the energy management of MPS-EV before 2016. Xue et al. used Q-learning (λ) to train the EMS of HEV and PHEV [51]. Two vehicle models were trained under ten working conditions and compared with the rule-based approach. The results showed an average reduction in fuel consumption of 28.8% for HEV and 30.4% for PHEV. The proposed method achieved better economy while the iterations of the algorithm during the training process converged within 3 h, which is much less than the vehicle service life, demonstrating the high solution efficiency and real-time application potential of the proposed method. The Q-learning (λ) algorithm was also used to manage the energy flow of a hybrid electric energy storage system with a 25% improvement in efficiency compared to a storage system using only lithium batteries [52]. Xue et al. set the battery discharge current and gear ratio as actions to optimize the EMS of parallel HEV and PHEV using Q-learning (λ) and calculate the total driving cost by combining the results of fuel consumption and battery losses. The results demonstrated that the proposed EMS could achieve a reduction in operating costs of up to 48%, as compared to the rule-based EMS [53].

c) Speedy Q-learning: In order to speed up the convergence rate, an algorithm called Fast/Speedy Q-learning has also been introduced into the EMS domain, which modifies the traditional Bellman equation as shown in equation (5):

\[
\begin{align*}
Q_{k+1}(s,a) &= Q_k(s,a) + \alpha \left[ r(s,a) + \gamma \max_a Q_{k+1}(s',a) - Q_k(s,a) \right] \\
&= Q_k(s,a) + \alpha \left[ r(s,a) + \gamma \max_a Q_{k}(s',a) - Q_k(s,a) \right] + (1-\alpha)(Q_{k}(s,a) - Q_k(s,a))
\end{align*}
\]

where \(Q_k\) and \(Q_{k+1}\) denote the valuation of the state-action by Q-tables for the current time step and the next time step, respectively; \(Du\) and \(Q\) denote the current state and action, respectively; \(\epsilon\) is a small positive constant. The proposed method could significantly improve the fuel economy by up to 21% compared with the conventional charge-depleting, charge-sustaining method, showing great potential in practical applications. The SARSA algorithm was trained to control the fuel cell power of FCEV under different fuel cell switching states, demand torque, and SOC states [57]. Liu et al. also used SARSA for state-action valuation in EMS [58]. Compared with Q-learning, the SARSA valuation is more conservative due to its On-policy feature, which allows SARSA to achieve better training results in some cases. SARSA and Q-learning were used to develop the EMS of an FCEV and were compared with each other [59]. The training results showed that the SARSA-based EMS kept the battery in better condition for working.

3.1.1.2. SARSA. Like Q-learning, SARSA is a TD-based algorithm that forms policy by estimating state-action values. SARSA also uses the tabular form to store information, and the valuation process is basically the same as Q-learning. The difference is that SARSA is an On-policy algorithm where the agent interacts with the environment using the same strategy as the one used to update the state-action value as shown in equation (6):

\[
Q(s,a) = \begin{cases} 
\arg\max_a Q(s',a) & \text{if } \text{random} > \epsilon \\
\text{random choice} & \text{if } \text{random} < \epsilon 
\end{cases}
\]

As shown above, in the equation for updating the Q-value, SARSA values the state-action value of the next state and updates the current state-action value according to the \(\epsilon\)-greedy strategy.

Chen et al. used the SARSA to solve the MPS-EV EMS problem [56]. Experimental results under real driving cycles showed that the proposed method could significantly improve the fuel economy by up to 21% compared with the conventional charge-depleting, charge-sustaining method, showing great potential in practical applications. The SARSA algorithm was trained to control the fuel cell power of FCEV under different fuel cell switching states, demand torque, and SOC states [57]. Liu et al. also used SARSA for state-action valuation in EMS [58]. Compared with Q-learning, the SARSA valuation is more conservative due to its On-policy feature, which allows SARSA to achieve better training results in some cases. SARSA and Q-learning were used to develop the EMS of an FCEV and were compared with each other [59]. The training results showed that the SARSA-based EMS kept the battery in better condition for working.

3.1.1.3. Dyna. The RL agent acquires sequences of states, actions, and rewards through real or simulated interactions with the environment to facilitate learning. An agent that learns directly from experiences gained in the actual environment is termed a direct RL agent, whereas an agent that learns from simulated experiences derived from a model is referred to as an indirect RL agent. Direct
RL, exemplified by Q-learning, involves learning directly from experiences generated by the interaction between the agent and the environment. Dyna is an artificial intelligence (AI) architecture that integrates learning, planning, and reacting [60], as shown in Fig. 7. In Dyna, real experience plays two roles: To refine the agent strategy through learning. It is also used to refine the model to make it more accurate to simulate the environment and subsequently improve the agent strategy based on the environment model.

Teng et al. extracted the transition probability matrix of demanded power at different speed ranges based on the experimental driving cycle information and used them to train a Dyna-based EMS [61]. The results showed that the fuel consumption of the EMS trained by Dyna was approximately the same as that of the SDP-based EMS, while the computational cost of Dyna was significantly lower than that of SDP. A related study applied the Dyna-H algorithm by combining the heuristic planning strategy with the Dyna algorithm. Liu et al. applied the proposed method to a parallel HEV, which is closer to the optimization effect of DP than the traditional Q-learning and has a shorter convergence time than Q-learning [62]. Dyna-H was also applied to the EMS development of HETV. The results showed that Dyna-H maintains SOC stability better while achieving a better fuel economy than conventional Dyna [63]. Q-learning, Dyna, and Dyna-H are compared and analyzed in an EMS for PHEV [64], and the authors also compared the training effect of different planning steps. Yang et al. introduced two new strategies in the Dyna framework, backward focusing and prioritized sweeping, to develop the queue-Dyna algorithm [65]. The Q-learning and Dyna were trained separately to compare with the proposed method. The results indicated a significant enhancement in learning speed through the algorithm, while concurrently maintaining relatively low fuel consumption.

3.1.2. Deep reinforcement learning

3.1.2.1. Deep Q-network. Reinforcement learning algorithms such as Q-learning and SARSA discretize the continuous state space and adopt the tabular form to record the estimation of state-action values. However, in complex control problems, where the agent needs to perceive state changes more precisely, the Q-table in a high-dimensional state space would lead to difficulties in storage and search. According to the results of the research on the influence of parameters on Q-learning-based EMS [68], the fuel economy of Q-learning-based EMS decreases when the discretization number of state variables increases, which also proves the shortcomings of Q-learning in terms of perceptual accuracy. Researchers have used neural networks instead of Q-table to address these issues. The DeepMind team used deep neural networks (DNNs) to fit the state-action value and proposed the deep Q-learning (DQL) algorithm to solve the Atari 2600 game [46]. With this approach, an agent can perceive any slight change in the continuous state space, and the Deep Q-network (DQN) approach is also considered a pioneering work in deep reinforcement learning. With deep neural networks, DQN can handle continuous state spaces. As shown in Fig. 8, after inputting a state vector, the neural network outputs the state-action values corresponding to different actions under this state, achieving the goal of allowing the agent to control the energy flow in continuous state space. DQN and its variants are widely used in MPS-EV EMS research.

a) The initial version of DQN: The initial version of DQN got the attention of researchers in the EMS field quickly after its being invented. Wu et al. used DQN to train EMS [67]. To show the advantages of decision-making in continuous state space, Q-learning based on the same model was also trained for comparison. On the training driving cycle WVUSUB, DQN achieved a fuel economy of 89.6 % compared to DP, while the value of Q-learning was 84 %. The test conditions are a combination of WVUSUB, JN1015, CTUDC, and other driving cycles. DQN achieved similar fuel economy results to DP while demonstrating the strong adaptability of the DQN algorithm.

b) Double DQN: Double DQN is an improved version that addresses the shortcomings of the initial DQN, as shown in Fig. 9. During the iterative process, the initial version of the DQN uses the same deep neural network to select the action and evaluate the state-action value in the next step, which leads to a biased estimation of the state-action value. To solve this problem, Double DQN (DDQN) is updated by two DNNs [68], the evaluate network and the target network, respectively. The evaluate network represents the behavioral strategy of the Double DQN, accepts the current state and chooses action to generate trajectories. These trajectories would be stored in the replay buffer, from which the target and evaluate networks would sample batches for parameter update. The sampling approach of the Double DQN disrupts the order in which trajectories are arranged so that adjacent trajectories are not highly correlated, reducing the variance of updates and improving the efficiency of data use. The target network can be synchronized with the evaluate network by hard or soft updates. The estimates generated by the old parameters can be unaffected by
the latest parameters of the evaluate network, thus significantly reducing the divergence and oscillation and improving the stability of the neural network. Given the aforementioned advantages, Double DQN finds extensive application in the EMS of MPS-EV.

Han et al. used Double DQN to train a HTEV-based EMS. Compared with the initial version of DQN, Double DQN achieved better results in terms of convergence speed and fuel economy. The validation results on test conditions demonstrated that the improvements made by Double DQN to the initial version of DQN enhance the robustness of the algorithm [69]. The Double DQN enhanced EMS was subjected to comparative analysis against the rule-based EMS [70]. The simulation shows that, on both training and testing driving cycles, the Double DQN-based EMS exhibited reduced fuel consumption in comparison to the rule-based EMS. Song et al. used Double DQN to develop EMS. The results of training data showed that the SOC curves of DP and the proposed method follow similar trends, and the difference between them in terms of fuel consumption is less than 6%, showing the ability of Double DQN to approximate the global optimum [71]. He et al. used Double DQN and Q-learning to verify the improvement of DQL on RL, and the results showed that the DQL-based EMS achieved better economy both in the training and testing driving cycles [72].

While the Double DQN algorithm has achieved good results in simulations, some novel approaches have been introduced into the Double DQN to improve the sampling efficiency of the experience and the update efficiency of the network. The heuristic experience replay (HER) was introduced to the framework of Double DQN [73], which achieved more reasonable experience sampling during the iteration of the algorithm. An adaptive moment estimation optimization method with the Nesterov accelerated gradient was also used to update the neural network parameters to accelerate the convergence. Simulation results showed that the proposed method achieved faster training speed and higher fuel economy and approached the global optimum compared to existing DQN methods. A Double DQN with preferred experience playback (PER) was applied to the EMS of FCEV [74], and the learning capability was compared with the standard Double DQN in the research. The results showed that the Double DQN with PER obtained a higher cumulative reward after convergence during the iterative process and kept reward values more stable. Du et al. employed a novel optimization technique, AMSGrad, for updating the neural network weights, and conducted a comparative assessment with the benchmark method DP and the conventional DQL [63]. The results demonstrated that the suggested deep reinforcement learning approach exhibited accelerated training speed and decreased fuel consumption in contrast to the traditional DQL. Zhang et al. proposed a Double DQN-based EMS

---

**Fig. 8.** DNN outputs state-action values.

**Fig. 9.** Double DQN algorithm structure.
method for FCEV [75]. PER was incorporated to enhance DQN algorithm convergence speed. Through simulation comparisons with DP and Q-learning, the results showed that the proposed method improved fuel economy by an average of 3.63% compared to Q-learning. The fuel economy difference between the proposed method and DP averaged within 6%, and the convergence speed of the proposed method was enhanced by an average of 30.54% compared to Double DQN without PER. The Dueling DQN divides the network output of the original DQN algorithm into two parts: the state-based value and the advantage of the action in this state [76], and calculates the action-independent state values separately, which improves the efficiency of the computation and enhances the robustness of the algorithm. Li et al. used Dueling DQN to develop EMS for HEVs and improved fuel economy by 3% compared to standard DQN [77]. Qi. et al. used Dueling DQN and showed that Dueling DQN obtained a faster convergence rate than the Double DQN [78].

3.1.2.2. Deep deterministic policy gradient. Because of the size limitation of the output layer, DQN needs to keep the discretization number of actions within a certain range. Some researchers have improved DQN to output continuous actions [144], but the commonly used DQN cannot be applied in continuous control spaces. Unlike DQN, Deep deterministic policy gradient (DDPG) directly outputs the continuous distribution for the action selection, and the specific value of action would be derived by sampling, and any value of the action may be selected; thus, DDPG solves the problem that deep Q networks cannot be applied to continuous action space [79], as shown in Fig. 10.

DDPG adopts the actor-critic architecture, where the actor network outputs action, and the critic network estimates the state-action value. The critic network outputs the state-action value to update the parameters of the actor network, and the algorithm calculates the deviation of the critic network estimation based on the reward returned by the environment and updates it. Meanwhile, DDPG incorporates the target network from DQN and employs experience replay to mitigate sample correlation and reduce estimation bias.

a) Standard DDPG: With the help of DDPG, related research [36,37,80] successfully attained control within the continuous action space, resulting in heightened precision for energy management problems. This increased control accuracy contributes to a seamless transition process within the MDP and augments the robustness of the RL-EMS. Roman et al. used DDPG to develop an EMS, and the training results in all four driving cycles approached the fuel economy of DP, demonstrating that the DDPG algorithm that outputs actions in continuous space can be used to train a better-performing EMS. Ma et al. trained the DDPG network under a real driving cycle. Compared with DP, DDPG achieved 91.3% of fuel economy, while Q-learning only reached 84.9% [81]. Han et al. studied the DDPG algorithm from three aspects: learning ability, generalization, and robustness [82]. The results showed that the fuel economy of DDPG is better than that of double DQN in both test and training data. In addition, the total fuel consumption curve and final SOC curve of each iteration of DDPG during the iteration process have less vibration, indicating that the training effect of the proposed method is more stable than DQN. DDPG was also used to train EMS based on FCEV [83]. During the training process, the neural network parameters converged within 1200 iterations, showing the algorithm’s fast learning speed. The fuel consumption under the three driving cycles has reached a gap of less than 10% compared with that of DP.

b) TD3: Twin Delayed Deep Deterministic Policy Gradient (TD3) [84] is an improvement of the deep deterministic policy gradient algorithm. To address the issue of overestimating state-action values, TD3 employs two critic target networks for value estimation and selects the output corresponding to the smaller value among them. The actor network undergoes less frequent updates compared to the critic network.

The application of TD3 for the training of EMS in HEV was discussed [85]. An innovative hybrid empirical replay (HER) method was introduced, utilizing the mixed experience buffer (MEB). The MEB incorporates both offline computed optimal experiences and

![Fig. 10. DDPG algorithm structure.](image-url)
online learned experiences, thereby enhancing the efficiency of leveraging learning experiences. Under varied driving conditions, the enhanced TD3 EMS achieved superior fuel economy, the swiftest convergence speed, and the highest level of robustness when compared to DQN and DDPG. In an EMS for HEV [86], the training effectiveness and convergence speed of TD3 and DDPG were compared under two different reward function settings. The results showed that the TD3-based EMS exhibited higher fuel economy and faster convergence speed in the task with a relatively complex reward function. However, in the task with a simple reward function, the fuel economy of the TD3-based EMS was similar to that of the DDPG-based EMS, and TD3 convergence time was much higher than that of DDPG. These results showed that TD3 is more suitable for more complex control problems due to its more complex network structure and process. A TD3-based EMS for FCEV was presented [87]. The TD3-based EMS was trained under randomly generated driving conditions and tested under real driving conditions with different numbers of passengers. The comparison with the benchmark Pontryagin’s Minimum Principle (PMP) algorithm showed that the hydrogen consumption of EMS based on TD3 was at most 3.6 % higher than that of PMP.

3.1.2.3. Advantage Actor Critic. Advantage Actor Critic (A2C) uses the state value as the benchmark to reduce the variance of the estimate and updates the actor behavior strategy and the critic evaluation function with the state value [143]. The algorithm flow chart is shown in Fig. 11.

Tan et al. applied the A2C algorithm to EMS [88] and compared the proposed algorithm with DP, rule-based, and Q-learning, demonstrating that the A2C algorithm can develop an EMS with high fuel economy and low training cost. In related research, the reinforcement learning framework with a distributed structure was introduced to improve training efficiency in complex problems. Tang et al. proposed EMS with asynchronous advantage actor-critic, compared with standard DQN. Simulation results showed that the distributed A2C algorithm improves the learning efficiency by a factor of 4 compared to DQN [89]. The distributed A2C was used in combination with the Markov chain model (MCM) to develop an online update framework for EMS [90].

3.1.2.4. Proximal Policy Optimization. As an algorithm that can output continuous actions in a continuous state space, the neural network of Proximal Policy Optimization (PPO) [91] outputs continuous actions according to the corresponding probability. Some algorithms perform a completely new sampling after the neural network parameters are updated, causing the problem of low data utilization. To avoid this drawback, PPO sets two actor networks to participate in the update, and the experience obtained by one actor interacting with the environment is used for the update of another actor, as shown in Fig. 12. The Kullback-Leibler (KL) divergence is used as the basis to measure the difference between the strategic distribution of two actors, reusing the accuracy of the learning experience while improving the training speed.

EMS based on PPO and TD3 were compared [35]. The results showed that under WLTC, PPO had 9.02 % lower fuel consumption than TD3 and lower engine start time and transient operating percentage than TD3. Hofstetter et al. designed a PPO-based EMS for HEV [92]. The average fuel reduction under the tested working conditions was 3.1 % compared to the reference strategy, and the pollution emissions were also reduced. Liu et al. used PPO in combination with transfer learning (TL) to develop a real-time EMS [38].

3.1.2.5E. Soft Actor Critic. Soft Actor Critic (SAC) is an Off-policy actor-critic algorithm based on the maximum entropy reinforcement learning framework [93]. The actor network is shown in Fig. 13 (a) and the critic network is shown in Fig. 13 (b). The optimization objective of SAC entails simultaneously maximizing the cumulative reward for the agent and maximizing the entropy value of the current policy $\pi_{\theta}$, as expressed in equation (7):

$$\max_{\pi_{\theta}} \mathbb{E} \left[ \sum_{t} \gamma^t \left( r(s_{t}, a_{t}) + \alpha \mathcal{H}(\pi_{\theta}(-|s_t)) \right) \right]$$  

(7)

Fig. 11. A2C algorithm structure.
Fig. 12. PPO algorithm structure.

Fig. 13. SAC algorithm structure, a) actor, b) critic.

Table 3

| Algorithm | Strength | Concern | Feature | References |
|-----------|----------|---------|---------|------------|
| Q-learning | Simple control logic | Low degree of accuracy | Possible online implementation | [47,50,51] |
| SARS | Low computation complexity | Dependency on discretization Precision | Limited energy-saving potential | [56,58,59] |
| Dyna | High data-using efficiency | Complex mathematics | Model-based | [61,62,64] |
| DQN | Incremental model learning | More computational expense | Long offline computing time | |
| DDPG | Good fitness | Over-estimation | Value-based | [67,71,78] |
| Continuous control | Suitable for continuous state space | Slow convergence | Accurate perception of the state | |
| Online performance | Possible local optimum | Slow policy update | |
| PPO | High sample efficiency | Sensitive to the parameters | On-policy | [35,38] |
| Fast convergence | Possible local optimum | | |
| SAC | Strong robustness | Dependency on reward scale | Better fuel economy | |
| Strong exploration ability | | | Flexible strategy | [96] |
Where $\mathcal{H}(\pi_\theta(s))$ represents the entropy of the current strategy, the larger this value is, the more random the strategy is. The parameter $\alpha$ serves as the entropy regularization factor in Soft Actor-Critic (SAC), contributing to the equilibrium between exploration and exploitation [94]. The inclusion of strategy entropy in the optimization objective holds significance in maximizing the reward while promoting flexible agent behavior.

SAC was applied to develop an EMS for hybrid energy storage systems [95] and compared with DQN, DDPG, and DP. In terms of learning capability, the convergence speed of SAC was slightly lower than that of DQN during the iterative process, but SAC had the highest cumulative reward after convergence. The fuel economy performance of SAC under the validation driving cycles was also the best, demonstrating the adaptability of the proposed algorithm. In addition, the computation time of SAC under the validation driving cycles was shorter than the running time of the other two DRL EMSs. These results showed the potential of SAC-based EMS for real-time applications. Wu et al. added the degradation cost of the battery to the available metrics to develop an EMS for HEV using SAC. The research tested the proposed EMS under different driving cycles. The results showed that SAC surpassed the existing DQN method in terms of convergence effectiveness and reduction in driving costs [96]. In an EMS for PHEV [97], SAC was compared with DDPG, and the results showed that the SAC-based EMS outperformed the DDPG-based EMS in the economy, and the convergence of SAC was about 36 % faster than that of DDPG.

The advantages and disadvantages of various algorithms are summarized in Table 3.

### 3.2. RL-EMS perception scheme design

In MDP, the agent makes decisions based on states. The state observed by the agent needs to describe the information needed for decision-making completely. A reasonable perception scheme design can help the agent make a more accurate estimation of state-action value and improve the training speed and effect of the algorithm [98]. In energy management problems, the agent needs to decide the distribution of energy flow only based on the specific state of the vehicle during the driving process. Deep reinforcement learning algorithms have solved the problem of perceptual accuracy, and the choice of state variables is essential for the perception scheme of advanced reinforcement learning algorithms. The principle of selecting state variables is to represent the noteworthy variable transition during the driving process, enhance the agent’s perception, and provide references for control signals. With the development of connectivity technology, Vehicle to Everything (V2X) technology has been utilized to obtain various state variables outside the vehicle to characterize or predict the traffic. This section explores the state perception schemes prevalent in RL-EMS research in terms of both in-vehicle and out-of-vehicle information.

#### 3.2.1. In-vehicle information

The relevant physical quantities of the vehicle itself are reflections of the vehicle’s working condition, as shown in Table 4. Speed [50,51,53,56], acceleration [37,80,82,87], SOC [50,66,67,77], and demand power [51,56,63,65] are the most intuitive and widely utilized state variables for the vehicle. Given the interconnected nature of these variables, the RL-EMS can effectively discern the vehicle’s operational state by selecting a subset of these variables. The choice of state variables varies depending on the vehicle type. For example, an EMS for a Plug-in Hybrid Electric Bus (PHEB) [99] chose to record the number of passengers in the vehicle as a state. Some special vehicles necessitate distinct configurations of state variables. For instance, in an EMS designed for HETV, the state variables encompass SOC, demand power, longitudinal speed, and angular speed [69]. Assuming a comprehensive description of the powertrain’s operational state, the configurations of state variables within vehicles of the same type typically exhibit minimal differences.

#### 3.2.2. Out-of-vehicle information

With the development of vehicular networking technology and related research, as shown in Fig. 14, vehicle perception of the traffic provided by Vehicle-to-everything (V2X) began to guide MPS-EV energy management, and a series of information outside the vehicle has been introduced into the state variables. For example, Chen et al. gathered a substantial dataset of high-definition images from five representative roads in the racing game Dust Rally 2.0. This dataset was employed to construct an environment model

| Information sources             | State variable                              | Reference               |
|---------------------------------|---------------------------------------------|-------------------------|
| In-vehicle information          | Speed                                      | [50,51,53,56]           |
|                                 | Acceleration                               | [37,80,82,87]           |
|                                 | Power demand                               | [51,56,63,65]           |
|                                 | Torque demand                              | [36,50,53,71]           |
|                                 | SOC                                        | [56,58,70,77]           |
|                                 | Gear ratio                                 | [36,50,66]              |
|                                 | Speed of generator                         | [48,54,55]              |
| Out-of-vehicle information      | V2V                                         | Distance from surrounding vehicles [102-104] |
|                                 | V2I                                         | Speed/acceleration of surrounding vehicles [102-104] |
|                                 | V2N                                         | Traffic light [103,104,106] |
|                                 |                                            | GPS location [129,108]  |
|                                 |                                            | Terrain [199]           |
encompassing various variables, including driving images, road gradient, longitudinal speed, and the number of passengers [100]. The ability to perceive information outside the vehicle has driven research on integrating V2X with the EMS.

3.2.2.1. V2V information. Vehicle to Vehicle (V2V) technology is not limited to fixed base stations and provides direct end-to-end wireless communication for vehicles. With V2V communication technology, vehicle terminals share wireless information directly with each other without forwarding through a base station [101]. The use of V2V information by RL-EMS focuses on obtaining the distance between vehicles and the speed of other vehicles in the traffic flow, both of which can influence the estimation of the vehicle’s future speed, showing the current and expected speed magnitude.

Li et al. incorporated preceding vehicle speed, preceding vehicle acceleration, and distance to the preceding vehicle into the state variables of a DDPG-based EMS [102]. Inuzuka et al. proposed an EMS introducing road congestion into the perception scheme. Distance to the preceding vehicle, preceding vehicle speed, preceding vehicle acceleration, distance between the vehicle in front and two vehicles ahead, speed of two vehicles ahead, and acceleration of two vehicles ahead were obtained as state variables. The traffic congestion level was quantified as a state for the reference of the agent decision-making [103]. He et al. proposed a cyber-physical system (CPS)-based EMS, calculated the average distance between vehicles in the traffic flow and the speed of the vehicle ahead by V2V. These information were used as the state variables to represent the traffic. The experimental results showed that the effectiveness of both DDPG and DQN was improved after the introduction of V2V information [104].

3.2.2.2. V2I information. Vehicle to Infrastructure (V2I) provides the information exchange between vehicles and road infrastructure. V2I technology allows vehicles to communicate with road infrastructure to transmit the vehicle’s speed and location to the central server. The server records the speed and location of all vehicles and provides this data for some applications [105]. With V2I information, vehicles can be informed of traffic signals and the traffic conditions of the road ahead.

In RL-EMS research, V2I technology is mainly used to obtain the signals of traffic lights and the distance of vehicles from the next traffic light. The macroscopic traffic sensing capability of RL-EMS is enhanced by using V2I information as the state variables. In an EMS for HEV [103], traffic information was input to the deep neural network as a basis for decision-making. The state variables include traffic light signals, the number of seconds, and the vehicle’s distance from the next traffic light. The comparison showed that the incorporation of traffic information enhances computational speed, and the fuel economy achieved by the proposed method surpasses that of MPC. Vehicle-to-Infrastructure (V2I) information, including details such as traffic light signals, their timing, and the distance from the traffic light, was acquired as state variables for an EMS [104]. The onboard computer vision hardware and software used the detected visual information, such as traffic lights, as the DDPG network’s state input [106]. The EMS fuel consumption with visual information was reduced by 4.3–8.8 % compared to the DDPG without visual information.

3.2.2.3. V2N information. Vehicle to Network (V2N) refers to the connection of in-vehicle devices to a cloud platform through an access network. The cloud platform and the vehicle interact, and the cloud platform stores and processes the acquired data to provide various application services the vehicle needs [107]. In the RL-EMS study, V2N technology is mainly applied to obtain the vehicle position in Global Positioning System (GPS).

For a given trip, the forthcoming energy consumption demonstrates a pronounced correlation with the remaining distance. Some researchers have used GPS to obtain the vehicle’s location during the trip to guide energy management. As the PHEB with the most stable working conditions, the agent can get the environment’s characteristics based on its position on the line. In an EMS for PHEB [99], the current line was included as one of the state variables. The proposed method achieved an effect close to DP in fuel economy and showed good generalization performance in the test driving cycles. Liu et al. put forward a reinforcement learning approach that considers trip details to cater to the features of PHEV. The algorithm obtained the remaining travel distance from GPS for energy optimization [108]. The agent could better control the SOC by introducing the remaining travel distance. The proposed method achieved better training effects than the rule-based method while maintaining the SOC better. Li et al. integrated historical cumulative trip information into a DDPG-based EMS for practical SOC guidance. The proposed method was systematically introduced from offline training to the online application, and the DP method was used to obtain the SOC trends of working conditions that match the vehicle driving habits. The travel distance after the last charge was added as a state variable to guide the SOC. Compared with MPC-based EMS, the proposed EMS was more adaptive and robust to dynamic driving conditions, and the proposed method could also learn to predict future traffic from historical data [12]. Due to the high mileage and load of commercial vehicles, terrain information was incorporated into the state variables of hybrid commercial vehicles to train the DDPG-based EMS [109]. Experimental results showed that the fuel
gap between the proposed method and the DP-based method was around 6 %. The inclusion of terrain information improved the fuel economy of the proposed method by approximately 2 %.

3.3. RL-EMS decision scheme design

The decision scheme of RL-EMS influences the state transition of MDP. There are different control frameworks of RL algorithms to directly determine or indirectly influence the energy distribution of the MPS-EV. The control frameworks applied in RL-EMS are summarized into three: end-to-end control, hierarchical control, and parallel control, as shown in Fig. 15.

3.3.1. End-to-end framework

Most of the RL-EMS research discussed in this paper employed an end-to-end control framework, depicted in Fig. 15(a). For the RL-EMS decision scheme using the end-to-end control framework, the actions made by the agent can be directly used as the final control signal to decide the energy distribution of the vehicle. In the energy management problem solved by the end-to-end control framework, the control of energy flow by RL-EMS can be divided into continuous and discrete action control. The agent can only pick its action from several predefined values for discrete action control. The control accuracy of the EMS is directly related to the discretization number [66]. The agent can output any value taken in the range of continuous action space for continuous control. The deep reinforcement learning algorithm for continuous control is achieved by sampling over the normal distribution, where the DNN outputs the mean and variance of the normal distribution. Then the final action is determined through sampling [80].

3.3.1.1. Discrete control solution

In RL-EMS research that adopts discrete control, different kinds of physical quantities are discretized. Engine torque [70,71], motor torque [47,50,66], and engine power [65,78] are the action variables commonly used. In an EMS for HEV [66], the motor torque was chosen as the action and was limited to the range of 100 to 250. To verify the effect of control accuracy improvement on the training effectiveness of the Q-learning algorithm, the motor torque was uniformly discretized into different numbers, 5, 10, 20, 50, and 100. Experimental results showed that increasing the discretization number enhanced the training effectiveness. However, as the discretization number increases, the training effect enhancement becomes smaller. In another Q-learning-based EMS [47], the motor torque was uniformly discretized into 100 values from 90 to +90. In DQN, the limitation of the output layers in the neural network does not allow a too larger number of discretization. Engine torque is selected as action in a DQN-based EMS with a discretization number of 24 [70]. In another DQN-based EMS [71], the action size of the DQN neural network is 10, and the engine torque is uniformly discrete from 0 to 207. Engine power is also often selected as the action in the MPS-EV EMS. Qi et al. defined the actions as 24 power level output from the engine [78]. In an FCEV-based EMS [59], the degree of hybridization (DOH), which is the ratio of fuel cell power to demand power, was uniformly discretized from 0.1 to 0.9 as ten values for the actions of the SARSA algorithm. The increase in discretization number helps to cover different intervals of the action space and to improve the control accuracy. An alternative approach was used to refine the energy distribution of the vehicle while meeting the dimensional requirements. The deep neural network output the action not as a specific value but as the change in the current physical quantity, and the amount of change in the engine power was set to six options [67]. This approach avoids excessive deep neural network dimensionality while achieving high control accuracy. Han et al. used the amount of change in engine speed as the action, selected from eight options [69]. On FCEV-based discrete control, the amount of change in fuel cell power was also used as the action [74].

Fig. 15. Control frameworks in RL-EMS. (a) End-to-end control, (b) Hierarchical control, (c) Parallel control.
3.3.2. Hierarchical control and parallel control in RL-EMS

3.3.2.1. Hierarchical control. As shown in Fig. 15(b), in RL-EMS with a hierarchical control framework, the RL agent does not directly determine energy distribution through its actions. Instead, the upper layer of the framework provides reference actions for the lower layer to follow, which ultimately outputs the control signals.

Li et al. constructed a hierarchical control framework using DDPG and A-ECMS. The framework consisted of two layers: an upper layer where a reinforcement learning agent planned the reference vehicle speed based on traffic information and powertrain state, and a lower layer where the A-ECMS optimized energy distribution based on the torque demand determined by the reference vehicle speed [102]. In an HEV-based EMS [110], the ECMS and RL were combined to develop a hierarchical control framework. The agent output a new co-state, which is similar to the equivalence factor, based on SOC, current co-state, torque demand, predicted power demand, predicted average vehicle speed, and predicted average acceleration. The lower layer decided the energy distribution of the vehicle based on the co-state. The experimental results showed that the improvement of the proposed method outperformed A-ECMS in terms of economy. A new hierarchical control framework was applied to the energy management problem of FCEV [57], in which both the upper and lower layers used reinforcement learning algorithms to output actions. The upper layer output the penalty function through the segment average vehicle speed, the trip average vehicle speed, the initial SOC, and the segment distance. The penalty function affected the optimization objective of the lower layer strategy. The lower layer strategy output the control signal to directly decide the fuel cell power through the penalty function from the upper layer and state variables such as SOC and demand power. Zhou et al. developed an EMS for FCEV based on reinforcement learning and thermostatic control. The RL strategy output upper and lower limits of the battery SOC, and thermostatic control controlled the energy distribution based on the SOC range determined by reinforcement learning. The results showed that the combination of reinforcement learning and thermostatic temperature control ensured that the fuel cell could achieve high fuel economy with stable output power [111]. Chen et al. trained an effective and convergent reinforcement learning controller based on the demand power distribution under multiple working conditions. The reinforcement controller was embedded into a stochastic MPC controller by constructing a multi-step Markov speed prediction model to determine the optimal power of the battery in the prediction time domain. A control flow similar to the hierarchical control framework was proposed [112]. The optimal brake specific fuel consumption curve of the engine was considered the prior knowledge as the upper layer strategy, while the RL algorithm was used for the lower layer. The upper layer strategy determined the appropriate action space for the RL agent in the current state, which simplified the action space of the engine and increased its average efficiency. The RL agent did not need to explore along the global map of the engine, which allowed the algorithm to search for the optimal solution in a smaller space. Compared with conventional RL, the proposed method achieved a better training effect with less computational cost.

3.3.2.2. Parallel control. As shown in Fig. 15(c), in the parallel control framework, the control signals are derived from the collaborative decision-making of different strategies. RL agent outputs actions that provide control signals in parallel with other controllers in the control framework. Wu et al. combined MPC and RL algorithms to develop an EMS [113], with each algorithm outputting actions to control the energy distribution. The fuzzy logic controller output adjustment factors to determine the linear weights of MPC and RL actions to determine the final control signal. Tang et al. proposed a parallel framework to output control signals by decentralized RL [114]. The DQN agent learned the gearshift strategy to determine the transmission ratio and the DDPG agent controlled the throttle opening of the engine. Hu et al. proposed a decentralized reinforcement-learning control framework to develop EMS in modular FCEV. The powertrain consisted of two parallel fuel cell systems and a battery pack. The proposed multi-agent parallel control framework generated a fully distributed learning strategy. The application effect of decentralized reinforcement learning in modular energy system was verified through simulation and experiment [147].

3.4. Reward functions in RL-EMS

RL-EMS aims to form a policy that can output a sequence of actions for the working condition. The ultimate goal of the action sequence is to maximize the reward accumulated for the driving cycle. The setting of the reward function establishes the policy goal of the agent in the MDP [21]. For the MPS-EV EMS problem, the reward function setting needs to achieve at least the goals of reducing fuel consumption and maintaining SOC. In order to quantify the driving cost and improve economic performance, different metrics are applied to the reward function. Different reward function settings are categorized and presented as follows.

3.4.1. Fuel consumption

To avoid specifying weights for different types of energy consumption, Song et al. optimized only fuel consumption. The environment rewarded the agent negatively when the power exceeded the physical limit of SOC to balance the energy distribution among energy sources [71], as shown in equation (8):
The new reward function setting could accelerate the convergence, and the total reward after convergence was higher. Y. Lin et al. adopted a new reward function to maintain power and protect the battery [34]. The training results showed that during the training that too high and too low values cannot achieve such control effect. Values that were too low do not maintain power well, and values results showed that values within a reasonable range can improve the fuel economy to more than 93% of the fuel economy of DP and training effect in a DDPG-based EMS [112]. Lian et al. investigated the effect of different weights on the variation is detrimental to its health. As shown in equation (14), Xt et al. constructed a model to quantify the degradation of the fuel cell and introduced it into the reward function [74]:

\[
G_{bat} = \omega_{bat} \cdot \Delta SOC \cdot I_{bat} \tag{12}
\]

where \( G_{bat} \) represents the power consumption part of the reward function, \( I_{bat} \) represents the battery current, and \( \omega_{bat} \) is the corresponding weight. Such a setting not only reflects the current SOC, but also takes the instantaneous energy consumption of the battery into consideration. As shown in equation (13), Han et al. introduced a new reward function by combining the equivalent fuel consumption with SOC [82].

\[
r = -[\alpha(fuel + elec) + \beta \cdot \Delta SOC^+] \tag{13}
\]

### 3.4.3. Variation of SOC
In the RL-EMS research mentioned in this paper, the combination of fuel consumption and SOC stands out as the prevalent design in the reward function. The typical configurations for the reward function are delineated by equation (11) [48,54,55,58,61,62,65,77]:

\[
r = a -(\gamma \cdot fuel + \delta \cdot elec) \tag{10}
\]

Where \( a \) is a constant, \( fuel \) is the fuel consumption in one time step, \( elec \) is the electricity consumption in one time step, and \( \gamma \), \( \delta \) are the corresponding weights. Such a setup also faces the problem inherent to the ECMS algorithm that the weights need to be set artificially beforehand. When the values do not correspond to the current driving cycle, the training effectiveness of the algorithm will be reduced.

### 3.4.2. Equivalent fuel consumption
Related research [47,56,66] drew on the idea of ECMS to assign weights to different types of energy consumption. The instantaneous equivalent fuel consumption is incorporated into the reward function as shown in equation (10):

\[
r = a -(\gamma \cdot fuel + \delta \cdot elec) \tag{10}
\]

Where \( a \) is a constant, \( fuel \) is the fuel consumption in one time step, \( elec \) is the electricity consumption in one time step, and \( \gamma \), \( \delta \) are the corresponding weights. However, the weights still need to be specified in advance, and the training effect is highly dependent on their values. Lian et al. investigated the effect of different weights on the training effect in a DDPG-based EMS [112]. \( \gamma \) was fixed at 1, and 8 values in the range of 105–3500 were selected and assigned to \( \beta \). The results showed that values within a reasonable range can improve the fuel economy to more than 93% of the fuel economy of DP and that too high and too low values cannot achieve such control effect. Values that were too low do not maintain power well, and values that are too high wasted the buffering effect of the battery.

As shown in equation (12), Kai et al. proposed a TD3-based energy management strategy for FCEV considering fuel cell aging and adopted a new reward function to maintain power and protect the battery [34]. The training results showed that during the training process, the new reward function setting could accelerate the convergence, and the total reward after convergence was higher.

\[
r = a -(\gamma \cdot fuel + \delta \cdot elec) \tag{10}
\]

Where \( a \) is a constant, \( fuel \) is the fuel consumption in one time step, \( elec \) is the electricity consumption in one time step, and \( \gamma \), \( \delta \) are the corresponding weights. However, the weights still need to be specified in advance, and the training effect is highly dependent on their values. Lian et al. investigated the effect of different weights on the training effect in a DDPG-based EMS [112]. \( \gamma \) was fixed at 1, and 8 values in the range of 105–3500 were selected and assigned to \( \beta \). The results showed that values within a reasonable range can improve the fuel economy to more than 93% of the fuel economy of DP and that too high and too low values cannot achieve such control effect. Values that were too low do not maintain power well, and values that are too high wasted the buffering effect of the battery.

As shown in equation (12), Kai et al. proposed a TD3-based energy management strategy for FCEV considering fuel cell aging and adopted a new reward function to maintain power and protect the battery [34]. The training results showed that during the training process, the new reward function setting could accelerate the convergence, and the total reward after convergence was higher.

\[
r = a -(\gamma \cdot fuel + \delta \cdot elec) \tag{10}
\]

Where \( a \) is a constant, \( fuel \) is the fuel consumption in one time step, \( elec \) is the electricity consumption in one time step, and \( \gamma \), \( \delta \) are the corresponding weights. However, the weights still need to be specified in advance, and the training effect is highly dependent on their values. Lian et al. investigated the effect of different weights on the training effect in a DDPG-based EMS [112]. \( \gamma \) was fixed at 1, and 8 values in the range of 105–3500 were selected and assigned to \( \beta \). The results showed that values within a reasonable range can improve the fuel economy to more than 93% of the fuel economy of DP and that too high and too low values cannot achieve such control effect. Values that were too low do not maintain power well, and values that are too high wasted the buffering effect of the battery.

As shown in equation (12), Kai et al. proposed a TD3-based energy management strategy for FCEV considering fuel cell aging and adopted a new reward function to maintain power and protect the battery [34]. The training results showed that during the training process, the new reward function setting could accelerate the convergence, and the total reward after convergence was higher.

\[
r = a -(\gamma \cdot fuel + \delta \cdot elec) \tag{10}
\]
Innovative training methods in RL-EMS.

Table 5 represents the reference battery temperature, and Y. Lin et al. incorporated fluctuations in fuel cell output power into the reward function [143]:

\[ r = - \left[ \alpha \cdot \text{fuel} + \beta \cdot (SOC - 0.7)^2 + \omega D_{\text{FC,deg}} \right] \]  

(14)

Where \( D_{\text{FC,deg}} \), \( \omega \) represents the degree of fuel cell degradation and the corresponding weights. As shown in equation (15), Li et al. incorporated fluctuations in fuel cell output power into the reward function [143]:

\[ r = \begin{cases} \frac{1}{m_{\text{H}} + \omega |\Delta P_{\text{FC}}|} m_{\text{H}} \neq 0 \land SOC_{\text{min}} \leq SOC \leq SOC_{\text{max}} \\ 0 \land SOC < 0.4 \text{ or}\ SOC > SOC_{\text{min}} \\ \frac{1}{SOC + 1} m_{\text{H}} = 0 \land SOC \geq SOC_{\text{min}} \\ \frac{1}{P_{\text{batt}}} m_{\text{H}} = 0 \land SOC < SOC_{\text{min}} \end{cases} \]

(15)

Where \( m_{\text{H}} \) represents the fuel consumption, \( P_{\text{FC}}, P_{\text{batt}} \) represents the fuel cell, and battery power. When the fuel cell power varies too much, the reward value decreases accordingly. In related studies, the temperature of the battery is also limited to prevent the over-temperature [96], as shown in equation (16):

\[ r_{\text{TM}} = \begin{cases} 0 \text{ if } T < T_{\text{tar}} \\ \tau(T - T_{\text{tar}})^2 \text{ if } T \geq T_{\text{tar}} \end{cases} \]

(16)

Where \( r_{\text{TM}} \) represents the part of the reward function related to the battery temperature, \( T \) represents the battery temperature, \( T_{\text{tar}} \) represents the reference battery temperature, and \( \tau \) represents the corresponding weight. In addition to the fuel cell, the engine service life is also strongly influenced by energy distribution. The reward function was subtracted by a fixed constant when frequent start occurred [112].

3.5. Innovative training methods of RL-EMS

After determining the algorithm to be used and completing the modeling of the main factors of the MDP, the RL algorithm can be applied to the EMS problem of MPS-EV. However, unlike some control problems, in the energy management of MPS-EV, the working conditions of the controlled vehicle are not immutable, and the agent’s environment has different styles, which affects the training effectiveness of the proposed method. The long exploration time of the agent to the environment also reduces the training efficiency. In order to make the reinforcement learning closer to the actual conditions of the vehicle, when building the RL-based EMS, some researchers adopted innovative training methods to enhance the adaptability of agents to the changing environment, and some other researchers shortened the training time of the agent by effectively initializing the behavioral strategies. The innovative training methods in RL-EMS are summarized in Table 5.

3.5.1. Adaptability to the changing environment

To effectively apply the reinforcement learning algorithm, it is necessary to provide the agent with a simulation environment for training before transferring it to the actual application. This is due to the algorithm’s reliance on a significant number of samples. In order to be successfully applied in the real world, the RL-EMS needs to have the ability to handle various working conditions to ensure that the strategy can achieve good economics even when the distribution of the training data varies from that of the testing data. Xu et al. investigated the adaptability of an HEV-based EMS developed through Q-learning [115]. Compared with ECMS, MPC, and thermostatic control strategies, experimental results proved that Q-learning demonstrated superior self-learning capabilities when there were changes in driving conditions, vehicle loading conditions, and road classes. Q-learning could adapt to the adjustment of working conditions and modify the strategy during iterations, demonstrating the agent’s adaptability to changing driving conditions due to the self-learning capability of RL. Related research [72,116] applied EMS based on training data to a different style of testing

| Goal | Way | Method | Reference |
|------|-----|--------|-----------|
| Adaptable to changing environment | Diversity of training data/driving cycles | Generate training data online [36,123] |
| | Characterize driving cycle characteristics to output corresponding strategies | Randomly add other driving cycles [117] |
| | Update online when driving style changes | Mark trip distance [118,119] |
| Initialization of agent policy | Initializing the Q-table by expertise | Mark speed [57,80] |
| | | Introducing state transition probability matrix [55,120-123] |
| | Initializing the network based on TL | Give higher value to high-efficiency actions [126] |
| | | Remove low-efficiency state-actions [127] |
| | | Get parameters from other EMS [37,38] |
driving style was judged to have changed. The proposed energy management strategy significantly improved fuel economy and thresholds were analyzed. The proposed method was compared with DP and rule-based and showed a great training effect \[122\]. An power with KL divergence to measure the difference between current and past working conditions, which allowed the EMS to rather than speed ranges. In such research, state transition probability matrices were generated for each time period. The induced convergence speed compared to conventional RL. Lin et al. used Markov chains to calculate the transfer probability matrix of demand \[57\].

3.5.2. Initialization of agent policy

Different styles of driving cycles have different state transition processes. The diversity of training data can undoubtedly enhance the adaptability of RL-EMS. However, during the application process, the agent also needs to identify the current driving style in order to achieve the goal of accelerating the policy update. When the driving style changes, the agent needs to make adjustments to accelerate learning the new driving style. Wang et al. enhanced the diversity of training driving cycles to accommodate trips with different distances and average speeds and characterized trip characteristics in detail. A series of information such as preset driving distance, driven distance, and driven time are used to guide the EMS in making decisions. In the test driving cycles, the proposed approach achieved an average fuel efficiency improvement of 21.8 % compared to the rule-based method \[118\]. In a PHEV-based EMS, the RL algorithm integrates neural dynamic planning (NDP) with forthcoming travel information to assess the anticipated travel cost for the vehicle. The driving cycle was characterized in terms of travel distance. Two machine learning algorithms, Q-Learning-Short Trip (QL-ST) and Q-Learning-Long Trip (QL-LT) were developed for energy control of the PHEV for short-distance and long-distance working conditions, respectively \[119\]. Guo et al. used transfer learning (TL) to adapt to different speed intervals of the controlled vehicle \[80\]. When the speed distribution of the training driving cycle differs significantly from that of the actual driving cycle, it may take a long time to build a new EMS. To solve this problem, a bi-level control framework was constructed. Since speed can represent driving style to a large extent, the algorithm can adopt similar behavioral strategies on different driving cycles with little difference in speed. The upper layer used the DDPG algorithm for EMS training on different driving cycles to develop three DDPG deep neural networks based on the division of speed intervals. The TL approach was then used to initialize the neural network for a new driving cycle, and knowledge transfer was achieved between similar speed intervals of different driving cycles. The results showed that the TL-initialized DDPG algorithm saved about 0.1 SOC and 9.9 % fuel compared with the standard DDPG algorithm on the testing driving cycles, which verified the effectiveness and adaptability of the framework. Yuan et al. classified frequently occurring routes based on the historical working conditions of the vehicle into segments by road location and traffic condition, obtained the long-term average speed of vehicles by k-nearest neighbor method and predicted short-term speed sequences by the new model averaging method. The characteristics of the current working conditions were extracted to enhance the ability to perceive and adapt to the working conditions \[57\].

Related research \[55,120,121\] chose to characterize the current working conditions in terms of state transition probability matrix rather than speed ranges. In such research, state transition probability matrices were generated for each time period. The induced matrix norm was used to quantify the difference between working conditions, and the corresponding strategies were adjusted when the driving style was judged to have changed. The proposed energy management strategy significantly improved fuel economy and convergence speed compared to conventional RL. Lin et al. used Markov chains to calculate the transfer probability matrix of demand power with KL divergence to measure the difference between current and past working conditions, which allowed the EMS to be updated with driving cycles. The computational cost and economic performance of RL-EMS with different KL divergence update thresholds were analyzed. The proposed method was compared with DP and rule-based and showed a great training effect \[122\]. An online update strategy was proposed \[123\]. RL and MPC were combined to design an online update strategy for a line-fixed HEV. The study generated the state transition probability matrix based on Markov chains to predict future working conditions and adjusted them in time. The algorithm used experience replay to accelerate the update of network parameters when the driving condition style changed.

3.5.2. Initialization of agent policy

RL-EMS, as a data-driven unsupervised AI method, requires a long training period to achieve a better energy optimization result, so providing an excellent starting point for the strategy by initializing behavioral strategies through supervised learning is believed to reduce the training cost of reinforcement learning strategies \[124,125\]. In the energy management problem of MPS-EV, the introduction of expert knowledge and other policy parameters to initialize reinforcement learning behavioral policy has been widely used. Unlike the previous research that initiated the Q-learning policy with zero or random Q values, Xu et al. were inspired by the idea of the warm-start to initialize the Q-table values, giving higher initial values to the more efficient actions of the engine. The proposed method reduced the learning iterations of the algorithm compared to the standard Q-Learning \[126\]. Conventional Q-learning combines all states of the powertrain into the Q-table. In an FCEV-based EMS \[127\], the state-actions were optimized based on expertise. The state-action was restricted to a reasonable range of outputs for fuel cells and lithium batteries. This approach reduced the computational cost while improving the training effectiveness.

Related research \[37,38\] used TL for knowledge transfer between hybrid vehicles with different structures to improve the efficiency of EMS development. Lian et al. studied knowledge transfer between four types of HEVs \[37\]. Firstly, the DDPG based on Toyota Prius was trained, the acquired deep neural network parameters were used to initialize the DNNs of the other models of EMS, and the DNNs initialized by TL were used for training. The results showed that the deep neural network initialized by TL improved convergence speed and fuel economy over the randomly initialized neural network. This result showed that reasonable initialization helps accelerate the development process of MPS-EV EMS.
4. Development Tendency of RL-EMS

This section presents trends that have not yet been widely used in the MPS-EV EMS field and explores the challenges and possibilities of applying these trends to RL-EMS in the future. The possible development direction of the algorithm in the future is also introduced.

4.1. Trends in the application of RL algorithms on EMS

As shown in Fig. 16, this paper lists the publishing time of some earlier research mentioned in this paper that applied reinforcement learning to MPS-EV energy management [36,47,61,67,70,85,92,96]. In addition, the time when these algorithms were proposed is also presented. There is a gap between the proposal of the RL algorithms and its gradual application to the MPS-EV energy management problem, which can also be divided into three phases. Before 2013, reinforcement learning had not been applied to energy management. Early reinforcement learning algorithms like Dyna and Q-learning were proposed in the 20th century [60,128]. Despite their great significance in reinforcement learning, there was much room for improvement because they were relatively primitive. For a long time, the reinforcement learning algorithms did not attract the attention of researchers in the field of EMS until 2013, when Deep mind first proposed DQN. Reinforcement learning used deep neural networks to achieve control over a continuous state space, demonstrating its self-learning effect on Atari games [46], and an improved version, Double DQN was proposed in 2015 [129]. After entering the second phase, a series of deep reinforcement learning algorithms were proposed between 2013 and 2018. Due to the influence of the explosive development of reinforcement learning, there has been a proliferation of EMS research based on reinforcement learning. These algorithms have demonstrated the potential of RL-EMS, but at the same time, RL-EMS also has disadvantages such as poor robustness, hyperparameter sensitivity, and difficulty in deployment. With the advancement of communication technology, the expansion of application scenarios, and the complexity of control problems, AI development focuses on big data, big models, and application scenarios with high data flux. After 2018, with the iteration of reinforcement learning algorithms, some latest distributed offline reinforcement learning algorithms emerged with the technical potential to further enhance RL-EMS.

4.2. Differences in the development of RL-EMS and advanced RL algorithms

4.2.1. Introduction of some RL algorithms that have not yet been applied

Although a series of advanced RL algorithms, such as SAC and TD3, have been applied, some advanced offline distributed algorithms have not been explored by relevant researchers in the energy management field for some reasons.

Distributed prioritized experience replay (APEX) [130] proposes a distributed architecture for large-scale deep reinforcement learning that enables learners to learn efficiently from orders of magnitude data, as shown in Fig. 17. APEX allows distributed Actors to acquire experience data and initialize the priority level of experience for traditional DQN and DDPG. The learner updates network parameters based on experience in the database, passes the parameters to actors, and updates the priority level of the experience. Thus, the reinforcement learning network can be trained more efficiently. Studies have shown significant experimental results using this architecture on DQN and DDPG, improving the exploration efficiency with an increasing number of actors [130].

As shown in Fig. 18, Importance Weighted Actor-Learner Architecture (IMPALA) [131] proposes a new offline distributed AI with a classical actor-critic structure, where the learner uses the GPU and the actor uses the CPU. Each actor individually and periodically receives data from the learner to synchronize parameters and then collects data. Given that the experience replay mechanism consumes high memory and the learner’s learning is batch-by-batch serial learning by mini-batch, which has a limitation on data

Fig. 16. Time of RL algorithm proposal and early application in MPS-EV EMS research.

Fig. 17. Time of RL algorithm proposal and early application in MPS-EV EMS research.
throughput, the data collected by all the actors of IMPALA are stored in the data sampling queue instantly. The actors and the learner do not interfere with each other. The actor regularly synchronizes parameters from the learner with no need to calculate the gradient, and the data throughput is larger. Learner quantitatively learns to update parameters. The introduction of V-trace solves the problem that the policy when actors collect data uses is different from the policy when learner learns, which can accept a larger delay between them and facilitate large-scale distributed deployment of actors. As shown in Fig. 18 (a) and Fig. 18 (b), IMPALA supports both single learner and multiple synchronous learners, and when the training scale increases, multiple learners with multiple actors can be considered. Each Learner only gets samples from its own actor to update. The learners exchange gradients and update network parameters periodically, and the actor updates parameters from any learner.

Recurrent Replay Distributed DQN (R2D2) [132] was proposed in 2019, and DeepMind introduced several new improvements to the previous year’s APEX-DQN to produce an even better R2D2 algorithm. Its distributed training framework was used to train in the mode of single learner and 256 actors, achieving high efficiency [132].

4.2.2. The advantages of offline distributed reinforcement learning and the reasons why it has not been applied

In reinforcement learning, the agent needs to generate a large amount of experience by interacting with the environment and learn online based on the experience to update its policy. Data is a crucial part of the training process for reinforcement learning, which can
also be divided into online and offline learning according to the data source. The above three algorithms belong to offline reinforcement learning, in which the agent can learn directly from the data shared in the database without taking action. By offline reinforcement learning, the cost of interaction can be reduced.

Past RL-EMS research has mostly been limited to online integrated AI: single agent learns by its intersection with the environment, learning experiences are generated solely from its own vehicle model and driving cycles with a single source of data, slow data acquisition, and wasted computational resources. However, a broad trend in deep learning is to combine more computation with more powerful models and larger data sets [133]. All three newly developed algorithms are computationally solved at scale as offline distributed AI, sharing data through a central database, enhancing learning sample diversity, and reducing the cost of repeated computations. The use of distributed actors to acquire experience data and upload it to learners makes it possible for learners to learn, acquire experience, and improve strategies from a broader range of scenarios.

The large amount of data requires more workers to interact with the environment to enrich the experience and also requires a computing platform with high data throughput to support the transmission and calculation. Most of the current RL-EMS research uses Matlab + Simulink for simulation, with fixed vehicle models, repetitive training data, and learning experience from a single source. Such a simulation environment does not have the conditions for large-scale data parallelism. In contrast, some distributed AI studies can use up to 44 actors running on CPUs in the experiment to generate data at up to 19K frames per second [134], while the single learner running on a GPU receives a large number of experiences and provides different exploration policies for different actors, expands the diversity of experiences, and makes progress on high-dimensional control problems. An approach like Matlab cannot be deployed in massive clusters of computers. At the same time, the decentralized authorization characteristic of Matlab makes it challenging to deploy and execute remotely, which leads to the inability to extend the research of the automobile control system to the current mainstream heterogeneous computing platform, thus making the latest research results of the general artificial intelligence, including deep reinforcement learning, unable to be conveniently applied.

4.3. Challenges and potential research directions

4.3.1. Challenges

The above researches have made considerable achievements in this field and successfully optimized the energy management strategy of MPS- EV by the reinforcement learning framework. While the methods used have shown good performance, some of the problems and limitations of RL-EMS remain unresolved. First, as a data-driven method, reinforcement learning needs to learn based on a large amount of data. The lack of diversity in the training driving cycle can lead to a lack of solid robustness. Different parameter settings will also affect the learning effect of reinforcement learning. Achieving good performance of RL-EMS in real-world driving conditions is a problem that must be addressed in developing hybrid vehicle energy management strategies. Secondly, there needs to be a universal standard for quantifying the optimization effect of energy management strategies on energy consumption. Existing research tends to demonstrate its effectiveness in simulation environments, needing more hardware testing data. After proving its energy-saving potential, reinforcement learning needs to reduce its application cost. RL-EMS has higher time and computing costs than the rule-based energy management strategy. Finally, with the introduction of connected car technology, massive environmental information can assist in reinforcement learning decision-making. The current literature still needs help in how to integrate it efficiently. The above issues pose challenges to the application of RL in EMS.

4.3.1.1. Dependency on training data and parameters. One of the main differences between the simulation environment and the actual application is the difference in data distribution. RL-EMS research has several related problems: 1) Part of the reinforcement learning research only develops energy management strategies based on a constant cycle. Based on a single data distribution, this strategy is obviously unable to adapt to changing driving styles. 2) In the offline optimization process, reinforcement learning is based on the data distribution of the simulation environment. However, when the actual data distribution is significantly different from the training data distribution, RL-EMS’s ability to save energy consumption may be weakened in online applications. Compared with the samples in the simulation environment, the samples in the real world are limited, and the online update of reinforcement learning has hysteresis characteristics. At this time, the limited samples in the real world may not be able to achieve the rapid update of energy management strategies. How to narrow the gap between the data distribution in the simulation environment and the real world, and how to use big data and make reinforcement learning generate strategies consistent with the driving style of the real world need to be addressed.

Secondly, similar to the rule-based energy management strategy, the influence of parameter settings on the testing effect of reinforcement learning cannot be ignored. Due to these preset settings not being automatically tuned during training, developers need to make choices based on expertise. The results of the learning process and the required learning time largely depend on their selection. Xu et al. explored the influence of state setting, action setting, learning rate, and other parameters on the Q-learning training effect [66]. Lian et al. proved that the setting of the reward function in RL-EMS would impact the final SOC value and energy-saving effect [112]. Under the same data distribution and vehicle model, different parameters can affect the training effect, posing a problem for RL-EMS. How to select parameters appropriately based on the vehicle’s power model and driving style to maximize the energy-saving performance of the algorithm remains a problem.

4.3.1.2. Quantification of costs. RL-EMS research has different indicators for evaluating the effectiveness of the strategies used, which also leads to confusion in evaluation criteria: 1) In terms of economy, due to the global optimal effect of DP, most studies choose to compare with DP. In some literature, there is little difference in driving style between test and training conditions. Researchers
compare the fuel consumption of different algorithms under the same boundary conditions, and the energy-saving effect of the algorithms is quantified. However, this measurement standard needs more universality due to the different driving styles and time lengths. 2) In terms of cost calculation, the learning process of RL-EMS should be shorter, and the extensive interaction with the environment increases the offline computing time. For future researchers, improving the reinforcement learning algorithm to achieve rapid convergence is challenging. In addition, in terms of algorithm deployment, the complex mathematical and neural network structures increase the difficulty of deploying on hardware. The onboard computing hardware that can update agent online needs to meet real-time requirements, which further increases costs. With the development of Vehicle-to-everything technology, RL-EMS needs more efficient computing architecture and deployment solutions to reduce time and economic costs.

4.3. 2. Algorithm improvement

4.3.2.1. Security exploration. RL explores the global optimum by trial-and-error search. Therefore, it must be shaped by a large amount of data interacting with its environment, but safety issues during execution are not necessarily avoided during trial-and-error [135], especially when the physical constraints of the vehicle cannot be exceeded when the properties of the environment change. In this case, learners should consider the consequences of their actions and explore safely in the real physical world. For the energy management problem of MPS-EV, the safety research of reinforcement learning solutions can be carried out in the following directions.

- a) Action Pruning: The exploration of reinforcement learning is often based on the random selection of actions or sampling of policy distributions. Such actions are potentially dangerous in MPS-EV energy management problems, where excessive charging or discharging of the battery at low or high SOC states can easily reduce the battery life, and overcharging of the battery may even trigger abnormal battery operation [136]. Therefore, filtering or masking some of the control actions of the agent that may lead to dangerous states during training and deployment of reinforcement learning is an intuitive disaster avoidance method, which is often referred to as action pruning or action masking. Gao et al. have shown that using action pruning methods in high-hazard action space problems can be effective in avoiding training disasters to enhance the training effectiveness of strategies [137].

- b) Human Intervention: With the further development of V2X and MPS-EV configurations, states and action spaces in the future EMS may break through the current research architecture, resulting in that the current research and definitions of hazard exploration actions cannot be inherited and applied. The introduction of human decision-making experience in critical states, which is often considered safe and trustworthy, would help improve the safety exploration process of reinforcement learning on MPS-EV energy management problems. Saunders et al. have shown that direct human intervention could reduce safety exploration disasters in reinforcement learning [138]. How to introduce and develop human interventions to improve the exploration process of RL-EMS to enhance the training effect of reinforcement learning will become a research direction of interest.

- c) Recovery: In addition to the safety issues caused by stochastic exploration, the noise of state perception, unreasonable reward functions, and differences in deployment environments can lead to potential safety problems [139]. Assuming that security incidents cannot be avoided, how to quickly recover from catastrophic events will be the focus of future research. Yang et al. proposed a safe reinforcement learning framework that switches between a safe recovery strategy that prevents an agent from entering an unsafe state and a learner strategy that optimizes for task completion. Their experimental results showed that the safe recovery strategy effectively reduces the occurrence of safety accidents and improves the training effectiveness of reinforcement learning strategies [140]. Safe recovery strategies for reinforcement learning in MPS-EV energy management problems have not been investigated. The introduction of safe recovery strategies in the foreseeable future will greatly improve the safety of exploration of reinforcement learning in MPS-EV energy management problems.

4.3.2.2. Robustness improvement. To extend the application of the reinforcement learning algorithm to a broader scope of energy management for MPS-EV, the road environment and driving styles of each agent should be taken into consideration. Decisions made only for instantaneous speed and acceleration information cannot be extended to many different working conditions with different styles. In the past, RL-EMS handled a limited sample, which made it difficult to handle the diversity of real-world testing driving cycles. There are three research directions to be explored for the poor robustness of RL-EMS.

- a) Real-time generation of training data: Although reinforcement learning algorithms can adapt to different driving cycles by self-learning effects through experience gained from interaction with the environment, iteration of the strategy requires computational costs and some time to adapt when facing real driving cycles that differ significantly from the training driving cycles. Therefore, the training data are required to be close to the real world and have diversity. Some research carried out training on fixed
driving cycles, which lacks adaptability. To address this drawback, some researchers have enhanced the diversity of training data by introducing driving cycles with a large period collected from real roadways [141], generating real-time conditions based on simulation software to simulate different styles of road conditions for the vehicle model [117], adding noise to the original training data [36]. Markov chain model is used to generate driving cycles close to the latest historical driving data in real time for simulation [90]. There are gaps between the strategies generated based on simulation models and the real world, real-world working conditions data are more difficult to obtain, the virtual model cannot perfectly fit the actual power performance of the vehicle, and there are deficiencies in on-board computing system arithmetic power. These require the agent to generate sufficient experience to guide energy distribution based on diverse information. Research on improving RL-EMS robustness through real-time generated driving context information is emerging as a potential research hotspot.

b) Feature extraction for working conditions: RL agent makes decisions through state variables. The richness of training working conditions allows the agent to train on driving cycles with changing styles, but the state transfer probability matrix changes when the overall driving style is unknown and changes. It is difficult for an agent to plan close to the global optimum in multiple driving cycles based only on transient information such as speed and acceleration within a single time step. This requires the agent to extract the characteristics of entire working conditions and to sense the changes in overall working conditions in time. Guo et al. trained DDPG networks according to different speed intervals [80]; Lin et al. calculated the state transfer probability matrix in different time intervals during the driving time and measured the difference in their KL scatter as a basis for updating the strategy to achieve adaptability [122]. How to improve the sensing capability of RL-EMS agents and enhance the strategy robustness by introducing statistical and temporal features of the working conditions will be a potential future research direction.

c) Macroscopic traffic information sensing: With the development of V2X technology, vehicles can sense real-time traffic conditions, such as congestion and traffic light on surrounding and preceding roads. Weather conditions are also an important factor affecting driver behavior as well as fuel and power costs [142]. Much of the information provided by V2X can reflect the driving cycles the agent deals with to some extent and predict future travel information to inspire RL. Both speed prediction and lane density prediction based on macroscopic traffic information will contribute to the application of RL-EMS to the MPS-EV energy management problem.

d) EMS test platform: currently, academic researchers evaluating RL-EMS are limited to their own simulation platforms without a unified standard. This makes it difficult to make convincing comparisons and verify RL-EMS across various models and working conditions. To improve the robustness of RL-EMS research, a potential solution is to upload simulation models and classified working condition data on an open source platform like Github. This would allow for the creation of test procedures for urban and suburban working conditions, where investigators can showcase their innovation in parameter adjustment and training methods. By reducing duplication of similar work and testing the effectiveness of developed EMS more fairly, the robustness of RL-EMS studies can be improved.

4.3.2.3. Deployment of advanced distributed RL Algorithm. In general, research on related papers is usually small-scale, and the development of energy management strategies is often based on limited data. However, in real-life scenarios, data is often massive, which requires a complete distributed system to handle such large-scale data.

Based on the above summary of the challenges and potential research directions of reinforcement learning algorithm in application scenarios, we can introduce advanced distributed reinforcement learning algorithm as a promising solution. Firstly, massive interaction data can be collected during the sample collection process and shared among multiple vehicles to improve the diversity of training data and compensate for insufficient training samples. Large-scale distributed deployment reduces the risk of reinforcement learning during the exploration process and avoids unnecessary repeated interactions. This can significantly improve the efficiency and security of the algorithm. Additionally, by collecting and clustering dynamic state, trip information, and orientation of urban vehicles, traffic flow characteristics and driving styles can be defined. This further improves the perception scheme of the RL algorithm and provides a more solid basis for decision-making. Finally, during training, the server with strong computing power updates the strategy using collected samples. The vehicle only transmits information and coordinates with the central processing unit to change the gradient of the neural network, avoiding computation costs.

4.3.3. Deployment solution

The current development of AI tends to rely on big models and big data, and advanced AI solutions show significant high computing power demand characteristics. Therefore, future RL-EMS research will certainly face AI deployment challenges. Deployment on a single vehicle requires relevant deployment architecture tools. The following three main deployment tools are widely used for AI solutions: Open Visual Inference & Neural Network Optimization (OpenVino) for CPU devices; TensorRT for GPU devices; and MediaPipe, a deployment tool for edge devices.

TensorRT is a high-performance deep learning optimizer that provides low-latency, high-throughput deployments for deep learning applications. TensorRT can be used to compute acceleration for large-scale data centers, embedded platforms, or autopilot platforms. TensorRT is now capable of supporting TensorFlow, Caffe, Mxnet, Pytorch, and almost all other deep reinforcement learning frameworks. Combining TensorRT with high-performance GPUs for mobile devices enables fast and efficient deployment and computation in a wide range of industrial scenarios.

The OpenVINO tool suite is compatible with trained models from a variety of open source frameworks and has various capabilities for on-line deployment of algorithmic models, allowing pre-trained models to be quickly deployed on the CPU.

MediaPipe is a graphics-based cross-platform framework for building machine learning pipelines for multi-mode (video, audio, and
sensor) applications. MediaPipe can run across platforms on mobile devices, workstations, and servers, and supports mobile GPU acceleration. MediaPipe can not only be deployed on the server side but also serve as an on-device machine learning inference framework on multiple mobile terminals and embedded platforms.

At present, RL-EMS based on deep reinforcement learning has achieved economy close to that of DP. However, in the future real vehicle deployment, no breakthrough has been made in how to deploy this effective AI scheme to vehicles without damage. Therefore, the combination of RL-EMS and AI deployment framework will become the focus of future research.

5. Conclusion

This paper reviews the energy management strategies for multi-energy source vehicles based on reinforcement learning. The basics of reinforcement learning and multi-power source vehicles are introduced. The general process of RL-EMS is explained. An extensive review is focused on the application of various RL algorithms in the EMS field, and it can be seen that advanced RL algorithms can facilitate the performance of EMS. State sensing and control action schemes for RL-EMS are outlined. The setting of the reward function in RL-EMS is summarized. A description of the improvements to the RL-EMS training method due to vehicle characteristics and real driving cycles is presented. In this paper, we believe that one of the future research directions is to take advantage of the large-scale data utilization of offline distributed deep reinforcement learning to address the shortcomings of centralized reinforcement learning in the past. Accelerating the implementation of advanced algorithms with corresponding hardware and software for the real-time real-world deployment and enhancing the safety and robustness of the algorithms are necessary to realize the construction of energy management strategies in intelligent transportation environments.

Additional information

No additional information is available for this paper.

Data availability statement

Data availability is not applicable to this article as no new data were created in this study.

CRediT authorship contribution statement

Yang Lin: Writing – original draft, Formal analysis, Data curation. Jincheng Hu: Writing – review & editing, Writing – original draft, Visualization, Conceptualization. Zhuoran Hou: Resources, Investigation. Jihao Li: Resources, Investigation. Jingjing Jiang: Writing – review & editing, Supervision, Resources. Yuanjian Zhang: Writing – review & editing, Supervision, Resources, Conceptualization.

Declaration of competing interest

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

References

[1] F. Sun, Green Energy and Intelligent Transportation—promoting green and intelligent mobility, Green Energy and Intelligent Transportation 1 (2022), 1000017, https://doi.org/10.1016/j.giet.2022.1000017.
[2] A. Solouk, M. Shakiba-Herfeh, J. Arora, M. Shahbakhhti, Fuel consumption assessment of an electrified powertrain with a multi-mode high-efficiency engine in various levels of hybridization, Energy Convers. Manag. 155 (2018) 100–115, https://doi.org/10.1016/j.enconman.2017.10.073.
[3] A.H. Ganesh, B. Xu, A review of reinforcement learning based energy management systems for electrified powertrains: progress, challenge, and potential solution, Renew. Sustain. Energy Rev. 154 (2022), 111833, https://doi.org/10.1016/j.rser.2021.111833.
[4] 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.
[5] J. Peng, H. Fan, H. He, D. Pan, A rule-based energy management strategy for a plug-in hybrid school bus based on a controller area network bus, Energies 8 (2015) 5122–5142, https://doi.org/10.3390/en8065122.
[6] Xiaolai He, M. Parten, T. Maxwell, Energy management strategies for a hybrid electric vehicle, in: 2005 IEEE Vehicle Power and Propulsion Conference, IEEE, Chicago, IL, 2005, pp. 536–540, https://doi.org/10.1109/VPPC.2005.1554610.
[7] S.G. Li, S.M. Sharkh, F.C. Walsh, C.N. Zhang, Energy and battery management of a plug-in series hybrid electric vehicle using fuzzy logic, IEEE Trans. Veh. Technol. 60 (2011) 3571–3585, https://doi.org/10.1109/TVT.2011.2165571.
[8] J.-P. Gao, G.-M. Zhu, E.G. Strangas, F.-C. Sun, Equivalent fuel consumption optimal control of a series hybrid electric vehicle, Proc. Inst. Mech. Eng. - Part D J. Automob. Eng. 223 (2009) 1003–1018, https://doi.org/10.1243/09544070JAUTO1074.
[9] A. Panday, H.O. Bansal, A review of optimal energy management strategies for hybrid electric vehicle, Int. J. Veh. Technol. 2014 (2014) 1–19, https://doi.org/10.1155/2014/160510.
[10] O. Sundström, L. Guzzella, P. Soltic, Optimal hybridization in two parallel hybrid electric vehicles using dynamic programming, IFAC Proc. Vol. 41 (2008) 4642–4647, https://doi.org/10.3182/20080706-5-KR-1001.00781.
[11] Chan-Chiao Lin, Huei Peng, J.W. Grizzle, Jun-Mo Kang, Power management strategy for a parallel hybrid electric truck, IEEE Trans. Control Syst. Technol. 11 (2003) 839–849, https://doi.org/10.1109/TCST.2003.815606.
[12] Y. Li, H. He, J. Peng, H. Wang, Deep reinforcement learning-based energy management for a series hybrid electric vehicle enabled by history cumulative trip information, IEEE Trans. Veh. Technol. 68 (2019) 7416–7430, https://doi.org/10.1109/TVT.2019.2926472.
X. Tang, H. Zhou, F. Wang, W. Wang, X. Lin, Longevity-conscious energy management strategy of fuel cell hybrid electric Vehicle Based on deep reinforcement learning, IEEE Trans. Energy Convers. 2021 (2021), 113388, https://doi.org/10.1109/TEC.2021.3094705.

J. Yuan, L. Yang, Q. Chen, Intelligent energy management strategy based on hierarchical approximate global optimization for plug-in fuel cell hybrid electric vehicles, Int. J. Hydrogen Energy 45 (2020) 18754-18770, https://doi.org/10.1016/j.ijhydene.2020.06.082.

Y. Hu, W. Li, K. Xu, T. Zahid, F. Qin, C. Li, Energy management strategy for a hybrid electric tracked vehicle, Energies 13 (2020) 4555, https://doi.org/10.3390/en13204555.

S. Yue, Y. Wang, Q. Xie, D. Zhu, M. Pedram, N. Chang, Model-free learning-based online management of hybrid electrical energy storage systems in electric vehicles, in: ICON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society, Dallas, TX, USA, 2014, pp. 3142-3148, https://doi.org/10.1109/IEEE.2014.7048959.

X. Lin, P. Bogdan, N. Chang, M. Pedram, Machine learning-based energy management in a hybrid electric vehicle to minimize total operating cost, in: 2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), IEEE, Austin, TX, USA, 2015, pp. 627-634, https://doi.org/10.1109/IEEE.2015.7412726.
C. Apps, D. Silver, Grandmaster level in StarCraft II using multi-agent reinforcement learning, Nature 575 (2019) 350-354, https://doi.org/10.1038/s41586-019-1724-z.

B. Xu, J. Hou, J. Shi, H. Li, D. Rathod, Z. Wang, Z. Filippi, Learning time reduction using warm-start methods for a reinforcement learning-based supervisory control in electric vehicle applications, IEEE Trans. Veh. Technol. 69 (2020) 626-635, https://doi.org/10.1109/TVT.2020.3019009.

Y. Zhang, R. Ma, D. Zhao, Y. Huangfu, W. Liu, A novel energy management strategy based on dual reward function Q-learning for fuel cell hybrid electric vehicle, IEEE Trans. Ind. Electron. 69 (2022) 1537-1547, https://doi.org/10.1109/TIE.2021.3062273.

L. Zou, L. Xia, P. Du, Z. Zhang, T. Bai, W. Liu, J.-Y. Nie, D. Yin, Pseudo dyna-Q: a reinforcement learning framework for interactive recommendation, in: Proceedings of the 13th International Conference on Web Search and Data Mining, ACM, Houston TX USA, 2020, pp. 816-824, https://doi.org/10.1145/3375267.3382279.

A. Khalatbarisoltani, M. Kandidayeni, L. Boulon, X. Hu, Power allocation strategy based on decentralized convex optimization in modular fuel cell systems for vehicular applications, IEEE Trans. Veh. Technol. 69 (2020) 14563-14574, https://doi.org/10.1109/TVT.2020.3028089.

A. Khalatbarisoltani, M. Kandidayeni, L. Boulon, X. Hu, Comparison of decentralized ADMM optimization algorithms for power allocation in modular fuel cell vehicles, IEEE ASME Trans. Mechatron. 27 (2022) 3297-3308, https://doi.org/10.1109/TMECH.2021.3105950.